Modeling Mutual Influence in Multi-Agent Reinforcement Learning

Ying Wen

A dissertation submitted in partial fulfillment

of the requirements for the degree of

Doctor of Philosophy

of

University College London.

Department of Computer Science

University College London

August 26, 2020

Abstract

In multi-agent systems (MAS), agents rarely act in isolation but tend to achieve their goals through interactions with other agents. To be able to achieve their ultimate goals, individual agents should actively evaluate the impacts on themselves of other agents' behaviors before they decide which actions to take. The impacts are reciprocal, and it is of great interest to model the mutual influence of agent's impacts with one another when they are observing the environment or taking actions in the environment. In this thesis, assuming that the agents are aware of each other's existence and their potential impact on themselves, I develop novel multi-agent reinforcement learning (MARL) methods that can measure the mutual influence between agents to shape learning. The first part of this thesis outlines the framework of recursive reasoning in deep multi-agent reinforcement learning. I hypothesize that it is beneficial for each agent to consider how other agents react to their behavior. I start from Probabilistic Recursive Reasoning (PR2) using level-1 reasoning and adopt variational Bayes methods to approximate the opponents' conditional policies. Each agent shapes the individual Q-value by marginalizing the conditional policies in the joint Q-value and finding the best response to improving their policies. I further extend PR2 to Generalized Recursive Reasoning (GR2) with different hierarchical levels of rationality. GR2 enables agents to possess various levels of thinking ability, thereby allowing higher-level agents to best respond to less sophisticated learners. The first part of the thesis shows that eliminating the joint Q-value to an individual Q-value via explicitly recursive reasoning would benefit the learning. In the second part of the thesis, in reverse, I measure the mutual influence by approximating the joint Q-value based on the individual Q-values. I establish Q-DPP, an extension of the Determinantal Point Process (DPP) with partition constraints, and apply it to multi-agent learning as a function approximator for the centralized value function. An attractive property of using Q-DPP is that when it reaches the optimum value, it can offer a natural factorization of the centralized value function, representing both quality (maximizing reward) and diversity (different behaviors). In the third part of the thesis, I depart from the action-level mutual influence and build a policy-space meta-game to analyze agents' relationship between adaptive policies. I present a Multi-Agent Trust Region Learning (MATRL) algorithm that augments single-agent trust region policy optimization with a weak stable fixed point approximated by the policy-space meta-game. The algorithm aims to find a game-theoretic mechanism to adjust the policy optimization steps that force the learning of all agents toward the stable point.

Impact Statement

This thesis deepens the understanding of the mutual influence in multi-agent reinforcement learning and the proposed algorithms to model this influence to help multi-agent learning, which has the potential to impact fields both inside and outside academia. In the research community, this thesis draws attention to the potential benefits of recursive reasoning in multi-agent learning, which may provide new insights for modeling the bounded rationality in human-machine decision-making problems. Furthermore, this thesis presents a new function approximator that stimulates agents' behavior diversity in discrete tasks. This diversity is crucial in fields such as gaming and self-driving cars, and there are potential extensions based on proposed methods that make its application possible in more complicated scenarios. Furthermore, the multi-agent trust region learning approach is proved to be helpful to stabilize and boost independent learners' training. In industry, this work also has many potential applications. For example, robotics equipped with a recursive reasoning ability can assist humans in more user-friendly ways. In addition, many real-world applications require fidelity simulations to train agents. For example, generating realistic traffic flow is an essential step in self-driving car simulation. The existing methods used to generate traffic flow usually follow some rules that are not really "intelligent" or realistic. Therefore, the Q-DPP model can be used to generate diverse behaviors to build a more realistic simulation environment. Furthermore, multi-agent trust region learning is a promising approach to stabilize and speed up many existing multi-agent reinforcement learning algorithms. MATRL brings a stable and efficient performance improvement without too much extra cost in multi-agent learning. MATRL can significantly help to reduce (by at least 90%) the training cost of many successful multi-agent applications.

Acknowledgements

I would like to take this opportunity to express my deepest gratitudes to all the people who have been tremendous supportive during my Ph.D. study.

First of all, I'm enormously thankful for the mentoring and guidance of my primary supervisor, Prof. Jun Wang. It is not only his expertise and those constructive suggestions, which I benefited extremely, but also his charming character of being kind and patient, have made this experience so precious.

I'd also thank Dr. Jinghao Xue, Prof. Emine Yilmaz, and Dr. Shi Zhou for all the help offered to my work. Besides, I thank Dr. Yuanchang Liu and Dr. Stefano V. Albrecht for carefully examining this thesis and the valuable feedbacks. It could not be in this better shape without their involvement.

I also enjoyed a great collaboration with Dr. Weinan Zhang, Dr. Wei Pan, and Dr. Haitham Bou Ammar with their preciseness suggestions, which are essentially valuable to this work.

My appreciation extends to all members of this team: Rui Luo, Yixin Wu, Hui Chen, Dr. Haifeng Zhang, Dr. Shuang Zhao, Dr. Xu Chen, and Dr. Yali Du. In particular, Yaodong Yang, Zheng Tian, and Minne Li helped me a lot in initiating the manuscript to achieve a good shape of results. It's a great honor to spend my Ph.D. years in this fantastic teamn and I learned a lot from them.

Furthermore, my gratitude goes to Mr. Konrad Feldman and Mrs. Jennie Feldman for offering me with the Feldman Scholarship in Statistical Machine Learning and colleagues Dr. Shuai Yuan and Rael Cline in MediaGamma Ltd. for jointly sponsored me three years of my Ph.D. study.

Plus, my sincere thanks go to all the people who were so friendly and made my stay in London a pleasant memory, especially to Qiyang Zhang, Guanyu Tao, Dr. Chen Zhang, Tianrui Zhao, Dr. Wenyao Li, Bowen Zheng, Huanyu Ma, Xinyu Weng, and Wenqing Wang.

More importantly, I would like to thank my family members with my deepest sincerity, especially my mother Yueying Hu, my father Yousheng Wen and girlfriend Yi Li. It is their unconditional support and love that give me confidence and courage to fully devoted to this academic path. It goes without saying that family is always my inner strength source.

To sum up, this thesis is a result of many people's contributions, direct or indirect. It is however difficult to name them all. I appreciate all those helps.

Contents

List of Figures

List of Tables

List of Abbreviations

AC	Actor-Critic.
AI	Artificial Intelligence.
BR	Best Response.
CHM	Cognitive Hierarchy Model.
CL	Centralized Learning.
CMA-ES	Covariance Matrix Adaptation Evolution Strategy.
COMA	Counterfactual Multi-Agent Policy Gradients.
CTDE	Centralized Training and Decentralized Execution.
DDPG	Deep Deterministic Policy Gradient.
DPP	Determinantal Point Process.
DRL	Deep Reinforcement Learning.
EGTA	Empirical Game-Theoretic Analysis.
GAN	Generative Adversarial Network.
GR2	Generalized Recursive Reasoning.
IGA	Infinitesimal Gradient Ascent.
IGA-PP	Infinitesimal Gradient Ascent with Policy Prediction.
IL	Independent Learner.
I-POMDP	Interactive Partially Observable Markov Decision Process.
KL-Divergence	Kullback–Leibler Divergence.
LOLA	Learning with Opponent-Learning Awareness.
MADDPG	Multi-Agent Deep Deterministic Policy Gradient.
MAL	Multi-Agent Learning.
MARL	Multi-Agent Reinforcement Learning.

MAS	Multi-Agent Systems.
MASQL	Multi-Agent Soft Q-Learning.
MATRL	Multi-Agent Trust Region Learning.
MAVEN	Multi-Agent Variational Exploration.
MCMC	Markov Chain Monte Carlo.
MDP	Markov Decision Process.
MF	Mean Field.
ML	Machine Learning.
MLP	Multi-Layer Perceptron.
MP	Matching Pennies.
NE	Nash Equilibrium.
NN	Neural Networks.
OM	Opponent-Modelling.
PBE	Perfect Bayesian Equilibrium.
POMDP	Partially Observable Markov Decision Process.
PR2	Probabilisitic Recursive Reasoning.
PD	Prisoner’s Dilemma.
P-DPP	Partition DPP.
PG	Policy Gradient.
PSRO	Policy Space Response Oracles.
PSD	Positive Semidefinite.
PPO	Proximal Policy Optimization.
Q-DPP	DPP for Joint Q-Value Decomposition.
QMIX	Mixing Networks for Joint Q-Value.
QTRAN	Transformation for Joint Q-Value.
RL	Reinforcement Learning.
RNN	Recurrent Neural Network.
SG	Stochastic Game.
SH	Stag Hunt.
SVGD	Stein Variational Gradient Descent.

TBR	Trust Stable Region.
TPR	Trust Payoff Region.
TRPO	Trust Region Policy Optimization.
ToM	Theory of Mind.
VDN	Value Decomposition Networks.
WoLF	Win or Learn Fast.

Chapter 1

Introduction

In the long journey of building artificial intelligence (AI) that can learn and think like humans, an AI agent's hallmark is its capability to think, understand, and interact with the world. Machine learning (ML) is one of the tools available to achieve this target, which extracts knowledge from data (Shalev-Shwartz and Ben-David, 2014). The most well-known ML branches are supervised learning and unsupervised learning, in which the input is usually static training data (e.g., images) and the aim is to recognize some patterns or knowledge (e.g., class). In the last decade, taking advantage of the powerful representation learning ability of deep learning (DL) (Goodfellow et al., 2016; LeCun et al., 2015), real-world AI applications have achieved many breakthroughs among domains such as image classification (Krizhevsky et al., 2012) and natural language processing (Devlin et al., 2019). In addition to pattern recognition, modern AI applications often need to interact with the dynamic world and make decisions based on learned knowledge. Therefore, with the success of pattern recognition, much research shifts from pattern recognition to decision-making/control, which is known as reinforcement learning (RL) (Sutton and Barto, 2018). RL is also a branch of ML, in which an agent can take actions, perceive states and receive rewards from the environment. The agent's target is learning how to behave optimally based on a trial and error procedure during its interaction with the environment. Historically, RL has many classic outcomes on dynamic programming (DP) (Bellman, 1952), temporal difference (TD) (Klopf, 1972), and policy gradient (PG) (Sutton et al., 2000; Williams, 1992). However, due to the curse of dimensionality, these methods hardly deal with real-world applications. Nevertheless, in recent years, Mnih et al. (2013) made a significant breakthrough by introducing deep Q-learning (DQN), which demonstrates human-level performance on 49 Atari games. Since then, many practical algorithms (e.g., DQN (Mnih et al., 2013), DDPG (Lillicrap et al., 2015), PPO (Schulman et al., 2017)), and successful applications (e.g., AlphaGo series) have been developed and mark the maturity of single-agent decision-making's milestones. However, the real-world contains environmental variations, such as other agents (Lake et al., 2017) or even humans. Therefore, agents should be aware that other agents in a shared environment are different from the other parts of the environment.

1.1 Motivation

At the cognitive level, real-world intelligent entities, such as rats or chimpanzees, are born able to understand or infer the interests of the other agents during their interactions (Pfeiffer and Foster, 2013; Tolman, 1948). Such interests are usually referred to as a high-level unobservable mental state, including desires, beliefs, and intentions (Gopnik and Wellman, 1992; Premack and Woodruff, 1978). Therefore, in daily life, humans always rely on this understanding to predict strangers' future behaviors (Gordon, 1986), planning effective interactions (Gallese and Goldman, 1998) or match one's own belief with the commonsense, which is called folk psychology (Dennett, 1991). Correspondingly, it is expected that AI agents that also possess this kind of ability will be developed by building predictive models to assess the influence of interactions with others (Albrecht and Stone, 2018). Such a social ability is a key factor that defines intelligence—the capability to interact with other agents or humans and be aware or their actions or power to producing effects on them. Intuitively knowing and learning these effects will benefit the agent's decision-making process in complex multi-agent environments. For example, by knowing another agent's future actions and goals, an agent can plan and interact more effectively with a well-prepared strategy (Gmytrasiewicz and Doshi, 2005).

To enable these higher-level reasoning abilities in real-world interactions, an extensive range of tools have emerged from human activities to make daily interactions smooth and efficient (Coricelli and Nagel, 2009). These tools include concepts such as beliefs (Albrecht and Stone, 2018), reciprocity (Eccles et al., 2019), and empathy (Raileanu et al., 2018). Many real-world challenges require an agent to possess such an ability to consider effects from the outside world (Foerster et al., 2016; Ziebart, 2010), which involve environments that contain many learning agents and are thus multi-agent in nature. Examples include self-driving cars (Cao et al., 2012), multi-player games (Berner et al., 2019; Vinyals et al., 2019), robotics (Kitano et al., 1997), supply chains (Pardoe and Stone, 2004) and so forth. In these scenarios, some distributed agents need to make independent decisions based on observations to contribute most effectively to an overall goal or to maximize individual rewards considering the presence of other agents in the environments (Bowling and Veloso, 2004). To this end, the need for a decision-making framework in multi-agent scenarios, together with the complexity of dealing with multiple interacting learners, leads to the development of multi-agent reinforcement learning (MARL).

In MARL, unlike single-agent cases, due to the existence of agents' interactions, the result of these actions is not standalone but usually depends on others' choices (Hernandez-Leal et al., 2017). More specifically, to illustrate the effects of multi-agent interactions, let us consider a motivating example of controlling an adaptive robot car to go through an intersection. At each time step, the robot car can move around by steering, throttling and/or braking. The goal is to pass the intersection safely and reach the destination. In addition to detecting environments, such as positions, traffic lights, and lane markings, the robot car also needs to be aware of other cars, which can be human cars or other adaptive cars. I aim to develop a policy that can control a robot car via a sequence of actions to achieve the goal. This difference causes two additional challenges compared to single-agent reinforcement learning. First, before making a decision, the robot car needs to consider the potential plan of other cars. For example, human drivers usually project other cars' movements in advance and then take strategic action (e.g., either give way to a rushing car or speed up to merge into another lane rapidly). Meanwhile, other drivers follow the same pattern, which implies the car must consider other cars' behaviors and act correspondingly to drive smoothly. Second, as there are multiple adaptive agents in a shared environment, they improve their policies simultaneously. Therefore, from a single agent's perspective, the environment it perceives is non-stationary. Dealing with non-stationary movement is one of the most critical issues in MARL research and makes it problematic to apply single-agent RL directly in the multi-agent context.

Both challenges involve the influence from others in an environment, so I call it mutual influence in multi-agent reinforcement learning. From the above example, it can be seen that agents can benefit mutually by considering the influence of others. In addition, in principle, it is possible for agents to observe other agents and to take actions that directly or indirectly affect other agents (Tuyls and Stone, 2018) to achieve the learning objectives. Therefore, multi-agent learners are able to figure out a way to reach the goals that require mutual effects during learning.

1.2 Contributions and Outline

The goal of this thesis is to consider mutual influence in learning. I am especially interested in modeling mutual influence to shape agents' learning processes. With this motivation in mind, this thesis answers three research questions regarding the mutual influence problem from multiple perspectives, as follows:

- *From an individual agent's view, how can the influence from the other agents' actions on its objective be estimated?*
- *From the collective point of view, how can each individual's contributions be managed to achieve the cooperative goals in tasks?*
- *Instead of focusing on action space influence, how can the policy space influence be controlled to stabilize multi-agent learning and improve convergence performance at the same time?*

To address the above questions, I develop novel MARL algorithms that allow agents to manage the influence of other agents. In the following section, I present the outline of this thesis to highlight the core of each part.

Part I: Explicit Mutual Influence Models

The most straightforward approach to model mutual influence is to consider the action effects on others. In this section, I explicitly approximate the opponent models, precisely those involving conditional behavioral policies, which can help eliminate the effects of the others on the joint Q-value to obtain an individual Q-value.

In **Chapter 3**, I seek to introduce a Probabilistic Recursive Reasoning (PR2) framework for multi-agent RL tasks. I believe it is beneficial for each agent to consider how opponents react to potential behaviors for their own decision making, which is essentially recursive reasoning. Under the PR2 framework, I discover a term of importance-sampling weight that can be used to quantify the difference between the independent-learning and centralized-learning methods. By employing variational Bayes methods (Kingma and Welling, 2014) to model the uncertainty of opponents' conditional policies, the results justify the unique value provided by endowing the agents with learning skills to recursively consider their self-impact on others.

In **Chapter 4**, I extend the PR2 to *Generalized Recursive Reasoning (GR2)*, which acknowledges the bounded rationality and requires no assumption that agents must play the Nash strategy for all stage games encountered Following the cognitive hierarchy theory (Camerer et al., 2004), GR2 develops a recursive model of reasoning by steps. In GR2, agent types are drawn from a hierarchy of reasoning capability with an arbitrary thinking depth. GR2 begins with level-0 (L0 for short) type agents who do not assume anything about the other agents. Level-1 thinkers believe all their opponents are in level-0, and level-2 thinkers believe all opponents are in either level-0 or level-1; they consider these beliefs before making the best decision. GR2 agents are all self-interested in the sense of best responding to their beliefs of others, i.e., level-k agents take the best response to either level-$k-1$ thinkers or a mixture of agents with the level ranging from 0 to $k-1$. Due to the recognition of agents' bounded rationality, the GR2 MARL algorithm is adept at capturing non-equilibrium behaviors that inform others of the correct expectations, and ultimately, the algorithm helps the convergence to reach equilibrium. I prove that when the level of reasoning is deep, GR2 methods can converge to at least one NE. Additionally, if the lower-level agent plays the Nash strategy, then all the higher-level agents will follow.

Part II: Behavioral Diversity in Mutual Influence

Instead of eliminating the effects from other agents in the joint Q-value, in **Chapter 5**, I try to manage the agents' influence on the joint Q-value and propose multi-agent determinantal Q-learning.the I establish *Q-DPP*, an extension of determinantal point process (DPP) (Kulesza et al., 2012) with a partition-matroid constraint to the multi-agent setting. Q-DPP promotes agents to acquire diverse behavioral models, which allows for natural factorization of the joint Q-functions with no need for *a priori* structural constraints on the value function or network architectures. I demonstrate that Q-DPP generalizes major solutions, including VDN (Sunehag et al., 2018) , QMIX (Rashid et al., 2018), and QTRAN (Son et al., 2019), on decentralizable cooperative tasks. To efficiently draw samples from Q-DPP, I adopt a sample-by-projection sampler with a theoretical approximation guarantee. The sampler also benefits exploration by coordinating agents to cover orthogonal directions in the state space during multi-agent training.

Part III: Game-Theoretic Analysis of Policy-Space Influence

In **Chapter 6**, I focus on analyzing the policy-space level mutual influence and try to find a trust region method to shape this influence for multi-agent policy optimization. Trust region policy optimization is widely used in single-agent reinforcement learning, where a trust region at each iteration provides a lower bound of the monotonic payoff improvement for policy optimization. However, in multi-agent systems, as the agent's payoff improvement depends on the other agents' adaptive behaviors, it is hard to measure the payoff improvement when all agents are learning simultaneously. Although researchers have proposed other solution concepts, such as fixed points (e.g., the Nash equilibrium), they are usually intractable. Therefore, directly exploiting the trust region improvement from a single agent's perspective is impossible because the agents' selfish improvement would rarely lead to a stable multi-agent solution concept In this chapter, I present a *Multi-Agent Trust Region Learning (MATRL)* algorithm that augments the single-agent trust region policy optimization with a policy-space meta-game analysis A weak stable fixed point approximated by the Nash equilibrium of a policy-space meta-game largely simplifies the complex game solution concept computation. I also find the lower bound for multi-agent trust region learning and prove the finding of the weak stable fixed point via meta-game equilibrium.

These chapters are based on the following papers and pre-prints, where I also clarify (indicated by author initials) the individual contributions:

- ‘Probabilistic Recursive Reasoning for Multi-Agent Reinforcement Learning’, Ying Wen, Yaodong Yang, Rui Luo, Jun Wang, Wei Pan, *International Conference on Learning Representations (ICLR)*, 2019. **Authors’ contributions**: The correlated policy idea was initially from J.W, and I specified the idea into a recursive reasoning framework. I developed all the methodology parts and part of theoretical proofs. I wrote the initial draft and re-edited the paper. I also conceived/performed all the experiments. Y.Y.; reviewed/re-edited the paper and provided the proof for a theorem. R.L., W.P.; reviewed the paper. J.W.; supervised the research.

- ‘Modeling Bounded Rationality in Multi-Agent Interactions by Generalized Recursive Reasoning’, Ying Wen, Yaodong Yang, Jun Wang, *International Joint Conferences on Artificial Intelligence (IJCAI)*, 2020. **Authors’ contributions**: The paper idea was initially from me. I developed all the methodology parts and part of theoretical proofs. I wrote the initial draft and re-edited the paper. I also conceived/performed all the experiments. Y.Y.; reviewed/re-edited the paper and provided the proof for a corollary. J.W. supervised the research.

- ‘Multi-Agent Determinantal Q-Learning’, Yaodong Yang*, Ying Wen*, Liheng Chen, Jun Wang, Kun Shao, David Mguni, Weinan Zhang, *International Conference on Machine Learning (ICML)*, 2020. **Authors’ contributions**: The paper idea, applying the DPP in MAL, was initially from Y.Y. I developed all the methodology parts and wrote the initial draft. I also wrote the experiment code and conceived/performed part of the experiments. Y.Y.;

reviewed/re-edited the paper and provided all the proof. K.S; L.C.; performed part of the experiments for discrete and continuous games, respectively. D.M.; W.Z; reviewed the paper. J.W. supervised the research.

- ‘Multi-Agent Trust Region Learning’, Ying Wen, Hui Chen, Yaodong Yang, Zheng Tian, Minne Li, Xu Chen, Jun Wang, 2020. **Authors’ contributions**: The paper idea was initially from me. I developed all the methodology parts and theoretical proofs. Besides, I wrote the initial draft and re-edited the paper. I also conceived the experiments and performed part of the experiments. H.C.; wrote the experiment code and performed the rest experiments. Y.Y., Z.T., M.L., X.C.; reviewed the paper. J.W. supervised the research.

To supplement this work, Chapter 2 presents the necessary fundamentals to understand all of the notations and background of this thesis. Chapters 3 through 6 present the contributions as listed above, in which I use pronoun ‘we’ because they are joint work. Lastly, Chapter 7 summarizes the thesis and discusses potential future research directions.

Chapter 2

Background

This chapter includes the necessary background and formulation on single-agent and multi-agent reinforcement learning. In particular I introduce single-agent reinforcement learning in Section 2.1, multi-agent formulations in Section 2.2, agent modeling in Section 2.4, value function decomposition in Section 2.3.1, and empirical game-theoretic analysis in Section 2.4. The concepts required for specific chapters are introduced as additional preliminaries in those chapters.

2.1 Single-Agent Reinforcement Learning

Single-agent Reinforcement Learning (RL) is used to study a sequential decision-making problem, in which an agent interacts with an environment This problem is usually formulated as a Markov Decision Process (MDP) (Sutton and Barto, 2018), denoted by a tuple $< \mathcal{S}, \mathcal{A}, \mathcal{R}, P, p_0, \gamma >$, where $\mathcal{S}$ is the state space, $\mathcal{A}$ is the action space, $\mathcal{R} = R(s, a) : \mathcal{S} \times \mathcal{A} \to \mathbb{R}$ is the reward function, $P : \mathcal{S} \times \mathcal{A} \times \mathcal{S} \to [0, 1]$ is the transition function, p_0 is the initial state distribution and γ is the discount factor. At time step t, given the state $s_t \in \mathcal{S}$, an agent takes an action a_t according the policy $\pi(a_t|s_t) : \mathcal{S} \times \mathcal{A} \to [0, 1]$ and observe the reward r_t from the environment. The environment then transits to the next state $s_{t+1} \sim P(s_{t+1}|s_t, a_t)$. Consider an infinite-horizon case; the agent aims to find a policy that maximizes the long-term discounted reward:

$$\eta(\pi) = \mathbb{E}_{s_0,a_0,\cdots} \left[\sum_{t=0}^{\infty} \gamma^t R\left(s_t, a_t\right) \right], \tag{2.1}$$

where $s_0 \sim p_0(s_0)$, $a_t \sim \pi(a_t|s_t)$ and $s_{t+1} \sim P(s_{t+1}|s_t, a_t)$. Similarly, the state value function and state-action value function can be defined accordingly:

$$V^\pi(s_t) = \mathbb{E}_{a_t, s_{t+1}, \cdots} \left[\sum_{l=0}^{\infty} \gamma^l R_{t+l} \right], \tag{2.2}$$

$$Q^\pi(s_t, a_t) = \mathbb{E}_{s_{t+1}, a_{t+1}, \cdots} \left[\sum_{l=0}^{\infty} \gamma^l R_{t+l} \right]. \tag{2.3}$$

In the model-free setting, where the environment is unknown, value-based (e.g., DQN (Mnih et al., 2013)) and policy-based methods (e.g., PG (Sutton et al., 2000), TRPO (Schulman et al., 2015) and PPO (Schulman et al., 2017)), are two main approaches to solve this problem. These methods can be easily extended to the multi-agent case known as *Independent Learner* (IL) (Tan, 1993), which ignores other agents in an environment.

2.2 Multi-Agent Formulation

2.2.1 Stochastic Game

A stochastic game (Littman, 1994; Shapley, 1953) can be defined as: $\mathcal{G} = \langle N, \mathcal{S}, \{\mathcal{A}_i\}, \{\mathcal{R}_i\}, P, p_0, \gamma \rangle$, where N is the agent number and $\mathcal{S}$ denotes the state space. $\mathcal{A}_i$ is the action space for agent i. $\mathcal{A} = \mathcal{A}_i \times \mathcal{A}_{-i}$ is the joint action space, where $-i$ denotes the other agent except agent i. $\mathcal{R}_i = R_i(s, a_i, a_{-i})$ is the reward function for agent i. $P : \mathcal{S} \times \mathcal{A} \times \mathcal{S} \to [0, 1]$ is the transition function. p_0 is the initial state distribution, and $\gamma \in [0, 1]$ is a discount factor. Each agent has a stochastic policy $\pi_i(a_i|s) : \mathcal{S} \times \mathcal{A}_i \to [0, 1]$. Each agent $i \in 1, \cdots, N$ aims to maximize its long term discounted reward:

$$\eta_i(\pi_i, \pi_{-i}) = \mathbb{E}_{s^0, a_i^0, a_{-i}^0 \cdots} \left[\sum_{t=0}^{\infty} \gamma^t R_i(s^t, a_i^t, a_{-i}^t) \right], \tag{2.4}$$

where $s^0 \sim p_0$, $s^{t+1} \sim P(s^{t+1}|s^t, a_i^t, a_{-i}^t)$, $a_i^t \sim \pi_i(a_i^t|\tau_i^t)$. Given the state and joint action, the standard definition of the state-action value function is as follows

$$V_i^{\pi_i,\pi_{-i}}(s^t) = \mathbb{E}_{s^{t+1},a_i^{t+1},a_{-i}^{t+1}\dots}[\sum_{l=0}^{\infty}\gamma^l R_i(s^{t+l}, a_i^{t+l}, a_{-i}^{t+l})], \tag{2.5}$$

the value function:

$$Q_i^{\pi_i,\pi_{-i}}(s^t, a_i^t, a_{-i}^t) = \mathbb{E}_{s^{t+1},a_i^{t+1},a_{-i}^{t+1}\dots}[\sum_{l=0}^{\infty}\gamma^l R_i(a_i^t, a_{-i}^t), s^{t+1}], \tag{2.6}$$

and the advantage function, with the lower variance compared to the state-action value function by taking the state-value off as the baseline:

$$A_i^{\pi_i,\pi_{-i}}(s^t, a_i^t, a_{-i}^t) = Q_i^{\pi_i,\pi_{-i}}(s^t, a_i^t, a_{-i}^t) - V_i^{\pi_i,\pi_{-i}}(s^t). \tag{2.7}$$

However, the agents' objectives are not always aligned, especially in non-cooperative games As a result, Nash equilibrium (Chatterjee et al., 2004; Nash et al., 1950) has been a popular solution for the multi-agent learning problem. In a Nash equilibrium, no agent can increase its expected payoff by changing its equilibrium strategy unilaterally (Nash et al., 1950). Nash equilibrium can be defined as follows:

Definition 2.1 (Nash Equilibrium and ϵ-Nash Equilibrium)**.** *For $\epsilon \geq 0$, a joint policy (π_i^*, π_{-i}^*) is a ϵ-Nash equilibrium if: $\forall i \in \{1, 2\}$, $\forall \pi_i \in \Pi_i$:$\eta_i(\pi_i^*, \pi_{-i}^*) \geq \eta_i(\pi_i, \pi_{-i}^*) - \epsilon$. When $\epsilon = 0$, the equilibrium is a Nash equilibrium.*

Basically, equilibrium is reached when each agent provides its best response to other agents' policies. Hence, no agent can have more benefits by changing policies given that all the other agents continue to follow the equilibrium policy.

2.2.2 Dec-POMDP

In Chapter 5, to investigate the mutual influence in the joint Q value, I adopt the common decentralized POMDP (Oliehoek et al., 2016) setting in joint value decomposition methods. Dec-POMDP models a multi-agent cooperation task in a partially-observed environment and is denoted by a tuple $\mathcal{G} =< \mathcal{S}, \mathcal{N}, \mathcal{A}, \mathcal{O}, \mathcal{Z}, P, \mathcal{R}, \gamma >$.

Within $\mathcal{G}$, $s \in \mathcal{S}$ denotes the global environmental state. At every time-step $t \in \mathbb{Z}^+$, each agent $i \in \mathcal{N} = \{1, \dots, N\}$ selects an action $a_i \in \mathcal{A}$ where a joint action stands for $\boldsymbol{a} := (a_i)_{i \in \mathcal{N}} \in \mathcal{A}^N$. Since the environment is partially observed, each agent only has access to its local observation $o \in \mathcal{O}$, which is acquired through an observation function $\mathcal{Z}(s, a) : \mathcal{S} \times \mathcal{A} \to \mathcal{O}$. The state transition dynamics are determined by $P(s'|s, \boldsymbol{a}) := \mathcal{S} \times \mathcal{A}^N \times \mathcal{S} \to [0, 1]$. Agents optimize towards one shared goal whose performance is measured by $\mathcal{R}(s, \boldsymbol{a}) : \mathcal{S} \times \mathcal{A}^N \to \mathbb{R}$, and $\gamma \in [0, 1)$ discounts the future rewards. Each agent recalls an observation-action history $\tau_i \in \mathcal{T} := (\mathcal{O} \times \mathcal{A})^*$, and executes a stochastic policy $\pi_i(a_i|\tau_i) : \mathcal{T} \times \mathcal{A} \to [0, 1]$ which is conditioned on τ_i. All of the agents' histories are defined $\boldsymbol{\tau} := (\tau_i)_{i \in \mathcal{N}} \in \mathcal{T}^N$. Given a joint policy $\boldsymbol{\pi} := (\pi_i)_{i \in \mathcal{N}}$, the joint action-value function at time t stands as $Q^{\boldsymbol{\pi}}(\boldsymbol{\tau}^t, \boldsymbol{a}^t) = \mathbb{E}_{s^{t+1:\infty}, \boldsymbol{a}^{t+1:\infty}}[G^t|\boldsymbol{\tau}^t, \boldsymbol{a}^t]$, where $G^t = \sum_{i=0}^{\infty} \gamma^i \mathcal{R}^{t+i}$ is the total accumulative rewards.

The goal is to find an optimal value function $Q^* = \max_{\boldsymbol{\pi}} Q^{\boldsymbol{\pi}}(\boldsymbol{\tau}^t, \boldsymbol{a}^t)$ and the corresponding policy $\boldsymbol{\pi}^*$. A direct centralized approach is to learn the joint value function, parameterized by θ, by minimizing the squared temporal-difference error $\mathcal{L}(\theta)$ (Watkins and Dayan, 1992) from a sampled mini-batch of transition data $\{\langle \boldsymbol{\tau}, \boldsymbol{a}, \mathcal{R}, \boldsymbol{\tau}' \rangle\}_{j=1}^{E}$, i.e.,

$$\mathcal{L}(\theta) = \sum_{j=1}^{E} \left\| \mathcal{R} + \gamma \max_{\boldsymbol{a}'} Q(\boldsymbol{\tau}', \boldsymbol{a}'; \theta^-) - Q^{\boldsymbol{\pi}}(\boldsymbol{\tau}, \boldsymbol{a}; \theta) \right\|^2, \tag{2.8}$$

where θ^- denotes the target parameters that can be periodically copied from θ during training.

2.3 Deep (Multi-Agent) Reinforcement Learning

In recent years, taking advantage of neural networks as function approximations for policies and values, there have emerged many studies on deep reinforcement learning (DRL) (Lillicrap et al., 2015; Mnih et al., 2013). Trust region policy optimization (Kakade and Langford, 2002; Schulman et al., 2015, 2017) is one of the most successful DRL methods in the single-agent setting, which places constraints on the step-size of policy updating to monotonically preserve improvements. Based on the monotonic improvement from TRPO (Schulman et al., 2015) in single-agent learning, MATRL extends the improvement guarantee to multi-agent levels towards a weak stable fixed point. On one hand, some works directly apply fully decentralized single-agent DRL methods (Tan, 1993), which are known as ILs; although, theoretically, ILs are prone to be unstable during learning due to the non-stationary issue. However, based on competitive returns in a range of environments, Papoudakis et al. (2020b) found that ILs can achieve such returns despite its simplicity. This conclusion agreed with the study results that are presented in Chapter 6, where the simplicity advantages of ILs are discussed and a centralized mechanism is provided to adjust the gradient step-size, which can largely speed up and stabilize ILs learning.

Interestingly, the maximum-entropy framework has also been explored in the RL domain through inference on graphical models (Levine, 2018); *soft* Q-learning (Haarnoja et al., 2017) and actor-critic (Haarnoja et al., 2018) methods were developed. Recently, *soft* learning has been adapted into the context of MARL Tian et al. (2019); Wei et al. (2018), which allows for suboptimal behavior and measurement of the Bayesian bounded rationality of the agent (Ziebart, 2010). The methods presented in Chapter 3 and 4 are based on soft RL, which embeds the bounded rationality into multi-agent recursive reasoning.

Despite the recent success of applying deep RL algorithms on the discrete (Mnih et al., 2015) and continuous (Lillicrap et al., 2015) control problems in the single-agent case, it is still challenging to transfer these methods into the multi-agent RL context. The reason is that independent learning will ignore others in the environment, which breaks the theoretical guarantee of convergence (Tuyls and Weiss, 2012). To address the non-stationary issue,Foerster et al. (2016); Peng et al. (2017); Sukhbaatar et al. (2016) added an extra communication channel during the training and execution in a centralized way to avoids non-stationarity. On the other hand, a modern framework is centralized training with decentralized execution (CTDE), such as MADDPG (Lowe et al., 2017) and COMA (Foerster et al., 2018b), which use a centralized critic to address the non-stationary issue. However, these frameworks require strong assumptions that the policy networks' parameters are fully observable (so does LOLA by Foerster et al. (2018a)), and centralized Q-network would prohibit the algorithms from scaling up. Additionally, there is a critical branch of CTDE methods and value decomposition for solving the Dec-POMDP problem, which would be discussed in the next section. Chapter 5 is based on value decomposition methods, and the detailed introduction is given in the next section. By contrast, the other approaches presented in Chapter 3, 4, and 6 are capable of employing decentralized training with no need to maintain a central critic; furthermore, the proposed method needs to know the parameters of the opponents' policies.

2.3.1 Joint Value Function Decomposition

In a multi-agent learning algorithm, apart from the joint value function, people usually need to obtain a decentralized policy for each agent. CTDE is a paradigm for solving Dec-POMDP (Oliehoek et al., 2008), which allows the algorithm access to all of the agents' local histories $\boldsymbol{\tau}$ during training. During testing, however, the algorithm uses the history of each of the agents τ_i for execution. CTDE methods provide valid solutions to multi-agent cooperative tasks that are *decentralizable*, which are formally defined as below.

Definition 2.2 (Decentralizable Cooperative Tasks, a.k.a. IGM Condition Son et al. (2019)). *A cooperative task is decentralizable if* $\exists \{Q_i\}_{i=1}^N$ *such that* $\forall \boldsymbol{\tau} \in \tau^N, \boldsymbol{a} \in \mathcal{A}^N$,

$$\arg\max_{\boldsymbol{a}} Q^{\boldsymbol{\pi}}(\boldsymbol{\tau}, \boldsymbol{a}) = \begin{bmatrix} \arg\max_{a_1} Q_1(\tau_1, a_1) \\ \vdots \\ \arg\max_{a_N} Q_N(\tau_N, a_N) \end{bmatrix}. \tag{2.9}$$

Equation 2.9 suggests that the local maxima on the extracted value function per agent needs to amount to the global maximum on the joint value function. A key challenge for CTDE methods is how to correctly extract each of the agent's individual Q-function $\{Q_i\}_{i=1}^N$, and as such an executable policy, from a centralized Q-function $Q^{\boldsymbol{\pi}}$. In previous works, VDN (Sunehag et al., 2018), QMIX (Rashid et al., 2018), and QTRAN (Son et al., 2019) have progressively enlarged the family of functions that can be represented by the mixing network. MAVEN (Mahajan et al., 2019) is built on top of the QMIX with additional exploration. More recently, Wang et al. (2020) and Su et al. (2020) extended value-decomposition to actor-critics. However, the structural constraints in the mixing network put forward by the above methods inhibit the representational power of the centralized value function Therefore, I propose a new joint value function approximator in Chapter 5, which can promote diversity while removing these structural constraints.

2.4 Agent Modeling

My work is closely tied to the traditional study of opponent modeling (OM). I refer to a comprehensive survey about OM from (Albrecht and Stone, 2018). OM constructs models of other agents and can be used as extra information to guide agents' learning. The first application of OM in multi-agent learning is in fictitious play (Brown, 1951), in which each agent models other agents' strategies by counting the frequency of their past play. Similarly, joint-action learners (Claus and Boutilier, 1998) consider their opponents by explicitly counting the joint-action of all the agents in the Q-learning update. However, both of these methods assume that agents play stationary strategies.

In the face of non-stationary opponents, discounting old experience and adaptively updating with the newest information has proved to be effective for solving repeated games (Bouzy and Métivier, 2010). A similar idea can also be found in a series of *WoLF* models (Bowling, 2005; Bowling and Veloso, 2001a, 2002)), where the speed of the learning rate is tuned based on a winning or losing situation. *WoLF* methods have been proved to converge in self-play with a two-person, two action repeated matrix game. Another effective method for solving games with non-stationary opponents is to make the model find the best response to a predefined target of opponents. Nash-Q citephu2003nash learns the best-response model when all agents play to try and reach Nash equilibrium. Correlated Q-learning targets towards the correlated equilibrium (Greenwald et al., 2003). Minimax-Q tackles cases in which the opponent is diametrically opposed (Littman, 1994). Friend-or-foe

Q (Littman, 2001) generalizes on both Minimax-Q and Nash-Q. One advantage of fixing the opponent type is that it is easier to provide guarantees against specific opponents than against general classes; however, when the opponent does not follow the assumptions, the algorithms will simply fail to converge or achieve the goals. Furthermore, knowing the opponent type, especially what equilibrium is will choose at different stages, is a strong assumption. In Nash-Q learning, apart from a toy example, such as the prison dilemma, it is computationally infeasible to know which equilibrium each agent will choose in advance. In contrast, our proposed method does not assume the type of the opponent, instead, for each agent, it models the posterior action distribution of others based on the newest information, and subsequently derives an acting policy based on it, By successfully modeling the opponents, the agent can reason about its opponents' behaviors and goals and adjust its policy to achieve optimal outcomes.

More recently, some works have applied neural networks to learn various representations in opponent models. He et al. (2016); Hong et al. (2018) presented a deep neural network to predict opponents' actions and intentions given the opponents' observations as inputs. On the contrary, Raileanu et al. (2018) modeled others using the policy of the agent itself. Grover et al. (2018) decoded the opponent's policy from encoded opponent trajectory representation. Based on the graph neural networks, Tacchetti et al. (2019) proposed relational forward models to model opponents. In addition, Hernandez-Leal et al. (2019) converted agent modeling as an auxiliary task in deep reinforcement learning. These methods usually assume some knowledge of the opponents. To minimize the information requirement, Papoudakis and Albrecht (2020); Papoudakis et al. (2020a) used a variational auto-encoders-based method that only requires agent's local information to approximate the opponent models. The recursive reasoning framework is closer to opponent modeling with correlated policies (Liu et al., 2020; Tian et al., 2019), which considers multi-agent settings with correlated joint policy factorization (Albrecht and Stone, 2018), i.e., where agents' actions influence each other. The proposed approaches presented in Chapter 3 and 4 learn opponent models to consider possible opponent actions during learning.

2.4.1 Recursive Reasoning

Recursive reasoning has been used to successfully profile human social behaviors, such as the scenario of school violence (Pynadath and Marsella, 2005). Interestingly, after studying a large corpus of human play data, researchers found that humans tend to reason, on average, at one or two levels of recursion (Camerer et al., 2004), and reasoning levels larger than two do not provide significant benefits over deeper levels (De Weerd et al., 2013a,b; de Weerd et al., 2017), possibly because of the biological effectiveness of the human brain (Kimbrough et al., 2014) or because co-operation favors lower levels of nested beliefs (Devaine et al., 2014). Game theorists take initiatives in modeling the infinite recursive reasoning procedures (Harsanyi, 1962, 1967). Since then, other quantitative alternatives have been developed to represent the idea of nested beliefs, such as those based on logic (Bolander and Andersen, 2011; Muise et al., 2015), or the graphical models (Doshi et al., 2009; Gal and Pfeffer, 2003, 2008). Studies on Theory of Mind (ToM) (Goldman et al., 2012; Rabinowitz et al., 2018) explicitly model agent's belief on opponents' mental states in the RL setting. ToM is the concept that describes the hierarchical cognitive mechanism of attributing *unobservable* mental beliefs to others. A representative of the ToM method is the "Recursive Modeling Method" (RMM) (Gmytrasiewicz and Durfee, 1995, 2000; Gmytrasiewicz et al., 1991), which incorporates the agent's uncertainty about its opponent's exact model, payoff, and recursion depth. Von Der Osten et al. (2017) developed the framework of Multi-agent ToM with more than one opponent by segmenting the agent population into groups. Shum et al. (2019) introduced a generative model of multi-agent action understanding based on a novel representation of these latent relationships to understand behavior in a multi-agent group. Doshi et al. (2020) reviewed various frameworks for recursive modeling that have been studied in games and decision-making.

Particularly, I-POMDP (Gmytrasiewicz and Doshi, 2005) is one of the most well-known frameworks, and it focuses on building the belief state space about opponents. The resulting hierarchical belief structure represents an agent's belief about the physical state, other agents, and others' beliefs and can be nested infinitely in this recursive manner (Gmytrasiewicz and Doshi, 2005; Han and Gmytrasiewicz, 2019). Based on the I-POMDP framework, Han and Gmytrasiewicz (2018) narrow the problem to learn an intentional model of others via Bayesian inference and sequential Monte Carlo sampling. Furthermore, IPOMDP-Net (Han and Gmytrasiewicz, 2019) extends the intentional models with deep neural networks. The recursive reasoning models presented in Chapter 3 and 4 have a strong connection with I-POMDP, aiming to build a nested belief about the opponents. However, my method is different from I-POMDP with regard to the original settings and targets. Although the partially observable setting makes the setting more general, constructing I-POMDP models manually or learning them from observations is difficult. Therefore, I simplify the setting by assuming fully observable states. Similarly to IPOMDP-Net (Han and Gmytrasiewicz, 2019), I focus on modeling the impact of the other agents' actions, which indirectly impacts the policies I present. I do not need to hold a full belief state; therefore, I build a model of other agents that helps agents recursively reason about their actions and consequently benefit decision-making. In addition, the model presented in this thesis can consider opponents with different levels of bounded rationality and whether an equilibrium exists in such a sophisticated hierarchical framework.

2.4.2 Cognitive Hierarchy

Assigning agents with different levels of reasoning power, e.g., the level-k model, was first introduced by (Stahl, 1993; Stahl II and Wilson, 1994) and (Nagel, 1995). In the level-k model, agents anchor their beliefs through thought-experiments with the iterated best response in a chain style, i.e., L1 best responds to L0, L2 best responds to L1 and so on. It is expected that an agent will exhibit a behavior pattern whose complexity grows as the cognitive levels increase (Gracia-Lázaro et al., 2017). Interestingly, across different games in the real world (Benndorf et al., 2017; Camerer et al., 2004), humans tend to reason between 1 and 2 levels of recursion, which explains why people commonly guess between 13 to 25 in the Keynes Beauty Contest. However, the level-k models are shown to exhibit an extreme stereotype bias due to concentrating on only one level below (Chong et al., 2016). (Camerer et al., 2004)) extended the level-k model by making the Lk best respond not only to the level-$k-1$ but also to a mixture of lower types of agents; this model is named the cognitive hierarchy model (CHM). The mixture of the lower hierarchy is usually approximated by a discrete probability distribution, with its parameter estimated from empirical, experimental data. CHM presents distinct advantages over the level-k model on making robust predictions in a series of games (Crawford et al., 2013). In Chapter 4, inspired by the idea of cognitive hierarchy, I embed the mixture of level-k reasoning into MARL and derive the soft actor-critic algorithm with generalized recursive reasoning (GR2). By recognizing bounded rationality in the mixture of lower hierarchy, GR2 MARL methods are generalized across different types of opponents, thereby showing robustness to their suboptimal behaviors, which I believe is a critical property for modern AI applications.

2.5 Empirical Game-Theoretic Analysis

Empirical Game-Theoretic Analysis (EGTA) (Jordan and Wellman, 2009; Tuyls et al., 2020, 2018) creates a policy-space meta-game for modeling multi-agent interactions, in which strategies in meta-game correspond to policies rather than primitive actions. The payoffs of the meta-game are then constructed by simulating games using all the joint policy combinations. Due to the much smaller game size, conducting a game-theoretic analysis on the meta-game becomes possible, and the results are analogous to the under-layer game. Policy-Space Response Oracle (PSRO) (Balduzzi et al., 2019; Lanctot et al., 2017; Omidshafiei et al., 2019) is one of the well-known methods based on EGTA. Given an under-layer game, PSRO consists of the following three iterated phases: complete, solve, and expand (Muller et al., 2020). In the complete phase, the meta-game policy sets are usually initialized using randomly-generated policies, and the missing entries in meta with are completed through game simulations. Then, in the solve phase, a meta-solver computes a meta-game solution (e.g., Nash or uniform distributions) over the agent policies and obtains the mixture policies based on the solution. Finally, in the expand phase, with the help of independent learners, the new policies are appended to the meta-game policy sets by approximating the best response to a mixture of other agents' policies generated, and the algorithm iterates. PSRO also generalizes Fictitious Play (FP) (Brown, 1951; Heinrich and Silver, 2016; Leslie and Collins, 2006), which provides a training framework in which multiple policies from agents play against each other. Although these methods have a good generalization ability in various multi-agent tasks, they require a large number of computing resources to estimate the empirical meta-game and solve it with its increasing size (Omidshafiei et al., 2019; Yang et al., 2019) In this Chapter 6, the proposed method adopts the idea of a policy-space meta-game to approximate the fixed-point. Unlike previous EGTA works, the adopted idea only maintains the current and predicted policies to construct the meta-game, which is computationally achievable in most cases. The payoff entry in MATRL's meta-game is the expected advantage, which has a lower estimation variance than the commonly used empirically estimated return in EGTAs. Additionally, I can reuse the trajectories in the TPR step to estimate the payoffs without extra sampling costs.

Part I

Explicit Mutual Influence Models

Chapter 3

Probabilistic Recursive Reasoning

Constructing the models of other agents, also known as opponent modeling, has a rich history in the multi-agent learning (Albrecht and Stone, 2018; Shoham et al., 2007). Even though equipped with modern machine learning methods that could enrich the representation of the opponent's behaviors (Foerster et al., 2018a; He et al., 2016; Yang et al., 2018a), those algorithms focus on either limited types of scenarios (e.g. cooperative games, mean-field games), pre-defined opponent strategies (e.g. Tit-fot-Tat in iterated Prisoner's Dilemma), or the cases where opponents are assumed to constantly return to the same strategy (Da Silva et al., 2006). Besides, a promising methodology from game theory, recursive reasoning (Camerer et al., 2004; De Weerd et al., 2013b; Gmytrasiewicz and Doshi, 2005; Gmytrasiewicz and Durfee, 2000), is rarely be mentioned. Similar to the way of thinking of humans, recursive reasoning refers to the belief reasoning process where each agent considers the reasoning process of other agents, based on which it expects to make better decisions. Importantly, recursive reasoning allows an opponent to reason about the modeling agent rather than being a fixed type; the process can therefore be nested in a form as "I believe that you believe that I believe ... ". Despite some initial trails (Gmytrasiewicz and Doshi, 2005; Von Der Osten et al., 2017), there has been little work that tries to adopt this idea into the mutli-agent deep reinforcement learning (DRL) setting. One main reason is that computing the optimal policy is prohibitively expensive (Doshi and Gmytrasiewicz, 2006; Seuken and Zilberstein, 2008).

In this chapter, we introduce a probabilistic recursive reasoning (PR2) framework for multi-agent DRL tasks. Unlike previous opponent models, each agent is to consider how the opponents would react to its potential behaviors before it tries to find the best response for its own decision making, which doesn't consider the mutual influence between its policy and opponents' policies. By employing variational Bayes methods to model the uncertainty of opponents' conditional policies, We develop decentralized-training-decentralized-execution algorithms, PR2-Q and PR2-Actor-Critic, and prove their convergence in the self-play scenario. The proposed methods are tested on the matrix game and the differential game. The games come with a non-trivial equilibrium where conventional gradient-based methods find challenging. We compare against multiple strong baselines. The results justify better convergence ability by agent's recursive reasoning throughout the learning. We expect our work to offer a new angel on incorporating conditional opponent modeling into the multi-agent DRL context.

3.1 Preliminaries: Joint Policy Factorization

In the multi-agent learning tasks, each agent can only control its own action; however, the resulting reward value depends on other agents' actions. In other words, the Q-function of each agent, $Q_i^{\pi^\theta}$, is subject to the joint policy π^θ consisting of all agents' policies. In the previous studies, one common approach is to decouple the joint policy assuming conditional independence of actions from different agents (Albrecht and Stone, 2018):

$$\pi^\theta(a_i, a_{-i}|s) = \pi_i^{\theta_i}(a_i|s)\pi_{-i}^{\theta_{-i}}(a_{-i}|s), \tag{3.1}$$

where the joint policy $\pi^\theta(a_i, a_{-i}|s)$ is factorized to the product of two independent policies $\pi_i^{\theta_i}(a_i|s)$ and $\pi_{-i}^{\theta_{-i}}(a_{-i}|s)$. The study regarding the topic of "centralized training with decentralized execution" in the deep RL domain, including MADDPG (Lowe et al., 2017), COMA (Foerster et al., 2018b), MF-AC (Yang et al., 2018a), Multi-Agent Soft-Q (Wei et al., 2018), and LOLA (Foerster et al., 2018a), can be classified into this category. Although the non-correlated factorization of the joint policy simplifies the algorithm, this simplification is typically invalid by ignoring the agents' connections, e.g. impacts of one agent's action on other agents,

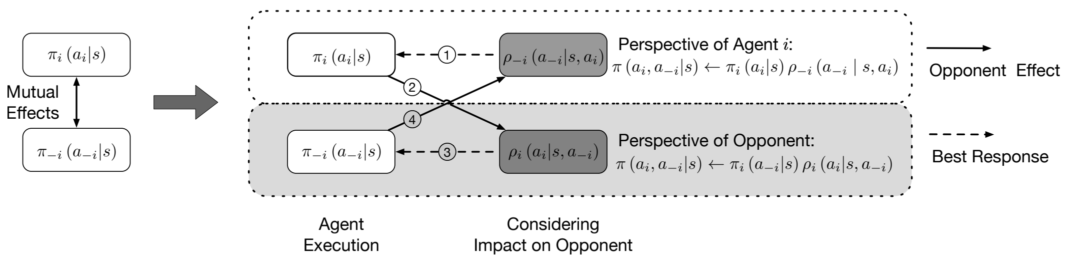

$\pi_i(a_i|s)$
Mutual Effects
$\pi_{-i}(a_{-i}|s)$
$\pi_i(a_i|s)$
1
$\rho_{-i}(a_{-i}|s,a_i)$
Perspective of Agent i:
$\pi(a_i,a_{-i}|s) \leftarrow \pi_i(a_i|s)\,\rho_{-i}(a_{-i} \mid s,a_i)$
2
4
$\pi_{-i}(a_{-i}|s)$
3
$\rho_i(a_i|s,a_{-i})$
Perspective of Opponent:
$\pi(a_i,a_{-i}|s) \leftarrow \pi_{-i}(a_{-i}|s)\,\rho_i(a_i|s,a_{-i})$
Opponent Effect
Best Response
Agent Execution
Considering Impact on Opponent

Figure 3.1: Diagram of our probabilistic recursive reasoning framework. PR2 decouples the connections between agents by Eq. 3.2. ①: agent i takes the best response after considering all the potential consequences of opponents' actions given its own action a^i. ②: how agent i behaves in the environment serves as the prior for the opponents to learn how their actions would affect a^i. ③: similar to ①, opponents take the best response to agent i. ④: similar to ②, opponents' actions are the prior knowledge to agent i on estimating how a^i will affect the opponents. Looping from step 1 to 4 forms recursive reasoning.

and the impacts from other agents. One might argue that during training, the joint Q-function should potentially guide each agent to learn to consider and act for the mutual interests of all the agents; nonetheless, a counter-example is that the non-correlated policy could not even solve the simplest two-player zero-sum differential game where two agents act in x and y with the reward functions defined by $(xy, -xy)$: following by Equation 3.1, both agents are reinforced to trace a cyclic trajectory that never converges to the equilibrium (Mescheder et al., 2017).

It is worth clarifying that the idea of non-correlated policy is still markedly different from the independent learner (IL). IL is a naive method that completely ignore other agents' behaviors. The objective of agent i is simplified to $\eta_i(\pi^{\theta_i})$, depending only on i's own policy π^{θ_i} compared to Equation 2.4. As Lowe et al. (2017) has pointed out, in IL, the probability of taking a gradient step in the correct direction decreases exponentially with the increasing number of agents, letting alone the major issue of the non-stationary environment due to the independence assumption (Tuyls and Weiss, 2012).

3.2 Probabilistic Recursive Reasoning

In the previous section, we have shown the weakness of the learning algorithms that build on the non-correlated factorization on the joint policy. Here we introduce the probabilistic recursive reasoning approach that aims to capture how the opponents believe about what the agent believes. Under such settings, we devise a new multi-agent policy gradient theorem. We start from assuming the true opponent conditional policy $\pi_{-i}^{\theta_{-i}}$ is given, and then move onward to the practical case where it is approximated through variational inference.

The issue on the non-correlated factorization is that it fails to help each agent to consider the impact of its action on others, which could lead to the ill-posed behaviors in the multi-agent learning tasks. On the contrary, people explicitly attribute contents such as beliefs, desires, and intentions to others in daily life. It is known that human beings are capable of using this ability recursively to make decisions. Inspired by this, here we integrate the concept of recursive reasoning into the joint policy modeling, and propose the new probabilistic recursive reasoning (PR2) framework. Specifically, we employ the nested process of belief reasoning where each agent simulates the reasoning process of other agents, thinking about how its action would affect others, and then make actions based on such predictions. The process can be nested in a form as "I believe [that you believe (that I believe)]". Here we start from considering the level-1 recursion, as psychologist have found that humans tend to reason on average at one or two level of recursion (Camerer et al., 2004), and levels higher than two do not provide significant benefits (De Weerd et al., 2013a,b; de Weerd et al., 2017). Based on this, we re-formulate the joint policy by

$$\pi^{\theta}(a_i, a_{-i}|s) = \underbrace{\pi_i^{\theta_i}(a_i|s)\pi_{-i}^{\theta_{-i}}(a_{-i}|s, a_i)}_{\text{Agent } i\text{'s perspective}} = \underbrace{\pi_{-i}^{\theta_{-i}}(a_{-i}|s)\pi_i^{\theta_i}(a_i|s, a_{-i})}_{\text{The opponents' perspective}}. \tag{3.2}$$

Similar way of decomposition can also be found in dual learning (Xia et al., 2017) on symmetrical tasks such as machine translation. From the perspective of agent i, the first equality in Equation 3.2 indicates that the joint policy can be essentially decomposed into two parts. The conditional part $\pi_{-i}^{\theta_{-i}}(a_{-i}|s, a_i)$ represents what actions would be taken by the opponents given the fact that the opponents know the current state of environment and agent i's action; this is based on what agent i believes other opponents might think about itself. Note that the way of thinking developed by agent i regarding how others would consider of itself is also shaped by opponents' original policy $\pi_{-i}^{\theta_{-i}}(a_{-i}|s)$, as this is also how the opponents actually act in the environment. Taking into account different potential actions that agent i thinks the opponents would take, agent i uses the marginal policy $\pi_i^{\theta_i}(a_i|s)$ to find the best response. To this end, a level-1 recursive procedure is established: $a_i \to a_{-i} \to a_i$. The same inference logic can be applied to the opponents from their perspectives, as shown in the second equality of Equation 3.2.

Albeit instructive, Equation 3.2 may not be practical due to the requirement on the full knowledge regarding the actual conditional policy $\pi_{-i}^{\theta_{-i}}(a_{-i}|s, a_i)$. A natural solution is that one approximates the actual policy via a best-fit model from a family of distributions. We denote this family as $\rho_{-i}^{\phi_{-i}}(a_{-i}|s, a_i)$ with learnable parameter ϕ_{-i}. PR2 is probabilistic as it considers the uncertainty of modeling $\pi_{-i}^{\theta_{-i}}(a_{-i}|s, a_i)$. The reasoning structure is now established as shown in Figure 3.1. With the recursive joint policy defined in Equation 3.2, the n-agent learning task can therefore be formulated as

$$\arg\max_{\theta_i, \phi_{-i}} \ \eta_i \left(\pi_i^{\theta_i}(a_i|s) \rho_{-i}^{\phi_{-i}}(a_{-i}|s, a_i) \right), \tag{3.3}$$

$$\arg\max_{\theta_{-i}, \phi_i} \ \eta_{-i} \left(\pi_{-i}^{\theta_{-i}}(a_{-i}|s) \rho_i^{\phi_i}(a_i|s, a_{-i}) \right). \tag{3.4}$$

With the new learning protocol defined in Equation 3.3 and 3.4, each agent now learns its own policy as well as the approximated conditional policy of other agents given its own actions. In such a way, both the agent and the opponents can keep track of the joint policy by $\pi_i^{\theta_i}(a_i|s)\rho_{-i}^{\phi_{-i}}(a_{-i}|s, a_i) \to \pi^{\theta}(a_i, a_{-i}|s) \leftarrow \pi_{-i}^{\theta_{-i}}(a_{-i}|s)\rho_i^{\phi_i}(a_i|s, a_{-i})$. Once agents' policies and the approximated opponent conditional policies converge to a stationary distribution, we can verify the resulting approximates satisfies at each state: $\pi^{\theta}(a_i, a_{-i}|s) = \pi_i^{\theta_i}(a_i|s)\rho_{-i}^{\phi_{-i}}(a_{-i}|s, a_i) = \pi_{-i}^{\theta_{-i}}(a_{-i}|s)\rho_i^{\phi_i}(a_i|s, a_{-i})$, according to Equation 3.2.

3.2.1 Probabilistic Recursive Reasoning Policy Gradient

In this section, given the true opponent policy $\pi_{-i}^{\theta_{-i}}$ and that each agent tries to maximize its objective defined in Equation 2.4, we establish the policy gradient theorem by accounting for the PR2 joint policy decomposition in Equation 3.2. We

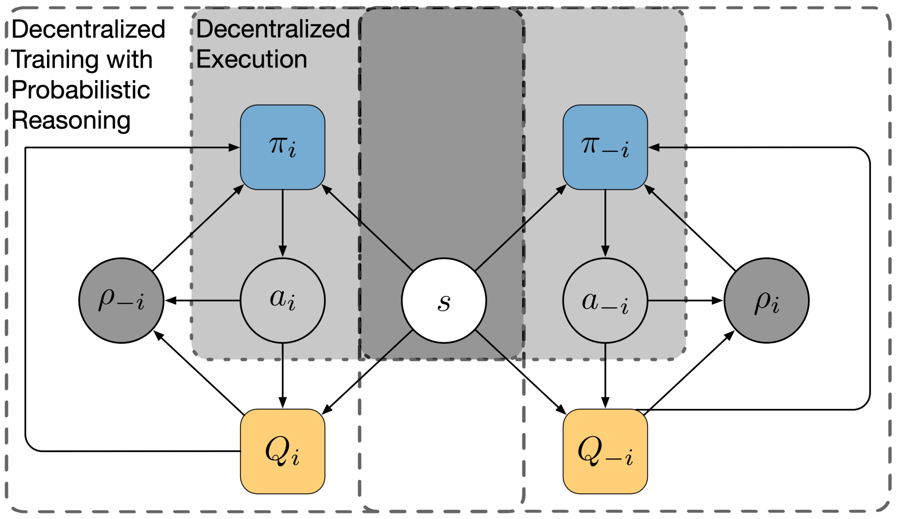


Figure 3.2: Diagram of multi-agent probabilistic recursive reasoning learning algorithms. It conducts decentralized training with decentralized execution. The light grey areas on two sides of middle indicate decentralized execution for each agent. White areas give the decentrilized learning procedures. All agents share the interaction experiences in the environment represented by dark area in the middle.

also quantify the connection between decentralized training and centralized learning via importance sampling weight in Proposition 3.2.

Proposition 3.1. *In a stochastic game, under the recursive reasoning framework defined by Equation 3.2, the update for the multi-agent recursive reasoning policy gradient method can be derived as follows:*

$$\nabla_{\theta^i}\eta_i = \mathbb{E}_{s\sim p, a_i\sim\pi_i}\left[\nabla_{\theta_i}\log\pi_i^{\theta^i}(a_i|s)\int_{a_{-i}}\pi_{-i}^{\theta_{-i}}(a_{-i}|s,a_i)Q_i(s,a_i,a_{-i})\,\mathrm{d}a_{-i}\right]. \tag{3.5}$$

Proof. If we apply the chain rule to factorize the joint policy to: $\pi^\theta(a_i,a_{-i}|s) = \pi_i^{\theta_i}(a_i|s)\pi_{-i}^{\theta_{-i}}(a_{-i}|s,a_i)$. Then, we can have multi-agent recursive reasoning objective function:

$$\begin{aligned}\eta_i &= \int_s\int_{a_i}\int_{a_{-i}}\pi(a_i,a_{-i}|s)Q_i(a_i,a_{-i})\,\mathrm{d}a_{-i}\,\mathrm{d}a_i\,\mathrm{d}s\\ &= \int_s\int_{a_i}\pi_i(a_i|s)\int_{a_{-i}}\pi_{-i}(a_{-i}|s,a_i)Q_i(s,a_i,a_{-i})\,\mathrm{d}a_{-i}\,\mathrm{d}a_i\,\mathrm{d}s.\end{aligned} \tag{3.6}$$

Compare to non-correlated factorization, a_{-i} in Equation 3.6 is additionally conditional on a_i where a_i is the prior. That is we introduce agent i'a action a_i into other agents $-i$'s policy, obtaining $\pi_{-i}(a_{-i}|s,a_i)$. It adds the consideration that agent i can have influence on other agents. We now compute the policy gradient analytically. Following the single agent Policy Gradient Theorem with Leibniz integral rule and Fubini's theorem, we get the multi-Agent Recursive Reasoning Policy Gradient:

$$\nabla^{\theta_i}\eta_i = \mathbb{E}_{s\sim p, a_i\sim\pi_i}[\nabla^{\theta_i}\log\pi_i(a_i|s)\int_{a_{-i}}\pi_{-i}(a_{-i}|s,a_i)Q_i(s,a_i,a_{-i})\,\mathrm{d}a_{-i}]. \tag{3.7}$$

However, in common case, the agent cannot get access to other agents' policies. We need to keep an approximation of other agents. We let $\rho_{-i}^{\phi_{-i}}(a_{-i}|s,a_i)$ denotes the parameterized opponent conditional policy of agent i to approximate other agents policies, i.e, $\pi_{-i}(a_{-i}|s,a_i)$. In such way, without the centralized knowledge of exact opponent policy, we have Decentralized Multi-Agent Recursive Reasoning Policy Gradient comes as:

$$\begin{aligned}\nabla^{\theta_i}\eta_i &\approx \mathbb{E}_{s\sim p, a_i\sim\pi_i}[\nabla^{\theta_i}\log\pi_i^{\theta_i}(a_i|s)\int_{a_{-i}}\rho_{-i}^{\phi_{-i}}(a_{-i}|s,a_i)Q_i(s,a_i,a_{-i})\,\mathrm{d}a_{-i}]\\ &= \mathbb{E}_{s\sim p, a_i\sim\pi_i}[\nabla^{\theta_i}\log\pi_i^{\theta_i}(a_i|s)Q_i^{\rho_{-i}^{\phi_{-i}}}(s,a_i)].\end{aligned} \tag{3.8}$$

In Equation 3.8, the gradient for agent i is scaled by $Q_i^{\rho_{-i}^{\phi_{-i}}}(s,a_i) = \int_{a_{-i}}\rho_{-i}^{\phi_{-i}}(a_{-i}|s,a_i)Q_i(s,a_i,a_{-i})\,\mathrm{d}a_{-i}$. Then, the trajectories generated by updated policy would help to train $\rho_{-i}^{\phi_{-i}}(a_{-i}|s,a_i)$ and $Q_i(s,a_i,a_{-i})$. These steps form a EM style learning procedures: fix $\rho_{-i}^{\phi_{-i}}$ and $Q_i(s,a_i,a_{-i})$, improve $\pi_i^{\theta_i}(a_i|s)$; then $\rho_{-i}^{\phi_{-i}}$ and $Q_i(s,a_i,a_{-i})$ would benefit from the trajectories brought by $\pi_i^{\theta_i}(a_i|s)$. Furthermore, because we do not need opponents' actual private policies, Decentralized Multi-Agent Recursive Reasoning Policy Gradient decouple from other agents' on-policies or target policies. This means the training can be done in an off-policy fashion by sampling batches from memory buffer D with the help of the on-policy learned $\rho_{-i}^{\phi_{-i}}(a_{-i}|s,a_i)$ from $Q_i(s,a_i,a_{-i})$. ■

Proposition 3.1 states that each agent should improve its policy toward the direction of the best response after it takes into account all kinds of possibilities of how other agents would react if that action is taken. The term of $\pi_{-i}^{\theta_{-i}}(a_{-i}|s,a_i)$ can be regarded as the posterior estimation of agent i's belief about how the opponents would respond to his action a_i, given opponents' true policy $\pi_{-i}^{\theta_{-i}}(a_{-i}|s)$ serving as the prior. Note that compared to the direction of policy update in the conventional multi-agent policy gradient theorem (Wei et al., 2018), $\int_{a_{-i}} \pi_{-i}^{\theta_{-i}}(a_{-i}|s)Q_i(s,a_i,a_{-i})\,\mathrm{d}a_{-i}$, the direction of the gradient update in PR2 is guided by the term $\int_{a_{-i}} \pi_{-i}^{\theta_{-i}}(a_{-i}|s,a_i)Q_i(s,a_i,a_{-i})\,\mathrm{d}a_{-i}$.

In practice, agent i might not have access to the opponents' actual policy parameters θ_{-i}, it is often needed to approximate $\pi_{-i}^{\theta_{-i}}(a_{-i}|s,a_i)$ by $\rho_{-i}^{\phi_{-i}}(a_{-i}|s,a_i)$, thereby we propose Proposition 3.2.

Proposition 3.2 raises an important point: the difference between decentralized training (algorithms that do not require the opponents' policies) with centralized learning (algorithms that require the opponents' policies) can in fact be quantified by a term of importance weights, similar to the connection between on-policy and off-policy methods. If we find a best-fit approximation such that $\rho_{-i}^{\phi_{-i}}(a_{-i}|s,a_i) \to \pi_{-i}^{\theta_{-i}}(a_{-i}|s,a_i)$, then Equation 3.9 collapses into Equation 3.5.

Proposition 3.2. *In a stochastic game, under the recursive reasoning framework defined by Equation 3.2, with the opponent policy approximated by $\rho_{-i}^{\phi_{-i}}(a_{-i}|s,a_i)$, the update for the multi-agent recursive reasoning policy gradient method can be formulated as follows:*

$$\nabla_{\theta^i}\eta_i = \mathbb{E}_{s\sim p, a_i\sim\pi_i}\left[\nabla_{\theta_i}\log\pi_i^{\theta^i}(a_i|s)\cdot\mathbb{E}_{a_{-i}\sim\rho_{-i}^{\phi_{-i}}}\left[\frac{\pi_{-i}^{\theta_{-i}}(a_{-i}|s,a_i)}{\rho_{-i}^{\phi_{-i}}(a_{-i}|s,a_i)}Q_i(s,a_i,a_{-i})\right]\right]. \tag{3.9}$$

Proof. Substituting the approximated model $\rho_{-i}^{\phi_{-i}}(a_{-i}|s,a_i)$ for the true policy $\pi_{-i}^{\theta_{-i}}$ in Equation 3.5. ■

Based on Proposition 3.2, Algorithm 3.1 gives the step by step learning procedures for PR2-AC algorithm. As illustrated in Figure 3.2, it is a decentralized-

Algorithm 3.1 Multi-Agent Probabilistic Recursive Reasoning Actor Critic (PR2-AC).

```
1: Policy:$\pi_i$, Opponent Recursive Reasoning: $\rho_{-i}(a_{-i}|s, a_i)$.
2: Initialize parameters $\theta_i, \phi_{-i}, \omega_i$ arbitrarily for each agent $i$, random process $\mathcal{N}$
   for action exploration.
3: Assign target parameters of joint action Q-function: $\omega_{i'} \leftarrow \omega_i$, and target policy
   parameter: $\theta_{i'} \leftarrow \theta_i$, $D_i \leftarrow$ empty replay buffer for each agent.
4: for each episode do
5:   Initialize random process $\mathcal{N}$ for action exploration.
6:   for each step $t$ do
7:     Given the current $s$, for each agent $i$, select action $a_i = \mu_i^{\theta_i}(s) + \mathcal{N}^t$
8:     Take the joint action $(a_i, a_{-i})$ and observe own reward $r_i$ and new state $s'$
9:     Add the tuple $(s, a_i, a_{-i}, r_i, s')$ in corresponding replay buffer $D_i$ $s \leftarrow s'$
10:    for each agent $i$ do
11:      Sample a random mini batch $\{(s, a_i^j, a_{-i}^j, r_i^j, s'^{,j})\}_{j=0}^N$ from $D_i$
12:      Get $a_{i'}^j = \mu_{i'}^{\theta_i}$ for each state $s'^{,j}$
13:      Sample $\{a_{-i'}^{k,j}\}_{k=0}^M \sim \rho_{-i}^{\phi_{-i}}(\cdot|s'^{,j}, a_{i'}^j)$ for each $a_{i'}^j$ and $s'^{,j}$
14:      Set $y_i^j = r_i^j + \gamma \frac{1}{M} \sum_{k=0}^M Q_i^{\mu_{i'}}(s', a_{i'}, a_{-i'}^{k,j})$
15:      Update the critic by minimizing the loss $\mathcal{L}(\omega_i) = \frac{1}{N}\sum_{j=0}^N (y_j -
         Q_i^{\mu_i}(s_j, a_i^j, a_{-i}^j))^2$
16:      Update the actor using the sampled policy gradient:
         $$\nabla^{\theta_i}\eta_i \approx \frac{1}{N}\sum_{j=0}^{N} \nabla^{\theta_i}\mu_i(s_j)\nabla_{a_i}\frac{1}{M}\sum_{k=0}^{M} Q_i^{\mu_i}(s_j, a_i^j, a_{-i}^{k,j});$$
17:      Compute $\Delta\rho_{-i}^{\phi_{-i}}$ using empirical estimation
18:      Compute empirical gradient $\hat{\nabla}_{\phi_{-i}} J_{\rho_{-i}}$
19:      Update $\phi_{-i}$ according to $\hat{\nabla}_{\phi_{-i}} J_{\rho_{-i}}$
20:    end for
21:    Update target network parameters for each agent $i$:
         $$\theta_{i'} \leftarrow \lambda\theta_i + (1-\lambda)\theta_{i'};$$
         $$\omega_{i'} \leftarrow \lambda\omega_i + (1-\lambda)\omega_{i'};$$
22:  end for
23: end for
```

training-with-decentralized-execution algorithm. In this setting, agents share the experiences in the environment including state and historical joint actions, while each agent receive its rewards privately. Our method does not require the knowledge of other agents' policy parameters. We also list the pseudo codes of PR2-Q

Algorithm 3.2 Multi-Agent Probabilistic Recursive Reasoning Q-Learning (PR2-Q).

```
1: Policy: $\pi_i$, Opponent Recursive Reasoning: $\rho_{-i}(a_{-i}|s, a_i)$.
2: Initialize $Q_i(s, a_i, a_{-i})$ arbitrarily, set $\alpha$ as the learning rate, $\gamma$ as discount factor
3: while not converge do
4:     Given the current $s$, calculate the opponent best response $\rho_{-i}(a_{-i}|s, a_i)$ according to:
           $$\rho_{-i}(a_{-i}|s, a_i) = \exp(Q_i(s, a_i, a_{-i}) - Q_i(s, a_i))$$
5:     Select and sample action $a_i$ based on the Recursive Reasoning $\rho_{-i}(a_{-i}|s, a_i)$
           $$\text{softmax}(\int_{a_{-i}} \phi_{-i}(a_{-i}|s, a_i) Q_i(s, a_i, a_{-i}))$$
6:     Observing joint-action $(a_i, a_{-i})$, reward $r_i$, and next state $s'$
           $$Q_i(s, a_i, a_{-i}) \leftarrow (1 - \alpha) Q_i(s, a_i, a_{-i}) + \alpha(r_i + \gamma V_i(s'))$$
           $$Q_i(s, a_i) \leftarrow (1 - \alpha) Q_i(s, a_i) + \alpha(r_i + \gamma V_i(s'))$$
7:     where,
           $$V_i(s) = \max_{a_i} \int_{a_{-i}} \rho_{-i}(a_{-i}|s, a_i) Q_i(s, a_i, a_{-i})$$
8: end while
```

in Algorithm 3.2. Finally, one important piece missing is how to find a best-fit approximation of $\rho_{-i}^{\phi_{-i}}(a_{-i}|s, a_i)$.

The Algorithm 3.2 shows the variant of Decentralized Multi-Agent Probabilistic Recursive Reasoning. We can simply approximate the $\rho_{-i}(a_{-i}|s, a_i)$ by counting:$\rho_{-i}(a_{-i}|s, a_i) = C(a_i, a_{-i}, s)/C(a_i, s)$
in tabular if the state and action spaces are small, where C is the counting function. It this case, an agent only needs to learn a joint action Q-function, and if the game is also static game, it would be same as Conditional Joint Action Learning (CJAL) (Banerjee and Sen, 2007).

3.2.2 Variational Inference on Opponent Conditional Policy

We adopt an optimization-based approximation to infer the unobservable $\rho_{-i}^{\phi_{-i}}(a_{-i}|s,a_i)$ via variational inference (Jordan et al., 1999). We first define the trajectory τ up to time t including the experiences of t consecutive time stages, i.e. $\tau = [(s_1, a_i^1, a_{-i}^1), \ldots, (s^t, a_i^t, a_{-i}^t)]$. In the probabilistic reinforcement learning (Levine, 2018), the probability of τ being generated can be derived as

$$p(\tau) = \left[p(s_1)\prod_{t=1}^{T} p(s_{t+1}|s^t, a_i^t, a_{-i}^t)\right] \exp\left(\int_{t=1}^{T} r_i(s^t, a^t, a_{-i}^t)\, \mathrm{d}t\right). \tag{3.10}$$

Assuming the dynamics is fixed (i.e. the agent can not influence the environment transition probability), our goal is then to find the best approximation of $\pi_i^{\theta_i}(a_i^t|s^t)\rho_{-i}^{\phi_{-i}}(a_{-i}^t|s^t, a_i^t)$ such that the induced trajectory distribution $\hat{p}(\tau)$ can match with the true trajectory probability $p(\tau)$:

$$\hat{p}(\tau) = p(s_1)\prod_{t=1}^{T} p(s_{t+1}|s^t, a_i^t, a_{-i}^t)\pi_i^{\theta_i}(a_i^t|s^t)\pi_{-i}^{\theta_{-i}}(a_{-i}^t|s^t, a_i^t). \tag{3.11}$$

In other words, we can optimize the opponents' policy $\rho_{-i}^{\phi_{-i}}$ via minimizing the KL-divergence between $\hat{p}(\tau)$ and $p(\tau)$, i.e.

$$\begin{aligned} D_{\mathrm{KL}}(\hat{p}(\tau)\|p(\tau)) = \;& -\mathbb{E}_{\tau\sim\hat{p}(\tau)}[\log p(\tau) - \log \hat{p}(\tau)] \\ = \;& -\int_{t=1}^{t=T} E_{\tau\sim\hat{p}(\tau)}\Big[r^i\left(s^t, a_i^t, a_{-i}^t\right) \\ & + \mathcal{H}\left(\pi_{\theta_i}^i\left(a_i^t|s^t\right)\rho_{-i}^{\phi_{-i}}\left(a_{-i}|s^t, a_i^t\right)\right)\Big]. \end{aligned} \tag{3.12}$$

Minimizing the KL-divergence is equivalent to maximizing the reward; however, besides the reward term, the objective introduces an additional term of the conditional entropy on the joint policy $\mathcal{H}\left(\pi_{\theta_i}^i\left(a_i^t|s^t\right)\rho_{-i}^{\phi_{-i}}\left(a_{-i}|s^t, a_i^t\right)\right)$, that potentially promotes the explorations for both the agent i's best response and the opponents' conditional policy. Note that the entropy here is conditioning not only on the state of environment but also on agent i's action. Minimizing Equation 3.12 gives us:

Theorem 3.1. *The optimal Q-function for agent i that satisfies minimizing Equa-*

tion 3.12 is formulated as:

$$Q_i^{\pi^\theta}(s, a_i) = \log \int_{a_{-i}} \exp(Q_i^{\pi^\theta}(s, a_i, a_{-i}))\, \mathrm{d}a_{-i}. \tag{3.13}$$

And the corresponding optimal opponent conditional policy reads:

$$\rho_{-i}^{\phi_{-i}}(a_{-i}|s, a_i) = \exp(Q_{\pi^\theta}^i(s, a_i, a_{-i}) - Q_{\pi^\theta}^i(s, a_i)). \tag{3.14}$$

Proof. Follow the proof in (Haarnoja et al., 2017; Levine, 2018), we give the overall distribution at first:

$$p(\tau) = [p(s_1) \prod_{t=1}^{T} p(s^{t+1}|s^t, a_i^t, a_{-i}^t)] \exp(\int_{t=1}^{T} r_i(s^t, a^t, a_{-i}^t)\, \mathrm{d}t). \tag{3.15}$$

We can simplify in the case of deterministic dynamics, and we can adopt an optimization-based approximate inference approach to this problem, in which case the goal is to fit an approximation $\pi(a_i^t, a_{-i}^t|s^t) \approx \pi_i(a_i^t|s^t)\rho_{-i}(a_{-i}^t|s^t, a_i^t)$ such that the trajectory distribution:

$$\hat{p}(\tau) = p(s_1) \prod_{t=1}^{T} p(s^{t+1}|s^t, a_i^t, a_{-i}^t)\pi_i^{\theta_i}(a_i^t|s^t)\pi_{-i}^{\theta_{-i}}(a_{-i}^t|s^t, a_i^t). \tag{3.16}$$

In the case of exact inference, as derived in the previous section, the match is exact, which means that $D_{\mathrm{KL}}(\hat{p}(\tau)\|p(\tau)) = 0$, we can therefore view the inference process as minimizing the KL-divergence:

$$D_{\mathrm{KL}}(\hat{p}(\tau)\|p(\tau)) = -\mathbb{E}_{\tau\sim\hat{p}(\tau)}[\log p(\tau) - \log \hat{p}(\tau)]. \tag{3.17}$$

Negating both sides and substituting, we get:

$$\begin{aligned}
&-D_{\mathrm{KL}}(\hat{p}(\tau)\|p(\tau)) \\
=\;& \mathbb{E}_{\tau\sim\hat{p}(\tau)}\Big[\log p(s_1)+\int_{t=1}^{T}(\log p(s^{t+1}|s^t,a^t,a^t_{-i})+r_i(s^t,a^t_i,a^t_{-i}))\,\mathrm{d}t \\
&-\log p(s_1)-\int_{t=1}^{T}(\log p(s^{t+1}|s^t,a^t_i,a^t_{-i})+\log\pi(a^t_i,a^t_{-i}|s^t))\,\mathrm{d}t\Big] \\
=\;& \mathbb{E}_{\tau\sim\hat{p}(\tau)}\Big[\int_{t=1}^{T} r_i(s^t,a^t_i,a^t_{-i})-\log\pi(a^t_i,a^t_{-i}|s^t)\,\mathrm{d}t\Big] \\
=\;& \int_{t=1}^{T}\mathbb{E}_{(s^t,a^t_i,a^t_{-i})\sim\hat{p}(s^t,a^t_i,a^t_{-i}))}\Big[r_i(s^t,a^t_i,a^t_{-i})\Big] \\
&+\mathbb{E}_{s^t,a^t_i\sim\hat{p}(s^t)}\Big[\mathcal{H}(\rho_{-i}(a^t_{-i}|s^t,a^t_i))\Big]+\mathbb{E}_{s^t\sim\hat{p}(s^t)}\Big[\mathcal{H}(\pi_i(a^t_i|s^t))\Big]\,\mathrm{d}t,
\end{aligned} \tag{3.18}$$

where $\mathcal{H}$ is entropy term. In the recursive case, we note that we can rewrite the objective and have:

$$Q_i(s,a_i)=\log\int_{a_{-i}}\exp(Q_i(s,a_i,a_{-i}))\,\mathrm{d}a_{-i}. \tag{3.19}$$

which corresponds to a standard bellman backup with a soft maximization for the value function. choosing optimal opponent recursive reasoning policy:

$$\rho_{-i}(a_{-i}|s,a_i)=\exp(Q_i(s,a_i,a_{-i})-Q_i(s,a_i)). \tag{3.20}$$

Then we can have the objective function:

$$\begin{aligned}
J_i(\phi_{-i})=\int_{t=1}^{T}&\mathbb{E}_{(s^t,a^t_i,a^t_{-i})\sim\hat{p}(s^t,a^t_i,a^t_{-i})}[r_i(s^t,a^t_i,a^t_{-i}) \\
&+\mathcal{H}(\rho^{\phi_{-i}}_{-i}(a^t_{-i}|s^t,a^t_i))+\mathcal{H}(\pi^{\theta_i}_i(a^t_i|s^t))]\,\mathrm{d}t.
\end{aligned} \tag{3.21}$$

Then the gradient is then given by:

$$\begin{aligned}
&\nabla_{\phi_{-i}} J_i(\phi_{-i}) \\
&= \int_{t=1}^{T} \mathbb{E}_{(s^t, a_i^t, a_{-i}^t) \sim p(s^t, a_i^t, a_{-i}^t)} [\nabla_{\phi_{-i}} \log \rho_{-i}^{\phi_{-i}}(a_{-i}^t | s^t, a_i^t) (\int_{t'=t}^{T} r_i(s^{t'}, a_i^{t'}, a_{-i}^{t'}) \, \mathrm{d}t] \\
&\quad + \nabla_{\phi_{-i}} \int_{t=1}^{T} \mathbb{E}_{(s^t, a_i^t, a_{-i}^t) \sim p(s^t, a_i^t, a_{-i}^t)} [\mathcal{H}(\rho_{-i}^{\phi_{-i}}(a_{-i}^t | s^t, a_i^t)) + \mathcal{H}(\pi_i^{\theta_i}(a_i^t | s^t))] \, \mathrm{d}t.
\end{aligned} \tag{3.22}$$

The gradient of the entropy terms is given by:

$$\begin{aligned}
&\nabla_{\phi_{-i}} \mathcal{H}(\rho_{-i}^{\phi_{-i}}) \\
&= -\nabla_{\phi} \mathbb{E}_{(s^t, a_i^t) \sim p(s^t, a_i^t, a_{-i}^t)} [\mathbb{E}_{a_{-i}^t \sim \rho_{-i}^{\phi_{-i}}(a_{-i}^t | s^t, a_i^t)} [\log \rho_{-i}^{\phi_{-i}}(a_{-i}^t | s^t, a_i^t)]] \\
&= -\mathbb{E}_{(s^t, a_i^t, a_{-i}^t) \sim p(s^t, a_i^t, a_{-i}^t)} [\nabla_{\phi} \log \rho_{-i}^{\phi_{-i}}(a_{-i}^t | s^t, a_i^t)(1 + \log \rho_{-i}^{\phi_{-i}}(a_{-i}^t | s^t, a_i^t)].
\end{aligned} \tag{3.23}$$

We can do the same for $\nabla_{\phi_{-i}} \mathcal{H}(\pi_i^{\theta_i})$, and substitute these back:

$$\begin{aligned}
\nabla_{\phi_{-i}} J_i(\phi_{-i}) &= \int_{t=1}^{T} \mathbb{E}_{(s^t, a_i^t, a_{-i}^t) \sim p(s^t, a_i^t, a_{-i}^t)} [\nabla_{\phi_{-i}} \log \rho_{-i}^{\phi_{-i}}(a_{-i}^t | s^t, a_i^t) \\
&\quad (\int_{t'=t}^{T} r_i(s^{t'}, a_i^{t'}, a_{-i}^{t'}) - \log \rho_{-i}^{\phi_{-i}}(a_{-i}^{t'} | s^t, a_i^{t'}) - \log \pi_i^{\theta_i}(a_i^t | s^t) - 1) \, \mathrm{d}t] \, \mathrm{d}t.
\end{aligned} \tag{3.24}$$

The -1 comes from the derivative of the entropy terms, and replacing -1 with a state and self-action dependent baseline $b(s_{t'}, a_i^{t'})$ we can obtain the approximated gradient for ϕ:

$$
\begin{aligned}
\nabla_{\phi_{-i}} J_i(\phi_{-i}) = &\int_{t=1}^{T} \mathbb{E}_{(s^t,a_i^t,a_{-i}^t)\sim p(s^t,a_i^t,a_{-i}^t)}[\nabla_\phi \log \rho_{-i}^{\phi_{-i}}(a_{-i}^t|s^t,a_i^t) \\
&(\int_{t'=t}^{T} r_i(s^{t'},a_i^{t'},a_{-i}^{t'}) - \log \rho_{-i}^{\phi_{-i}}(a_{-i}^{t'}|s^{t'},a_i^{t'}) - \log \pi_i^{\theta_i}(a_i^{t'}|s^{t'}) - 1)\,\mathrm{d}t]\,\mathrm{d}t \\
\approx &\int_{t=1}^{T} \mathbb{E}_{(s^t,a_i^t,a_{-i}^t)\sim p(s^t,a_i^t,a_{-i}^t)}[\nabla_{\phi_{-i}} \log \rho_{-i}^{\phi_{-i}}(a_{-i}^t|s^t,a_i^t) \\
&(r_i(s^t,a_i^t,a_{-i}^t) - \underbrace{\log \pi_i^{\theta_i}(a_i^t|s^t)}_{Q_i^t(s^t,a_i^t)-V_i^t(s^t)} - \underbrace{\log \rho_{-i}^{\phi_{-i}}(a_{-i}^t|s^t,a_i^t)}_{Q_i^t(s^t,a_i^t,a_{-i}^t)-Q_i^t(s^t,a_i^t)} \\
&\underbrace{+ \int_{t'=t+1}^{T} r_i(s^{t'},a_i^{t'},a_{-i}^{t'}) - \log \rho_{-i}^{\phi_{-i}}(a_{-i}^{t'}|s^{t'},a_i^{t'}) - \log \pi_i^{\theta_i}(a_i^{t'}|s^{t'}))\,\mathrm{d}t']\,\mathrm{d}t}_{\approx Q_i^t(s^{t+1},a_i^{t+1},a_{-i}^{t+1})} \\
= &\int_{t=1}^{T} \mathbb{E}_{(s^t,a_i^t,a_{-i}^t)\sim p(s^t,a_i^t,a_{-i}^t)}[\nabla_\phi \log \rho_{-i}^{\phi_{-i}}(a_{-i}^t|s^t,a_i^t) \\
&(r_i(s^{t'},a_i^{t'},a_{-i}^{t'}) + Q_i^t(s^{t+1},a_i^{t+1},a_{-i}^{t+1}) - Q_i^t(s^t,a_i^t,a_{-i}^t) + \underbrace{V_i^t(s^t)}_{\text{ignore}})]\,\mathrm{d}t \\
= &\int_{t=1}^{T} \mathbb{E}_{(s^t,a_i^t,a_{-i}^t)\sim p(s^t,a_i^t,a_{-i}^t)}[(\nabla_{\phi_{-i}} Q_i^t(s^t,a_i^t,a_{-i}^t) - \nabla_{\phi_{-i}} Q_i^t(s^t,a_i^t)) \\
&(r_i(s^{t'},a_i^{t'},a_{-i}^{t'}) + Q_i^t(s^{t+1},a_i^{t+1},a_{-i}^{t+1}) - Q_i^t(s^t,a_i^t,a_{-i}^t) + \underbrace{V_i^t(s^t)}_{\text{ignore}})]\,\mathrm{d}t \\
= &\int_{t=1}^{T} \mathbb{E}_{(s^t,a_i^t,a_{-i}^t)\sim p(s^t,a_i^t,a_{-i}^t)}[(\nabla_{\phi_{-i}} Q_i^t(s^t,a_i^t,a_{-i}^t) - \nabla_{\phi_{-i}} Q_i^t(s^t,a_i^t)) \\
&(\hat{Q}_i^t(s^t,a_i^t,a_{-i}^t) - Q_i^t(s^t,a_i^t,a_{-i}^t))]\,\mathrm{d}t,
\end{aligned}
\tag{3.25}
$$

where $\hat{Q}_i^t(s^t,a_i^t,a_{-i}^t)$ is is an empirical estimate of the Q-value of the policy. ■

Theorem 3.1 states that the learning of $\rho_{-i}^{\phi_{-i}}(a_{-i}|s,a_i)$ can be further converted to minimizing the KL-divergence between the estimated policy $\rho_{-i}^{\phi_{-i}}$ and the *advantage* function: $D_{\mathrm{KL}}\left(\rho_{-i}^{\phi_{-i}}(a_{-i}|s,a_i)\|\exp(Q^i(s,a_i,a_{-i}) - Q_i(s,a_i))\right)$. We can obtain a solution to Equation 3.14 by maintaining two Q-functions, and then iteratively update them. We prove the convergence in the symmetric with only one equilibrium. This leads to a fixed-point iteration that resembles value iteration.

Theorem 3.2. *In a symmetric game with only one equilibrium, and the equilibrium meets one of the conditions: 1) the global optimum, i.e.* $\mathbb{E}_{\pi_*}[Q_i^t(s)] \geq \mathbb{E}_{\pi}[Q_i^t(s)]$*; 2)* *a saddle point, i.e.* $\mathbb{E}_{\pi_*}\left[Q_i^t(s)\right] \geq \mathbb{E}_{\pi_i}\mathbb{E}_{\pi_{-i}^*}\left[Q_i^t(s)\right]$ *or* $\mathbb{E}_{\pi_*}\left[Q_i^t(s)\right] \geq \mathbb{E}_{\pi_i^*}\mathbb{E}_{\pi_{-i}}\left[Q_i^t(s)\right]$*; where* Q_* *and* π_* *are the equilibrium value function and policy, respectively. The PR2 soft value iteration operator defined by:*

$$\begin{aligned}\mathcal{T}Q_i(s,a_i,a_{-i}) \triangleq \quad & r_i(s,a_i,a_{-i}) \\ & + \gamma\mathbb{E}_{s',a^{i\prime}\sim p_s,\pi_i}\left[\log\int_{a^{-i\prime}}\exp(Q_i(s',a^{i\prime},a^{-i\prime}))\,\mathrm{d}a^{-i\prime}\right],\end{aligned} \tag{3.26}$$

is a contraction mapping.

Proof. Based on Equation 3.13 & 3.14 in Theorem 3.1, we can have the PR2 soft value iteration rules shown as:

$$\begin{aligned}Q_i^\pi(s,a_i,a_{-i}) = \quad & r_i(s,a_i,a_{-i}) + \gamma\mathbb{E}_{s'\sim p_s}[\mathcal{H}(\pi_i(a_i|s)\pi_{-i}(a_{-i}|s,a_i)) \\ & + \mathbb{E}_{a_{-i'}\sim\pi_{-i}(\cdot|s',a_{i'})}[Q_i^\pi(s',a_{i'},a_{-i'})]] \\ = \quad & r_i(s,a_i,a_{-i}) + \gamma\mathbb{E}_{s'\sim p_s}[Q_i^\pi(s',a_{i'})].\end{aligned} \tag{3.27}$$

Correspondingly, we define the soft value iteration operator $\mathcal{T}$:

$$\mathcal{T}Q_i(s,a_i,a_{-i}) \triangleq r_i(s,a_i,a_{-i}) + \gamma\mathbb{E}_{s',a_{i'}\sim p_s,\pi_i}[\log\int_{a_{-i'}}\exp(Q_i(s',a_{i'},a_{-i'}))\,\mathrm{d}a_{-i'}]. \tag{3.28}$$

In a symmetric game with either one global equilibrium or saddle equilibrium, it has been shown by (Yang et al., 2018a) (see condition 1&2 in Theorem 1) that the payoff at the equilibrium point is unique. This validates applying the similar idea in proving the contraction mapping of soft-value iteration operator in the single agent case (see Lemma 1 in (Fox et al., 2016)). We include it here to stay self-contained. We first define a norm on Q-values as $\|Q_i^1 - Q_i^2\| \triangleq \max_{s,a_i,a_{-i}}|Q_i^1(s,a_i,a_{-i}) - Q_i^2(s,a_i,a_{-i})|$. Suppose $\varepsilon = \|Q_i^1 - Q_i^2\|$, then

$$\begin{aligned}\log\int_{a_{-i'}}\exp(Q_i^1(s',a_{i'},a_{-i'}))\,\mathrm{d}a_{-i'} &\leq \log\int_{a_{-i'}}\exp(Q_i^2(s',a_{i'},a_{-i'})+\varepsilon)\,\mathrm{d}a_{-i'} \\ &= \varepsilon + \log\int_{a_{-i'}}\exp(Q_i^2(s',a_{i'},a_{-i'}))\,\mathrm{d}a_{-i'}.\end{aligned} \tag{3.29}$$

Similarly,

$$\log \int_{a_{-i'}} \exp(Q_i^1(s', a_{i'}, a_{-i'}))\, \mathrm{d}a_{-i'} \leq -\varepsilon + \log \int_{a_{-i'}} \exp(Q_i^2(s', a_{i'}, a_{-i'}))\, \mathrm{d}a_{-i'}.$$

Therefore $\|\mathcal{T}Q_i^1 - \mathcal{T}Q_i^2\| \leq \gamma\varepsilon = \gamma\|Q_i^1 - Q_i^2\|$. ■

3.2.3 Sampling in Continuous Action Space

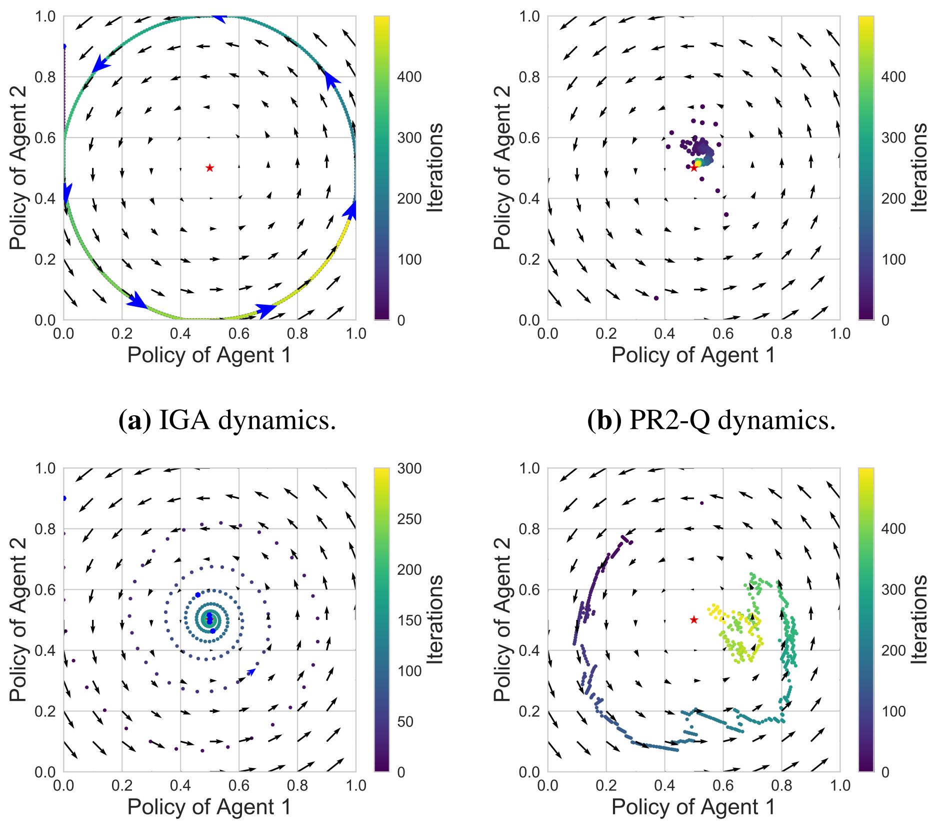
In dealing with the continuous action space, getting the actions from the opponent policy is challenging, as $\rho_{-i}^{\phi_{-i}}(a_{-i}|s, a_i) \sim \exp(Q^i(s, a_i, a_{-i}) - Q_i(s, a_i))$. In this chapter, we follow (Haarnoja et al., 2017) to adopt the amortized Stein Variational Gradient Descent (SVGD) (Liu and Wang, 2016; Wang and Liu, 2016) in sampling from the soft Q-function. Compared to MCMC, Amortized SVGD is a computationally-efficient way to estimate $\rho_{-i}^{\phi_{-i}}(a_{-i}|s, a_i)$. Thanks to SVGD, agent i is able to reason about potential consequences of opponent bavhaviors $\int_{a_{-i}} \pi_{-i}^{\theta_{-i}}(a_{-i}|s, a_i) Q_i(s, a_i, a_{-i})\, \mathrm{d}a_{-i}$, and finally find the corresponding best response.

3.2.4 Alternative Approach

In learning the opponent conditional policy $\rho_{-i}^{\phi_{-i}}(a_{-i}|s, a_i)$, one could also conduct variational inference directly on minimizing $D_{\mathrm{KL}}(\rho_{-i}^{\phi_{-i}}(a_{-i}|s, a_i)||\pi_{-i}^{\theta_{-i}}(a_{-i}|s, a_i))$. In such a way, it is equivalent to maximizing the evidence of lower bound, that is

$$\begin{aligned}\mathcal{L}(\theta_i, \phi_{-i}, a_i) = \ & \mathbb{E}_{a_{-i} \sim \rho_{-i}^{\phi_{-i}}(a_{-i}|s, a_i)}[\log \pi_i^{\theta_i}(a_i|s, a_{-i})] \\ & - D_{\mathrm{KL}}(\rho_{-i}^{\phi_{-i}}(a_{-i}|s, a_i)||\pi_{-i}^{\theta_{-i}}(a_{-i}|s)).\end{aligned}$$

However we believe this is not feasible. The main reason is that we have no information on either $\pi_{-i}^{\theta_{-i}}(a_{-i}|s)$ or $\pi_i^{\theta_i}(a_i|s, a_{-i})$; therefore, we have to construct two additional models to learn from experiences in an supervised way. Despite the added complexity, this approach introduces another two origins of approximation errors.



(a) IGA dynamics. **(b)** PR2-Q dynamics.

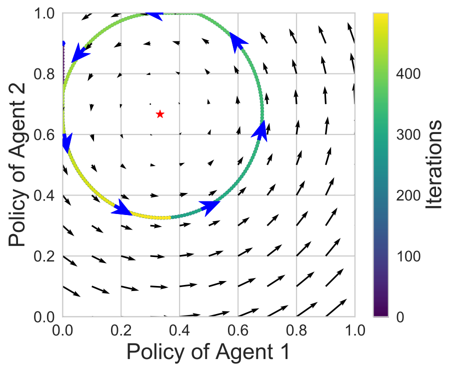
(c) IGA-PP dynamics. **(d)** WoLF-PHC dynamics.

Figure 3.3: Learning paths on the iterated matrix game 1.

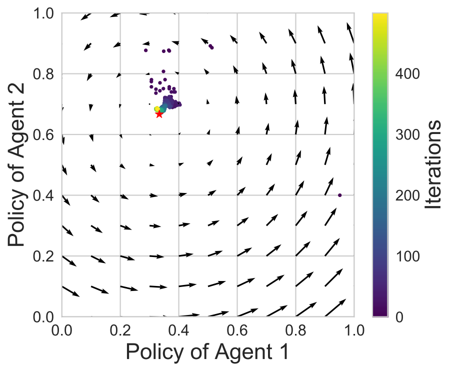
3.3 Experiments

We evaluate the performance of our algorithm on iterated matrix games, and differential games. Those games have a non-trivial equilibrium that requires certain levels of intelligent reasoning between agents. We compared our algorithm with a series of baselines. In the matrix games, we compare against IGA (Infinitesimal Gradient Ascent) (Singh et al., 2000), IGA-PP (Infinitesimal Gradient Ascent with Policy Prediction) (Zhang and Lesser, 2010), and WoLF-PHC (Win or Learn Fast Policy Hill-Climbing) (Bowling and Veloso, 2002). In IGA and IGA-PP, the policy learning rate was 0.01 and the short-term prediction rate for IGA-PP was 0.008. For WoLF-PHC, we used larger learning rates $\delta_l = 0.015, \delta_w = 0.03$ which makes algorithm converges within the smallest number of time steps.

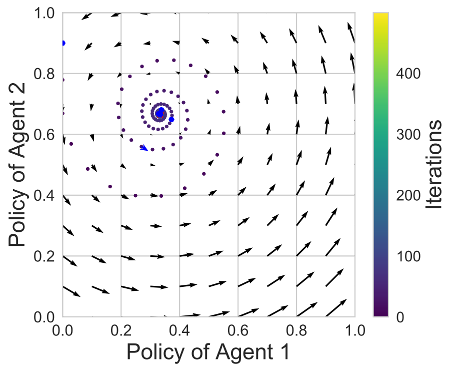
In the differential games, the baselines from multi-agent learning algorithms are MASQL (Multi-Agent Soft-Q) (Wei et al., 2018) and MADDPG (Lowe et al.,



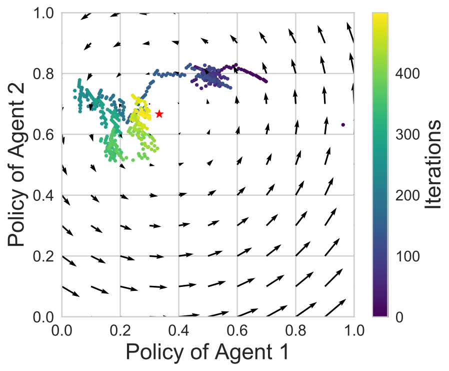
(a) IGA dynamics.



(b) PR2-Q dynamics.



(c) IGA-PP dynamics.

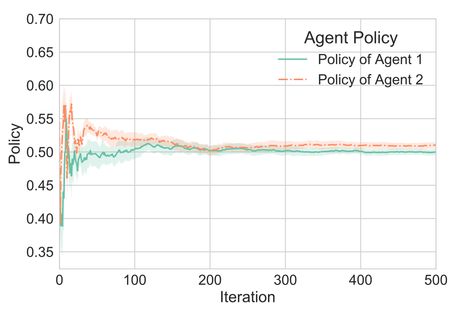


(d) WoLF-PHC dynamics.

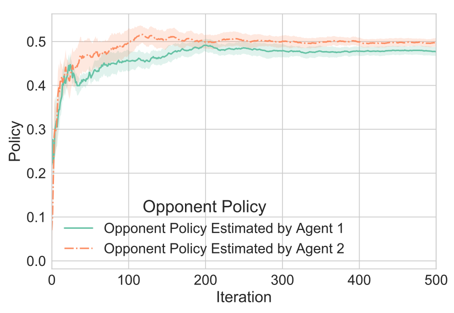
Figure 3.4: Learning paths on the iterated matrix game 2.

2017). We also including independent learning algorithms implemented through DDPG (Lillicrap et al., 2015). To compare against traditional method of opponent modeling, we include one baseline that is also based on DDPG but with one additional opponent model that is trained in an online and supervised way to learn the most recent opponent policy, which is then fed into the critic. Similar approach has been implemented by (Rabinowitz et al., 2018) in realizing machine theory of mind.

For the experiment settings, all the policies, opponent models and Q-functions are parameterized by the MLP with 2 hidden layers and 100 units each with the ReLU activation. The sampling network ξ for the $\rho_{-i}^{\phi_{-i}}$ in SGVD follows the standard normal distribution. In the iterated matrix game, we trained all the methods including the baselines for 500 iterations. In the differential game, we trained the agents for 350 iterations with 25 steps per iteration. For the actor-critic methods, we set the exploration noise to 0.1 in first 1000 steps, and the annealing parameters for PR2-AC

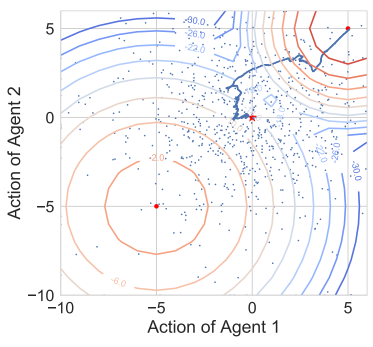


(a) PR2-Q Agent Policies.

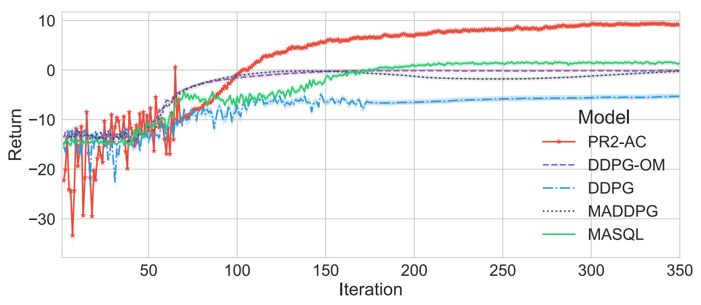


(b) PR2-Q Opponent Policies

Figure 3.5: Learned policies on the iterated matrix game 1.



(a) The learning path of PR2-AC vs. PR2-AC.



(b) The learning curves.

Figure 3.6: Max of Two Quadratic Game.

and MASQL are set to 0.5 to balance between the exploration and acting as the best response.

3.3.1 Iterated Matrix Game

Firstly, we conducted the experiment on two matrix games. The payoffs for the first matrix game are defined by:

$$\begin{array}{ll|c|c|}
 & & \multicolumn{2}{c}{\text{Agent 2}} \\
 & & \text{Action 1} & \text{Action 2} \\
\cline{3-4}
\multirow{2}{*}{\text{Agent 1}} & \text{Action 1} & 0,3 & 3,2 \\
\cline{3-4}
 & \text{Action 2} & 1,0 & 2,1 \\
\cline{3-4}
\end{array}.$$

Matrix Game 1

Similarly, the payoffs for the second matrix game are defined by:

$$
\begin{array}{cc|c|c|}
 & & \multicolumn{2}{c}{\text{Agent 2}} \\
 & & \text{Action 1} & \text{Action 2} \\
\cline{3-4}
\text{Agent 1} & \text{Action 1} & -1, 3 & 3, 1 \\
\cline{3-4}
 & \text{Action 2} & 0, -1 & 1, 0 \\
\cline{3-4}
\end{array}.
$$

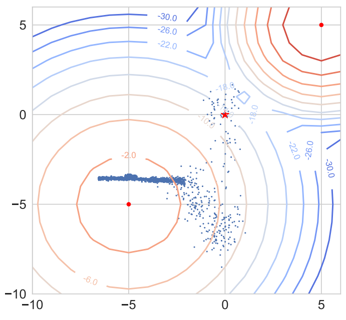
Matrix Game 2

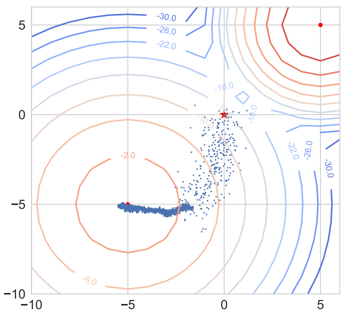
Both games have only one mixed Nash Equilibrium, which respectively are $(\frac{1}{2}, \frac{1}{2})$ and $(\frac{1}{3}, \frac{2}{3})$.

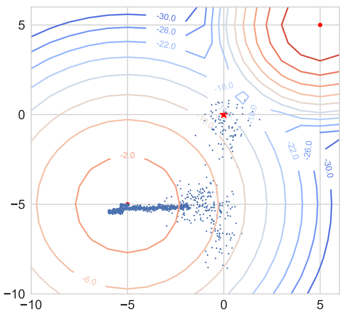
Above games have been extensively investigated in the studies on multi-agent learning (Bowling and Veloso, 2001a,b). One reason is that in solving the Nash Equilibrium for this game, simply taking simultaneous gradient steps on both agent's value functions will present the rotational behaviors on the gradient vector field; this leads to an endlessly iterative change of behaviors. Without considering the consequence of one agent's action on the other agent beforehand, it is challenging for both players to find the equilibrium. Interestingly, similar issue has also been reported in training the GANs (Goodfellow et al., 2014). Mescheder et al. (2017) has pointed out that the reason that game has a strong rotation gradient vector field is due to the imaginary part in the eigenvalue of IGA learning matrix.

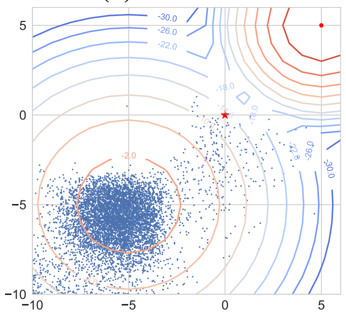
The results for two matrix games are shown in Figure 3.3 and 3.4. In both games, as expected, IGA fails to converge to the equilibrium but rotate around the equilibrium point. IGA-PP can reach the equilibrium, while IGA-PP assumes the knowledge about the other's value function. Similar to WoLF-PHC, our method can find the central equilibrium with a fully distributed fashion (see Figure 3.3b). The convergence of PR2-Q can also be confirmed by agents' policies in Figure 3.5a, and opponent's policy that is maintained by each agent in Figure 3.5b.

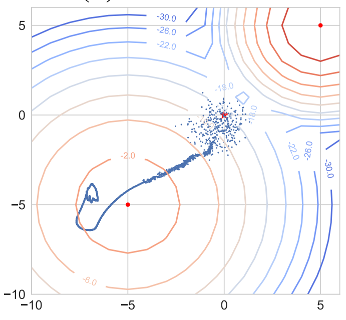
Besides, we also tested the computation time for all the methods using 2.3 GHz Quad-Core Intel Core i5, and recorded the average time in millisecond (ms) for taking 1000 steps. IGA and IGA-PP take about 5ms to process 1000 steps because they do not need to approximate the value functions. On the contrary, WoLF-PHC and PR2-Q need to approximate the value functions by themselves. Therefore, WoLF-PHC and PR2-Q require 271ms and 400ms to process 1000 steps, respectively. This result shows that although PR2-Q can find the equilibrium with minimal knowledge about the game, but it requires more complicated computation to achieve equivalent performance compared to the WoLF-PHC.

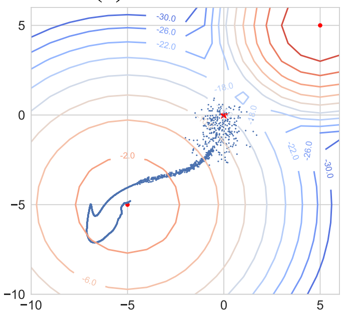

(a) DDPG / DDPG.

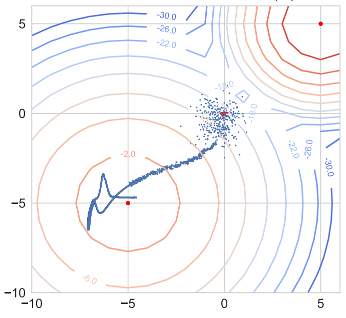

(b) DDPG- / DDPG-OM.


(c) MA- / MADDPG.


(d) MASQL / MASQL.


(e) PR2-AC / DDPG.


(f) PR2-AC / DDPG-OM.


(g) PR2-AC / MADDPG.

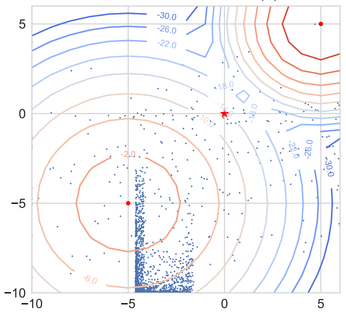

(h) PR2-AC / MASQL.

Figure 3.7: The learning path of Agent 1 (x-axis) vs. Agent 2 (y-axis). The self-play baselines and the PR2-AC control groups can only learn the pathologies to the suboptimal Nash equilibrium.

3.3.2 Differential Game

We adopt the same differential game, the Max of Two Quadratic Game, as (Panait et al., 2006; Wei et al., 2018). The agents have continuous action space of $[-10, 10]$. Each agent's reward depends on the joint action following the equations: $r^1\left(a^1, a^2\right) = r^2\left(a^1, a^2\right) = \max\left(f_1, f_2\right)$, where:

$$f_1 = 0.8 \times \left[-\left(\frac{a^1+5}{3}\right)^2 - \left(\frac{a^2+5}{3}\right)^2\right],$$
$$f_2 = 1.0 \times \left[-\left(\frac{a^1-5}{1}\right)^2 - \left(\frac{a^2-5}{1}\right)^2\right] + 10.$$

The task formulation is relatively simple, but it poses a great challenge to general gradient-based algorithms because gradient tends to points to the sub-optimal solution. The reward surface is shown in Figure 3.6a; there is a local maximum 0 at $(-5,-5)$ and a global maximum 10 at $(5,5)$, with a deep valley staying in the middle. If the agents' policies are initialized to $(0,0)$ (the red starred point) that lies within the basin of the left local maximum, the gradient based methods would tend to fail to find the global maximum equilibrium point due to the valley blocking the upper right area. The pathology of a suboptimal Nash Equilibrium in the joint space of actions being preferred over an optimal Nash Equilibrium is also called *relative over-generalization* (Wei and Luke, 2016).

We present the results in Figure 3.6b. PR2-AC shows superior performance that manages to converge to the global equilibrium, while all the other baselines fall into the local basin on the left, except that the MASQL has small chance to find the optimal point. On top of the convergence result, it is worth noting that as the temperature annealing is required for energy-based RL methods, the learning outcomes of MASQL are extremely sensitive to the way of annealing, i.e. when and how to anneal the temperature to a small value during training is non-trivial. However, our method does not need to tune the the annealing parameter precisely, because the each agent is acting the best response to the approximated conditional policy, which has considered all potential consequences of the opponent's response if this action was taken.

Interestingly, by comparing the learning path in Figure 3.6a against Figure 3.7(a-d) where the scattered dots are the exploration trails at the beginning, we can tell that if the PR2-AC model finds the peak point in joint action space, the agents can quickly go through the shortcut out of the local basin in a *clever* way, while

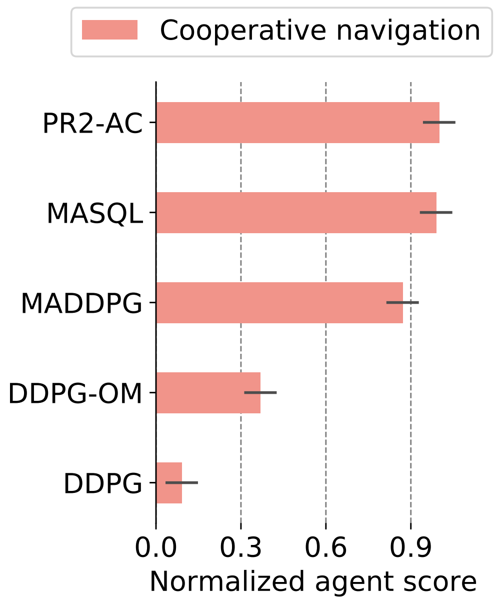


Figure 3.8: Performance of PR2-AC in the cooperative navigation game.

other algorithms cannot. This further justifies the effectiveness and benefits of conducting recursive reasoning with opponents. DDPG in Figure 3.7a and MASQL in Figure 3.7d even miss the local equilibrium; we believe this is because of the inborn defect from the independent learning methods, and the sensitivity to the annealing process respectively.

Apart from testing in the self-play setting, we also test the scenario when the opponent type is different. We pair PR2-AC with all four baseline algorithms in Figures 3.7(e-h). Similar result can be found, that is, algorithm that has the function of taking into account the opponents (i.e. DDPG-OM & MADDPG) can converge to the local equilibrium even though not global, while DDPG and MASQL completely fails due to the same reasons as in self-plays. Finally, we want to highlight the difference between PR2 methods and traditional OM, that is, PR2-Q/PR2-AC agent models how the opponents would believe about what it would behave, and then finds the best response to that belief, whereas OM agent tends to only model how the opponents behave based on the history. Such difference this seems to be a decisive factor in overcoming the rotational dynamics or the *relative over-generalization* issue.

3.3.3 Particle World Environments

We further test our method on the multi-state multi-player Particle World Environments (Lowe et al., 2017). We choose a cooperative game, *Cooperative Navigation*, in which three agents are collectively rewarded based on the proximity of any agent to three landmarks while avoiding collisions. The PR2 methods are compared to a series of centralized and decentralized MARL methods in Fig. 3.8 and all the methods are trained five random seeds. Even though PR2AC is a decentralized algorithm that does not have access to the exact opponent policies, it presents similar performance compared to the centralized method, including MADDPG and MASQL, Furthermore, PR2AC can outperform the other independent learners, such as DDPG and DDPG with opponent modeling.

3.4 Summary

Inspired by the recursive reasoning capability of human intelligent, in this chapter, we introduce a probabilistic recursive reasoning framework for multi-agent RL that follows "I believe that you believe that I believe". We adopt variational Bayes approaches to approximating the opponents' conditional policy, to which each agent then finds the best response to improve their own policy. The training and execution is full decentralized and the resulting algorithms, PR2-Q and PR2-AC, converge in the symmetric games with only one Nash equilibrium. Our results on both the matrix game and the differential game show the advantages of learning to reason about the opponents in a recursive manner.

Chapter 4

Generalized Recursive Reasoning

In this chapter, we bring the idea of mixing the cognitive hierarchy into the deep reinforcement learning (RL) framework, and develop algorithms that could both incorporate opponents' non-equilibrium behaviors in the games and converge to the NE faster. This is different from the work (Hu and Wellman, 2003; Wen et al., 2019; Yang et al., 2018a) that the convergence to NE requires agents to solve NE at each stage of the game.

Our proposed recursive reasoning models, modeling the opponents in a recursive manner, i.e. “I believe how you would believe about how I believe", can be regarded as a special type of opponent modeling (Albrecht and Stone, 2018). It was first introduced by game theorists (Harsanyi, 1962, 1967). The study on Theory of Mind (ToM) (Goldman et al., 2012; Rabinowitz et al., 2018), which refers to the ability of humans to infer and understand the beliefs, desires, and intentions of others, introduced the uncertainty of the agent's belief on opponents' behaviors, their payoffs, and the recursion depths. The Interactive POMDP (I-POMDP) (Gmytrasiewicz and Doshi, 2005) implements the idea of ToM to tackle the multi-agent reinforcement learning problem. It extends the partially observed MDP (Sondik, 1971) by introducing an extra space of models of other agents into the environment state, as such, an agent can build belief models about how it believes other agents know and believe. Despite the additional flexibility in the framework, I-POMDP has limitations in its solvability (Seuken and Zilberstein, 2008). Solving I-POMDP with N models considered in each of recursive level with K maximum level equals to solving $O(N^K)$ PODMPs. Such inherent complexity requires high precision on the approximation solution methods, including particle filtering (Doshi and Gmytrasiewicz, 2009), value iteration (Doshi and Perez, 2008), or policy iteration (Sonu and Doshi, 2015). (Ng et al., 2012) further extends the I-POMDP to incorporate the uncertainty of the transition and observation models of the environment, and develop the solution in an online way. In this work, we simplify the problems to a fully observable setting. We focus on investigating if a recursive model of other agents can consequently benefit our decision making. Therefore, this work's complexity is simpler than I-POMDP that we do not estimate the belief about both the physical state and other agents; instead, we provide a probabilistic framework to implement the recursive reason about opponents' policies in the MDP. We approximate the opponent's conditional policy through variational Bayes methods. The induced recursive reasoning algorithms are model-free and can practically be used as the replacement to other multi-agent RL algorithms such as MADDPG (Lowe et al., 2017).

Despite the recent success of applying deep reinforcement learning algorithms on both single-agent discrete (Mnih et al., 2015) and continuous (Lillicrap et al., 2015) control problems, independent learners ignores other agents in the multi-agent context; the theoretical convergence guarantee therefore breaks (Tuyls and Weiss, 2012). A modern framework is to maintain a centralised Q-network during training, e.g. MADDPG (Lowe et al., 2017) and multi-agent soft Q-learning (Wei et al., 2018); however, they require assumptions that both the actions of others and the parameters of the policy networks are fully observable (so as LOLA by (Foerster et al., 2018a)), let alone the centralized Q-network potentially prohibits the algorithms from scaling up. Besides, Haarnoja et al. (2017) enables a hierarchical structure for agent policy, but it is for single-agent case and needs layer-wise pre-training. Our approach employs decentralized training and decentralized execution, which requires less centralized information during the training and only needs to observe agents' historical actions.

In this chapter, we embed recursive reasoning into the RL framework via probabilistic inference. (Levine, 2018) built the connection between inferring the optimal policy in RL and the probabilistic inference on graphical models. Within this framework, (Haarnoja et al., 2017, 2018) designed the soft Q-learning and soft actor-critic methods. (Grau-Moya et al., 2018) and (Wei et al., 2018) further generalized the soft Q-learning to the multi-agent scenario; however, these methods optimize towards the optimal Q-function of the joint actions; by doing this, it implies that the opponents will always collaborate with the central agent to reach its best equilibrium. However, forming the incorrect beliefs about the opponents is detrimental in many scenarios (Albrecht and Stone, 2018) (e.g. the prisoner's dilemma).

4.1 Preliminaries: Multi-Agent Soft Learning

Interestingly, the maximum-entropy framework which can model the sub-optimal behaviors has also been explored in the RL domain through inference on graphical models Levine (2018); *soft* Q-learning Haarnoja et al. (2017) and actor-critic Haarnoja et al. (2018) methods were developed. Recently, *soft* learning has been further adapted into the context of MARL Tian et al. (2019); Wei et al. (2018).

Our GR2 method is built upon the framework of n-agent stochastic game (Shapley, 1953) using *soft* learning. Agents are self-interested to learn only the optimal policy in the game. The variable O_i represents how optimal agent i is at time t; it is proportional to the reward $P(O_i^t = 1|s^t, a_i^t) \propto \exp(r_i(s^t, a_i^t, a_{-i}^t))$ (Levine, 2018). In other words, O_i can describe how likely the trajectory τ_i will be observed, e.g. if agents were egocentric, altruism actions are less likely to be observed. The optimal action for agent i is then stated as $p(a_i^t|s^t, a_{-i}^t, O_i^{t:T} = 1)$.

The *best response* to the opponent policy π_{-i} is the policy π_i^* s.t. $V_i(s; \pi_i^*, \pi_{-i}) \geq V_i(s; \pi_i, \pi_{-i})$ for all valid π_i. When all the agents act in their best response, $(\pi_1^*, \ldots, \pi_*^n)$ forms a Nash Equilibrium (Nash et al., 1950). Soft Q-learning, also called maximum entropy RL, is the outcome if one applies variational inference to solve the optimal policy on the graphical model of RL (Levine, 2018). Compared to the normal Q-learning, it has one additional entropy term in the objective. When the soft Q-learning is adapted into the multi-agent case, the entropy

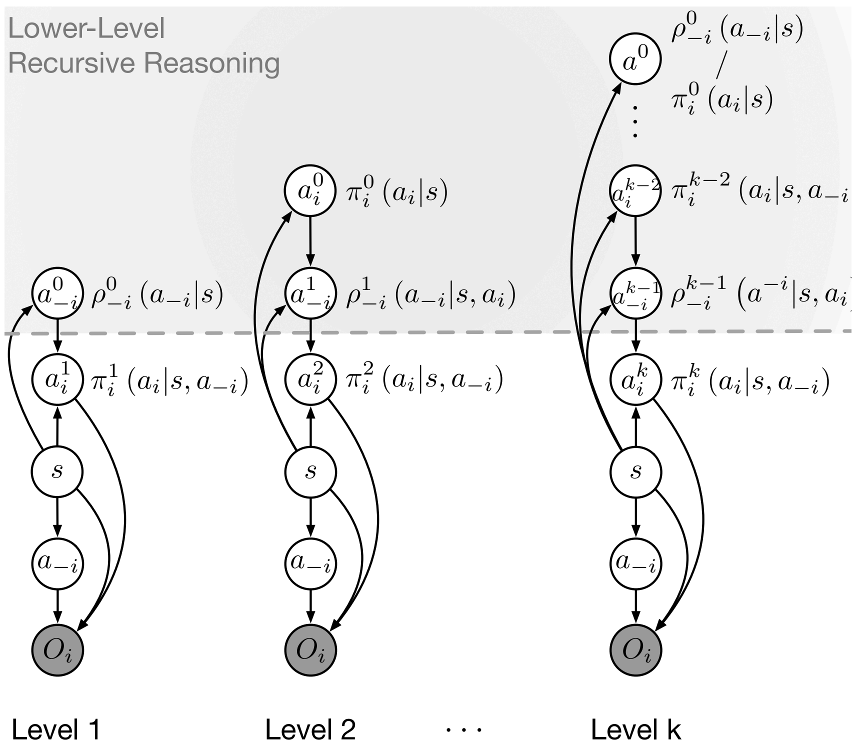


Figure 4.1: Graphical model of the level-k recursive reasoning. Note that the suffix a_* here stands for the level of thinking not the time step. The unobservable opponent policies are approximated by ρ_{-i}. The omitted level-0 model considers opponents fully randomized. Agent i rolls out the recursive reasoning about opponents in its mind (grey area). In the recursion, agents with higher-level beliefs take the best response to the lower-level thinkers' actions. Higher-level models would conduct all the computations that the lower-level models have done, e.g. level-2 contains level-1.

term $\mathcal{H}$ is instead applied on the joint policy $\pi(a_i^t, a_{-i}^t|s^t)$ (Grau-Moya et al., 2018). The soft value function therefore reads as follows:

$$V_i(s) = \mathbb{E}[\sum_{t=0}^{\infty} r_i(s^t, a_i^t, a_{-i}^t) + \mathcal{H}(\pi(a_i^t, a_{-i}^t|s^t))]. \tag{4.1}$$

For policy evaluation, a modified Bellman operator $\mathcal{T}^\pi$ can be applied,

$$\mathcal{T}^\pi Q_i(s^t, a_i^t, a_{-i}^t) \triangleq r_i(s^t, a_i^t, a_{-i}^t) + \gamma \mathbb{E}_{s_{t+1},p}[V_i(s_{t+1})],$$

where

$$V_i(s^t) = \mathbb{E}_{a_i, a_{-i} \sim \pi}[Q_i(s, a_i, a_{-i}) - \log \pi(a_i, a_{-i}|s)].$$

Compared to the max operator in the Bellman equation for $V_i(s)$ in the case

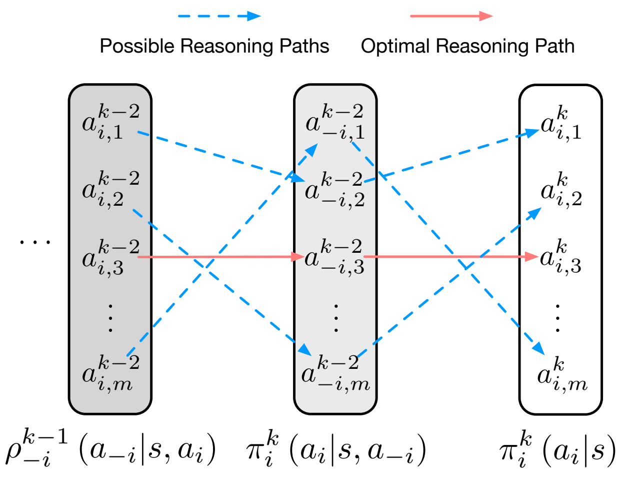


Figure 4.2: Reasoning paths of the level-k policy.

without the entropy term, it is *soft* because $V_i(s^t) = \log \int \exp(Q_i(s^t, a^t))\, \mathrm{d}a^t \approx \max_{a^t} Q(s^t, a^t)$. Correspondingly, the independent policy $\pi_i(a_i|s)$ can be learned by minimizing $D_{\mathrm{KL}}(\pi_i(a_i|s)||\exp(Q_i(s, a_i)))$, which leads to the Soft RL (Haarnoja et al., 2017, 2018).

However, policy improvement becomes tricky in the multi-agent soft Q-learning because the Q-function guides the improvement direction for the joint policy rather than for each individual agent. One naive approach is to discard the opponents actions sampled from the joint policy $\pi(a_i^t, a_{-i}^t|s^t)$ and take the marginalized output (Wei et al., 2018).

Probabilistic Recursive Reasoning PR2 (Wen et al., 2019) adopted the so-called *correlated factorization* on the joint policy, i.e. $\pi(a_i, a_{-i}|s) = \pi_i(a_i|s)\pi_{-i}(a_{-i}|s, a_i)$, where $\pi_{-i}(a_{-i}|s, a_i)$ represents the level-1 reasoning process of the opponent's consideration of the action from agent i. PR2 approximates the actual opponent conditional policy π_{-i} via a best-fit model $\rho_{-i}^{\phi_{-i}}$ from a family of distributions parameterized by ϕ_{-i}. Through variational inference, the optimal opponent conditional policy in PR2 Q-learning is described to be

$$\rho_{-i}^{\phi_{-i}}(a_{-i}|s, a_i) \propto \exp(Q_i^{\pi^\theta}(s, a_i, a_{-i}) - Q_i^{\pi^\theta}(s, a_i)). \tag{4.2}$$

Based on the optimal opponent model given by Equation 4.2, agent i can learn the best response policy $\pi_i(a_i|s)$ according to the optimal Q-function: $Q_i(s,a_i) = \int_{a_{-i}} \rho_{-i}^{\phi_{-i}}(a_{-i}|s,a_i)Q_i(s,a_i,a_{-i})\,\mathrm{d}a_{-i}$.

Together with the Bellman equation for policy evaluation on Equation 4.1, PR2 is proved to converge to the Nash equilibrium for the symmetric games with only one Nash equilibrium.

4.2 Generalized Recursive Reasoning

Our goal is to develop more generalized and flexible recursive reasoning into MARL. In this section, we first introduce the protocol of level-k recursive reasoning (GR2-L), and then propose a mixture model that takes into account agents with different hierarchical levels of policies (GR2-M).

4.2.1 Higher Level Recursive Reasoning

We start by constructing a level-k recursive reasoning process, naming it *GR2-L*. A level-k policy for agent i is the best response to the opponents' policy at level $k-1$ (see Figure 4.1). We assume π_{-i} is not directly observable, and agents have to approximate it by ρ_{-i}. The models on a_i and a_{-i} can be regarded as latent variable model, with the marginal policies $\pi_i(a_i|s)$, $\rho_{-i}(a_{-i}|s)$ serving as the *prior*, and the conditional policies $\pi_i(a_i|s,a_{-i})$, $\rho_{-i}(a_{-i}|s,a_i)$ as the *posterior*. Actions can be sampled in a sequential manner. The level-k policy is therefore constructed by integrating over best responses from all lower-level policies,

$$\pi_i^k(a_i^k|s) \propto \int_{a_{-i}^{k-1}} \mathrm{d}a_{-i}^{k-1} \Big\{ \pi_i^k(a_i^k|s,a_{-i}^{k-1}) \int_{a_i^{k-2}} \mathrm{d}a_i^{k-2} \Big[\rho_{-i}^{k-1}(a_{-i}^{k-1}|s,a_i^{k-2})\pi_i^{k-2}(a_i^{k-2}|s)\Big] \Big\}. \tag{4.3}$$

From agent i's perspective, it believes that the opponents will take the *best response* to its own (fictitious) action a_i^{k-2} that are two levels below: $\rho_{-i}^{k-1}(a_{-i}^{k-1}|s) = \int \rho_{-i}^{k-1}(a_{-i}^{k-1}|s,a_i^{k-2})\pi_i^{k-2}(a_i^{k-2}|s)\,\mathrm{d}a_i^{k-2}$, even though the beliefs, ρ_{-i}, that agent i holds for its opponents might be inaccurate yet. π_i^{k-2} can be further expanded recursively until meeting π^0 that is assumed uniformly distributed. For brevity, we mostly omit the subscript level k in actions a from now on while acknowledging that the actions in the lower level are imagined (fictitiously taken). As shown in Figure 4.1 and Figure 4.2, considering the computational feasibility, we assume that each of the agents adopts the same functional form for its policy across the reasoning process in different depths. It means we can reuse the policy parameters to produce a higher level π_i^k via best response to the lower level ρ_{-i}^{k-1}, and same procedure also works for ρ_i^k.

4.2.2 Mixture of Hierarchy Recursive Reasoning

While the level-k policy assumes agents best respond to the level $k-1$ opponents , we can further generalize the model to let agent best respond to a mixture of agents that are distributed over the lower hierarchies. We name it *GR2-M*.

We *assume* that the level-k agents have an accurate guess about the relative proportion of agents who are doing less thinking than them. This specification implies that even though the agent could maintain an inaccurate belief on the probability distribution of the agents' recursive depths, as k increases, the deviation between agent's belief and the actual distribution profile of k will shrink. This further suggests that when k is very large, there is no benefit for level-k thinkers to reason even harder, as they will have the same beliefs, subsequently make the same decisions as agents at $k+1$. Since more steps of computations are required as the k grows, it is reasonable to make such assumption so that with the increasing k, fewer agents are willing to do the reasoning beyond k. Such assumption can also be justified by the limited amount of working memory in the strategic thinking on human beings (Devetag and Warglien, 2003).

In order to meet this assumption, we model the distribution of recursive depth k by the Poisson distribution $f(k) = \frac{e^{-\lambda}\lambda^k}{k!}$, where the mean, also the variance, of the distribution is λ. A nice property of Poisson is that $f(k)/f(k-1)$ is inversely proportional to k, which satisfies our previous assumption. Interestingly, a study on humans suggests that on average people tend to act with $\lambda \approx 1.5$ (Camerer et al., 2004).

With the mixture of hierarchy modeled by Poisson, we can now mix all k levels'

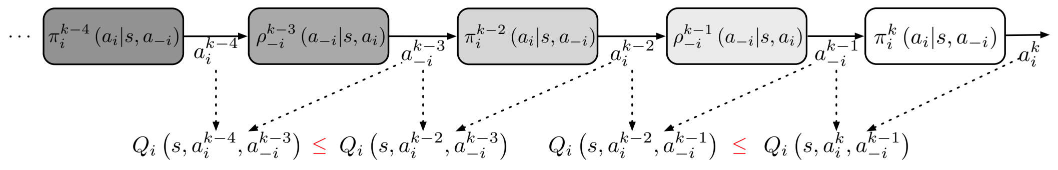

$\cdots$
$\pi_i^{k-4}(a_i|s,a_{-i})$
a_i^{k-4}
$\rho_{-i}^{k-3}(a_{-i}|s,a_i)$
a_{-i}^{k-3}
$\pi_i^{k-2}(a_i|s,a_{-i})$
a_i^{k-2}
$\rho_{-i}^{k-1}(a_{-i}|s,a_i)$
a_{-i}^{k-1}
$\pi_i^{k}(a_i|s,a_{-i})$
a_i^{k}
$Q_i(s,a_i^{k-4},a_{-i}^{k-3}) \leq Q_i(s,a_i^{k-2},a_{-i}^{k-3})$
$Q_i(s,a_i^{k-2},a_{-i}^{k-1}) \leq Q_i(s,a_i^{k},a_{-i}^{k-1})$

Figure 4.3: Inter-level policy improvement. Higher-order strategies should weakly dominant lower-order strategies, e.g. with the opponent behaviors a_{-i}^{k-1} unchanged, a_i^k should perform at least as good as $a_i^{k-2}, a_i^{k-4}, \ldots, a_i^1$.

thinkings into one perception, and build each agent's belief on its opponents by

$$\pi_{i,\lambda}^k := \frac{e^{-\lambda}}{Z}\left(\frac{\lambda^0}{0!}\pi_i^0 + \frac{\lambda^1}{1!}\pi_i^1 \cdots \frac{\lambda^k}{k!}\pi_i^k\right), \tag{4.4}$$

where Z is a normalization term. In the reinforcement learning framework to be shown later, λ can be set as a hyper-parameter, similar to the TD-λ (Tesauro, 1995). Note that GR2-L is in fact a special case of GR2-M. When the depth k follows Poisson distribution, we have $f(k-1)/f(k-2) = \lambda/(k-1)$; the model will put heavy weights on the $k-1$ level if $\lambda \gg k$; that is to say, a level-k agent will act as if all the opponents are reasoning at level $k-1$.

4.2.3 Theoretical Convergence

Recursive reasoning is essentially to let each agent take the best response to its opponents with the opponents' actions being the best response to the agent's best-responded action. A natural question to ask is then under the GR2 setting, does the equilibrium ever exist? If so, will GR2 ever converge?

We show that on two-player games, the learning dynamics of GR2 is asymptotically stable in the sense of Lyapunov. In addition, the dynamic game induced by GR2 has Perfect Bayesian Equilibrium.

Theorem 4.1. *In two-player two-action games with one mixed strategy equilibrium, when the opponent model's step size is sufficiently small, the learning dynamics of GR2 methods to the equilibrium is asymptotic stable in the sense of Lyapunov (see Definition 4.2).*

Proof. We start by defining the matrix game that a mixed-strategy equilibrium exists, and they we show that on such game level-0 independent learner through iterated gradient ascent will not converge, and finally derive why the level-k methods would converge in this case. Our tool is Lyapunov function and its stability analysis.

Lyapunov function (Lyapunov, 1992) is used to verify the stability of a dynamical system in control theory, here we apply it in convergence proof for level-k methods. It is defined as following:

Definition 4.1 (Lyapunov Function)**.** *Give a function $F(x, y)$ be continuously differentiable in a neighborhood σ of the origin. The function $F(x, y)$ is called the Lyapunov function for an autonomous system if that satisfies the following properties:*

1. *(nonnegative) $F(x, y) > 0$ for all $(x, y) \in \sigma \backslash (0, 0)$;*
2. *(zero at fixed-point) $F(0, 0) = 0$;*
3. *(decreasing) $\frac{\mathrm{d}F}{\mathrm{d}t} \le 0$ for all $(x, y) \in \sigma$.*

Definition 4.2 (Lyapunov Asymptotic Stability)**.** *For an autonomous system, if there is a Lyapunov function $F(x, y)$ with a negative definite derivative $\frac{\mathrm{d}F}{\mathrm{d}t} < 0$ (strictly negative, negative definite LaSalle's invariance principle (Haddad and Chellaboina, 2011)) for all $(x, y) \in \sigma \backslash (0, 0)$, then the equilibrium point $(x, y) = (0, 0)$ of the system is asymptotically stable (Haddad and Chellaboina, 2011; Marquez, 2003).*

Single State Game

Given a two-player, two-action matrix game, which is a single-state stage game, we have the payoff matrices for row player and column player as follows:

$$\mathbf{R}_r = \begin{bmatrix} r_{11} & r_{12} \\ r_{21} & r_{22} \end{bmatrix} \quad \text{and} \quad \mathbf{R}_c = \begin{bmatrix} c_{11} & c_{12} \\ c_{21} & c_{22} \end{bmatrix}.$$

Each player selects an action from the action space $\{1, 2\}$ which determines the payoffs to the players. If the row player chooses action i and the player 2 chooses action j, then the row player and column player receive the rewards r_{ij} and c_{ij} respectively. We use $\alpha \in [0, 1]$ to represent the strategy for row player, where α corresponds to the probability of player 1 selecting the first action (action 1), and $1 - \alpha$ is the probability of choosing the second action (action 2). Similarly, we use

β to be the strategy for column player. With a joint strategy (α, β), the expected payoffs of players are:

$$V_r(\alpha, \beta) = \alpha\beta r_{11} + \alpha(1-\beta)r_{12} + (1-\alpha)\beta r_{21} + (1-\alpha)(1-\beta)r_{22},$$

$$V_c(\alpha, \beta) = \alpha\beta c_{11} + \alpha(1-\beta)c_{12} + (1-\alpha)\beta c_{21} + (1-\alpha)(1-\beta)c_{22}.$$

One crucial aspect to the learning dynamics analysis are the points of zero-gradient in the constrained dynamics, which they show to correspond to the equilibria which is called the center and denoted (α^*, β^*). This point can be found mathematically $(\alpha^*, \beta^*) = (\frac{-b_c}{u_c}, \frac{-b_r}{u_r})$.

Here we are more interested in the case that there exists a mixed strategy equilibrium, i.e., the location of the equilibrium point (α^*, β^*) is in the interior of the unit square, equivalently, it means $u_r u_c < 0$. In other cases where the Nash strategy on the boundary of the unit square (Bowling and Veloso, 2001a; Marquez, 2003), we are not going to discuss in this proof.

Learning in Level-0 Gradient Ascent

One common level-0 policy is Infinitesimal Gradient Ascent (IGA), which assumes independent learners and is a level-0 method, a player increases its expected payoff by moving its strategy in the direction of the current gradient with fixed step size. The gradient is then computed as the partial derivative of the agent's expected payoff with respect to its strategy, we then have the policies dynamic partial differential equations:

$$\partial V_r(\alpha, \beta)/\partial\alpha = u_r\beta + b_r, \quad \partial V_c(\alpha, \beta)/\partial\beta = u_c\alpha + b_c,$$

where $u_r = r_{11} - r_{12} - r_{21} + r_{22}$, $b_r = r_{12} - r_{22}$, $u_c = c_{11} - c_{12} - c_{21} + c_{22}$, and $b_c = c_{21} - c_{22}$. In the gradient ascent algorithm, a player will adjust its strategy after each iteration so as to increase its expected payoffs. This means the player will move their strategy in the direction of the current gradient with some step size. Then we can have dynamics are defined by the differential equation at time t:

$$\left[\begin{array}{c}\partial\alpha/\partial t\\ \partial\beta/\partial t\end{array}\right] = \underbrace{\left[\begin{array}{cc}0 & u_r\\ u_c & 0\end{array}\right]}_{U}\left[\begin{array}{c}\alpha\\ \beta\end{array}\right] + \left[\begin{array}{c}b_r\\ b_c\end{array}\right].$$

By defining multiplicative matrix term U above with off-diagonal values u_r and u_c, we can classify the dynamics of the system based on properties of U. As we mentioned, we are interested in the case that the game has just one mixed center strategy equilibrium point (not saddle point) that in the interior of the unit square, which means U has purely imaginary eigenvalues and $u_r u_c < 0$ (Zhang and Lesser, 2010). Consider the quadratic Lyapunov function which is continuously differentiable and $F(0,0) = 0$:

$$F(x, y) = 1/2(u_c x^2 - u_r y^2),$$

where we suppose $u_c > 0, u_r < 0$ (we can have identity case when $u_c < 0, u^r > 0$ by switching the sign of the function). Its derivatives along the trajectories by setting $x = \alpha - \alpha^*$ and $y = \beta - \beta^*$ to move the the equilibrium point to origin can be calculated as:

$$\frac{\mathrm{d}F}{\mathrm{d}t} = \frac{\partial F}{\partial x}\frac{\mathrm{d}x}{\mathrm{d}t} + \frac{\partial F}{\partial y}\frac{\mathrm{d}y}{\mathrm{d}t} = xy(u_r u_c - u_r u_c) = 0,$$

where the derivative of the Lyapunov function is identically zero. Hence, the condition of asymptotic stability is not satisfied (Marquez, 2003; Taylor et al., 2018) and the IGA level-0 dynamics is unstable. There are some IGA based methods (WoLF-IGA, WPL etc. (Abdallah and Lesser, 2008; Bowling and Veloso, 2002)) with varying learning step, which change the U to $\left[\begin{array}{cc}0 & l_r(t)u_r\\ l_c(t)u_c & 0\end{array}\right]$. The time dependent learning steps $l_r(t)$ and $l_c(t)$ are chose to force the dynamics would converge. Note that diagonal elements in U are still zero, which means a player's personal influences to the system dynamics are not reflected on its policy adjustment.

Learning in Level-k Gradient Ascent

Consider a level-1 gradient ascent, where agent learns in term of $\pi_r(\alpha)\pi_c^1(\beta|\alpha)$, the gradient is computed as the partial derivative of the agent's expected payoff after considering the opponent will have level-1 prediction to its current strategy. We then have the level-1 policies dynamic partial differential equations:

$$\partial V_r(\alpha, \beta_1)/\partial\alpha = u_r(\beta + \zeta\partial_\alpha V_r(\alpha, \beta))) + b_r$$
$$\partial V_c(\alpha_1, \beta)/\partial\beta, = u_c(\alpha + \zeta\partial_\beta V_c(\alpha, \beta)) + b_c,$$

where ζ is short-term prediction of the opponent's strategy. Its corresponding level-1 dynamic partial differential equations:

$$[\begin{matrix}\partial\alpha/\partial t \\ \partial\beta/\partial t\end{matrix}] = \underbrace{[\begin{matrix}\zeta u_r u_c & u_r \\ u^2 & \zeta u_r u_c\end{matrix}]}_{U}[\begin{matrix}\alpha \\ \beta\end{matrix}] + [\begin{matrix}\zeta u_r b_c + b_r \\ \zeta u_c b_r + b_c\end{matrix}].$$

Apply the same quadratic Lyapunov function: $F(x, y) = 1/2(u_c x^2 - u_r y^2)$, where $u_c > 0, u_r < 0$, and its derivatives along the trajectories by setting $x = \alpha - \alpha^*$ and $y = \beta - \beta^*$ to move the coordinates of equilibrium point to origin:

$$\frac{\mathrm{d}F}{\mathrm{d}t} = \zeta u_r u_c(u_c x^2 - u_r y^2) + xy(u_r u_c - u_r u_c) = \zeta u_r u_c(u_c x^2 - u_r y^2),$$

where the conditions of asymptotic stability is satisfied due to $u_r u_c < 0, u_c > 0$ and $u_r < 0$, and it indicates the derivative $\frac{\mathrm{d}F}{\mathrm{d}t} < 0$. In addition, unlike the level-0's case, we can find that the diagonal of U in this case is non-zero, it measures the mutual influences between players after level-1 looks ahead and helps the player to update it's policy to a better position.

This conclusion can be easily extended and proved in level-k gradient ascent policy ($k > 1$) . In level-k gradient ascent policy, we can have the derivatives of same quadratic Lyapunov function in level-2 dynamics:

$$\begin{aligned}\frac{\mathrm{d}F}{\mathrm{d}t} &= \zeta u_r u_c(u_c x^2 - u_r y^2) + xy(1 + \zeta^2 u_r u_c)(u_r u_c - u_r u_c) \\ &= \zeta u_r u_c(u_c x^2 - u_r y^2),\end{aligned}$$

and level-3 dynamics:

$$\frac{\mathrm{d}F}{\mathrm{d}t} = \zeta u_r u_c (2 + \zeta^2 u_r u_c)(u_c x^2 - u_r y^2).$$

Repeat the above procedures, we can easily write the general derivatives of quadratic Lyapunov function in level-k dynamics:

$$\frac{\mathrm{d}F}{\mathrm{d}t} = \zeta u_r u_c (k - 1 + \cdots + \zeta^{k-1}(u_r u_c)^{k-2})(u_c x^2 - u_r y^2),$$

where $k \geq 3$. These level-k policies still owns the asymptotic stability property when ζ^2 is sufficiently small (which is trivial to meet in practice) to satisfy $k - 1 + \cdots + \zeta^{k-1}(u_r u_c)^{k-2} > 0$, which meets the asymptotic stability conditions, therefore coverges. ■

Theorem 4.2. *GR2 strategies extend a norm-form game into extensive-form game, and there exists a Perfect Bayesian Equilibrium in that game.*

Proof. Consider an extensive game, which is extended from a normal form game by level-k strategies, with perfect information and recall played by two players $(i, -i)$: $(\pi_i, \pi_{-i}, u_i, u_{-i}, \Lambda)$, where $\pi_{i/-i}$ and $u_{i/-i}$ are strategy pairs and payoff functions for player $i, -i$ correspondingly. Λ denotes a level-k reasoning path/tree. An intermediate reasoning action/node in the level-k reasoning procedure is denoted by h^t. The set of the intermediate reasoning actions at which player i chooses to move is denoted H_i (a.k.a information set). Let $\pi_{i/-i}^{\tilde{k}}$ denote the strategies of a level-$\tilde{k}$ player and $\tilde{k} \in \{0, 1, 2 \cdots k\}$. A level-$k$ player holds a prior belief that the opponent is a level-$\tilde{k}$ player with probability $\lambda_{\tilde{k}}$, where $\lambda_{\tilde{k}} \in [0, 1]$ and $\sum_{\tilde{k}=0}^{k} \lambda_{\tilde{k}} = 1$. We denote the belief that the opponent is a level-$\tilde{k}$ player as $p_i^{\tilde{k}}(h^t)$. In equilibrium, a level-k player chooses an optimal strategy according to her belief at every decision node, which implies choice is sequentially rational as following defined:

Definition 4.3. *(Sequential Rationality). A strategy pair $\{\pi_i^*, \pi_{-i}^*\}$ is sequentially rational with respect to the belief pair $\{p_i, p_{-i}\}$ if for both $i, -i$, all strategy pairs*

$\{\pi_i, \pi_{-i}\}$ *and all nodes* $h_i^t \in H_i$*:*

$$\sum_{\tilde{k}=0}^{k} \lambda_{\tilde{k}} p_i^{\tilde{k}}(h_i^t) u_i(\pi_i^*, \pi_{-i}^* | h_i^t) \geq \sum_{\tilde{k}=0}^{k} \lambda_{\tilde{k}} p_i^{\tilde{k}}(h_i^t) u_i(\pi_i, \pi_{-i}^* | h_i^t),$$

Based on Definition 4.3, we have the strategy π_i is **sequentially rational** given p_i. It means strategy of player i is optimal in the part of the game that follows given the strategy profile and her belief about the history in the information set that has occurred.

In addition, we also require the beliefs of an level-k player are consistent. Let $p_i(h^t|\pi_i, \pi_{-i})$ denote the probability that reasoning action h^t is reached according to the strategy pair, $\{\pi_i, \pi_{-i}\}$. Then we have the consistency definition:

Definition 4.4. *(Consistency). The belief pair* $\{\rho_{-i}^*, \rho_i^*\}$ *is consistent with the subjective prior* $\lambda_{\tilde{k}}$*, and the strategy pair* $\{\pi_i, \pi_{-i}\}$ *if and only if for* $i, -i$ *and all nodes* $h_i^t \in H_i$*:*

$$p_i^{\tilde{k},*}(h_i^t) = \frac{\lambda_{\tilde{k}} p_i^{\tilde{k}}(h_i^t|\pi_i, \pi_{-i}^0)}{\sum_{\hat{k}=0}^{k} \lambda^{\hat{k}} p_i^{\hat{k}}(h_i^t|\pi_i, \pi_{-i})},$$

where there is at least one $\hat{k} \in \{0, 1, 2 \cdots, k\}$ *and* $p_i^{\hat{k}}(h_i^t|\pi_i, \pi_{-i}) > 0$*.*

The belief p_i is **consistent** given π_i, π_{-i} indicates that for every intermediate reasoning actions reached with positive probability given the strategy profile π_i, π_{-i} , the probability assigned to each history in the reasoning path by the belief system p_i is given by Bayes' rule. In summary, sequential rationality implies each player's strategy optimal at the beginning of the game given others' strategies and beliefs (Levin and Zhang, 2019). Consistency ensures correctness of the beliefs.

Although the game itself has perfect information, the belief structure in our level-k thinking makes our solution concept an analogy of a Perfect Bayesian Equilibrium. Based on above two definitions, we have the existence of Perfect Bayesian Equilibrium in level-k thinking game.

Proposition. *For any* $\lambda_{\tilde{k}}$*, where* $\lambda_{\tilde{k}} \in [0, 1]$ *and* $\sum_{\tilde{k}=0}^{k} \lambda_{\tilde{k}} = 1$*, there is a Perfect Bayesian Equilibrium exists.*

Now, consider an extensive game of incomplete information,

$$(\pi_i, \pi_{-i}, u_i, u_{-i}, p_i, p_{-i}, \lambda^k, \Lambda),$$

where λ^k denotes the possible levels/types for agents, which can be arbitrary level-k player. Then, according to Kreps and Wilson (1982), for every finite extensive form game, there exists at least one sequential equilibrium should satisfy Definition. 4.3 and 4.4 for sequential rationality and consistency, and the details proof as following:

We use $E_i(\pi, p, \lambda^k, h_i) = \sum_{\tilde{k}=0}^{k} \lambda_{\tilde{k}} p_i^{\tilde{k}}(h_i^t) u_i(\pi_i, \pi_{-i}|h_i^t)$ as expected payoff for player i, for every player i and each reasoning path h_i^t. Choose a large integer $m(m > 0)$ and consider the sequence of strategy pairs and consistent belief pairs $\{\pi_m, p_m\}_m$, there exists a (π_m, p_m):

$$E_i(\pi^m, p^m, \lambda^k, h_i^{t_i}) \geq E_i((\pi_{-i}^m, \pi_i), p_n(\pi_{-i}^m, \pi_i), \lambda^k, h_i^{t_i}),$$

for any strategy π_i with induced probability distributions in $\Pi_{t_i=1}^{T} = \Delta^{\frac{1}{m}}(p(h_i^{t_i}))$ (Chakrabarti and Topolyan, 2011). Then, consider the strategy and belief pair $\hat{\pi}, \hat{p}$ given by:

$$(\hat{\pi}, \hat{p}) = \lim_{m \leftarrow \infty} (\pi^m, p^m).$$

Such a limit exists because $\Pi_{j=1}^{m} \Pi_{t_j=1}^{T_j} \Delta^{\frac{1}{m}}(p(h_{t_j}^j))$ forms a compact subset of a Euclidean space, and every sequence $\{\pi_m, p_m\}_m$ has a limit point. We claim that for each player i and each reasoning path $h_i^{t_i}$:

$$E_i(\hat{\pi}^m, \hat{p}^m, \lambda^k, h_i^{t_i}) \geq E_i((\hat{\pi}_{-i}^m, \pi_i), p(\hat{\pi}_{-i}^m, \pi_i), \lambda^k, h_i^{t_i}), \tag{4.5}$$

for any strategy π_i of player i.

If not, then for some player i and some strategy π_i of player i, we have:

$$E_i(\hat{\pi}^m, \hat{p}^m, \lambda^k, h_i^{t_i}) < E_i((\hat{\pi}_{-i}^m, \lambda^k, \pi_i), p(\hat{\pi}_{-i}^m, \pi_i), \lambda^k, \lambda^k, h_i^{t_i}).$$

Then, we let

$$E_i((\hat{\pi}^m_{-i}, \pi_i), p(\hat{\pi}^m_{-i}, \pi_i), \lambda^k, h_i^{t_i}) - E_i(\hat{\pi}^m, \hat{p}^m, \lambda^k, h_i^{t_i}) = b > 0.$$

Now as the expected payoffs are continuous in the probability distributions at the reasoning paths and the beliefs, it follows that there is an m_0 sufficiently large such that for all $m \geq m_0$ (Chakrabarti and Topolyan, 2011),

$$|E_i(\pi^m, p^m, \lambda^k, h_i^{t_i}) - E_i(\hat{\pi}^m, \hat{p}^m, \lambda^k, h_i^{t_i})| \leq \frac{b}{4},$$

and

$$E_i((\hat{\pi}^m_{-i}, \pi_i), p_n(\hat{\pi}^m_{-i}, \pi_i), \lambda^k, h_i^{t_i}) - E_i(\hat{\pi}^m, \hat{p}^m, \lambda^k, h_i^{t_i}) \leq \frac{b}{4}.$$

From above equations and for all $m \geq m_0$, we have

$$\begin{aligned} E_i((\pi^m_{-i}, \pi_i), p(\pi^m_{-i}, \pi_i), \lambda^k, h_i^{t_i}) &\geq E_i((\hat{\pi}^m_{-i}, \pi_i), p(\hat{\pi}^m_{-i}, \pi_i), \lambda^k, h_i^{t_i}) - \frac{b}{4} \\ &= E_i(\hat{\pi}, \hat{p}, \lambda^k, h_i^{t_i}) + \frac{3b}{4} \\ &\geq E_i((\pi^m_{-i}, \pi_i), p(\pi^m_{-i}, \pi_i), \lambda^k, h_i^{t_i}) + \frac{b}{2}. \end{aligned}$$

for a given sequential game, there is a $T > 0$ such that

$$|E_i((\pi^\xi_{-i}, \pi_i), p_n(\pi^\xi_{-i}, \pi_i), \lambda^k, h_i^{t_i}) - E_i(\hat{\pi}^\xi, \hat{p}^\xi, \lambda^k, h_i^{t_i})| < \frac{T}{\xi},$$

where $\pi_i = \lim_{\xi \leftarrow \infty} \pi_i^\xi$ of a sequence $\{\pi_i^\xi\}^\xi$ of $\frac{1}{\xi}$ bounded strategies of player i. For the sequence $\{\pi_m, p_m\}$ we now choose an m_1 sufficiently large such that $\frac{T}{m} < \frac{b}{4}$. Therefore, for any strategy π_i of player i, we have

$$\begin{aligned} E_i((\pi^m_{-i}, \pi_i), p_n(\pi^m_{-i}, \pi_i), \lambda^k, h_i^{t_i}) &\geq E_i((\pi^m_{-i}, \pi_i), p(\pi^m_{-i}, \pi_i), \lambda^k, h_i^{t_i}) - \frac{T}{m} \\ &= E_i(\pi_m, p_m, \lambda^k, h_i^{t_i}) + \frac{b}{4}. \end{aligned}$$

But this result contradicts the previous claim in Equation 4.5, which indicates the claim must hold. In other words, Perfect Bayesian Equilibrium must exist.

Remark 4.1. *When $\lambda^k = 1$, it is the special case where the policy is level-k strategy, and it coincides with Perfect Bayesian Equilibrium.*

■

Apart from the main theorems, we could also have two interesting propositions.

Proposition 4.1. *In both the GR2-L & GR2-M model, if the agents play pure strategies, once level-k agent reaches a Nash Equilibrium, all higher-level agents will follow it too.*

Proof. Consider the following two cases GR2-L and GR2-M.

GR2-L. Since agents are assumed to play pure strategies, if a level-k agent reaches the equilibrium, $\pi_i^{k,*}$, in the GR2-L model, then all the higher-level agents will play that equilibrium strategy too, i.e. $\pi_{-i}^{k+1,*} = \pi_i^{k,*}$. The reason is because high-order thinkers will conduct at least the same amount of computations as the lower-order thinkers, and level-k model only needs to best respond to level-$(k-1)$. On the other hand, as it is showed by the Equation 4.3, higher-level recursive model contains the lower-level models by incorporating it into the inner loop of the integration.

GR2-M. In the GR2-M model, if the level-k step agent play the equilibrium strategy $\pi_i^{k,*}$, it means the agent finds the best response to a mixture type of agents that are among level-0 to level-$(k-1)$. Such strategy $\pi_i^{k,*}$ is at least weakly dominant over other pure strategies. For level-$(k+1)$ agent, it will face a mixture type of level-0 to level-$(k-1)$, plus level-k.

For mixture of level-0 to level-$(k-1)$, the strategy $\pi_i^{k,*}$ is already the best response by definition. For level-k, $\pi_{i,*}^k$ is still the best response due to the conclusion in the above GR2-L. Considering the linearity of the expected reward for GR2-M:

$$\begin{aligned}&\mathbb{E}[\lambda_0 V_i(s;\pi_i^{0,*},\pi_{-i}) + \cdots + \lambda^k V_i(s;\pi_i^{k,*},\pi_{-i})]\\ &= \lambda_0 \mathbb{E}[V_i(s;\pi_i^{0,*},\pi_{-i})] + \cdots + \lambda^k \mathbb{E}[V_i(s;\pi_i^{k,*},\pi_{-i})],\end{aligned}$$

where λ^k is level-k policy's proportion. Therefore, $\pi_{i,*}^k$ is the best response to the mixture of level-0 to level-k agent, i.e. the best response for level-$k+1$ agent.

Given that $\pi_{i,*}^{k}$ is the best response to both level-k and level 0- $(k-1)$, it is therefore the best response of the level-$(k+1)$ thinker.

Combining the above two results, therefore, in GR2, once a level-k agent reaches a pure Nash strategy, all higher-level agents will play it too.

■

Corollary 4.1. *In the GR2 setting, higher-level strategies weakly dominant lower-level strategies.*

Proof. By considering all possible actions from lower-level agents, higher-level thinkers will always conduct at least the same amount of computations as the lower-level thinkers as shown in Figure 4.1. ■

4.3 Multi-Agent GR2 Reinforcement Learning

In this section, we incorporate the GR2 models into the MARL, in particular the framework of soft learning, and propose the GR2 Soft Actor-Critic algorithm.

4.3.1 GR2 Soft Actor-Critic Algorithm

To evaluate the level-k policy, since agent i cannot access the joint soft $Q_i(s, a_i, a_{-i})$ as it depends on the exact opponent policies π_{-i} that are unknown, we instead compute the independent soft $Q_i(s, a_i)$ by marginalizing the joint Q-function via the estimated opponent model $\rho_{-i}^{\phi_{-i}}$:

$$Q_i(s, a_i) = \log \int \rho_{-i}^{\phi_{-i}}(a_{-i}|s, a_i) \exp(Q_i(s, a_i, a_{-i})) \, \mathrm{d}a_{-i}. \tag{4.6}$$

With the marginalized $Q_i(s, a_i)$, combining with the soft value function defined in Equation 4.1, we can obtain the value function of the level-k policy $\pi_i^k(a_i|s)$ by

$$V_i(s) = \mathbb{E}_{a_i \sim \pi_i^k}[Q_i(s, a_i) - \log \pi_i^k(a_i|s)]. \tag{4.7}$$

Each agent rolls out the recursive reasoning to level k, either through GR2-L or through GR2-M, and then sample the best response. With the value function in Equation 4.7, we can train the parameter ω_i of the joint soft Q-function via minimizing

Algorithm 4.3 GR2-L/M Soft Actor-Critic Algorithm

```
 1: Set hyper-parameter $\lambda, k$ and $\psi$ (learning rates).
 2: Initialize $\theta_i, \phi_{-i}, \omega_i$ for each agent $i$.
    Assign target parameters: $\bar{\omega}_i \leftarrow \omega_i$; $\mathcal{D}_i \leftarrow$ empty replay buffer for each agent $i$.
 3: for each episode do
 4:     for each step $t$ do
 5:         For each agent $i$, sample action $a_i$ according to $\pi_i^{\theta_i,k}(s)$ in Equation 4.3 or
            $\pi_{i,\lambda}^{\theta_i,k}(s)$ in Equation 4.4.
 6:         Sample next state: $s' \sim p(s'|s, a_i, a_{-i})$.
 7:         Add the tuple $(s, a_i, a_{-i}, r_i, s')$ to $\mathcal{D}_i$.
 8:         for each agent $i$ do
 9:             Sample $\{(s'^{,j}, a_i^j, a_{-i}^j, r_i^j, s'^{,j})\}_{j=0}^M \sim \mathcal{D}_i$. Roll out policy from level
                $0 \leftarrow k$ to get $a_{i'}^j$, with each level take the best response.
10:             Record inter-level results $(a_{i'}^{j,k}, a_{-i'}^{j,k-1}, \cdots)$ for auxiliary objective in
                Equation 4.12.
11:             Sample $a_{-i'}^j \sim \rho_{-i}^{\phi_{-i}}(\cdot|s'^{,j}, a_{i'}^j)$ for each $a_{i'}^j$, $s'^{,j}$.
12:             $\omega_i \leftarrow \omega_i - \psi_{Q_i} \hat{\nabla}_{\omega_i} J_{Q_i}(\omega_i)$.
13:             $\theta_i \leftarrow \theta_i - \psi_{\pi_i} \hat{\nabla}^{\theta_i}(J^{\pi_i^k}(\theta_i) + J^{\pi_i^{\tilde{k}}}(\theta_i))$.
14:             $\phi_{-i} \leftarrow \phi_{-i} - \psi_{\rho_{-i}} \hat{\nabla}_{\phi_{-i}} J_{\rho_{-i}}(\phi_{-i})$.
15:         end for
16:         Update target network for each agent $i$:

                $\bar{\omega}_i \leftarrow \psi_{target}\omega_i + (1 - \psi_{target})\bar{\omega}_i$.

17:     end for
18: end for
```

the soft Bellman residual,

$$J_{Q_i}(\omega_i) = \mathbb{E}_{(s,a_i,a_{-i})\sim\mathcal{D}_i}[\frac{1}{2}(Q_i^{\omega_i}(s, a_i, a_{-i}) - \hat{Q}_i(s, a_i, a_{-i}))^2], \tag{4.8}$$

where s' is next state, and

$$\hat{Q}_i(s, a_i, a_{-i}) = r_i(s, a_i, a_{-i}) + \gamma\mathbb{E}_{s'\sim p}[V_i(s')]. \tag{4.9}$$

The level-k policy parameter θ_i can be learned by the following objective, which is in fact equivalent to minimizing the KL divergence between $\pi_i^{\theta_i,k}$ and

$Q_i^{\omega_i}(s, a_i)$ (Haarnoja et al., 2017):

$$J^{\pi_i^k}(\theta_i) = \mathbb{E}_{\substack{s\sim\mathcal{D}_i,\\ a_i^k\sim\pi_i^{\theta_i,k}}}[\log \pi_i^{\theta_i,k}(a_i|s) - Q_i^{\omega_i}(s, a_i)], \tag{4.10}$$

In the multi-agent soft learning setting, the optimal opponent model still ρ_{-i} follows Equation 4.2. We can therefore update ϕ_{-i} by minimizing the KL-divergence between the current estimated policy $\rho_{-i}^{\phi_{-i}}$ and the *advantage* function:

$$J_{\rho_{-i}}(\phi_i) = D_{\mathrm{KL}}\left[\rho_{-i}^{\phi_{-i}}(a_{-i}|s, a_i)\middle\| \exp(Q_i^{\omega_i}(s, a_i, a_{-i}) - Q_i^{\omega_i}(s, a_i))\right]. \tag{4.11}$$

In practice, we can maintain two approximated Q-functions of $Q_i^{\omega_i}(s, a_i, a_{-i})$ and $Q_i^{\omega_i}(s, a_i)$ separately, and then iteratively update the $\rho_i^{\phi_{-i}}$ via SVGD (Haarnoja et al., 2017; Liu and Wang, 2016).

We summarize the whole algorithm in Algorithm 4.3.

4.3.2 Inter-level Policy Improvement

As pointed out by Corollary. 4.1, higher-level policy should respond at least as good as lower-level policy to the opponent actions. In fact, the highest-level policy π_i^k directly receives the feedback reward from the environment and then back propagates it to lower level policies. The intermediate level policy then receives the reward signal through the chain. As shown in Figure 4.3 , we shall have $Q_i(s, a_i^{\tilde{k}}, a_{-i}^{\tilde{k}-1}) \geq Q_i(s, a_i^{\tilde{k}-2}, a_{-i}^{\tilde{k}-1})$ for $\tilde{k} \leq k$. In order to maintain this property, we introduce an auxiliary objective during training, that is, for $\tilde{k} \geq 2$,

$$J^{\pi_i^{\tilde{k}}}(\theta_i) = \mathbb{E}_{s,(a_i^{\tilde{k}}, a_{-i}^{\tilde{k}})\sim\mathcal{D}_i,\pi_i^{\theta_i,k}}\left[Q_i(s, a_i^{\tilde{k}-2}, a_{-i}^{\tilde{k}-1}) - Q_i(s, a_i^{\tilde{k}-2}, a_{-i}^{\tilde{k}-1})\right] \tag{4.12}$$

As it is showed in the later section, such auxiliary objective plays a critical role in helping the convergence.

4.3.3 Best Response as Deterministic Strategy

Considering the computational feasibility, we formulate the best response in the form of deterministic strategy. Figure 4.2 shows an example of the possible combination

Table 4.1: The converged equilibrium on the Keynes Beauty Contest with different p and agent number n settings.

DEPTH	NASH	LEVEL 3	LEVEL 2	LEVEL 1		LEVEL 0			
		GR2-L3	GR2-L2	GR2-L1	PR2	MADDPG	DDPG-OM	MASQL	DDPG
$p = 0.7, n = 2$	0.0	0.0	0.0	0.0	4.4	10.6	8.7	8.3	18.6
$p = 0.7, n = 10$	0.0	0.0	0.1	0.3	9.8	18.1	12.0	8.7	30.2
$p = 1.1, n = 10$	100.0	97.6	94.2	92.2	64.0	68.2	61.7	87.5	52.2

of paths at the k-th level reasoning. As the reasoning process involves iterated usages of $\pi_i(a_i|s, a_{-i})$ and $\rho_{-i}(a_{-i}|s, a_i)$, should they be stochastic policies, the cost of integrating over actions from all possible lower level actions would be unsustainable when k increases. Besides, the reasoning process is affected by the stochasticity of the environment, the variance will be further amplified on the stochastic policies.

Our approach to model the deterministic strategy is based on the bijective transformation. (Dinh et al., 2016) first applied it on the deep generative models. Let a_{-i} be the random variable, by employing the change of variable rule, we have

$$\pi_i^k(a_i|s, a_{-i}) = \rho_{-i}^{k-1}(a_{-i}|s)|\det(\frac{\mathrm{d}\pi_i^k(a_i|s, a_{-i})}{\mathrm{d}a_{-i}})|^{-1}. \tag{4.13}$$

The computation of the determinant can be further simplified by choosing a special bijective transformations that have friendly Jacobian matrix. For example, in the GR2-L model, level-0 is usually assumed as uniformly distributed, we have:

$$\begin{aligned}\mathbb{E}_{a_i}[\pi_i^1(a_i|s)] &= \mathbb{E}_{a_i}[\mathbb{E}_{a_{-i}}[\rho_{-i}^0(a_{-i}|s)\pi_i^1(a_i|s, a_{-i})]] \\ &= \mathbb{E}_{a_{-i}}[\rho_{-i}^0(a_{-i}|s)\pi_i^1(a_i|s, a_{-i})].\end{aligned}$$

4.4 Experiments

We compare the proposed GR2-L/GR2-M Soft Actor-Critic algorithm with a series of baselines including PR2 (Wen et al., 2019), MASQL (Grau-Moya et al., 2018; Wei et al., 2018), MADDPG (Lowe et al., 2017) and independent learner via DDPG (Lillicrap et al., 2015). To compare against traditional opponent modeling methods, similar to (He et al., 2016; Rabinowitz et al., 2018), we implement an additional baseline of DDPG with an opponent module that is trained online with supervision in order to capture the latest opponent behaviors, called DDPG-OM.

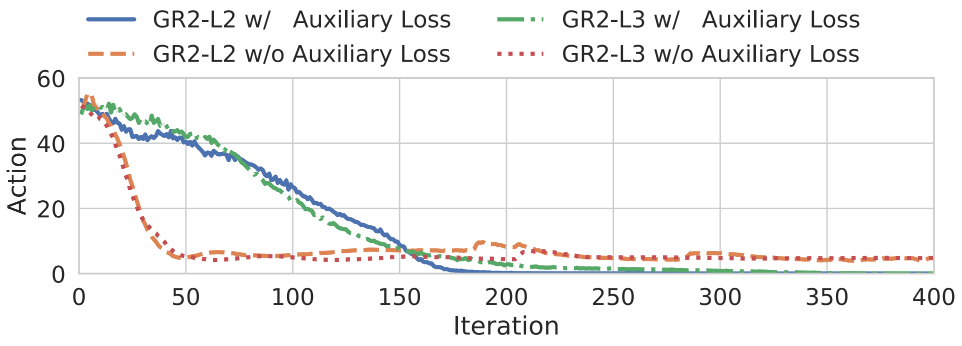


Figure 4.4: Learning curves with or without the auxiliary loss of Equation 4.12

The Recursive Level. We regard DDPG, DDPG-OM, MASQL, MADDPG as level-0 reasoning models because from the policy level, they do not explicitly model the impact of one agent's action on the other agents or consider the reactions from the other agents. Even though the value function of the joint policy is learned in MASQL and MADDPG, but they conduct a *non-correlated factorization* (Wen et al., 2019) when it comes to each individual agent's policy. PR2 is in fact a level-1 reasoning model, but note that the level-1 model in GR2 stands for $\pi_i^1(a_i|s) = \int_{a_{-i}} \pi_i^1(a_i|s, a_{-i})\rho_{-i}^0(a_{-i}|s)\,\mathrm{d}a_{-i}$, while the level-1 model in PR2 starts from the opponent's angel, that is $\rho_{-i}^1(a_{-i}|s) = \int_{a_i} \rho_i^1(a_{-i}|s, a_i)\pi_i^0(a_i|s)\,\mathrm{d}a_i$.

Experimental Settings. We describe k as the *highest order* of reasoning in GR2-Lk and GR2-Mk. All the policies and Q-functions are parameterized by the MLP with 2 hidden layers, where each has 100 units with the ReLU activation. In the Keynes Beauty Contest, we train all the methods including the baselines for 400 iterations with 10 steps per iteration. In the matrix game, we train the agents for 200 iterations with 25 steps per iteration. And for the particle games, all the models are trained up to $300k$ steps with maximum 25 episode length. In the actor-critic algorithms, we set the exploration noise to 0.1 in the first 1000 steps. The annealing parameters in soft algorithms are set to 0.5 to balance the exploration/exploitation. We adopt $k = 1, 2, 3$ and set $\lambda = 1.5$.

4.4.1 Keynes Beauty Contest

In the Keynes Beauty Contest game, all n agents pick a number between 0 and 100. The winner is the agent whose “guess" is closest to p times of the average number.

The reward is set as the absolute difference to the winner guess. We evaluate the effect of reward shaping in *Section 4.4.4*.

We first conduct the experiments by setting different p and n values. Table. 4.1 presents the results. We notice that the GR2-L algorithms can effectively approach the equilibrium, while the other baselines struggle to reach. We believe it is because the level-0 methods are unable to model the explicit dependency between agents' actions, so that the synergy of guessing smaller/larger together will not happen. In addition, with the maximum level of recursive reasoning grows, the equilibrium that GR2-Lk finds gets increasingly closer to the true NE, which is expected. In the case of $p = 0.7, n = 2$, the equilibrium is first reached by GR2-L1, the other higher-order models follow the same equilibrium afterwards; this is in line with the Proposition 4.1. The equilibrium found by GR2-L3 is 97.6 when $p = 1.1$, the error is due to the saturated reward design that stops the momentum of agents' changing the actions anywhere further.

Furthermore, we analyze the effectiveness of the auxiliary loss in Equation 4.12 that aims to guarantee the inter-level policy improvement. In Figure 4.4, we can find that although the GR2-L model without auxiliary loss learns fast in the beginning, it however fails to reach a better equilibrium. We believe adding the auxiliary loss in the objective can help the joint Q-function to guide a better direction for agents' reasoning.

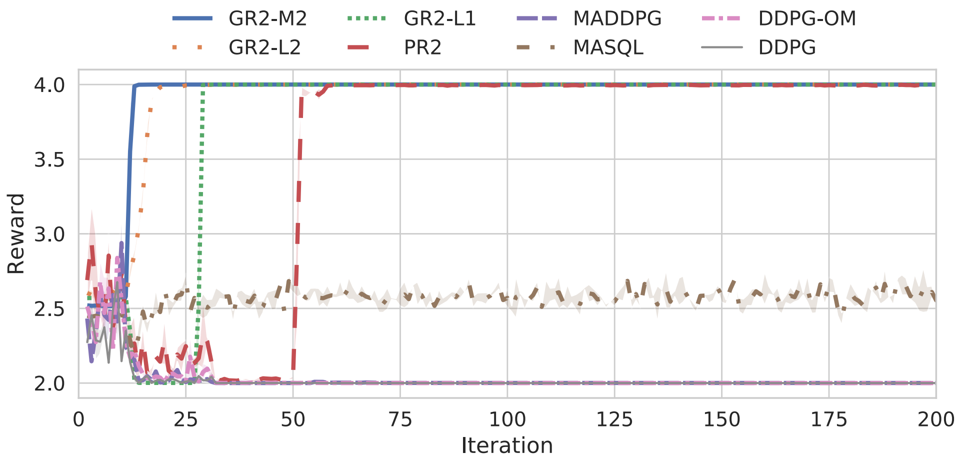
4.4.2 Learning Matrix Games

We then evaluate on two matrix games: Prisoner's Dilemma (PD) and Stag Hunt (SH). The reward matrix of PD is:

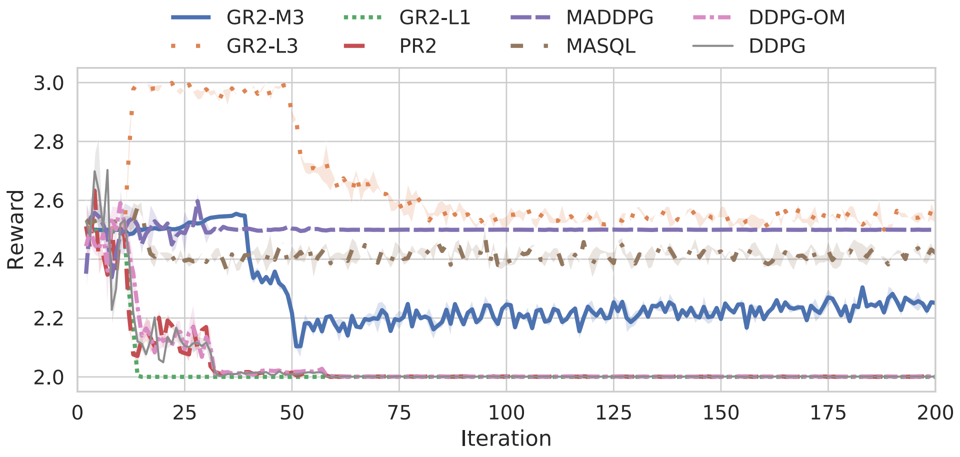
		Agent 2	
		Cooperate	Defect
Agent 1	Cooperate	$3, 3$	$1, 4$
	Defect	$4, 1$	$2, 2$

Prisoner's Dilemma (PD)

Agents can choose to cooperate (C) or defect (D). Defect is the dominant strategy for both agents, while there exists a Pareto optima (C, C).



(a) Stag Hunt Game



(b) Prisoner's Dilemma

Figure 4.5: Learning curves on the matrix games.

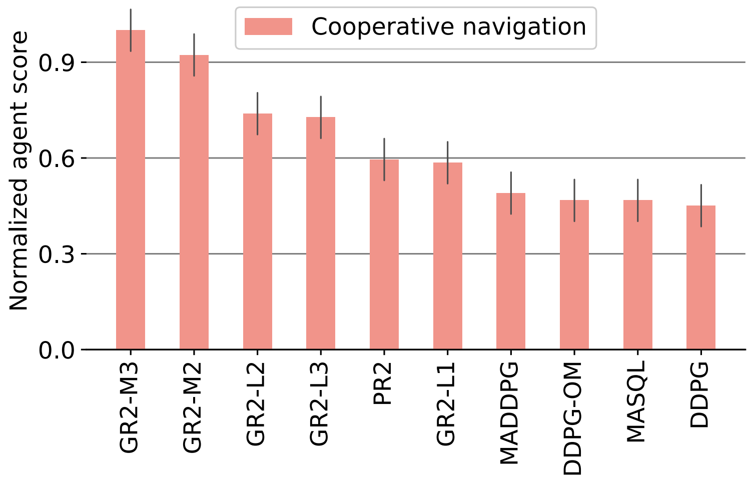


Figure 4.6: Performance comparison in the cooperative game.

In SH, the reward matrix is:

$$\begin{array}{cc|c|c|} & & \multicolumn{2}{c}{\text{Agent 2}} \\ & & \text{Stag} & \text{Rabbit} \\ \cline{3-4} \text{Agent 1} & \text{Stag} & 3,3 & 1,4 \\ \cline{3-4} & \text{Rabbit} & 4,1 & 2,2 \\ \cline{3-4} \end{array}$$

Stag Hunt (SH)

Agents can either cooperate to hunt the large stag (S) or to chase a small rabbit (P) alone. In this game, there is no dominant strategy, and it comes with two Nash equilibria, i.e. (S, S) that is also Pareto-optimal, and the deficient equilibrium (P, P). To turn the matrix games compatible with the actor-critic setting, we make the actor output the probability of choosing one action and define the reward based on the payoff matrix. We define the state at time t to be $s^t = (a^1_{t-1}, a^2_{t-1})$.

In SH, we expect the agents to reach the Pareto optima, so that both agents receive the maximum reward. Figure 4.5a compares the average training reward for a list of algorithms. GR2-L/GR2-M models, together with PR2, can easily reach the Pareto optima with maximum average return 4, whereas other models are trapped into deficient equilibrium with average return 2. On the convergence speed, higher-order thinkers are faster than lower-order thinkers, and mixture models are faster than level-k models. SH is a coordination game that motivates players to cooperate, but the challenging part is the conflict between individual safety and social welfare. Without knowing the choice of the other, GR2 has to anchor the belief that opponents would like to choose the social welfare, and then reinforce this belief into the high-order reasoning to finally build the trust among agents. Since there is no dominant strategy in SH, agents will in the end choose to stay fixed in that strategy. The level-0 methods cannot build such synergy because they cannot discriminate the individual safety from the social welfare; both NEs can saturate the value function of the joint action.

PD is different from SH in that there is a dominant strategy of defection which leads the Pareto optima to be unstable. In Figure 4.5b, although GR2-L3 could reach the Pareto optima within the first 50 steps, but soon it chooses to defect. Since defection is dominant, as a level-2 thinker, it will surely defect; such reasoning will be passed into the level-3 thinker, and GR2-L3 soon realizes and best respond with defection immediately. GR2-M3 under-performs GR2-L3 because it is a mixture strategy, which still considers the level-1 thinker who still chases for cooperation. As for the rest of the baselines, they do not present such behavioral changes at all; meanwhile, they converge to a worse equilibrium.

4.4.3 Particle World Environments

We further adopt the Particle World Environments (Lowe et al., 2017) to test our method on high-dimensional state/action space scenarios. More specifically, it is a fully-cooperative *Cooperative Navigation* task with 3 agents and 3 landmarks. In this task, 3 agents must cooperate through physical actions to reach 3 different landmarks. Agents can observe the relative positions of other agents and landmarks. Agents are collectively rewarded based on the proximity of any agent to each landmark while avoiding collisions;

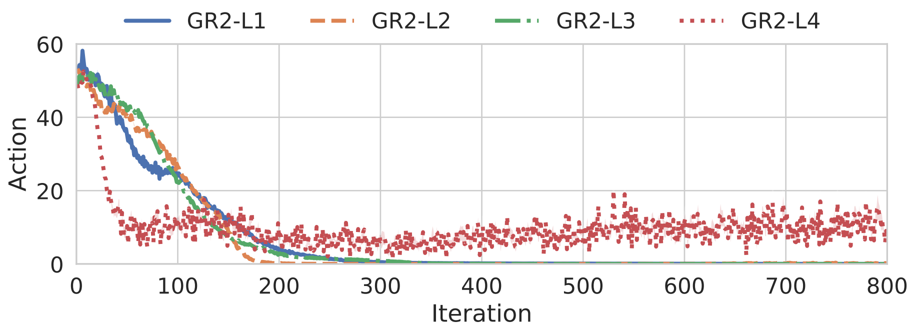
We measure the performance of agents trained by different algorithms in each of the environment. The results are demonstrated in Figure 4.6; we report the rewards of the agents and normalize them to 0-1. We notice that in this the scenario, the GR2-M methods can achieve the highest score. The level-3 models can out-perform the lower-level models. In the cooperative game, the GR2-L/M methods have critical advantages over traditional baselines, this is inline with the finding in SH that GR2 methods are good at finding the Pareto optimality.

4.4.4 Ablation Study

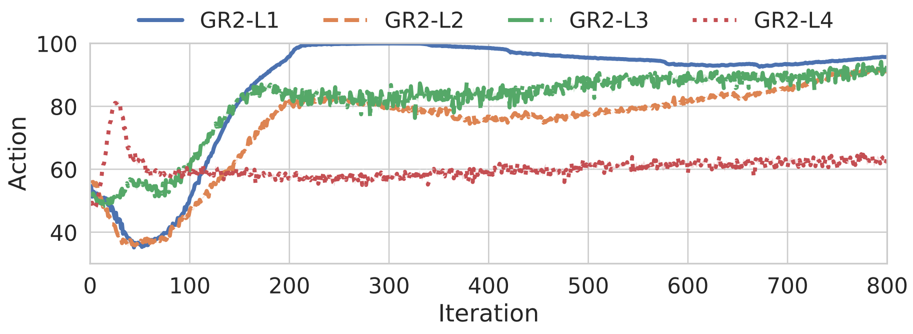
The results in the experiment section suggest that GR2 algorithms can outperform other multi-agent RL methods various tasks. In this section, we examine how sensitive GR2 methods is to some of the most important hyper-parameters, including the level-k and the choice of the Poisson mean λ in GR2-M methods.

4.4.4.1 Choice of Level-k

First, we investigate the choice of level-k by testing the GR2-L models with various k on the Keynes Beauty Contest. According to the Figure 4.7, in both setting, the



(a) $p = 0.7, n = 10$



(b) $p = 1.1, n = 10$

Figure 4.7: Learning curves on the Keynes Beauty Contest game with GR2-L policies from level-1 to level-4.

GR3-L with level form $1-3$ can converge to the equilibrium, but the GR3-L4 cannot. The learning processes show that the GR3-L4 models have high variance during the learning. This phenomenon has two reasons: with k increases, the reasoning path would have higher variance; and in GR2-L4 policy, it uses the approximated opponent conditional policy $\rho_{-i}(a_{-i}|s, a_i)$ twice (only once in GR2-L2/3), which would further amplify the variance.

4.4.4.2 Poisson λ for the GR2-M methods.

We investigate the effect of hyper-parameter λ in the GR2-M models. We test the GR2-M model on the cooperative navigation game; empirically, the test selection of $\lambda = 1.5$ on both GR2-M3 and GR2-M2 would lead to best performance. We therefore use $\lambda = 1.5$ in the experiment section of this chapter.

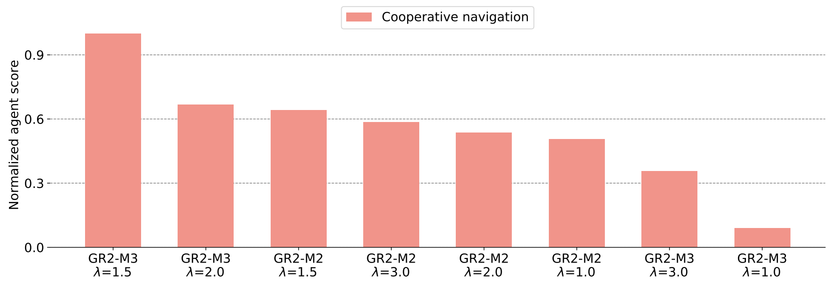


Figure 4.8: Effect of varying λ in GR2-M methods, the score is normalized to $0 - 1$.

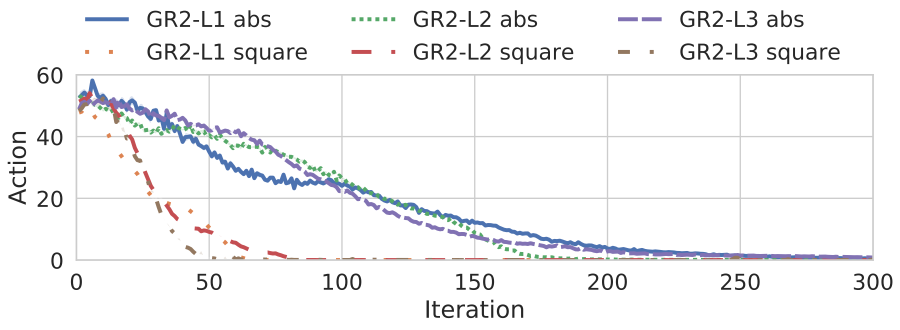


Figure 4.9: Learning curves with two reward schemes: absolute difference (default) and squared absolute difference.

4.4.4.3 Incentive Intensity in Keynes Beauty Contest

One sensible finding from human players suggests that when prize of winning gets higher, people tend to use more steps of reasoning and they may think others will think harder too. We simulate a similar scenario by reward shaping. We consider two reward schemes of absolute difference and squared absolute difference. Interestingly, we find similar phenomenon in Figure 4.9 that the amplified reward can significantly speed up the convergence for GR2-L methods.

4.5 Summary

In this chapter, following the study of bounded rationality (Simon, 1972) and behavior game theory (Camerer, 2003), we introduce the protocol of generalized recursive reasoning (GR2) that allows agents to have different levels of recursive reasoning capability when dealing with other agents. In GR2, each agent takes the best response to a mixed type of agents that think and behave at lower levels. We integrate GR2 into the MARL framework. The induced GR2 Soft Actor-Critic Algorithm shows superior performance on building the correct beliefs about the opponents behaviors even when they are non-equilibrium. We prove in theory that GR2 can reach NE when the level approximates infinity. Empirically, when compared to conventional opponent models (level 0), learning with higher level recursive reasoning capability would lead to faster convergence.

Part II

Behavioral Diversity in Mutual Influence

Chapter 5

Multi-Agent Determinantal Q-Learning

Multi-agent reinforcement learning (MARL) methods hold great potential to solve a variety of real-world problems, such as mastering multi-player video games (Peng et al., 2017), dispatching taxi orders (Li et al., 2019), and studying population dynamics (Yang et al., 2018b). In this chapter, we consider the multi-agent cooperative setting (Panait and Luke, 2005) where a team of agents collaborate to achieve one common goal in a partially observed environment.

A full spectrum of MARL algorithms has been developed to solve cooperative tasks (Panait and Luke, 2005); the two endpoints of the spectrum are independent and centralized learning (see Figure 5.1). Independent Learner (IL) (Tan, 1993) merely treats other agents' influence to the system as part of the environment. The learning agent not only faces a non-stationary environment, but also suffers from *spurious* rewards (Sunehag et al., 2018). Centralized Learner (CL), in the other extreme, treats a multi-agent problem as a single-agent problem despite the fact that many real-world applications require local autonomy. Importantly, the CL approaches exhibit combinatorial complexity and can hardly scale to more than tens of agents (Yang et al., 2019).

Another paradigm typically considered is a hybrid of centralized training and decentralized execution (CTDE) (Oliehoek et al., 2008). For value-based approaches in the framework of CTDE, a fundamental challenge is how to correctly decompose

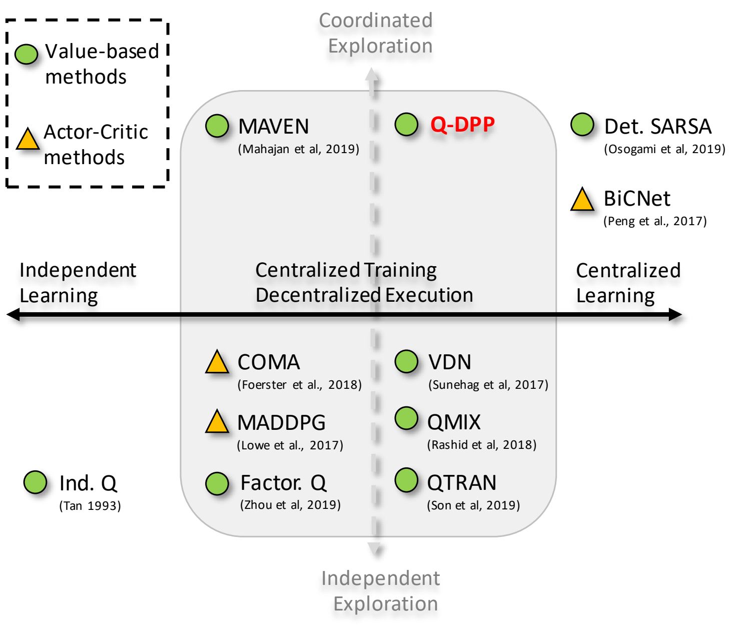

Value-based methods
Actor-Critic methods
Coordinated Exploration
MAVEN (Mahajan et al, 2019)
Q-DPP
Det. SARSA (Osogami et al, 2019)
BiCNet (Peng et al., 2017)
Independent Learning
Centralized Training Decentralized Execution
Centralized Learning
COMA (Foerster et al., 2018)
VDN (Sunehag et al, 2017)
MADDPG (Lowe et al., 2017)
QMIX (Rashid et al, 2018)
Ind. Q (Tan 1993)
Factor. Q (Zhou et al, 2019)
QTRAN (Son et al, 2019)
Independent Exploration

Figure 5.1: Spectrum of MARL methods on cooperative tasks.

the centralized value function among agents for decentralized execution. For a cooperative task to be deemed decentralizable, it is required that local maxima on the value function per every agent should amount to the global maximum on the joint value function. In enforcing such a condition, current state-of-the-art methods rely on restrictive structural constraints and/or network architectures. For instance, Value Decomposition Network (VDN) (Sunehag et al., 2018) and Factorized-Q (Zhou et al., 2019) propose to directly factorize the joint value function into a summation of individual value functions. QMIX (Rashid et al., 2018) augments the summation to be non-linear aggregations, while maintaining a monotonic relationship between centralized and individual value functions. QTRAN (Son et al., 2019) introduces a refined learning objective on top of QMIX along with specific network designs.

Unsurprisingly, the structural constraints put forward by VDN / QMIX / QTRAN inhibit the representational power of the centralized value function (Son et al., 2019); as a result, the class of decentralizable cooperative tasks these methods can tackle is limited. For example, poor empirical results of QTRAN have been reported on multiple multi-agent cooperative benchmarks (Mahajan et al., 2019).

Apart from the aforementioned problems, structural constraints also hinder efficient explorations when applied to value function decomposition. In fact, since agents behave independently during the execution stage, CTDE methods usually lack a coordinated exploration strategy (Matignon et al., 2007). Clearly, an increasing per-agent exploration rate of ϵ-greedy in the single-agent setting can help exploration; however, it has been proved (Mahajan et al., 2019) that due to the structural constraints (e.g. the monotonicity assumption in QMIX), in the multi-agent setting, increasing ϵ will only lower the probability of obtaining the optimal value function. As a treatment, MAVEN (Mahajan et al., 2019) introduces a hierarchical model to coordinate diverse explorations among agents. Yet, a principled exploration strategy with minor structural constraints on the value function is still missing for value-based CTDE methods.

In many tasks that require the division of labor, one reasonable solution is to make agents acquire a **diverse** set of skills, targets or behavioral models (Albrecht and Ramamoorthy, 2012) during training so that the joint goal can be achieved by individual targets. In such scenarios, the equivalence between the local maxima on the per-agent value function and the global maximum on the joint value function can be automatically achieved. As a result, the centralized value function can enjoy a natural factorization with no need for any structural constraints beforehand.

In this chapter, we present a new value-based solution in the CTDE paradigm to multi-agent cooperative tasks. We establish Q-DPP, an extension of determinantal point process (DPP) (Macchi, 1977) with partition constraint, and apply it to multi-agent learning. DPPs are elegant probabilistic models on sets that can capture both quality and diversity when a subset is sampled from a ground set; this makes them ideal for modeling the set that contains different agents' observation-action pairs in the multi-agent learning context. We adopt Q-DPP as a function approximator for the centralized value function. An attractive property of using Q-DPP is that, when reaching the optimum, it can offer a natural factorization on the centralized value function, assuming agents have acquired a diverse set of behaviors. Our method eliminates the need for *a priori* structural constraints or bespoke neural architectures. In fact, we demonstrate that Q-DPP generalizes current solvers including VDN, QMIX, and QTRAN, where all these methods can be derived as special cases from Q-DPP. As an additional contribution, we adopt a tractable sampler, based on the idea of sample-by-projection in P-DPP (Celis et al., 2018), for Q-DPP with theoretical approximation guarantee. Our sampler makes agents explore in a sequential manner; agents who act later are coordinated to visit only the orthogonal areas in the state space that previous agents haven't explored. Such coordinated way of explorations effectively boosts the sampling efficiency in the CTDE setting. Building upon these advantages, finally, we demonstrate that our proposed Q-DPP algorithm is superior to the existing state-of-the-art solutions on a variety of multi-agent cooperation benchmarks.

5.1 Preliminaries: Determinantal Point Process

DPP is a probabilistic framework that characterizes how likely a subset is going to be sampled from a ground set. Originated from quantum physics for modeling repulsive Fermion particles (Macchi, 1977), DPP has recently been introduced to the machine learning community due to its probabilistic nature (Kulesza et al., 2012).

Definition 5.1 (DPP)**.** *For a ground set of items $\mathcal{Y} = \{1, 2, \dots, M\}$, a DPP, denoted by $\mathbb{P}$, is a probability measure on the set of all subsets of $\mathcal{Y}$, i.e., $2^{\mathcal{Y}}$. Given an $M \times M$ positive semi-definite (PSD) kernel $\boldsymbol{\mathcal{L}}$ that measures similarity for any pairs of items in $\mathcal{Y}$, let $\boldsymbol{Y}$ be a random subset drawn according to $\mathbb{P}$, then we have, $\forall Y \subseteq \mathcal{Y}$,*

$$\mathbb{P}_{\boldsymbol{\mathcal{L}}}\big(\boldsymbol{Y} = Y\big) \propto \det\big(\boldsymbol{\mathcal{L}}_Y\big) = \mathrm{Vol}^2\big(\{\boldsymbol{w}_i\}_{i \in Y}\big), \tag{5.1}$$

where $\boldsymbol{\mathcal{L}}_Y := [\boldsymbol{\mathcal{L}}_{i,j}]_{i,j \in Y}$ denotes the sub-matrix of $\boldsymbol{\mathcal{L}}$ whose entries are indexed by the items included in Y. If we write $\boldsymbol{\mathcal{L}} := \boldsymbol{\mathcal{W}}\boldsymbol{\mathcal{W}}^\top$ with $\boldsymbol{\mathcal{W}} \in \mathbb{R}^{M \times P}, P \leq M$, and rows of $\boldsymbol{\mathcal{W}}$ being $\{\boldsymbol{w}_i\}$, then the determinant value is essentially the squared $|Y|$-dimensional volume of parallelepiped spanned by the rows of $\boldsymbol{\mathcal{W}}$ corresponding to elements in Y.

A PSD matrix ensures all principal minors of $\boldsymbol{\mathcal{L}}$ are non-negative $\det(\boldsymbol{\mathcal{L}}_Y) \geq 0$; it thus suffices to be a proper probability distribution. The normalizer can be computed as: $\sum_{Y \subseteq \mathcal{Y}} \det(\boldsymbol{\mathcal{L}}_Y) = \det(\boldsymbol{\mathcal{L}} + \boldsymbol{I})$, where $\boldsymbol{I}$ is an $M \times M$ identity matrix. Intuitively, one can think of a diagonal entry $\boldsymbol{\mathcal{L}}_{i,i}$ as capturing the quality of item i, while an off-diagonal entry $\boldsymbol{\mathcal{L}}_{i,j}$ measures the similarity between items i and j. DPP models the **repulsive** connections among **multiple** items in a sampled subset. In the example of two items, $\mathbb{P}_{\boldsymbol{\mathcal{L}}}(\{i,j\}) \propto \begin{vmatrix} \boldsymbol{\mathcal{L}}_{i,i} & \boldsymbol{\mathcal{L}}_{i,j} \\ \boldsymbol{\mathcal{L}}_{j,i} & \boldsymbol{\mathcal{L}}_{j,j} \end{vmatrix} = \boldsymbol{\mathcal{L}}_{i,i}\boldsymbol{\mathcal{L}}_{j,j} - \boldsymbol{\mathcal{L}}_{i,j}\boldsymbol{\mathcal{L}}_{j,i}$, which suggests, if item i and item j are perfectly similar, such that $\boldsymbol{\mathcal{L}}_{i,j} = \sqrt{\boldsymbol{\mathcal{L}}_{i,i}\boldsymbol{\mathcal{L}}_{j,j}}$, then we know these two items will almost surely not co-occur, hence such two-item subset of $\{i,j\}$ from the ground set will never be sampled.

DPPs are attractive in that they only require training the kernel matrix $\boldsymbol{\mathcal{L}}$, which can be learned via maximum likelihood (Affandi et al., 2014). A trainable DPP favors many supervised learning tasks where diversified outcomes are desired, such as image generation (Elfeki et al., 2019), video summarization (Sharghi et al., 2018), model ensemble (Pang et al., 2019), and recommender system (Chen et al., 2018). It is, however, non-trivial to adapt DPPs to a multi-agent setting since additional restrictions are required to put on the ground set so that valid samples can be drawn for the purpose of multi-agent training. This leads to our Q-DPPs.

5.2 Multi-Agent Determinantal Q-Learning

We offer a new value-based solution to multi-agent cooperative tasks. In particular, we introduce Q-DPPs as general function approximators for the centralized value functions, similar to neural networks in deep Q-learning (Mnih et al., 2015). We start from the problem formulation.

To satisfy Equation 2.9, current solutions rely on restrictive assumptions that enforce structural constraints on the factorization of the joint Q-function. For example, VDN (Sunehag et al., 2018) adopts the additivity assumption by assuming $Q^{\pi}(\boldsymbol{\tau}, \boldsymbol{a}) := \sum_{i=1}^{N} Q_i(\tau_i, a_i)$. QMIX (Rashid et al., 2018) applies the monotonicity assumption to ensure $\frac{\partial Q^{\pi}(\boldsymbol{\tau},\boldsymbol{a})}{\partial Q_i(\tau_i,a_i)} \geq 0, \forall i \in \mathcal{N}$. QTRAN (Son et al., 2019) introduces a refined factorizable learning objective in addition to QMIX. Nonetheless, structural constraints harm the representational power of the centralized value function, and also hinder efficient explorations (Son et al., 2019). To mitigate these problems, we

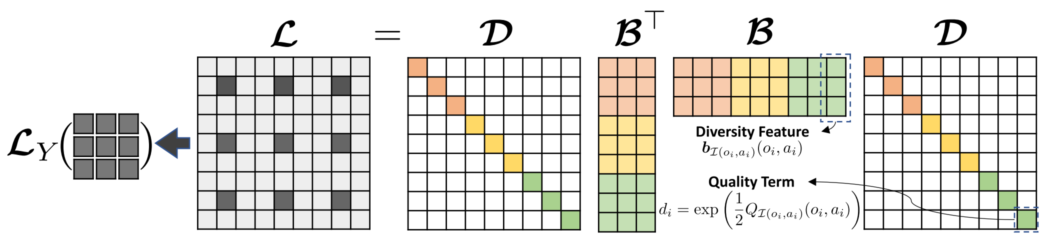


Figure 5.2: Example of Q-DPP with quality-diversity kernel decomposition in a single-state three-player learning task, each agent has three actions to choose. The size of the ground set is $|\mathcal{Y}| = 9$, and the size of valid subsets $|\mathcal{C}(\boldsymbol{o})|$ is $3^3 = 27$. Different colors represent different partitions of each agent's observation-action pairs. Suppose all three agents select the 2nd action, then the Q-value of the joint action according to Equation 5.3 is $Q^{\boldsymbol{\pi}}\left(\boldsymbol{o}, \boldsymbol{a}\right) = \log\det\left([\boldsymbol{\mathcal{L}}_{[i,j],i,j\in\{2,5,8\}}]\right)$.

propose Q-DPP as an alternative that naturally factorizes the joint Q-function by learning a diverse set of behavioral models among agents.

5.2.1 Q-DPP: A Constrained DPP for MARL

Our method is established on Q-DPP which is an extension of DPP that suits MARL. We assume that local observation o_i encodes all history information τ_i at each time-step. We model the ground set of all agents' observation-action pairs by a DPP, i.e., $\mathcal{Y} = \left\{(o_1^1, a_1^1), \ldots, (o_N^{|\mathcal{O}|}, a_N^{|\mathcal{A}|})\right\}$ with the size of the ground set being $|\mathcal{Y}| = N|\mathcal{O}||\mathcal{A}|$.

In the context of multi-agent learning, each agent takes one valid action depending on its local observation. A valid sample from DPP, therefore, is expected to include one valid observation-action pair for each agent, and the observations from the sampled pairs must match the true observations that agents receive at every time step. To meet such requirements, we propose a new type of DPP, named **Q-DPP**.

Definition 5.2 (Q-DPP)**.** *Given a ground set $\mathcal{Y}$ of size M that includes N agents' all possible observation-action pairs $\mathcal{Y} = \left\{(o_1^1, a_1^1), \ldots, (o_N^{|\mathcal{O}|}, a_N^{|\mathcal{A}|})\right\}$, we partition $\mathcal{Y}$ into N disjoint parts, i.e., $\mathcal{Y} = \bigcup_{i=1}^N \mathcal{Y}_i$ and $\sum_{i=1}^N |\mathcal{Y}_i| = M = N|\mathcal{O}||\mathcal{A}|$, where each partition represents each individual agent's all possible observation-action pairs. At every time-step, given agents' observations, $\boldsymbol{o} = (o_i)_{i\in\mathcal{N}}$, we define $\mathcal{C}(\boldsymbol{o}) \subseteq \mathcal{Y}$ to be a set of valid subsets including only observation-action pairs that agents are allowed* *to take,*

$$\mathcal{C}(\boldsymbol{o}) := \big\{Y \subseteq \mathcal{Y} : |Y \cap \mathcal{Y}_i(o_i)| = 1, \forall i \in \{1, \dots, N\}\big\},$$

with $|\mathcal{C}(\boldsymbol{o})| = |\mathcal{A}|^N$, *and* $\mathcal{Y}_i(o_i)$ *of size* $|\mathcal{A}|$ *denotes the set of pairs in partition* $\mathcal{Y}_i$ *with only* o_i *as the observation,*

$$\mathcal{Y}_i(o_i) = \big\{(o_i, a_i^1), \dots, (o_i, a_i^{|\mathcal{A}|})\big\}.$$

Q-DPP, denoted by $\tilde{\mathbb{P}}$, *defines a probability measure over the valid subsets* $Y \in \mathcal{C}(\boldsymbol{o}) \subseteq \mathcal{Y}$. *Let* $\boldsymbol{Y}$ *be a random subset drawn according to* $\tilde{\mathbb{P}}$, *its probability distribution is defined:*

$$\tilde{\mathbb{P}}_{\boldsymbol{\mathcal{L}}}\big(\boldsymbol{Y} = Y | \boldsymbol{Y} \in \mathcal{C}(\boldsymbol{o})\big) := \frac{\det\big(\boldsymbol{\mathcal{L}}_Y\big)}{\sum_{Y' \in \mathcal{C}(\boldsymbol{o})} \det\big(\boldsymbol{\mathcal{L}}_{Y'}\big)}. \tag{5.2}$$

In addition, given a valid sample $Y \in \mathcal{C}(\boldsymbol{o})$, *we define an identifying function* $\mathcal{I} : Y \to \mathcal{N}$ *that specifies the agent number for each valid pair in* Y, *and an index function* $\mathcal{J} : \mathcal{Y} \to \{1, \dots, M\}$ *that specifies the cardinality of each item in* Y *in the ground set* $\mathcal{Y}$.

The construction of Q-DPP is inspired by P-DPP (Celis et al., 2018). However, the partitioned sets in P-DPP stay fixed, while in Q-DPP, $\mathcal{C}(\boldsymbol{o_t})$ changes at every time-step with the new observation, and the kernel $\boldsymbol{\mathcal{L}}$ is learned through the process of reinforcement learning rather than being given.

Difference between Q-DPP and P-DPP: The design of Q-DPP and its samply-by-projection sampling process is inspired by and based on P-DPP (Celis et al., 2018). However, we would like to highlight the multiple differences in that **1)** P-DPP is designed for modeling the fairness for data summarization whereas Q-DPP serves as a function approximator for the joint Q-function in the context of multi-agent learning; **2)** though we analyze Equation 5.19 based on $\boldsymbol{\mathcal{W}}$, the actual orthorgonalziation step of our sampler only needs performing on the vectors of $\boldsymbol{b}_j$ rather than the entire matrix $\boldsymbol{\mathcal{W}}$ due to our unique quality-diversity decomposition on the joint Q-function in Equation 5.4; **3)** the set of elements in each partition $\mathcal{Y}_i(o_i)$ of

Q-DPP change with the observation at each time-step, while the partitions stay fixed in the case of P-DPP; **4)** the parameters of $\boldsymbol{\mathcal{W}}$ are learned through a trail-and-error multi-agent reinforcement learning process compared to the cases in P-DPP where the kernel is given by hand-crafted features (e.g. SIFT features on images); **5)** we implement the constraint in Assumption 5.1 via a penalty term during the CTDE learning process, while P-DPP does not consider meeting such assumption through optimization.

Given Q-DPPs, we can represent the centralized value function by adopting Q-DPPs as general function approximators:

$$Q^{\pi}\big(\boldsymbol{o}, \boldsymbol{a}\big) := \log\det\Big(\boldsymbol{\mathcal{L}}_{Y=\big\{(o_1,a_1),\dots,(o_N,a_N)\big\}\in\mathcal{C}(\boldsymbol{o})}\Big), \tag{5.3}$$

where $\boldsymbol{\mathcal{L}}_Y$ denotes the sub-matrix of $\boldsymbol{\mathcal{L}}$ whose entries are indexed by the pairs included in Y. Q-DPP embeds the connection between the joint action and each agent's individual actions into a subset-sampling process, and the Q-value is quantified by the determinant of a kernel matrix whose elements are indexed by the associated observation-action pairs. The goal of multi-agent learning is to learn an optimal joint Q-function. Equation 5.3 states $\det(\boldsymbol{\mathcal{L}}_Y) = \exp(Q^{\pi}(\boldsymbol{o}, \boldsymbol{a}))$, meaning Q-DPP actually assigns large probability to the subsets that have large Q-values. Given $\det(\boldsymbol{\mathcal{L}}_Y)$ is always positive, the $\log$ operator ensures Q-DPPs, as general function approximators, can recover any real Q-functions.

DPPs can capture both the quality and diversity of a sampled subset; the joint Q-function represented by Q-DPP in theory should not only acknowledge the quality of each agent's individual action towards a large reward, but the diversification of agents' actions as well. The remaining question is, then, how to obtain such quality-diversity representation.

5.2.2 Representation of Q-DPP Kernels

For any PSD matrix $\boldsymbol{\mathcal{L}}$, such a $\boldsymbol{\mathcal{W}}$ can always be found so that $\boldsymbol{\mathcal{L}} = \boldsymbol{\mathcal{W}}\boldsymbol{\mathcal{W}}^{\top}$ where $\boldsymbol{\mathcal{W}} \in \mathbb{R}^{M\times P}, P \leq M$. Since the diagonal and off-diagonal entries of $\boldsymbol{\mathcal{L}}$ represent *quality* and *diversity* respectively, we adopt an interpretable decomposition by expressing each row of $\boldsymbol{\mathcal{W}}$ as a product of a **quality** term $d_i \in \mathbb{R}^+$ and a **diversity** feature term $\boldsymbol{b}_i \in \mathbb{R}^{P \times 1}$ with $\|\boldsymbol{b}_i\| \leq 1$, i.e., $\boldsymbol{w}_i = d_i \boldsymbol{b}_i^\top$. An example of such decomposition is visualized in Figure 5.2 where we define $\boldsymbol{\mathcal{B}} = [\boldsymbol{b}_1, \ldots, \boldsymbol{b}_M]$ and $\boldsymbol{\mathcal{D}} = \mathrm{diag}(d_1, \ldots, d_M)$. Note that both $\boldsymbol{\mathcal{D}}$ and $\boldsymbol{\mathcal{B}}$ are free parameters that can be learned from the environment during the Q-learning process in Equation 2.8.

If we denote the quality term as each agent's individual Q-value for a given observation-action pair, i.e., $\forall (o_i, a_i) \in \mathcal{Y}, i = \{1, \ldots, M\}, d_i := \exp\left(\frac{1}{2} Q_{\mathcal{I}(o_i,a_i)}(o_i, a_i)\right)$, then Equation 5.3 can be further written into

$$\begin{aligned} Q^{\boldsymbol{\pi}}\big(\boldsymbol{o}, \boldsymbol{a}\big) &= \log\det\Big(\boldsymbol{\mathcal{W}}_Y \boldsymbol{\mathcal{W}}_Y^\top\Big) \\ &= \log\Big({}^{\top}\big(\boldsymbol{\mathcal{D}}_Y^\top \boldsymbol{\mathcal{D}}_Y\big)\det\big(\boldsymbol{\mathcal{B}}_Y^\top \boldsymbol{\mathcal{B}}_Y\big)\Big) \\ &= \sum_{i=1}^{N} Q_{\mathcal{I}(o_i,a_i)}\big(o_i, a_i\big) + \log\det\big(\boldsymbol{\mathcal{B}}_Y^\top \boldsymbol{\mathcal{B}}_Y\big). \end{aligned} \tag{5.4}$$

Since a determinant value only reaches the maximum when the associated vectors in $\boldsymbol{\mathcal{B}}_Y$ are mutually orthogonal (Noble et al., 1988), Equation 5.4 essentially stipulates that Q-DPP represents the joint value function by taking into account not only the quality of each agent's contribution towards reward maximization, more importantly, from a holistic perspective, the orthogonalization of agents' actions.

In fact, the inclusion of diversifying agents' behaviors is an important factor in satisfying the condition in Equation 2.9. Intuitively, in a decentralizable task with a shared goal, promoting orthogonality between agent's actions can help clarify the functionality and responsibility of each agent, which in return leads to a better instantiation of Equation 2.9. On the other hand, diversity does not means that agents have to take different actions all the time. Since the goal is still to achieve large reward via optimizing Equation 2.8, certain scenarios, such as agents need to take identical actions to accomplish a task, will not be excluded as a result of promoting diversity.

5.2.3 Connections to Current Methods

Based on the quality-diversity representation, one can draw a key connection between Q-DPP and the existing methods. It turns out that, under the sufficient condition that if the learned diversity features that correspond to the optimal actions are mutually orthogonal, then Q-DPP degenerates to VDN (Sunehag et al., 2018), QMIX (Rashid et al., 2018), and QTRAN (Son et al., 2019) respectively.

To elaborate such condition, let us denote $a_i^* = \arg\max Q_i(o_i, a_i)$, $\boldsymbol{a}^* = (a_i^*)_{i\in\mathcal{N}}$, $Y^* = \{(o_i, a_i^*)\}_{i=1}^N$, with $\|\boldsymbol{b}_i\| = 1$ and $\boldsymbol{b}_i^\top \boldsymbol{b}_j = 0, \forall i \neq j$, then we have

$$\det\left(\boldsymbol{\mathcal{B}}_{Y^*}^\top \boldsymbol{\mathcal{B}}_{Y^*}\right) = 1. \tag{5.5}$$

Connection to VDN. When $\{\boldsymbol{b}_j\}_{j=1}^M$ are pairwise orthognal, by plugging Equation 5.5 into Equation 5.4, we can obtain

$$Q^{\boldsymbol{\pi}}\left(\boldsymbol{o}, \boldsymbol{a}^*\right) = \sum_{i=1}^N Q_{\mathcal{I}(o_i,a_i^*)}\left(o_i, a_i^*\right). \tag{5.6}$$

Equation 5.6 recovers the exact additivity constraint that VDN applies to factorize the joint value function in meeting Equation 2.9.

Connection to QMIX. Q-DPP also generalizes QMIX, which adopts a monotonic constraint on the centralized value function to meet Equation 2.9. Under the special condition when $\{\boldsymbol{b}_j\}_{j=1}^M$ are mutually orthogonal, we can easily show that Q-DPP meets the monotonicity condition because

$$\frac{\partial Q^{\boldsymbol{\pi}}\left(\boldsymbol{o}, \boldsymbol{a}^*\right)}{\partial Q_{\mathcal{I}(o_i,a_i^*)}\left(o_i, a_i^*\right)} = 1 \geq 0, \quad \forall \mathcal{I}(o_i, a_i^*) \in \mathcal{N}. \tag{5.7}$$

Connection to QTRAN. Q-DPP also meets the sufficient conditions that QTRAN proposes for meeting Equation 2.9, that is,

$$\sum_{i=1}^N Q_i\left(o_i, a_i\right) - Q^{\boldsymbol{\pi}}(\boldsymbol{o}, \boldsymbol{a}) + V(\boldsymbol{o}) = \begin{cases} 0 & \boldsymbol{a} = \boldsymbol{a}^* \\ \geq 0 & \boldsymbol{a} \neq \boldsymbol{a}^* \end{cases}, \tag{5.8}$$

where $V(\boldsymbol{o}) = \max_{\boldsymbol{a}} Q^{\boldsymbol{\pi}}(\boldsymbol{o}, \boldsymbol{a}) - \sum_{i=1}^N Q_i\left(o_i, a_i^*\right)$. Through Equation 5.4, we know

Q-DPP can have Equation 5.8 written as

$$-\log\det\left(\boldsymbol{\mathcal{B}}_Y^\top \boldsymbol{\mathcal{B}}_Y\right) + \max_{\boldsymbol{a}} Q^{\boldsymbol{\pi}}(\boldsymbol{o}, \boldsymbol{a}) - \sum_{i=1}^{N} Q_i\big(o_i, a_i^*\big). \tag{5.9}$$

When $\boldsymbol{a} = \boldsymbol{a}^*$, for pairwise orthogonal $\{\boldsymbol{b}_j\}_{j=1}^M$, Q-DPP satisfies the first condition since Equation 5.9 equals to zero due to $\log\det(\boldsymbol{\mathcal{B}}_{Y^*}^\top \boldsymbol{\mathcal{B}}_{Y^*}) = 0$. When $\boldsymbol{a} \neq \boldsymbol{a}^*$, Equation 5.9 equals to $-\log\det\left(\boldsymbol{\mathcal{B}}_Y^\top \boldsymbol{\mathcal{B}}_Y\right) + \log\det\left(\boldsymbol{\mathcal{B}}_{Y^*}^\top \boldsymbol{\mathcal{B}}_Y^*\right)$, which is always positive since $\det(\boldsymbol{\mathcal{B}}_Y^\top \boldsymbol{\mathcal{B}}_Y) < 1, \forall Y \neq Y^*$; Q-DPP thereby meets the second condition of Equation 5.8 and recovers QTRAN.

Other Related Work. Determinantal SARSA (Osogami and Raymond, 2019) applies a normal DPP to model the ground set of the joint state-action pairs $\big\{(s^0, a_1^0, \ldots, a_N^0), \ldots, (s^{|\mathcal{S}|}, a_1^{|\mathcal{A}|}, \ldots, a_N^{|\mathcal{A}|})\big\}$. It fails to consider at all a proper ground set that suits multi-agent problems, which leads to the size of subsets being $2^{|\mathcal{S}||\mathcal{A}|^N}$ that is double-exponential to the number of agents. Furthermore, unlike Q-DPP that learns decentralized policies, Det. SARSA learns the centralized joint-action policy, which strongly limits its applicability for scalable real-world tasks.

5.2.4 Sampling from Q-DPP

Agents need to explore the environment effectively during training; however, how to sample from Q-DPPs defined in Equation 5.2 is still unknown. In fact, sampling from the DPPs with partition-matroid constraint is a non-trivial task. So far, the best known exact sampler for partitioned DPPs has $\mathcal{O}(m^p)$ time complexity with m being the ground-set size and p being the number of partitions (Celis et al., 2016; Li et al., 2016). Nonetheless, these samplers still pose great computational challenges for multi-agent learning tasks and cannot scale to large number of agents because we have $m = |\mathcal{C}(\boldsymbol{o})| = |\mathcal{A}|^N$ for multi-agent learning tasks.

In this chapter, we instead adopt a biased yet tractable sampler for Q-DPP. Our sampler is an application of the sampling-by-projection idea in Celis et al. (2018) which leverages the property that Gram-Schmidt process preserves the determinant. One benefit of our sampler is that it promotes efficient explorations among agents during training. Most importantly, it enjoys linear-time complexity *w.r.t.* the number

Algorithm 5.4 Multi-Agent Determinantal Q-Learning

```
1:  DEF Orthogonalizing-Sampler ($\mathcal{Y}, \boldsymbol{\mathcal{D}}, \boldsymbol{\mathcal{B}}, \boldsymbol{o}$):
2:  Init: $\boldsymbol{b}_j \leftarrow \boldsymbol{\mathcal{B}}_{[:,j]}, Y \leftarrow \emptyset, B \leftarrow \emptyset, J \leftarrow \emptyset$.
3:  for each partition $\mathcal{Y}_i$ do
4:      Define $\forall(o,a) \in \mathcal{Y}_i(o_i) q(o,a) := \left\|\boldsymbol{b}_{\mathcal{J}(o,a)}\right\|^2 \exp\left(\boldsymbol{\mathcal{D}}_{\mathcal{J}(o,a),\mathcal{J}(o,a)}\right)$.
5:      Sample $(\tilde{o}_i, \tilde{a}_i) \in \mathcal{Y}_i(o_i)$ from the distribution:
            $$\left\{\frac{q(o,a)}{\sum_{(\hat{o},\hat{a})\in\mathcal{Y}_i(o_i)} q(\hat{o},\hat{a})}\right\}_{(o,a)\in\mathcal{Y}_i(o_i)}.$$
6:      Let $Y \leftarrow Y \cup (\tilde{o}_i, \tilde{a}_i), B \leftarrow B \cup \boldsymbol{b}_{\mathcal{J}(\tilde{o}_i,\tilde{a}_i)}, J \leftarrow J \cup \mathcal{J}(\tilde{o}_i, \tilde{a}_i)$.
7:      // Gram-Schmidt orthogonalization
8:      Set $\boldsymbol{b}_j = \amalg_{\mathrm{span}\{B\}}(\boldsymbol{b}_j), \forall j \in \{1, ..., M\} - J$
9:  end for
10: Return: $Y$.
11:
12: DEF Determinantal-Q-Learning ($\theta = [\theta_{\boldsymbol{\mathcal{D}}}, \theta_{\boldsymbol{\mathcal{B}}}], \mathcal{Y}$):
13: Init: $\theta^- \leftarrow \theta, D \leftarrow \emptyset$.
14: for each time-step do
15:     Collect observations $\boldsymbol{o} = [o_1, \ldots, o_N]$ for all agents.
16:     $\boldsymbol{a}$ = Orthogonalizing-Sampler($\mathcal{Y}, \theta_{\boldsymbol{\mathcal{D}}}, \theta_{\boldsymbol{\mathcal{B}}}, \boldsymbol{o}$).
17:     Execute $\boldsymbol{a}$, store the transition $\langle \boldsymbol{o}, \boldsymbol{a}, \mathcal{R}, \boldsymbol{o}' \rangle$ in $D$.
18:     Sample a mini-batch of $\{\langle \boldsymbol{o}, \boldsymbol{a}, \mathcal{R}, \boldsymbol{o}' \rangle\}_{j=1}^{E}$ from $D$.
19:     Compute for each transition in the mini-batch
        $\max_{\boldsymbol{a}'} Q(\boldsymbol{o}', \boldsymbol{a}'; \theta^-) = \log\det\left(\boldsymbol{\mathcal{L}}_{Y=\{(o_1',a_1^*),\ldots,(o_N',a_N^*)\}}\right)$
20:     // centralized training
21:     Update $\theta$ by minimizing $\mathcal{L}(\theta)$ defined in Equation 2.8.
22:     Update target $\theta^- = \theta$ periodically.
23: end for
24: Return: $\theta_{\boldsymbol{\mathcal{D}}}, \theta_{\boldsymbol{\mathcal{B}}}$.
```

of agents. We now start from showing its intuition.

Additional Notations. In a Euclidean space $\mathbb{R}^n$ equipped with an inner product $\langle\cdot,\cdot\rangle$, let $\mathcal{U} \subseteq \mathbb{R}^n$ be any linear subspace, and $\mathcal{U}^\perp$ be its orthogonal complement $\mathcal{U}^\perp := \{x \in \mathbb{R}^n | \langle x, y\rangle = 0, \forall y \in \mathcal{U}\}$. We define an orthogonal projection operator, $\amalg_{\mathcal{U}} : \mathbb{R}^n \to \mathbb{R}^n$, such that $\forall \boldsymbol{u} \in \mathbb{R}^n$, if $\boldsymbol{u} = \boldsymbol{u}_1 + \boldsymbol{u}_2$ with $\boldsymbol{u}_1 \in \mathcal{U}$ and $\boldsymbol{u}_2 \in \mathcal{U}^\perp$, then $\amalg_{\mathcal{U}}(\boldsymbol{u}) = \boldsymbol{u}_2$.

Gram-Schmidt (Noble et al., 1988) is a process for orthogonalizing a set of vectors; given a set of linearly independent vectors $\{\boldsymbol{w}_i\}$, it outputs a mutually orthogonal set of vectors $\{\hat{\boldsymbol{w}}_i\}$ by computing $\hat{\boldsymbol{w}}_i := \amalg_{\mathcal{U}_i}(\boldsymbol{w}_i)$ where $\mathcal{U}_i = \operatorname{span}\{\boldsymbol{w}_1, \ldots, \boldsymbol{w}_{i-1}\}$. Note that we neglect the normalizing step of Gram-Schmidt in this chapter. Finally, if the rows $\{\boldsymbol{w}_i\}$ of a matrix $\boldsymbol{\mathcal{W}}$ are mutually orthogonal, we can compute the determinant by $\det(\boldsymbol{\mathcal{W}}\boldsymbol{\mathcal{W}}^\top) = \prod \|\boldsymbol{w}_i\|^2$. The Q-DPP sampler is built upon the following property.

Proposition 5.1 (Volume preservation of Gram-Schmidt, see Chapter 7 in Shafarevich and Remizov (2012), also Lemma 3.1 in Celis et al. (2018).)**.** *Let* $\mathcal{U}_i = \operatorname{span}\{\boldsymbol{w}_1, \ldots, \boldsymbol{w}_{i-1}\}$ *and* $\boldsymbol{w}_i \in \mathbb{R}^P$ *be the i-th row of* $\boldsymbol{\mathcal{W}} \in \mathbb{R}^{M\times P}$*, then* $\prod_{i=1}^M \| \amalg_{\mathcal{U}_i}(\boldsymbol{w}_i)\|^2 = \det(\boldsymbol{\mathcal{W}}\boldsymbol{\mathcal{W}}^\top)$.

Proof. Such property has been mentioned in linear algebra textbook, e.g., Chapter 7 in Shafarevich and Remizov (2012). Celis et al. (2018) also gave out a proof by induction[1] in Lemma 3.1. Here we provide our own intuition of such property through the classical Gaussian elimination method.

We first define an orthogonalization operator $\sqcap_{\boldsymbol{w}_i}(\boldsymbol{w}_j)$ that takes an input of a vector $\boldsymbol{w}_j \in \mathbb{R}^P$ and outputs another vector that is orthogonal to a given vector $\boldsymbol{w}_i \in \mathbb{R}^P$ by

$$\sqcap_{\boldsymbol{w}_i}(\boldsymbol{w}_j) := \boldsymbol{w}_j - \boldsymbol{w}_i\langle \boldsymbol{w}_i, \boldsymbol{w}_j\rangle / \|\boldsymbol{w}_i\|^2 \,. \tag{5.10}$$

Based on the Equation 5.10, we know that $\forall \boldsymbol{w}_i, \boldsymbol{w}_j, \boldsymbol{w}_k \in \mathbb{R}^P$,

$$\sqcap_{\boldsymbol{w}_i}(\boldsymbol{w}_j + \boldsymbol{w}_k) = \sqcap_{\boldsymbol{w}_i}(\boldsymbol{w}_j) + \sqcap_{\boldsymbol{w}_i}(\boldsymbol{w}_k).$$

Besides, we have two properties for the orthogonalization operator that will be used later; we present as lemmas.

Lemma 5.1 (Change of Projection Base)**.** *Let* $\boldsymbol{w}_i$*,* $\boldsymbol{w}_j$*,* $\boldsymbol{w}_k \in \mathbb{R}^P$*, we have* $\boldsymbol{w}_j \cdot \sqcap_{\boldsymbol{w}_i}(\boldsymbol{w}_k)^\top = \sqcap_{\boldsymbol{w}_i}(\boldsymbol{w}_j) \cdot \sqcap_{\boldsymbol{w}_i}(\boldsymbol{w}_k)^\top$.

Proof. Based the definition of Equation 5.10, one can easily write that the left hand side equals to the right hand side. ■

[1] We believe their proof is a special case, as interchanging the order of rows can actually change the determinant value, i.e., $\det(WW^\top) \neq \begin{bmatrix} w_k \\ W' \end{bmatrix}\begin{bmatrix} w_k^\top & W'^\top \end{bmatrix}$ where the row vectors are denoted as $W = \{w_1, \ldots, w_k\}$ and $W' = \{w_1, \ldots, w_{k-1}\}$.

Lemma 5.2 (Subspace Orthogonalization)**.** *Let $\boldsymbol{w}_i$, $\boldsymbol{w}_j$, $\boldsymbol{w}_k \in \mathbb{R}^P$, we have* $\sqcap_{\sqcap_{\boldsymbol{w}_i}(\boldsymbol{w}_j)}(\boldsymbol{w}_k) \cdot \sqcap_{\boldsymbol{w}_i}(\boldsymbol{w}_k)^\top = \left\|\amalg_{\mathcal{U}_k}(\boldsymbol{w}_k)\right\|^2$ *where* $\mathcal{U}_k = \operatorname{span}\{\boldsymbol{w}_i, \boldsymbol{w}_j\}$*.*

Proof. The left-hand side of equation can be written by

$$\begin{aligned}
&\sqcap_{\sqcap_{\boldsymbol{w}_i}(\boldsymbol{w}_j)}(\boldsymbol{w}_k) \cdot \sqcap_{\boldsymbol{w}_i}(\boldsymbol{w}_k)^\top \\
=&\Big(\boldsymbol{w}_k - \sqcap_{\boldsymbol{w}_i}(\boldsymbol{w}_j)\frac{\langle \sqcap_{\boldsymbol{w}_i}(\boldsymbol{w}_j), \boldsymbol{w}_k\rangle}{\|\sqcap_{\boldsymbol{w}_i}(\boldsymbol{w}_j)\|^2)}\Big) \cdot \Big(\boldsymbol{w}_k - \boldsymbol{w}_i\frac{\langle \boldsymbol{w}_i, \boldsymbol{w}_k\rangle}{\|\boldsymbol{w}_i\|^2}\Big)^\top \\
=&\, \boldsymbol{w}_k\boldsymbol{w}_k^\top - \frac{\sqcap_{\boldsymbol{w}_i}(\boldsymbol{w}_j)\cdot \boldsymbol{w}_k^\top}{\|\sqcap_{\boldsymbol{w}_i}(\boldsymbol{w}_j)\|}\langle \frac{\sqcap_{\boldsymbol{w}_i}(\boldsymbol{w}_j)}{\|\sqcap_{\boldsymbol{w}_i}(\boldsymbol{w}_j)\|}, \boldsymbol{w}_k\rangle \\
&\qquad - \frac{\boldsymbol{w}_k\cdot \boldsymbol{w}_i^\top}{\|\boldsymbol{w}_i\|}\langle \frac{\boldsymbol{w}_i^\top}{\|\boldsymbol{w}_i\|}, \boldsymbol{w}_k\rangle \ .
\end{aligned} \tag{5.11}$$

On the other hand, $\amalg_{\mathcal{U}_k}(\boldsymbol{w}_k)$ represents the orthogonal projection of $\boldsymbol{w}_k$ to the subspace that is spanned by $\boldsymbol{w}_i$ and $\boldsymbol{w}_j$. Since $\left\{\frac{\boldsymbol{w}_i}{\|\boldsymbol{w}_i\|}, \frac{\sqcap_{\boldsymbol{w}_i}(\boldsymbol{w}_j)}{\|\sqcap_{\boldsymbol{w}_i}(\boldsymbol{w}_j)\|}\right\}$ form a set of orthornormal basis for the subspace $\mathcal{U}_k = \operatorname{span}\{\boldsymbol{w}_i, \boldsymbol{w}_j\}$, according to the definition of $\amalg_{\mathcal{U}_k}(\boldsymbol{w}_k)$ in Section 5.2.4, we can write $\amalg_{\mathcal{U}_k}(\boldsymbol{w}_k)$ as $\boldsymbol{w}_k$ minus the projection of $\boldsymbol{w}_k$ on the subspace that is spanned by $\left\{\frac{\boldsymbol{w}_i}{\|\boldsymbol{w}_i\|}, \frac{\sqcap_{\boldsymbol{w}_i}(\boldsymbol{w}_j)}{\|\sqcap_{\boldsymbol{w}_i}(\boldsymbol{w}_j)\|}\right\}$, i.e.,

$$\begin{aligned}
\amalg_{\mathcal{U}_k}(\boldsymbol{w}_k) = \boldsymbol{w}_k \ &- \ \langle \boldsymbol{w}_k, \frac{\boldsymbol{w}_i}{\|\boldsymbol{w}_i\|}\rangle \frac{\boldsymbol{w}_i}{\|\boldsymbol{w}_i\|} \\
&- \ \langle \boldsymbol{w}_k, \frac{\sqcap_{\boldsymbol{w}_i}(\boldsymbol{w}_j)}{\|\sqcap_{\boldsymbol{w}_i}(\boldsymbol{w}_j)\|}\rangle \frac{\sqcap_{\boldsymbol{w}_i}(\boldsymbol{w}_j)}{\|\sqcap_{\boldsymbol{w}_i}(\boldsymbol{w}_j)\|}.
\end{aligned} \tag{5.12}$$

Under the orthonormal property of $\frac{\boldsymbol{w}_i}{\|\boldsymbol{w}_i\|} \cdot \frac{\sqcap_{\boldsymbol{w}_i}(\boldsymbol{w}_j)}{\|\sqcap_{\boldsymbol{w}_i}(\boldsymbol{w}_j)\|}^\top = 0$ and $\left\|\frac{\boldsymbol{w}_i}{\|\boldsymbol{w}_i\|}\right\|^2 = \left\|\frac{\sqcap_{\boldsymbol{w}_i}(\boldsymbol{w}_j)}{\|\sqcap_{\boldsymbol{w}_i}(\boldsymbol{w}_j)\|}\right\|^2 = 1$, finally, squaring the Equation 5.12 from both sides leads us to the Equation 5.11, i.e., $\left\|\amalg_{\mathcal{U}_k}(\boldsymbol{w}_k)\right\|^2 = \sqcap_{\sqcap_{\boldsymbol{w}_i}(\boldsymbol{w}_j)}(\boldsymbol{w}_k) \cdot \sqcap_{\boldsymbol{w}_i}(\boldsymbol{w}_k)^\top$. ■

Assuming $\{\boldsymbol{w}_1, \dots, \boldsymbol{w}_M\}$ being the rows of $\boldsymbol{\mathcal{W}}$, then applying the Gram-Schmidt orthogonalization process gives

$$\text{Gram-Schmidt}\Big(\{\boldsymbol{w}_i\}_{i=1}^M\Big) = \Big\{\amalg_{\mathcal{U}_i}(\boldsymbol{w}_i)\Big\}_{i=1}^M$$

where $\mathcal{U}_i = \operatorname{span}\{\boldsymbol{w}_1, \dots, \boldsymbol{w}_{i-1}\}$. Note that we don't consider normalizing each $\amalg_{\mathcal{U}_i}(\boldsymbol{w}_i)$ in this chapter.

In fact, the effect on the Gram matrix determinant $\det(\boldsymbol{\mathcal{W}}\boldsymbol{\mathcal{W}}^\top)$ of applying the Gram-Schmidt process on the rows of $\boldsymbol{\mathcal{W}}$ is equivalent to applying Gaussian elimination (Noble et al., 1988) to transform the Gram matrix to be upper triangular. Since adding a row/column of a matrix multiplied by a scalar to another row/column of that matrix will not change the determinant value of the original matrix (Noble et al., 1988), Gaussian elimination, so as the Gram-Schmidt process, preserves the determinant.

To illustrate the above equivalence, we demonstrate the Gaussian elimination process step-by-step on the case of $M = 3$, the determinant of such a Gram matrix is

$$\det\left(\boldsymbol{\mathcal{W}}\boldsymbol{\mathcal{W}}^\top\right) = \det\begin{pmatrix} \boldsymbol{w}_1\boldsymbol{w}_1^\top & \boldsymbol{w}_1\boldsymbol{w}_2^\top & \boldsymbol{w}_1\boldsymbol{w}_3^\top \\ \boldsymbol{w}_2\boldsymbol{w}_1^\top & \boldsymbol{w}_2\boldsymbol{w}_2^\top & \boldsymbol{w}_2\boldsymbol{w}_3^\top \\ \boldsymbol{w}_3\boldsymbol{w}_1^\top & \boldsymbol{w}_3\boldsymbol{w}_2^\top & \boldsymbol{w}_3\boldsymbol{w}_3^\top \end{pmatrix}. \tag{5.13}$$

To apply Gaussian elimination to turn the Gram matrix to be upper triangular, first, we multiply the 1-st row by $-\frac{\boldsymbol{w}_2\boldsymbol{w}_1^\top}{\boldsymbol{w}_1\boldsymbol{w}_1^\top}$ and then add the result to the 2-nd row; without affecting the determinant, we have the 2-nd row transformed into

$$\begin{aligned} &\left[0, \boldsymbol{w}_2\boldsymbol{w}_2^\top - \frac{\boldsymbol{w}_2\boldsymbol{w}_1^\top}{\boldsymbol{w}_1\boldsymbol{w}_1^\top}\boldsymbol{w}_1\boldsymbol{w}_2^\top, \boldsymbol{w}_2\boldsymbol{w}_3^\top - \frac{\boldsymbol{w}_2\boldsymbol{w}_1^\top}{\boldsymbol{w}_1\boldsymbol{w}_1^\top}\boldsymbol{w}_1\boldsymbol{w}_3^\top\right] \\ =&\left[0, \boldsymbol{w}_2 \cdot \sqcap_{\boldsymbol{w}_1}(\boldsymbol{w}_2)^\top, \boldsymbol{w}_3 \cdot \sqcap_{\boldsymbol{w}_1}(\boldsymbol{w}_2)^\top\right] \\ =&\left[0, \sqcap_{\boldsymbol{w}_1}(\boldsymbol{w}_2) \cdot \sqcap_{\boldsymbol{w}_1}(\boldsymbol{w}_2)^\top, \sqcap_{\boldsymbol{w}_1}(\boldsymbol{w}_3) \cdot \sqcap_{\boldsymbol{w}_1}(\boldsymbol{w}_2)^\top\right]. \quad (\textit{Lemma 5.1}) \end{aligned} \tag{5.14}$$

Similarly, we can apply the same process on the 3-rd row, which can be written as

$$\begin{aligned} &\left[0, \boldsymbol{w}_3\boldsymbol{w}_2^\top - \frac{\boldsymbol{w}_2\boldsymbol{w}_1^\top}{\boldsymbol{w}_1\boldsymbol{w}_1^\top}\boldsymbol{w}_1\boldsymbol{w}_2^\top, \boldsymbol{w}_3\boldsymbol{w}_3^\top - \frac{\boldsymbol{w}_2\boldsymbol{w}_1^\top}{\boldsymbol{w}_1\boldsymbol{w}_1^\top}\boldsymbol{w}_1\boldsymbol{w}_3^\top\right] \\ =&\left[0, \sqcap_{\boldsymbol{w}_1}(\boldsymbol{w}_2) \cdot \sqcap_{\boldsymbol{w}_1}(\boldsymbol{w}_3)^\top, \sqcap_{\boldsymbol{w}_1}(\boldsymbol{w}_3) \cdot \sqcap_{\boldsymbol{w}_1}(\boldsymbol{w}_3)^\top\right]. \end{aligned} \tag{5.15}$$

To make $\boldsymbol{\mathcal{W}}\boldsymbol{\mathcal{W}}^\top$ upper triangular, we need to make the 2-nd element in the 3-rd row be zero. To achieve that, we multiply $-\frac{\sqcap_{\boldsymbol{w}_1}(\boldsymbol{w}_2)\cdot\sqcap_{\boldsymbol{w}_1}(\boldsymbol{w}_3)^\top}{\sqcap_{\boldsymbol{w}_1}(\boldsymbol{w}_2)\cdot\sqcap_{\boldsymbol{w}_1}(\boldsymbol{w}_2)^\top}$ to Equation 5.14 and add the multiplication to Equation 5.15, and the 3-rd row can be further transformed into

$$\begin{aligned}
&\Big[0, 0, \sqcap_{\boldsymbol{w}_1}(\boldsymbol{w}_3) \cdot \sqcap_{\boldsymbol{w}_1}(\boldsymbol{w}_3)^\top - \\
&\qquad \frac{\sqcap_{\boldsymbol{w}_1}(\boldsymbol{w}_2) \cdot \sqcap_{\boldsymbol{w}_1}(\boldsymbol{w}_3)^\top}{\sqcap_{\boldsymbol{w}_1}(\boldsymbol{w}_2) \cdot \sqcap_{\boldsymbol{w}_1}(\boldsymbol{w}_2)^\top} \sqcap_{\boldsymbol{w}_1}(\boldsymbol{w}_3) \cdot \sqcap_{\boldsymbol{w}_1}(\boldsymbol{w}_2)^\top\Big] \\
=&\Big[0, 0, \sqcap_{\boldsymbol{w}_1}(\boldsymbol{w}_3) \cdot \sqcap_{\sqcap_{\boldsymbol{w}_1}(\boldsymbol{w}_2)}\big(\sqcap_{\boldsymbol{w}_1}(\boldsymbol{w}_3)\big)^\top\Big] \\
=&\Big[0, 0, \sqcap_{\boldsymbol{w}_1}(\boldsymbol{w}_3) \cdot \sqcap_{\sqcap_{\boldsymbol{w}_1}(\boldsymbol{w}_2)}\Big(\boldsymbol{w}_3 - \boldsymbol{w}_1 \frac{\langle \boldsymbol{w}_1, \boldsymbol{w}_3\rangle}{\|\boldsymbol{w}_1\|^2}\Big)^\top\Big] \\
=&\Big[0, 0, \sqcap_{\boldsymbol{w}_1}(\boldsymbol{w}_3) \cdot \Big(\sqcap_{\sqcap_{\boldsymbol{w}_1}(\boldsymbol{w}_2)}(\boldsymbol{w}_3) \\
&\qquad - \sqcap_{\sqcap_{\boldsymbol{w}_1}(\boldsymbol{w}_2)}\big(\boldsymbol{w}_1 \frac{\langle \boldsymbol{w}_1, \boldsymbol{w}_3\rangle}{\|\boldsymbol{w}_1\|^2}\big)\Big)^\top\Big] \\
=&\Big[0, 0, \sqcap_{\boldsymbol{w}_1}(\boldsymbol{w}_3) \cdot \sqcap_{\sqcap_{\boldsymbol{w}_1}(\boldsymbol{w}_2)}(\boldsymbol{w}_3)^\top\Big] \\
=&\Big[0, 0, \big\|\amalg_{\mathcal{U}_3}(\boldsymbol{w}_3)\big\|^2\Big]. \qquad (\textit{Lemma } 5.2)
\end{aligned} \tag{5.16}$$

In the fourth equation of Eq.5.16, we use the property that $\sqcap_{\boldsymbol{w}_1}(\cdot) \cdot \sqcap_{\sqcap_{\boldsymbol{w}_1}(\cdot)}(\boldsymbol{w}_1)^\top = 0$, i.e., the inner product between a vector and its own orthogonalization equals to zero.

Given the Gran matrix is now upper triangular, by putting Equation 5.14 and Equation 5.16 into Equation 5.13, and define $\mathcal{U}_1 = \emptyset, \mathcal{U}_2 = \{\boldsymbol{w}_1\}, \mathcal{U}_3 = \{\boldsymbol{w}_1, \boldsymbol{w}_2\}$, we can write the determinant to be

$$\begin{aligned}
&\det\left(\boldsymbol{\mathcal{W}}\boldsymbol{\mathcal{W}}^\top\right) \\
=& \det\begin{pmatrix} \boldsymbol{w}_1\boldsymbol{w}_1^\top & \boldsymbol{w}_1\boldsymbol{w}_2^\top & \boldsymbol{w}_1\boldsymbol{w}_3^\top \\ 0 & \big\|\sqcap_{\boldsymbol{w}_1}(\boldsymbol{w}_2)\big\|^2 & \sqcap_{\boldsymbol{w}_1}(\boldsymbol{w}_3) \cdot \sqcap_{\boldsymbol{w}_1}(\boldsymbol{w}_2)^\top \\ 0 & 0 & \big\|\amalg_{\mathcal{U}_3}(\boldsymbol{w}_3)\big\|^2 \end{pmatrix} \\
=& \prod_{i=1}^{3} \Big\|\amalg_{\mathcal{U}_i}(\boldsymbol{w}_i)\Big\|^2 \quad .
\end{aligned}$$

When $M \geq 3$, the consequence of eliminating all j-th elements ($j < i$) in the i-th row of the Gram matrix $\boldsymbol{\mathcal{W}}\boldsymbol{\mathcal{W}}^\top_{(i,j)}$ by Gaussian elimination is equivalent to the i-th step of the Gran-Schmidt process applied on the vector set $\{\boldsymbol{w}_i\}_{i=1}^{M}$, in other words, the (i,i)-th element of the Gram matrix after Gaussian elimination is essentially the squared norm of $\amalg_{\mathcal{U}_i}(\boldsymbol{w}_i)$. Finally, since the determinant of an upper-triangular matrix is simply the multiplication of its diagonal elements, we have $\prod_{i=1}^{M} \left\| \amalg_{\mathcal{U}_i}(\boldsymbol{w}_i) \right\|^2$. ■

Proposition 5.1 suggests that the determinant of a Gram matrix is invariant to applying the Gram-Schmidt orthogonalization on the rows of that Gram matrix. In Q-DPP's case, a kernel matrix with mutually orthogonal rows can largely simplify the sampling process. In such scenarios, an effective sampler can be that, from each partition $\mathcal{Y}_i$, sample an item $i \in \mathcal{Y}_i$ with $\mathbb{P}(i) \propto \|d_i \boldsymbol{b}_i^\top\|^2$, then add i to the output sample Y and move to the next partition; the above steps iterate until all partitions are covered. It is effortless to see that the probability of obtaining sample Y in such a way is

$$\begin{aligned}\mathbb{P}(Y) \propto \prod_{i\in Y} \|d_i \boldsymbol{b}_i^\top\|^2 = \prod_{i\in Y} \|\boldsymbol{w}_i\|^2 = \det(\boldsymbol{\mathcal{W}}_Y \boldsymbol{\mathcal{W}}_Y^\top) \\ \propto \det(\boldsymbol{\mathcal{L}}_Y). \end{aligned} \tag{5.17}$$

We formally describe the orthogonalizing sampling procedures in Algorithm 5.4. As it is suggested in Celis et al. (2018), the time complexity of the sampling function is $\mathcal{O}(NMP)$, given the input size being $\mathcal{O}(MP)$, our sampler thus enjoys linear-time complexity *w.r.t* the agent number. The following is the details complexity analysis:

Time Complexity of Algorithm: Let's analyze the time complexity of the proposed Q-DPP sampler in steps $1-10$ of Algorithm 5.4. Given the observation o, and the input matrices $\boldsymbol{\mathcal{D}}, \boldsymbol{\mathcal{B}}$ (whose sizes are $M \times M$, $P \times M$, with $M = |A| \times N$ being the size of all N agents' allowed actions under o and P being the diversity feature dimension), the sampler samples one action for each agent sequentially, so the outer loop of step 3 is $\mathcal{O}(N)$. Within the partition of each agent, step 4 is $\mathcal{O}(P)$, step 5 is $\mathcal{O}(P|A|)$, step 6 is $\mathcal{O}(1)$, so the complexity so far is $\mathcal{O}(NP|A|)$. Computing step 8 for ALL partitions is of $\mathcal{O}(N^2P|A|)^\dagger$. The overall complexity is $\mathcal{O}(N^2P|A|) = \mathcal{O}(NMP)$, since the input is $\mathcal{O}(MP)$ and the agent number N is a constant, our sampler has linear-time complexity with respect to the input, also linear-time with respect to the number of agents. Such argument is in line with the project-and-sample sampler in Celis et al. (2018). †: In the Gram-Schmidt process, orthogonalizing a vector to another takes $\mathcal{O}(P)$. Considering all valid actions for each agent takes $\mathcal{O}(P|A|)$. Note that while looping over different partitions, the remaining unsampled partitions do not need repeatedly orthogonalizing to all the previous samples, in fact, they only need orthogonalizing to the LATEST sample. In the example of Fig 2, after agent 2 selects action 5, agent 3's three actions only need orthogonalizing to action 5 but not action 2 because it has been performed when the partition of agent 1 was visited. So the total number of orthogonalization is $(N-1)N/2$ across all partitions, leading to $\mathcal{O}(N^2P|A|)$ time for step 8.

Though the Gram-Schmidt process can preserve the determinant and simply the sampling process, it comes at a prize of introducing **bias** on the normalization term. Specifically, the normalization in our proposed sampler is conducted at each agent/partition level $\mathcal{Y}_i(o_i)$ (see the red in line 5) which does not match Equation 5.2 that suggests normalizing by listing all valid samples considering all partitions $\mathcal{C}(\boldsymbol{o})$; this directly leads to a sampled subset from our sampler having *larger* probability than what Q-DPP defines. Interestingly, it turns out that such violation can be controlled through bounding the singular values of each partition in the kernel matrix (see Assumption 5.1), a technique also known as the β-balance condition in P-DPP (Celis et al., 2018).

Assumption 5.1 (Singular-Value Constraint on Partitions)**.** *For a Q-DPP defined in Definition 5.1, which is parameterized by* $\boldsymbol{\mathcal{D}} \in \mathbb{R}^{M\times M}$, $\boldsymbol{\mathcal{B}} \in \mathbb{R}^{P\times M}$ *and* $\boldsymbol{\mathcal{W}} := \boldsymbol{\mathcal{D}}\boldsymbol{\mathcal{B}}^\top$, *let* $\sigma_1 \geq \ldots \geq \sigma_P$ *represent the singular values of* $\boldsymbol{\mathcal{W}}$, *and* $\hat{\sigma}_{i,1} \geq \ldots \geq \hat{\sigma}_{i,P}$ *denote the singular values of* $\boldsymbol{\mathcal{W}}_{\mathcal{Y}_i}$ *that is the submatrix of* $\boldsymbol{\mathcal{W}}$ *with the rows and columns corresponding to the* i*-th partition* $\mathcal{Y}_i$ *, we assume* $\forall j \in \{1,\ldots,P\}$, $\exists\, \delta \in (0,1]$ *, s.t.,* $\min_{i\in\{1,\ldots,N\}} \hat{\sigma}^2_{i,j}/\delta \geq \sigma^2_j$ *holds.*

Theorem 5.1 (Approximation Guarantee of Orthogonalizing Sampler)**.** *For a Q-DPP defined in Definition 5.1, under Assumption 5.1, the Orthogonalizing Sampler described in Algorithm 5.4 returns a sampled subset* $Y \in \mathcal{C}(\boldsymbol{o})$ *with probability* $\mathbb{P}(Y) \leq 1/\delta^N \cdot \tilde{\mathbb{P}}(\boldsymbol{Y} = Y)$ *where* N *is the number of agents,* $\tilde{\mathbb{P}}$ *is defined in Equation 5.2,* δ *is defined in Assumption 5.1.*

Proof. This result can be regarded as a special case of Theorem 3.2 in Celis et al.

(2018) when the number of sample from each partition in P-DPP is set to one.

Difference between Q-DPP and P-DPP: The design of Q-DPP and its samply-by-projection sampling process is inspired by and based on P-DPP (Celis et al., 2018). However, we would like to highlight the multiple differences in that **1)** P-DPP is designed for modeling the fairness for data summarization whereas Q-DPP serves as a function approximator for the joint Q-function in the context of multi-agent learning; **2)** though we analyze Equation 5.19 based on $\boldsymbol{\mathcal{W}}$, the actual orthorgonalziation step of our sampler only needs performing on the vectors of $\boldsymbol{b}_j$ rather than the entire matrix $\boldsymbol{\mathcal{W}}$ due to our unique quality-diversity decomposition on the joint Q-function in Equation 5.4; **3)** the set of elements in each partition $\mathcal{Y}_i(o_i)$ of Q-DPP change with the observation at each time-step, while the partitions stay fixed in the case of P-DPP; **4)** the parameters of $\boldsymbol{\mathcal{W}}$ are learned through a trail-and-error multi-agent reinforcement learning process compared to the cases in P-DPP where the kernel is given by hand-crafted features (e.g. SIFT features on images); **5)** we implement the constraint in Assumption 5.1 via a penalty term during the CTDE learning process, while P-DPP does not consider meeting such assumption through optimization.

Sine our sampling algorithm generates samples with the probability in proportional to the determinant value $\det(\boldsymbol{\mathcal{L}}_Y)$, which is also the nominator in Equation 5.2, it is then necessary to bound the denominator of the probability of samples from our proposed sampler so that the error to the exact denominator defined in Equation 5.2 can be controlled. We start from the Lemma that is going to be used.

Lemma 5.3 (Eckart-Young-Mirsky Theorem)**.** *For a real matrix $\boldsymbol{\mathcal{W}} \in \mathbb{R}^{M\times P}$ with $M \geq P$, suppose that $\boldsymbol{\mathcal{W}} = \boldsymbol{U}\Sigma\boldsymbol{V}^\top$ is the singular value decomposition (SVD) of $\boldsymbol{\mathcal{W}}$, then the best rank k approximation to $\boldsymbol{\mathcal{W}}$ under the Frobenius norm $\|\cdot\|_F$ described as*

$$\min_{\boldsymbol{\mathcal{W}}':\mathrm{rank}(\boldsymbol{\mathcal{W}}')=k} \left\|\boldsymbol{\mathcal{W}} - \boldsymbol{\mathcal{W}}'\right\|_F^2$$

is given by $\boldsymbol{\mathcal{W}}' = \boldsymbol{\mathcal{W}}^k = \sum_{i=1}^k \sigma_i \boldsymbol{u}_i \boldsymbol{v}_i^\top$ where $\boldsymbol{u}_i$ and $\boldsymbol{v}_i$ denote the i-th column of

$\boldsymbol{U}$ and $\boldsymbol{V}$ respectively, and,

$$\left\|\boldsymbol{\mathcal{W}} - \boldsymbol{\mathcal{W}}^k\right\|_F^2 = \left\| \sum_{i=k+1}^{P} \sigma_i \boldsymbol{u}_i \boldsymbol{v}_i^\top \right\|_F^2 = \sum_{i=k+1}^{P} \sigma_i^2.$$

Note that the singular values σ_i in Σ is ranked by size by the SVD procedures such that $\sigma_1 \geq \ldots \geq \sigma_P$.

Lemma 5.4 (Lemma 3.1 in (Deshpande et al., 2006))**.** *For a matrix $\boldsymbol{\mathcal{W}} \in \mathbb{R}^{M \times P}$ with $M \geq P \geq N$, assume $\{\sigma_i\}_{i=1}^P$ are the singular values of $\boldsymbol{\mathcal{W}}$ and $\boldsymbol{\mathcal{W}}_Y$ is the submatrix of $\boldsymbol{\mathcal{W}}$ with rows indexed by the elements in Y, then we have*

$$\sum_{|Y|=N} \det\left(\boldsymbol{\mathcal{W}}_Y \boldsymbol{\mathcal{W}}_Y^\top\right) = \sum_{k_1 < \cdots < k_N} \sigma_{k_1}^2 \cdots \sigma_{k_N}^2.$$

To stay consistency on notations, we use N for number of agents, M for the size of ground set of Q-DPP, P is the dimension of diverse feature vectors, we assume $M \geq P \geq N$. Let $\boldsymbol{Y}$ be the random variable representing the output of our proposed sampler in Algorithm 5.4. Since the algorithm visit each partition in Q-DPP sequentially, a sample $\tilde{Y} = \left\{(o_1^1, a_1^1), \ldots, (o_N^{|\mathcal{O}|}, a_N^{|\mathcal{A}|})\right\}$ is therefore an ordered set. Note that the algorithm is agnostic to the partition number (i.e. the agent identity), without losing generality, we denote the first partition chosen as $\mathcal{Y}_1$. We further denote $\tilde{Y}_i, i \in \{1, \ldots, N\}$ as the i-th observation-state pair in $\tilde{Y}$, and $\mathcal{I}(\tilde{Y}_i) \in \{1, \ldots, N\}$ denotes the partition number where i-th pair is sampled.

According to the Algorithm 5.4, at first step, we choose $\mathcal{Y}_1$, and based on the corresponding observation o_1, we then locate the valid subsets $\forall (o, a) \in \mathcal{Y}_i(o_i)$, and finally sample one observation-action pair from the valid set $\mathcal{Y}_i(o_i)$ with probability proportional to the norm of the vector defined in the Line $4-5$ in Algorithm 5.4, that is,

$$\mathbb{P}(\tilde{Y}_i) \propto \left\|\boldsymbol{w}_{\mathcal{J}(o,a)}\right\|^2 = \left\|\boldsymbol{b}_{\mathcal{J}(o,a)}\right\|^2 \exp\left(\boldsymbol{\mathcal{D}}_{\mathcal{J}(o,a),\mathcal{J}(o,a)}\right). \tag{5.18}$$

After $\tilde{Y}_i$ is sampled, the algorithm then moves to the next partition and repeat the same process until all N partitions are covered.

The specialty of this sampler is that before sampling at each partition $i \in$

$\{1, \ldots, N\}$, the Gram-Schmidt process will be applied to ensure all the rows in the i-th partition of $\boldsymbol{\mathcal{W}}$ to be orthogonal to all previous sampled pairs

$$\boldsymbol{b}_j^i = \amalg_{\text{span}\{B^i\}}\left(\boldsymbol{b}_j^{i-1}\right), \forall j \in \{1, ..., M\} - J.$$

where $B^i = \{\boldsymbol{b}^t_{\mathcal{J}(o^t, a^t)}\}_{t=1}^{i-1}$, $J = \{\mathcal{J}(o^t, a^t)\}_{t=1}^{i-1}$. Note that since $\boldsymbol{\mathcal{D}}$ only contributes a scalar to $\boldsymbol{w}_j$, and $\boldsymbol{b}_j$ is a P-dimensional vector same as $\boldsymbol{w}_j$, in practice, the Gram-Schmidt orthorgonalization needs only conducting on $\boldsymbol{b}_j$ in order to make rows of $\boldsymbol{\mathcal{W}}$ mutually orthogonal.

Based on the above sampling process and each time-step i, we can write the probability of getting a sample $\tilde{Y}$ by

$$\begin{aligned}
&\mathbb{P}(\boldsymbol{Y} = \tilde{Y}) \\
&= \mathbb{P}(\tilde{Y}_1) \prod_{i=2}^{N} \mathbb{P}(\tilde{Y}_i | \tilde{Y}_1, \ldots, \tilde{Y}_{i-1}) \\
&= \prod_{i=1}^{N} \frac{\left\| \amalg_{\text{span}\{B^i\}}\left(\boldsymbol{w}_{\mathcal{I}(\tilde{Y}_i)}\right) \right\|^2}{\sum_{(o,a)\in\mathcal{Y}_{\mathcal{I}(\tilde{Y}_i)}} \left\| \amalg_{\text{span}\{B^i\}}\left(\boldsymbol{w}_{\mathcal{I}(o,a)}\right) \right\|^2} \\
&= \frac{\prod_{i=1}^{N} \left(\left\| \amalg_{\text{span}\{B^i\}}\left(\boldsymbol{w}_{\mathcal{I}(\tilde{Y}_i)}\right) \right\|^2 \right)}{\prod_{i=1}^{N} \left(\sum_{(o,a)\in\mathcal{Y}_{\mathcal{I}(\tilde{Y}_i)}} \left\| \amalg_{\text{span}\{B^i\}}\left(\boldsymbol{w}_{\mathcal{I}(o,a)}\right) \right\|^2 \right)} \\
&= \frac{\det\left(\boldsymbol{\mathcal{W}}_{\tilde{Y}} \boldsymbol{\mathcal{W}}_{\tilde{Y}}^\top\right)}{\prod_{i=1}^{N} \left(\sum_{(o,a)\in\mathcal{Y}_{\mathcal{I}(\tilde{Y}_i)}} \left\| \amalg_{\text{span}\{B^i\}}\left(\boldsymbol{w}_{\mathcal{I}(o,a)}\right) \right\|^2 \right)},
\end{aligned} \tag{5.19}$$

where the 4-th equation in Equation 5.19 is valid because of Proposition 5.1.

For each term in the denominator, according to the definition of the operator $\amalg_{\text{span}\{B^i\}}$, we can rewrite into

$$\sum_{(o,a)\in\mathcal{Y}_{\mathcal{I}(\tilde{Y}_i)}} \left\| \amalg_{\text{span}\{B^i\}}\left(\boldsymbol{w}_{\mathcal{I}(o,a)}\right) \right\|^2 = \left\| \boldsymbol{\mathcal{W}}_{\mathcal{I}(\tilde{Y}_i)} - \boldsymbol{\mathcal{W}}'_{\mathcal{I}(\tilde{Y}_i)} \right\|_F^2$$

where the rows of $\boldsymbol{\mathcal{W}}_{\mathcal{I}(\tilde{Y}_i)}$ are $\{\boldsymbol{w}_{\mathcal{I}(o,a)}\}_{(o,a)\in\mathcal{Y}_{\mathcal{I}(\tilde{Y}_i)}}$ which are essentially the submatrix of $\boldsymbol{\mathcal{W}}$ that corresponds to partition $\mathcal{I}(\tilde{Y}_i)$, and the rows of $\boldsymbol{\mathcal{W}}'_{\mathcal{I}(\tilde{Y}_i)}$ are the orthogonal projections of $\{\boldsymbol{w}_{\mathcal{I}(o,a)}\}_{(o,a)\in\mathcal{Y}_{\mathcal{I}}}$ onto $\operatorname{span}\{B^i\}$, and we know $\operatorname{rank}\left(\boldsymbol{\mathcal{W}}'_{\mathcal{I}(\tilde{Y}_i)}\right) = |B^i| = i-1$. According to Lemma 5.3, with $\hat{\sigma}_{\mathcal{I}(\tilde{Y}_i),k}$ being the k-th singular value of $\boldsymbol{\mathcal{W}}_{\mathcal{I}(\tilde{Y}_i)}$, we know that

$$\left\|\boldsymbol{\mathcal{W}}_{\mathcal{I}(\tilde{Y}_i)} - \boldsymbol{\mathcal{W}}'_{\mathcal{I}(\tilde{Y}_i)}\right\|_F^2 \geq \sum_{k=i}^{P} \hat{\sigma}^2_{\mathcal{I}(\tilde{Y}_i),k} \quad . \tag{5.20}$$

Therefore, we have the denominator of Equation 5.19 as:

$$\begin{aligned}
&\prod_{i=1}^{N}\Big(\sum_{(o,a)\in\mathcal{Y}_{\mathcal{I}(\tilde{Y}_i)}} \left\| \Pi_{\operatorname{span}\{B^i\}}\left(\boldsymbol{w}_{\mathcal{I}(o,a)}\right) \right\|^2 \Big) \\
&\geq \prod_{i=1}^{N}\sum_{k=i}^{P} \hat{\sigma}^2_{\mathcal{I}(\tilde{Y}_i),k} \;\geq\; \prod_{i=1}^{N}\sum_{k=i}^{P} \delta\cdot\sigma_k^2 \qquad (\textit{Assumption 5.1}) \\
&\geq\; \delta^N \cdot \sum_{k_1<\cdots<k_N} \sigma^2_{k_1}\cdots\sigma^2_{k_N} \\
&= \delta^N \cdot \sum_{Y\subseteq\mathcal{Y}:|Y|=N} \det\left(\boldsymbol{\mathcal{W}}_Y\boldsymbol{\mathcal{W}}_Y^{\top}\right) \qquad (\textit{Lemma 5.4}) \\
&\geq\; \delta^N \cdot \sum_{Y\in\mathcal{C}(\boldsymbol{o})} \det\left(\boldsymbol{\mathcal{W}}_Y\boldsymbol{\mathcal{W}}_Y^{\top}\right)
\end{aligned} \tag{5.21}$$

Taking Equation 5.21 into Equation 5.19, we can obtain that

$$\mathbb{P}(\boldsymbol{Y}=\tilde{Y}) \leq \frac{\delta^N\cdot\det\left(\boldsymbol{\mathcal{W}}_{\tilde{Y}}\boldsymbol{\mathcal{W}}_{\tilde{Y}}^{\top}\right)}{\sum_{Y\in\mathcal{C}(\boldsymbol{o})}\det\left(\boldsymbol{\mathcal{W}}_Y\boldsymbol{\mathcal{W}}_Y^{\top}\right)} = 1/\delta^N\cdot\tilde{\mathbb{P}}(\boldsymbol{Y}=\tilde{Y})$$

where $\mathbb{P}(\boldsymbol{Y}=\tilde{Y})$ is the probability of obtaining the sample $\tilde{Y}$ from our proposed sampler and $\tilde{\mathbb{P}}(\boldsymbol{Y}=\tilde{Y})$ is the probability of getting that sample $\tilde{Y}$ under Equation 5.2.

■

Theorem 5.1 effectively suggests a way to bound the error between our sampler and the true distribution of Q-DPP through minimizing the difference between σ_j^2 and $\hat{\sigma}^2_{i,j}$.

5.2.5 Determinantal Q-Learning

We present the full learning procedures in Algorithm 5.4. Determinantal Q-Learning is a CTDE method. During training, agents' explorations are conducted through the orthogonalizing-sampler. The parameters of $\mathcal{B}$ and $\mathcal{D}$ are updated through Q-learning in a centralized way by following Equation 2.8. To meet Assumption 5.1, one can implement an auxiliary loss function of $\max(0, \sigma_j^2 - \hat{\sigma}_{i,j}^2/\delta)$ in addition to where δ is a hyper-parameter. Given Theorem 5.1, for large N, we know δ should be set close to 1 to make the bound tight. In fact, it is worth mentioning that the Gram-Schmidt process adopted in the sampler can boost the sampling efficiency for multi-agent training. Since agents' diversity features of observation-action pairs are orthogonalized every time after a partition is visited, agents who act later are essentially coordinated to explore the observation-action space that is distinctive to all previous agents. This speeds up training in early stages.

During execution, agents only need to access the parameters in their own partitions to compute the greedy action (see line 19). Note that neural networks can be seamlessly applied to represent both $\mathcal{B}$ and $\mathcal{D}$ to tackle continuous states. Though a full treatment of deep Q-DPP needs substantial future work, we show a proof of concept in *Section 5.3*. Hereafter, we use Q-DPP to represent our proposed algorithm.

5.3 Solution for Continuous States: Deep Q-DPP

Although our proposed Q-DPP serves as a new type of function approximator for the value function in multi-agent reinforcement learning, deep neural networks can also be seamlessly applied on Q-DPP. Specifically, one can adopt deep networks to respectively represent the quality and diversity terms in the kernels of Q-DPP to tackle continuous state-action space, and we name such approach Deep Q-DPP. In Figure 5.2, one can think of Deep Q-DPP as modeling $\mathcal{D}$ and $\mathcal{B}$ by neural networks rather than look-up tables. An analogy of Deep Q-DPP to Q-DPP would be Deep Q-learning (Mnih et al., 2015) to Q-learning (Watkins and Dayan, 1992). As the main motivation of introducing Q-DPP is to eliminate structural constraints and bespoke

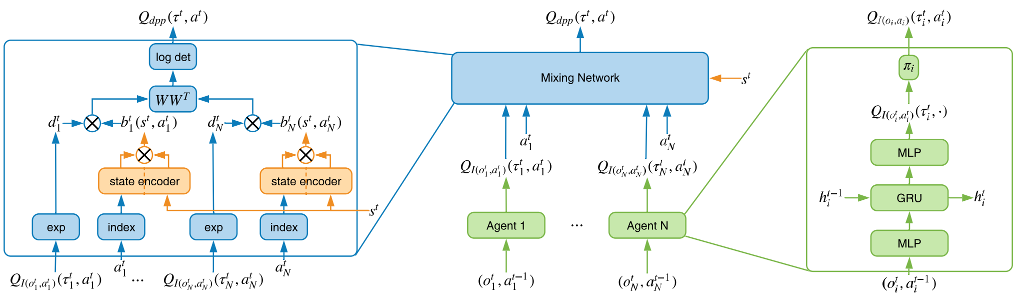
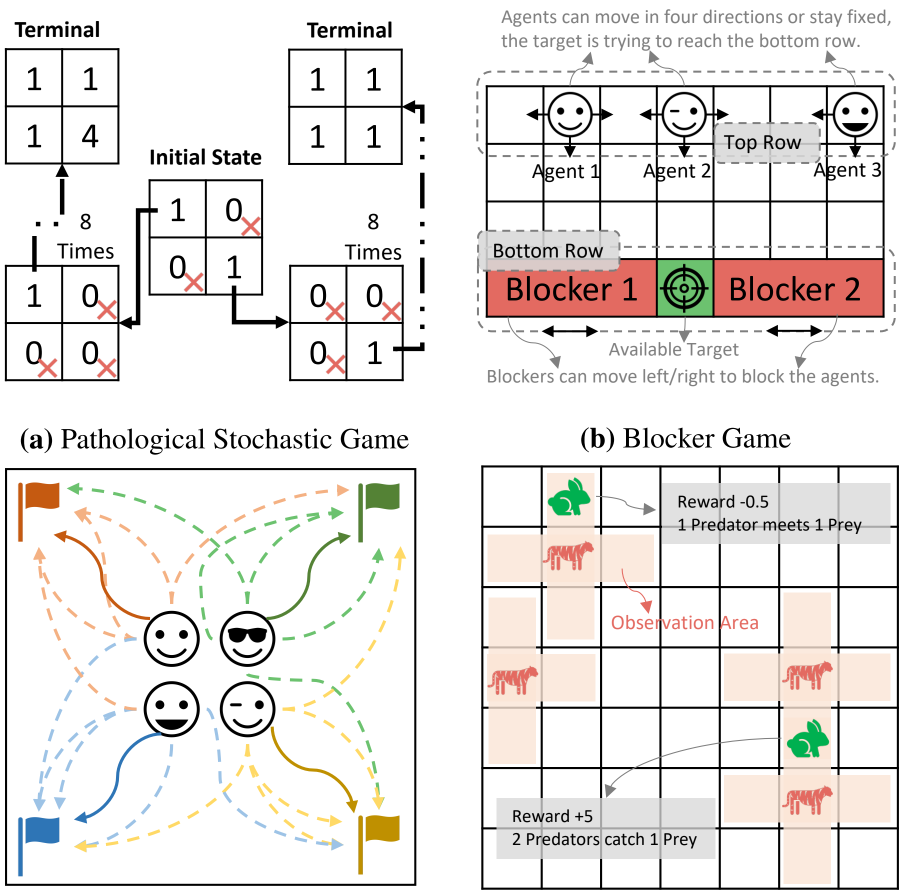


Figure 5.3: Neural Architecture of Deep Q-DPP. The middle part of the diagram shows the overall architecture of Q-DPP, which consists of each agent's individual Q-networks and a centralized mixing network. Details of the mixing network are presented in the left. We compute the quality term, d_i, by applying the exponential operator on the individual Q-value, and compute the diversity feature term, $\boldsymbol{b}_i$, by index the corresponding vector in $\boldsymbol{\mathcal{B}}$ through the global state s and each action a_i.

neural architecture designs in solving multi-agent cooperative tasks, we omit the study of Deep Q-DPP in the main body of this chapter. Here we demonstrate a proof of concept for Deep Q-DPP and its effectiveness on StarCraft II micro-management tasks (Samvelyan et al., 2019) as an initiative. However, we do believe a full treatment needs substantial future work.

5.3.1 Neural Architectures for Deep Q-DPP.

A critical advantage of Deep Q-DPP is that it can deal with continuous states/observations. When the input state s is continuous, we first index the raw diversity feature $\boldsymbol{b}_i'$ based on the embedding of discrete action a_i. To integrate the information of the continuous state, we use two multi-layer feed-forward neural networks f_d and f_n, which encodes the direction and norm of the diversity feature separately. f_d outputs a feature vector with same shape as $\boldsymbol{b}_i'$ indicating the **direction**, and f_n outputs a real value for computing the **norm**. In practice, we find modeling the direction and norm of the diversity features separately by two neural networks helps stabilize training, and the diversity feature vector is computed as $\boldsymbol{b_i} = f_d(\boldsymbol{b}_i', s) \times \sigma(f_n(\boldsymbol{b}_i', s))$. Finally, the centralized Q-value can then be computed from d_i and $\boldsymbol{b_i}$ following Equation 5.4.



(a) Pathological Stochastic Game **(b)** Blocker Game

(c) Coordinated Navigation **(d)** Predator-Prey World

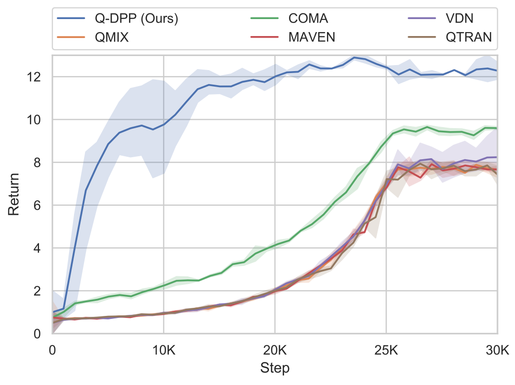
Figure 5.4: Multi-agent cooperative tasks. The size of the ground set for each task is a) 176, b) 420, c) 720, d) 3920.

5.4 Experiments

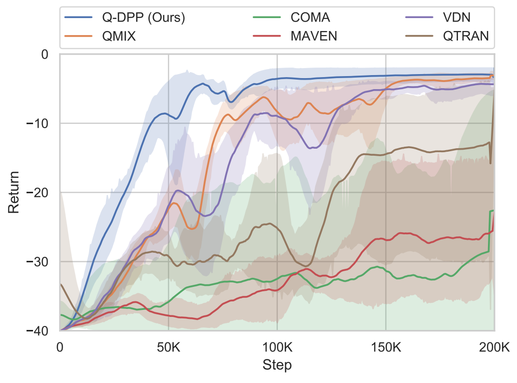
We compare Q-DPP with state-of-the-art CTDE solvers for multi-agent cooperative tasks, including COMA (Foerster et al., 2018b), VDN (Sunehag et al., 2018), QMIX (Rashid et al., 2018), QTRAN (Son et al., 2019), and MAVEN (Mahajan et al., 2019). All baselines are imported from PyMARL (Samvelyan et al., 2019). We consider four cooperative tasks based on grid worlds in Figure 5.4, all of which require different behaviors to achieve cooperative goals. Therefore, the above models need a non-trivial value function decomposition to achieve the largest reward.

5.4.1 Discrete State and Action Games

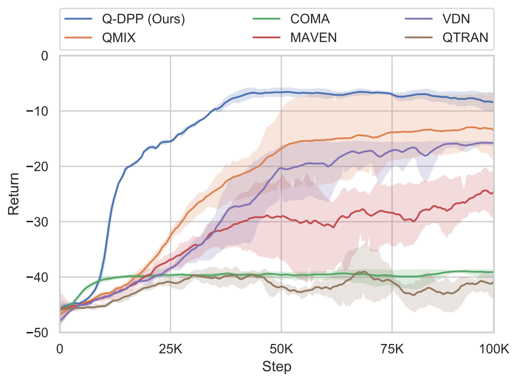
Pathological Stochastic Game. The optimal policy of this game is to let both agents keep acting top left until the 10-th step to change to bottom right, which results in the



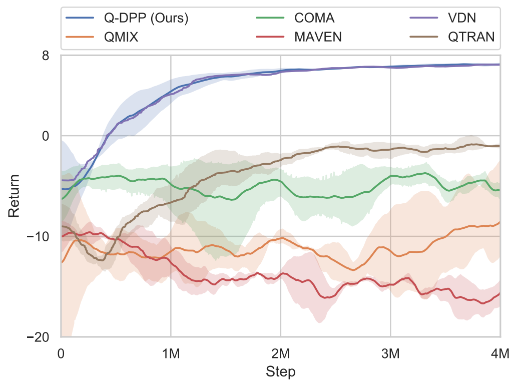
(a) Multi-Step Matrix Game



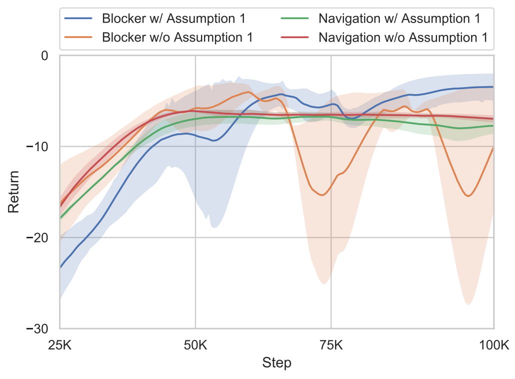
(b) Blocker Game



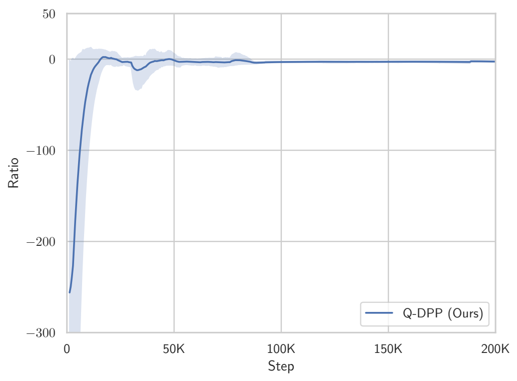
(c) Coordinated Navigation



(d) Predator-Prey World



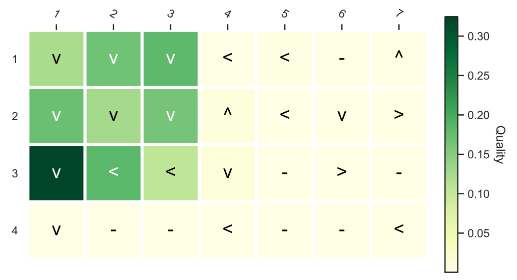
(e) Ablation study on Assumption 5.1



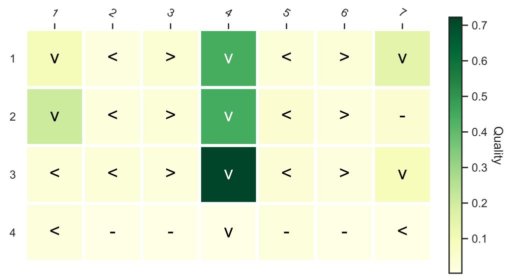
(f) Diversity / Quality Ratio

Figure 5.5: **(a)**-**(d)**:Performance over time on different tasks. **(e)**: Ablation study on Assumption 5.1 on Blocker game. **(f)**: The ratio of diversity to quality, i.e., $\log\det(\boldsymbol{\mathcal{B}}_Y^\top \boldsymbol{\mathcal{B}}_Y)/\sum_{i=1}^N Q_{\mathcal{I}(o_i,a_i)}(o_i,a_i)$, during training on Blocker game.

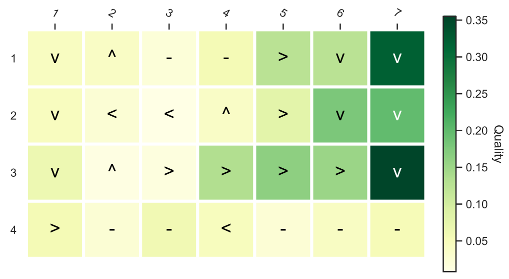
optimal reward of 13. The design of such stochastic game intends to be pathological. First, it is non-monotonic (thus QMIX surely fails), second, it demonstrates *relative overgeneralization* (Wei et al., 2018) because both agents playing the 1st action on average offer a higher reward 10 when matched with arbitrary actions from the other



(a) Agent 1.



(b) Agent 2.



(c) Agent 3.

Figure 5.6: **(a)**-**(c)**: Each of the agent's decentralized policy, i.e., $\arg\max_a Q_i(o_i, a)$, during execution on Blocker game.

agent. We allow agent to observe the current step number and the joint action in the last time-step. Zero reward leads to immediate termination. Figure 5.5a shows Q-DPP can converge to the global optimal in only $20K$ steps while other baselines struggle.

Blocker Game & Coordinated Navigation. *Blocker game* (Heess et al., 2012) requires agents to reach the bottom row by coordinating with its teammates to deceive the blockers that can move left/right to block them. The *coordinated navigation* requires four agents to reach four different landmarks. For both tasks, it costs all agents -1 reward per time-step before they all reach the destination. Depending on the starting points, the largest reward of the game are -3 and -6 respectively. Both tasks are challenging in the sense that coordination is rather challenging for agents that only have decentralized policies and local observations. Figure 5.5b & 5.5c suggest Q-DPP still achieves the best performance.

Predator-Prey World. In this task, four predators attempt to capture two randomly-moving preys. Each predator can move in four directions but they only have local views. The predators get a team reward of 1 if two or more predators are capturing the same prey at the same time, and they are penalized for -0.5 if only one of them captures a prey. The game terminates when all preys are caught. Figure 5.5d shows Q-DPP's superior performance than all other baselines.

Apart from the best performance in terms of rewards, here we offer more insights of why and how Q-DPP works well.

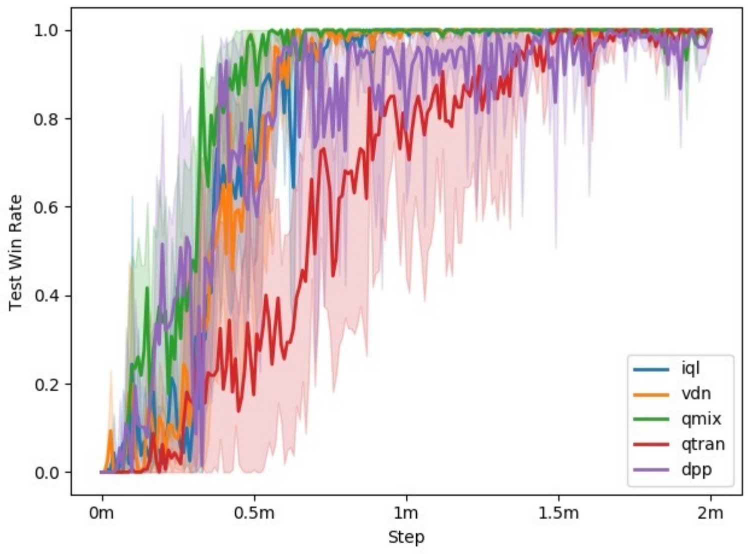
The Importance of Assumption 5.1. Assumption 5.1 is the premise for the correctness of Q-DPP sampler to hold. To investigate its impact in practice, we conduct the ablation study on Blocker and Navigation games. We implement such assumption via an auxiliary loss function of $\max(0, \sigma_j^2 - \hat{\sigma}_{i,j}^2/\delta)$ that penalizes the violation of the assumption, we set $\delta = 0.5$. Figure 5.5e presents the performance comparisons of the Q-DPPs with and without such additional loss function. We can tell that maintaining such a condition, though not helping improve the performance, stablizes the training process by significantly reducing the variance of the rewards. We believe this is because violating Assumption 5.4 leads to over-estimating the probability of certain observation-action pairs in the partition where the violation happens, such over-estimation can make the agent stick to a poor local observation-action pair for some time.

The Satisfaction of Equation 2.9. We show empirical evidence on Blocker game that the natural factorization that Q-DPP offers indeed satisfy Equation 2.9. Intuitively, Q-DPPs encourage agents to acquire diverse behavorial models during training so that the optimal action of one agent does not depend on the actions of the other agents during the decentralized execution stage, as a result, Equation 2.9 can be satisfied. Figures 5.6 (a-c) justify such intuition by showing Q-DPP learns mutually orthogonal behavioral models. Given the distinction among agents' individual policies, one can tell that the joint optimum is reached through individual optima.

Quality versus Diversity. We investigate the change of the relative importance of quality versus diversity during training. On Blocker game, we show the ratio of $\log\det\left(\boldsymbol{\mathcal{B}}_Y^\top \boldsymbol{\mathcal{B}}_Y\right) / \sum_{i=1}^N Q_{\mathcal{I}(o_i,a_i)}\left(o_i, a_i\right)$, which reflects how the learning algorithm balances maximizing reward against encouraging diverse behaviors. In Figure 5.5f, we can see that the ratio gradually converges to 0. The diversity term plays a less important role with the development of training; this is also expected since explorations tend to be rewarded more at the early stage of a task.

5.4.2 StarCraft II Micro-Management

(a) Scenario Screenshot



(b) 2m_vs_1z

Figure 5.7: StarCraft II micro-management on the scenario of *2 Marines vs. 1 Zealot* and its performance.

We study one of the simplest continuous state-action micro-management games in StarCraft II in SMAC (Samvelyan et al., 2019), i.e., **2m_vs_1z**, the screenshots of scenarios are given in Figure 5.7a. In the 2m_vs_1z map, we control a team of

2 Marines to fight with 1 enemy Zergling. In this task, it requires the Marine units to take advantage of their larger firing range to defeat over Zergling which can only attack local enemies. The agents can observe a **continuous** feature vector including the information of health, positions and weapon cooldown of other agents. In terms of reward design, we keep the default setting. All agents receive a large final reward for winning a battle, at the meantime, they also receive immediate rewards that are proportional to the difference of total damages between the two teams in every time-step. We compare Q-DPP with aforementioned baseline models, i.e., COMA, VDN, QMIX, MAVEN, and QTRAN, and plot the results in Figure 5.7b. The results show that Q-DPP can perform as good as the state-of-the-art model, QMIX, even when the state feature is continuous. However, the performance is not stable and presents high variance. We believe full treatments need substantial future work to reduce the sampling bias in neural DPP kernel.

5.5 Summary

We proposed Q-DPP, a new type of value-function approximator for cooperative multi-agent reinforcement learning. Q-DPP, as a probabilistic way of modeling sets, considers not only the quality of agents' actions towards reward maximization, but the diversity of agents' behaviors. We have demonstrated that Q-DPP addresses the joint exploration limitation of current major solutions including VDN, QMIX, and QTRAN by learning the value function decomposition without structural constraints.

Part III

Game-Theoretic Analysis of Policy-Space Influence

Chapter 6

Multi-Agent Trust Region Learning

Due to the complexity of multi-agent problems (Chatterjee et al., 2004), investigating if agents can learn to behave effectively during interactions with environments and other agents is essential (Fudenberg et al., 1998). This can be achieved naively through the *independent learner* (IL) (Tan, 1993), which ignores the other agents and optimizes the policy assuming a stable environment (Buşoniu et al., 2010; Hernandez-Leal et al., 2017). Due to their theoretical guarantee and good empirical performance in real-world applications, *trust region* methods (e.g., PPO (Schulman et al., 2015, 2017)) based ILs are popular (Berner et al., 2019; Vinyals et al., 2019). In single-agent learning, trust region methods can produce a monotonic payoff improvement guarantee (Kakade and Langford, 2002) via line search (Schulman et al., 2015). However, in multi-agent scenarios, an agent's improvement is affected by the other agent's behaviors (i.e., the multi-agent environment is *non-stationary* (Hernandez-Leal et al., 2017)), which means that trust region based ILs act less well in single-agent tasks. As shown in Figure 6.1, trust region learners can measure the policy improvements of the agents' current policies, but the improvements of the updated opponents' policies are unknown. Therefore, using trust region ILs to play optimally against stationary agents is insufficient as a multi-agent learning method. Moreover, the convergence to a *fixed point*, such as a *Nash equilibrium* (Bowling and Veloso, 2004; Mazumdar et al., 2020), is a common and widely accepted solution concept for multi-agent learning. Thus, although independent learners can best respond to other agents' current policies, they lose their convergence guarantee (Laurent et al.,

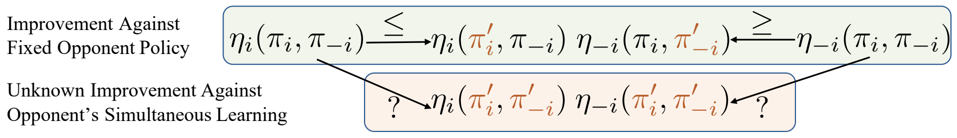
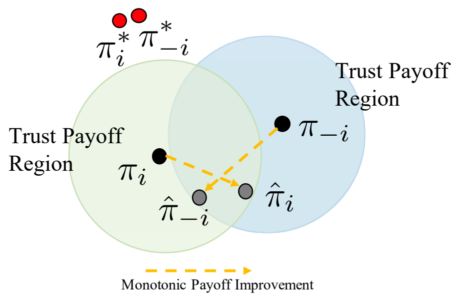

Improvement Against Fixed Opponent Policy
Unknown Improvement Against Opponent's Simultaneous Learning
$\eta_i(\pi_i, \pi_{-i}) \xrightarrow{\leq} \eta_i(\pi_i', \pi_{-i})$
$\eta_{-i}(\pi_i, \pi_{-i}') \xleftarrow{\geq} \eta_{-i}(\pi_i, \pi_{-i})$
$? \rightarrow \eta_i(\pi_i', \pi_{-i}')$
$\eta_{-i}(\pi_i', \pi_{-i}') \leftarrow ?$

Figure 6.1: The relationship of discounted returns η_i for an agent i given the different joint policy pairs, where π_i is the current policy, π_i' is the simultaneously updated policy.

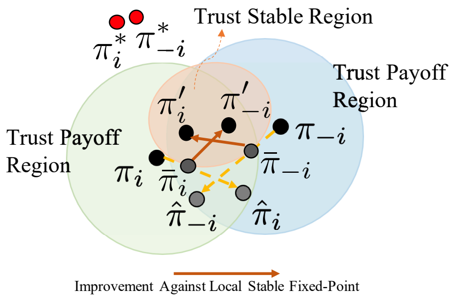
2011).

One solution to address the convergence problem for independent learners is Empirical Game-Theoretic Analysis (EGTA) (Wellman, 2006), which approximates the best response to the policies generated by the independent learners (Lanctot et al., 2017; Muller et al., 2020). Although EGTA based methods (Balduzzi et al., 2019; Lanctot et al., 2017; Omidshafiei et al., 2019) establish convergence guarantees in several games classes, the computational cost is also large when empirically approximating and solving the increasing meta-game (Yang et al., 2019). Other multi-agent learning approaches collect or approximate additional information such as communication (Foerster et al., 2016) and centralized joint critics (Foerster et al., 2018b; Lowe et al., 2017; Rashid et al., 2018; Sunehag et al., 2018). Nevertheless, these methods usually require centralized parameters or centralized communication assumptions. Thus, there is considerable interest in multi-agent learning to find an algorithm that, while having minimal requirements and computational cost as independent learners, also improves convergence performance at the same time.

This chapter presents the *Multi-Agent Trust Region Learning* (MATRL) algorithm that augments the trust-region ILs with a restricted meta-game to improve the stability and efficiency of learning. In MATRL, a trust region trial step on agents' payoff improvement is implemented by independent learners, which gives a predicted policy based on the current policy. Then, an empirical policy-space meta-game is constructed comparing the expected advantage of predicted policies with the current policies. By solving the meta-game, MATRL finds a restricted step by aggregating the current and predicted policies using meta-game Nash Equilibrium Finally, MATRL takes the best responses based on the aggregated policies from last step for each agent to explore and avoid the potential saddle point. MATRL



(a) Independent trust region learning.



(b) Multi-agent trust region learning.

Figure 6.2: Comparisons between independent trust region learner and multi-agent trust region learner. π_i, π_{-i} are the current policies for two agents. $\hat{\pi}_i, \hat{\pi}_{-i}$ predicted policies within TPR, (π_i^*, π_{-i}^*) forms Nash equilibrium, π'_i and π'_{-i} are the best responses to the fixed point $(\bar{\pi}_i, \bar{\pi}_{-i})$ constrained by TSR.

is, therefore, able to provide a more stable point compared with the naive independent learners. Based on trust region independent learners, MATRL does not need extra parameters, simulations, or modifications to the independent learner itself. We provide insights into the empirical meta-game in Section 6.2.2, showing that an approximated Nash equilibrium of the meta-game is a weak stable fixed point of the underlying game. We also prove that MATRL has a tighter lower bound and better convergence performance than independent learners in various games. The experiments demonstrate that MATRL significantly outperforms deep independent learners (Schulman et al., 2017) with the same hyper-parameters, centralized VDN (Sunehag et al., 2018), QMIX (Rashid et al., 2018) methods in discrete action grid-worlds and centralized MADDPG (Lowe et al., 2017) in a continuous action multi-agent MuJoCo task (de Witt et al., 2020).

6.1 Related Work

The study of gradient-based methods in multi-agent learning is quite extensive (Buşoniu et al., 2010; Mazumdar et al., 2020). Some works on learning in games have mostly focused on adjusting the step size, which attempts to use a multiple-timescales learning scheme (Bowling and Veloso, 2002; Leslie et al., 2003; Leslie and Collins, 2005) to achieve convergence. Balduzzi et al. (2018); Letcher et al. (2019); Mazumdar et al. (2019) tried to utilize the second-order methods to shape the step size. However, the computational cost for second-order methods is very limiting in many cases. Alternatively, MATRL approximates the second-order fixed-point information via a small meta-game with less cost comparing to real Hessian computation. An alternative augments the gradient-based algorithms with the best response to predicted polices (Antipin, 2003; Foerster et al., 2018a; Lin et al., 2020; Lockhart et al., 2019; Tang et al., 2018; Zhang and Lesser, 2010), which target the challenge of instability caused by agents' change policies. Instead of taking the best response to the approximated opponent's policy, MATRL exploits the ideas from both streams and and introduces the improvement over the weak stable fixed point.

The research also focuses on the EGTA (Jordan and Wellman, 2009; Tuyls et al., 2020, 2018), which creates a policy-space meta-game for modeling the multi-agent interactions. Using various evaluation metrics, it then updates and extends the policies based on the analysis of the meta policies (Balduzzi et al., 2019; Lanctot et al., 2017; Muller et al., 2020; Omidshafiei et al., 2019; Yang et al., 2019). Although these methods are broad with respect to multi-agent tasks, they require extensive computing resources to estimate the empirical meta-game and solve it with its increasing size (Omidshafiei et al., 2019; Yang et al., 2019). In our method, we adopt the idea of a policy-space meta-game to approximate the fixed point. Unlike previous works, we only maintain the current policies and predicted policies to construct the meta-game, which is computationally achievable in most cases. The payoff entry in MATRL's meta-game is the expected advantage, which has a lower estimation variance compared to the commonly used empirically-estimated return in EGTAs. Regardless, we can reuse the trajectories in the TPR step to estimate the payoffs without incurring additional sampling costs.

Recently, due to the use of neural networks as a function approximation for policies and values, there have emerged many works on deep reinforcement learning (DRL) (Lillicrap et al., 2015; Mnih et al., 2013). Trust region policy optimization (Kakade and Langford, 2002; Schulman et al., 2015, 2017) is one of the most successful DRL methods in the single-agent setting, which puts constraints on the step size of policy updates, preserving any improvements monotonically. Based the monotonic improvement in single-agent trust region policy optimization

(TRPO) (Schulman et al., 2015), MATRL extends the improvement guarantee to the multi-agent level, towards a weak stable fixed point Some works directly apply fully decentralized single-agent DRL methods (Tan, 1993), which can be unstable during when learning due to the non-stationary issue. Whereas Foerster et al. (2016); Peng et al. (2017); Sukhbaatar et al. (2016) added an extra communication channel during the training and execution in a centralized way to avoid this non-stationarity issue. Foerster et al. (2018b); Lowe et al. (2017); Rashid et al. (2018); Sunehag et al. (2018) further exploit the setting of centralized learning through a decentralized execution. These methods provide solutions for training agents in complex multi-agent environments, and the experimental results show the effectiveness compared with independent learners. MATRL also provides fully decentralized execution and only requires a centralized mechanism to adjust the step size rather than the centralized critic or communication.

6.2 Multi-Agent Trust Region Policy Optimization

A trust region algorithm aims to determine how to compute the trust region trial step and whether a trial step should be accepted. In multi-agent learning, the trust region trial step towards agents' payoff improvement can be implemented with independent learners, and the independent payoff improvement is called the ***Trust Payoff Region***(**TPR**). Then, the issue revolves around finding a restricted step leading to a stable point in the joint policy space, denoted as ***Trust Stable Region***(**TSR**). In other words, multi-agent trust region learning (MATRL) decomposes the trust region learning into two parts: firstly, find a trust payoff region between the current policy π_i and the predicted policy $\hat{\pi}_i$; then, with the help of the predicted policy, a precise method can, to some extent, approximate a weak stable fixed point. Instead of line searching in a single-agent payoff improvement, MATRL searches for the joint policy space to achieve a weak stable fixed point

Essentially, MATRL is a simple extension of the single-agent TRPO where independent learners with an approximated restricted stable point. For conciseness, this section presents two agents (played by agent $i \in \{1, 2\}$) interacting with each other simultaneously. Our method can be easily extended to a larger number of agents because our algorithm is entirely based on independent learners. Firstly, behaving like the independent learners, an agent i collects a set of trajectories using current policies π_i. A predicted policy $\hat{\pi}_i$ can be estimated using the single-agent trust region methods, which has a trust payoff improvement against the other agents' current policy π_{-i}. However, the trust payoff improvements would not benefit convergence requirements for the multi-agent system. Therefore, by reusing the trajectories we approximate a two-agent two-action meta-game in policy-space. In the meta-game, the higher-level policies become 'actions', and the expected advantages of the joint policy pairs are the payoffs. Here, each agent i has two actions: current policy π_i and predicted policy $\hat{\pi}_i$, making the complex multi-agent interactions applicable to game-theoretic analysis concerning the restricted under-layer game between π_i and $\hat{\pi}_i$. By solving the meta-game, we can obtain a weak stable fixed point as TSR within the TPR. However, if the fixed point is a saddle point we then conduct the best response to the weak stable fixed point to get the next iterate's policies. This can encourage exploration and avoid stagnation at an unexpected saddle point. Figure 6.3 shows the overview of MATRL. We also give the pseudocode of MATRL in Algo. 6.5 at the end of this section, which is compatible with any policy-based independent learner.

In summary, in MTARL, the independent trust region learners will be constrained by a weak stable fixed point. Providing additional rollouts or simulations are not required, we can build a policy-space meta-game and easily approximate a weak stable fixed point due to the small meta-game and conservative policy change. Although MATRL's training is centralized, its execution is fully decentralized. It also does not require any extra centralized parameters or higher-order gradient computation. We give the precise steps of MATRL in the following sections.

6.2.1 Independent Trust Payoff Region

Single-agent reinforcement learning algorithms can be straightforwardly applied to multi-agent learning, where we assume that all agents behave independently (Tan, 1993). In this chapter, we have chosen the policy-based reinforcement learning

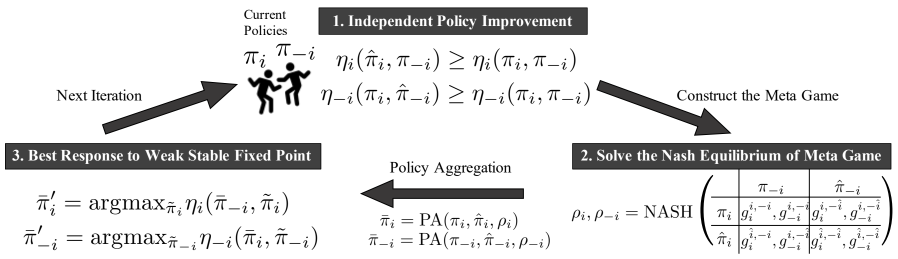

Current Policies
π_i π_{-i}
1. Independent Policy Improvement
$\eta_i(\hat{\pi}_i, \pi_{-i}) \geq \eta_i(\pi_i, \pi_{-i})$
$\eta_{-i}(\pi_i, \hat{\pi}_{-i}) \geq \eta_{-i}(\pi_i, \pi_{-i})$
Next Iteration
Construct the Meta Game
3. Best Response to Weak Stable Fixed Point
$\bar{\pi}'_i = \text{argmax}_{\tilde{\pi}_i} \eta_i(\bar{\pi}_{-i}, \tilde{\pi}_i)$
$\bar{\pi}'_{-i} = \text{argmax}_{\tilde{\pi}_{-i}} \eta_{-i}(\bar{\pi}_i, \tilde{\pi}_{-i})$
Policy Aggregation
$\bar{\pi}_i = \text{PA}(\pi_i, \hat{\pi}_i, \rho_i)$
$\bar{\pi}_{-i} = \text{PA}(\pi_{-i}, \hat{\pi}_{-i}, \rho_{-i})$
2. Solve the Nash Equilibrium of Meta Game
$\rho_i, \rho_{-i} = \text{NASH} \left(\begin{array}{c|c|c} & \pi_{-i} & \hat{\pi}_{-i} \\ \hline \pi_i & g_i^{i,-i}, g_{-i}^{i,-i} & g_i^{i,-\hat{i}}, g_{-i}^{i,-\hat{i}} \\ \hline \hat{\pi}_i & g_i^{\hat{i},-i}, g_{-i}^{\hat{i},-i} & g_i^{\hat{i},-\hat{i}}, g_{-i}^{\hat{i},-\hat{i}} \end{array} \right)$

Figure 6.3: Overview of the multi-agent trust region learning phases in two-agent games. It can be easily extended to the n-agent case by solving the n-agent two-action matrix form meta-game.

method as independent learners. For agent $i \in \{1, 2\}$, it searches for the optimal policy within the policy-space Π_i. In multi-agent environments, the environment becomes a Markov decision process for one agent when the other agents play according to a fixed policy, and the goal is to make a monotonic improvement. Thus, at iterate k, the policy is updated by maximizing the utility function η_i over a local neighborhood of the current joint policy π_i^k, π_{-i}^k: $\hat{\pi}_i = \arg\max_{\pi_i \in \Pi_i} \eta_i(\pi_i, \pi_{-i}^k)$ based on the trajectories τ_i^k, τ_{-i}^k sampled by π_i^k, π_{-i}^k. We can adopt trust region policy optimization (e.g., TRPO (Schulman et al., 2015) , PPO (Schulman et al., 2017)), which constrains step size in the policy update:

$$\hat{\pi}_i^k = \arg \max_{\pi_i \in \Pi_{\theta_i}} \eta_i(\pi_i, \pi_{-i}^k) \quad \text{s.t. } D\left(\pi_i, \pi_{-i}^k\right) \leq \delta_i, \tag{6.1}$$

where D is a distance measurement and δ is a constraint. Independent trust region learners produce the monotonically improved policy $\hat{\pi}_i$ which guarantees $\eta_i\left(\hat{\pi}_i, \pi_{-i}^k\right) \geq \eta_i\left(\pi_i^k, \pi_{-i}^k\right)$ and gives a trust payoff bound by $\hat{\pi}_i$. Due to the simultaneous trust payoff region improvement, the lower bound of the trust region for single-agent improvement in Schulman et al. (2015) no longer holds. Following the similar proof procedures, we can obtain the precise lower bound for a multi-agent simultaneous trust payoff region in Theorem 6.1:

Theorem 6.1 (Independent Trust Payoff Region)**.** *Let* $(\pi_i, \hat{\pi}_i)$ *and* $(\pi_{-i}, \hat{\pi}_{-i})$ *be* α*-coupled policy pairs (Schulman et al., 2015), which are coupled by* α_i *and* α_{-i}

respectively. Denote the expected advantage gain when $\pi_i, \pi_{-i} \to \hat{\pi}_i, \hat{\pi}_{-i}$ as:

$$G_i^{\pi_i,\pi_{-i}}(\hat{\pi}_i, \hat{\pi}_{-i}) := \sum_s p^{\pi_i,\pi_{-i}}(s) \sum_{a_i} \hat{\pi}_i(a_i|s) \sum_{a_{-i}} \hat{\pi}_{-i}(a_{-i}|s) A_i^{\pi_i,\pi_{-i}}(s, a_i, a_{-i}). \tag{6.2}$$

Then, the following lower bound can be derived for independent trust region optimization:

$$\eta_i(\hat{\pi}_i, \hat{\pi}_{-i}) - \eta_i(\pi_i, \pi_{-i}) \geq G_i^{\pi_i,\pi_{-i}}(\hat{\pi}_i, \hat{\pi}_{-i}) - \frac{4\gamma\epsilon_i}{(1-\gamma)^2}(\alpha_i + \alpha_{-i} - \alpha_i\alpha_{-i})^2, \tag{6.3}$$

where $\epsilon_i = \max_{s,a_{-i},a_{-i}} \left|A_i^{\pi_i,\pi_{-i}}(s, a_i, a_{-i})\right|$.

Proof. We use the total variation divergence, which is defined by $D_{TV}(p\|q) = \frac{1}{2}\sum_j |p_j - q_j|$ for discrete probability distributions p, q (Schulman et al., 2015). $D_{\text{TV}}^{\max}(\pi, \tilde{\pi})$ is defined as:

$$D_{\text{TV}}^{\max}(\pi, \tilde{\pi}) = \max_s D_{TV}(\pi(\cdot|s)\|\tilde{\pi}(\cdot|s)). \tag{6.4}$$

Based on this, we can define α-coupled policy as:

Definition 6.1 (α-Coupled Policy (Schulman et al., 2015))**.** *(π, π') is an α-coupled policy pair if it defines a joint distribution $(a, a')|s$, such that $P(a \neq a'|s) \leq \alpha$ for all s. π and π' will denote the marginal distributions of a and a', respectively.*

When the joint policy pair π_i, π_{-i} changes to π'_i, π'_{-i} and coupled with α_i and α_{-i} correspondingly:

$$\eta_i(\pi'_i, \pi'_{-i}) - \eta_i(\pi_i, \pi_{-i}) \geq A_i^{\pi_i,\pi_{-i}}(\pi'_i, \pi'_{-i}) - \frac{4\gamma\epsilon}{(1-\gamma)^2}(\alpha_i + \alpha_{-i} - \alpha_i\alpha_{-i})^2, \tag{6.5}$$

where

$$\epsilon = \max_{s,a_i,a_{-i}} \left|A_i^{\pi_i,\pi_{-i}}(s, a_i, a_{-i})\right|.$$

The proofs are as following:

Lemma 6.1. *Given that (π_i, π'_i) and (π_{-i}, π'_{-i}) are both α-coupled policies bounded* *by α_i and α_{-i} respectively, for all s,*

$$|A_i^{\pi_i,\pi_{-i}}(s)| \leq 2(\alpha_i + \alpha_{-i} - \alpha_i\alpha_{-i}) \max_{s,a_{-i},a_{-i}} \left|A_i^{\pi_i,\pi_{-i}}(s, a_i, a_{-i})\right| \tag{6.6}$$

Proof.

$$\begin{aligned} A_i^{\pi_i,\pi_{-i}}(s) &= \mathbb{E}_{a'_i,a'_{-i}\sim\pi'_i,\pi'_{-i}}\left[A_i^{\pi_i,\pi_{-i}}(s, a'_i, a'_{-i})\right] \\ &= \mathbb{E}_{(a_i,a'_i)\sim(\pi_i,\pi'_i),(a_{-i},a'_{-i})\sim(\pi_{-i},\pi'_{-i})}\left[A_i^{\pi_i,\pi_{-i}}(s, a'_i, a'_{-i}) - A_i^{\pi_i,\pi_{-i}}(s, a_i, a_{-i})\right] \\ &= P(a_i \neq a'_i \vee a_{-i} \neq a'_{-i}|s)\mathbb{E}_{(a_i,a'_i)\sim(\pi_i,\pi'_i),(a_{-i},a'_{-i})\sim(\pi_{-i},\pi'_{-i})}\Big[\\ &\qquad A_i^{\pi_i,\pi_{-i}}(s, a'_i, a'_{-i}) - A_i^{\pi_i,\pi_{-i}}(s, a_i, a_{-i})\Big] \\ &\leq (\alpha_i + \alpha_{-i} - \alpha_i\alpha_{-i}) \cdot 2 \max_{s,a_{-i},a_{-i}} \left|A_i^{\pi_i,\pi_{-i}}(s, a_i, a_{-i})\right|, \end{aligned} \tag{6.7}$$

where $P(a_i \neq a'_i \vee a_{-i} \neq a'_{-i}|s) = 1 - (1 - \alpha_i)(1 - \alpha_{-i}) = \alpha_i + \alpha_{-i} - \alpha_i\alpha_{-i}$.

■

Lemma 6.2. *Let (π_i, π'_i) and (π_{-i}, π'_{-i}) are α-coupled policy pairs. Then,*

$$\begin{aligned} &\left|\mathbb{E}_{s_t\sim\pi'_i,\pi'_{-i}}\left[A_i^{\pi_i,\pi_{-i}}(s)\right] - \mathbb{E}_{s_t\sim\pi_i,\pi_{-i}}\left[A_i^{\pi_i,\pi_{-i}}(s)\right]\right| \\ &\leq 4(\alpha_i + \alpha_{-i} - \alpha_i\alpha_{-i})(1 - (1 - \alpha_i)^t(1 - \alpha_{-i})^t) \max_{s,a_{-i},a_{-i}} \left|A_i^{\pi_i,\pi_{-i}}(s, a_i, a_{-i})\right|. \end{aligned} \tag{6.8}$$

Proof. The preceding Lemma bounds the difference in expected advantage at each time step t. When $t' = 0$ indicates that π_i, π_{-i} and π'_i, π'_{-i} both agreed on all time steps less than t. By the definition of α_i, α_{-i}, $P(\pi_i, \pi_{-i} := \pi'_i, \pi'_{-i}|t = i) \geq (1 - \alpha_i)(1 - \alpha_{-i})$, so $P(t' = 0) \geq (1 - \alpha_i)^t(1 - \alpha_{-i})^t$ and $P(t' > 0) \leq 1 - (1 - \alpha_i)^t(1 - \alpha_{-i})^t$. We can sum over time to bind the difference between $\eta_i(\pi'_i, \pi'_{-i})$ and

$\eta_i(\pi_i, \pi_{-i})$.

$$\begin{aligned}
&\left|\eta_i(\pi'_i, \pi'_{-i}) - L_i^{\pi_i,\pi_{-i}}(\pi'_i, \pi'_{-i})\right| \\
&= \sum_{t=0}^{\infty} \gamma^t \left|\mathbb{E}_{s_t \sim \pi'_i,\pi'_{-i}}\left[A_i^{\pi_i,\pi_{-i}}(s)\right] - \mathbb{E}_{s_t \sim \pi_i,\pi_{-i}}\left[A_i^{\pi_i,\pi_{-i}}(s)\right]\right| \\
&\leq \sum_{t=0}^{\infty} \gamma^t \cdot 4\epsilon(\alpha_i + \alpha_{-i} - \alpha_i\alpha_{-i})(1 - (1-\alpha_i)^t(1-\alpha_{-i})^t) \\
&= 4\epsilon(\alpha_i + \alpha_{-i} - \alpha_i\alpha_{-i})\Big(\frac{1}{1-\gamma} - \frac{1}{1-\gamma(1-\alpha_i)(1-\alpha_{-i})}\Big) \\
&= \frac{4\epsilon(\alpha_i + \alpha_{-i} - \alpha_i\alpha_{-i})^2}{(1-\gamma)(1-\gamma(1-\alpha_i)(1-\alpha_{-i}))} \\
&\leq \frac{4\epsilon(\alpha_i + \alpha_{-i} - \alpha_i\alpha_{-i})^2}{(1-\gamma)^2},
\end{aligned} \tag{6.9}$$

where $\epsilon = \max_{s,a_i,a_{-i}} \left|A_i^{\pi_i,\pi_{-i}}(s, a_i, a_{-i})\right|$. ■

Note that

$$\begin{aligned}
&L_i^{\pi_i,\pi_{-i}}(\pi'_i, \pi'_{-i}) = \eta_i(\pi_i, \pi_{-i}) \\
&+ \sum_s \rho^{\pi_i,\pi_{-i}}(s) \sum_{a_i} \pi'_i(a_i|s) \sum_{a_{-i}} \pi'_{-i}(a_{-i}|s) A_i^{\pi_i,\pi_{-i}}(s, a_i, a_{-i}).
\end{aligned} \tag{6.10}$$

Then, we can have

$$\eta_i(\pi'_i, \pi'_{-i}) - \eta_i(\pi_i, \pi_{-i}) \geq A_i^{\pi_i,\pi_{-i}}(\pi'_i, \pi'_{-i}) - \frac{4\gamma\epsilon}{(1-\gamma)^2}(\alpha_i + \alpha_{-i} - \alpha_i\alpha_{-i})^2. \tag{6.11}$$

■

Based on the independent trust region learning, although the predicted policy $\hat{\pi}_i$ will guide us in determining the size of the TPR, the stability of $(\hat{\pi}_i, \hat{\pi}_{-i})$ is still unknown. As shown in Theorem 6.1, in a two-agent case, an agent's lower bound is roughly $O(4\alpha^2)$, which is four times larger than the single-agent lower bound trust region of $O(\alpha^2)$ (Kakade and Langford, 2002). Furthermore, $\epsilon_i = \max_{s,a_{-i},a_{-i}} \left|A_i^{\pi_i,\pi_{-i}}(s, a_i, a_{-i})\right|$ depends on the other agents that would be too large in worst-case scenarios like zero-sum games. Therefore, the most critical issue underlying the multi-agent trust region learning is to find a TSR after the TPR. The next section will illustrate how to search for a weak stable fixed point within the TPR, based on the policy-space meta-game.

6.2.2 Approximating Weak Stable Fixed Point via Restricted Policy-Space Meta-Game

In multi-agent trust region learning, TSR is one of the essential parts. Since each iteration of MATRL requires solving (exactly or inexactly) the TPR and TSR sub-problems, finding the efficient solver for stable trust region sub-problems is very important. Given that we already have the TPR, which produces a predicted policy $\hat{\pi}_i^k$, it is natural to conduct an empirical game game-theoretic analysis (Tuyls et al., 2018) to search for a weak stable fixed point in the predicted joint policy $\hat{\pi}_i^k, \hat{\pi}_{-i}^k$. We define a restricted meta-game that has only two strategies $\hat{\pi}_i^k, \hat{\pi}_{-i}^k$; for each agent i:

$$\mathcal{M}(\pi_i^k, \hat{\pi}_i^k, \pi_{-i}^k, \hat{\pi}_{-i}^k) = \begin{pmatrix} g_i^{i,-i}, g_{-i}^{i,-i} & g_i^{i,-\hat{i}}, g_{-i}^{i,-\hat{i}} \\ g_i^{\hat{i},-i}, g_{-i}^{\hat{i},-i} & g_i^{\hat{i},-\hat{i}}, g_{-i}^{\hat{i},-\hat{i}} \end{pmatrix}, \tag{6.12}$$

where $g_i^{\hat{i},-\hat{i}} = G_i^{\pi_i^k, \pi_{-i}^k}(\hat{\pi}_i^k, \hat{\pi}_{-i}^k)$ for $i \in \{1, 2\}$ is an empirical payoff entry of the meta-game, and note $g_i^{i,-i} = 0$ as it has an expected advantage over itself. Compared with using the $\eta_i(\hat{\pi}_i, \hat{\pi}_{-i}) = \eta_i(\pi_i, \pi_{-i}) + g_i^{\hat{i},-\hat{i}}$ as the meta-game payoff, $g_i^{\hat{i},-\hat{i}}$ has lower variance and is easier to approximate because $\eta_i(\pi_i, \pi_{-i})$ is a constant baseline. However, due to the complex dependency of $\pi_i^k(s)$ on $\hat{\pi}_i^k(s)$, it is still difficult to estimate $g_i^{\hat{i},-\hat{i}}$ directly. Instead, we reuse the trajectories τ_i^k, τ_{-i}^k in the TPR step to approximate the $g_i^{\hat{i},-\hat{i}}$ by ignoring small changes in state visitation density caused by the $\hat{\pi}_i^k \to \bar{\pi}_i^k$ (Schulman et al., 2015).

As we can see in Equation 6.12, in a two-player case, the meta-game is a 2×2 matrix-form game, which is much smaller in size than the under-layer game. To this end, we can use the existing Nash solvers (e.g. CMA-ES (Hansen et al., 2003)) for matrix-form games to compute a Nash equilibrium $\rho_i^k, \rho_{-i}^k = \mathrm{NashSolver}(\mathcal{M})$ for the meta-game $\mathcal{M}$. Then, the trust stable region policies $\bar{\pi}_i^k, \bar{\pi}_{-i}^k$ can be aggregated based on the current policy π_i and monotonic improved policy $\hat{\pi}_i$ in TPR for each agent i. In the TPR step, the change from π_i to $\hat{\pi}_i$ is usually constrained by a small step size, and it is reasonable to assume there is a continuous and monotonic change in the restricted policy space between π_i and $\hat{\pi}_i$. In this case, with the mixture weight ρ_i^k of meta-game Nash, $\bar{\pi}_i^k$ can be derived via linear mixture: $\bar{\pi}_i^k = \rho_i^k \pi_i^k + (1-\rho_i^k)\hat{\pi}_i^k$, which determines the trust stable region. Here we can prove that $(\bar{\pi}_i^k, \bar{\pi}_{-i}^k)$ is a weak stable fixed point for the restricted under-layer game:

Theorem 6.2 (Multi-Agent Trust Stable Region)**.** *At iterate k, consider the restricted under-layer game $\mathcal{G}$ bounded by $\pi_i \in [\pi_i^k, \hat{\pi}_i^k]$ for $i \in \{1, 2\}$, where $\hat{\pi}_i^k$ is learned from π_i^k within TPR. (ρ_i^k, ρ_{-i}^k) is a Nash equilibrium of the meta-game $\mathcal{M}$. Then, the linear mixture joint policy $(\bar{\pi}_i^k, \bar{\pi}_{-i}^k)$ is a weak stable fixed point for the restricted under-layer game $\mathcal{G}$ in a linear continuous policy-space $[\pi_i^k, \hat{\pi}_i^k]$.*

Proof Sketch. Denote the simultaneous gradient of the restricted game $\mathcal{G}$ as $\boldsymbol{\xi} = (\nabla_{\pi_i} G_i, \nabla_{\pi_{-i}} G_{-i})$ and Hessian $H = \nabla\boldsymbol{\xi}$. A point $(\bar{\pi}_i^k, \bar{\pi}_{-i}^k)$, $\bar{\pi}_i^k \in [\pi_i^k, \hat{\pi}_i^k]$, is a fixed point of the restricted under-layer game $\mathcal{G}$ due to $\boldsymbol{\xi}(\bar{\pi}_i^k, \bar{\pi}_{-i}^k) = \mathbf{0}$, and it is weak stable because $H(\bar{\pi}_i^k, \bar{\pi}_{-i}^k) \not\prec 0$, which avoids unstable fixed points. More specifically, it is stable if $H(\bar{\pi}_i, \bar{\pi}_{-i}) \succeq 0$; it is a saddle point if the $H(\bar{\pi}_i^k, \bar{\pi}_{-i}^k)$ has both positive and negative eigenvalues. Here is the proof: At iteration k, denote $\nabla_i G_i = \nabla_{\pi_i} G_i^{\pi_i^k, \pi_{-i}^k}$ and $\nabla_{i,-i} G_i = \nabla_{\pi_i} \nabla_{\pi_{-i}} G_i^{\pi_i^k, \pi_{-i}^k}$ for $i \in i, 2$. Consider the simultaneous gradient $\boldsymbol{\xi}$ of the expected advantage gains and the corresponding Hessian H:

$$\boldsymbol{\xi}(\pi_i, \pi_{-i}) = (\nabla_i G_i, \nabla_{-i} G_{-i}), \tag{6.13}$$

$$H = \nabla\xi = \begin{pmatrix} \nabla_{i,i} G_i & \nabla_{i,-i} G_i \\ \nabla_{-i,i} G_{-i} & \nabla_{-i,-i} G_{-i} \end{pmatrix}. \tag{6.14}$$

For a restricted game $\mathcal{G}$, where policy space is bounded: $\pi_i \in [\pi_i^k, \hat{\pi}_i^k]$. Assume π_i is the linear mixture of $\pi_i^k, \hat{\pi}_i^k$, and $\pi_i = \rho_i \pi_i^k + (1 - \rho_i)\hat{\pi}_i^k$, where $\rho_i \in [0, 1]$. Therefore, we can re-write the $G_i^{\pi_i^k, \pi_{-i}^k}(\pi_i, \pi_{-i})$ in the form of:

$$\begin{aligned} G_i^{\pi_i^k, \pi_{-i}^k}(\pi_i, \pi_{-i}) &= G_i^{\pi_i^k, \pi_{-i}^k}(\rho_i, \rho_{-i}) \\ &= \rho_i(1 - \rho_{-i}) g_i^{i, -\hat{i}} + (1 - \rho_i)\rho_{-i} g_i^{\hat{i}, -i} + (1 - \rho_i)(1 - \rho_{-i}) g_i^{\hat{i}, -\hat{i}}. \end{aligned} \tag{6.15}$$

Then we have

$$\nabla_i G_i(\rho_{-i}) = (1-\rho_{-i}) g_i^{i,-\hat{i}} - \rho_{-i} g_i^{\hat{i},-i} - (1-\rho_{-i}) g_i^{\hat{i},\hat{-}i}, \tag{6.16}$$

and $\boldsymbol{\xi}(\pi_i, \pi_{-i}) = \boldsymbol{\xi}(\rho_i, \rho_{-i})$. Given a meta Nash policy pair $(\bar{\pi}_i, \bar{\pi}_{-i})$, where $\bar{\pi}_i = \bar{\rho}^i \pi_i^k + (1-\bar{\rho}^i)\hat{\pi}_i^k$, according to the Nash definition, we have:

$$\begin{pmatrix} \bar{\rho}_i \\ 1-\bar{\rho}_i \end{pmatrix}^T \begin{pmatrix} g_i^{i,-i} & g_i^{i,-\hat{i}} \\ g_i^{\hat{i},-i} & g_i^{\hat{i},-\hat{i}} \end{pmatrix} \begin{pmatrix} \bar{\rho}_{-i} \\ 1-\bar{\rho}_{-i} \end{pmatrix} \geq \begin{pmatrix} \rho_i \\ 1-\rho_i \end{pmatrix}^T \begin{pmatrix} g_i^{i,-i} & g_i^{i,-\hat{i}} \\ g_i^{\hat{i},-i} & g_i^{\hat{i},-\hat{i}} \end{pmatrix} \begin{pmatrix} \bar{\rho}_{-i} \\ 1-\bar{\rho}_{-i} \end{pmatrix}.$$

This implies:

$$\begin{aligned} (\bar{\rho}_i - \rho_i)\nabla_i G_i(\bar{\rho}_{-i}) \geq 0, \quad \bar{\rho}_i, \forall \rho_{-i} \in [0,1], \\ (\bar{\rho}_{-i} - \rho_{-i})\nabla_{-i} G_{-i}(\bar{\rho}_i) \geq 0, \quad \bar{\rho}_i, \forall \rho_{-i} \in [0,1]. \end{aligned} \tag{6.17}$$

When $\bar{\rho}_i, \bar{\rho}_{-i} \in (0,1)$ in accordance with the Nash condition in Equation 6.17, $\nabla_i G_i(\bar{\rho}_{-i}) = \nabla_{-i} G_{-i}(\bar{\rho}_i) = 0$. It shows that $(\bar{\pi}_i, \bar{\pi}_{-i})$ is a fixed point due to $\boldsymbol{\xi}(\bar{\pi}_i, \bar{\pi}_{-i}) = \boldsymbol{\xi}(\bar{\rho}_i, \bar{\rho}_{-i}) = \mathbf{0}$. For the boundary case, where $\bar{\rho}_i$ or $\bar{\rho}_{-i} \in \{0,1\}$, because they are constrained to the unit square $[0,1] \times [0,1]$, the gradients on the boundaries of the unit square are projected onto the unit square, which means additional points of zero gradient exist. In other words, $\nabla_i G_i$ and $\nabla_{-i} G_{-i}$ are still equal to zero in boundary case, and the $(\bar{\pi}_i, \bar{\pi}_{-i})$ is a fixed point in both cases.

Next, we determine what types of the fixed point that $(\bar{\pi}_i, \bar{\pi}_{-i})$ belongs to.

Definition 6.2. *A point $(\bar{\rho}_i, \bar{\rho}_{-i})$ is a fixed point if $\xi(\bar{\rho}_i, \bar{\rho}_{-i}) = 0$. It is stable if $H(\bar{\rho}_i, \bar{\rho}_{-i}) \succeq 0$, weak stable if $H(\bar{\rho}_i, \bar{\rho}_{-i}) \nprec 0$, unstable if $H(\bar{\rho}_i, \bar{\rho}_{-i}) \prec 0$, positive stable if all of its eigenvalues have a positive real part, and a strict saddle if $H(\bar{\theta})$ has an eigenvalue with a negative real part.*

According to the Equation 6.14, we have the exact Hessian Matrix for the restricted game:

$$H = \nabla \xi = \begin{pmatrix} 0 & g_i^{\hat{i},-\hat{i}} - g_i^{i,-\hat{i}} - g_i^{\hat{i},-i} \\ g_{-i}^{\hat{i},-\hat{i}} - g_{-i}^{i,-\hat{i}} - g_{-i}^{\hat{i},-i} & 0 \end{pmatrix}. \tag{6.18}$$

The eigenvalue λ of H can be computed:

$$\lambda^2 - \mathrm{Tr}(H)\lambda + \det(H) = \lambda^2 - (g_i^{\hat{i},-\hat{i}} - g_i^{i,-\hat{i}} - g_i^{\hat{i},-i})(g_{-i}^{\hat{i},-\hat{i}} - g_{-i}^{i,-\hat{i}} - g_{-i}^{\hat{i},-i}) = 0.$$

Denotes $\bar{g}_i := g_i^{\hat{i},-\hat{i}} - g_i^{i,-\hat{i}} - g_i^{\hat{i},-i}$, we have $\boldsymbol{\lambda} = \pm\sqrt{\bar{g}_i, \bar{g}_{-i}}$.

1. $\bar{g}_i \geq 0, \bar{g}_{-i} \geq 0$, $(\bar{\rho}_i, \bar{\rho}_{-i})$ is a stable point.

2. $\bar{g}_i > 0, \bar{g}_{-i} < 0$ or $\bar{g}_i < 0, \bar{g}_{-i} > 0$, all $\boldsymbol{\lambda}$ have a real imaginary and an imaginary negative eigenvalue, where $(\bar{\rho}_i, \bar{\rho}_{-i})$ is a saddle point.

3. $\bar{g}_i < 0, \bar{g}_{-i} < 0$, $\boldsymbol{\lambda}$ have one positive and one negative real part eigenvalue, where $(\bar{\rho}_i, \bar{\rho}_{-i})$ is a saddle point.

Therefore, in all the situations, $(\bar{\rho}_i, \bar{\rho}_{-i})$ is not unstable, and could be a stable point or saddle point. This is called a weak stable point. ■

According to Theorem 6.2, $(\bar{\pi}_i^k, \bar{\pi}_{-i}^k)$ is a weak stable fixed point of the under-layer game. When the game has multiple Nash equilibria, all the equilibria can derive the fixed-points for under-layer. Therefore, here an equilibrium is randomly selected. However, some equilibria can produce a more stable fixed point; in these cases, we leave it for future work. It also has a tighter lower bound than the independent trust region improvement seen in Remark 6.1:

Remark 6.1. *Let (ρ_i, ρ_{-i}) be a Nash equilibrium of the policy-space meta-game $\mathcal{M}(\pi_i, \hat{\pi}_i, \pi_{-i}, \hat{\pi}_{-i})$, which is used for computing the linear mixture policies $\bar{\pi}_i, \bar{\pi}_{-i}$. For simplicity, define $\bar{\rho}_i = 1 - \rho_i$, then we have the payoff improvement lower bound for $\bar{\pi}_i, \bar{\pi}_{-i}$:*

$$\eta_i(\bar{\pi}_i, \bar{\pi}_{-i}) - \eta_i(\pi_i, \pi_{-i}) \geq G_i^{\pi_i,\pi_{-i}}(\bar{\pi}_i, \bar{\pi}_{-i}) - \frac{4\gamma\epsilon_i}{(1-\gamma)^2}(\alpha_i\bar{\rho}_i + \alpha_{-i}\bar{\rho}_{-i} - \alpha_i\alpha_{-i}\bar{\rho}_i\bar{\rho}_{-i})^2, \tag{6.19}$$

that is a tighter lower bound compared with Theorem 6.1.

Extra Cost for Approximating and Solving Meta-Game. There are two major-cost sources in common meta-game analysis: approximating and solving the meta-game (Muller et al., 2020).). In our case, the meta-game is restricted to a local two-action game, where two actions π_i and $\hat{\pi}_i$ are close to each other. This locality property reduces the meta-game approximation cost (without extra sampling) by reusing the collected trajectories in the TPR step (Tuyls et al., 2020). The next crucial problem is how to solve the n-agent two-action meta-game, which consists of the 2^n entries of each of the n payoff matrices. This is much simpler than solving the under-layer game, which increases exponentially with state size, action size, agent number, and time horizons. As the general-sum matrix-form game has no fully polynomial-time approximation for computing Nash equilibria (Chen et al., 2006), it usually costs a lot to solve the game (Daskalakis et al., 2009). If we only require an approximated Nash equilibrium, when n is small, for example, $n \leq 10$, it is affordable to find a meta-game Nash equilibrium in a sub-exponential complexity (Lipton et al., 2003). However, this problem still exists when n is large. In this case, we could try utilizing special payoff structure assumptions (e.g., mean-field approximation (Yang et al., 2018a)) in the meta-game to reduce the matrix-form meta-game into the graphical game, which is polynomial-time computable (Daskalakis et al., 2009; Littman et al., 2002). It is then left for future work.

6.2.3 Improvement Against Weak Stable Fixed Point

Although the weak stable fixed point, $(\bar{\pi}_i^k, \bar{\pi}_{-i}^k)$, binds the policy update to another fixed point, there are still undesired saddle points according to Theorem 6.1. It is difficult to generalize for the other parts of the policy-space not reached by these saddle points, especially in the anti-coordination games (Lanctot et al., 2017). Similar to the extra-gradient method (Mertikopoulos et al., 2019), to escape the saddle points we apply the best response against the weak stable fixed point:

$$\pi_i^{k+1} = \arg\max_{\pi_i} \eta_i \left(\pi_i, \bar{\pi}_{-i}^k\right), \tag{6.20}$$

To perform the best response, we need another round to collect the experiences and do a gradient step in Equation 6.20. However, in practice, since we already have the trajectories in the TPR step and so the best response to the weak stable fixed point

Algorithm 6.5 Multi-Agent Trust Region Learning

```
1: Input: The initial policies: $\pi_1^0$, $\pi_2^0$.
2: for $k \in \{0, 1, 2, \cdots\}$ do
3:     Using $\pi_1^k$, $\pi_2^k$ to collect trajectories $\tau^k$.
4:     for $i \in \{1, 2\}$ do
5:         Compute a trust payoff region policy $\hat{\pi}_i$ using Equation 6.1. {Trust Payoff Region.}
6:     end for
7:     Solve meta-game $\mathcal{M}(\pi_1^k, \hat{\pi}_1^k, \pi_2^k, \hat{\pi}_2^k)$ and obtain a meta-game Nash $\rho_1^k, \rho_2^k$.
8:     Compute weak stable fixed point $\bar{\pi}_1^k$, $\bar{\pi}_2^k$. {Trust Stable Region.}
9:     for $i \in \{1, 2\}$ do
10:        Compute best response $\pi_i^{k+1}$ using Equation 6.20.{Best Response to a Weak Stable Fixed Point.}
11:    end for
12: end for
13: Output: $\pi_1$, $\pi_2$.
```

can be easily estimated through importance sampling. Alternatively, through defining $c_i \overset{\text{def}}{=} \min\left(1+\bar{c}, \max(1-\bar{c}, \frac{\pi_i(a_i|s)}{\bar{\pi}_i(a_i|s)})\right)$ as truncated importance sampling weights, we can re-write the best response update to Equation 6.20 into an equivalent form to the following one in terms of expectations: $\pi_i^{k+1} = \arg\max_{\pi_i} \mathbb{E}_{a_{-i}\sim\bar{\pi}_{-i}}[c_{-i}\eta_i\left(\pi_i, \pi_{-i}^k\right)]$.

6.2.4 Connections to Existing Methods

MATRL generalizes many existing methods with the best response. In extreme cases where the meta-game Nash is $(1, 1)$, which means always keeping the current policy, MATRL degenerates to independent learners. Here, the always best response to the other agents' current policies (π_i^k, π_{-i}^k) and $\pi_i^{k+1} = \arg\max_{\pi_i} \eta_i(\pi_i, \pi_{-i}^k)$. The policy prediction (Foerster et al., 2018a; Letcher et al., 2019; Zhang and Lesser, 2010), extra-gradient (Antipin, 2003) and exploitability descent (Lockhart et al., 2019; Tang et al., 2018) methods are also the instances of MATRL when meta-game Nash is $(0, 0)$. This is the best response to the most aggressive predicted policies $(\hat{\pi}_i^k, \hat{\pi}_{-i}^k)$ and $\pi_i^{k+1} = \arg\max_{\pi_i} \eta_i(\pi_i, \hat{\pi}_{-i}^k)$.

6.3 Experiments

We design the experiments to answer the following questions: 1). Can the MATRL method empirically contribute to the convergence? 2). How is the performance

Algorithm 6.6 Multi-Agent Trust Region Learning Algorithm (PPO Based).

```
1: **Input**: The initial policy parameters $\theta_1^0, theta_2^0$, initial value function parameters $\phi_1^0, \phi_2^0$ and $\epsilon$.
2: **for** $k \in 0, 1, 2, \cdots$ **do**
3:     Using $\pi_1^k(\theta_1^k)$, $\pi_2^k(\theta_2^k)$ to collect trajectories $\boldsymbol{\tau}_1^k, \boldsymbol{\tau}_2^k$.
4:     Compute GAE reward $\hat{R}_i$ for $i \in \{1, 2\}$.
5:     Compute estimated advantages $\hat{A}_1^k, \hat{A}_2^k$ based on the current value functions $V_{\phi_1^k}, V_{\phi_2^k}$.
6:     **for** $i \in 1, 2$ **do**
7:         Compute a trust payoff region policy $\hat{\pi}_i^k$ using Equation 6.1.
8:         Update the policy by maximizing the PPO-Clip objective: $\hat{\theta}_i^k = \arg\max_{\theta_i} \frac{1}{|\boldsymbol{\tau}_i^k| T} \sum_{\tau \in \boldsymbol{\tau}_1^k} \sum_{t=0}^{T} \min\left( \frac{\pi_i(a_t|s_t;\theta)}{\pi_i^k(a_{1,t}|s_t;\theta_i^k)} A_i^{\pi_1,\pi_2}(s_t, a_{1,t}, a_{2,t}), \quad g\left(\epsilon, A_i^{\pi_1,\pi_2}(s_t, a_{1,t}, a_{2,t})\right)\right)$, where $g$ is a clipping function.
9:         Fit value function by regression on mean-squared error:
               $$\phi_i^{k+1} = \arg\min_{\phi_i} \frac{1}{|\boldsymbol{\tau}_i^k| T} \sum_{\tau \in \boldsymbol{\tau}_i^k} \sum_{t=0}^{T} \left( V_\phi(s_t) - \hat{R}_{i,t} \right)^2$$
10:    **end for**
11:    Construct the meta-game $\mathcal{M}(\pi_1^k(\theta_1^k), \hat{\pi}_1^k(\hat{\theta}_1^k), \pi_2^k(\theta_2^k), \hat{\pi}_2^k(\hat{\theta}_2^k))$.
12:    Solve $\mathcal{M}$ and obtain meta Nash $\rho_1^k, \rho_2^k$.
13:    Compute aggregated weak stable fixed point $(\bar{\pi}_1^k, \bar{\pi}_2^k)$.
14:    **for** $i \in 1, 2$ **do**
15:        Compute $\pi_i^{(k+1)}$ which best responses to $\bar{\pi}_{-i}^k$ using Equation 6.20.
16:        Estimate the best response by importance sampling:
               $$\theta_i^{k+1} = \frac{\hat{\theta}_i^k}{|\boldsymbol{\tau}_i^k| T} \sum_{\tau \in \boldsymbol{\tau}_i^k} \sum_{t=0}^{T} g\left(\epsilon, \pi_i^k / \bar{\pi}_{-i}^k\right)$$
17:    **end for**
18: **end for**
19: **Output**: $\pi_1, \pi_2$.
```

of MATRL compared to the ILs with the same hyper-parameters and other strong MARL baselines in the discrete and continuous games with various agent number? 3). Do the meta-game and best response to the weak stable fixed point bring benefits? We first evaluate the convergence performance of MATRL in matrix form games to answer the first question and validate the effectiveness of convergence. For Question 2, we show that MATRL largely outperforms ILs (PPO (Schulman et al., 2017)) and other centralized baselines (QMIX (Rashid et al., 2018), and VDN (Sunehag

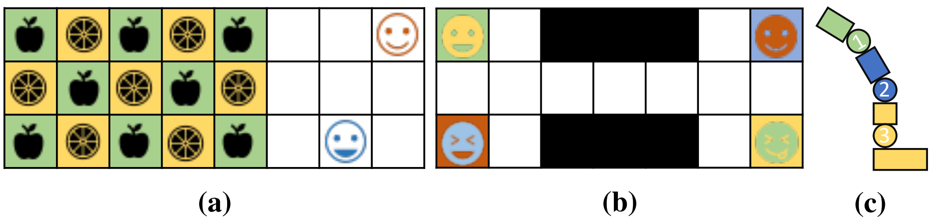


Figure 6.4: Multi-agent discrete and continuous action tasks: (a) two-agent checker (discrete), (b) four-agent switch (discrete), (c) three-agent MuJoCo hopper (continuous).

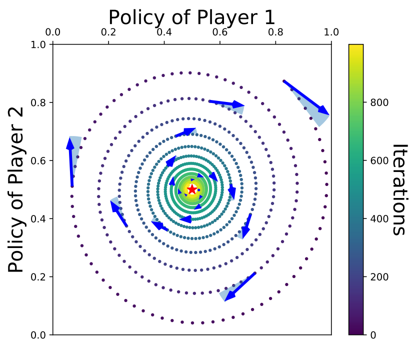


Figure 6.5: Learning dynamics in matching pennies. The blue arrow is trust payoff direction and the pale blue area is trust stable region.

et al., 2018) for discrete grid world games that have coordination problems. It also outperforms MADDPG (Lowe et al., 2017) for continuous multi-agent MuJoCo games. In these tasks, MATRL uses the same PPO configurations as ILs to examine the effectiveness of the trust region gradient-update mechanism, and we used official implementations for the other baselines. The step-by-step PPO based MATRL algorithm is given in Algorithm 6.6. Finally, the ablation study is conducted by removing the best response, called the MATRL w/o BR, and skipping the trust-stable region estimation, named IL-PP, which produces the best response to the predicted policy. These configurations provide insights about if the trust stable region and the best response can affect the MATRL's performance.

6.3.1 Experiment Environment Details

Random Matrix Games. We created a generator of 2×2 matrix games based on the category provided by (Pangallo et al., 2017). Coordination games have characteristics enabling one agent to improve the payoff without decreasing the payoff of the other agent. Anti-coordination games are ones where one agent improves the payoff while the other agent's payoff decreases. Both coordination and anti-coordination games can have two pure Nash equilibria and one mixed strategy Nash equilibrium. In cyclic games, the action selections of agents that is based on their actions will form a cycle, ensuring that there is no pure NE in the game. Instead only mixed strategy NE will be found.

Grid World Games. In two-player checker, as shown in Figure 6.4a, there is one sensitive player who gets reward 5 when they collect an apple and 5 when they collect a lemon; a less sensitive player gets 1 for apple and 1 for lemon. The learning goal is to let the sensitive player get apples and the other one get lemons to have a higher total reward. In four-player switch, as shown in Figure 6.4b, to reach the targets, agents need to figure out a way to go through a narrow corridor. The agent gets -1 for taking each step and 5 when arriving at a target. Four-player switch uses the same map as two-player switch, where two agents start from the left side and the others from the right side to go through the corridor to reach the targets. With more agents in four-player switch, learning becomes more challenging. MATRL agents achieved higher total rewards compared to baseline algorithms within the same number of steps.

Multi-Agent MuJoCo Tasks. We used the three-agent Hopper environment described in (de Witt et al., 2020), and Figure 6.4c, where three agents control three joints of the robot and learn to cooperate to move forward as far as possible. The agent is rewarded by the number of time steps that they move without falling. Each agent has 3 continuous output values as the action, and all the agents have a full observation of the states of size 17. We use the same hyper-parameters for MATRL, MATRL w/o BR, and IL-PP. For MADDPG agent, we use the hyper-parameters described in the paper (de Witt et al., 2020).

6.3.2 Matrix Game and Random Matrix Games

To illustrate the effectiveness of MATRL, we conducted an experiment on well known zero-sum matching pennies (MP) (Bruns, 2015) game and devise the 2×2 random matrix games. Using IGA (Singh et al., 2000) as ILs of MATRL, the learning

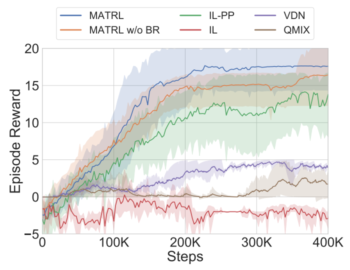
Table 6.1: Convergence rate and average convergence step in random 2×2 matrix games, where WoLF-IGA assumes the knowledge about the Nash equilibrium.

	CONVERGENCE RATE / AVERAGE CONVERGENCE STEP		
ALGORITHM	COORDINATION	ANTI-COORD.	CYCLIC
IGA	0.99 / 140.67	0.975 / 88.95	0.78 / 452.92
IGA-PP	0.99 / 138.56	0.975 / 83.11	0.809 / 432.98
WOLF-IGA	1.0 / 71.62	1.0 / 41.79	1.0 / 175.945
MATRL	0.99 / 86.54	0.9825 / 75.52	0.846 / 369.40

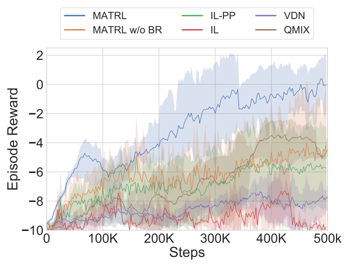
dynamics of MATRL on MP are shown in Figure 4.5. The MATRL reaches the Nash Equilibrium (central red star point) by updating the policies with the constraints from the trust stable region (the pale blue area). It would be trapped to a cyclic loop if following the original trust pay off direction (the dark blue arrow). To adequately examine the MATRL on border matrix games, we randomly generate three thousand 2×2 games for three types: coordination, anti-coordination, and cyclic (Pangallo et al., 2017). More details are provided about the game generation in Section 6.3.1. We choose the IGA and IGA-PP (Zhang and Lesser, 2010) as baselines, and the results in Table 6.1 show that MATRL has a higher convergence rate and needs fewer steps for convergence in all types of games.

6.3.3 Grid World Checker and Switch

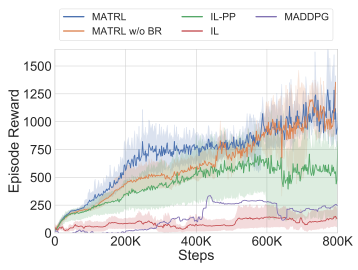
We evaluated MATRL in two grid world games from MA-Gym (Koul, 2019), two-agent checker, and four-agent switch, which are similar to games in Sunehag et al. (2018), but with more agents to examine if the MATRL can handle the games that have more than two agents. In the checker game, two agents cooperate in collecting fruits on the map; the sensitive agent gets 5 for apple and -5 for lemon, while the other one gets 1 and -1 respectively. So the optimal solution is to let the sensitive agent get the apple and the less sensitive one get the lemon. In the four-agent switch game, each side of the corridor has a room, each room has two agents, and the four agents try to go through one corridor to the target in the opposite room. Only one agent can pass the corridor at one time, and agents get -0.1 for each step and 5 for reaching targets, so they need to cooperate to get optimal scores. In both games, The agents can move in four directions and only partially observe their position.



(a) Two-agent checker.



(b) Four-agent switch.



(c) Three-agent hopper.

Figure 6.6: Learning curves in discrete and continuous tasks, each with 5 random seeds.

We compare the MATRL with the PPO based IL and two off-policy centralized training and decentralized execution baselines: VDN (Sunehag et al., 2018) and QMIX (Rashid et al., 2018). Results are given in Figure 6.6a and 6.6b, where MATRL has a stable improvement and outperforms other baselines. In two-player checker, using the best response, our method can achieve a total reward of 18, while the independent learners' rewards stay at -2. Besides, although PPO-based MATRL uses on-policy learning, it achieved better final results in fewer time steps compared to the off-policy baselines. As for the four-player switch, as shown in Figure 6.6b, MATRL can continuously improve the total rewards to 6.5, which is closest to the optimal score for this game when compared with other baselines. The result in the four-agent switch also demonstrates the effectiveness of MATRL in guaranteeing the stable policy improvement for the games that have more than two agents.

6.3.4 Multi-Agent MuJoCo Game

We also examined MATRL in a multi-agent continuous control task with a three-agent hopper from (de Witt et al., 2020). Here, three agents cooperatively control each part of a hopper to move forward. The agents are rewarded with distance and the number of steps they make before falling. Figure 6.6c shows that MATRL significantly outperforms IL, MADDPG, and also the benchmarks in de Witt et al. (2020) within the same amount of time.

6.3.5 Effect and Cost of Trust Stable Region and Best Response to Fixed Point

This section analyzes the effect of the TSR from meta-game Nash and the best response against the weak stable fixed point. In Figure 6.6 the ablation settings are obtained by removing the trust stable region (IL-PP) and the best response (MATRL w/o BR). In Figure 6.6, we can observe that in all the tasks, without the best response to the fixed point, the learning curves of MATRL o/w BR have higher variance and the lowest final scores. This establishes the importance of the best response to stabilize and improve performance. On the other hand, without the TSR to select a fixed point, the MATRL recovers independent learner with the policy prediction (IL-PP) (Foerster et al., 2018a; Zhang and Lesser, 2010). Similarly, the curves of IL-PP have lower final scores, and the convergence speed is not as good as the MATRL, which suggests that the TSR provides benefits. The MATRL w/o BR has lower variance compared to the IL-PP, which reveals the trust stable region can stabilize the learning via weak stable fixed point constraints. Finally, when comparing to IL and IL-PP, the time for each training step in MATRL is empirically about 1.1 times slower. We think this extra computational cost from the TSR and the best response is acceptable given the performance improvement brought by these operations.

6.4 Summary

We proposed and analyzed the trust region method for multi-agent learning problems, which considers the trust payoff region and the trust stable region to meed the multi-agent learning objectives. In practice, using independent trust payoff learners, we provide a convenient way to approximate the trust stable region via policy-space meta-game. This ensures that the MATRL is generalized, flexible, and easily implemented to deal with any kind of multi-agent scenarios. Our experiment's results show that the MATRL significantly outperforms the independent learners with the same configurations and even other MATRL baselines on both continuous and discrete games with various agent numbers.

Chapter 7

Conclusion and Future Work

One long-term goal of AI is to create autonomous agents that can be deployed in the dynamics of the real-world that are capable of interacting with others. Two crucial capabilities in service of this goal are learning and interacting with others. Learning is necessary because complex tasks cannot be solved manually and many elements in an environment are unknown, such as the transition, reward, and policies of others. Thus, agents need to learn to adapt to the environment and other agents. In the study of multi-agent systems in which one or more of the autonomous entities improves automatically through experience (Tuyls and Stone, 2018), reinforcement learning is one of the most popular solutions for agent learning. On one hand, reinforcement learning in single-agent learning has been widely studied, but there are additional interactions with others when shifting from single to multi-agent reinforcement learning. The interactions indicate that agents' behaviors impact other agents and need to take others agents' impacts into account during learning. On the other hand, in many cases, other agents are learning and updating their policies simultaneously. Therefore, for a single naive agent, the perceived environment is non-stationary, and I set out to address these questions. Further, the modeling of mutual influence is examined to benefit multi-agent learning. In this chapter, I summarize the contributions of this thesis to address these problems. I also present an outlook for future mutual influence modeling.

7.1 Contributions

This thesis aimed to investigate taking mutual influence (during interactions) into account to shape multi-agent learning. Further, this work explored several multi-agent learning algorithms that focus on different aspects of modeling mutual influence, including learning recursive reasoning, learning diverse behaviors, and learning to improve in a trust region. I also proposed and assessed methods for different challenges, clearly demonstrating progress on different fronts.

In **Chapter** 3 and **Chapter** 4, inspired by the recursive reasoning capability of human intelligence, I immerse the recursive reasoning framework into MARL and formulate the multi-agent policy search problem into a hierarchy of nested single agent policy search problems. I propose a novel framework, *Probabilistic Recursive Reasoning (PR2)*, and its extension, *Generalized Recursive Reasoning (GR2)*, that recognize agents' bounded rationality and can model their corresponding suboptimal behaviors. The recursive reasoning idea is inspired by the cognitive hierarchy theory proposed by Camerer et al. (2004), assuming that agents can possess different reasoning levels during interactions. The idea begins with $level$-0 (L0 for short) non-strategic thinkers who do not model their opponents. L1 thinkers are more sophisticated than L0 thinkers; they believe the opponents are all at L0 and then act accordingly. With the growth of k, Lk agents think in increasing order of sophistication then provide the best response to all possible lower-level opponents. I immerse the PR2/GR2 framework into MARL through graphical models and derive the practical GR2 soft actor-critic algorithm. Theoretically, I prove the existence of the Perfect Bayesian Equilibrium in the GR2 framework and the convergence of GR2 policy gradient methods on two-player normal-form games The results justify main theoretical findings and the effectiveness of bounded-rationality modeling.

Work on behavioral diversity is included in **Chapter** 5, which demonstrates one of the first value decomposition approaches to enforce the emergence of diverse behaviors during learning. I try to measure the mutual influence by approximating the joint Q-value based on the individual values as inputs via quality and diversity value factorization methods with centralized training and a decentralized execution paradigm. The tool I use to measure mutual influence promotes diverse behaviors by a probabilistic framework DPP, which can measure how likely a diverse joint action will be sampled from a valid action space. To improve the sampling efficiency, I instead perform sequential sampling across different partitions that enjoy linear time complexity in Q-DPP. The proposed decomposition also recovers many existing value factorization methods, such as VDN (Sunehag et al., 2018), QMIX (Rashid et al., 2018), and QTRAN (Son et al., 2019), when achieving the maximum diversity.

Chapter 6 presents the first instance of a multi-agent trust region method that allows agents to consider the policy-space mutual influence and shape learning behavior that gives both trust payoff region and trust stable region guarantees This work provides theoretical and experimental results, demonstrating MATRL to be a promising approach that is able to boost many existing multi-agent reinforcement learning algorithms. MATRL is essentially a step-size control mechanism for ILs to plug into any policy-based independent learner. On one hand, MATRL keeps the scalable and convenient advantages of independent learners. On the other hand, MATRL brings a stable and efficient performance improvement without too much extra cost in multi-agent learning. MATRL can help to significantly reduce the multi-agent training cost in many successful multi-agent applications.

7.2 Future Work

While this thesis explored some of the questions mentioned above in modeling mutual influence, the following important open challenges remain:

- **Extensions to Recursive Reasoning Methods**. The recursive reasoning frameworks proposed in this thesis are limited, typically to a small number of agents and simple scenarios. One hand, when the agent number increases, there is a scalability issue in which it is difficult to approximate the interactions within a large population of agents. Therefore, performing recursive reasoning with massive agents in complex environments is difficult for now due to the curse of dimensionality and the exponential growth of agent interactions. It is worth investigating how to embed solutions such as mean-field theory (Yang et al., 2018a)) into mutual influence models. On the other hand, learning an accurate value function to guide recursive reasoning in complex scenarios is challenging, so problems should be simplified further. Instead of learning value functions from scratch, the focus can be on using the proposed recursive reasoning methods on modeling interactions. More specifically, soft RL and cognitive hierarchy can be used to model the bounded rationality in human-machine interactions.

- **Exploring Advanced Methods and Emergence of Behavior Diversity**. Q-DPP has expanded DPP to enforce the action level diversity in various discrete grid-world games. Making the Q-DPP work well in continuous spaces and policy-space diversity problems is still a challenge. Furthermore, although DPP is a good measurement of the diversity of given input feature vectors, it is still difficult to represent many things (e.g., neural network-based policies) via vectors. Therefore, it worth developing models or mechanisms for complex and diverse behaviors that can emerge automatically. Such emergence of behavior diversity is an essential step toward many real-world applications, such as self-driving cars and online games.

- **Modeling Temporal Influence**. In most existing multi-agent RL settings, the feedback of agent actions is supposed to be instant. However, actions in the real-world sometimes have durations and can be performed consecutively or successively. For example, a player passes the ball in a football game; however, the action at one time step might have negligible reciprocal effects on other agents. Therefore, arguably, modeling the action duration or temporal abstraction may be of particular importance to consider the influence of others. Based on this argument, the following questions can be investigated: can agents learn an action duration or temporal abstraction from its experiences in a multi-agent game? Does learning fine-grained temporal abstractions in games give a better understanding of other agents?

- **Improving the Multi-Agent Trust Region Method**. The multi-agent trust region method proposed in this thesis is still problematic for estimating meta-game Nash equilibrium when a large number of agents exists. It is meaningful to find a lower-cost way to approximate the meta-game equilibrium. Alternatively, a better solution should be found to approximate the fixed-stable points of under-layer games than Nash equilibrium to guide multi-agent learning. Furthermore, games commonly have multiple equilibria; therefore, it is worth investigating a better way to select the best meta-agent Nash equilibrium rather than it being randomly chosen.

Although many challenges remain to be addressed in MARL problems, I believe this thesis takes a significant step forward in MARL research as it introduces novel algorithms that consider mutual influence and provides new insights to the community.

Bibliography

Abdallah, S. and Lesser, V. (2008). A multiagent reinforcement learning algorithm with non-linear dynamics. *JAIR*, 33:521–549.

Affandi, R. H., Fox, E., Adams, R., and Taskar, B. (2014). Learning the parameters of determinantal point process kernels. In *International Conference on Machine Learning*, pages 1224–1232.

Albrecht, S. V. and Ramamoorthy, S. (2012). Comparative evaluation of mal algorithms in a diverse set of ad hoc team problems. In *Proceedings of the 11th International Conference on Autonomous Agents and Multiagent Systems - Volume 1*, AAMAS '12, page 349–356, Richland, SC. International Foundation for Autonomous Agents and Multiagent Systems.

Albrecht, S. V. and Stone, P. (2018). Autonomous agents modelling other agents: A comprehensive survey and open problems. *Artificial Intelligence*, 258:66–95.

Antipin, A. (2003). Extragradient approach to the solution of two person non-zero sum games. In *Optimization and Optimal Control*, pages 1–28. World Scientific.

Balduzzi, D., Garnelo, M., Bachrach, Y., Czarnecki, W., Perolat, J., Jaderberg, M., and Graepel, T. (2019). Open-ended learning in symmetric zero-sum games. In Chaudhuri, K. and Salakhutdinov, R., editors, *Proceedings of the 36th International Conference on Machine Learning*, volume 97 of *Proceedings of Machine Learning Research*, pages 434–443, Long Beach, California, USA. PMLR.

Balduzzi, D., Racaniere, S., Martens, J., Foerster, J., Tuyls, K., and Graepel, T. (2018). The mechanics of n-player differentiable games. In Dy, J. and Krause, A., editors, *Proceedings of the 35th International Conference on Machine Learning*, volume 80 of *Proceedings of Machine Learning Research*, pages 354–363, Stockholmsmässan, Stockholm Sweden. PMLR.

Banerjee, D. and Sen, S. (2007). Reaching pareto-optimality in prisoner's dilemma using conditional joint action learning. *Autonomous Agents and Multi-Agent Systems*, 15(1):91–108.

Bellman, R. (1952). On the theory of dynamic programming. *Proceedings of the National Academy of Sciences of the United States of America*, 38(8):716.

Benndorf, V., Kübler, D., and Normann, H.-T. (2017). Depth of reasoning and information revelation: An experiment on the distribution of k-levels. *International Game Theory Review*, 19(04):1750021.

Berner, C., Brockman, G., Chan, B., Cheung, V., Dębiak, P., Dennison, C., Farhi, D., Fischer, Q., Hashme, S., Hesse, C., et al. (2019). Dota 2 with large scale deep reinforcement learning. *arXiv preprint arXiv:1912.06680*.

Bolander, T. and Andersen, M. B. (2011). Epistemic planning for single-and multi-agent systems. *Journal of Applied Non-Classical Logics*, 21(1):9–34.

Bouzy, B. and Métivier, M. (2010). Multi-agent learning experiments on repeated matrix games. In *Proceedings of the 27th International Conference on Machine Learning (ICML-10)*, pages 119–126.

Bowling, M. (2005). Convergence and no-regret in multiagent learning. In *NIPS*, pages 209–216.

Bowling, M. and Veloso, M. (2001a). Convergence of gradient dynamics with a variable learning rate. In *ICML*, pages 27–34.

Bowling, M. and Veloso, M. (2001b). Rational and convergent learning in stochastic games. In *International joint conference on artificial intelligence*, volume 17, pages 1021–1026. Lawrence Erlbaum Associates Ltd.

Bowling, M. and Veloso, M. (2002). Multiagent learning using a variable learning rate. *Artificial Intelligence*, 136(2):215–250.

Bowling, M. and Veloso, M. (2004). Existence of multiagent equilibria with limited agents. *Journal of Artificial Intelligence Research*, 22:353–384.

Brown, G. W. (1951). Iterative solution of games by fictitious play. *Activity analysis of production and allocation*, 13(1):374–376.

Bruns, B. R. (2015). Names for games: Locating 2$\times$ 2 games. *Games*, 6(4):495–520.

Buşoniu, L., Babuška, R., and De Schutter, B. (2010). Multi-agent reinforcement learning: An overview. In *Innovations in multi-agent systems and applications-1*, pages 183–221. Springer.

Camerer, C. F. (2003). Behavioural studies of strategic thinking in games. *Trends in cognitive sciences*, 7(5):225–231.

Camerer, C. F., Ho, T.-H., and Chong, J.-K. (2004). A cognitive hierarchy model of games. *The Quarterly Journal of Economics*, 119(3):861–898.

Cao, Y., Yu, W., Ren, W., and Chen, G. (2012). An overview of recent progress in the study of distributed multi-agent coordination. *IEEE Transactions on Industrial informatics*, 9(1):427–438.

Celis, L. E., Deshpande, A., Kathuria, T., Straszak, D., and Vishnoi, N. K. (2016). On the complexity of constrained determinantal point processes. *arXiv preprint arXiv:1608.00554*.

Celis, L. E., Keswani, V., Straszak, D., Deshpande, A., Kathuria, T., and Vishnoi, N. K. (2018). Fair and diverse dpp-based data summarization. *ICML*.

Chakrabarti, S. and Topolyan, I. (2011). A direct proof of the existence of sequential equilibrium and a backward induction characterization.

Chatterjee, K., Majumdar, R., and Jurdziński, M. (2004). On nash equilibria in stochastic games. In *International Workshop on Computer Science Logic*, pages 26–40. Springer.

Chen, L., Zhang, G., and Zhou, E. (2018). Fast greedy map inference for determinantal point process to improve recommendation diversity. In *Advances in Neural Information Processing Systems*, pages 5622–5633.

Chen, X., Deng, X., and Teng, S.-H. (2006). Computing nash equilibria: Approximation and smoothed complexity. In *2006 47th Annual IEEE Symposium on Foundations of Computer Science (FOCS'06)*, pages 603–612. IEEE.

Chong, J.-K., Ho, T.-H., and Camerer, C. (2016). A generalized cognitive hierarchy model of games. *Games and Economic Behavior*, 99:257–274.

Claus, C. and Boutilier, C. (1998). The dynamics of reinforcement learning in cooperative multiagent systems. *AAAI*, 1998:746–752.

Coricelli, G. and Nagel, R. (2009). Neural correlates of depth of strategic reasoning in medial prefrontal cortex. *Proceedings of the National Academy of Sciences*, 106(23):9163–9168.

Crawford, V. P., Costa-Gomes, M. A., and Iriberri, N. (2013). Structural models of nonequilibrium strategic thinking: Theory, evidence, and applications. *Journal of Economic Literature*, 51(1):5–62.

Da Silva, B. C., Basso, E. W., Bazzan, A. L., and Engel, P. M. (2006). Dealing with non-stationary environments using context detection. In *Proceedings of the 23rd international conference on Machine learning*, pages 217–224. ACM.

Daskalakis, C., Goldberg, P. W., and Papadimitriou, C. H. (2009). The complexity of computing a nash equilibrium. *SIAM Journal on Computing*, 39(1):195–259.

De Weerd, H., Verbrugge, R., and Verheij, B. (2013a). Higher-order theory of mind in negotiations under incomplete information. In *International Conference on Principles and Practice of Multi-Agent Systems*, pages 101–116. Springer.

De Weerd, H., Verbrugge, R., and Verheij, B. (2013b). How much does it help to know what she knows you know? an agent-based simulation study. *Artificial Intelligence*, 199:67–92.

de Weerd, H., Verbrugge, R., and Verheij, B. (2017). Negotiating with other minds: the role of recursive theory of mind in negotiation with incomplete information. *Autonomous Agents and Multi-Agent Systems*, 31(2):250–287.

de Witt, C. S., Peng, B., Kamienny, P.-A., Torr, P., Böhmer, W., and Whiteson, S. (2020). Deep multi-agent reinforcement learning for decentralized continuous cooperative control.

Dennett, D. C. (1991). Two contrasts: folk craft versus folk science, and belief versus opinion. *The future of folk psychology: Intentionality and cognitive science*, pages 135–148.

Deshpande, A., Rademacher, L., Vempala, S., and Wang, G. (2006). Matrix approximation and projective clustering via volume sampling. *Theory of Computing*, 2(1):225–247.

Devaine, M., Hollard, G., and Daunizeau, J. (2014). Theory of mind: did evolution fool us? *PloS One*, 9(2):e87619.

Devetag, G. and Warglien, M. (2003). Games and phone numbers: Do short-term memory bounds affect strategic behavior? *Journal of Economic Psychology*, 24(2):189–202.

Devlin, J., Chang, M.-W., Lee, K., and Toutanova, K. (2019). BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4171–4186, Minneapolis, Minnesota. Association for Computational Linguistics.

Dinh, L., Sohl-Dickstein, J., and Bengio, S. (2016). Density estimation using real nvp. *ICLR*.

Doshi, P., Gmytrasiewicz, P., and Durfee, E. (2020). Recursively modeling other agents for decision making: A research perspective. *Artificial Intelligence*, 279:103202.

Doshi, P. and Gmytrasiewicz, P. J. (2006). On the difficulty of achieving equilibrium in interactive pomdps. In *Proceedings of the National Conference on Artificial Intelligence*, volume 21, page 1131. Menlo Park, CA; Cambridge, MA; London; AAAI Press; MIT Press; 1999.

Doshi, P. and Gmytrasiewicz, P. J. (2009). Monte carlo sampling methods for approximating interactive pomdps. *Journal of Artificial Intelligence Research*, 34:297–337.

Doshi, P. and Perez, D. (2008). Generalized point based value iteration for interactive pomdps. In *AAAI*, pages 63–68.

Doshi, P., Zeng, Y., and Chen, Q. (2009). Graphical models for interactive pomdps: representations and solutions. *Autonomous Agents and Multi-Agent Systems*, 18(3):376.

Eccles, T., Hughes, E., Kramár, J., Wheelwright, S., and Leibo, J. Z. (2019). Learning reciprocity in complex sequential social dilemmas. *arXiv preprint arXiv:1903.08082*.

Elfeki, M., Couprie, C., Riviere, M., and Elhoseiny, M. (2019). Gdpp: Learning diverse generations using determinantal point processes. In *International Conference on Machine Learning*, pages 1774–1783.

Foerster, J., Chen, R. Y., Al-Shedivat, M., Whiteson, S., Abbeel, P., and Mordatch, I. (2018a). Learning with opponent-learning awareness. In *Proceedings of the 17th International Conference on Autonomous Agents and MultiAgent Systems*, pages 122–130. International Foundation for Autonomous Agents and Multiagent Systems.

Foerster, J., Farquhar, G., Afouras, T., Nardelli, N., and Whiteson, S. (2018b). Counterfactual multi-agent policy gradients. *AAAI*.

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

Fox, R., Pakman, A., and Tishby, N. (2016). Taming the noise in reinforcement learning via soft updates. In *Proceedings of the Thirty-Second Conference on Uncertainty in Artificial Intelligence*, pages 202–211. AUAI Press.

Fudenberg, D., Drew, F., Levine, D. K., and Levine, D. K. (1998). *The theory of learning in games*, volume 2. MIT press.

Gal, Y. and Pfeffer, A. (2003). A language for modeling agents' decision making processes in games. In *Proceedings of the second international joint conference on Autonomous agents and multiagent systems*, pages 265–272. ACM.

Gal, Y. and Pfeffer, A. (2008). Networks of influence diagrams: a formalism for representing agents' beliefs and decision-making processes. *Journal of Artificial Intelligence Research*, 33:109–147.

Gallese, V. and Goldman, A. (1998). Mirror neurons and the simulation theory of mind-reading. *Trends in cognitive sciences*, 2(12):493–501.

Gmytrasiewicz, P. J. and Doshi, P. (2005). A framework for sequential planning in multi-agent settings. *JAIR*, 24:49–79.

Gmytrasiewicz, P. J. and Durfee, E. H. (1995). A rigorous, operational formalization of recursive modeling. In *ICMAS*, pages 125–132.

Gmytrasiewicz, P. J. and Durfee, E. H. (2000). Rational coordination in multi-agent environments. *Autonomous Agents and Multi-Agent Systems*, 3(4):319–350.

Gmytrasiewicz, P. J., Durfee, E. H., and Wehe, D. K. (1991). A decision-theoretic approach to coordinating multi-agent interactions. In *IJCAI*, volume 91, pages 63–68.

Goldman, A. I. et al. (2012). Theory of mind. *The Oxford handbook of philosophy of cognitive science*, pages 402–424.

Goodfellow, I., Bengio, Y., and Courville, A. (2016). *Deep learning*. MIT press.

Goodfellow, I., Pouget-Abadie, J., Mirza, M., Xu, B., Warde-Farley, D., Ozair, S., Courville, A., and Bengio, Y. (2014). Generative adversarial nets. In *NIPS*, pages 2672–2680.

Gopnik, A. and Wellman, H. M. (1992). Why the child's theory of mind really is a theory. *Mind & Language*, 7(1-2):145–171.

Gordon, R. M. (1986). Folk psychology as simulation. *Mind & Language*, 1(2):158–171.

Gracia-Lázaro, C., Floría, L. M., and Moreno, Y. (2017). Cognitive hierarchy theory and two-person games. *Games*, 8(1):1.

Grau-Moya, J., Leibfried, F., and Bou-Ammar, H. (2018). Balancing two-player stochastic games with soft q-learning. *IJCAI*.

Greenwald, A., Hall, K., and Serrano, R. (2003). Correlated q-learning. In *ICML*, volume 3, pages 242–249.

Grover, A., Al-Shedivat, M., Gupta, J. K., Burda, Y., and Edwards, H. (2018). Learning policy representations in multiagent systems. *ICML*.

Haarnoja, T., Tang, H., Abbeel, P., and Levine, S. (2017). Reinforcement learning with deep energy-based policies. *NIPS*.

Haarnoja, T., Zhou, A., Abbeel, P., and Levine, S. (2018). Soft actor-critic: Off-policy maximum entropy deep reinforcement learning with a stochastic actor. In Dy, J. and Krause, A., editors, *Proceedings of the 35th International Conference on Machine Learning*, volume 80 of *Proceedings of Machine Learning Research*, pages 1861–1870, Stockholmsmässan, Stockholm Sweden. PMLR.

Haddad, W. M. and Chellaboina, V. (2011). *Nonlinear dynamical systems and control: a Lyapunov-based approach*. Princeton University Press.

Han, Y. and Gmytrasiewicz, P. (2018). Learning others' intentional models in multi-agent settings using interactive pomdps. In *Advances in Neural Information Processing Systems*, pages 5634–5642.

Han, Y. and Gmytrasiewicz, P. (2019). Ipomdp-net: A deep neural network for partially observable multi-agent planning using interactive pomdps. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 33, pages 6062–6069.

Hansen, N., Müller, S. D., and Koumoutsakos, P. (2003). Reducing the time complexity of the derandomized evolution strategy with covariance matrix adaptation (cma-es). *Evolutionary computation*, 11(1):1–18.

Harsanyi, J. C. (1962). Bargaining in ignorance of the opponent's utility function. *Journal of Conflict Resolution*, 6(1):29–38.

Harsanyi, J. C. (1967). Games with incomplete information played by bayesian players, i–iii part i. the basic model. *Management science*, 14(3):159–182.

He, H., Boyd-Graber, J., Kwok, K., and Daumé III, H. (2016). Opponent modeling in deep reinforcement learning. In *ICML*, pages 1804–1813.

Heess, N., Silver, D., and Teh, Y. W. (2012). Actor-critic reinforcement learning with energy-based policies. In *EWRL*, pages 43–58.

Heinrich, J. and Silver, D. (2016). Deep reinforcement learning from self-play in imperfect-information games. *arXiv preprint arXiv:1603.01121*.

Hernandez-Leal, P., Kaisers, M., Baarslag, T., and de Cote, E. M. (2017). A survey of learning in multiagent environments: Dealing with non-stationarity. *arXiv preprint arXiv:1707.09183*.

Hernandez-Leal, P., Kartal, B., and Taylor, M. E. (2019). Agent modeling as auxiliary task for deep reinforcement learning. In *Proceedings of the AAAI Conference on Artificial Intelligence and Interactive Digital Entertainment*, volume 15, pages 31–37.

Hong, Z.-W., Su, S.-Y., Shann, T.-Y., Chang, Y.-H., and Lee, C.-Y. (2018). A deep policy inference q-network for multi-agent systems. In *Proceedings of the 17th International Conference on Autonomous Agents and MultiAgent Systems*, AAMAS '18, page 1388–1396, Richland, SC. International Foundation for Autonomous Agents and Multiagent Systems.

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

Jordan, M. I., Ghahramani, Z., Jaakkola, T. S., and Saul, L. K. (1999). An introduction to variational methods for graphical models. *Machine learning*, 37(2):183–233.

Jordan, P. R. and Wellman, M. P. (2009). Generalization risk minimization in empirical game models. In *Proceedings of The 8th International Conference on Autonomous Agents and Multiagent Systems-Volume 1*, pages 553–560.

Kakade, S. M. and Langford, J. (2002). Approximately optimal approximate reinforcement learning. In *ICML*.

Kimbrough, E. O., Robalino, N., and Robson, A. J. (2014). The evolution of'theory of mind': Theory and experiments. *Cowles Foundation Discussion Paper*.

Kingma, D. P. and Welling, M. (2014). Auto-encoding variational bayes. *ICLR*.

Kitano, H., Tambe, M., Stone, P., Veloso, M., Coradeschi, S., Osawa, E., Matsubara, H., Noda, I., and Asada, M. (1997). The robocup synthetic agent challenge 97. In *Robot Soccer World Cup*, pages 62–73. Springer.

Klopf, A. H. (1972). *Brain function and adaptive systems: a heterostatic theory*. Number 133. Air Force Cambridge Research Laboratories, Air Force Systems Command, United ....

Koul, A. (2019). A collection of multi agent environments based on OpenAI gym.

Kreps, D. M. and Wilson, R. (1982). Sequential equilibria. *Econometrica: Journal of the Econometric Society*, pages 863–894.

Krizhevsky, A., Sutskever, I., and Hinton, G. E. (2012). Imagenet classification with deep convolutional neural networks. In *Advances in neural information processing systems*, pages 1097–1105.

Kulesza, A., Taskar, B., et al. (2012). Determinantal point processes for machine learning. *Foundations and Trends® in Machine Learning*, 5(2–3):123–286.

Lake, B. M., Ullman, T. D., Tenenbaum, J. B., and Gershman, S. J. (2017). Building machines that learn and think like people. *Behavioral and Brain Sciences*, 40.

Lanctot, M., Zambaldi, V., Gruslys, A., Lazaridou, A., Perolat, J., Silver, D., Graepel, T., et al. (2017). A unified game-theoretic approach to multiagent reinforcement learning. In *Advances in Neural Information Processing Systems*, pages 4190–4203.

Laurent, G. J., Matignon, L., Fort-Piat, L., et al. (2011). The world of independent learners is not markovian. *International Journal of Knowledge-based and Intelligent Engineering Systems*, 15(1):55–64.

LeCun, Y., Bengio, Y., and Hinton, G. (2015). Deep learning. *nature*, 521(7553):436–444.

Leslie, D. S., Collins, E., et al. (2003). Convergent multiple-timescales reinforcement learning algorithms in normal form games. *The Annals of Applied Probability*, 13(4):1231–1251.

Leslie, D. S. and Collins, E. J. (2005). Individual q-learning in normal form games. *SIAM Journal on Control and Optimization*, 44(2):495–514.

Leslie, D. S. and Collins, E. J. (2006). Generalised weakened fictitious play. *Games and Economic Behavior*, 56(2):285–298.

Letcher, A., Foerster, J., Balduzzi, D., Rocktäschel, T., and Whiteson, S. (2019). Stable opponent shaping in differentiable games. In *International Conference on Learning Representations*.

Levin, D. and Zhang, L. (2019). Bridging level-k to nash equilibrium. *Available at SSRN 2934696*.

Levine, S. (2018). Reinforcement learning and control as probabilistic inference: Tutorial and review. *arXiv preprint arXiv:1805.00909*.

Li, C., Sra, S., and Jegelka, S. (2016). Fast mixing markov chains for strongly rayleigh measures, dpps, and constrained sampling. In *Advances in Neural Information Processing Systems*, pages 4188–4196.

Li, M., Qin, Z., Jiao, Y., Yang, Y., Wang, J., Wang, C., Wu, G., and Ye, J. (2019). Efficient ridesharing order dispatching with mean field multi-agent reinforcement learning. In *The World Wide Web Conference*, pages 983–994. ACM.

Lillicrap, T. P., Hunt, J. J., Pritzel, A., Heess, N., Erez, T., Tassa, Y., Silver, D., and Wierstra, D. (2015). Continuous control with deep reinforcement learning. *arXiv preprint arXiv:1509.02971*.

Lin, T., Zhou, Z., Mertikopoulos, P., and Jordan, M. I. (2020). Finite-time last-iterate convergence for multi-agent learning in games. *arXiv preprint arXiv:2002.09806*.

Lipton, R. J., Markakis, E., and Mehta, A. (2003). Playing large games using simple strategies. In *Proceedings of the 4th ACM conference on Electronic commerce*, pages 36–41.

Littman, M. L. (1994). Markov games as a framework for multi-agent reinforcement learning. In *Machine Learning Proceedings 1994*, pages 157–163. Elsevier.

Littman, M. L. (2001). Friend-or-foe q-learning in general-sum games. In *ICML*, volume 1, pages 322–328.

Littman, M. L., Kearns, M. J., and Singh, S. P. (2002). An efficient, exact algorithm for solving tree-structured graphical games. In *Advances in Neural Information Processing Systems*, pages 817–823.

Liu, M., Zhou, M., Zhang, W., Zhuang, Y., Wang, J., Liu, W., and Yu, Y. (2020). Multi-agent interactions modeling with correlated policies. In *International Conference on Learning Representations*.

Liu, Q. and Wang, D. (2016). Stein variational gradient descent: A general purpose bayesian inference algorithm. In *NIPS*, pages 2378–2386.

Lockhart, E., Lanctot, M., Julien, P., Lespiau, J.-B., Morrill, D., TImbers, F., and Tuyls, K. (2019). Computing approximate equilibria in sequential adversarial games by exploitability descent. In *IJCAI 2019*, pages 464–470.

Lowe, R., Wu, Y., Tamar, A., Harb, J., Abbeel, O. P., and Mordatch, I. (2017). Multi-agent actor-critic for mixed cooperative-competitive environments. In *NIPS*, pages 6379–6390.

Lyapunov, A. (1992). *General Problem of the Stability Of Motion*. Control Theory and Applications Series. Taylor & Francis.

Macchi, O. (1977). The fermion process—a model of stochastic point process with repulsive points. In *Transactions of the Seventh Prague Conference on Information Theory, Statistical Decision Functions, Random Processes and of the 1974 European Meeting of Statisticians*, pages 391–398. Springer.

Mahajan, A., Rashid, T., Samvelyan, M., and Whiteson, S. (2019). Maven: Multi-agent variational exploration. In *Advances in Neural Information Processing Systems*, pages 7611–7622.

Marquez, H. J. (2003). *Nonlinear control systems: analysis and design*, volume 1. Wiley.

Matignon, L., Laurent, G. J., and Le Fort-Piat, N. (2007). Hysteretic q-learning: an algorithm for decentralized reinforcement learning in cooperative multi-agent teams. In *2007 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 64–69. IEEE.

Mazumdar, E., Ratliff, L. J., and Sastry, S. S. (2020). On gradient-based learning in continuous games. *SIAM Journal on Mathematics of Data Science*, 2(1):103–131.

Mazumdar, E. V., Jordan, M. I., and Sastry, S. S. (2019). On finding local nash equilibria (and only local nash equilibria) in zero-sum games. *arXiv preprint arXiv:1901.00838*.

Mertikopoulos, P., Lecouat, B., Zenati, H., Foo, C.-S., Chandrasekhar, V., and Piliouras, G. (2019). Optimistic mirror descent in saddle-point problems: Going the extra(-gradient) mile. In *International Conference on Learning Representations*.

Mescheder, L., Nowozin, S., and Geiger, A. (2017). The numerics of gans. In *NIPS*, pages 1825–1835.

Mnih, V., Kavukcuoglu, K., Silver, D., Graves, A., Antonoglou, I., Wierstra, D., and Riedmiller, M. (2013). Playing atari with deep reinforcement learning. *NIPS Deep Learning Workshop*.

Mnih, V., Kavukcuoglu, K., Silver, D., Rusu, A. A., Veness, J., Bellemare, M. G., Graves, A., Riedmiller, M., Fidjeland, A. K., Ostrovski, G., et al. (2015). Human-level control through deep reinforcement learning. *Nature*, 518(7540):529.

Muise, C. J., Belle, V., Felli, P., McIlraith, S. A., Miller, T., Pearce, A. R., and Sonenberg, L. (2015). Planning over multi-agent epistemic states: A classical planning approach. In *AAAI*, pages 3327–3334.

Muller, P., Omidshafiei, S., Rowland, M., Tuyls, K., Perolat, J., Liu, S., Hennes, D., Marris, L., Lanctot, M., Hughes, E., Wang, Z., Lever, G., Heess, N., Graepel, T., and Munos, R. (2020). A generalized training approach for multiagent learning. In *International Conference on Learning Representations*.

Nagel, R. (1995). Unraveling in guessing games: An experimental study. *The American Economic Review*, 85(5):1313–1326.

Nash, J. F. et al. (1950). Equilibrium points in n-person games. *Proceedings of the national academy of sciences*, 36(1):48–49.

Ng, B., Boakye, K., Meyers, C., and Wang, A. (2012). Bayes-adaptive interactive pomdps. In *AAAI*.

Noble, B., Daniel, J. W., et al. (1988). *Applied linear algebra*, volume 3. Prentice-Hall New Jersey.

Oliehoek, F. A., Amato, C., et al. (2016). *A concise introduction to decentralized POMDPs*, volume 1. Springer.

Oliehoek, F. A., Spaan, M. T., and Vlassis, N. (2008). Optimal and approximate q-value functions for decentralized pomdps. *Journal of Artificial Intelligence Research*, 32:289–353.

Omidshafiei, S., Papadimitriou, C., Piliouras, G., Tuyls, K., Rowland, M., Lespiau, J.-B., Czarnecki, W. M., Lanctot, M., Perolat, J., and Munos, R. (2019). α-rank: Multi-agent evaluation by evolution. *Scientific reports*, 9(1):1–29.

Osogami, T. and Raymond, R. (2019). Determinantal reinforcement learning. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 33, pages 4659–4666.

Panait, L. and Luke, S. (2005). Cooperative multi-agent learning: The state of the art. *Autonomous agents and multi-agent systems*, 11(3):387–434.

Panait, L., Luke, S., and Wiegand, R. P. (2006). Biasing coevolutionary search for optimal multiagent behaviors. *IEEE Transactions on Evolutionary Computation*, 10(6):629–645.

Pang, T., Xu, K., Du, C., Chen, N., and Zhu, J. (2019). Improving adversarial robustness via promoting ensemble diversity. In *International Conference on Machine Learning*, pages 4970–4979.

Pangallo, M., Sanders, J., Galla, T., and Farmer, D. (2017). A taxonomy of learning dynamics in 2 x 2 games.

Papoudakis, G. and Albrecht, S. V. (2020). Variational autoencoders for opponent modeling in multi-agent systems.

Papoudakis, G., Christianos, F., and Albrecht, S. V. (2020a). Opponent modelling with local information variational autoencoders.

Papoudakis, G., Christianos, F., Schäfer, L., and Albrecht, S. V. (2020b). Comparative evaluation of multi-agent deep reinforcement learning algorithms.

Pardoe, D. and Stone, P. (2004). Tactex-03: A supply chain management agent. *ACM SIGecom Exchanges*, 4(3):19–28.

Peng, P., Wen, Y., Yang, Y., Yuan, Q., Tang, Z., Long, H., and Wang, J. (2017). Multiagent bidirectionally-coordinated nets: Emergence of human-level coordination in learning to play starcraft combat games. *arXiv preprint arXiv:1703.10069*.

Pfeiffer, B. E. and Foster, D. J. (2013). Hippocampal place-cell sequences depict future paths to remembered goals. *Nature*, 497(7447):74.

Premack, D. and Woodruff, G. (1978). Does the chimpanzee have a theory of mind? *Behavioral and brain sciences*, 1(4):515–526.

Pynadath, D. V. and Marsella, S. C. (2005). Psychsim: Modeling theory of mind with decision-theoretic agents. In *IJCAI*, volume 5, pages 1181–1186.

Rabinowitz, N. C., Perbet, F., Song, H. F., Zhang, C., Eslami, S., and Botvinick, M. (2018). Machine theory of mind. *ICML*.

Raileanu, R., Denton, E., Szlam, A., and Fergus, R. (2018). Modeling others using oneself in multi-agent reinforcement learning. In Dy, J. and Krause, A., editors, *Proceedings of the 35th International Conference on Machine Learning*, volume 80 of *Proceedings of Machine Learning Research*, pages 4257–4266, Stockholmsmässan, Stockholm Sweden. PMLR.

Rashid, T., Samvelyan, M., Schroeder, C., Farquhar, G., Foerster, J., and Whiteson, S. (2018). QMIX: Monotonic value function factorisation for deep multi-agent reinforcement learning. In Dy, J. and Krause, A., editors, *Proceedings of the 35th International Conference on Machine Learning*, volume 80 of *Proceedings of Machine Learning Research*, pages 4295–4304, Stockholmsmässan, Stockholm Sweden. PMLR.

Samvelyan, M., Rashid, T., Schroeder de Witt, C., Farquhar, G., Nardelli, N., Rudner, T. G., Hung, C.-M., Torr, P. H., Foerster, J., and Whiteson, S. (2019). The starcraft multi-agent challenge. In *Proceedings of the 18th International Conference on Autonomous Agents and MultiAgent Systems*, pages 2186–2188. International Foundation for Autonomous Agents and Multiagent Systems.

Schulman, J., Levine, S., Abbeel, P., Jordan, M., and Moritz, P. (2015). Trust region policy optimization. In Bach, F. and Blei, D., editors, *Proceedings of the 32nd International Conference on Machine Learning*, volume 37 of *Proceedings of Machine Learning Research*, pages 1889–1897, Lille, France. PMLR.

Schulman, J., Wolski, F., Dhariwal, P., Radford, A., and Klimov, O. (2017). Proximal policy optimization algorithms. *CoRR*, abs/1707.06347.

Seuken, S. and Zilberstein, S. (2008). Formal models and algorithms for decentralized decision making under uncertainty. *Autonomous Agents and Multi-Agent Systems*, 17(2):190–250.

Shafarevich, I. R. and Remizov, A. O. (2012). *Linear algebra and geometry*. Springer Science & Business Media.

Shalev-Shwartz, S. and Ben-David, S. (2014). *Understanding machine learning: From theory to algorithms*. Cambridge university press.

Shapley, L. S. (1953). Stochastic games. *Proceedings of the national academy of sciences*, 39(10):1095–1100.

Sharghi, A., Borji, A., Li, C., Yang, T., and Gong, B. (2018). Improving sequential determinantal point processes for supervised video summarization. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 517–533.

Shoham, Y., Powers, R., Grenager, T., et al. (2007). If multi-agent learning is the answer, what is the question? *Artificial Intelligence*, 171(7):365–377.

Shum, M., Kleiman-Weiner, M., Littman, M. L., and Tenenbaum, J. B. (2019). Theory of minds: Understanding behavior in groups through inverse planning. *AAAI*.

Simon, H. A. (1972). Theories of bounded rationality. *Decision and organization*, 1(1):161–176.

Singh, S., Kearns, M., and Mansour, Y. (2000). Nash convergence of gradient dynamics in general-sum games. In *Proceedings of the Sixteenth conference on Uncertainty in artificial intelligence*, pages 541–548. Morgan Kaufmann Publishers Inc.

Son, K., Kim, D., Kang, W. J., Hostallero, D. E., and Yi, Y. (2019). QTRAN: Learning to factorize with transformation for cooperative multi-agent reinforcement learning. In Chaudhuri, K. and Salakhutdinov, R., editors, *Proceedings of the 36th International Conference on Machine Learning*, volume 97 of *Proceedings* *of Machine Learning Research*, pages 5887–5896, Long Beach, California, USA. PMLR.

Sondik, E. J. (1971). The optimal control of partially observable markov processes. Technical report, STANFORD UNIV CALIF STANFORD ELECTRONICS LABS.

Sonu, E. and Doshi, P. (2015). Scalable solutions of interactive pomdps using generalized and bounded policy iteration. *Autonomous Agents and Multi-Agent Systems*, 29(3):455–494.

Stahl, D. O. (1993). Evolution of smartn players. *Games and Economic Behavior*, 5(4):604–617.

Stahl II, D. O. and Wilson, P. W. (1994). Experimental evidence on players' models of other players. *Journal of economic behavior & organization*, 25(3):309–327.

Su, J., Adams, S., and Beling, P. A. (2020). Value-decomposition multi-agent actor-critics. *arXiv preprint arXiv:2007.12306*.

Sukhbaatar, S., Fergus, R., et al. (2016). Learning multiagent communication with backpropagation. In *Advances in Neural Information Processing Systems*, pages 2244–2252.

Sunehag, P., Lever, G., Gruslys, A., Czarnecki, W. M., Zambaldi, V., Jaderberg, M., Lanctot, M., Sonnerat, N., Leibo, J. Z., Tuyls, K., and Graepel, T. (2018). Value-decomposition networks for cooperative multi-agent learning based on team reward. In *Proceedings of the 17th International Conference on Autonomous Agents and MultiAgent Systems*, AAMAS '18, page 2085–2087, Richland, SC. International Foundation for Autonomous Agents and Multiagent Systems.

Sutton, R. S. and Barto, A. G. (2018). *Reinforcement learning: An introduction*. MIT press.

Sutton, R. S., McAllester, D. A., Singh, S. P., and Mansour, Y. (2000). Policy gradient methods for reinforcement learning with function approximation. In *Advances in neural information processing systems*, pages 1057–1063.

Tacchetti, A., Song, H. F., Mediano, P. A. M., Zambaldi, V., Kramár, J., Rabinowitz, N. C., Graepel, T., Botvinick, M., and Battaglia, P. W. (2019). Relational forward models for multi-agent learning. In *International Conference on Learning Representations*.

Tan, M. (1993). Multi-agent reinforcement learning: Independent vs. cooperative agents. In *Proceedings of the tenth international conference on machine learning*, pages 330–337.

Tang, J., Paster, K., , and Abbeel, P. (2018). Equilibrium finding via asymmetric self-play reinforcement learning. *Deep Reinforcement Learning Workshop NeurIPS 2018*.

Taylor, A., Van Scoy, B., and Lessard, L. (2018). Lyapunov functions for first-order methods: Tight automated convergence guarantees. *arXiv preprint arXiv:1803.06073*.

Tesauro, G. (1995). Temporal difference learning and td-gammon. *Communications of the ACM*, 38(3):58–68.

Tian, Z., Wen, Y., Gong, Z., Punakkath, F., Zou, S., and Wang, J. (2019). A regularized opponent model with maximum entropy objective. *IJCAI*.

Tolman, E. C. (1948). Cognitive maps in rats and men. *Psychological review*, 55(4):189.

Tuyls, K., Perolat, J., Lanctot, M., Hughes, E., Everett, R., Leibo, J. Z., Szepesvári, C., and Graepel, T. (2020). Bounds and dynamics for empirical game theoretic analysis. *Autonomous Agents and Multi-Agent Systems*, 34(1):7.

Tuyls, K., Perolat, J., Lanctot, M., Leibo, J. Z., and Graepel, T. (2018). A generalised method for empirical game theoretic analysis. In *Proceedings of the 17th*

International Conference on Autonomous Agents and MultiAgent Systems, pages 77–85. International Foundation for Autonomous Agents and Multiagent Systems.

Tuyls, K. and Stone, P. (2018). Multiagent learning paradigms. In Belardinelli, F. and Argente, E., editors, *Multi-Agent Systems and Agreement Technologies*, volume 10767 of *Lecture Notes in Artificial Intelligence*, pages 3–21. Springer.

Tuyls, K. and Weiss, G. (2012). Multiagent learning: Basics, challenges, and prospects. *Ai Magazine*, 33(3):41.

Vinyals, O., Babuschkin, I., Czarnecki, W. M., Mathieu, M., Dudzik, A., Chung, J., Choi, D. H., Powell, R., Ewalds, T., Georgiev, P., et al. (2019). Grandmaster level in starcraft ii using multi-agent reinforcement learning. *Nature*, 575(7782):350–354.

Von Der Osten, F. B., Kirley, M., and Miller, T. (2017). The minds of many: opponent modelling in a stochastic game. In *IJCAI*, pages 3845–3851.

Wang, D. and Liu, Q. (2016). Learning to draw samples: With application to amortized mle for generative adversarial learning. *arXiv preprint arXiv:1611.01722*.

Wang, Y., Han, B., Wang, T., Dong, H., and Zhang, C. (2020). Off-policy multi-agent decomposed policy gradients. *arXiv preprint arXiv:2007.12322*.

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

Wei, E. and Luke, S. (2016). Lenient learning in independent-learner stochastic cooperative games. *The Journal of Machine Learning Research*, 17(1):2914–2955.

Wei, E., Wicke, D., Freelan, D., and Luke, S. (2018). Multiagent soft q-learning. In *2018 AAAI Spring Symposium Series*.

Wellman, M. P. (2006). Methods for empirical game-theoretic analysis. In *AAAI*, pages 1552–1556.

Wen, Y., Yang, Y., Luo, R., Wang, J., and Pan, W. (2019). Probabilistic recursive reasoning for multi-agent reinforcement learning. *ICLR*.

Williams, R. J. (1992). Simple statistical gradient-following algorithms for connectionist reinforcement learning. In *Reinforcement Learning*, pages 5–32. Springer.

Xia, Y., Qin, T., Chen, W., Bian, J., Yu, N., and Liu, T.-Y. (2017). Dual supervised learning. In *Proceedings of the 34th International Conference on Machine Learning - Volume 70*, ICML'17, page 3789–3798. JMLR.org.

Yang, Y., Luo, R., Li, M., Zhou, M., Zhang, W., and Wang, J. (2018a). Mean field multi-agent reinforcement learning. In Dy, J. and Krause, A., editors, *Proceedings of the 35th International Conference on Machine Learning*, volume 80 of *Proceedings of Machine Learning Research*, pages 5571–5580, Stockholmsmassan, Stockholm Sweden. PMLR.

Yang, Y., Tutunov, R., Sakulwongtana, P., Ammar, H. B., and Wang, J. (2019). Alpha-alpha-rank: Scalable multi-agent evaluation through evolution. *AAMAS 2020.*

Yang, Y., Yu, L., Bai, Y., Wen, Y., Zhang, W., and Wang, J. (2018b). A study of ai population dynamics with million-agent reinforcement learning. In *17th AAMAS*, pages 2133–2135. International Foundation for Autonomous Agents and Multiagent Systems.

Zhang, C. and Lesser, V. (2010). Multi-agent learning with policy prediction. In *AAAI*.

Zhou, M., Chen, Y., Wen, Y., Yang, Y., Su, Y., Zhang, W., Zhang, D., and Wang, J. (2019). Factorized q-learning for large-scale multi-agent systems. In *Proceedings of the First International Conference on Distributed Artificial Intelligence*, pages 1–7.

Ziebart, B. D. (2010). *Modeling Purposeful Adaptive Behavior with the Principle of Maximum Causal Entropy*. PhD thesis, Carnegie Mellon University, Pittsburgh, PA, USA. AAI3438449.