

Many-Agent Reinforcement Learning

Yaodong Yang

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

I, Yaodong Yang, confirm that the work presented in this thesis is my own.
Where information has been derived from other sources, I confirm that this has
been indicated in the work.

Abstract

Multi-agent reinforcement learning (RL) solves the problem of how each agent should behave optimally in a stochastic environment in which multiple agents are learning simultaneously. It is an interdisciplinary domain with a long history that lies in the joint area of psychology, control theory, game theory, reinforcement learning, and deep learning. Following the remarkable success of the AlphaGO series in single-agent RL, 2019 was a booming year that witnessed significant advances in multi-agent RL techniques; impressive breakthroughs have been made on developing AIs that outperform humans on many challenging tasks, especially multi-player video games. Nonetheless, one of the key challenges of multi-agent RL techniques is the scalability; it is still non-trivial to design efficient learning algorithms that can solve tasks including far more than two agents ($N \gg 2$), which I name by *many-agent reinforcement learning* (MARL¹) problems.

In this thesis, I contribute to tackling MARL problems from four aspects. Firstly, I offer a self-contained overview of multi-agent RL techniques from a game-theoretical perspective. This overview fills the research gap that most of the existing work either fails to cover the recent advances since 2010 or does not pay adequate attention to game theory, which I believe is the cornerstone to solving many-agent learning problems. Secondly, I develop a tractable policy evaluation algorithm – α^α -Rank – in many-agent systems. The critical advantage of α^α -Rank is that it can compute the solution concept of α -Rank tractably in multi-player general-sum games with no need to store the entire pay-off matrix. This is in contrast to classic solution concepts such as Nash equilibrium which is known to be *PPAD*-hard in

¹I use the word of “MARL” to denote multi-agent reinforcement learning with a particular focus on the cases of many agents; otherwise, it is denoted as “Multi-Agent RL” by default.

even two-player cases. α^α -Rank allows us, for the first time, to practically conduct large-scale multi-agent evaluations. Thirdly, I introduce a scalable policy learning algorithm – mean-field MARL – in many-agent systems. The mean-field MARL method takes advantage of the mean-field approximation from physics, and it is the first provably convergent algorithm that tries to break the curse of dimensionality for MARL tasks. With the proposed algorithm, I report the first result of solving the Ising model and multi-agent battle games through a MARL approach. Fourthly, I investigate the many-agent learning problem in open-ended meta-games (i.e., the game of a game in the policy space). Specifically, I focus on modelling the behavioural diversity in meta-games, and developing algorithms that guarantee to enlarge diversity during training. The proposed metric based on determinantal point processes serves as the first mathematically rigorous definition for diversity. Importantly, the diversity-aware learning algorithms beat the existing state-of-the-art game solvers in terms of exploitability by a large margin.

On top of the algorithmic developments, I also contribute two real-world applications of MARL techniques. Specifically, I demonstrate the great potential of applying MARL to study the emergent population dynamics in nature, and model diverse and realistic interactions in autonomous driving. Both applications embody the prospect that MARL techniques could achieve huge impacts in the real physical world, outside of purely video games.

Impact Statement

This thesis focuses on designing algorithms to solve many-agent reinforcement learning (MARL) problems. MARL provides a general modelling framework that essentially tries to answer how each agent should behave optimally in a stochastic environment when many agents are involved. This thesis contains three algorithmic innovations (Chapters 4,5,6). The proposed algorithms draw our attention to the scalable policy evaluation problem and the scalable policy learning problem under the scenarios of far more than two agents (i.e., the many-agent cases). From the scientific point of view, the developed methods can easily see numerous impactful applications in the real world since the multi-agent interaction is one common aspect of many problems in nature. I have listed two MARL applications in this thesis: one is to understand the emergent population dynamics from intelligent agents (Chapter 3), and the other is to model diverse and realistic interactions for autonomous driving (Chapter 7). Both applications demonstrate the great potentials of applying MARL techniques to real-world problems. Other impactful applications in the industry include applying mean-field MARL methods to tackle Uber (China) order dispatching problem, which was reported by [MIT Technology Review](#), and designing game AIs that can outperform human players in sophisticated environments, such as Dota2, [StarCraft II](#), and Chess. Overall, understanding the mechanism of multi-agent interactions between AIs and between humans and AIs present long-term research challenges. They need remarkable efforts from different domains, such as macro-biology, psychology, economics, and machine learning. This thesis could achieve more impact by benefiting all the relevant domains.

Acknowledgements

This is probably the most challenging chapter to write throughout the whole thesis manuscript. The reason is that there is an endless list of people that I want to say thank you to. Therefore, I sincerely apologise, in advance, to those people whose names are mistakenly missed in this chapter but has generously offered me kind helps during my PhD study.

Among all the people I would like to acknowledge, Prof. Jun Wang, my primary supervisor, is the first and foremost person I want to express my gratitude. Jun is undoubtedly a remarkable supervisor, and I am pleased that I was fortunate enough to be introduced to him five years ago (through Dr. Weinan Zhang) and became his PhD student. Jun's philosophy and perseverance in finding principled research questions and solutions always motivate me to think about what should be the right question to ask and the principled answer to seek. This philosophy of doing research often awakened me especially when I am obsessed with some trivial ideas that could lead to possible good publications but not necessarily an excellent, memorable work, a work that is truly contributive work to the community. Admittedly, doing only principled research needs extra patience, the benefit is often not immediate, and the process is always far more frustrating than cutting corners or finding quick answers. However, I did realise its benefit and stick to his philosophy after I enter into senior years of my PhD study. Since I will become an assistant professor after graduation, I sincerely believe that Jun's philosophy of doing only principled research will become an extremely valuable treasure to me; it will accompany me throughout my academic career and guide me to become a great supervisor like him. Beyond doing research, Jun is also a nice person who has a

very understanding personality. In the past five years, it is truly a great pleasure to have almost every discussion, every reading group, every paper collaboration with him. All in all, without his rigorous guidance and kind personality, I would never image I could publish more than 20 research papers during my PhD study. To Jun, I sincerely hope he would feel proud and also find the time he has spent with me worthwhile and enjoyable.

The second persons I want to thank the most are my family members, my wife Yi Qu in particular. I knew Yi for more than 15 years, and luckily, we got married at the beginning of my PhD study. Yi's companion has always been my spiritual anchor, and frankly, the unconditional love and encouragement I received from Yi are imperative to my accomplishments in the PhD study. Without her, I would never imagine I could survive from the most challenging time of my PhD journey.

During my PhD study, I was fortunate enough to work as a research scientist in AIG and Huawei UK companies. I am incredibly grateful to my industrial supervisors Dr. Reza Khorshidi, Dr. Yao Jun, and Dr. Haitham Bou Ammar, along with my colleagues Dr. Rasul Tutunov, Dr. David Mguni, and many others who worked in the same lab. I want to thank them for their generous support and undoubted confidence in me. I would particularly like to thank Dr. Haitham Bou Ammar and Dr. Rasul Tutunov for their consistently critical, yet insightful, comments on the contemporary work in the multi-agent reinforcement learning domain, which drive me to think deeper and push my research skill to the next level.

Fourthly, I would like to thank Prof. John Shawe-Taylor and Prof. Thore Graepel for acknowledging my work during the first and second PhD transfer viva at University College London. In particular, the discussions with Prof. Thore and his team's work have enlightened many of my later work. I also would like to thank Dr. Sebastian Stein and Dr. Laura Toni for serving as my PhD viva examiners.

Moreover, I would like to express my wholehearted appreciations to my supportive collaborators: Dr. Ying Wen, Dr. Weinan Zhang, Ming Zhou, Dr. Haifeng Zhang, Dr. Jun Luo, Prof. Matthew Taylor, Prof. Xiaotie Deng, Rui Luo, Zheng Tian, Minne Li, Yixin Wu. Meanwhile, I also would like to thank Dr. Ao Li at

USTC, Dr. James Rosindell and Dr. Daniel Reuman at Imperial College London, Prof. Johnnie Johnson and Prof. Ming-Chen Sung at the University of Southampton for guiding and cultivating me in doing high-quality academic research at the early times before I joined UCL.

Lastly, I would like to thank Prof. Song-Chun Zhu and Prof. Luc Moreau for their appreciations on my research work, and their trust in offering me the assistant professor roles at Peking University and King's College London respectively. I am extremely grateful to their recognitions, which means a lot to me, especially under the hard time of COVID-19 pandemics.

PhD study is a tough even mentally torturing journey; doing academic research that naturally involves many types of uncertainty is not always sweet. However, it is the love and unreserved support that all the people mentioned above have generously offered that mitigate the bitterness of this journey, and because of this reason, when you reach the destination and look back upon the research contributions you have made, you still feel it is very much worth the time, accompanied by these people. I am extremely obliged to every one who has offered me helps throughout my PhD study!

Contents

1	Introduction	24
1.1	Background of RL and Multi-Agent RL	24
1.1.1	A Short History of RL	26
1.1.2	2019: A Booming Year for Multi-Agent RL	28
1.2	Single-Agent RL	30
1.2.1	Problem Formulation: Markov Decision Process (MDP) . .	31
1.2.2	Justification of Reward Maximisation	32
1.2.3	Solving Markov Decision Processes	33
1.3	Multi-Agent RL	36
1.3.1	Problem Formulation: Stochastic Game	37
1.3.2	Solving Stochastic Games	38
1.3.3	Problem Formulation: Extensive-Form Game	50
1.3.4	Solving Extensive-Form Games	58
1.4	Grand Challenges of MARL	62
1.4.1	The Combinatorial Complexity	62
1.4.2	The Multi-Dimensional Learning Objectives	63
1.4.3	The Non-Stationarity Issue	65
1.5	Thesis Structure and Contributions	66
1.6	Related Publications	69
2	Literature Overview	73
2.1	A Survey of Surveys on Multi-Agent RL	73
2.1.1	Taxonomy of Multi-Agent RL Algorithms	74

	<i>Contents</i>	10
2.1.2 A Survey of Surveys	77	
2.2 Learning in Identical-Interest Games	81	
2.2.1 Stochastic Team Games	81	
2.2.2 Dec-POMDPs	87	
2.2.3 Networked Multi-Agent MDPs	88	
2.2.4 Stochastic Potential Games	91	
2.3 Learning in Zero-Sum Games	92	
2.3.1 Discrete State-Action Games	93	
2.3.2 Continuous State-Action Games	95	
2.3.3 Extensive-Form Games	98	
2.3.4 Online Markov Decision Processes	107	
2.3.5 Turn-Based Stochastic Games	111	
2.3.6 Open-Ended Meta-Games	112	
2.4 Learning in General-Sum Games	116	
2.4.1 Solutions by Mathematical Programming	116	
2.4.2 Solutions by Value-Based Methods	118	
2.4.3 Solutions by Two-Timescale Analysis	119	
2.4.4 Solutions by Policy-Based Methods	119	
2.5 Learning in Games when $N \rightarrow +\infty$	122	
2.5.1 Non-cooperative Mean-Field Game	124	
2.5.2 Cooperative Mean-Field Control	127	
2.5.3 Mean-Field MARL	130	
3 Emergent Population Dynamics from Million-Agent RL	134	
3.1 Background and Motivation	135	
3.2 Design of the Predator-Prey World	138	
3.2.1 The <i>Axioms</i> Observed in Nature	138	
3.2.2 Realisation of the <i>Axioms</i>	141	
3.3 The AI Population Driven by MARL	143	
3.3.1 The Implementation of Intelligent Agents	144	
3.4 Experiments and Findings	146	

3.4.1	The Emergent Population Dynamics	147
3.4.2	The Emergent Grouping Behaviours	150
3.4.3	Connection to Self-Organising Theory	152
3.5	Chapter Summary	153
4	Many-Agent Policy Evaluation: α^α-Rank	154
4.1	Background and Motivation	155
4.2	Preliminary: α -Rank	156
4.3	Reconsidering α -Rank's Complexity	160
4.3.1	Conjecture: Computing α -Rank is <i>NP</i> -Hard	161
4.3.2	Approximation Solutions to α -Rank	162
4.3.3	On the Definition of <i>Agents</i>	166
4.3.4	Dollars Spent: A Non-Refutable Metric	168
4.4	α^α -Rank: A Scalable Solution to α -Rank	170
4.4.1	Solution by Stochastic Optimisation	170
4.4.2	Time and Memory Complexity of α^α -Rank	172
4.4.3	Efficient Exploration via Oracles	175
4.5	Experiments and Results	176
4.5.1	Implementations of Sparse Vectors	177
4.5.2	Hyper-Parameter Settings	178
4.5.3	Analysis on Normal-Form Games	178
4.5.4	Analysis on Random Matrices	179
4.5.5	Autonomous Driving on Highway	180
4.5.6	Ising Model	182
4.6	Chapter Appendix	183
4.7	Chapter Summary	184
5	Many-Agent Policy Learning: A Mean-Field Approach	185
5.1	Background and Motivation	186
5.2	Related Work	187
5.3	Preliminary and Notations	189

Contents 12

5.3.1	Stochastic Game	189
5.3.2	Nash Q-Learning	190
5.4	Mean-Field MARL	191
5.4.1	Mean-Field Approximation in RL	192
5.4.2	Mean-Field Q-Learning and Actor-Critic	197
5.5	Convergence Analysis of MF-Q	199
5.5.1	MF-Q with Tabular Form Representation	199
5.5.2	MF-Q with Linear Function Approximation	205
5.5.3	MF-Q is Rational	207
5.6	Experiments and Results	208
5.6.1	Gaussian Squeeze	208
5.6.2	MARL Solutions to Ising Models	211
5.6.3	Many-Agent Battle Games	214
5.7	Chapter Summary	216
6	Many-Agent Learning in Open-Ended Meta-Games	218
6.1	Background and Motivation	220
6.2	Related Work	222
6.3	Preliminary and Notations	224
6.3.1	Solution Concepts of Games	224
6.3.2	Formulation of Open-Ended Meta-Game	226
6.3.3	Solving Open-Ended Meta-Games	227
6.3.4	Effective Diversity	229
6.4	A New Measurement for Diversity	231
6.4.1	Determinantal Point Process	231
6.4.2	Expected Cardinality	232
6.4.3	Expected Cardinality vs. Matrix Rank	234
6.4.4	Expected Cardinality vs. Effective Diversity	237
6.5	Diversity Aware Learning Algorithms	239
6.5.1	Diverse Fictitious Play	240
6.5.2	Diverse Policy Space Response Oracle	244

<i>Contents</i>	13
6.5.3 Implementation of Oracles.	245
6.5.4 Diverse Oracle for α -Rank	247
6.6 Experiments and Results	249
6.6.1 Random Games of Skill	251
6.6.2 2D-Rock Paper Scissors	252
6.6.3 Colonel Blotto	253
6.6.4 Diverse α -PSRO	253
6.7 Chapter Summary	254
7 Modelling Diverse Interactions in Autonomous Driving	255
7.1 Background of Autonomous Driving	256
7.2 Challenges in Modelling Diverse Interactions	257
7.3 When MARL Meets Autonomous Driving	259
7.4 SMARTS: A Bespoke AD Testbed for MARL	261
7.4.1 Key Features	263
7.4.2 Support for RL Study	265
7.4.3 MARL Benchmarking Suite	266
7.5 Blue Sky Idea: Diverse Auto-Curriculum is the Key to AD	269
7.5.1 The Necessity of A Diverse Auto-Curriculum	271
7.5.2 Open Challenges of Applying Auto-Curricula in AD	273
7.6 Chapter Summary	276
8 Conclusions and Future Work	278
Bibliography	282

List of Figures

1.1	Modern AI applications are being transformed from pure feature recognition (for example, detecting a cat in an image) to decision making (driving through a traffic intersection safely), where interaction among multiple agents inevitably occurs. As a result, each agent has to behave strategically. Furthermore, the problem becomes more challenging because current decisions influence future outcomes.	26
1.2	The success of the AlphaGo series marks the maturation of the single-agent decision-making process. The year 2019 was a booming year for MARL techniques; remarkable progress was achieved in solving immensely challenging multi-player real-strategy video games and multi-player incomplete-information poker games. . . .	28
1.3	Diagram of a single-agent MDP (left) and a multi-agent MDP (right).	30
1.4	A snapshot of stochastic time in the intersection example. The scenario is abstracted such that there are two cars, with each car taking one of two possible actions: to yield or to rush. The outcome of each joint action pair is represented by a normal-form game, with the reward value for the row player denoted in red and that for the column player denoted in black. The Nash equilibria (NE) of this game are (rush, yield) and (yield, rush). If both cars maximise their own reward selfishly without considering the others, they will end up in an accident.	37

1.5	The landscape of different complexity classes. Relevant examples are 1) solving the NE in a two-player zero-sum game, P -complete (Neumann, 1928), 2) solving the NE in a general-sum game, $PPAD$ -hard (Daskalakis et al., 2009), 3) checking the uniqueness of the NE, NP -hard (Conitzer and Sandholm, 2002), 4) checking whether a pure-strategy NE exists in a stochastic game, $PSPACE$ -hard (Conitzer and Sandholm, 2008), and 5) solving Dec-POMDP, $NEXPTIME$ -hard (Bernstein et al., 2002).	43
1.6	Venn diagram of different types of games in the context of POSGs. The intersection of SG and Dec-POMDP is the team game. In the upper-half SG, we have $MDP \subset$ team games \subset potential games \subset identical-interest games \subset SGs, and zero-sum games \subset SGs. In the bottom-half Dec-POMDP, we have $MDP \subset$ team games \subset Dec-MDP \subset Dec-POMDPs, and $MDP \subset POMDP \subset$ Dec-POMDP. We refer to Sections (1.3.2.4 & 1.3.2.5) for detailed definitions of these games.	45
1.7	Game tree of two-player Kuhn poker. Each node (i.e., circles, squares and rectangles) represents the choice of one player, each edge represents a possible action, and the leaves (i.e., diamond) represent final outcomes over which each player has a reward function (only player one's reward is shown in the graph since Kuhn poker is a zero-sum game). Each player can observe only their own card; for example, when player one holds a Jack, it cannot tell whether player two is holding a Queen or a King, so the choice nodes of player one in each of the two scenarios stay within the same information set.	52
1.8	The scope of the research in this thesis consists of three pillars. Deep learning serves as a powerful function approximation tool for the learning process. Game theory provides an effective approach to describe learning outcomes. RL offers a valid approach to describe agents' incentives in multi-agent systems.	66

1.9	The structure of following chapters in this thesis, associated with the three listed challenges in Chapter 1.4 that each chapter tries to address.	67
2.1	Common learning paradigms of MARL algorithms. (1) Independent learners with shared policy. (2) Independent learners with independent policies (i.e., denoted by the difference in wheels). (3) Independent learners with shared policy within a group. (4) One central controller controls all agents: agents can exchange information with any other agents at any time. (5) Centralised training with decentralised execution (CTDE): only during training, agents can exchange information with others; during execution, they act independently. (6) Decentralised training with networked agents: during training, agents can exchange information with their neighbours in the network; during execution, they act independently. . .	76
2.2	Graphical model of the <i>level-k</i> reasoning model (Wen et al., 2019b). The red part is the equivalent graphical model for the multi-agent learning problem. The blue part corresponds to the recursive reasoning steps. Subscript a_* stands for the level of thinking, not the time step. The opponent policies are approximated by ρ^{-i} . The omitted <i>level-0</i> model considers opponents that are fully randomised. Agent i rolls out the recursive reasoning about opponents in its mind (blue area). In the recursion, agents with higher-level beliefs take the best response to the lower-level agents. The higher-level models conduct all the computations that the lower-level models have done, e.g., the <i>level-2</i> model contains the <i>level-1</i> model by integrating out $\pi_0^i(a^i s)$	84
2.3	Relations of mean-field learning algorithms in games with large N . .	123

3.1	Illustration of the predator-prey world. In the 2D world, there exist preys, predators, and obstacles. Predators hunt the prey so as to survive from starvation. The reward is the proportion of prey that the predator obtains. Each predator has its own health bar and limited eyesight view. Predators can form a group to hunt the prey so that the chance of capturing can increase, but this also means that the captured prey will be shared among all group members. When there are multiple groups targeting the same prey, the largest group within the capture radius will win. In this example, predators $\{2, 3, 4\}$ form a group and win the prey over the group $\{5, 6\}$. Predator 5 soon dies due to starvation.	141
3.2	Million-agent Q-learning System in the Predator-prey World.	143
3.3	Population dynamics in both the time space (1^{st} row) and the phase space (2^{nd} row). The orange circles denote the theoretical solutions to the <i>Lotka-Volterra</i> equation, with the red spot as the equilibrium. The green-blue circles denote the simulation results. a): The simulated birth rate of preys is 0.006. Fitted LV model: $\alpha = 0.0067, \beta = 3.75 \times 10^{-7}, \delta = 6.11 \times 10^{-7}, \gamma = 0.001$. b): The simulated birth rate of preys is 0.01. Fitted LV model: $\alpha = 0.0086, \beta = 3.57 \times 10^{-7}, \delta = 9.47 \times 10^{-7}, \gamma = 0.0012$, where α in the LV model represents the birth rate.	148
3.4	Population dynamics in the time space and the phase space. A new type of prey (green line) is introduced, which can be captured by a single agent. The AI population shows ordered dynamics in the 3-D phase space.	149
3.5	Population dynamics with the learning function of AI population disabled. The simulation follows the same setting as Figure 3.3(b). No ordered dynamics are found any more.	149

3.6	a) Grouping proportion in the predator-prey world where two kinds of preys are fed alternatively. \uparrow points out the time step that preys are fed. It indicates that when the number of the prey sheep (that requires group hunting) increases, the proportion of groups in the AI population increases, and adapting to grouping becomes a collective behaviour. Vice versa for the case when the prey rabbit are fed. b) The same experiment on a two-million AI population.	151
4.1	Example of α -Rank evaluation on $N = 3$ players (star, triangle, circle) each with $ s = 3$ strategies (denoted by the colours) and $m = 5$ copies. a) Each population obtains a fitness value \mathcal{P}_i depending on the strategies chosen, b) one mutation strategy (red star) occurs, and c) the population either selects the original strategy, or being fixated by the mutation strategy.	157
4.2	Money cost of constructing the transition matrix \mathbf{T} in computing α -Rank (line 3 in Algorithm 3). Note that one trillion dollar is the world's total hardware budget (Department, 2020). The projected contours show that due to the exponentially-growing size of α -Rank's "input", under reasonable budget, α -Rank is unable to handle more than ten agents.	167
4.3	Sparse vector implementation in α^α -Rank with an example on addition.	177
4.4	Ranking intensity sweep on (a) Battle of Sexes (b) Biased RPS (c) Prisoner's Dilemma.	179
4.5	Comparisons of time and memory complexities on varying sizes of random matrices.	180
4.6	Large-scale multi-agent evaluations. (a) Convergence of the optimal joint-strategy profile in self-driving simulation. (b) Status of the Ising-model equilibrium measured by $\xi = \frac{ N_\uparrow - N_\downarrow }{ N }$. (c) Change of the top-rank profile from α^α -Oracle.	181

4.7	Cost breakup for Table 4.2. The full spread sheet can also be found at https://docs.google.com/spreadsheets/d/1XJM4C9WWRJTBrIzSuBkpAnleoo3B9TanMC17nWVIn_s .	183
5.1	Mean-field approximation. Each agent is represented as a node in the grid, which is only affected by the mean effect from its neighbours (the blue area). Many-agent interactions are effectively converted into two-agent interactions.	195
5.2	MF- Q iterations on a 3×3 stateless toy example. The goal is to coordinate the agents to an agreed direction. Each agent has two choices of actions: <i>up</i> \uparrow or <i>down</i> \downarrow . The reward of each agent's staying in the same direction as its $[0, 1, 2, 3, 4]$ neighbours are $[-2.0, -1.0, 0.0, 1.0, 2.0]$, respectively. The neighbours are specified by the four directions on the grid with cyclic structure on all directions, <i>e.g.</i> the first row and the third row are adjacent. The reward for the highlighted agent j on the bottom left at time $t + 1$ is 2.0, as all neighbouring agents stay down in the same time. We listed the Q-tables for agent j at three time steps where \bar{a}^j is the percentage of neighbouring ups. Following Eq. (5.11), we have $Q_{t+1}^j(\uparrow, \bar{a}^j = 0) = Q_t^j(\uparrow, \bar{a}^j = 0) + \alpha[r^j - Q_t^j(\uparrow, \bar{a}^j = 0)] = 0.82 + 0.1 \times (2.0 - 0.82) = 0.93$. The rightmost plot shows the convergent scenario where the Q -value of staying down is 2.0, which is the largest reward in the environment.	200
5.3	Learning with N agents in the GS environment with $\mu = 400$ and $\sigma = 200$. With the increasing number of agents, MF-Q methods demonstrate remarkable performance improvement.	208
5.4	The <i>order parameter</i> at equilibrium <i>v.s.</i> temperature in the Ising model with 20×20 grid.	210
5.5	Training performance of MF- Q in the Ising model with 20×20 grid.	210
5.6	The spins of the Ising model at equilibrium under different temperatures.	213

5.7	The battle game: 64 v.s. 64.	215
5.8	Performance comparisons in the battle game. “IL” stands for the independent learning method.	215
6.1	Game-DPP. The squared volume of the grey cube equals to $\det(\mathcal{L}_{\{S_1^i, S_2^i, S_3^i\}})$. Since S_2^i, S_3^i share similar payoff vectors, this leads to a smaller yellow area, and thus the probability of these two strategies co-occurring is low. The diversity (expected cardinality) of the population $\{S_1^i\}, \{S_1^i, S_2^i\}, \{S_1^i, S_2^i, S_3^i\}$ are 0, 1, 1.21 respectively. . .	232
6.2	Results on Games of Skill with payoff matrices sizes ranging from 100×100 to 1500×1500 . For each subplot, the upper half shows the exploitability versus the population size (i.e., the PSRO iteration), and lower half shows the diversity in terms of expected cardinality.	251
6.3	2D-Rock Paper Scissors. a): Exploration trajectories during training. b): Performance vs. Diversity comparisons.	252
6.4	Results of a) exploitability on the Blotto Game , b) PCS-Score against α -PSRO on NFGs	253
7.1	The double merge scenario in autonomous driving.	260
7.2	Bootstrapping interaction realism and diversity.	262
7.3	The SMARTS platform proposed by Zhou et al. (2020). (a): Multiple ego agents in training (red vehicles) can run simultaneously within a shared SMARTS instance. Each agent runs in a separate process and can also run remotely. (b): Bubbles are regions in which social vehicles (yellow) may be controlled by agents from the Social Agent Zoo and interact meaningfully with ego vehicles (red). Inside blue Zone, social vehicles are controlled by the traffic provider. Inside red Zone, they are controlled by agents from the Social Agent Zoo by MARL algorithms. Blue areas are transition zones.	263

7.4	Scenarios of driving interaction specifiable in SMARTS	265
7.5	Results reported in Zhou et al. (2020) on four behaviour metrics in the MARL benchmarking suite of SMARTS. The larger the coverage, the more desirable the behaviour. The wider scattered the curves, the more diverse the behaviours. The upper three and bottom three plots differ in whether there are social vehicles on the road provided controlled by a traffic provider such as SUMO.	269

List of Tables

2.1	Common assumptions on the level of local knowledge made by MARL algorithms.	75
2.2	Summary of the five agendas for multi-agent learning research Shoham et al. (2007).	78
2.3	Variations of Different Meta-Game Solvers	113
3.1	The axioms adopted in the simulated predator-prey world (also see Sumpter (2006)[Section 5]).	140
3.2	Parameters settings for the predator-prey world. Code is released at https://github.com/geek-ai/1m-agents	147
4.1	Time and space complexity comparison given N (number of agents) \times k (number of strategies) as inputs.	162
4.2	Cost of getting the payoff table $\mathcal{P}_{\text{joint}}$ (line 2 in Algorithm 3) for the experiments conducted in Omidshafiei et al. (2019a). We list the numbers by the cost of running one joint-strategy profile \times the number of joint-strategy profiles considered. A detailed breakup can be found in Figure 4.7.	168
4.3	Hyper-parameter settings for the experiments. m : Population size, α : Ranking intensity, η : Learning rate	178
4.4	Reward settings in Self-driving Car Simulation.	180
6.1	Variations of Different Meta-Game Solvers	228
6.2	Hyper-parameter settings for our diverse-PSRO method vs. other baselines on four experiments.	250

Chapter 1

Introduction

This chapter offers a self-contained introduction of multi-agent reinforcement learning (RL). Starting from a motivating example and a brief overview of RL’s history, I then show the precise problem formulations, essential solutions, and existing challenges of multi-agent RL. Specifically, I present the multi-agent RL formulations through two common frameworks: stochastic games and extensive-form games, along with different variations of games that can be addressed. This chapter aims to enable readers, even those with minimal related background, to grasp the key ideas in multi-agent RL research and the fundamental challenges of solving MARL problems. Finally, I end this chapter by presenting chapter arrangements, highlighting the main contributions of this thesis, and listing related publications.

1.1 Background of RL and Multi-Agent RL

Machine learning can be considered as the process of converting data into knowledge ([Shalev-Shwartz and Ben-David, 2014](#)). The input of a learning algorithm is training data (for example, images containing cats), and the output is some knowledge (for example, rules about how to detect cats in an image). This knowledge is usually represented as a computer program that can perform certain task(s) (for example, an automatic cat detector). In the past decade, considerable progress has been made by means of a special kind of machine learning technique: deep learning ([LeCun et al., 2015](#)). One of the critical embodiments of deep learning is different kinds of deep neural networks (DNNs) ([Schmidhuber, 2015](#)) that can find

disentangled representations (Bengio, 2009) in high-dimensional data, which allows the software to train itself to perform new tasks rather than merely relying on the programmer for designing hand-crafted rules. An uncountable number of breakthroughs in real-world AI applications have been achieved through the usage of DNNs, with the domains of computer vision (Krizhevsky et al., 2012) and natural language processing (Brown et al., 2020; Devlin et al., 2018) being the greatest beneficiaries.

In addition to feature recognition from existing data, modern AI applications often require computer programs to make decisions based on acquired knowledge (see Figure 1.1). To illustrate the key components of decision making, let us consider the real-world example of controlling a car to drive safely through an intersection. At each time step, a robot car can move by steering, accelerating and braking. The goal is to safely exit the intersection and reach the destination (with possible decisions of going straight or turning left/right into another lane). Therefore, in addition to being able to detect objects, such as traffic lights, lane markings, and other cars (by converting data to knowledge), we aim to find a steering policy that can control the car to make a sequence of manoeuvres to achieve the goal (making decisions based on the knowledge gained). In a decision-making setting such as this, two additional challenges arise:

1. First, during the decision-making process, at each time step, the robot car should consider not only the immediate value of its current action but also the consequences of its current action in the future. For example, in the case of driving through an intersection, it would be detrimental to have a policy that chooses to steer in a “safe” direction at the beginning of the process if it would eventually lead to a car crash later on.
2. Second, to make each decision correctly and safely, the car must also consider other cars’ behaviour and act accordingly. Human drivers, for example, often predict in advance other cars’ movements and then take strategic moves in response (like giving way to an oncoming car or accelerating to merge into another lane).

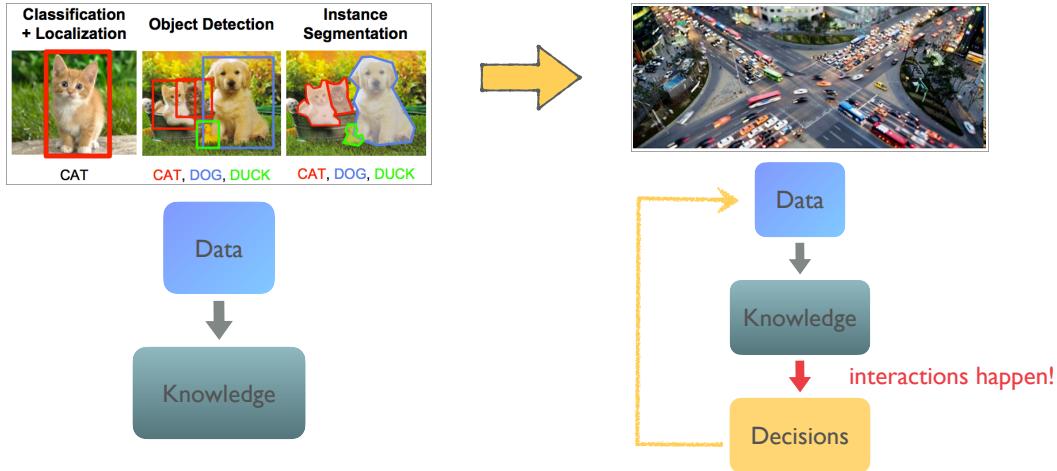


Figure 1.1: Modern AI applications are being transformed from pure feature recognition (for example, detecting a cat in an image) to decision making (driving through a traffic intersection safely), where interaction among multiple agents inevitably occurs. As a result, each agent has to behave strategically. Furthermore, the problem becomes more challenging because current decisions influence future outcomes.

The need for an adaptive decision-making framework, together with the complexity of addressing multiple interacting learners, has led to the development of multi-agent RL. Multi-agent RL tackles the sequential decision-making problem of having multiple intelligent agents that operate in a shared stochastic environment, each of which targets to maximise its long-term reward through interacting with the environment and other agents. Multi-agent RL is built on the knowledge of both multi-agent systems (MAS) and RL. In the next section, we provide a brief overview of (single-agent) RL and the research developments in recent decades.

1.1.1 A Short History of RL

RL is a sub-field of machine learning, where agents learn how to behave optimally based on a trial-and-error procedure during their interaction with the environment. Unlike supervised learning, which takes labelled data as the input (for example, an image labelled with cats), RL is goal-oriented: it constructs a learning model that learns to achieve the optimal long-term goal by improvement through trial and error, with the learner having no labelled data to obtain knowledge from. The word “reinforcement” refers to the learning mechanism since the actions that lead to sat-

isfactory outcomes are reinforced in the learner’s set of behaviours.

Historically, the RL mechanism was originally developed based on studying cats’ behaviour in a puzzle box (Thorndike, 1898). Minsky (1954) first proposed the computational model of RL in his Ph.D. thesis and named his resulting analog machine the *stochastic neural-analog reinforcement calculator*. Several years later, he first suggested the connection between dynamic programming (Bellman, 1952) and RL (Minsky, 1961). In 1972, Klop (1972) integrated the trial-and-error learning process with the finding of *temporal difference (TD)* learning from psychology. TD learning quickly became indispensable in scaling RL for larger systems. On the basis of dynamic programming and TD learning, Watkins and Dayan (1992) laid the foundations for present day RL using the Markov decision process (MDP) and proposing the famous Q-learning method as the solver. As a dynamic programming method, the original Q-learning process inherits Bellman’s “curse of dimensionality” (Bellman, 1952), which strongly limits its applications when the number of state variables is large. To overcome such a bottleneck, Bertsekas and Tsitsiklis (1996) proposed approximate dynamic programming methods based on neural networks. More recently, Mnih et al. (2015) from DeepMind made a significant breakthrough by introducing the deep Q-learning (DQN) architecture, which leverages the representation power of DNNs for approximate dynamic programming methods. DQN has demonstrated human-level performance on 49 Atari games. Since then, deep RL techniques have become common in machine learning/AI and have attracted considerable attention from the research community.

RL originates from an understanding of animal behaviour where animals use trial-and-error to reinforce beneficial behaviours, which they then perform more frequently. During its development, computational RL incorporated ideas such as optimal control theory and other findings from psychology that help mimic the way humans make decisions to maximise the long-term profit of decision-making tasks. As a result, RL methods can naturally be used to train a computer program (an agent) to a performance level comparable to that of a human on certain tasks. The earliest success of RL methods against human players can be traced back to the game of

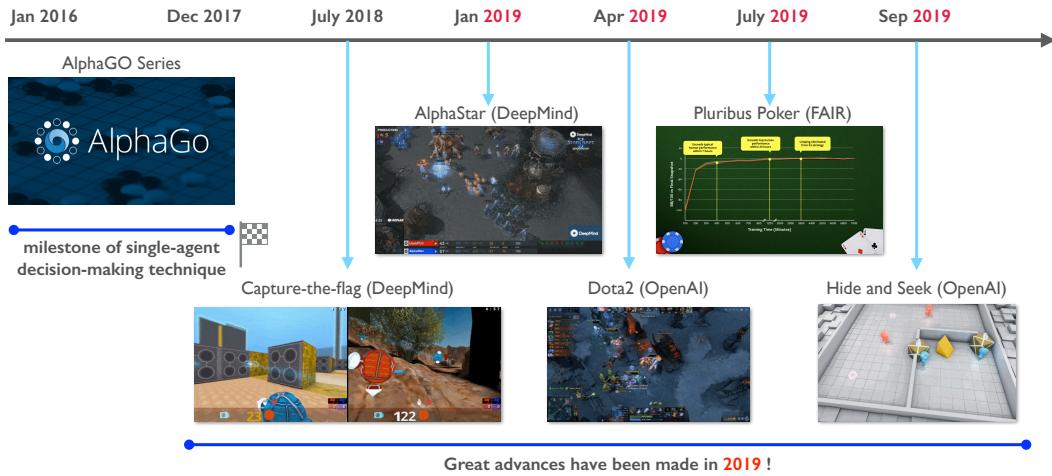


Figure 1.2: The success of the AlphaGo series marks the maturation of the single-agent decision-making process. The year 2019 was a booming year for MARL techniques; remarkable progress was achieved in solving immensely challenging multi-player real-strategy video games and multi-player incomplete-information poker games.

backgammon (Tesauro, 1995). More recently, the advancement of applying RL to solve sequential decision-making problems was marked by the remarkable success of the AlphaGo series (Silver et al., 2016, 2018, 2017), a self-taught RL agent that beats top professional players of the game GO, a game whose search space (10^{761} possible games) is even greater than the number of atoms in the universe¹.

In fact, the majority of successful RL applications, such as those for the game GO², robotic control (Kober et al., 2013), and autonomous driving (Shalev-Shwartz et al., 2016a), naturally involve the participation of multiple AI agents, which probe into the realm of MARL. As we would expect, the significant progress achieved by single-agent RL methods – marked by the 2016 success in GO – foreshadowed the breakthroughs of multi-agent RL techniques in the following years.

1.1.2 2019: A Booming Year for Multi-Agent RL

2019 was a booming year for MARL development as a series of breakthroughs were made in immensely challenging multi-agent tasks that people used to believe were

¹There are an estimated 10^{82} atoms in the universe. If one had one trillion computers, each processing one trillion states per second for one trillion years, one could only reach 10^{43} states.

²Arguably, AlphaGo can also be treated as a multi-agent technique if we consider the opponent in self-play as another agent.

impossible to solve via AI. Nevertheless, the progress made in the field of MARL, though remarkable, has been overshadowed to some extent by the prior success of AlphaGo (Chalmers, 2020). It is possible that the AlphaGo series (Silver et al., 2016, 2018, 2017) has largely fulfilled people’s expectations for the effectiveness of RL methods, such that there is a lack of interest in further advancements in the field. The ripples caused by the progress of MARL were relatively mild among the research community. In this section, we highlight several pieces of work that we believe are important and could profoundly impact the future development of MARL techniques.

One popular test-bed of MARL is StarCraft II (Vinyals et al., 2017), a multi-player real-time strategy computer game that has its own professional league. In this game, each player has only limited information about the game state, and the dimension of the search space is orders of magnitude larger than that of GO (10^{26} possible choices for every move). The design of effective RL methods for StarCraft II was once believed to be a long-term challenge for AI (Vinyals et al., 2017). However, a breakthrough was accomplished by AlphaStar in 2019 (Vinyals et al., 2019a), which has exhibited grandmaster-level skills by ranking above 99.8% of human players.

Another prominent video game-based test-bed for MARL is Dota2, a zero-sum game played by two teams, each composed of five players. From each agent’s perspective, in addition to the difficulty of incomplete information (similar to StarCraft II), Dota2 is more challenging, in the sense that both cooperation among team members and competition against the opponents must be considered. The OpenAI Five AI system (Pachocki et al., 2018) demonstrated superhuman performance in Dota2 by defeating world champions in a public e-sports competition.

In addition to StarCraft II and Dota2, Jaderberg et al. (2019) and Baker et al. (2019a) showed human-level performance in capture-the-flag and hide-and-seek games, respectively. Although the games themselves are less sophisticated than either StarCraft II or Dota2, it is still non-trivial for AI agents to master their tactics, so the agents’ impressive performance again demonstrates the efficacy of MARL.

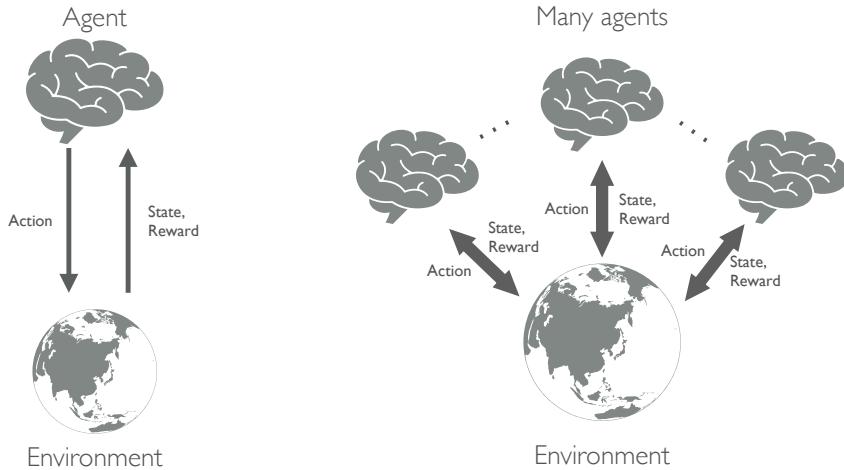


Figure 1.3: Diagram of a single-agent MDP (left) and a multi-agent MDP (right).

Interestingly, both authors reported emergent behaviours induced by their proposed MARL methods that humans can understand and are grounded in physical theory.

One last remarkable achievement of MARL worth mentioning is its application to the poker game Texas hold’ em, which is a multi-player extensive-form game with incomplete information accessible to the player. Heads-up (namely, two player) no-limit hold’em has more than 6×10^{161} information states. Only recently have ground-breaking achievements in the game been made, thanks to MARL. Two independent programs, *DeepStack* (Moravčík et al., 2017) and *Libratus* (Brown and Sandholm, 2018), are able to beat professional human players. Even more recently, Libratus was upgraded to Pluribus (Brown and Sandholm, 2019) and showed remarkable performance by winning over one million dollars from five elite human professionals in a no-limit setting.

For a deeper understanding of RL and MARL, mathematical notation and deconstruction of the concepts are needed. In the next section, we provide mathematical formulations for these concepts, starting from single-agent RL and progressing to multi-agent RL methods.

1.2 Single-Agent RL

Through trial and error, an RL agent attempts to find the optimal policy to maximise its long-term reward. This process is formulated by Markov Decision Processes.

1.2.1 Problem Formulation: Markov Decision Process (MDP)

Definition 1 (Markov Decision Process). *An MDP can be described by a tuple of key elements $\langle \mathbb{S}, \mathbb{A}, P, R, \gamma \rangle$.*

- \mathbb{S} : *the set of environmental states.*
- \mathbb{A} : *the set of agent's possible actions.*
- $P : \mathbb{S} \times \mathbb{A} \rightarrow \Delta(\mathbb{S})$: *for each time step $t \in \mathbb{N}$, given agent's action $a \in \mathbb{A}$, the transition probability from a state $s \in \mathbb{S}$ to the state in the next time step $s' \in \mathbb{S}$.*
- $R : \mathbb{S} \times \mathbb{A} \times \mathbb{S} \rightarrow \mathbb{R}$: *the reward function that returns a scalar value to the agent for a transition from s to s' as a result of action a . The rewards have absolute values uniformly bounded by R_{\max} .*
- $\gamma \in [0, 1]$ *is the discount factor that represents the value of time.*

At each time step t , the environment has a state s_t . The learning agent observes this state³ and executes an action a_t . The action makes the environment transition into the next state $s_{t+1} \sim P(\cdot | s_t, a_t)$, and the new environment returns an immediate reward $R(s_t, a_t, s_{t+1})$ to the agent. The reward function can be also written as $R : \mathbb{S} \times \mathbb{A} \rightarrow \mathbb{R}$, which is interchangeable with $R : \mathbb{S} \times \mathbb{A} \times \mathbb{S} \rightarrow \mathbb{R}$ (see [Van Otterlo and Wiering \(2012\)](#), page 10). The goal of the agent is to solve the MDP: to find the optimal policy that maximises the reward over time. Mathematically, one common objective is for the agent to find a Markovian (i.e., the input depends on only the current state) and stationary (i.e., function form is time-independent) policy function⁴ $\pi : \mathbb{S} \rightarrow \Delta(\mathbb{A})$, with $\Delta(\cdot)$ denoting the probability simplex, which can guide it to take sequential actions such that the discounted cumulative reward is maximised:

$$\mathbb{E}_{s_{t+1} \sim P(\cdot | s_t, a_t)} \left[\sum_{t \geq 0} \gamma^t R(s_t, a_t, s_{t+1}) \mid a_t \sim \pi(\cdot | s_t), s_0 \right]. \quad (1.1)$$

³The agent can only observe part of the full environment state. The partially observable setting is introduced in Definition 7 as a special case of Dec-PODMP.

⁴Such an optimal policy exists as long as the transition function and the reward function are both Markovian and stationary ([Feinberg, 2010](#)).

Another common mathematical objective of an MDP is to maximise the time-average reward:

$$\lim_{T \rightarrow \infty} \mathbb{E}_{s_{t+1} \sim P(\cdot | s_t, a_t)} \left[\frac{1}{T} \sum_{t=0}^{T-1} R(s_t, a_t, s_{t+1}) \mid a_t \sim \pi(\cdot | s_t), s_0 \right], \quad (1.2)$$

which we do not consider in this work and refer to [Mahadevan \(1996\)](#) for a full analysis of the objective of time-average reward.

Based on the objective function of Eq. (1.1), under a given policy π , we can define the state-action function (namely, the Q-function, which determines the expected return from undertaking action a in state s) and the value function (which determines the return associated with the policy in state s) as:

$$Q^\pi(s, a) = \mathbb{E}^\pi \left[\sum_{t \geq 0} \gamma^t R(s_t, a_t, s_{t+1}) \mid a_0 = a, s_0 = s \right], \forall s \in \mathbb{S}, a \in \mathbb{A} \quad (1.3)$$

$$V^\pi(s) = \mathbb{E}^\pi \left[\sum_{t \geq 0} \gamma^t R(s_t, a_t, s_{t+1}) \mid s_0 = s \right], \forall s \in \mathbb{S} \quad (1.4)$$

where \mathbb{E}^π is the expectation under the probability measure \mathbb{P}^π over the set of infinitely long state-action trajectories $\tau = (s_0, a_0, s_1, a_1, \dots)$ and where \mathbb{P}^π is induced by state transition probability P , the policy π , the initial state s and initial action a (in the case of the Q-function). The connection between the Q-function and value function is $V^\pi(s) = \mathbb{E}_{a \sim \pi(\cdot | s)} [Q^\pi(s, a)]$ and $Q^\pi = \mathbb{E}_{s' \sim P(\cdot | s, a)} [R(s, a, s') + V^\pi(s')]$.

1.2.2 Justification of Reward Maximisation

The current model for RL, as given by Eq. (1.1), suggests that the expected value of a single reward function is sufficient for any problem we want our “intelligent agents” to solve. The justification for this idea is deeply rooted in the *von Neumann-Morgenstern (VNM) utility theory* ([Von Neumann and Morgenstern, 2007](#)). This theory essentially proves that an agent is *VNM-rational* if and only if there exists a real-valued utility (or, reward) function such that every preference of the agent is characterised by maximising the single expected reward. The VNM utility theorem is the basis for the well-known *expected utility theory* ([Schoemaker, 2013](#)), which

essentially states that *rationality* can be modelled as maximising an expected value. Specifically, the VNM utility theorem provides both necessary and sufficient conditions under which the expected utility hypothesis holds. In other words, rationality is equivalent to VNM-rationality, and it is safe to assume an intelligent entity will always choose the action with the highest expected utility in any complex scenarios.

Admittedly, it was accepted long before that some of the assumptions on rationality could be violated by real decision-makers in practice (Gigerenzer and Selten, 2002). In fact, those conditions are rather taken as the “axioms” of rational decision making. In the case of the multi-objective MDP, we are still able to convert multiple objectives into a single-objective MDP with the help of a *scalarisation function* through a two-timescale process; we refer to Roijers et al. (2013) for more details.

1.2.3 Solving Markov Decision Processes

One commonly used notion in MDPs is the (discounted-normalised) occupancy measure $\mu^\pi(s, a)$, which uniquely corresponds to a given policy π and vice versa (Syed et al., 2008, Theorem 2), defined by

$$\begin{aligned}\mu^\pi(s, a) &= \mathbb{E}_{s_t \sim P, a_t \sim \pi} \left[(1 - \gamma) \sum_{t \geq 0} \gamma^t \mathbb{1}_{(s_t = s \wedge a_t = a)} \right] \\ &= (1 - \gamma) \sum_{t \geq 0} \gamma^t \mathbb{P}^\pi(s_t = s, a_t = a),\end{aligned}\tag{1.5}$$

where $\mathbb{1}$ is an indicator function. Note that in Eq. (1.5), P is the state transitional probability and \mathbb{P}^π is the probability of specific state-action pairs when following stationary policy π . The physical meaning of $\mu^\pi(s, a)$ is that of a probability measure that counts the expected discounted number of visits to the individual admissible state-action pairs. Correspondingly, $\mu^\pi(s) = \sum_a \mu^\pi(s, a)$ is the discounted state visitation frequency, i.e., the stationary distribution of the Markov process induced by π . With the occupancy measure, we can write Eq. (1.4) as an inner product of $V^\pi(s) = \frac{1}{1-\gamma} \langle \mu^\pi(s, a), R(s, a) \rangle$. This implies that solving an MDP can be regarded as solving a linear program (LP) of $\max_\mu \langle \mu(s, a), R(s, a) \rangle$, and the optimal policy

is then

$$\pi^*(a|s) = \mu^*(s, a) / \mu^*(s) \quad (1.6)$$

However, this method for solving the MDP remains at a textbook level, aiming to offer theoretical insights but lacking practically in the case of a large-scale LP with millions of variables (Papadimitriou and Tsitsiklis, 1987). When the state-action space of an MDP is continuous, LP formulation cannot help solve either.

In the context of optimal control (Bertsekas, 2005), dynamic-programming approaches, such as policy iteration and value iteration, can also be applied to solve for the optimal policy that maximises Eq. (1.3) & Eq. (1.4), but these approaches require knowledge of the exact form of the model: the transition function $P(\cdot|s, a)$, and the reward function $R(s, a, s')$.

On the other hand, in the setting of RL, the agent learns the optimal policy by a trial-and-error process during its interaction with the environment rather than using prior knowledge of the model. The word “learning” essentially means that the agent turns its experience gained during the interaction into knowledge about the model of the environment. Based on the solution target, either the optimal policy or the optimal value function, RL algorithms can be categorised into two types: value-based methods and policy-based methods.

1.2.3.1 Value-Based Methods

For all MDPs with finite states and actions, there exists at least one deterministic stationary optimal policy (Sutton and Barto, 1998; Szepesvári, 2010). Value-based methods are introduced to find the optimal Q-function Q^* that maximises Eq. (1.3). Correspondingly, the optimal policy can be derived from the Q-function by taking the greedy action of $\pi^* = \arg \max_a Q^*(s, a)$. The classic Q-learning algorithm (Watkins and Dayan, 1992) approximates Q^* by \hat{Q} , and updates its value via temporal-difference learning (Sutton, 1988).

$$\underbrace{\hat{Q}(s_t, a_t)}_{\text{new value}} \leftarrow \underbrace{\hat{Q}(s_t, a_t)}_{\text{old value}} + \underbrace{\alpha}_{\text{learning rate}} \cdot \underbrace{\left(R_t + \gamma \cdot \max_{a \in \mathbb{A}} \hat{Q}(s_{t+1}, a) - \underbrace{\hat{Q}(s_t, a_t)}_{\text{old value}} \right)}_{\text{temporal difference target}} \quad (1.7)$$

Theoretically, given the Bellman optimality operator \mathbf{H}^* , defined by

$$(\mathbf{H}^*Q)(s, a) = \sum_{s'} P(s'|s, a) \left[R(s, a, s') + \gamma \max_{b \in \mathbb{A}} Q(s, b) \right], \quad (1.8)$$

we know it is a contraction mapping and the optimal Q-function is the unique⁵ fixed point, i.e., $\mathbf{H}^*(Q^*) = Q^*$. The Q-learning algorithm draws random samples of (s, a, R, s') in Eq. (1.7) to approximate Eq. (1.8), but is still guaranteed to converge to the optimal Q-function (Szepesvári and Littman, 1999) under the assumptions that the state-action sets are discrete and finite and are visited an infinite number of times. Munos and Szepesvári (2008) extended the convergence result to a more realistic setting by deriving the high probability error bound for an infinite state space with a finite number of samples.

Recently, Mnih et al. (2015) applied neural networks as a function approximator for the Q-function in updating Eq. (1.7). Specifically, DQN optimises the following equation:

$$\min_{\theta} \mathbb{E}_{(s_t, a_t, R_t, s_{t+1}) \sim \mathcal{D}} \left[\left(R_t + \gamma \max_{a \in \mathbb{A}} Q_{\theta^-}(s_{t+1}, a) - Q_{\theta}(s_t, a_t) \right)^2 \right]. \quad (1.9)$$

The neural network parameters θ is fitted by drawing i.i.d. samples from the replay buffer \mathcal{D} and then updating in a supervised learning fashion. Q_{θ^-} is a slowly updated target network that helps stabilise training. The convergence property and finite sample analysis of DQN have been studied by Yang et al. (2019c).

1.2.3.2 Policy-Based Methods

Policy-based methods are designed to directly search over the policy space to find the optimal policy π^* . One can parameterise the policy expression $\pi^* \approx \pi_{\theta}(\cdot|s)$ and update the parameter θ in the direction that maximises the cumulative reward $\theta \leftarrow \theta + \alpha \nabla_{\theta} V^{\pi_{\theta}}(s)$ to find the optimal policy. However, the gradient will depend on the unknown effects of policy changes on the state distribution. The famous policy gradient (PG) theorem (Sutton et al., 2000) derives an analytical solution

⁵Note that although the optimal Q-function is unique, its corresponding optimal policies may have multiple candidates.

that does not involve the state distribution, that is:

$$\nabla_{\theta} V^{\pi_{\theta}}(s) = \mathbb{E}_{s \sim \mu^{\pi_{\theta}}(\cdot), a \sim \pi_{\theta}(\cdot|s)} \left[\nabla_{\theta} \log \pi_{\theta}(a|s) \cdot Q^{\pi_{\theta}}(s, a) \right] \quad (1.10)$$

where $\mu^{\pi_{\theta}}$ is the state occupancy measure under policy π_{θ} and $\nabla \log \pi_{\theta}(a|s)$ is the updating score of the policy. When the policy is deterministic and the action set is continuous, one obtains the deterministic policy gradient (DPG) theorem ([Silver et al., 2014](#)) as

$$\nabla_{\theta} V^{\pi_{\theta}}(s) = \mathbb{E}_{s \sim \mu^{\pi_{\theta}}(\cdot)} \left[\nabla_{\theta} \pi_{\theta}(a|s) \cdot \nabla_a Q^{\pi_{\theta}}(s, a) \Big|_{a=\pi_{\theta}(s)} \right]. \quad (1.11)$$

A classic implementation of the PG theorem is REINFORCE ([Williams, 1992](#)), which uses a sample return $R_t = \sum_{i=t}^T \gamma^{i-t} r_i$ to estimate $Q^{\pi_{\theta}}$. Alternatively, one can use a model of Q_{ω} (also called *critic*) to approximate the true $Q^{\pi_{\theta}}$ and update the parameter ω via TD learning. This approach gives rise to the famous actor-critic methods ([Konda and Tsitsiklis, 2000a; Peters and Schaal, 2008](#)). Important variants of actor-critic methods include trust-region methods ([Schulman et al., 2015, 2017](#)), PG with optimal baselines ([Weaver and Tao, 2001; Zhao et al., 2011](#)), soft actor-critic methods ([Haarnoja et al., 2018](#)), and deep deterministic policy gradient (DDPG) methods ([Lillicrap et al., 2015](#)).

1.3 Multi-Agent RL

In the multi-agent scenario, much like in the single-agent scenario, each agent is still trying to solve the sequential decision-making problem through a trial-and-error procedure. The difference is that the evolution of the environmental state and the reward function that each agent receives is now determined by all agents' joint actions (see Figure 1.3). As a result, agents need to take into account and interact with not only the environment but also other learning agents. A decision-making process that involves multiple agents is usually modelled through a stochastic game ([Shapley, 1953](#)), also known as a Markov game ([Littman, 1994](#)).



Figure 1.4: A snapshot of stochastic time in the intersection example. The scenario is abstracted such that there are two cars, with each car taking one of two possible actions: to yield or to rush. The outcome of each joint action pair is represented by a normal-form game, with the reward value for the row player denoted in red and that for the column player denoted in black. The Nash equilibria (NE) of this game are (rush, yield) and (yield, rush). If both cars maximise their own reward selfishly without considering the others, they will end up in an accident.

1.3.1 Problem Formulation: Stochastic Game

Definition 2 (Stochastic Game). *A stochastic game can be regarded as a multi-player⁶ extension to the MDP in Definition 1. Therefore, it is also defined by a set of key elements $\langle N, \mathbb{S}, \{\mathbb{A}^i\}_{i \in \{1, \dots, N\}}, P, \{R^i\}_{i \in \{1, \dots, N\}}, \gamma \rangle$.*

- N : the number of agents, $N = 1$ degenerates to a single-agent MDP, $N \gg 2$ is referred as many-agent cases in this thesis.
- \mathbb{S} : the set of environmental states shared by all agents.
- \mathbb{A}^i : the set of actions of agent i . We denote $\mathbb{A} := \mathbb{A}^1 \times \dots \times \mathbb{A}^N$.
- $P : \mathbb{S} \times \mathbb{A} \rightarrow \Delta(\mathbb{S})$: for each time step $t \in \mathbb{N}$, given agents' joint actions $\mathbf{a} \in \mathbb{A}$, the transition probability from state $s \in \mathbb{S}$ to state $s' \in \mathbb{S}$ in the next time step.
- $R^i : \mathbb{S} \times \mathbb{A} \times \mathbb{S} \rightarrow \mathbb{R}$: the reward function that returns a scalar value to the i -th agent for a transition from (s, \mathbf{a}) to s' . The rewards have absolute values uniformly bounded by R_{max} .
- $\gamma \in [0, 1]$ is the discount factor that represents the value of time.

We use the superscript of (\cdot^i, \cdot^{-i}) (for example, $\mathbf{a} = (a^i, a^{-i})$), when it is necessary to distinguish between agent i and all other $N - 1$ opponents.

⁶Player is a common word used in game theory; agent is more commonly used in machine learning. We do not discriminate between their usages in this work. The same holds for strategy vs policy and utility/payoff vs reward. Each pair refers to the game theory usage vs machine learning usage.

Ultimately, the stochastic game (SG) acts as a framework that allows simultaneous moves from agents in a decision-making scenario⁷. The game can be described sequentially, as follows: At each time step t , the environment has a state s_t , and given s_t , each agent executes its action a_t^i , simultaneously with all other agents. The joint action from all agents makes the environment transition into the next state $s_{t+1} \sim P(\cdot | s_t, \mathbf{a}_t)$; then, the environment determines an immediate reward $R^i(s_t, \mathbf{a}_t, s_{t+1})$ for each agent. As seen in the single-agent MDP scenario, the goal of each agent i is to solve the SG. In other words, each agent aims to find a behavioural policy (or, a mixed strategy⁸ in game theory terminology (Osborne and Rubinstein, 1994)), that is, $\pi^i \in \Pi^i : \mathbb{S} \rightarrow \Delta(\mathbb{A}^i)$ that can guide the agent to take sequential actions such that the discounted cumulative reward⁹ in Eq. (1.12) is maximised. Here, $\Delta(\cdot)$ is the probability simplex on a set. In game theory, π^i is also called a pure strategy (vs a mixed strategy) if $\Delta(\cdot)$ is replaced by a Dirac measure.

$$V^{\pi^i, \pi^{-i}}(s) = \mathbb{E}_{s_{t+1} \sim P(\cdot | s_t, a_t), a^{-i} \sim \pi^{-i}(\cdot | s_t)} \left[\sum_{t \geq 0} \gamma^t R_t^i(s_t, \mathbf{a}_t, s_{t+1}) \mid a_t^i \sim \pi^i(\cdot | s_t), s_0 \right]. \quad (1.12)$$

Comparison of Eq. (1.12) with Eq. (1.4) indicates that the optimal policy of each agent is influenced by not only its own policy but also the policies of the other agents in the game. This scenario leads to fundamental differences in the *solution concept* between single-agent RL and multi-agent RL.

1.3.2 Solving Stochastic Games

An SG can be considered as a sequence of normal-form games, which are games that can be represented in a matrix. Take the original intersection scenario as an example (see Figure 1.4). A snapshot of the SG at time t (stage game) can be

⁷Extensive-form games allow agents to take sequential moves; the full description can be found in (Shoham and Leyton-Brown, 2008, Chapter 5).

⁸A behavioural policy refers to a function map from the history $(s_0, a_0^i, s_1, a_1^i, \dots, s_{t-1})$ to an action. The policy is typically assumed to be Markovian such that it depends on only the current state s_t rather than the entire history. A mixed strategy refers to a randomisation over pure strategies (for example, the actions). In SGs, the behavioural policy and mixed policy are exactly the same. In extensive-form games, they are different, but if the agent retains the history of previous actions and states (has perfect recall), each behavioural strategy has a realisation-equivalent mixed strategy, and vice versa (Kuhn, 1950a).

⁹Similar to single-agent MDP, we can adopt the objective of time-average rewards.

represented as a normal-form game in a matrix format. The rows correspond to the action set \mathbb{A}^1 for agent 1, and the columns correspond to the action set \mathbb{A}^2 for agent 2. The values of the matrix are the rewards given for each of the joint action pairs. In this scenario, if both agents care only about maximising their own possible reward with no consideration of other agents (the solution concept in a single-agent RL problem) and choose the action to rush, they will reach the outcome of crashing into each other. Clearly, this state is unsafe and is thus sub-optimal for each agent, despite the fact that the possible reward was the highest for each agent when rushing. Therefore, to solve an SG and truly maximise the cumulative reward, each agent must take strategic actions with consideration of others when determining their policies.

Unfortunately, in contrast to MDPs, which have polynomial time-solvable linear-programming formulations, solving SGs usually involves applying Newton's method for solving nonlinear programs. However, there are two special cases of two-player general-sum discounted-reward SGs that can still be written as LPs ([Shoham and Leyton-Brown, 2008](#), Chapter 6.2). They are as follows:

- *single-controller SG*: the transition dynamics are determined by a single player, i.e., $P(\cdot|\mathbf{a}, s) = P(\cdot|a^i, s)$ if the i -th index in the vector \mathbf{a} is $\mathbf{a}[i] = a^i, \forall s \in \mathbb{S}, \forall \mathbf{a} \in \mathbb{A}$.
- *separable reward state independent transition (SR-SIT) SG*: the states and the actions have independent effects on the reward function and the transition function depends on only the joint actions, i.e., $\exists \alpha : \mathbb{S} \rightarrow \mathbb{R}, \beta : \mathbb{A} \rightarrow \mathbb{R}$ such that these two conditions hold: 1) $R^i(s, \mathbf{a}) = \alpha(s) + \beta(\mathbf{a}), \forall i \in \{1, \dots, N\}, \forall s \in \mathbb{S}, \forall \mathbf{a} \in \mathbb{A}$, and 2) $P(\cdot|s', \mathbf{a}) = P(\cdot|s, \mathbf{a}), \forall \mathbf{a} \in \mathbb{A}, \forall s, s' \in \mathbb{S}$.

1.3.2.1 Value-Based MARL Methods

The single-agent Q-learning update in Eq. (1.7) still holds in the multi-agent case. In the t -th iteration, for each agent i , given the transition data $\{(s_t, \mathbf{a}_t, R^i, s_{t+1})\}_{t \geq 0}$ sampled from the replay buffer, it updates only the value of $Q(s_t, \mathbf{a}_t)$ and keeps the

other entries of the Q-function unchanged. Specifically, we have

$$\mathcal{Q}^i(s_t, \mathbf{a}_t) \leftarrow \mathcal{Q}^i(s_t, \mathbf{a}_t) + \alpha \cdot \left(R^i + \gamma \cdot \mathbf{eval}^i \left(\{\mathcal{Q}^i(s_{t+1}, \cdot)\}_{i \in \{1, \dots, N\}} \right) - \mathcal{Q}^i(s_t, \mathbf{a}_t) \right). \quad (1.13)$$

Compared to Eq. (1.7), the max operator is changed to $\mathbf{eval}^i(\{\mathcal{Q}^i(s_{t+1}, \cdot)\}_{i \in \{1, \dots, N\}})$ in Eq. (1.13) to reflect the fact that each agent can no longer consider only itself but must evaluate the situation of the stage game at time step $t+1$ by considering all agents' interests, as represented by the set of their Q-functions. Then, the optimal policy can be solved by $\mathbf{solve}^i(\{\mathcal{Q}^i(s_{t+1}, \cdot)\}_{i \in \{1, \dots, N\}}) = \pi^{i,*}$. Therefore, we can further write the evaluation operator as

$$\mathbf{eval}^i \left(\{\mathcal{Q}^i(s_{t+1}, \cdot)\}_{i \in \{1, \dots, N\}} \right) = V^i \left(s_{t+1}, \left\{ \mathbf{solve}^i \left(\{\mathcal{Q}^i(s_{t+1}, \cdot)\}_{i \in \{1, \dots, N\}} \right) \right\}_{i \in \{1, \dots, N\}} \right). \quad (1.14)$$

In summary, \mathbf{solve}^i returns agent i 's part of the optimal policy at some equilibrium point (not necessarily corresponding to its largest possible reward), and \mathbf{eval}^i gives agent i 's expected long-term reward under this equilibrium, assuming all other agents agree to play the same equilibrium.

1.3.2.2 Policy-Based MARL Methods

The value-based approach suffers from the curse of dimensionality due to the combinatorial nature of multi-agent systems (for further discussion, see Section 1.4.1). This characteristic necessitates the development of policy-based algorithms with function approximations. Specifically, each agent learns its own optimal policy $\pi_{\theta^i}^i : \mathbb{S} \rightarrow \Delta(\mathbb{A}^i)$ by updating the parameter θ^i of, for example, a neural network. Let $\theta = (\theta^i)_{i \in \{1, \dots, N\}}$ represent the collection of policy parameters for all agents, and let $\boldsymbol{\pi}_\theta := \prod_{i \in \{1, \dots, N\}} \pi_{\theta^i}^i(a^i | s)$ be the joint policy. To optimise the parameter θ^i , the policy gradient theorem in Section 1.2.3.2 can be extended to the multi-agent context. Given agent i 's objective function $J^i(\theta) = \mathbb{E}_{s \sim P, \mathbf{a} \sim \boldsymbol{\pi}_\theta} [\sum_{t \geq 0} \gamma^t R_t^i]$, we have:

$$\nabla_{\theta^i} J^i(\theta) = \mathbb{E}_{s \sim \mu^{\boldsymbol{\pi}_\theta}(\cdot), \mathbf{a} \sim \boldsymbol{\pi}_\theta(\cdot | s)} \left[\nabla_{\theta^i} \log \pi_{\theta^i}(a^i | s) \cdot Q^{i, \boldsymbol{\pi}_\theta}(s, \mathbf{a}) \right]. \quad (1.15)$$

Considering a continuous action set with a deterministic policy, we have the multi-agent deterministic policy gradient (MADDPG) (Lowe et al., 2017a) written as

$$\nabla_{\theta^i} J(\theta) = \mathbb{E}_{s \sim \mu^{\pi_\theta}(\cdot)} \left[\nabla_{\theta^i} \log \pi_{\theta^i}(a^i | s) \cdot \nabla_{a_i} Q^{i, \pi_\theta}(s, \mathbf{a}) \Big|_{\mathbf{a}=\pi_\theta(s)} \right]. \quad (1.16)$$

Note that in both Eqs. (1.15) & (1.16), the expectation over the joint policy π_θ implies that other agents' policies must be observed; this is often a strong assumption for many real-world applications.

1.3.2.3 Solution Concept of the Nash Equilibrium

Game theory plays an essential role in multi-agent learning by offering so-called *solution concepts* that describe the outcomes of a game by showing which strategies will finally be adopted by players. Many types of solution concepts exist for MARL (see Section 1.4.2), among which the most famous is probably the Nash equilibrium (NE) in non-cooperative game theory (Nash, 1951). The word “non-cooperative” does not mean agents cannot collaborate or have to fight against each other all the time, it merely means that each agent maximises its own reward independently and that agents cannot group into coalitions to make collective decisions.

In a normal-form game, the NE characterises an equilibrium point of the joint strategy profile $(\pi^{1,*}, \dots, \pi^{N,*})$, where each agent acts according to their **best response** to the others. The best response produces the optimal outcome for the player once all other players' strategies have been considered. Player i 's best response¹⁰ to π^{-i} is a set of policies in which the following condition is satisfied.

$$\pi^{i,*} \in \mathbf{Br}(\pi^{-i}) := \left\{ \arg \max_{\hat{\pi} \in \Delta(\mathbb{A}^i)} \mathbb{E}_{\hat{\pi}^i, \pi^{-i}} [R^i(a^i, a^{-i})] \right\}. \quad (1.17)$$

NE states that if all players are perfectly rational, none of them will have a motivation to deviate from their best response $\pi^{i,*}$ given others are playing $\pi^{-i,*}$. Note that NE is defined in terms of the best response, which relies on relative reward values, suggesting that the exact values of rewards are not required for identifying NE.

¹⁰Best responses may not be unique; if a mixed-strategy best response exists, there must be at least one best response that is also a pure strategy.

In fact, NE is invariant under positive affine transformations of a players' reward functions. By applying Brouwer's fixed point theorem, [Nash \(1951\)](#) proved that a mixed-strategy NE always exists for any games with a finite set of actions. In the example of driving through an intersection in Figure 1.4, the NE are $(yield, rush)$ and $(rush, yield)$.

For a SG, one commonly used equilibrium is a stronger version of the NE, called the Markov perfect NE ([Maskin and Tirole, 2001](#)), which is defined by:

Definition 3 (Nash Equilibrium for Stochastic Game). A *Markovian strategy profile* $\pi^* = (\pi^{i,*}, \pi^{-i,*})$ is a Markov perfect NE of a SG defined in Definition 2 if the following condition holds

$$V^{\pi^{i,*}, \pi^{-i,*}}(s) \geq V^{\pi^i, \pi^{-i,*}}(s), \quad \forall s \in \mathbb{S}, \forall \pi^i \in \Pi^i, \forall i \in \{1, \dots, N\}. \quad (1.18)$$

“Markovian” means the Nash policies are measurable with respect to a particular partition of possible histories (usually referring to the last state). The word “perfect” means that the equilibrium is also subgame-perfect ([Selten, 1965](#)) regardless of the starting state. Considering the sequential nature of SGs, these assumptions are necessary, while still maintaining generality. Hereafter, the Markov perfect NE will be referred to as NE.

A mixed-strategy NE¹¹ always exists for both discounted and average-reward¹² SGs ([Filar and Vrieze, 2012](#)), though they may not be unique. In fact, checking for uniqueness is *NP*-hard ([Conitzer and Sandholm, 2002](#)). With the NE as the solution concept of optimality, we can re-write Eq. (1.14) as:

$$\text{eval}_{\text{Nash}}^i \left(\{Q^i(s_{t+1}, \cdot)\}_{i \in \{1, \dots, N\}} \right) = V^i \left(s_{t+1}, \left\{ \text{Nash}^i \left(\{Q^i(s_{t+1}, \cdot)\}_{i \in \{1, \dots, N\}} \right) \right\}_{i \in \{1, \dots, N\}} \right). \quad (1.19)$$

In the above equation, $\text{Nash}^i(\cdot) = \pi^{i,*}$ computes the NE of agent i 's strategy, and

¹¹Note that this is different from a single-agent MDP, where a single, “pure” strategy optimal policy always exists. A simple example is the rock-paper-scissors game, where none of the pure strategies is the NE and the only NE is to mix between the three equally.

¹²Average-reward SGs entail more subtleties because the limit of Eq. (1.2) in the multi-agent setting may be a cycle and thus not exist. Instead, NE are proved to exist on a special class of irreducible SGs, where every stage game can be reached regardless of the adopted policy.

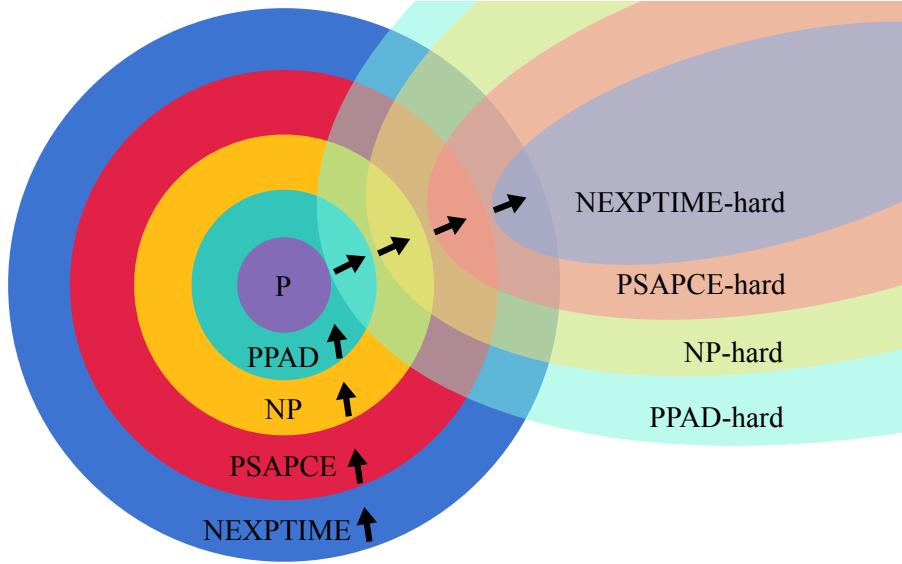


Figure 1.5: The landscape of different complexity classes. Relevant examples are 1) solving the NE in a two-player zero-sum game, *P*-complete (Neumann, 1928), 2) solving the NE in a general-sum game, *PPAD*-hard (Daskalakis et al., 2009), 3) checking the uniqueness of the NE, *NP*-hard (Conitzer and Sandholm, 2002), 4) checking whether a pure-strategy NE exists in a stochastic game, *PSPACE*-hard (Conitzer and Sandholm, 2008), and 5) solving Dec-POMDP, *NEXPTIME*-hard (Bernstein et al., 2002).

$V^i(s, \{\text{Nash}^i\}_{i \in \{1, \dots, N\}})$ is the expected payoff for agent i from state s onwards under this equilibrium. Eq. (1.19) and Eq. (1.13) form the learning steps of Nash Q-learning (Hu et al., 1998). This process essentially leads to the outcome of a learnt set of optimal policies that reach NE for every single-stage game encountered. In the case when NE is not unique, Nash-Q adopts hand-crafted rules for equilibrium selection (e.g., all players choose the first NE). Furthermore, similar to normal Q-learning, the Nash-Q operator defined in Eq. (1.20) is also proved to be a contraction mapping, and the stochastic updating rule provably converges to the NE for all states when the NE is unique:

$$(\mathbf{H}^{\text{Nash}} Q)(s, a) = \sum_{s'} P(s'|s, a) \left[R(s, a, s') + \gamma \cdot \text{eval}_{\text{Nash}}^i \left(\{Q^i(s_{t+1}, \cdot)\}_{i \in \{1, \dots, N\}} \right) \right]. \quad (1.20)$$

The process of finding a NE in a two-player general-sum game can be formulated as a linear complementarity problem (LCP), which can then be solved using the *Lemke-Howson* algorithm (Shapley, 1974). However, the exact solution for

games with more than three players is unknown. In fact, the process of finding the NE is computationally demanding. Even in the case of two-player games, the complexity of solving the NE is *PPAD*-hard (polynomial parity arguments on directed graphs) (Chen and Deng, 2006; Daskalakis et al., 2009); therefore, in the worst-case scenario, the solution could take time that is exponential in relation to the game size. This complexity¹³ prohibits any brute force or exhaustive search solutions unless $P = NP$ (see Figure 1.5). As we would expect, the NE is much more difficult to solve for general SGs, where determining whether a pure-strategy NE exists is *PSPACE*-hard. Even if the SG has a finite-time horizon, the calculation remains *NP*-hard (Conitzer and Sandholm, 2008). When it comes to approximation methods to ε -NE, the best known polynomially computable algorithm can achieve $\varepsilon = 0.3393$ on bimatrix games (Tsaknakis and Spirakis, 2007); its approach is to turn the problem of finding NE into an optimisation problem that searches for a stationary point.

1.3.2.4 Special Types of Stochastic Games

To summarise the solutions to SGs, one can think of the “master” equation

$$\text{Normal-form game solver} + \text{MDP solver} = \text{Stochastic game solver},$$

which was first summarised by Bowling and Veloso (2000) (in Table 4). The first term refers to solving an equilibrium (NE) for the stage game encountered at every time step. It assumes the transition and reward function is known. The second term refers to applying a RL technique (such as Q-learning) to model the temporal structure in the sequential decision-making process. It assumes to only receive observations of the transition and reward function. The combination of the two gives a solution to SGs, where agents reach a certain type of equilibrium at each and every time step during the game.

¹³The class of *NP*-complete is not suitable to describe the complexity of solving the NE because the NE is proven to always exist (Nash, 1951), while a typical *NP*-complete problem – the travelling salesman problem (TSP), for example – searches for the solution to the question: “Given a distance matrix and a budget B , find a tour that is cheaper than B , or report that none exists (Daskalakis et al., 2009).”

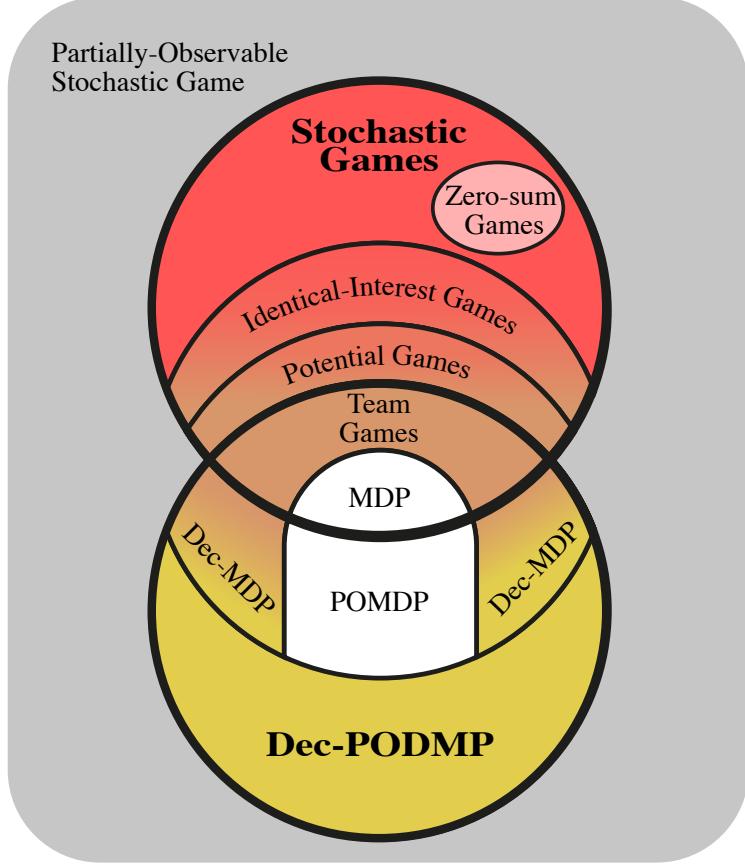


Figure 1.6: Venn diagram of different types of games in the context of POSGs. The intersection of SG and Dec-POMDP is the team game. In the upper-half SG, we have $MDP \subset \text{team games} \subset \text{potential games} \subset \text{identical-interest games} \subset SGs$, and zero-sum games $\subset SGs$. In the bottom-half Dec-POMDP, we have $MDP \subset \text{team games} \subset \text{Dec-MDP} \subset \text{Dec-POMDPs}$, and $MDP \subset POMDP \subset \text{Dec-POMDP}$. We refer to Sections (1.3.2.4 & 1.3.2.5) for detailed definitions of these games.

Since solving general SGs with NE as the solution concept for the normal-form game is computationally challenging, researchers instead aim to study special types of SGs that have tractable solution concepts. In this section, we provide a brief summary of these special types of games.

Definition 4 (Special Types of Stochastic Games). *Given the general form of SG in Definition 2, we have the following special cases:*

- **normal-form game/repeated game:** $|S| = 1$, see the example in Figure 1.4. These games have only a single state. Though not theoretically grounded, it is practically easier to solve a small-scale SG.

- **identical-interest setting**¹⁴: agents share the same learning objective, which we denote as R . Since all agents are treated independently, each agent can safely choose the action that maximises its own reward. As a result, single-agent RL algorithms can be applied safely, and a decentralised method developed. Several types of SGs fall into this category.
 - **team games/fully cooperative games/multi-agent MDP (MMDP)**: agents are assumed to be homogeneous and interchangeable, so importantly, they share the same reward function¹⁵, $R = R^1 = R^2 = \dots = R^N$.
 - **team-average reward games/networked multi-agent MDP (M-MDP)**: agents can have different reward functions, but they share the same objective, $R = \frac{1}{N} \sum_{i=1}^N R^i$.
 - **stochastic potential games**: agents can have different reward functions, but their mutual interests are described by a shared potential function $R = \phi$, defined as $\phi : \mathbb{S} \times \mathbb{A} \rightarrow \mathbb{R}$ such that $\forall (a^i, a^{-i}), (b^i, a^{-i}) \in \mathbb{A}, \forall i \in \{1, \dots, N\}, \forall s \in \mathbb{S}$ and the following equation holds:

$$R^i(s, (a^i, a^{-i})) - R^i(s, (b^i, a^{-i})) = \phi(s, (a^i, a^{-i})) - \phi(s, (b^i, a^{-i})). \quad (1.21)$$

Games of this type are guaranteed to have a pure-strategy NE ([Mguni, 2020](#)). Moreover, potential games degenerate to team games if one chooses the reward function to be a potential function.

- **zero-sum setting**: agents share opposite interests and act competitively, and each agent optimises against the worst-case scenario. The NE in a zero-sum setting can be solved using a linear program (LP) in polynomial time because of the minimax theorem developed by [Neumann \(1928\)](#). The idea of min-max

¹⁴In some of the literature on this topic, identical-interest games are equivalent to team games. Here, we refer to this type of game as a more general class of games that involve a shared objective function that all agents collectively optimise, although their individual reward functions can still be different.

¹⁵In some of the literature on this topic (for example, [Wang and Sandholm \(2003\)](#)), agents are assumed to receive the same expected reward in a team game, which means in the presence of noise, different agents may receive different reward values at a particular moment.

values is also related to robustness in machine learning. We can subdivide the zero-sum setting as follows:

- **two-player constant-sum games**: $R^1(s, a, s') + R^2(s, a, s') = c, \forall (s, a, s')$, where c is a constant and usually $c = 0$. For cases when $c \neq 0$, one can always subtract the constant c for all payoff entries to make the game zero-sum.
- **two-team competitive games**: two teams compete against each other, with team sizes N_1 and N_2 . Their reward functions are:

$$\{R^{1,1}, \dots, R^{1,N_1}, R^{2,1}, \dots, R^{2,N_2}\}.$$

Team members within a team share the same objective of either

$$R^1 = \sum_{i \in \{1, \dots, N_1\}} R^{1,i} / N_1,$$

or

$$R^2 = \sum_{j \in \{1, \dots, N_2\}} R^{2,j} / N_2,$$

and $R^1 + R^2 = 0$.

- **harmonic games**: Any normal-form game can be decomposed into a potential game plus a harmonic game (Candogan et al., 2011). A harmonic game (for example, rock-paper-scissors) can be regarded as a general class of zero-sum games with a harmonic property. Let $\forall \mathbf{p} \in \mathbb{A}$ be a joint pure-strategy profile, and let $\mathbb{A}^{[-i]} = \{\mathbf{q} \in \mathbb{A} : \mathbf{q}^i \neq \mathbf{p}^i, \mathbf{q}^{-i} = \mathbf{p}^{-i}\}$ be the set of strategies that differ from \mathbf{p} on agent i ; then, the harmonic property is:

$$\sum_{i \in \{1, \dots, N\}} \sum_{\mathbf{q} \in \mathbb{A}^{[-i]}} (R^i(\mathbf{p}) - R^i(\mathbf{q})) = 0, \quad \forall \mathbf{p} \in \mathbb{A}.$$

- **linear-quadratic (LQ) setting**: the transition model follows linear dynamics, and the reward function is quadratic with respect to the states and actions.

Compared to a black-box reward function, LQ games offer a simple setting. For example, actor-critic methods are known to facilitate convergence to the NE of zero-sum LQ games (Al-Tamimi et al., 2007). Again, the LQ setting can be subdivided as follows:

- **two-player zero-sum LQ games**: $Q \in \mathbb{R}^{|\mathbb{S}|}$, $U^1 \in \mathbb{R}^{|\mathbb{A}^1|}$ and $W^2 \in \mathbb{R}^{|\mathbb{A}^2|}$ are the known cost matrices for the state and action spaces, respectively, while the matrices $A \in \mathbb{R}^{|\mathbb{S}| \times |\mathbb{S}|}$, $B \in \mathbb{R}^{|\mathbb{S}| \times |\mathbb{A}^1|}$, $C \in \mathbb{R}^{|\mathbb{S}| \times |\mathbb{A}^2|}$ are usually unknown to the agent:

$$\begin{aligned} s_{t+1} &= As_t + Ba_t^1 + Ca_t^2, \quad s_0 \sim P_0, \\ R^1(a_t^1, a_t^2) &= -R^2(a_t^1, a_t^2) = -\mathbb{E}_{s_0 \sim P_0} \left[\sum_{t \geq 0} s_t^T Q s_t + a_t^{1T} U^1 a_t^1 - a_t^{2T} W^2 a_t^2 \right]. \end{aligned} \quad (1.22)$$

- **multi-player general-sum LQ games**: the difference with respect to a two-player game is that the summation of the agents' rewards does not necessarily equal zero:

$$\begin{aligned} s_{t+1} &= As_t + B\mathbf{a}_t, \quad s_0 \sim P_0, \\ R^i(\mathbf{a}) &= -\mathbb{E}_{s_0 \sim P_0} \left[\sum_{t \geq 0} s_t^T Q^i s_t + a_t^{iT} U^i a_t^i \right]. \end{aligned} \quad (1.23)$$

1.3.2.5 Partially Observable Settings

A partially observable stochastic game (POSG) assumes that agents have no access to the exact environmental state but only an observation of the true state through an observation function. Formally, this scenario is defined by:

Definition 5 (partially-observable stochastic games). A POSG is defined by the set $\langle N, \mathbb{S}, \{\mathbb{A}^i\}_{i \in \{1, \dots, N\}}, P, \{R^i\}_{i \in \{1, \dots, N\}}, \gamma, \underbrace{\{\mathbb{O}^i\}_{i \in \{1, \dots, N\}}}_{\text{newly added}}, O \rangle$. In addition to the SG defined in Definition 2, POSGs add the following terms:

- \mathbb{O}^i : an observation set for each agent i . The joint observation set is defined as $\mathbb{O} := \mathbb{O}^1 \times \dots \times \mathbb{O}^N$.
- $O : S \times \mathbb{A} \rightarrow \Delta(\mathbb{O})$: an observation function $O(\mathbf{o}|\mathbf{a}, s')$ denotes the probability of observing $\mathbf{o} \in \mathbb{O}$ given the action $\mathbf{a} \in \mathbb{A}$, and the new state $s' \in \mathbb{S}$ from the environment transition.

Each agent's policy now changes to $\pi^i \in \Pi^i : \mathbb{O} \rightarrow \Delta(\mathbb{A}^i)$.

Although the added partial-observability constraint is common in practice for many real-world applications, theoretically it exacerbates the difficulty of solving SGs. Even in the simplest setting of a two-player fully cooperative finite-horizon game, solving a POSG is *NEXP*-hard (see Figure 1.5), which means it requires super-exponential time to solve in the worst-case scenario (Bernstein et al., 2002). However, the benefits of studying games in the partially observable setting come from the algorithmic advantages. Centralised-training-with-decentralised-execution methods (Foerster et al., 2017a; Lowe et al., 2017a; Oliehoek et al., 2016; Rashid et al., 2018; Yang et al., 2020) have achieved many empirical successes, and together with DNNs, they hold great promise.

A POSG is one of the most general classes of games. An important subclass of POSGs is decentralised partially observable MDP (Dec-POMDP), where all agents share the same reward. Formally, this scenario is defined as follows:

Definition 6 (Dec-POMDP). *A Dec-POMDP is a special type of POSG defined in Definition 5 with $R^1 = R^2 = \dots = R^N$.*

Dec-POMDPs are related to single-agent MDPs through the partial observability condition, and they are also related to stochastic team games through the assumption of identical rewards. In other words, versions of both single-agent MDPs and stochastic team games are particular types of Dec-POMDPs (see Figure 1.6).

Definition 7 (Special types of Dec-POMDPs). *The following games are special types of Dec-POMDPs.*

- **partially observable MDP (POMDP)**: there is only one agent of interest, $N = 1$. This scenario is equivalent to a single-agent MDP in Definition 1 with a partial-observability constraint.
- **decentralised MDP (Dec-MDP)**: the agents in a Dec-MDP have joint full observability. That is, if all agents share their observations, they can recover the state of the Dec-MDP unanimously. Mathematically, we have $\forall \mathbf{o} \in \mathbb{O}, \exists s \in \mathbb{S}$ such that $\mathbb{P}(S_t = s | \mathbb{O}_t = \mathbf{o}) = 1$.
- **fully cooperative stochastic games**: assuming each agent has full observability, $\forall i = \{1, \dots, N\}, \forall o^i \in O^i, \exists s \in \mathbb{S}$ such that $\mathbb{P}(S_t = s | \mathbb{O}_t = o^i) = 1$. The fully-cooperative SG from Definition 4 is a type of Dec-POMDP.

I conclude Section 1.3 by presenting the relationships between the many different types of POSGs through a Venn diagram in Figure 1.6.

1.3.3 Problem Formulation: Extensive-Form Game

An SG assumes that a game is represented as a large table in each stage where the rows and columns of the table correspond to the actions of the two players¹⁶. Based on the big table, SGs model the situations in which agents act simultaneously and then receive their rewards. Nonetheless, for many real-world games, players take actions alternately. Poker is one class of games in which who plays first has a critical role in the players' decision-making process. Games with alternating actions are naturally described by an extensive-form game (EFG) (Osborne and Rubinstein, 1994; Von Neumann and Morgenstern, 1945) through a tree structure. Recently, Kovářík et al. (2019) has made a significant contribution in unifying the framework of EFGs and the framework of POSGs.

Figure 1.7 shows the game tree of two-player Kuhn poker (Kuhn, 1950b). In Kuhn poker, the dealer has three cards, a King, Queen, and Jack (King > Queen > Jack), each player is dealt one card (the orange nodes in Figure 1.7), and the third card is put aside unseen. The game then develops as follows.

¹⁶A multi-player game is represented as a high-dimensional tensor in an SG.

- **Player one** acts first; he/she can *check* or *bet*.
- If **player one** *checks*, then **player two** decides to *check* or *bet*.
- If **player two** *checks*, then the higher card wins 1\$ from the other player.
- If **player two** *bets*, then **player one** can *fold* or *call*.
- If **player one** *folds*, then **player two** wins 1\$ from **player one**.
- If **player one** *calls*, then the higher card wins 2\$ from the other player.
- If **player one** *bets*, then **player two** decides to *fold* or *call*.
- If **player two** *folds*, then **player one** wins 1\$ from **player two**.
- If **player two** calls, then the higher card wins 2\$ from the other player.

An important feature of EFGs is that they can handle imperfect information for multi-player decision making. In the example of Kuhn poker, the players do not know which card the opponent holds. However, unlike Dec-POMDP, which also models imperfect information in the SG setting but is intractable to solve, EFG, represented in an equivalent sequence form, can be solved by an LP in polynomial time in terms of game states (Koller and Megiddo, 1992). In the next section, we first introduce EFG and then consider the sequence form of EFG.

Definition 8 (Extensive-form Game). *An (imperfect-information) EFG can be described by a tuple of key elements $\langle N, \mathbb{A}, \mathbb{H}, \mathbb{T}, \{R^i\}_{i \in \{1, \dots, N\}}, \chi, \rho, P, \{\mathbb{S}^i\}_{i \in \{1, \dots, N\}} \rangle$.*

- N : *the number of players. Some EFGs involve a special player called “chance”, which has a fixed stochastic policy that represents the randomness of the environment. For example, the chance player in Kuhn poker is the dealer, who distributes cards to the players at the beginning.*
- \mathbb{A} : *the (finite) set of all agents’ possible actions.*
- \mathbb{H} : *the (finite) set of non-terminal choice nodes.*

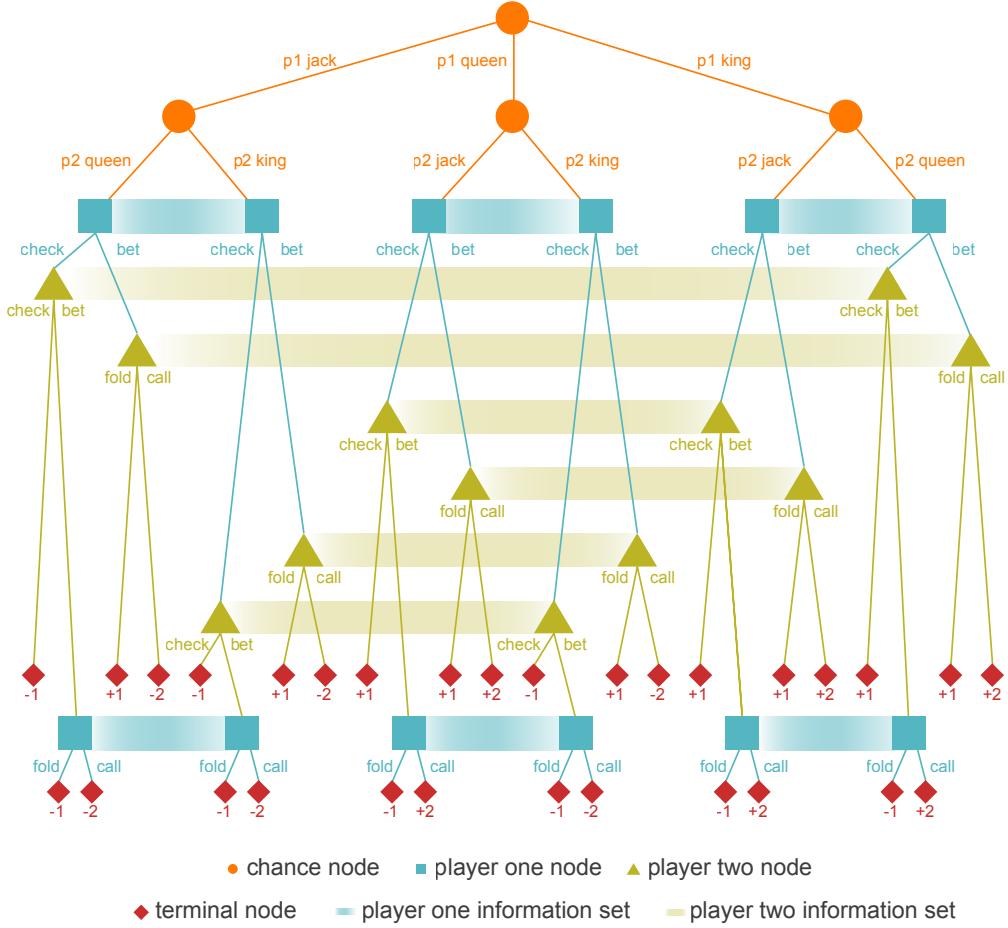


Figure 1.7: Game tree of two-player Kuhn poker. Each node (i.e., circles, squares and rectangles) represents the choice of one player, each edge represents a possible action, and the leaves (i.e., diamond) represent final outcomes over which each player has a reward function (only player one's reward is shown in the graph since Kuhn poker is a zero-sum game). Each player can observe only their own card; for example, when player one holds a Jack, it cannot tell whether player two is holding a Queen or a King, so the choice nodes of player one in each of the two scenarios stay within the same information set.

- \mathbb{T} : the (finite) set of terminal choice nodes, disjoint from \mathbb{H} .
- $\chi : \mathbb{H} \rightarrow 2^{|\mathbb{A}|}$ is the action function that assigns a set of valid actions to each choice node.
- $\rho : \mathbb{H} \rightarrow \{1, \dots, N\}$ is the player indicating function that assigns, to each non-terminal node, a player who is due to choose an action at that node.
- $P : \mathbb{H} \times \mathbb{A} \rightarrow \mathbb{H} \cup \mathbb{T}$ is the transition function that maps a choice node and an action to a new choice/terminal node such that $\forall h_1, h_2 \in \mathbb{H}$ and $\forall a_1, a_2 \in \mathbb{A}$,

if $P(h_1, a_1) = P(h_2, a_2)$, then $h_1 = h_2$ and $a_1 = a_2$.

- $R^i : \mathbb{T} \rightarrow \mathbb{R}$ is a real-valued reward function for player i on the terminal node.

Kuhn poker is a zero-sum game since $R^1 + R^2 = 0$.

- \mathbb{S}^i : a set of equivalence classes/partitions $\mathbb{S}^i = (S_1^i, \dots, S_{k^i}^i)$ for agent i on $\{h \in \mathbb{H} : \rho(h) = i\}$ with the property that $\forall j \in \{1, \dots, k^i\}, \forall h, h' \in S_j^i$, we have $\chi(h) = \chi(h')$ and $\rho(h) = \rho(h')$. The set S_j^i is also called an **information state**. The physical meaning of the information state is that the choice nodes of an information state are indistinguishable. In other words, the set of valid actions and agent identities for the choice nodes within an information state are the same; one can thus use $\chi(S_j^i), \rho(S_j^i)$ to denote $\chi(h), \rho(h), \forall h \in S_j^i$.

Inclusion of the information sets in EFG helps to model the imperfect-information cases in which players have only partial or no knowledge about their opponents. In the case of Kuhn poker, each player can only observe their own card. For example, when player one holds a Jack, it cannot tell whether player two is holding a Queen or a King, so the choice nodes of player one under each of the two scenarios (Queen or King) stay within the same information set. Perfect-information EFGs (e.g., GO or chess) are a special case where the information set is a singleton, i.e., $|S_j^i| = 1, \forall j$, so a choice node can be equated to the unique history that leads to it. Imperfect-information EFGs (e.g., Kuhn poker or Texas hold'em) are those in which there exists i, j such that $|S_j^i| \geq 1$, so the information state can represent more than one possible history. However, with the assumption of perfect recall (described later), the history that leads to an information state is still unique.

1.3.3.1 Normal-Form Representation

A (simultaneous-move) NFG can be equivalently transformed into an imperfect-information EFG¹⁷ (Shoham and Leyton-Brown, 2008) [Chapter 5]. Specifically, since the choices of actions by other agents are unknown to the central agent, this

¹⁷Note that this transformation is not unique, but they share the same equilibria as the original game. Moreover, this transformation from NFG to EFG does not hold for perfect-information EFGs.

could potentially lead to different histories (triggered by other agents) that can be aggregated into one information state for the central agent.

On the other direction, an imperfect-information EFG can also be transformed into an equivalent NFG in which the pure strategies of each agent i are defined by the Cartesian product $\prod_{S_j^i \in \mathbb{S}^i} \chi(S_j^i)$, which is a complete specification¹⁸ of which action to take at every information state of that agent. In the Kuhn poker example, one pure strategy for player one can be check-bet-check-fold-call-fold; altogether, player one has $2^6 = 64$ pure strategies, corresponding to $3 \times 2^3 = 24$ pure strategies for the chance node and $2^6 = 64$ pure strategies for player two. The mixed strategy of each player is then a distribution over all its pure strategies. In this way, the NE in NFG in Eq. (1.17) can still be applied to the EFG, and the NE of an EFG can be solved in two steps: first, convert the EFG into an NFG; second, solve the NE of the induced NFG by means of the Lemke-Howson algorithm (Shapley, 1974). If one further restricts the action space to be state-dependent and adopts the discounted accumulated reward at the terminal node, then the EFG recovers to an SG. While the NE of an EFG can be solved through its equivalent normal form, the computational benefit can be achieved by dealing with the extensive form directly; this motivates the adoption of the sequence-form representation of EFGs.

1.3.3.2 Sequence-Form Representation

Solving EFGs via the NFG representation, though universal, is inefficient because the size of the induced NFG is exponential in the number of information states. In addition, the NFG representation does not consider the temporal structure of games. One way to address these problems is to operate on the sequence form of the EFG, also known as the realisation-plan representation, the size of which is only linear in the number of game states and is thus exponentially smaller than that of the NFG. Importantly, this approach enables polynomial-time solutions to EFGs (Koller and Megiddo, 1992).

In the sequence form of EFGs, the main focus shifts from mixed strategies to

¹⁸One subtlety of the pure strategy is that it designates a decision at each choice node, regardless of whether it is possible to reach that node given the other choice nodes.

behavioural strategies in which, rather than randomising over complete pure strategies, the agents randomise independently at each information state $S^i \in \mathbb{S}^i$, i.e., $\pi^i : \mathbb{S}^i \rightarrow \Delta(\chi(S^i))$. With the help of behavioural strategies, the key insight of the sequence form is that rather than building a player's strategy around the notion of pure strategies that can be exponentially many, one can build the strategy based on the paths in the game tree from the root to each node.

In general, the expressive power of behavioural strategy and mixed strategy are non-comparable. However, if the game has *perfect recall*, which intuitively¹⁹ means that each agent remembers all his historical moves in different information states precisely, then the behavioural strategy and mixed strategy are somehow equivalent. Specifically, suppose all choice nodes in an information state share the same history that led to them (otherwise the agent can distinguish between the choice nodes). In that case, the well-known Kuhn's theorem (Kuhn, 1950a) guarantees that the expressive power of behavioural strategies and that of mixed strategies coincides in the sense that they induce the same probability on outcomes for games of perfect recall. As a result, the set of NE does not change if one considers only behavioural strategies. In fact, the sequence-form representation is primarily useful for describing imperfect-information EFGs of perfect recall, defined as follows.

Definition 9 (Sequence-form Representation). *The sequence-form representation of an imperfect-information EFG, defined in Definition 8, of perfect recall is described by $(N, \Sigma, \{G^i\}_{i \in \{1, \dots, N\}}, \{\pi^i\}_{i \in \{1, \dots, N\}}, \{\mu^{\pi^i}\}_{i \in \{1, \dots, N\}}, \{C^i\}_{i \in \{1, \dots, N\}})$ where*

- N : the number of agents, including the chance node, if any, denoted by c .
- $\Sigma = \prod_{i=1}^N \Sigma^i$: where Σ^i is the set of sequences available to agent i . A sequence of actions of player i , $\sigma^i \in \Sigma^i$, defined by a choice node $h \in \mathbb{H} \cup \mathbb{T}$, is the

¹⁹More formally, on the path from the root node to a decision node $h \in S_t^i$ of player i , list in chronological order which information sets of i were encountered, i.e., $S_t^i \in \mathbb{S}^i$, and what action player i took at that information set, i.e., $a_t^i \in \chi(S_t^i)$. If one calls this list of $(S_0^i, a_0^i, \dots, S_{t-1}^i, a_{t-1}^i, S_t^i)$ the *experience* of player i in reaching node $h \in S_t^i$, then the game has perfect recall if and only if, for all players, any nodes in the same information set have the same experience. In other words, there exists one and only one experience that leads to each information state and the decision nodes in that information state; because of this, all perfect-information EFGs are games of perfect recall.

ordered set of player i 's actions that has been taken from the root to node h . Let \emptyset be the sequence that corresponds to the root node.

Note that other players' actions are not part of agent i 's sequence. In the example of Kuhn poker, $\Sigma^c = \{\emptyset, \text{Jack}, \text{Queen}, \text{King}, \text{Jack-Queen}, \text{Jack-King}, \text{Queen-Jack}, \text{Queen-King}, \text{King-Jack}, \text{King-Queen}\}$, $\Sigma^1 = \{\emptyset, \text{check}, \text{bet}, \text{check-fold}, \text{check-bet}\}$, and $\Sigma^2 = \{\emptyset, \text{check}, \text{bet}, \text{fold}, \text{call}\}$.

- $\pi^i : \mathbb{S}^i \rightarrow \Delta(\chi(S^i))$ is the behavioural policy that assigns a probability of taking a valid action $a^i \in \chi(S^i)$ at an information state $S^i \in \mathbb{S}^i$. This policy randomises independently over different information states. In the example of Kuhn poker, each player has six information states; their behavioural strategy is therefore a list of six independent probability distributions.
- $\mu^{\pi^i} : \Sigma^i \rightarrow [0, 1]$ is the realisation plan that provides the realisation probability, i.e., $\mu^{\pi^i}(\sigma^i) = \prod_{c \in \sigma^i} \pi^i(c)$, that a sequence $\sigma^i \in \Sigma^i$ would arise under a given behavioural policy π^i of player i . In the Kuhn poker case, the realisation probability that player one chooses the sequence of check and then fold is $\mu^{\pi^1}(\text{check-fold}) = \pi^1(\text{check}) \times \pi^1(\text{fold})$.

Based on the realisation plan, one can recover the underlying behavioural strategy²⁰ (an idea similar to Eq. (1.6)). To do so, we need three additional pieces of notation. Let $\text{Seq} : \mathbb{S}^i \rightarrow \Sigma_i$ return the sequence $\sigma^i \in \Sigma^i$ that leads to a given information state $S^i \in \mathbb{S}^i$. Since the game assumes perfect recall, $\text{Seq}(S^i)$ is known to be unique. Let $\sigma^i a^i$ denote a sequence that consists of the sequence σ^i followed by the single action a^i . Since there are many possible actions a^i to choose, let $\text{Ext} : \Sigma^i \rightarrow 2^{\Sigma^i}$ denote the set of all possible sequences that extend the given sequence by taking one additional action. It is trivial to see that sequences that include a terminal node cannot be extended, i.e., $\text{Ext}(T) = \emptyset$. Finally, we can write the behavioural policy π^i for an informa-

²⁰Empirically, it is often the case that working on the realisation plan of a behavioural strategy is more computationally friendly than working on the behavioural strategy directly.

tion state S^i as

$$\pi^i(a^i \in \chi(S^i)) = \frac{\mu^{\pi^i}(\text{Seq}(S^i)a^i)}{\mu^{\pi^i}(\text{Seq}(S^i))}, \quad \forall S^i \in \mathbb{S}^i, \quad \forall (\text{Seq}(S^i)a^i) \in \text{Ext}(\text{Seq}(S^i)). \quad (1.24)$$

- $G^i : \Sigma \rightarrow \mathbb{R}$ is the reward function for agent i given by $G^i(\sigma) = R^i(T)$ if a terminal node $T \in \mathbb{T}$ is reached when each player plays their part of the sequence in $\sigma \in \Sigma$, and $G^i(\sigma) = 0$ if non-terminal nodes are reached.

Note that since each payoff that corresponds to a terminal node is stored only once in the sequence-form representation (due to the perfect recall, each terminal node has only one sequence that leads to it), compared to the normal-form representation, which is a Cartesian product over all information sets for each agent and is thus exponential in size, the sequence form is only linear in the size of the EFG. In the example of Kuhn poker, the normal-form representation is a tensor with $64 \times 64 \times 32$ elements, while in the sequence-form representation, since there are 30 terminal nodes and each node has only one unique sequence leading to it, the payoff tensor has only 30 elements (plus \emptyset for each player).

- C^i : is a set of linear constraints on the realisation probability of μ^{π^i} . Under the notations of Seq and Ext defined in the bullet points of μ^{π^i} , we know the realisation plan must meet the condition that

$$\begin{aligned} \mu^{\pi^i}(\emptyset) &= 1, \quad \mu^{\pi^i}(\sigma^i) \geq 0, \quad \forall \sigma^i \in \Sigma^i \\ \mu^{\pi^i}(\text{Seq}(S^i)) &= \sum_{\sigma^i \in \text{Ext}(\text{Seq}(S^i))} \mu^{\pi^i}(\sigma^i), \quad \forall S^i \in \mathbb{S}^i. \end{aligned} \quad (1.25)$$

The first constraint requires that μ^{π^i} is a proper probability distribution. In addition, the second constraint in Eq. (1.25) indicates that in order for a realisation plan to be valid to recover a behavioural strategy, at each information state of agent i , the probability of reaching that information state must equal the summation of the realisation probabilities of all the extended

sequences. In the example of Kuhn poker, we have C^1 for player one by $\mu^{\pi^1}(\text{check}) = \mu^{\pi^1}(\text{check-fold}) + \mu^{\pi^1}(\text{check-call})$.

1.3.4 Solving Extensive-Form Games

In the sequence-form EFG, given a joint (behavioural) policy $\boldsymbol{\pi} = (\pi^1, \dots, \pi^N)$, we can write the realisation probability of agents reaching a terminal node $T \in \mathbb{T}$, assuming the sequence that leads to the node T is $\boldsymbol{\sigma}_T$, in which each player, including the chance player, follows its own path σ_T^i as

$$\mu^{\boldsymbol{\pi}}(\boldsymbol{\sigma}_T) = \prod_{i \in \{1, \dots, N\}} \mu^{\pi^i}(\sigma_T^i). \quad (1.26)$$

The expected reward for agent i , which covers all possible terminal nodes following the joint policy $\boldsymbol{\pi}$, is thus given by Eq. (1.27).

$$R^i(\boldsymbol{\pi}) = \sum_{T \in \mathbb{T}} \mu^{\boldsymbol{\pi}}(\boldsymbol{\sigma}_T) \cdot G^i(\boldsymbol{\sigma}_T) = \sum_{T \in \mathbb{T}} \mu^{\boldsymbol{\pi}}(\boldsymbol{\sigma}_T) \cdot R^i(T). \quad (1.27)$$

If we denote the expected reward by $R^i(\boldsymbol{\pi})$ for simplicity, then the solution concept of NE for the EFG can be written as

$$R^i(\boldsymbol{\pi}^{i,*}, \boldsymbol{\pi}^{-i,*}) \geq R^i(\boldsymbol{\pi}^i, \boldsymbol{\pi}^{-i,*}), \quad \text{for any policy } \boldsymbol{\pi}^i \text{ of agent } i \text{ and for all } i. \quad (1.28)$$

1.3.4.1 Perfect-Information Games

Every finite perfect-information EFG has a pure-strategy NE ([Zermelo and Borel, 1913](#)). Since players take turns and every agent sees everything that has occurred thus far, it is unnecessary to introduce randomness or mixed strategies into the action selection. However, the NE can be too weak of a solution concept for the EFG. In contrast to that in NFGs, the NE in EFGs can represent *non-credible threats*, which represent the situation where the Nash strategy is not executed as claimed if agents truly reach that decision node. A refinement of the NE in the perfect-information EFG is a *subgame-perfect equilibrium* (SPE). The SPE rules out non-credible threats by picking only the NE that is the best response at every subgame

of the original game.

The fundamental principle in solving the SPE is *backward induction*, which identifies the NE from the bottom-most subgame and assumes those NE will be played as considers increasingly large trees. Specifically, backward induction can be implemented through a depth-first search algorithm on the game tree, which requires time that is only linear in the size of the EFG. In contrast, finding NE in NFG is known to be *PPAD*-hard, let alone the NFG representation is exponential in the size of an EFG.

In the case of two-player zero-sum EFGs, backward induction needs to propagate only a single payoff from the terminal node to the root node in the game tree. Furthermore, due to the strictly opposing interests between players, one can further *prune* the backward induction process by recognising that certain subtrees will never be reached in NE, even without examining those subtree nodes²¹, which leads to the well-known Alpha-Beta-Pruning algorithm ([Shoham and Leyton-Brown, 2008](#), Chapter 5.1). For games with very deep game trees, such as Chess or GO, a common approach is to search only nodes up to certain depths and use an approximate value function to estimate those nodes' value without roll outing to the end ([Silver et al., 2016](#)).

Finally, backward induction can identify one NE in linear time; yet, it does not provide an effective way to find all NE. A theoretical result suggests that finding all NE in a two-player perfect-information EFG (not necessarily zero-sum) requires $\mathcal{O}(|\mathbb{T}|^3)$, which is still tractable ([Shoham and Leyton-Brown, 2008](#), Theorem 5.1.6).

1.3.4.2 Imperfect-Information Games

By means of the sequence-form representation, one can write the solution of a two-player EFG as a LP. Given a fixed behavioural strategy of player two, in the form of realisation plan μ^{π^2} , the best response for player one can be written as

$$\max_{\mu^{\pi^1}} \sum_{\sigma^1 \in \Sigma^1} \mu^{\pi^1}(\sigma^1) \left(\sum_{\sigma^2 \in \Sigma^2} g^1(\sigma^1, \sigma^2) \mu^{\pi^2}(\sigma^2) \right)$$

²¹This occurs, for example, in the case that the worst case of one player in one subgame is better than the best case of that player in another subgame.

subject to the constraints in Eq. (1.25). In NE, player one and player two form a mutual best response. However, if we treat both μ^{π^1} and μ^{π^2} as variables, then the objective becomes nonlinear. The key to address this issue is to adopt the dual form of the LP ([Koller and Megiddo, 1996](#)), which is written as

$$\begin{aligned} & \min v_0 \\ \text{s.t. } & v_{\mathcal{I}(\sigma^1)} - \sum_{I' \in \mathcal{I}(\text{Ext}(\sigma^1))} v_{I'} \geq \sum_{\sigma^2 \in \Sigma^2} g^1(\sigma^1, \sigma^2) \mu^{\pi^2}(\sigma^2), \quad \forall \sigma^1 \in \Sigma^1 \end{aligned} \quad (1.29)$$

where $\mathcal{I} : \Sigma^i \rightarrow \mathbb{S}^i$ is a mapping function that returns the information set²² encountered when the final action in σ^i was taken. With slight abuse of notation, we let $\mathcal{I}(\text{Ext}(\sigma^1))$ ²³ denote the set of final information states encountered in the set of the extension of σ^i . The variable v_0 represents, given μ^{π^2} , player one's expected reward under its own realisation plan μ^{π^1} , and $v_{I'}$ can be considered as the part of this expected utility in the subgame starting from information state I' . Note that the constraint needs to hold for every sequence of player one.

In the dual form of best response in Eq. (1.29), if one treats μ^{π^2} as an optimising variable rather than a constant, which means μ^{π^2} must meet the requirements in Eq. (1.25) to be a proper realisation plan, then the LP formulation for a two-player zero-sum EFG can be written as follows.

$$\min v_0 \quad (1.30)$$

$$\text{s.t. } v_{\mathcal{I}(\sigma^1)} - \sum_{I' \in \mathcal{I}(\text{Ext}(\sigma^1))} v_{I'} \geq \sum_{\sigma^2 \in \Sigma^2} g^1(\sigma^1, \sigma^2) \mu^{\pi^2}(\sigma^2), \quad \forall \sigma^1 \in \Sigma^1 \quad (1.31)$$

$$\mu^{\pi^2}(\emptyset) = 1, \quad \mu^{\pi^2}(\sigma^2) \geq 0, \quad \forall \sigma^2 \in \Sigma^2 \quad (1.32)$$

$$\mu^{\pi^2}(\text{Seq}(S^2)) = \sum_{\sigma^2 \in \text{Ext}(\text{Seq}(S^2))} \mu^{\pi^2}(\sigma^2), \quad \forall S^2 \in \mathbb{S}^2. \quad (1.33)$$

Player two's realisation plan is now selected to minimise player one's expected utility. Based on the minimax theorem ([Von Neumann and Morgenstern, 1945](#)), we

²²Recall that this information set is unique under the assumption of perfect recall.

²³Recall that $\text{Ext}(\sigma^1)$ is the set of all possible sequences that extend σ^1 one step ahead.

know this process will lead to a NE. Notably, though the zero-sum EFG and zero-sum SG (see the formulation in Eq. (2.17)) both adopt the LP formulation to solve the NE and can be solved in polynomial time, the size of the representation for the game itself is very different. If one chooses first to transform the EFG into an NFG presentation and then solve it by LP, then the time complexity would in fact become exponential in the size of the original EFG.

The solution to a two-player general-sum EFG can also be formulated using an approach similar to that used for the zero-sum EFG. The difference is that there will be no objective function such as Eq. (1.30) since in the general-sum context, one agent's reward can no longer be determined based on the other player's reward. The LP with only Eqs. (1.31 - 1.33) thus becomes a constraint satisfaction problem. Specifically, one would need to repeat Eqs. (1.31 - 1.33) twice to consider each player independently. One final subtlety required in solving the two-player general-sum EFG is that to ensure v^1 and v^2 are bounded²⁴, a *complementary slack condition* must be further imposed; we have $\forall \sigma^1 \in \Sigma^1$ (vice versa $\forall \sigma^2 \in \Sigma^2$ for player two):

$$\mu^{\pi^1}(\sigma^1) \left[\left(v_{\mathcal{I}(\sigma^1)}^1 - \sum_{I' \in \mathcal{I}(\text{Ext}(\sigma^1))} v_{I'}^1 \right) - \left(\sum_{\sigma^2 \in \Sigma^2} g^1(\sigma^1, \sigma^2) \mu^{\pi^2}(\sigma^2) \right) \right] = 0. \quad (1.34)$$

The above condition indicates that for each player, either the sequence σ^i is never played, i.e., $\mu^{\pi^i}(\sigma^i) = 0$, or all sequences that are played by that player with positive probability must induce the same expected payoff such that v^i takes arbitrarily large values, thus being bounded. Eqs. (1.31 - 1.33), together with Eq. (1.34), turns the solution to the NE into an LCP problem that can be solved by the generalised Lemke-Howson method (Lemke and Howson, 1964). Although in the worst case, polynomial time complexity cannot be achieved, as can for zero-sum games, this approach is still exponentially faster than running the Lemke-Howson method to solve the NE in a normal-form representation.

For a perfect-information EFG, recall that the SPE is a more informative so-

²⁴Since the constraints are linear, they remain satisfied when both v^1 and v^2 are increased by the same constant to any arbitrarily large values.

lution concept than NE. Extending SPE to the imperfect-information scenario is therefore valuable. However, such an extension is non-trivial because a well-defined notion of a subgame is lacking. However, for EFGs with perfect recall, the intuition of subgame perfection can be effectively extended to a new solution concept, named the sequential equilibrium (SE) (Kreps and Wilson, 1982), which is guaranteed to exist and coincides with the SPE if all players in the game have perfect information.

1.4 Grand Challenges of MARL

Compared to single-agent RL, multi-agent RL is a general framework that better matches the broad scope of real-world AI applications. However, due to the existence of multiple agents that learn simultaneously, MARL methods pose more theoretical challenges, in addition to those already present in single-agent RL. Compared to classic MARL settings where there are usually two agents, solving a many-agent RL problem is even more challenging. As a matter of fact, ① **the combinatorial complexity**, ② **the multi-dimensional learning objectives**, and ③ **the issue of non-stationarity** all result in the majority of MARL algorithms being capable of solving games with only two players, in particular, two-player zero-sum games. In this chapter, I will elaborate each of the grand challenge in many-agent RL.

1.4.1 The Combinatorial Complexity

In the context of multi-agent learning, each agent has to consider the other opponents' actions when determining the best response; this characteristic is deeply rooted in each agent's reward function and for example is represented by the joint action \mathbf{a} in their Q-function $Q^i(s, \mathbf{a})$ in Eq. (1.13). The size of the joint action space, $|\mathbb{A}|^N$, grows exponentially with the number of agents and thus largely constrains the scalability of MARL methods. Furthermore, the combinatorial complexity is worsened by the fact that solving a NE in game theory is *PPAD*-hard, even for two-player games. Therefore, for multi-player general-sum games (neither team games nor zero-sum games), it is non-trivial to find an applicable solution concept.

One common way to address this issue is by assuming specific factorised structures on action dependency such that the reward function or Q-function can be sig-

nificantly simplified. For example, a graphical game assumes an agent’s reward is affected by only its neighbouring agents, as defined by the graph from (Kearns, 2007). This assumption leads directly to a polynomial-time solution for the computation of a NE in specific tree graphs (Kearns et al., 2013), though the scope of applications is somewhat limited beyond this specific scenario.

Recent progress has also been made toward leveraging particular neural network architectures for Q-function decomposition (Rashid et al., 2018; Sunehag et al., 2018; Yang et al., 2020). In addition to the fact that these methods work only for the team-game setting, the majority of them lack theoretical backing. There remain open questions to answer, such as understanding the representational power (the approximation error) of the factorised Q-functions in a multi-agent task and how factorisation itself can be learnt from scratch.

1.4.2 The Multi-Dimensional Learning Objectives

Compared to single-agent RL, where the only goal is to maximise the learning agent’s long-term reward, the learning goals in MARL are naturally multi-dimensional, as the objective of all agents are not necessarily aligned by one metric. Bowling and Veloso (2001b, 2002) proposed to classify the goals of the learning task into two types: **rationality** and **convergence**. Rationality ensures an agent takes the best possible response to the opponents when they are stationary, and convergence ensures the learning dynamics eventually lead to a stable policy against a given class of opponents. Reaching both rationality and convergence gives rise to reaching the NE.

In terms of rationality, the NE characterises a fixed point of a joint optimal strategy profile from which no agents would be motivated to deviate as long as they are all perfectly rational. However, in practice, an agent’s rationality can easily be bound by either cognitive limitations and/or the tractability of the decision problem. In these scenarios, the rationality assumption can be relaxed to include other types of solution concepts, such as the recursive reasoning equilibrium, which results from modelling the reasoning process recursively among agents with finite levels of hierarchical thinking (for example, an agent may reason in the follow-

ing way: I believe that you believe that I believe ...) (Wen et al., 2019b, 2018); best response against a target type of opponent (Powers and Shoham, 2005b); the mean-field game equilibrium, which describes multi-agent interactions as a two-agent interaction between each agent itself and the population mean (Guo et al., 2019; Yang et al., 2018a,b); evolutionary stable strategies, which describe an equilibrium strategy based on its evolutionary advantage of resisting invasion by rare emerging mutant strategies (Bloembergen et al., 2015; Maynard Smith, 1972; Tuyls and Nowé, 2005; Tuyls and Parsons, 2007); Stackelberg equilibrium (Zhang et al., 2019a), which assumes specific sequential order when agents take decisions; and the robust equilibrium (also called the trembling-hand perfect equilibrium in game theory), which is stable against adversarial disturbance (Goodfellow et al., 2014b; Li et al., 2019b; Yabu et al., 2007).

In terms of convergence, although most MARL algorithms are contrived to converge to the NE, the majority either lack a rigorous convergence guarantee (Zhang et al., 2019c), potentially converge only under strong assumptions such as the existence of a unique NE (Hu and Wellman, 2003; Littman, 2001b), or are provably non-convergent in all cases (Mazumdar et al., 2019a). Zinkevich et al. (2006) identified the non-convergent behaviour of value-iteration methods in general-sum SGs and instead proposed an alternative solution concept to the NE – *cyclic equilibria* – that value-based methods converge to. The concept of no regret (also called the Hannan consistency in game theory (Hansen et al., 2003)), measures convergence by comparison against the best possible strategy in hindsight. This was also proposed as a new criterion to evaluate convergence in zero-sum self-plays (Bowling, 2005; Hart and Mas-Colell, 2001; Zinkevich et al., 2008). In two-player zero-sum games with a non-convex non-concave loss landscape (training GANs (Goodfellow et al., 2014a)), gradient-descent-ascent methods are found to reach a Stackelberg equilibrium (Fiez et al., 2019; Lin et al., 2019) or a local differential NE (Mazumdar et al., 2019b) rather than the general NE.

Finally, although the above solution concepts account for convergence, building a convergent objective for MARL methods with DNNs remains an uncharted

area. This is partly because the global convergence result of a single-agent deep RL algorithm, for example, neural policy gradient methods (Liu et al., 2019a; Wang et al., 2019) and neural TD learning algorithms (Cai et al., 2019b), has not been extensively studied yet.

1.4.3 The Non-Stationarity Issue

The most well-known challenge of multi-agent learning versus single-agent learning is probably the non-stationarity issue. Since multiple agents concurrently improve their policies according to their own interests, from each agent’s perspective, the environmental dynamics become non-stationary and challenging to interpret when learning. This problem occurs because the agent itself cannot tell whether the state transition – or the change in reward – is an actual outcome due to its own action or if it is due to its opponent’s explorations. Although learning independently by completely ignoring the other agents can sometimes yield surprisingly powerful empirical performance (Matignon et al., 2012; Papoudakis et al., 2020), this approach essentially harms the stationarity assumption that supports the theoretical convergence guarantee of single-agent learning methods (Tan, 1993). As a result, the Markovian property of the environment is lost, and the state occupancy measure of the stationary policy in Eq. (1.5) no longer exists. For example, the convergence result of single-agent policy gradient methods in MARL is provably non-convergent in simple linear-quadratic games (Mazumdar et al., 2019b).

The non-stationarity issue can be further aggravated by TD learning, which occurs with the replay buffer that most deep RL methods currently adopt (Foerster et al., 2017b). In single-agent TD learning (see Eq. (1.9)), the agent bootstraps the current estimate of the TD error, saves it in the replay buffer, and samples the data in the replay buffer to update the value function. In the context of multi-agent learning, since the value function for one agent also depends on other agents’ actions, the bootstrap process in TD learning also requires sampling other agents’ actions, which leads to two problems. First, the sampled actions barely represent the full behaviour of other agents’ underlying policies across different states. Second, an agent’s policy can change during training, so the samples in the replay buffer

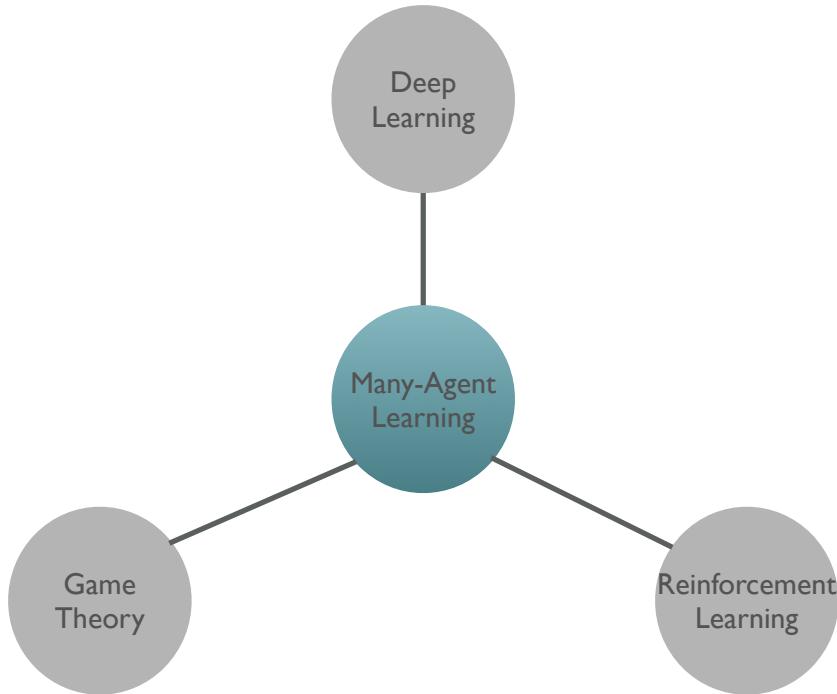


Figure 1.8: The scope of the research in this thesis consists of three pillars. Deep learning serves as a powerful function approximation tool for the learning process. Game theory provides an effective approach to describe learning outcomes. RL offers a valid approach to describe agents' incentives in multi-agent systems.

can quickly become outdated. Therefore, the dynamics that yielded the data in the agent's replay buffer must be constantly updated to reflect the current dynamics in which it is learning. This process further exacerbates the non-stationarity issue.

In general, the non-stationarity issue forbids the reuse of the same mathematical tool for analysing single-agent algorithms in the multi-agent context. However, one exception exists: the identical-interest game in Definition 4. In such settings, each agent can safely perform selfishly without considering other agents' policies since the agent knows the other agents will also act in their own interest. The stationarity is thus maintained, so single-agent RL algorithms can still be applied.

1.5 Thesis Structure and Contributions

This thesis focuses on the research topic of many-agent reinforcement learning. The methods that I contribute towards this topic lie in the scope of the three research domains listed in Figure 1.8; they are game theory, which provides realistic and tractable solution concepts to describe learning outcomes of a many-agent system;

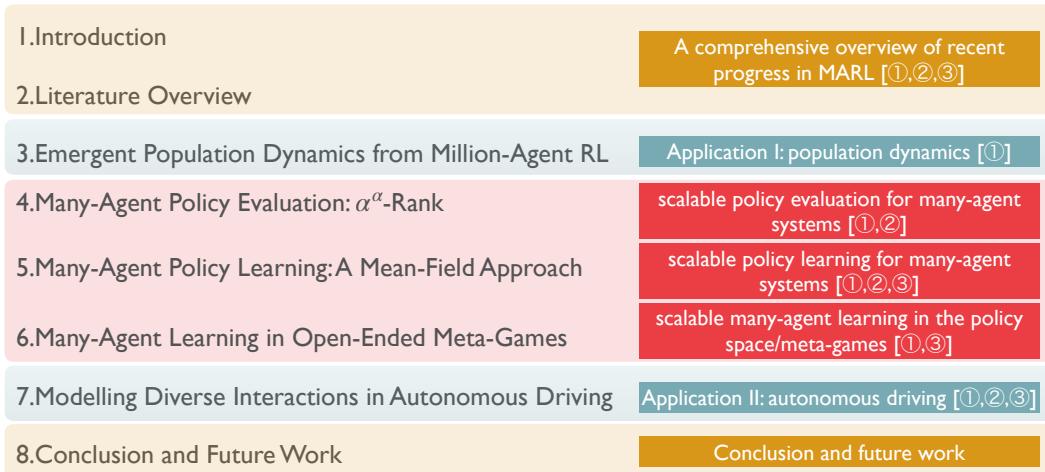


Figure 1.9: The structure of following chapters in this thesis, associated with the three listed challenges in Chapter 1.4 that each chapter tries to address.

RL algorithms, which offer provably convergent learning algorithms that can reach stable and rational equilibria in the sequential decision-making process; and finally deep learning techniques, which provide the learning algorithms expressive function approximators.

The structure and contribution of each following chapter are as follows (also see Figure 1.9):

- **Chapter 2:** Since the scalability issue of MARL is deeply rooted in its game-theoretical foundations, in this chapter, I will first conduct an overview that elaborates the game theory side of modern MARL methods, along with the recent advances. I believe this overview is an important contribution to the community since most existing surveys are either not game theory focused or outdated in terms of missing most of the recent literature since 2010. Chapter 1 together with Chapter 2 form a self-contained monograph of MARL. The goal of the monograph is to provide a dedicated assessment of the current state-of-the-art MARL techniques from a game theoretical perspective. I expect this work to serve as a stepping stone for both new researchers who are about to enter this fast-growing domain and existing domain experts who want to obtain a panoramic view and identify new directions based on recent advances.
- **Chapter 3:** This chapter offers an application of MARL technique on under-

standing the emergent population dynamics of AI agents. The goal of this chapter is to serve as an appetiser to show the great potentials of MARL methods before I introduce methodological developments. Specifically, in this work, I put RL agents into a simulated predator-prey world and verify if the principles developed in nature could be used in understanding an artificially-created intelligent population, and vice versa. This work’s key contribution is that it has inspired many population biologists and computational biologists by offering them a new methodology based on MARL approaches when modelling self-interested agents in microbiology studies.

- **Chapter 4:** This chapter introduces a new policy evaluation method for many-agent systems: α^α -Rank. α^α -Rank is a stochastic variation of α -rank, a novel solution concept that is unique and has polynomial-time solution in multi-player general-sum games. A key benefit of α^α -Rank is that one can now tractably evaluate a large-scale multi-agent system (i.e., multi-player general-sum games), for example, a multi-agent system with $\mathcal{O}(2^{25})$ joint strategies profiles by just one single machine; this is in contrast to computing Nash equilibrium, which is known to be *PPAD*-hard in even two-player cases.
- **Chapter 5:** In this chapter, I focus on tackling the core problem of policy learning in many-agent systems. Specifically, I propose mean-field MARL (MF-MARL) methods, which leverage the classic idea of mean-field approximation from physics. MF-MARL effectively transforms a many-agent learning problem into a two-agent problem by considering only the mean effect from the population. With MF-MARL methods, one can efficiently train millions of agents in solving a large-scale cooperative game. I test the MF-MARL algorithms on solving the Ising model, a notoriously hard problem in physics due to its combinatorial nature, and report the first Ising model solution based on MARL. Overall, the main contribution of this chapter is to provide the first provably convergent scalable MARL algorithms, and demonstrate its effectiveness in scenarios where there are far more than two agents.

- **Chapter 6:** This chapter looks into the problem of many-agent learning in open-ended meta-games (i.e., games at the policy level, also known as league training or auto-curricula) where behavioural diversity is a critical yet under-explored topic. This chapter contributes the first mathematically rigorous definition for behavioural diversity in the policy space, and proposes learning algorithms that are proved to enlarge diversity during policy training. Empirical results on zero-sum games suggest that the proposed methods outperform the existing state-of-the-art by a large margin. This study could have a substantial economic impact because the proposed algorithms can be directly plugged into the league training of developing game AIs (e.g., training a population of AIs that can beat human players on Poker games).
- **Chapter 7:** In addition to Chapter 3, this chapter introduces the second application of MARL, which is about autonomous driving (AD). I demonstrate the great potentials of using MARL techniques to model realistic and diverse multi-agent interactions in AD. Specifically, I present the SMARTS platform: the first AD simulator that dedicatedly supports RL and MARL training. Based on SMART, I share a blue sky idea that creating diverse auto-curricula in MARL is the key to modelling realistic interactions in AD. I elaborate on the necessity of diverse auto-curricula and list four open challenges of applying this technique. Contributions of this chapter are two-fold: first, I show that MARL techniques could have impactful applications in the real physical world, outside of purely video games; second, I introduce a new methodology to the researchers in AD so that they can generate high-quality interactions that are currently missing.
- **Chapter 8:** In this last chapter, I summarise this thesis, and propose four future research directions to work on; they are deep MARL theory, safe and robust MARL, model-based MARL, and multi-agent meta-RL.

1.6 Related Publications

This thesis is established on multiple published works. I enumerate relevant references in the following list. Necessary demarcations have been made when pre-

senting the joint work in which other collaborators are involved. For those published works that are not present as main chapters in this thesis, I briefly review them as related work in corresponding sections. The full publication list can be found at <https://scholar.google.co.uk/citations?user=6yL0xw8AAAAJ>.

- [1] **Yang Y.** and Wang J., 2020. [An Overview of Multi-Agent Reinforcement Learning from Game Theoretical Perspective](#). *arXiv preprint arXiv:2011.00583*. [Main body of Chapter 1 and Chapter 2]
- [2] **Yang Y.**, Yu L., Bai Y., Wang J., Zhang W., Wen Y. and Yu Y., 2018. [A Study of AI Population Dynamics with Million-agent Reinforcement Learning](#). In *Proceedings of the 17th Conference on Autonomous Agents and Multi-Agent Systems (AAMAS 2018)*. [Main body of Chapter 3]
- [3] **Yang Y.**, Tutunov R., Sakulwongtana P., Ammar HB., and Wang J., 2020. [\$\alpha^\alpha\$ -Rank: Practically Scaling \$\alpha\$ -Rank through Stochastic Optimisation](#). In *Proceedings of the 19th International Conference on Autonomous Agents and Multi-Agent Systems (AAMAS 2020)*. [Main body of Chapter 4]
- [4] **Yang Y.**, Luo R., Li M., Zhou M., Zhang W., and Wang J., 2018. [Mean Field Multi-agent Reinforcement Learning](#). In *Proceedings of the 35th International Conference on Machine Learning (ICML 2018, Long Talk (top 3%))*. [Main body of Chapter 5]
- [5] **Yang Y.**, Nieves NP., Slumbers O., Mguni D., and Wang J., 2021. [Modelling Behavioural Diversity for Open-Ended Learning in Games](#). In *Proceedings of the 38th International Conference on Machine Learning (ICML 2021, Long Talk (top 3%))*. [Main body of Chapter 6]
- [6] **Yang Y.**, Taylor M., Luo J., Wen Y., Slumbers O., Graves D., Ammar HB., and Wang J., 2020. [Diverse Auto-Curriculum is Critical for Successful Real-World Multi-agent Learning Systems](#). In *Proceedings of the 20th Interna-*

tional Conference on Autonomous Agents and Multi-Agent Systems (AAMAS 2021, *Best Paper Award in Blue Sky Ideas*). [Main body of Chapter 7]

- [7] Zhou M., Luo J., Villela J., **Yang Y.**, Rusu D., Miao J., Zhang W., Alban M., Fadakar I., Chen Z. and Huang A.C., 2020. SMARTS: Scalable Multi-Agent Reinforcement Learning Training School for Autonomous Driving. In *Proceedings of the 4th Conference on Robotic Learning (CoRL 2020, Best System Paper Award)*. [Part of Chapter 7]

Works that are related to this thesis topic but NOT incorporated as main chapters are listed as follows:

- [8] Zhang H., Chen W., Huang Z., Li M., **Yang Y.**, Zhang W. and Wang J., 2020, April. Bi-level Actor-Critic for Multi-agent Coordination. In *Proceedings of the 34th AAAI Conference on Artificial Intelligence (AAAI 2020)*. [Reviewed as related work in Chapter 1.4.2]
- [9] **Yang Y.**, Wen Y., Chen L., Wang J., Shao K., David M., and Zhang W. Multi-agent Determinantal Q-learning. In *Proceedings of the 37th International Conference on Machine Learning (ICML 2020)*. [Reviewed as related work in Chapter 2.2.1.1]
- [10] Zhou M., Chen Y., Wen Y., **Yang Y.**, Su Y., Zhang W., Zhang D. and Wang J., 2019, October. Factorised Q-learning for large-scale multi-agent systems. In *Proceedings of the 1st International Conference on Distributed Artificial Intelligence (DAI 2019)*. [Reviewed as related work in Chapter 2.2.1.1]
- [11] Wen Y., **Yang Y.**, Luo R., Pan W., and Wang J., Probabilistic Recursive Reasoning for Multi-agent Reinforcement Learning. In *Proceedings of the 7th International Conference on Learning Representations (ICLR 2019)*. [Reviewed as related work in Chapter 2.2.1.2]
- [12] Wen Y., **Yang Y.**, and Wang J. Modelling Bounded Rationality in Multi-Agent Interactions by Generalised Recursive Reasoning. In *Proceedings of the*

29th International Joint Conference on Artificial Intelligence (IJCAI 2020).

[Reviewed as related work in Chapter 2.2.1.2]

- [13] Li M., Qin Z., Jiao Y., **Yang Y.**, Wang J., Wang C., Wu G., and Ye J. [Efficient Ridesharing Order Dispatching with Mean Field Multi-Agent Reinforcement Learning](#). In *Proceedings of The Web Conference (WWW 2019)*. [Reviewed as related work in Chapter 2.5.3]
- [14] Peng P., Wen Y., **Yang Y.**, Yuan Q., Tang Z., Long H. and Wang J., 2017. [Multi-agent Bidirectionally-Coordinated Nets: Emergence of Human-level Coordination in Learning to Play StarCraft Combat Games](#). *The 31st Annual Conference on Neural Information Processing Systems (NeurIPS 2017, workshop of emergent communication)*. [Reviewed as related work in Chapter 7.4.3]

Chapter 2

Literature Overview

Following a comprehensive introduction of MARL in Chapter 1, this chapter will present an overview of recent advances of MARL developments, with a major focus from the game-theoretical perspective. Since this chapter is itself a survey, I first conduct a survey of existing survey papers, and introduce new taxonomies for MARL methods. In later sections, I highlight several current topics in MARL research categorised by game types; these topics include Q-function factorisation, multi-agent soft learning, networked multi-agent MDP, stochastic potential games, zero-sum continuous games, online MDP, turn-based stochastic games, policy space response oracle, approximation methods in general-sum games, and mean-field type learning in games with infinite agents. Within each topic, I select both the most fundamental and cutting-edge algorithms. This chapter aims to provide an assessment of the current state-of-the-art MARL techniques, with a much deeper scope. I hope this work could serve as a stepping stone for both new researchers who are about to enter this fast-growing domain and existing domain experts who want to obtain a panoramic view and identify new directions based on recent advances.

2.1 A Survey of Surveys on Multi-Agent RL

In this section, I provide a non-comprehensive review of MARL algorithms. To begin, I introduce different taxonomies that can be applied to categorise prior approaches. Given multiple high-quality, comprehensive surveys on MARL methods already exist, a survey of those surveys is provided. Based on the proposed tax-

onomy, I review related MARL algorithms, covering works on identical interest games, zero-sum games, and games with an infinite number of players. This section is written to be selective, focusing on the algorithms that have theoretical guarantees and less focus on those with only empirical success or those that are purely driven by specific applications.

2.1.1 Taxonomy of Multi-Agent RL Algorithms

One significant difference between the taxonomy of single-agent RL algorithms and MARL algorithms is that in the single-agent setting, since the problem is unanimously defined, the taxonomy is driven mainly by the type of solution (Kaelbling et al., 1996; Li, 2017), for example, model-free vs model-based, on-policy vs off-policy, TD learning vs Monte-Carlo methods. By contrast, in the multi-agent setting, due to the existence of multiple learning objectives (see Section 1.4.2), the taxonomy is driven mainly by the type of problem rather than the solution. In fact, asking the right question for MARL algorithms is itself a research problem, which is referred to as the problem problem (Balduzzi et al., 2018b; Shoham et al., 2007).

2.1.1.1 Based on Stage Games Types.

Since the solution concept varies considerably according to the game type, one principal component of the MARL taxonomy is the nature of stage games. A common division¹ includes team games (more generally, potential games), zero-sum games (more generally, harmonic games), and a mixed setting of the two games, namely, general-sum games. Other types of “exotic” games, such as potential games (Monderer and Shapley, 1996) and mean-field games (Lasry and Lions, 2007), that originate from non-game-theoretical research domains exist and have recently attracted tremendous attention. Based on the type of stage game, the taxonomy can be further enriched by how many times they are played. A repeated game is where one stage game is played repeatedly without considering the state transition. An SG is a sequence of stage games, which can be infinitely long, with the order of the games to

¹Such a division is complementary because any multi-player normal-form game can be decomposed into a potential game plus a harmonic game (Candogan et al., 2011) (also see Definition 4); in the two-player case, it corresponds to a team game plus a zero-sum game.

Table 2.1: Common assumptions on the level of local knowledge made by MARL algorithms.

LEVELS	ASSUMPTIONS
0	EACH AGENT OBSERVES THE REWARD OF HIS SELECTED ACTION.
1	EACH AGENT OBSERVES THE REWARDS OF ALL POSSIBLE ACTIONS.
2	EACH AGENT OBSERVES OTHERS' SELECTED ACTIONS.
3	EACH AGENT OBSERVES OTHERS' REWARD VALUES.
4	EACH AGENT KNOWS OTHERS' EXACT POLICIES.
5	EACH AGENT KNOWS OTHERS' EXACT REWARD FUNCTIONS.
6	EACH AGENT KNOWS THE EQUILIBRIUM OF THE STAGE GAME.

play determined by the state-transition probability. Since solving a general-sum SG is at least *PSPACE*-hard (Conitzer and Sandholm, 2002), MARL algorithms usually have a clear boundary on what types of game they can solve. For general-sum games, there are few MARL algorithms that have a provable convergence guarantee without strong, even unrealistic, assumptions (e.g., the NE is unique) (Shoham et al., 2007; Zhang et al., 2019c).

2.1.1.2 Based on Level of Local Knowledge.

The assumption on the level of local knowledge, i.e., what agents can and cannot know during training and execution time, is another major component to differentiate MARL algorithms. Having access to different levels of local knowledge leads to different local behaviours by agents and various levels of difficulty in developing theoretical analysis. I list the common assumptions that most MARL methods adopt in Table (2.1). The seven levels of assumptions are ranked based on how strong, or unrealistic, they are in general. The two extreme cases are that the agent can observe nothing apart from itself and that the agent knows the equilibrium point, i.e., the direct answer of the game. Among the multiple levels, the nuance between level 0 and level 1, which has been mainly investigated in the online learning literature, is referred to as the *bandit* setting vs *full-information* setting. In addition, knowledge of the agents' exact policy/reward function forms is a much stronger assumption than being able to observe their sampled actions/rewards. In fact, knowing the exact policy parameters of other agents in most cases are only possible in simulations.

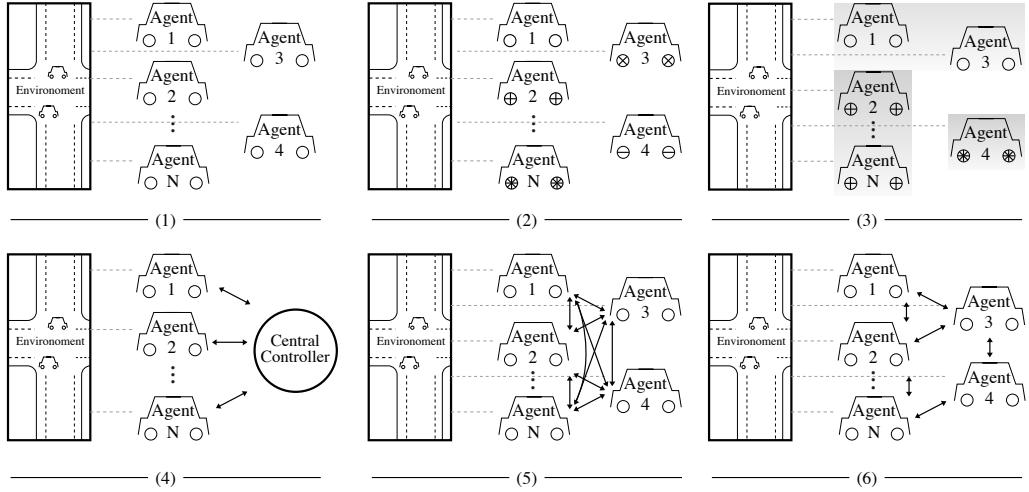


Figure 2.1: Common learning paradigms of MARL algorithms. (1) Independent learners with shared policy. (2) Independent learners with independent policies (i.e., denoted by the difference in wheels). (3) Independent learners with shared policy within a group. (4) One central controller controls all agents: agents can exchange information with any other agents at any time. (5) Centralised training with decentralised execution (CTDE): only during training, agents can exchange information with others; during execution, they act independently. (6) Decentralised training with networked agents: during training, agents can exchange information with their neighbours in the network; during execution, they act independently.

Furthermore, from an applicability perspective, observing other agents' rewards is also more unrealistic than observing their actions.

2.1.1.3 Based on Learning Paradigms.

In addition to various levels of local knowledge, MARL algorithms can be classified based on the learning paradigm, as shown in Figure 2.1. For example, the 4th learning paradigm addresses multi-agent problems by building a single-agent controller, which takes the joint information from all agents as inputs and outputs the joint policies for all agents. In this paradigm, agents can exchange any information with any other opponent through the central controller. The information that can be exchanged depends on the assumptions about the level of local knowledge described in Table (2.1), e.g., private observations from each agent, the reward value, or policy parameters for each agent. The 5th learning paradigm allows agents to exchange information with other agents only during training; during execution, each

agent has to act in a decentralised manner, making decisions based on its own observations only. The 6th paradigm can be regarded as a particular case of Paradigm 5 in that agents are assumed to be interconnected via a (time-varying) network such that information can still spread across the whole network if agents communicate with their neighbours. The most general case is Paradigm 2, where agents are fully decentralised, with no information exchange of any kind allowed at any time, and each agent executes its own policy. Relaxation of Paradigm 2 yields the 1st and the 3rd paradigms, where the agents, although they cannot exchange information, share a single set of policy parameters, or, within a pre-defined group, share a single set of policy parameters.

2.1.1.4 Based on Five AI Agendas.

In order for MARL researchers to be specific about the problem being addressed and the associated evaluation criteria, [Shoham et al. \(2007\)](#) identified five coherent agendas for MARL studies, each of which has a clear motivation and success criterion. Though proposed more than a decade ago, these five distinct goals are still useful in evaluating and categorising recent contributions. I, therefore, choose to incorporate them into the taxonomy of MARL algorithms.

2.1.2 A Survey of Surveys

A multi-agent system (MAS) is a generic concept that could refer to many different domains of research across different academic subjects; general overviews are given by [Weiss \(1999\)](#), [Wooldridge \(2009\)](#), and [Shoham and Leyton-Brown \(2008\)](#). Due to the many possible ways of categorising multi-agent (reinforcement) learning algorithms, it is impossible to have a single survey that includes all relevant works considering all directions of categorisations. In the past two decades, there has been no lack of survey papers that summarise the current progress of specific categories of multi-agent learning research. In fact, there are so many that these surveys themselves deserve a comprehensive review. Before proceeding to review MARL algorithms based on the proposed taxonomy in Section 2.1.1, in this section, I provide an overview of relevant surveys that study multi-agent systems from the

Table 2.2: Summary of the five agendas for multi-agent learning research [Shoham et al. \(2007\)](#).

ID	AGENDA	DESCRIPTION
1	COMPUTATIONAL	TO DEVELOP EFFICIENT METHODS THAT CAN COMPUTE SOLUTION CONCEPTS OF THE GAME. EXAMPLES: BERGER (2007) ; LEYTON-BROWN AND TENNENHOLTZ (2005)
2	DESCRIPTIVE	TO DEVELOP FORMAL MODELS OF LEARNING THAT AGREE WITH THE BEHAVIOURS OF PEOPLE/ANIMALS/ORGANISATIONS. EXAMPLES: CAMERER ET AL. (2002) ; EREV AND ROTH (1998)
3	NORMATIVE	TO DETERMINE WHICH SETS OF LEARNING RULES ARE IN EQUILIBRIUM WITH EACH OTHER. FOR EXAMPLE, WE CAN ASK IF FICTITIOUS PLAY AND Q-LEARNING CAN REACH EQUILIBRIUM WITH EACH OTHER IN A REPEATED PRISONER'S DILEMMA GAME.
4	PRESCRIPTIVE, CO-OPERATIVE	TO DEVELOP DISTRIBUTED LEARNING ALGORITHMS FOR TEAM GAMES. IN THIS AGENDA, THERE IS RARELY A ROLE FOR EQUILIBRIUM ANALYSIS SINCE THE AGENTS HAVE NO MOTIVATION TO DEVIATE FROM THE PRESCRIBED ALGORITHM. EXAMPLES: CLAUS AND BOUTILIER (1998A)
5	PRESCRIPTIVE, NON-COOPERATIVE	TO DEVELOP EFFECTIVE METHODS FOR OBTAINING A “HIGH REWARD” IN A GIVEN ENVIRONMENT, FOR EXAMPLE, AN ENVIRONMENT WITH A SELECTED CLASS OF OPPONENTS. EXAMPLES: POWERS AND SHOHAM (2005A,B)

machine learning, in particular, the RL, perspective.

One of the earliest studies that surveyed MASs in the context of machine learning/AI was published by [Stone and Veloso \(2000\)](#): the research works up to that time were summarised into four major scenarios considering whether agents were homogeneous or heterogeneous and whether or not agents were allowed to communicate with each other. [Shoham et al. \(2007\)](#) considered the game theory and RL perspective and introspectively asked the question of “if multi-agent learning is the

answer, what is the question?”. Upon failing to find a single answer, Shoham et al. (2007) proposed the famous five AI agendas for future research work to address. Stone (2007) tried to answer Shoham’s question by emphasising that MARL can be more broadly framed than through game theoretic terms, and he noted that how to apply the MARL technique remains an open question, rather than being an answer, in contrast to the suggestion of Shoham et al. (2007). The survey of Tuyls and Weiss (2012) also reflected on Stone’s viewpoint; they believed that the entanglement of only RL and game theory is too narrow in its conceptual scope, and MARL should embrace other ideas, such as transfer learning (Taylor and Stone, 2009), swarm intelligence (Kennedy, 2006), and co-evolution (Tuyls and Parsons, 2007).

Panait and Luke (2005) investigated the cooperative MARL setting; instead of considering only reinforcement learners, they reviewed learning algorithms based on the division of *team learning* (i.e., applying a single learner to search for the optimal joint behaviour for the whole team) and *concurrent learning* (i.e., applying one learner per agent), which includes broader areas of evolutionary computation, complex systems, etc. Matignon et al. (2012) surveyed the solutions for fully-cooperative games only; in particular, they focused on evaluating independent RL solutions powered by Q-learning and its many variants. Jan’t Hoen et al. (2005) conducted an overview with a similar scope; moreover, they extended the work to include fully competitive games in addition to fully cooperative games. Buşoniu et al. (2010), to the best of my knowledge, presented the first comprehensive survey on MARL techniques, covering both value iteration-based and policy search-based methods, together with their strengths and weaknesses. In their survey, they considered not only fully cooperative or competitive games but also the effectiveness of different algorithms in the general-sum setting. Nowé et al. (2012), in the 14th chapter, addressed the same topic as Buşoniu et al. (2010) but with a much narrower coverage of multi-agent RL algorithms.

Tuyls and Nowé (2005) and Bloembergen et al. (2015) both surveyed the dynamic models that have been derived for various MARL algorithms and revealed the deep connection between evolutionary game theory and MARL methods. We

refer to Table 1 in [Tuyls and Nowé \(2005\)](#) for a summary of this connection.

[Hernandez-Leal et al. \(2017\)](#) provided a different perspective on the taxonomy of how existing MARL algorithms cope with the issue of non-stationarity induced by opponents. On the basis of the opponent and environment characteristics, they categorised the MARL algorithms according to the type of opponent modelling.

[Da Silva and Costa \(2019\)](#) introduced a new perspective of reviewing MARL algorithms based on how knowledge is reused, i.e., transfer learning. Specifically, they grouped the surveyed algorithms into *intra-agent* and *inter-agent* methods, which correspond to the reuse of knowledge from experience gathered from the agent itself and that acquired from other agents, respectively.

Most recently, deep MARL techniques have received considerable attention. [Nguyen et al. \(2020\)](#) surveyed how deep learning techniques were used to address the challenges in multi-agent learning, such as partial observability, continuous state and action spaces, and transfer learning. [OroojlooyJadid and Hajinezhad \(2019\)](#) reviewed the application of deep MARL techniques in fully cooperative games: the survey on this setting is thorough. [Hernandez-Leal et al. \(2019\)](#) summarised how the classic ideas from traditional MAS research, such as emergent behaviour, learning communication, and opponent modelling, were incorporated into deep MARL domains, based on which they proposed a new categorisation for deep MARL methods. [Zhang et al. \(2019c\)](#) performed a selective survey on MARL algorithms that have theoretical convergence guarantees and complexity analysis. To the best of my knowledge, their review is the only one to cover more advanced topics such as decentralised MARL with networked agents, mean-field MARL, and MARL for stochastic potential games.

On the application side, [Müller and Fischer \(2014\)](#) surveyed 152 real-world applications in various sectors powered by MAS techniques. [Campos-Rodriguez et al. \(2017\)](#) reviewed the application of multi-agent techniques for automotive industry applications, such as traffic coordination and route balancing. [Derakhshan and Yousefi \(2019\)](#) focused on real-world applications for wireless sensor networks, [Shakshuki and Reid \(2015\)](#) studied multi-agent applications for the healthcare in-

dustry, and Kober et al. (2013) investigated the application of robotic control and summarised profitable RL approaches that can be applied to robots in the real world.

2.2 Learning in Identical-Interest Games

The majority of MARL algorithms assume that agents collaborate with each other to achieve shared goals. In this setting, agents are usually considered homogeneous and play an interchangeable role in the environmental dynamics. In a two-player normal-form game or repeated game, for example, this means the payoff matrix is symmetrical.

2.2.1 Stochastic Team Games

One benefit of studying identical interest games is that single-agent RL algorithms with a theoretical guarantee can be safely applied. For example, in the team game² setting, since all agents' rewards are always the same, the Q-functions are identical among all agents. As a result, one can simply apply the single-agent RL algorithms over the joint action space $\mathbf{a} \in \mathbb{A}$, equivalently, Eq. (1.14) can be written as

$$\mathbf{eval}^i\left(\left\{Q^i(s_{t+1}, \cdot)\right\}_{i \in \{1, \dots, N\}}\right) = V^i\left(s_{t+1}, \arg \max_{\mathbf{a} \in \mathbb{A}} Q^i(s_{t+1}, \mathbf{a})\right). \quad (2.1)$$

Littman (1994) first studied this approach in SGs. However, one issue with this approach is that when multiple equilibria exist (e.g., a normal-form game with reward $R = \begin{bmatrix} 0,0 & 2,2 \\ 2,2 & 0,0 \end{bmatrix}$), unless the selection process is coordinated among agents, the agents' optimal policy can end up with a worse scenario even though their value functions have reached the optimal values. To address this issue, Claus and Boutilier (1998b) proposed to build belief models about other agents' policies. Similar to fictitious play (Berger, 2007), each agent chooses actions in accordance with its belief about the other agents. Empirical effectiveness, as well as convergence, have been reported for repeated games; however, the convergent equilibrium may not be optimal. In solving this problem, Wang and Sandholm (2003) proposed optimal

²The terms Markov team games, stochastic team games, and dynamic team games are interchangeably used across different domains of the literature.

adaptive learning (OAL) methods that provably converge to the optimal NE almost surely in any team SG. The main novelty of OAL is that it learns the game structure by building so-called *weakly acyclic games* that eliminate all the joint actions with sub-optimal NE values and then applies adaptive play (Young, 1993) to address the equilibrium selection problem for weakly acyclic games specifically. Following this approach, Arslan and Yüksel (2016) proposed decentralised Q-learning algorithms that, under the help of two-timescale analysis (Leslie et al., 2003), converge to an equilibrium policy for weakly acyclic SGs. To avoid sub-optimal equilibria for weakly acyclic SGs, Yongacoglu et al. (2019) further refined the decentralised Q-learners and derived theorems with stronger almost-surely convergence guarantees for optimal policies.

2.2.1.1 Solutions via Q-function Factorisation

Another vital reason that team games have been repeatedly studied is that solving team games is a crucial step in building distributed AI (DAI) (Gasser and Huhns, 2014; Huhns, 2012). The logic is that if each agent only needs to maintain the Q-function of $Q^i(s, a^i)$, which depends on the state and local action a^i , rather than joint action \mathbf{a} , then the combinatorial nature of multi-agent problems can be avoided. Unfortunately, Tan (1993) previously noted that such independent Q-learning methods do not converge in team games. Lauer and Riedmiller (2000) reported similar negative results; however, when the state transition dynamics are deterministic, independent learning through distributed Q-learning can still obtain a convergence guarantee. No additional expense is needed in comparison to the non-distributed case for computing the optimal policies.

Factorised MDPs (Boutilier et al., 1999) are an effective way to avoid exponential blowups. For a coordination task, if the joint-Q function can be naturally written as

$$Q = Q^1(a^1, a^2) + Q^2(a^2, a^4) + Q^3(a^1, a^3) + Q^4(a^3, a^4),$$

then the nested structure can be exploited. For example, Q^1 and Q^3 are irrelevant

in finding the optimal a^4 ; thus, given a^4 , Q^1 becomes irrelevant for optimising a^3 . Given a^3, a^4 , one can then optimise a^1, a^2 . Inspired by this result, [Guestrin et al. \(2002a,b\)](#); [Kok and Vlassis \(2004\)](#) studied the idea of *coordination graphs*, which combine value function approximation with a message-passing scheme by which agents can efficiently find the globally optimal joint action.

However, the coordination graph may not always be available in real-world applications; thus, the ideal approach is to let agents learn the Q-function factorisation from the tasks automatically. Deep neural networks are an effective way to learn such factorisations. Specifically, the scope of the problem is then narrowed to the so-called *decentralisable tasks* in the Dec-POMDP setting, that is, $\exists \{Q_i\}_{i \in \{1, \dots, N\}}$ $\forall \mathbf{o} \in \mathbb{O}, \mathbf{a} \in \mathbb{A}$, the following condition holds.

$$\arg \max_{\mathbf{a}} Q^{\pi}(\mathbf{o}, \mathbf{a}) = \begin{bmatrix} \arg \max_{a^1} Q^1(o^1, a^1) \\ \vdots \\ \arg \max_{a^N} Q^N(o^N, a^N) \end{bmatrix}. \quad (2.2)$$

Eq. (2.2) suggests that a task is decentralisable only if the local maxima on the individual value function per every agent amounts to the global maximum on the joint value function. Different structural constraints, enforced by particular neural architectures, have been proposed to satisfy this condition. For example, VDN ([Sunehag et al., 2018](#)) maintains an additivity structure by making $Q^{\pi}(\mathbf{o}, \mathbf{a}) := \sum_{i=1}^N Q^i(o^i, a^i)$. QMIX ([Rashid et al., 2018](#)) adopts a monotonic structure by means of a mixing network to ensure $\frac{\partial Q^{\pi}(\mathbf{o}, \mathbf{a})}{\partial Q^i(o^i, a^i)} \geq 0, \forall i \in \{1, \dots, N\}$. QTRAN ([Son et al., 2019](#)) introduces a more rigorous learning objective on top of QMIX that proves to be a sufficient condition for Eq. (2.2). However, these structure constraints heavily depend on specially designed neural architectures, which makes understanding the representational power (i.e., the approximation error) of the above methods almost infeasible. Another drawback is that the structure constraint also damages agents' efficient exploration during training. To mitigate these issues, [Yang et al. \(2020\)](#) proposed Q-DPP, which eradicates the structure constraints by approximating the Q-function through a *determinantal point process (DPP)* ([Kulesza et al., 2012](#)). DPP pushes

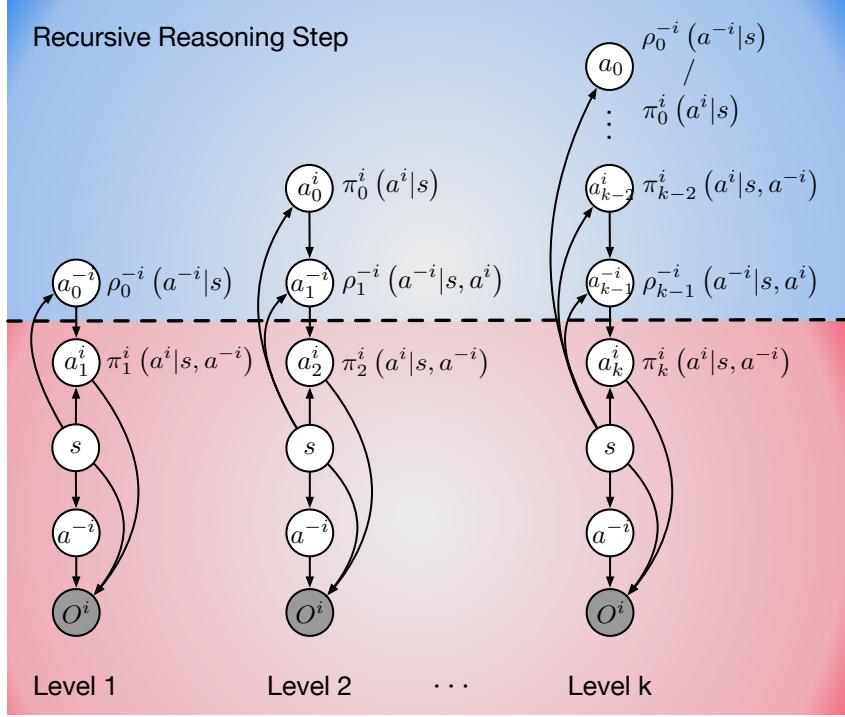


Figure 2.2: Graphical model of the *level-k* reasoning model (Wen et al., 2019b). The red part is the equivalent graphical model for the multi-agent learning problem. The blue part corresponds to the recursive reasoning steps. Subscript a_* stands for the level of thinking, not the time step. The opponent policies are approximated by ρ^{-i} . The omitted *level-0* model considers opponents that are fully randomised. Agent i rolls out the recursive reasoning about opponents in its mind (blue area). In the recursion, agents with higher-level beliefs take the best response to the lower-level agents. The higher-level models conduct all the computations that the lower-level models have done, e.g., the *level-2* model contains the *level-1* model by integrating out $\pi_0^i(a^i|s)$.

agents to explore and acquire diverse behaviours; consequently, it leads to natural decomposition of the joint Q-function with no need for *a priori* structure constraints. In fact, VDN/QMIX/QTRAN prove to be the exceptional cases of Q-DPP.

2.2.1.2 Solutions via Multi-Agent Soft Learning

In single-agent RL, the process of finding the optimal policy can be equivalently transformed into a probabilistic inference problem on a graphical model (Levine, 2018). The pivotal insight is that by introducing an additional binary random variable $P(\mathcal{O} = 1|s_t, a_t) \propto \exp(R(s_t, a_t))$, which denotes the *optimality* of the state-action pair at time step t , one can draw an equal connection between searching the optimal policies by RL methods and computing the marginal probability of

$p(\mathcal{O}_t^i = 1)$ by probabilistic inference methods, such as message passing or variational inference (Blei et al., 2017). This equivalence between optimal control and probabilistic inference also holds in the multi-agent setting (Grau-Moya et al., 2018; Shi et al., 2019; Tian et al., 2019; Wen et al., 2019b, 2018). In the context of SG (see the red part in Figure 2.2), the optimality variable for each agent i is defined by $p(\mathcal{O}_t^i = 1 | \mathcal{O}_t^{-i} = 1, \tau_t^i) \propto \exp(r^i(s_t, a_t^i, a_t^{-i}))$, which implies that the optimality of trajectory $\tau_t^i = (s_0, a_0^i, a_0^{-i}, \dots, s_t, a_t^i, a_t^{-i})$ depends on whether agent i acts according to its best response against other agents, and $\mathcal{O}_t^{-i} = 1$ indicates that all other agents are perfectly rational and attempt to maximise their rewards. Therefore, from each agent's perspective, its objective becomes maximising $p(\mathcal{O}_{1:T}^i = 1 | \mathcal{O}_{1:T}^{-i} = 1)$. As we assume no knowledge of the optimal policies and the model of the environment, we treat states and actions as latent variables and apply variational inference (Blei et al., 2017) to approximate this objective, which leads to

$$\begin{aligned} \max_{\theta^i} J(\boldsymbol{\pi}_\theta) &= \log p(\mathcal{O}_{1:T}^i = 1 | \mathcal{O}_{1:T}^{-i} = 1) \\ &\geq \sum_{t=1}^T \mathbb{E}_{s \sim P(\cdot|s, \mathbf{a}), \mathbf{a} \sim \boldsymbol{\pi}_\theta(s)} \left[r^i(s_t, a_t^i, a_t^{-i}) + \mathcal{H}(\boldsymbol{\pi}_\theta(a_t^i, a_t^{-i}|s_t)) \right]. \end{aligned} \quad (2.3)$$

One major difference from traditional RL is the additional entropy term³ in Eq. (2.3). Under this new objective, the value function is written as $V^i(s) = \mathbb{E}_{\boldsymbol{\pi}_\theta} [Q^i(s_t, a_t^i, a_t^{-i}) - \log(\boldsymbol{\pi}_\theta(a_t^i, a_t^{-i}|s_t))]$, and the corresponding optimal Bellman operator is

$$(\mathbf{H}^{\text{soft}} Q^i)(s, a^i, a^{-i}) \triangleq r^i(s, a^i, a^{-i}) + \gamma \cdot \mathbb{E}_{s' \sim P(\cdot|s, \mathbf{a})} \left[\log \sum_{\mathbf{a}} Q^i(s', \mathbf{a}) \right]. \quad (2.4)$$

This process is called *soft learning* because $\log \sum_{\mathbf{a}} \exp(Q(s, \mathbf{a})) \approx \max_{\mathbf{a}} Q(s, \mathbf{a})$.

One substantial benefit of developing a probabilistic framework for multi-agent learning is that it can help model the *bounded rationality* (Simon, 1972). Instead of assuming perfect rationality and agents reaching NE, bounded rationality accounts for situations in which rationality is compromised; it can be constrained by

³Soft learning is also called maximum-entropy RL (Haarnoja et al., 2018).

either the difficulty of the decision problem or the agents' own cognitive limitations. One intuitive example is the psychological experiment of the Keynes beauty contest ([Keynes, 1936](#)), in which all players are asked to guess a number between 0 and 100 and the winner is the person whose number is closest to the 1/2 of the average number of all guesses. Readers are recommended to pause here and think about which number you would guess. Although the NE of this game is 0, the majority of people guess a number between 13 and 25 ([Coricelli and Nagel, 2009](#)), which suggests that human beings tend to reason only by 1-2 levels of recursion in strategic games [Camerer et al. \(2004\)](#), i.e., "I believe how you believe how I believe".

[Wen et al. \(2018\)](#) developed the first MARL powered reasoning model that accounts for bounded rationality, which they called *probabilistic recursive reasoning* (PR2). The key idea of PR2 is that a dependency structure is assumed when splitting the joint policy π_θ , written by

$$\pi_\theta(a^i, a^{-i}|s) = \pi_{\theta^i}^i(a^i|s) \rho_{\theta^{-i}}^{-i}(a^{-i}|s, a^i) \quad (\text{PR2, Level-1}), \quad (2.5)$$

that is, the opponent is considering how the learning agent is going to affect its actions, i.e., a Level-1 model. The unobserved opponent model is approximated by a best-fit model $\rho_{\theta^{-i}}$ when optimising Eq. (2.3). In the team game setting, since agents' objectives are fully aligned, the optimal $\rho_{\phi^{-i}}$ has a closed-form solution $\rho_{\phi^{-i}}^{-i}(a^{-i}|s, a^i) \propto \exp(Q^i(s, a^i, a^{-i}) - Q^i(s, a^i))$. Following the direction of recursive reasoning, [Tian et al. \(2019\)](#) proposed an algorithm named ROMMEO that splits the joint policy by

$$\pi_\theta(a^i, a^{-i}|s) = \pi_{\theta^i}^i(a^i|s, a^{-i}) \rho_{\theta^{-i}}^{-i}(a^{-i}|s) \quad (\text{ROMMEO, Level-1}), \quad (2.6)$$

in which a Level-1 model is built from the learning agent's perspective. [Grau-Moya et al. \(2018\)](#); [Shi et al. \(2019\)](#) introduced a Level-0 model where no explicit recursive reasoning is considered.

$$\pi_\theta(a^i, a^{-i}|s) = \pi_{\theta^i}^i(a^i|s) \rho_{\theta^{-i}}^{-i}(a^{-i}|s) \quad (\text{Level-0}). \quad (2.7)$$

However, they generalised the multi-agent soft learning framework to include the zero-sum setting. [Wen et al. \(2019b\)](#) recently proposed a mixture of hierarchy Level- k models in which agents can reason at different recursion levels, and higher-level agents make the best response to lower-level agents (see the blue part in Figure 2.2). They called this method *generalised recursive reasoning* (GR2).

$$\pi_k^i(a_k^i|s) \propto \int_{a_{k-1}^{-i}} \left\{ \pi_k^i(a_k^i|s, a_{k-1}^{-i}) \cdot \underbrace{\int_{a_{k-2}^i} \left[\rho_{k-1}^{-i}(a_{k-1}^{-i}|s, a_{k-2}^i) \pi_{k-2}^i(a_{k-2}^i|s) \right] da_{k-2}^i}_{\text{opponents of level } k-1 \text{ best responds to agent } i \text{ of level } k-2} \right\} da_{k-1}^{-i}. \quad (\textbf{GR2, Level-}K). \quad (2.8)$$

In GR2, practical multi-agent soft actor-critic methods with convergence guarantee were introduced to make large- K reasoning tractable.

2.2.2 Dec-POMDPs

Dec-POMDP is a stochastic team game with partial observability. However, optimally solving Dec-POMDPs is a challenging combinatorial problem that is *NEXP*-complete ([Bernstein et al., 2002](#)). As the horizon increases, the doubly exponential growth in the number of possible policies quickly makes solution methods intractable. Most of the solution algorithms for Dec-POMDPs, including the above VDN/QMIX/QTRAN/Q-DPP, are based on the learning paradigm of centralised training with decentralised execution (CTDE) ([Oliehoek et al., 2016](#)). CTDE methods assume a centralised controller that can access observations across all agents during training. A typical implementation is through a centralised critic with a decentralised actor ([Lowe et al., 2017a](#)). In representing agents' local policies, stochastic finite-state controllers and a correlation device are commonly applied ([Bernstein et al., 2009](#)). Through this representation, Dec-POMDP can be formulated as non-linear programmes ([Amato et al., 2010](#)); this process allows the use of a wide range of off-the-shelf optimisation algorithms. [Dibangoye and Buffet \(2018\)](#); [Dibangoye et al. \(2016\)](#); [Szer et al. \(2005\)](#) introduced the transformation from Dec-POMDP into a continuous-state MDP, named the *occupancy-state MDP (oMDP)*.

The occupancy state is essentially a distribution over hidden states and the joint histories of observation-action pairs. In contrast to the standard MDP, where the agent learns an optimal value function that maps histories (or states) to real values, the learner in oMDP learns an optimal value function that maps occupancy states and joint actions to real values (they call the corresponding policy a *plan*). These value functions in oMDP are piece-wise linear and convex. Importantly, the benefit of restricting attention on the occupancy state is that the resulting algorithms are guaranteed to converge to a near-optimal plan for any finite Dec-POMDP with a probability of one, while traditional RL methods, such as REINFORCE, may only converge towards a local optimum.

In addition to CTDE methods, famous approximation solutions to Dec-POMDP include the Monte Carlo policy iteration method (Wu et al., 2010), which enjoys linear-time complexity in terms of the number of agents, planning by maximum-likelihood methods (Toussaint et al., 2008; Wu et al., 2013), which easily scales up to thousands of agents, and a method that decentralises POMDP by maintaining shared memory among agents (Nayyar et al., 2013).

2.2.3 Networked Multi-Agent MDPs

A rapidly growing area in the optimisation domain for addressing decentralised learning for cooperative tasks is the networked multi-agent MDP (M-MDP). In the context of M-MDP, agents are considered heterogeneous rather than homogeneous; they have different reward functions but still form a team to maximise the team-average reward $R = \frac{1}{N} \sum_{i=1}^N R^i(s, \mathbf{a}, s')$. Furthermore, in M-MDP, the centralised controller is assumed to be non-existent; instead, agents can only exchange information with their neighbours in a time-varying communication network defined by $G_t = ([N], E_t)$, where E_t represents the set of all communicative links between any two of the N neighbouring agents at time step t . The states and joint actions are assumed to be globally observable, but each agent's reward is only locally observable to itself. Compared to stochastic team games, this setting is believed to be more realistic for real-world applications such as smart grids (Dall'Anese et al., 2013) or transport management (Adler and Blue, 2002).

The cooperative goal of the agents in M-MDP is to maximise the team average cumulative discounted reward obtained by all agents over the network, that is,

$$\max_{\boldsymbol{\pi}} \frac{1}{N} \sum_{i=1}^N \mathbb{E} \left[\sum_{t \geq 0} \gamma^t R_t^i(s_t, \mathbf{a}_t) \right]. \quad (2.9)$$

Accordingly, under the joint policy $\boldsymbol{\pi} = \prod_{i \in \{1, \dots, N\}} \pi^i(a^i|s)$, the Q-function is defined as

$$Q^{\boldsymbol{\pi}}(s, \mathbf{a}) = \frac{1}{N} \sum_{i=1}^N \mathbb{E}_{\mathbf{a}_t \sim \boldsymbol{\pi}(\cdot|s_t), s_t \sim P(\cdot|s_t, \mathbf{a}_t)} \left[\sum_{t \geq 0} \gamma^t R_t^i(s_t, \mathbf{a}_t) \middle| s_0 = s, \mathbf{a}_0 = \mathbf{a} \right]. \quad (2.10)$$

To optimise Eq. (2.16), the optimal Bellman operator is written as

$$(\mathbf{H}^{\text{M-MDP}} Q)(s, \mathbf{a}) = \frac{1}{N} \sum_{i=1}^N R^i(s, \mathbf{a}) + \gamma \cdot \mathbb{E}_{s' \sim P(\cdot|s, \mathbf{a})} \left[\max_{\mathbf{a}' \sim \mathbb{A}} Q(s', \mathbf{a}') \right]. \quad (2.11)$$

However, since agents can know only their own reward, they do not share the estimation of the Q function but rather maintain their own copy. Therefore, from each agent's perspective, the individual optimal Bellman operator is written as

$$(\mathbf{H}^{\text{M-MDP}, i} Q^i)(s, \mathbf{a}) = R^i(s, \mathbf{a}) + \gamma \cdot \mathbb{E}_{s' \sim P(\cdot|s, \mathbf{a})} \left[\max_{\mathbf{a}' \sim \mathbb{A}} Q^i(s', \mathbf{a}') \right]. \quad (2.12)$$

To solve the optimal joint policy $\boldsymbol{\pi}^*$, the agents must reach **consensus** over the global optimal policy estimation, that is, if $Q^1 = \dots = Q^N = Q^*$, we know

$$(\mathbf{H}^{\text{M-MDP}} Q^*)(s, \mathbf{a}) = \frac{1}{N} \sum_{i=1}^N (\mathbf{H}^{\text{M-MDP}, i} Q^i). \quad (2.13)$$

To satisfy Eq. (2.13), [Zhang et al. \(2018b\)](#) proposed a method based on neural fitted-Q iteration (FQI) ([Riedmiller, 2005](#)) in the batch RL setting ([Lange et al., 2012](#)). Specifically, let \mathcal{F}_θ denote the parametric function class of neural networks that approximate Q-functions, let $\mathcal{D} = \{(s_k, \mathbf{a}_k^i, s'_k)\}$ be the replay buffer that contains all the transition data available to all agents, and let $\{R_k^i\}$ be the local reward known

only to each agent. The objective of FQI can be written as

$$\min_{f \in \mathcal{F}_\theta} \frac{1}{N} \sum_{i=1}^N \frac{1}{2K} \sum_{j=1}^K \left[y_k^i - f(s_k, \mathbf{a}_k; \theta) \right]^2, \quad \text{with } y_k^i = R_k^i + \gamma \cdot \max_{\mathbf{a} \in \mathbb{A}} Q_k^i(s'_k, \mathbf{a}). \quad (2.14)$$

In each iteration, K samples are drawn from \mathcal{D} . Since y_k^i is known only to each agent i , Eq. (2.14) becomes a typical consensus optimisation problem (i.e., consensus must be reached for θ) (Nedic and Ozdaglar, 2009). Multiple effective distributed optimisers can be applied to solve this problem, including the *DIGing* algorithm (Nedic et al., 2017). Let $g^i(\theta^i) = \frac{1}{2K} \sum_{j=1}^K [y_k^i - f(s_k, \mathbf{a}_k; \theta)]^2$, α be the learning rate, and $G([N], E_l)$ be the topology of the network in the l st iteration; the *DIGing* algorithm designs the gradient updates for each agent i as

$$\theta_{l+1}^i = \sum_{j=1}^N E_l(i, j) \cdot \theta_l^j - \alpha \cdot \rho_l^i, \quad \rho_{l+1}^i = \sum_{j=1}^N E_l(i, j) \cdot \rho_l^j + \nabla g^i(\theta_{l+1}^i) - \nabla g^i(\theta_l^i). \quad (2.15)$$

Intuitively, Eq. (2.15) implies that if all agents aim to reach a consensus on θ , they must incorporate a weighted combination of their neighbours' estimates into their own gradient updates. However, due to the usage of neural networks, the agents may not reach an exact consensus. Zhang et al. (2018b) also studied the finite-sample bound in a high-probability sense that quantifies the generalisation error of the proposed neural FQI algorithm.

The idea of reaching consensus can be directly applied to solving Eq. (2.9) via policy-gradient methods. Zhang et al. (2018c) proposed an actor-critic algorithm in which the global Q-function is approximated individually by each agent. On the basis of Eq. (1.15), the critic of $Q^{i, \pi_\theta}(s, \mathbf{a})$ is modelled by another neural network parameterised by ω^i , i.e., $Q^i(\cdot, \cdot; \omega^i)$, and the parameter ω^i is updated as

$$\omega_{t+1}^i = \sum_{j=1}^N E_t(i, j) \cdot \left(\omega_t^j + \alpha \cdot \delta_t^j \cdot \nabla_\omega Q_t^j(\omega_t^j) \right) \quad (2.16)$$

where $\delta_t^j = R_t^j + \gamma \cdot \max_{\mathbf{a} \in \mathbb{A}} Q_t^j(s'_t, \mathbf{a}; \omega_t^j) - Q_t^j(s'_t, \mathbf{a}; \omega_t^j)$ is the TD error. Similar

to Eq. (2.15), the update in Eq. (2.16) is a weighted sum of all the neighbouring gradients. The same group of authors later extended this approach to cover the continuous-action space in which a deterministic policy gradient method of Eq. (1.16) is applied (Zhang et al., 2018a). Moreover, (Zhang et al., 2018c) and (Zhang et al., 2018a) applied a linear function approximation to achieve an almost sure convergence guarantee. Following this thread, Suttle et al. (2019) and Zhang and Zavlanos (2019) extended the actor-critic method to an off-policy setting, rendering more data-efficient MARL algorithms.

2.2.4 Stochastic Potential Games

The potential game (PG) first appeared in Monderer and Shapley (1996). The physical meaning of Eq. (1.21) is that if any agent changes its policy unilaterally, the changes in reward will be represented on the potential function shared by all agents. A PG is guaranteed to have a pure-strategy NE – a desirable property that does not generally hold in normal-form games. Many efforts have since been dedicated to finding the NE of (static) PGs (Lă et al., 2016), among which fictitious play (Berger, 2007) and generalised weakened fictitious play (Leslie and Collins, 2006) are probably the most common solutions.

Generally, stochastic PGs (SPGs)⁴ can be regarded as the “single-agent component” of a multi-agent stochastic game (Candogan et al., 2011) since all agents’ interests in SPGs are described by a single potential function. However, the analysis of SPGs is exceptionally sparse. Zazo et al. (2015) studied an SPG with deterministic transition dynamics in which agents consider only *open-loop policies*⁵. In fact, generalising a PG to the stochastic setting is further complicated because agents must now execute policies that depend on the state and consider the actions of other players. In this setting, González-Sánchez and Hernández-Lerma (2013) investigated a type of SPG in which they derive a sufficient condition for NE, but it requires each agent’s reward function to be a concave function of the state and

⁴As with team games, stochastic PG is also called dynamic PG or Markov PG.

⁵Open loop means that agents’ actions are a function of time only. By contrast, close-loop policies take into account the state. In deterministic systems, these policies can be optimal and coincide in value. For a stochastic system, an open-loop strategy is unlikely to be optimal since it cannot adapt to state transitions.

the transition function to be invertible. [Macua et al. \(2018\)](#) studied a general form of SPG where a *closed-loop* NE can be found. Although they demonstrated the equivalence between solving the closed-loop NE and solving a single-agent optimal control problem, the agents' policies must depend only on disjoint subsets of components of the state. Notably, both [González-Sánchez and Hernández-Lerma \(2013\)](#) and [Macua et al. \(2018\)](#) proposed centralised methods; optimisation over the joint action space surely results in a combinatorial complexity when solving the SPGs. In addition, they do not consider an RL setting in which the system is *a priori* unknown.

The work of [Mguni \(2020\)](#) is probably the most comprehensive treatment of SPGs in a model-free setting. Similar to [Macua et al. \(2018\)](#), the authors revealed that the NE of the PG in pure strategies could be found by solving a dual-form MDP, but they reached the conclusion without the disjoint state assumption: the transition dynamics and potential function must be known. Specifically, they provided an algorithm to estimate the potential function based on the reward samples. To avoid combinatorial explosion, they also proposed a distributed policy-gradient method based on generalised weakened fictitious play ([Leslie and Collins, 2006](#)) that has linear-time complexity.

Recently, [Mazumdar and Ratliff \(2018\)](#) studied the dynamics of gradient-based learning on potential games. They found that in a general superclass of potential games named *Morse-Smale games* ([Hirsch, 2012](#)), the limit sets of competitive gradient-based learning with stochastic updates are attractors almost surely, and those attractors are either local Nash equilibria or non-Nash locally asymptotically stable equilibria but not saddle points.

2.3 Learning in Zero-Sum Games

Zero-sum games represent a competitive relationship among players in a game. Solving three-player zero-sum games is believed to be *PPAD*-hard ([Daskalakis and Papadimitriou, 2005](#)). In the two-player case, the NE $(\pi^{1,*}, \pi^{2,*})$ is essentially a saddle point $\mathbb{E}_{\pi^1, \pi^{2,*}}[R] \leq \mathbb{E}_{\pi^{1,*}, \pi^{2,*}}[R] \leq \mathbb{E}_{\pi^{1,*}, \pi^2}[R], \forall \pi^1, \pi^2$, and can be formulated

as an LP problem in Eq. (2.17).

$$\begin{aligned}
 & \min U_1^* \\
 & \sum_{a^2 \in \mathbb{A}^2} R^1(a^1, a^2) \cdot \pi^2(a^2) \leq U_1^*, \quad \forall a^1 \in \mathbb{A}^1 \\
 \text{s.t. } & \sum_{a^2 \in \mathbb{A}^2} \pi^2(a^2) = 1 \\
 & \pi^2(a^2) \geq 0, \quad \forall a^2 \in \mathbb{A}^2
 \end{aligned} \tag{2.17}$$

Eq. (2.17) is considered from the min-player's perspective. One can also derive a dual-form LP from the max-player's perspective. In discrete games, the minimax theorem ([Von Neumann and Morgenstern, 1945](#)) is a simple consequence of the strong duality theorem of LP⁶ ([Matousek and Gärtner, 2007](#)),

$$\min_{\pi^1} \max_{\pi^2} \mathbb{E}[R(\pi^1, \pi^2)] = \max_{\pi^2} \min_{\pi^1} \mathbb{E}[R(\pi^1, \pi^2)] \tag{2.18}$$

which suggests the fact that whether the min player acts first or the max player acts first does not matter. However, the minimax theorem does not hold in general for multi-player zero-sum continuous games in which the reward function is nonconvex-nonconcave. In fact, a barrier to tractability exists for multi-player zero-sum games and two-player zero-sum games with continuous states and actions.

2.3.1 Discrete State-Action Games

Similar to single-agent MDP, value-based methods aim to find an optimal value function, which in the context of zero-sum SGs, corresponds to the minimax NE of the game. In two-player zero-sum SGs with discrete states and actions, we know $V^{1,\pi^1,\pi^2} = -V^{2,\pi^1,\pi^2}$, and by the minimax theorem ([Von Neumann and Morgenstern, 1945](#)), the optimal value function is $V^* = \max_{\pi^2} \min_{\pi^1} V^{1,\pi^1,\pi^2} = \min_{\pi^1} \max_{\pi^2} V^{1,\pi^1,\pi^2}$. In each stage game defined by $Q^1 = -Q^2$, the optimal value can be solved by a matrix zero-sum game through a linear program in Eq. (2.17). [Shapley \(1953\)](#) introduced the first value-iteration method, written as

⁶Solving zero-sum games is equivalent to solving a LP; [Dantzig \(1951\)](#) also proved the correctness of the other direction, that is, any LP can be reduced to a zero-sum game, though some degenerate solutions need careful treatments ([Adler, 2013](#)).

$$(\mathbf{H}^{\text{Shapley}}V)(s) = \min_{\pi^1 \in \Delta(\mathbb{A}^1)} \max_{\pi^2 \in \Delta(\mathbb{A}^2)} \mathbb{E}_{a^1 \sim \pi^1, a^2 \sim \pi^2, s' \sim P} [R^1(s, a^1, a^2) + \gamma \cdot V(s')], \quad (2.19)$$

and proved $\mathbf{H}^{\text{Shapley}}$ is a contraction mapping (in the sense of the infinity norm) in solving two-player zero-sum SGs. In other words, assuming the transitional dynamics and reward function are known, the value-iteration method will generate a sequence of value functions $\{V_t\}_{t \geq 0}$ that asymptotically converges to the fixed point V^* , and the corresponding policies will converge to the NE policies $\boldsymbol{\pi}^* = (\pi^{1,*}, \pi^{2,*})$.

In contrast to Shapley's model-based value-iteration method, [Littman \(1994\)](#) proposed a model-free Q-learning method – Minimax-Q – that extends the classic Q-learning algorithm defined in Eq. (1.13) to solve zero-sum SGs. Specifically, in Minimax-Q, Eq. (1.14) can be equivalently written as

$$\begin{aligned} \mathbf{eval}^1(\{Q^1(s_{t+1}, \cdot)\}) &= -\mathbf{eval}^2(\{Q^2(s_{t+1}, \cdot)\}) \\ &= \min_{\pi^1 \in \Delta(\mathbb{A}^1)} \max_{\pi^2 \in \Delta(\mathbb{A}^2)} \mathbb{E}_{a^1 \sim \pi^1, a^2 \sim \pi^2} [Q^1(s_{t+1}, a^1, a^2)]. \end{aligned} \quad (2.20)$$

The Q-learning update rule of Minimax-Q is exactly the same as that in Eq. (1.13). Minimax-Q can be considered an approximation algorithm for computing the fixed point Q^* of the Bellman operator of Eq. (1.20) through stochastic sampling. Importantly, it assumes no knowledge about the environment. [Szepesvári and Littman \(1999\)](#) showed that under similar assumptions to those for Q-learning ([Watkins and Dayan, 1992](#)), the Bellman operator of Minimax-Q is a contraction mapping operator, and the stochastic updates made by Minimax-Q eventually lead to a unique fixed point that corresponds to the NE value. In addition to the tabular-form Q-function in Minimax-Q, various Q-function approximators have been developed. For example, [Lagoudakis and Parr \(2003\)](#) studied the factorised linear architectures for Q-function representation. [Yang et al. \(2019c\)](#) adopted deep neural networks and derived a rigorous finite-sample error bound. [Zhang et al. \(2018b\)](#) also derived a finite-sample bound for linear function approximators in the competitive M-MDPs.

2.3.2 Continuous State-Action Games

Recently, the challenge of training generative adversarial networks (GANs) (Goodfellow et al., 2014a) has ignited tremendous research interest in understanding policy gradient methods in two-player continuous games, specifically, games with a continuous state-action space and nonconvex-nonconcave loss landscape. In GANs, two neural network parameterised models – the generator G and the discriminator D – play a zero-sum game. In this game, the generator attempts to generate data that “look” authentic such that the discriminator cannot tell the difference from the true data; on the other hand, the discriminator tries not to be deceived by the generator. The loss function in this scenario is written as

$$\min_{\theta_G \in \mathbb{R}^d} \max_{\theta_D \in \mathbb{R}^d} f(\theta_G, \theta_D) = \quad (2.21)$$

$$\left[\mathbb{E}_{x \sim p_{\text{data}}} [\log D_{\theta_D}(x)] + \mathbb{E}_{z \sim p(z)} [\log (1 - D_{\theta_D}(G_{\theta_G}(z)))] \right]$$

where θ_G and θ_D represent neural networks parameters and z is a random signal, serving as the input to the generator. In searching for the NE, one naive approach is to update both θ_G and θ_D by simultaneously implementing the gradient-descent-ascent (GDA) updates with the same step size in Eq. (2.21). This approach is equivalent to a MARL algorithm in which both agents are applying policy-gradient methods. With trivial adjustments to the step size (Bowling, 2005; Bowling and Veloso, 2002; Zhang and Lesser, 2010), GDA methods can work effectively in two-player two-action (thus convex-concave) games. However, in the nonconvex-nonconcave case, where the minimax theorem no longer holds, GDA methods are notoriously flawed from three aspects. First, GDA algorithms may not converge at all (Balduzzi et al., 2018a; Daskalakis and Panageas, 2018; Mertikopoulos et al., 2018), resulting in limited cycles⁷ in which even the time average⁸ does not coincide with NE (Mazumdar et al., 2019a). Second, there exist undesired stable stationary points for

⁷Limited cycle is a terminology in the study of dynamical systems, which describes oscillatory systems. In game theory, an example of limit cycles in the strategy space can be found in Rock-Paper-Scissor game.

⁸In two-player two-action games, Singh et al. (2000b) showed that the time average payoffs would converge to a NE value if their policies do not.

the GDA algorithms that are not local optima of the game (Adolphs et al., 2019; Mazumdar et al., 2019a). Third, there exist games whose equilibria are not the attractors of GDA methods at all (Mazumdar et al., 2019a). These problems are partly caused by the intransitive dynamics (e.g., a typical intransitive game is rock-paper-scissors game) that are inherent in zero-sum games (Balduzzi et al., 2018a; Omidshafiei et al., 2020) and the fact that each agent may have a non-smooth objective function. In fact, even in simple linear-quadratic games, the reward function cannot satisfy the smoothness condition⁹ globally, and the games are surprisingly not convex either (Fazel et al., 2018; Mazumdar et al., 2019a; Zhang et al., 2019d).

Three mainstream approaches have been followed to develop algorithms that have at least a local convergence guarantee. One natural idea is to make the inner loop solvable at a reasonably high level and then focus on a simpler type of game. In other words, the algorithm tries to find a stationary point of the function $\Phi(\cdot) := \max_{\theta_D \in \mathbb{R}^d} f(\cdot, \theta_D)$, instead of Eq. (2.21). For example, by considering games with a nonconvex and (strongly) concave loss landscape, Kong and Monteiro (2019); Lin et al. (2019); Lu et al. (2020a); Nouiehed et al. (2019); Rafique et al. (2018); Thekumpampil et al. (2019) presented an affirmative answer that GDA methods can converge to a stationary point in the outer loop of optimising $\Phi(\cdot) := \max_{\theta_D \in \mathbb{R}^d} f(\cdot, \theta_D)$. Based on this understanding, they developed various GDA variants that apply the “best response” in the inner loop while maintaining an inexact gradient descent in the outer loop. We refer to Lin et al. (2019) [Table 1] for a detailed summary of the time complexity of the above methods.

The second mainstream idea is to shift the equilibrium of interest from the NE, which is induced by simultaneous gradient updates, to the Stackelberg equilibrium, which is a solution concept in leader-follower (i.e., alternating update) games. Jin et al. (2019) introduced the concept of the local Stackelberg equilibrium, named *local minimax*, based on which he established the connection to GDA methods by showing that all stable limit points of GDA are exactly local minimax points. Fiez et al. (2019) also built connections between the NE and Stackelberg equilibrium

⁹A differentiable function is said to be smooth if the gradients of the function are continuous.

by formulating the conditions under which attracting points of GDA dynamics are Stackelberg equilibria in zero-sum games. When the loss function is bilinear, theoretical evidence was found that alternating updates converge faster than simultaneous GDA methods (Zhang and Yu, 2019).

The third mainstream idea is to analyse the loss landscape from a game-theoretic perspective and design corresponding algorithms that mitigate oscillatory behaviour. Compared to the previous two mainstream ideas, which helped generate more theoretical insights than applicable algorithms, works within this stream demonstrate strong empirical improvements in training GANs. Mescheder et al. (2017) investigated the game Hessian and identified that issues on the eigenvalues trigger the limited cycles. As a result, they proposed a new type of update rule based on consensus optimisation, together with a convergence guarantee to a local NE in smooth two-player zero-sum games. Adolphs et al. (2019) leveraged the curvature information of the loss landscape to propose algorithms in which all stable limit points are guaranteed to be local NEs. Similarly, Mazumdar et al. (2019b) took advantage of the differential structure of the game and constructed an algorithm for which the local NEs are the only attracting fixed points. In addition, Daskalakis et al. (2017); Mertikopoulos et al. (2018) addressed the issue of limit cycling behaviour in training GANs by proposing the technique of *optimistic mirror descent* (OMD). OMD achieves the last-iterate convergent guarantee in bilinear convex-concave games. Specifically, at each time step, OMD adjusts the gradient of that time step by considering the opponent policy at the next time step. Let M_{t+1} be the predictor of the next iteration gradient¹⁰; we can write OMD as follows.

$$\begin{aligned}\theta_{G,t+1} &= \theta_{G,t} + \alpha \cdot (\nabla_{\theta_G,t} f(\theta_G, \theta_D) + M_{\theta_G,t+1} - M_{\theta_G,t}) \\ \theta_{D,t+1} &= \theta_{D,t} - \alpha \cdot (\nabla_{\theta_D,t} f(\theta_G, \theta_D) + M_{\theta_D,t+1} - M_{\theta_D,t})\end{aligned}\quad (2.22)$$

In fact, the pivotal idea of opponent prediction in OMD, developed in the optimisation domain, resembles the idea of approximate policy prediction in the MARL domain (Foerster et al., 2018a; Zhang and Lesser, 2010).

¹⁰In practice, it is usually set as the last iteration gradient.

Thus far, the most promising results are probably those of [Bu et al. \(2019\)](#) and [Zhang et al. \(2019d\)](#), which reported the first results in solving zero-sum LQ games with a global convergence guarantee. Specifically, [Zhang et al. \(2019d\)](#) developed the solution through projected nested-gradient methods, while [Bu et al. \(2019\)](#) solved the problem through a projection-free Stackelberg leadership model. Both of the models achieve a sublinear rate for convergence.

2.3.3 Extensive-Form Games

As briefly introduced in Section 1.3.4, zero-sum EFG with imperfect information can be efficiently solved via LP in sequence form representations ([Koller and Megiddo, 1992, 1996](#)). However, these approaches are limited to solving only small-scale problems (e.g., games with $\mathcal{O}(10^7)$ information states). In fact, considerable additional effort is needed to address real-world games (e.g., limit Texas hold’em, which has $\mathcal{O}(10^{18})$ game states); to name a few, Monte Carlo Tree Search (MCTS) techniques¹¹ ([Browne et al., 2012](#); [Cowling et al., 2012](#); [Silver et al., 2016](#)), isomorphic abstraction techniques ([Billings et al., 2003](#); [Gilpin and Sandholm, 2006](#)), and iterative (policy) gradient-based approaches ([Gilpin et al., 2007](#); [Gordon, 2007](#); [Zinkevich, 2003](#)).

A central idea of iterative policy gradient-based methods is minimising regret¹². A learning rule achieves no-regret, also called *Hannan consistency* in game theoretical terms ([Hannan, 1957](#)), if, intuitively speaking, against any set of opponents it yields a payoff that is no less than the payoff the learning agent could have obtained by playing any one of its pure strategies in hindsight. Recall the reward function under a given policy $\boldsymbol{\pi} = (\pi^i, \pi^{-i})$ in Eq. (1.27); the (average) regret of player i is defined by:

$$\text{Reg}_T^i = \frac{1}{T} \max_{\pi^i} \sum_{t=1}^T \left[R^i(\boldsymbol{\pi}^i, \boldsymbol{\pi}_t^{-i}) - R^i(\boldsymbol{\pi}_t^i, \boldsymbol{\pi}_t^{-i}) \right]. \quad (2.23)$$

¹¹Notably, though MCTS methods such as UCT ([Kocsis and Szepesvári, 2006](#)) work remarkably well in turn-based EFGs, such as GO and chess, they cannot converge to a NE trivially in (even perfect-information) simultaneous-move games ([Schaeffer et al., 2009](#)). See a rigorous treatment for remedy in [Lisy et al. \(2013\)](#).

¹²One can regard minimising regret as one solution concept for multi-agent learning problems, similar to the reward maximisation in single-agent learning.

A no-regret algorithm satisfies $\text{Reg}_T^i \rightarrow 0$ as $T \rightarrow \infty$ with probability 1. When Eq. (2.23) equals zero, all agents are acting with their best response to others, which essentially forms a NE. Therefore, one can regard regret as a type of “distance” to NE. As one would expect, the single-agent Q-learning procedure can be shown to be Hannan consistent in a stochastic game against opponents playing stationary policies (Shoham and Leyton-Brown, 2008) [Chapter 7] since the optimal Q-function guarantees the best response. In contrast, the Minimax-Q algorithm in Eq. (2.20) is not Hannan consistent because if the opponent plays a sub-optimal strategy, Minimax-Q is unable to exploit the opponent due to the over-conservativeness in terms of over-estimating its opponents.

An important result about regret states is that in a zero-sum game at time T , if both players’ average regret is less than ε , then their average strategy constitutes a 2ε -NE of the game (Zinkevich et al., 2008, Theorem 2). In general-sum games, the average strategy of the ε -regret algorithm will reach an ε -coarse correlated equilibrium of the game (Michael, 2020, Theorem 6.3.1). This result essentially implies that regret-minimising algorithms (or, algorithms with Hannan consistency) applied in a self-play manner can be used as a general technique to approximate the NE of zero-sum games. Building upon this finding, two families of methods are developed, namely, fictitious play types of methods (Berger, 2007) and counterfactual regret minimisation (Zinkevich et al., 2008), which lay the theoretical foundations for modern techniques to solve real-world games.

2.3.3.1 Variations of Fictitious Play

Fictitious play (FP) (Berger, 2007) is one of the oldest learning procedures in game theory that is provably convergent for zero-sum games, potential games, and two-player n-action games with generic payoffs. In FP, each player maintains a belief about the empirical mean of the opponents’ average policy, based on which the player selects the best response. With the best response defined in Eq. (1.17), we can write the FP updates as

$$a_t^{i,*} \in \mathbf{Br}^i \left(\pi_t^{-i} = \frac{1}{t} \sum_{\tau=0}^{t-1} \mathbb{1} \{ a_\tau^{-i} = a, a \in \mathbb{A} \} \right), \pi_{t+1}^i = \left(1 - \frac{1}{t} \right) \pi_t^i + \frac{1}{t} a_t^{i,*}, \forall i. \quad (2.24)$$

In the FP scheme, each agent is oblivious to the other agents' reward; however, they need full access to their own payoff matrix in the stage game. In the continuous case with an infinitesimal learning rate of $1/t \rightarrow 0$, Eq. (2.24) is equivalent to $d\boldsymbol{\pi}_t/dt \in \mathbf{Br}(\boldsymbol{\pi}_t) - \boldsymbol{\pi}_t$ in which $\mathbf{Br}(\boldsymbol{\pi}_t) = (\mathbf{Br}(\boldsymbol{\pi}_t^{-1}), \dots, \mathbf{Br}(\boldsymbol{\pi}_t^{-N}))$. [Viossat and Zapechelnyuk \(2013\)](#) proved that continuous FP leads to no regret and is thus Hannan consistent. If the empirical distribution of each $\boldsymbol{\pi}_t^i$ converges in FP, then it converges to a NE¹³.

Although standard discrete-time FP is not Hannan consistent ([Cesa-Bianchi and Lugosi, 2006](#), Exercise 3.8), various extensions have been proposed that guarantee such a property; see a full list summarised in [Hart \(2013\)](#) [Section 10.9]. Smooth FP ([Fudenberg and Kreps, 1993](#); [Fudenberg and Levine, 1995](#)) is a stochastic variant of FP (thus also called stochastic FP) that considers a smooth ε -best response in which the probability of each action is a softmax function of that action's utility/reward against the historical frequency of the opponents' play. In smooth FP, each player's strategy is a genuine mixed strategy. Let $R^i(a_1^i, \boldsymbol{\pi}_t^{-i})$ be the expected reward of player i 's action $a_1^i \in \mathbb{A}^i$ under opponents' strategy $\boldsymbol{\pi}^{-i}$; the probability of playing a_1^i in the best response is written as

$$\mathbf{Br}_\lambda^i(\boldsymbol{\pi}_t^{-i}) := \frac{\exp\left(\frac{1}{\lambda} R^i(a_1^i, \boldsymbol{\pi}_t^{-i})\right)}{\sum_{k=1}^{|\mathbb{A}^i|} \exp\left(\frac{1}{\lambda} R^i(a_k^i, \boldsymbol{\pi}_t^{-i})\right)}. \quad (2.25)$$

[Benaïm and Faure \(2013\)](#) verified the Hannan consistency of the smooth best response with the smoothing parameter λ being time dependent and vanishing asymptotically. In potential games, smooth FP is known to converge to a neighbourhood of the set of NE ([Hofbauer and Sandholm, 2002](#)). Recently, [Swenson and Poor \(2019\)](#) showed a generic result that in almost all $N \times 2$ potential games, smooth FP converges to the neighbourhood of a pure-strategy NE with a probability of one.

In fact, “smoothing” the cumulative payoffs before computing the best re-

¹³Note that the convergence in Nash strategy does not necessarily mean the agents will receive the expected payoff value at NE. In the example of Rock-Paper-Scissor games, agents' actions are still miscorrelated after convergence, flipping between one of the three strategies, though their average policies do converge to $(1/3, 1/3, 1/3)$.

sponse is crucial to designing learning procedures that achieve Hannan consistency ([Kaniovski and Young, 1995](#)). One way to achieve such smoothness is through stochastic smoothing or adding perturbations¹⁴. For example, the smooth best response in Eq. (2.25) is a closed-form solution if one perturbs the cumulative reward by an additional entropy function, that is,

$$\pi^{i,*} \in \mathbf{Br}(\pi^{-i}) = \left\{ \arg \max_{\hat{\pi} \in \Delta(\mathbb{A}^i)} \mathbb{E}_{\hat{\pi}^i, \pi^{-i}} [R^i + \lambda \cdot \log(\hat{\pi})] \right\}. \quad (2.26)$$

Apart from smooth FP, another way to add perturbation is the *sampled FP* in which during each round, the player samples historical time points using a randomised sampling scheme, and plays the best response to the other players' moves, restricted to the set of sampled time points. Sampled FP is shown to be Hannan consistent when used with Bernoulli sampling ([Li and Tewari, 2018](#)).

Among the many extensions of FP, the most important is probably *generalised weakened FP* (GWFP) ([Leslie and Collins, 2006](#)), which releases the standard FP by allowing both approximate best response and perturbed average strategy updates. Specifically, if we write the ε -best response of player i as

$$R^i(\mathbf{Br}_\varepsilon(\pi^{-i}), \pi^{-i}) \geq \sup_{\pi \in \Delta(\mathbb{A}^i)} R^i(\pi, \pi^{-i}) - \varepsilon. \quad (2.27)$$

then the GWFP updating steps change from Eq. (2.24) to

$$\pi_{t+1}^i = (1 - \alpha^{t+1}) \pi_t^i + \alpha_{t+1} \left(\mathbf{Br}_\varepsilon^i(\pi^{-i}) + M_{t+1}^i \right), \quad \forall i. \quad (2.28)$$

GWFP is Hannan consistent if $\alpha_t \rightarrow 0, \varepsilon_t \rightarrow 0, \sum \alpha_t = \infty$ when $t \rightarrow \infty$, and $\{M_t\}$ meets $\lim_{t \rightarrow \infty} \sup_k \left\{ \left\| \sum_{i=t}^{k-1} \alpha^{i+1} M^{i+1} \right\| \text{ s.t. } \sum_{i=t}^{k-1} \alpha^{i+1} \leq T \right\} = 0$. It is trivial to see that GWFP recovers FP when $\alpha_t = 1/t, \varepsilon_t = 0, M_t = 0$. GWFP is an important extension of FP in that it provides two key components for bridging game theoretic ideas with RL techniques. With the approximate best response (highlighted in blue,

¹⁴The physical meaning of perturbing the cumulative payoff is to consider the incomplete information about what the opponent has been playing, variability in their payoffs, and unexplained trembles.

also named as the “weakened” term), this approach allows one to adopt a model-free RL algorithm, such as deep Q-learning, to compute the best response. Moreover, the perturbation term (highlighted in red, also named as the “generalised” term) enables one to incorporate policy exploration; if one applies an entropy term as the perturbation in addition to the best response (in which the smooth FP in Eq. (2.26) is also recovered), the scheme of maximum-entropy RL methods ([Haarnoja et al., 2018](#)) is recovered. In fact, the generalised term also accounts for the perturbation that comes from the fact the beliefs are not updated towards the exact mixed strategy π^{-i} but instead towards the observed actions ([Benaim and Hirsch, 1999](#)). As a direct application, [Perolat et al. \(2018\)](#) implemented the GWFP process through an actor-critic framework ([Konda and Tsitsiklis, 2000a](#)) in the MARL setting.

Brown’s original version of FP ([Berger, 2007](#)) describes alternating updates by players; yet, the modern usage of FP involves players updating their beliefs simultaneously ([Berger, 2007](#)). In fact, [Heinrich et al. \(2015\)](#) only recently proposed the first FP algorithm for EFG using the sequence-form representation. The extensive-form FP is essentially an adaptation of GWFP from NFG to EFG based on the insight that a mixture of normal-form strategies can be implemented by a weighted combination of behavioural strategies that have the same realisation plan (recall Section 1.3.3.2). Specifically, let π and β be two behavioural strategies, Π and B be the two realisation-equivalent mixed strategies¹⁵, and $\alpha \in \mathbb{R}^+$; then, for each information state S , we have

$$\tilde{\pi}(S) = \pi(S) + \frac{\alpha \mu^\beta(\sigma_S)}{(1 - \alpha) \mu^\pi(\sigma_S) + \alpha \mu^\beta(\sigma_S)} (\beta(S) - \pi(S)), \quad \forall S \in \mathbb{S}, \quad (2.29)$$

where σ_S is the sequence leading to S , $\mu^{\pi/\beta}(\sigma_S)$ is the realisation probability of σ_S under a given policy, and $\tilde{\pi}(S)$ defines a new behaviour that is realisation equivalent to the mixed strategy $(1 - \alpha)\Pi + \alpha B$. The extensive-form FP essentially iterates between Eq. (2.27), which computes the ε -best response, and Eq. (2.29), which

¹⁵Recall that in games with perfect recall, Kuhn’s theorem ([Kuhn, 1950a](#)) suggests that the behavioural strategy and mixed strategies are equivalent in terms of the realisation probability of different outcomes.

updates the old behavioural strategy with a step size of α . Note that these two steps must iterate over all information states of the game in each iteration. Similar to the normal-form FP in Eq. (2.24), extensive-form FP generates a sequence of $\{\boldsymbol{\pi}_t\}_{t \geq 1}$ that provably converges to the NE of a zero-sum game under self-play if the step size α goes to zero asymptotically. As a further enhancement, Heinrich and Silver (2016) implemented neural fictitious self-play (NFSP), in which the best response step is computed by deep Q-learning (Mnih et al., 2015) and the policy mixture step is computed through supervised learning. NFSP requires the storage of large replay buffers of past experiences; Lockhart et al. (2019) removes this requirement by obtaining the policy mixture for each player through an independent policy-gradient step against the respective best-responding opponent. All these amendments help make extensive-form FP applicable to real-world games with large-scale information states.

2.3.3.2 Counterfactual Regret Minimisation

Another family of methods achieve Hannan consistency by directly minimising the regret, in particular, a special kind of regret named counterfactual regret (CFR) (Zinkevich et al., 2008). Unlike FP methods, which are developed from the stochastic approximation perspective and generally have asymptotic convergence guarantees, CFR methods are established on the framework of online learning and online convex optimisation (Shalev-Shwartz et al., 2011), which makes analysing the speed of convergence, i.e., the regret bound, to the NE possible.

The key insight from CFR methods is that in order to minimise the total regret in Eq. (2.23) to approximate the NE, it suffices to minimise the *immediate counterfactual regret* at the level of each information state. Mathematically, Zinkevich et al. (2008) [Theorem 3] shows that the sum of the immediate counterfactual regret over all encountered information states provides an upper bound for the total regret in Eq. (2.23), i.e.,

$$\text{Reg}_T^i \leq \sum_{S \in \mathbb{S}^i} \max \left\{ \text{Reg}_{T,imm}^i(S), 0 \right\}, \quad \forall i. \quad (2.30)$$

To fully describe $\text{Reg}_{T,\text{imm}}^i(S)$, we need two additional notations. Let $\mu^\pi(\sigma_S \rightarrow \sigma_T)$ denote, given agents' behavioural policies π , the realisation probability of going from the sequence σ_S ¹⁶, which leads to the information state $S \in \mathbb{S}^i$ to its extended sequence σ_T , which continues from S and reaches the terminal state T . Let $\hat{v}^i(\pi, S)$ be the *counterfactual value function*, i.e., the expected reward of agent i in non-terminal information state S , which is written as

$$\hat{v}^i(\pi, S) = \sum_{s \in S, T \in \mathbb{T}} \mu^{\pi^{-i}}(\sigma_s) \mu^\pi(\sigma_s \rightarrow \sigma_T) R^i(T). \quad (2.31)$$

Note that in Eq. (2.31), the contribution from player i in realising σ_s is excluded; we treat whatever action current player i needs to reach state s as having a probability of one, that is, $\mu^{\pi^i}(\sigma_s) = 1$. The motivation is that now one can make the value function $\hat{v}^i(\pi, S)$ “counterfactual” simply by writing the consequence of player i not playing action a in the information state S as $(\hat{v}^i(\pi|_{S \rightarrow a}, S) - \hat{v}^i(\pi, S))$, in which $\pi|_{S \rightarrow a}$ is a joint strategy profile identical to π , except player i always chooses action a when information state S is encountered. Finally, based on Eq. (2.31), the immediate counterfactual regret can be expressed as

$$\text{Reg}_{T,\text{imm}}^i(S) = \max_{a \in \chi(S)} \text{Reg}_T^i(S, a), \quad \text{Reg}_T^i(S, a) = \frac{1}{T} \sum_{t=1}^T \left(\hat{v}^i(\pi_t|_{S \rightarrow a}, S) - \hat{v}^i(\pi_t, S) \right). \quad (2.32)$$

Note that the T in Eq. (2.31) is different from that in Eq. (2.32).

Since minimising the immediate counterfactual regret minimises the overall regret, we can find an approximate NE by choosing a specific behavioural policy $\pi^i(S)$ that minimises Eq. (2.32). To this end, one can apply Blackwell's approachability theorem (Blackwell et al., 1956) to minimise the regret independently on each information set, also known as *regret matching* (Hart and Mas-Colell, 2001). As we are most concerned with positive regret, denoted by $\lfloor \cdot \rfloor_+$, we have $\forall S \in \mathbb{S}^i, \forall a \in \chi(S)$, the strategy of player i at time $T + 1$ as Eq. (2.33).

¹⁶Recall that for games of perfect recall, the sequence that leads to the information state, including all the choice nodes within that information state, is unique.

$$\pi_{T+1}^i(S, a) = \begin{cases} \frac{\lfloor \text{Reg}_T^i(S, a) \rfloor_+}{\sum_{a \in \chi(S)} \lfloor \text{Reg}_T^i(S, a) \rfloor_+} & \text{if } \sum_{a \in \chi(S)} \lfloor \text{Reg}_T^i(S, a) \rfloor_+ > 0 \\ \frac{1}{|\chi(S)|} & \text{otherwise} \end{cases}. \quad (2.33)$$

In the standard CFR algorithm, for each information set, Eq. (2.33) is used to compute action probabilities in proportion to the positive cumulative regrets. In addition to regret matching, another online learning tool that minimises regret is *Hedge* (Freund and Schapire, 1997; Littlestone and Warmuth, 1994), in which an exponentially weighted function is used to derive a new strategy, which is

$$\pi_{t+1}(a_k) = \frac{\pi_t(a_k) e^{-\eta R_t(a_k)}}{\sum_{j=1}^K \pi_t(a_j) e^{-\eta R_t(a_j)}}, \quad \pi_1(\cdot) = \frac{1}{K}. \quad (2.34)$$

In computing Eq. (2.34), Hedge needs access to the full information of the reward values for all actions, including those that are not selected. *EXP3* (Auer et al., 1995) extended the Hedge algorithm for a *partial information game* in which the player knows only the reward of the chosen action (i.e., a bandit version) and has to estimate the loss of the actions that it does not select. Brown et al. (2017) augmented the Hedge algorithm with a tree-pruning technique based on dynamic thresholding. Gordon (2007) developed *Lagrangian hedging*, which unifies no-regret algorithms, including both regret matching and Hedge, through a class of potential functions. We recommend Cesa-Bianchi and Lugosi (2006) for a comprehensive overview of no-regret algorithms.

No-regret algorithms, under the framework of online learning, offer a natural way to study the regret bound (i.e., how fast the regret decays with time). For example, CFR and its variants ensure a counterfactual regret bound of $\mathcal{O}(\sqrt{T})^{17}$, as a result of Eq. (2.30), the convergence rate for the total regret is upper bounded by $\mathcal{O}(\sqrt{T} \cdot |\mathbb{S}|)$, which is linear in the number of information states. In other words, the average policy of applying CFR-type methods in a two-player zero-sum EFG

¹⁷According to Zinkevich (2003), any online convex optimisation problem can be made to incur $\text{Reg}_T = \Omega(\sqrt{T})$.

generates an $\mathcal{O}(|\mathbb{S}|/\sqrt{T})$ -approximate NE after T steps through self-play¹⁸.

Compared with the LP approach (recall Eq. (1.33)), which is applicable only for small-scale EFGs, the standard CFR method can be applied to limit Texas hold'em with as many as 10^{12} states. CFR⁺, the fastest implementation of CFR, can solve games with up to 10^{14} states (Tammelin et al., 2015). However, CFR methods still have a bottleneck in that computing Eq. (2.31) requires a traversal of the entire game tree to the terminal nodes in each iteration. Pruning the sub-optimal paths in the game tree is a natural solution (Brown et al., 2017; Brown and Sandholm, 2015, 2017). Many CFR variants have been developed to improve computational efficiency further. Lanctot et al. (2009) integrated Monte Carlo sampling with CFR (MCCFR) to significantly reduce the per iteration time cost of CFR by traversing a smaller sampled portion of the tree. Burch et al. (2012) improved MCCFR by sampling only a subset of a player's actions, which provides even faster convergence rate in games that contain many player actions. Gibson et al. (2012); Schmid et al. (2019) investigated the sampling variance and proposed MCCFR variants with a variance reduction module. Johanson et al. (2012b) introduced a more accurate MCCFR sampler by considering the set of outcomes from the chance node, rather than sampling only one outcome, as in all previous methods. Apart from Monte Carlo methods, function approximation methods have also been introduced (Jin et al., 2018; Waugh et al., 2014). The idea of these methods is to predict regret directly, and the no-regret algorithm then uses these predictions in place of the true regret to define a sequence of policies. To this end, the application of deep neural networks has led to great success (Brown et al., 2019).

Interestingly, there exists a hidden equivalence between model-free policy-based/actor-critic MARL methods and the CFR algorithm (Jin et al., 2018; Srivivasan et al., 2018). In particular, if we consider the counterfactual value function in Eq. (2.31) to be explicitly dependent on the action a that player i chooses at state S , in which we have $\hat{v}^i(\boldsymbol{\pi}, S) = \sum_{a \in \chi(S)} \pi^i(S, a) \hat{q}^i(\boldsymbol{\pi}, S, a)$, then it is shown

¹⁸The self-play assumption can in fact be released. Johanson et al. (2012a) shows that in two-player zero-sum games, as long as both agents minimise their regret, not necessarily through the same algorithm, their time-average policies will converge to NE with the same regret bound $\mathcal{O}(\sqrt{T})$. An example is to let a CFR player play against a best-response opponent.

in Srinivasan et al. (2018) [Section 3.2] that the Q-function in standard MARL $Q^{i,\pi}(s, \mathbf{a}) = \mathbb{E}_{s' \sim P, \mathbf{a} \sim \pi} [\sum_t \gamma^t R^i(s, \mathbf{a}, s') | s, \mathbf{a}]$ differs from $\hat{q}^i(\boldsymbol{\pi}, S, a)$ in CFR only by a constant of the probability of reaching S , that is,

$$Q^{i,\pi}(s, \mathbf{a}) = \frac{\hat{q}^i(\boldsymbol{\pi}, S, a)}{\sum_{s \in S} \mu^{\pi^{-i}}(\sigma_s)}. \quad (2.35)$$

Subtracting a value function on both sides of Eq. (2.35) leads to the fact that the counterfactual regret of $\text{Reg}_T^i(S, a)$ in Eq. (2.32) differs from the advantage function in MARL, i.e., $Q^{i,\pi}(s, a^i, a^{-i}) - V^{i,\pi}(s, a^{-i})$, only by a constant of the realisation probability. As a result, the multi-agent actor-critic algorithm (Foerster et al., 2018b) can be formulated as a special type of CFR method, thus sharing a similar convergence guarantee and regret bound in two-player zero-sum games. The equivalence has also been found by (Hennes et al., 2019), where the CFR method with Hedge can be written as a particular actor-critic method that computes the policy gradient through replicator dynamics.

2.3.4 Online Markov Decision Processes

A common situation in which online learning techniques are applied is in stateless games, where the learning agent faces an identical decision problem in each trial (e.g., playing a multi-arm bandit in the casino). However, real-world decision problems often occur in a dynamic and changing environment. Such an environment is commonly captured by a state variable which, when incorporated into online learning, leads to an online MDP. Online MDP (Auer et al., 2009; Even-Dar et al., 2009; Yu et al., 2009), also called adversarial MDP¹⁹, focuses on the problem in which the reward and transition dynamics can change over time, i.e., they are non-stationary and time-dependent.

In contrast to an ordinary stochastic game, the opponent/adversary in an online MDP is not necessarily rational or even self-optimising. The aim of studying online MDP is to provide the agent with policies that perform well against every possible

¹⁹The word “adversarial” is inherited from the online learning literature, i.e., *stochastic bandit* vs *adversarial bandit* (Auer et al., 2002). Adversary means there exists a virtual adversary (or, nature) who has complete control over the reward function and transition dynamics, and the adversary does not necessarily maintain a fully competitive relationship with the learning agent.

opponent (including but not limited to adversarial opponents), and the objective of the learning agent is to minimise its average loss during the learning process. Quantitatively, the loss is measured by how worse off the agent is compared to the best stationary policy in retrospect. The *expected regret* is thus different from Eq. (2.23) (unless in repeated games) and is written as

$$\text{Reg}_T = \frac{1}{T} \sup_{\pi \in \Pi} \mathbb{E}_{\pi} \left[\sum_{t=1}^T R_t(s_t^*, a_t^*) - R_t(s_t, a_t) \right] \quad (2.36)$$

where \mathbb{E}_{π} denotes the expectation over the sequence of (s_t^*, a_t^*) induced by the stationary policy π . Note that the reward function sequence and the transition kernel sequence are given by the adversary, and they are not influenced by the retrospective sequence (s_t^*, a_t^*) .

The goal is to find a no-regret algorithm that can satisfy $\text{Reg}_T \rightarrow 0$ as $T \rightarrow \infty$ with probability 1. A sufficient condition that ensures the existence of no-regret algorithms for online MDPs is the *oblivious* assumption – both the reward functions and transition kernels are fixed in advance, although they are unknown to the learning agent. This scenario is in contrast to the stateless setting in which no-regret is achievable, even if the opponent is allowed to be *adaptive/non-oblivious*: they can choose the reward function and transition kernels in accordance to (s_0, a_0, \dots, s_t) from the learning agent. In short, [Mannor and Shimkin \(2003\)](#); [Yu et al. \(2009\)](#) demonstrated that in order to achieve sub-linear regret, it is essential that the changing rewards are chosen obliviously. Furthermore, [Yadkori et al. \(2013\)](#) showed with the example of an online shortest path problem that there does not exist a polynomial-time solution (in terms of the size of the state-action space) where both the reward functions and transition dynamics are adversarially chosen, even if the adversary is *oblivious* (i.e., it cannot adapt to the other agent's historical actions). Most recently, [Cheung et al. \(2020\)](#); [Ortner et al. \(2020\)](#) investigated online MDPs where the transitional dynamics are allowed to change slowly (i.e., the total variation does not exceed a specific budget). Therefore, the majority of existing no-regret algorithms for online MDP focus on an oblivious adversary for the reward function

only. The nuances of different algorithms lie in whether the transitional kernel is assumed to be known to the learning agent and whether the feedback reward that the agent receives is in the full-information setting or in the bandit setting (i.e., one can only observe the reward of a taken action).

Two design principles can lead to no-regret algorithms that solve online MDPs with an oblivious adversary controlling the reward function. One is to leverage the local-global regret decomposition result (Even-Dar et al., 2005, 2009) [Lemma 5.4], which demonstrates that one can in fact achieve no regret globally by running a local regret-minimisation algorithm at each state; a similar result is observed for the CFR algorithm described in Eq. (2.32). Let $\mu^*(\cdot)$ denote the state occupancy induced by policy π^* ; we then obtain the decomposition result by

$$\text{Reg}_T = \sum_{s \in \mathbb{S}} \mu^*(s) \sum_{t=1}^T \underbrace{\sum_{a \in \mathbb{A}} (\pi^*(a | s) - \pi_t(a | s)) Q_t(s, a)}_{\text{local regret in state } s \text{ with reward function } Q_t(\cdot, \cdot)} . \quad (2.37)$$

Under full knowledge of the transition function and full-information feedback about the reward, Even-Dar et al. (2009) proposed the famous *MDP-Expert (MDP-E)* algorithm, which adopts *Hedge* (Freund and Schapire, 1997) as the regret minimiser and achieves $\mathcal{O}(\sqrt{\tau^3 T \ln |\mathbb{A}|})$ regret, where τ is the bound on the mixing time of MDP²⁰. For comparison, the theoretical lower bound for regret in a fixed MDP (i.e., no adversary perturbs the reward function) is $\Omega(\sqrt{|\mathbb{S}| |\mathbb{A}| T})$ ²¹ (Auer et al., 2009). Interestingly, Neu et al. (2017) showed that there in fact exists an equivalence between TRPO methods (Schulman et al., 2015) and MDP-E methods. Under bandit feedback, Neu et al. (2010) analysed *MDP-EXP3*, which achieves a regret bound of $\mathcal{O}(\sqrt{\tau^3 T |\mathbb{A}| \log |\mathbb{A}| / \beta})$, where β is a lower bound on the probability of reaching a certain state under a given policy. Later, Neu et al. (2014) removed the dependency on β and achieved $\mathcal{O}(\sqrt{T \log T})$ regret. One major advantage of local-global design principle is that it can work seamlessly with function approximation methods (Bertsekas and Tsitsiklis, 1996). For example, Yu et al. (2009)

²⁰Roughly, it can be considered as the time that a policy needs to reach the stationary status in MDPs. See a precise definition in Even-Dar et al. (2009) [Assumption 3.1].

²¹This lower bound has recently been achieved by Azar et al. (2017) up to a logarithmic factor.

eliminated the requirement of knowing the transition kernel by incorporating Q-learning methods; their proposed *Q-follow the perturbed leader (Q-FPL)* method achieved $\mathcal{O}(T^{2/3})$ regret. [Abbasi-Yadkori et al. \(2019\)](#) proposed *POLITEX*, which adopted a least square policy evaluation (LSPE) with linear function approximation and achieved $\mathcal{O}(T^{3/4} + \varepsilon_0 T)$ regret, in which ε_0 is the worst-case approximation error, and [Cai et al. \(2019a\)](#) used the same LSPE method. However, the proposed *OPPO* algorithm achieves $\mathcal{O}(\sqrt{T})$ regret.

Apart from the local-global decomposition principle, another design principle is to formulate the regret minimisation problem as an online linear optimisation (OLO) problem and then apply gradient-descent type methods. Specifically, since the regret in Eq. (2.37) can be further written as the inner product of $\text{Reg}_T = \sum_{t=1}^T \langle \mu^* - \mu_t, R_t \rangle$, one can run the gradient descent method by

$$\mu_{t+1} = \arg \max_{\mu \in \mathcal{U}} \left\{ \langle \mu, R_t \rangle - \frac{1}{\eta} \mathcal{D}(\mu | \mu_t) \right\}, \quad (2.38)$$

where $\mathcal{U} = \{\mu \in \Delta_{\mathbb{S} \times \mathbb{A}} : \sum_a \mu(s, a) = \sum_{s', a'} P(s|s', a') \mu(s', a')\}$ is the set of all valid stationary distributions²², where \mathcal{D} denotes a certain form of divergence and the policy can be extracted by $\pi_{t+1}(a|s) = \mu_{t+1}(s, a)/\mu(s)$. One significant advantage of this type of method is that it can flexibly handle different model constraints and extensions. If one uses Bregman divergence as \mathcal{D} , then online mirror descent is recovered ([Nemirovsky and Yudin, 1983](#)) and is guaranteed to achieve a nearly optimal regret for OLO problems ([Srebro et al., 2011](#)). [Zimin and Neu \(2013\)](#) and [Dick et al. \(2014\)](#) adopted a relative entropy for \mathcal{D} ; the subsequent *online relative entropy policy search (O-REPS)* algorithm achieves an $\mathcal{O}(\sqrt{\tau T \log(|\mathbb{S}||\mathbb{A}|)})$ regret in the full-information setting and an $\mathcal{O}(\sqrt{T|\mathbb{S}||\mathbb{A}| \log(|\mathbb{S}||\mathbb{A}|)})$ regret in the bandit setting. For comparison, the aforementioned MDP-E algorithm achieves $\mathcal{O}(\sqrt{\tau^3 T \ln |\mathbb{A}|})$ and $\mathcal{O}(\sqrt{\tau^3 T |\mathbb{A}| \log |\mathbb{A}| / \beta})$, respectively. When the transition dynamics are unknown to the agent, [Rosenberg and Mansour \(2019\)](#) extended O-REPS by incorporating the classic idea of *optimism in the face of uncertainty* in [Auer et al. \(2009\)](#), and the

²²In the online MDP literature, it is generally assumed that every policy reaches its stationary distribution immediately; see the policy mixing time assumption in [Yu et al. \(2009\)](#) [Assumption 2.1].

induced *UC-O-REPS* algorithm achieved $\mathcal{O}(|\mathbb{S}| \sqrt{|\mathbb{A}|T})$ regret.

2.3.5 Turn-Based Stochastic Games

An important class of games that lie in the middle of SG and EFG is the two-player zero-sum turn-based SG (2-TBSG). In TBSG, the state space is split between two agents, $\mathbb{S} = \mathbb{S}^1 \cup \mathbb{S}^2$, $\mathbb{S}^1 \cap \mathbb{S}^2 = \emptyset$, and in every time step, the game is in exactly one of the states, either \mathbb{S}^1 or \mathbb{S}^2 . Two players alternate taking turns to make decisions, and each state is controlled²³ by only one of the players $\pi^i : \mathbb{S}^i \rightarrow \mathbb{A}^i$, $i = 1, 2$. The state then transitions into the next state with probability $P : \mathbb{S}^i \times \mathbb{A}^i \rightarrow \mathbb{S}^j$, $i, j = 1, 2$. Given a joint policy $\boldsymbol{\pi} = (\pi^1, \pi^2)$, the first player seeks to maximise the value function $V^{\boldsymbol{\pi}(s)} = \mathbb{E} \left[\sum_{t=0}^{\infty} \gamma^t R(s_t, \boldsymbol{\pi}(s_t)) | s_0 = s \right]$, while the second player seeks to minimise it, and the saddle point is the NE of the game.

Research on 2-TBSG leads to many important finite-sample bounds, i.e., how many samples one would need before reaching the NE at a given precision, for understanding multi-agent learning algorithms. [Hansen et al. \(2013\)](#) extended [Ye \(2005, 2010\)](#)'s result from single-agent MDP to 2-TBSG and proved that the strongly polynomial time complexity of policy iteration algorithms also holds in the context of 2-TBSG if the payoff matrix is fully accessible. In the RL setting, in which the transition model is unknown, [Sidford et al. \(2018, 2020\)](#) provided a near-optimal Q-learning algorithm that computes an ε -optimal strategy with high-probability given $\mathcal{O}((1 - \gamma)^{-3} \varepsilon^{-2})$ samples from the transition function for each state-action pair. This result of polynomial-time sample complexity is remarkable since it was believed to hold for only single-agent MDPs. Recently, [Jia et al. \(2019\)](#) showed that if the transition model can be embedded in some state-action feature space, i.e., $\exists \psi_k(s')$ such that $P(s'|s, a) = \sum_{k=1}^K \phi_k(s, a) \psi_k(s')$, $\forall s' \in \mathbb{S}, (s, a) \in \mathbb{S} \times \mathbb{A}$, then the sample complexity of the two-player Q-learning algorithm towards finding an ε -NE is only linear to the number of features $\mathcal{O}(K / (\varepsilon^2(1 - \gamma)^4))$.

All the above works focus on the offline domain, where they assume that there exists an *oracle* that can unconditionally provide state-action transition samples. [Wei et al. \(2017\)](#) studied an online setting in an averaged-reward two-player SG.

²³Note that since the game is turned based, the Nash policies are deterministic.

Algorithm 1 A General Solver for Open-Ended Meta-Games

```

1: Initialise: the “high-level” policy set  $\mathbb{S} = \prod_{i \in \mathcal{N}} \mathbb{S}^i$ , the meta-game payoff
    $M, \forall S \in \mathbb{S}$ , and meta-policy  $\boldsymbol{\pi}^i = \text{UNIFORM}(\mathbb{S}^i)$ .
2: for iteration  $t \in \{1, 2, \dots\}$  do:
3:   for each player  $i \in \mathcal{N}$  do:
4:     Compute the meta-policy  $\boldsymbol{\pi}_t$  by meta-game solver  $\mathcal{S}(M_t)$ .
5:     Find a new policy against others by Oracle:  $S_t^i = \mathcal{O}^i(\boldsymbol{\pi}_t^{-i})$ .
6:     Expand  $\mathbb{S}_{t+1}^i \leftarrow \mathbb{S}_t^i \cup \{S_t^i\}$  and update meta-payoff  $M_{t+1}$ .
7:   terminate if:  $\mathbb{S}_{t+1}^i = \mathbb{S}_t^i, \forall i \in \mathcal{N}$ .
8: Return:  $\boldsymbol{\pi}$  and  $\mathbb{S}$ .

```

They achieved a polynomial sample-complexity bound if the opponent plays an optimistic best response, and a sublinear regret round against an arbitrary opponent.

2.3.6 Open-Ended Meta-Games

In solving real-world zero-sum games, such as GO or StarCraft, since the number of atomic pure strategy can be prohibitively large, one feasible approach instead is to focus on *meta-games*. A meta-game is constructed by simulating games that cover combinations of “high-level” policies in the policy space (e.g., “bluff” in Poker or “rushing” in StarCraft), with entries corresponding to the players’ empirical payoffs under a certain joint “high-level” policy profile; therefore, meta-game analysis is often called as *empirical game-theoretic analysis (EGTA)* ([Tuyls et al., 2018](#); [Wellman, 2006](#)). Analysing meta-games is a practical approach to tackling games that have huge pure-strategy space, since the number of “high-level” policies is usually far smaller than the number of pure strategies. For example, the number of tactics in StarCraft is at hundreds, compared to the vast raw action space of approximately 10^8 possibilities ([Vinyals et al., 2017](#)). Traditional game-theoretical concepts such as NE can still be computed on meta-games, but in a much more scalable manner; this is because the number of “higher-level” strategies in the meta-game is usually far smaller than the number of atomic actions of the underlying game. Furthermore, it has been shown that an ϵ -NE of the meta-game is in fact a 2ϵ -NE of the underlying game ([Tuyls et al., 2018](#)). Meta-games are often **open-ended** because in general there exists an infinite number of policies to play a real-world game, and, as new strategies will be discovered and added to agents’ strategy sets during training,

Table 2.3: Variations of Different Meta-Game Solvers

Method	\mathcal{S}	\mathcal{O}	Game type
Self-play (Fudenberg et al., 1998)	$[0, \dots, 0, 1]^N$	$\mathbf{Br}(\cdot)$	multi-player potential
GWFP (Leslie and Collins, 2006)	UNIFORM	$\mathbf{Br}_\varepsilon(\cdot)$	two-player zero-sum / potential
Double Oracle (McMahan et al., 2003)	NE	$\mathbf{Br}(\cdot)$	two-player zero-sum
PSRO _N (Lanctot et al., 2017)	NE	$\mathbf{Br}_\varepsilon(\cdot)$	two-player zero-sum
PSRO _{rN} (Balduzzi et al., 2019)	NE	Rectified $\mathbf{Br}_\varepsilon(\cdot)$	symmetric zero-sum
α -PSRO (Muller et al., 2019)	α -Rank	$\mathbf{PBr}(\cdot)$	multi-player general-sum

the dimension of the meta-game payoff table will also be expanded. If one writes the game evaluation engine as $\phi : \mathbb{S}^1 \times \mathbb{S}^2 \rightarrow \mathbb{R}$ such that if $S^1 \in \mathbb{S}^1$ beats $S^2 \in \mathbb{S}^2$, we have $\phi(S^1, S^2) > 0$, and $\phi < 0, \phi = 0$ refers to losses and ties, then the meta-game payoff can be represented by $M = \{\phi(S^1, S^2) : (S^1, S^2) \in \mathbb{S}^1 \times \mathbb{S}^2\}$. The sets of \mathbb{S}^1 and \mathbb{S}^2 can be regarded as, for example, two populations of deep neural networks (DNNs) and each S^1, S^2 is a DNN with independent weights. In such a context, the goal of learning in meta-games is to find \mathbb{S}^i and policy $\boldsymbol{\pi}^i \in \Delta(\mathbb{S}^i)$ such that the *exploitability* can be minimised, which is,

$$\text{Exploitability}(\boldsymbol{\pi}) = \sum_{i \in \{1, 2\}} \left[M^i(\mathbf{Br}^i(\boldsymbol{\pi}^{-i}), \boldsymbol{\pi}^{-i}) - M^i(\boldsymbol{\pi}) \right]. \quad (2.39)$$

It is easy to see Eq. (2.39) reaches zero when $\boldsymbol{\pi}$ is a NE.

A general solver for open-ended meta-games is the *policy space response oracle (PSRO)* (Lanctot et al., 2017). Inspired by the *double oracle* algorithm (McMahan et al., 2003), which leverages the *Benders' decomposition* (Benders, 1962) on solving large-scale linear programming for two-player zero-sum games, PSRO is a direct extension of double oracle (McMahan et al., 2003) by incorporating an RL

subroutine as an approximate best response. Specifically, one can write PSRO and its variations in Algorithm 1, which essentially involves an iterative two-step process of solving for the meta-policy first (e.g., Nash over the meta-game), and then based on the meta-policy, finding a new better-performing policy, against the opponent’s current meta-policy, to augment the existing population. The meta-policy solver, denoted as $\mathcal{S}(\cdot)$, computes a joint meta-policy profile $\boldsymbol{\pi}$ based on the current payoff M where different solution concepts can be adopted (e.g., NE). Finding a new policy is equivalent to solving a single-player optimisation problem given opponents’ policy sets \mathbb{S}^{-i} and meta-policies $\boldsymbol{\pi}^{-i}$, which are fixed and known. One can regard a new policy as given by an *Oracle*, denoted by \mathcal{O} . In two-player zero-sum cases, an oracle represents $\mathcal{O}^1(\boldsymbol{\pi}^2) = \{S^1 : \sum_{S^2 \in \mathbb{S}^2} \boldsymbol{\pi}^2(S^2) \cdot \phi(S^1, S^2) > 0\}$. Generally, Oracles can be implemented through optimisation subroutines such as RL algorithms. Finally, after a new policy is found, the payoff table M is expanded, and the missing entries are filled by running new game simulations. The above two-step process loops over each player at each iteration, and it terminates if no new policies can be found for any players.

Algorithm 1 is a general framework, with appropriate choices of meta-game solver \mathcal{S} and Oracle \mathcal{O} , it can represent solvers for different types of meta-games. We summarise variations of meta-game solvers in Table 2.3. For example, it is trivial to see that FP/GWFP is recovered when $\mathcal{S} = \text{UNIFORM}(\cdot)$ and $\mathcal{O}^i = \text{Br}^i(\cdot)/\text{Br}_\varepsilon^i(\cdot)$. The double oracle (McMahan et al., 2003) and PSRO methods (Lanctot et al., 2017) refer to the cases when the meta-solver computes NE. On solving symmetric zero-sum games (i.e., $\mathbb{S}^1 = \mathbb{S}^2$, and $\phi(S^1, S^2) = -\phi(S^2, S^1), \forall S^1, S^2 \in \mathbb{S}^1$), Balduzzi et al. (2019) proposed the *rectified best response* to promote behavioural diversity, written as

$$\text{Rectified Br}_\varepsilon(\boldsymbol{\pi}^2) \subseteq \arg \max_{S^1} \sum_{S^2 \in \mathbb{S}^2} \boldsymbol{\pi}^{2,*}(S^2) \cdot [\phi(S^1, S^2)]_+. \quad (2.40)$$

Through rectifying only the positive values on $\phi(S^1, S^2)$ in Eq. (2.40), player 1 is encouraged to amplify its strengths and ignore its weaknesses in finding a new

policy when it plays with the NE of player 2 during training; this turns out to be a critical component to tackle zero-sum games with strong non-transitive dynamics²⁴.

Double oracle and PSRO methods can only solve zero-sum games. When it comes to multi-player general-sum games, a new solution concept named α -Rank ([Omidshafiei et al., 2019b](#)) can be used to replace the intractable NE. The idea of α -Rank is built on the *response graph* of a game. On the response graph, each joint pure-strategy profile is a node, and a directed edge points from node $\sigma \in \mathbb{S}$ to node $S \in \mathbb{S}$ if 1) σ and S differ in only one single player's strategy, and 2) that deviating player, denoted by i , benefits from deviating from S to σ such that $M^i(\sigma) > M^i(S)$. The *sink strongly-connected components (SSCC)* nodes on the response graph that have only incoming edges but no outgoing edges are of great interest. To find those SSCC nodes, α -Rank constructs a random walk along the directed response graph, which can be equivalently described by a Markov chain, with the transition probability matrix \mathbf{C} being:

$$\mathbf{C}_{S,\sigma} = \begin{cases} \eta \frac{1 - \exp\left(-\alpha(M^k(\sigma) - M^k(S))\right)}{1 - \exp\left(-\alpha m(M^k(\sigma) - M^k(S))\right)} & \text{if } M^k(\sigma) \neq M^k(S) \\ \frac{\eta}{m} & \text{otherwise} \end{cases},$$

$$\mathbf{C}_{S,S} = 1 - \sum_{i \in \mathcal{N}} \mathbf{C}_{S,\sigma} \quad (2.41)$$

$\eta = (\sum_{i \in \mathcal{N}} (|S^i| - 1))^{-1}$, $m \in \mathbb{N}$, $\alpha > 0$ are three constants. Large α ensures the Markov chain is irreducible, and thus guarantees the existence and uniqueness of the α -Rank solution, which is the resulting unique stationary distribution $\boldsymbol{\pi}$ of the Markov chain, $\mathbf{C}^\top \boldsymbol{\pi} = \boldsymbol{\pi}$. The probability mass of each joint strategy in $\boldsymbol{\pi}$ can be interpreted as the longevity of that strategy during an evolution process ([Omidshafiei et al., 2019b](#)). The main advantage of α -Rank is that it is unique and its solution is P -complete even on multi-player general-sum games. α^α -Rank developed by [Yang](#)

²⁴Any symmetric zero-sum games consist of both transitive and non-transitive components ([Balduzzi et al., 2019](#)). A game is transitive if the ϕ can be represented by a monotonic rating function f such that performance on the game is the difference in ratings: $\phi(S^1, S^2) = f(S^1) - f(S^2)$, and it is non-transitive if ϕ satisfies $\int_{S^2 \in \mathbb{S}^2} \phi(S^1, S^2) \cdot dS^2 = 0$, meaning that winning against some strategies will be counterbalanced with losses against other strategies in the population.

et al. (2019a) computes α -Rank based on stochastic gradient methods such that there is no need to store the whole transition matrix in Eq. (2.41) before getting the final output of $\boldsymbol{\pi}$, this is particularly important when meta-games are prohibitively large in real-world domains.

When PSRO adopts α -Rank as the meta-solver, it is found that a simple best response fails to converge to the SSCC of a response graph before termination (Muller et al., 2019). To suit α -Rank, Muller et al. (2019) later proposed *preference-based best response oracle*, written as

$$\mathbf{PBr}^i(\boldsymbol{\pi}^{-i}) \subseteq \arg \max_{\sigma \in \mathbb{S}^i} \mathbb{E}_{S^{-i} \sim \boldsymbol{\pi}^{-i}} \left[\mathbb{1}[M^i(\sigma, S^{-i}) > M^i(S^i, S^{-i})] \right], \quad (2.42)$$

and the combination of α -Rank with $\mathbf{PBr}(\cdot)$ in Eq. (2.42) is called α -PSRO. Due to the tractability of α -Rank on general-sum games, the α -PSRO is credited as a generalised training approach for multi-agent learning.

2.4 Learning in General-Sum Games

Solving general-sum SGs entails an entirely different level of difficulty than solving team games or zero-sum games. In a static two-player normal-form game, finding the NE is known to be *PPAD*-complete (Chen and Deng, 2006).

2.4.1 Solutions by Mathematical Programming

To solve a two-player general-sum discounted stochastic game with discrete states and discrete actions, Filar and Vrieze (2012) [Chapter 3.8] formulated the problem as a nonlinear programme; the matrix form is written as follows:

$$\begin{aligned} \min_{\mathbf{V}, \boldsymbol{\pi}} f(\mathbf{V}, \boldsymbol{\pi}) &= \sum_{i=1}^2 \mathbf{1}_{|\mathbb{S}|}^T \left[V^i - (\mathbf{R}^i(\boldsymbol{\pi}) + \gamma \cdot \mathbf{P}(\boldsymbol{\pi}) V^i) \right] \\ \text{s.t.} \quad \text{(a)} \quad &\boldsymbol{\pi}^2(s)^T \left[\mathbf{R}^1(s) + \gamma \cdot \sum_{s'} \mathbf{P}(s'|s) V^1(s') \right] \leq V^1(s) \mathbf{1}_{|\mathbb{A}^1|}^T, \quad \forall s \in \mathbb{S} \\ &\left[\mathbf{R}^2(s) + \gamma \cdot \sum_{s'} \mathbf{P}(s'|s) V^2(s') \right] \boldsymbol{\pi}^1(s) \leq V^2(s) \mathbf{1}_{|\mathbb{A}^2|}^T, \quad \forall s \in \mathbb{S} \\ &\boldsymbol{\pi}^1(s) \geq \mathbf{0}, \quad \boldsymbol{\pi}^1(s)^T \mathbf{1}_{|\mathbb{A}^1|} = 1, \quad \forall s \in \mathbb{S} \\ &\boldsymbol{\pi}^2(s) \geq \mathbf{0}, \quad \boldsymbol{\pi}^2(s)^T \mathbf{1}_{|\mathbb{A}^2|} = 1, \quad \forall s \in \mathbb{S} \end{aligned} \quad (2.43)$$

where

- $\mathbf{V} = \langle V^i : i = 1, 2 \rangle$ is the vector of agents' values over all states, $V^i = \langle V^i(s) : s \in \mathbb{S} \rangle$ is the value vector for the i -th agent.
- $\boldsymbol{\pi} = \langle \pi^i : i = 1, 2 \rangle$ and $\pi^i = \langle \pi^i(s) : s \in \mathbb{S} \rangle$, where $\pi^i(s) = \langle \pi^i(a|s) : a \in \mathbb{A}^i \rangle$ is the vector representing the stochastic policy in state $s \in \mathbb{S}$ for the i -th agent.
- $\mathbf{R}^i(s) = [R^i(s, a^1, a^2) : a^1 \in \mathbb{A}^1, a^2 \in \mathbb{A}^2]$ is the reward matrix for the i th agent in state $s \in \mathbb{S}$. The rows correspond to the actions of the second agent, and the columns correspond to those of the first agent. With a slight abuse of notation, we use $\mathbf{R}^i(\boldsymbol{\pi}) = \mathbf{R}^i(\langle \pi^1, \pi^2 \rangle) = \langle \pi^2(s)^T \mathbf{R}^i(s) \pi^1(s) : s \in \mathbb{S} \rangle$ to represent the expected reward vector over all states under joint policy $\boldsymbol{\pi}$.
- $\mathbf{P}(s'|s) = [P(s'|s, \mathbf{a}) : \mathbf{a} = \langle a^1, a^2 \rangle, a^1 \in \mathbb{A}^1, a^2 \in \mathbb{A}^2]$ is a matrix representing the probability of transitioning from the current state $s \in \mathbb{S}$ to the next state $s' \in \mathbb{S}$. The rows represent the actions of the second agent, and the columns represent those of the first agent. With a slight abuse of notation, we use $\mathbf{P}(\boldsymbol{\pi}) = \mathbf{P}(\langle \pi^1, \pi^2 \rangle) = [\pi^2(s)^T \mathbf{P}(s'|s) \pi^1(s) : s \in \mathbb{S}, s' \in \mathbb{S}]$ to represent the expected transition probability over all state pairs under joint policy $\boldsymbol{\pi}$.

This is a nonlinear programme because the inequality constraints in the optimisation problem are quadratic in \mathbf{V} and $\boldsymbol{\pi}$. The objective function in Eq. (2.43) aims to minimise the TD error for a given policy $\boldsymbol{\pi}$ over all states, similar to the policy evaluation step in the traditional policy iteration method, and the constraints of (a) and (b) in Eq. (2.43) act as the policy improvement step, which satisfies the equation when the optimal value function is achieved. Finally, constraints (c) and (d) ensure the policy is properly defined.

Although the NE is proved to exist in general-sum SGs in the form of stationary strategies, solving Eq. (2.43) in the two-player case is notoriously challenging. First, Eq. (2.43) has a non-convex feasible region; second, only the global optimum²⁵ of Eq. (2.43) corresponds to the NE of SGs, while the common gradient-descent type of methods can only guarantee convergence to a local minimum. Apart

²⁵Note that in the zero-sum case, every local optimum is global.

from the efforts by [Filar and Vrieze \(2012\)](#), [Breton et al. \(1986\)](#) [Chapter 4] developed a formulation that has nonlinear objectives but linear constraints. Furthermore, [Dermed and Isbell \(2009\)](#) formulated the NE solution as multi-objective linear program. [Herings and Peeters \(2010\)](#); [Herings et al. \(2004\)](#) proposed an algorithm in which a *homotopic path* between the equilibrium points of N independent MDPs and the N -player SG is traced numerically. This approach yields a Nash equilibrium point of the stochastic game of interest. However, all these methods are tractable only in small-size SGs with at most tens of states and only two players.

2.4.2 Solutions by Value-Based Methods

A series of value-based methods have been proposed to address general-sum SGs. A majority of these methods adopt classic Q-learning ([Watkins and Dayan, 1992](#)) as a centralised controller, with the differences being what solution concept the central Q-learner should apply to guide the agents to converge in each iteration. For example, the Nash-Q learner in Eqs. (1.19 & 1.20) applies NE as the solution concept, the correlated-Q learner adopts correlated equilibrium ([Greenwald et al., 2003](#)), and the friend-or-foe learner considers both cooperative (see Eq. (2.1)) and competitive equilibrium (see Eq. (2.20)) ([Littman, 2001a](#)). Although many algorithms come with convergence guarantees, the corresponding assumptions are often overly restrictive to be applicable in general. When Nash-Q learning was first proposed ([Hu et al., 1998](#)), it required the NE of the SG be unique such that the convergence property could hold. Though strong, this assumption was still noted by [Bowling \(2000\)](#) to be insufficient to justify the convergence of the Nash-Q algorithm. Later, [Hu and Wellman \(2003\)](#) corrected her convergence proof by tightening the assumption even further; the uniqueness of the NE must hold for every single stage game encountered during state transitions. Years later, a strikingly negative result by [Zinkevich et al. \(2006\)](#) concluded that the entire class of value-iteration methods could be excluded from consideration for computing stationary equilibria, including both NE and correlated equilibrium, in general-sum SGs. Unlike those in single-agent RL, the Q values in the multi-agent case are inherently defective for reconstructing the equilibrium policy.

2.4.3 Solutions by Two-Timescale Analysis

In addition to the centralised Q-learning approach, decentralised Q-learning algorithms have recently received considerable attention because of their potential for scalability. Although independent learners have been accused of having convergence issues (Tan, 1993), decentralised methods have made substantial progress with the help of two-timescale stochastic analysis (Borkar, 1997) and its application in RL (Borkar, 2002).

Two-timescale stochastic analysis is a set of tools certifying that, in a system with two coupled stochastic processes that evolve at different speeds, if the fast process converges to a unique limit point for any particular fixed value of the slow process, we can, quantitatively, analyse the asymptotic behaviour of the algorithm as if the fast process is always fully calibrated to the current value of the slow process (Borkar, 1997). As a direct application, Leslie et al. (2003); Leslie and Collins (2005) noted that independent Q-learners with agent-dependent learning rates could break the symmetry that leads to the non-convergent limited cycles; as a result, they can converge almost surely to the NE in two-player collaboration games, two-player zero-sum games, and multi-player matching pennies. Similarly, Prasad et al. (2015) introduced a two-timescale update rule that ensures the training dynamics reach a stationary local NE in general-sum SGs if the critic learns faster than the actor. Later, Perkins et al. (2015) proposed a distributed actor-critic algorithm that enjoys provable convergence in solving static potential games with continuous actions. Similarly, Arslan and Yüksel (2016) developed a two-timescale variant of Q-learning that is guaranteed to converge to an equilibrium in SGs with weakly acyclic characteristics, which generalises potential games. Other applications include developing two-timescale update rules for training GANs (Heusel et al., 2017) and developing a two-timescale algorithm with guaranteed asymptotic convergence to the Stackelberg equilibrium in general-sum Stackelberg games.

2.4.4 Solutions by Policy-Based Methods

Convergence to NE via direct policy search has been extensively studied; however, early results were limited mainly by stateless two-player two-action games (Ab-

dallah and Lesser, 2008; Bowling, 2005; Bowling and Veloso, 2002; Conitzer and Sandholm, 2007; Singh et al., 2000b; Zhang and Lesser, 2010). Recently, GAN training has posed a new challenge, thereby rekindling interest in understanding the policy gradient dynamics of continuous games (Heusel et al., 2017; Mescheder et al., 2018, 2017; Nagarajan and Kolter, 2017).

Analysing gradient-based algorithms through dynamic systems (Shub, 2013) is a natural approach to yield more significant insights into convergence behaviour. However, a fundamental difference is observed when one attempts to apply the same analysis from the single-agent case to the multi-agent case because the combined dynamics of gradient-based learning schemes in multi-agent games do not necessarily correspond to a proper *gradient flow* – a critical premise for almost sure convergence to a local minimum. In fact, the difficulty of solving general-sum continuous games is exacerbated by the usage of deep networks with stochastic gradient descent. In this context, a key equilibrium concept of interest is the *local NE* (Ratliff et al., 2013) or *differential NE* (Ratliff et al., 2014), defined as follows.

Definition 10 (Local Nash Equilibrium). *For an N -player continuous game denoted by $\{\ell_i : \mathbb{R}^d \rightarrow \mathbb{R}\}_{i \in \{1, \dots, N\}}$ with each agent's loss ℓ_i being twice continuously differentiable, the parameters are $\mathbf{w} = (\mathbf{w}_1, \dots, \mathbf{w}_n) \in \mathbb{R}^d$, and each player controls $\mathbf{w}_i \in \mathbb{R}^{d_i}, \sum_i d_i = d$. Let $\boldsymbol{\xi}(\mathbf{w}) = (\nabla_{\mathbf{w}_1} \ell_1, \dots, \nabla_{\mathbf{w}_n} \ell_n) \in \mathbb{R}^d$ be the simultaneous gradient of the losses w.r.t. the parameters of the respective players, and let $\mathbf{H}(\mathbf{w}) := \nabla_{\mathbf{w}} \cdot \boldsymbol{\xi}(\mathbf{w})^\top$ be the $(d \times d)$ Hessian matrix of the gradient, written as*

$$\mathbf{H}(\mathbf{w}) = \begin{pmatrix} \nabla_{\mathbf{w}_1}^2 \ell_1 & \nabla_{\mathbf{w}_1, \mathbf{w}_2}^2 \ell_1 & \cdots & \nabla_{\mathbf{w}_1, \mathbf{w}_n}^2 \ell_1 \\ \nabla_{\mathbf{w}_2, \mathbf{w}_1}^2 \ell_2 & \nabla_{\mathbf{w}_2}^2 \ell_2 & \cdots & \nabla_{\mathbf{w}_2, \mathbf{w}_n}^2 \ell_2 \\ \vdots & & & \vdots \\ \nabla_{\mathbf{w}_n, \mathbf{w}_1}^2 \ell_n & \nabla_{\mathbf{w}_n, \mathbf{w}_2}^2 \ell_n & \cdots & \nabla_{\mathbf{w}_n}^2 \ell_n \end{pmatrix}$$

where $\nabla_{\mathbf{w}_i, \mathbf{w}_j}^2 \ell_k$ is the $(d_i \times d_j)$ block of 2nd-order derivatives. A differentiable NE for the game is \mathbf{w}^* if $\boldsymbol{\xi}(\mathbf{w}^*) = 0$ and $\nabla_{\mathbf{w}_i}^2 \ell_i \succ 0, \forall i \in \{1, \dots, N\}$; furthermore, this result is a local NE if $\det \mathbf{H}(\mathbf{w}^*) \neq 0$.

A recent result by Mazumdar and Ratliff (2018) suggested that gradient-based

algorithms can almost surely avoid a subset of local NE in general-sum games; even worse, there exist non-Nash stationary points. As a tentative treatment, [Balduzzi et al. \(2018a\)](#) applied *Helmholtz decomposition*²⁶ to decompose the game Hessian $\mathbf{H}(\mathbf{w})$ into a potential part plus a Hamiltonian part. Based on the decomposition, they designed a gradient-based method to address each part and combined them into *symplectic gradient adjustment (GDA)*, which is able to find all local NE for zero-sum games and a subset of local NE for general-sum games. More recently, [Chasnov et al. \(2019\)](#) separately considered the cases of 1) agents with oracle access to the exact gradient $\xi(\mathbf{w})$ and 2) agents with only an unbiased estimator for $\xi(\mathbf{w})$. In the first case, they provided asymptotic and finite-time convergence rates for the gradient-based learning process to reach the differential NE. In the second case, they derived concentration bounds guaranteeing with high probability that agents will converge to a neighbourhood of a stable local NE in finite time. In the same framework, [Fiez et al. \(2019\)](#) studied Stackelberg games in which agents take turns to conduct the gradient update rather than acting simultaneously and established the connection under which the equilibrium points of simultaneous gradient descent are Stackelberg equilibria in zero-sum games. [Mertikopoulos and Zhou \(2019\)](#) investigated the local convergence of no-regret learning and found local NE is attracting under gradient play if and only if a NE satisfies a property known as *variational stability*. This idea is inspired by the seminal notion of *evolutionary stability* observed in animal populations ([Smith and Price, 1973](#)).

Finally, it is worth highlighting that the above theoretical analysis of the performance of gradient-based methods on stateless continuous games cannot be taken for granted in SGs. The main reason is that the assumption on the differentiability of the loss function required in continuous games may not hold in general-sum SGs. As clearly noted by [Fazel et al. \(2018\)](#); [Mazumdar et al. \(2019a\)](#); [Zhang et al. \(2019d\)](#), even in the extreme setting of linear-quadratic games, the value functions are not guaranteed to be globally smooth (w.r.t. each agent's policy parameter).

²⁶This approach is similar in ideology to the work by [Candogan et al. \(2011\)](#), where they leverage the combinatorial Hodge decomposition to decompose any multi-player normal-form game into a potential game plus a harmonic game. However, their equivalence is an open question.

2.5 Learning in Games when $N \rightarrow +\infty$

As detailed in Section 1.4, designing learning algorithms in a multi-agent system with $N \gg 2$ is a challenging task. One major reason is that the solution concept, such as Nash equilibrium, is difficult to compute in general due to the curse of dimensionality of the multi-agent problem itself. However, if one considers a continuum of agents with $N \rightarrow +\infty$, then the learning problem becomes surprisingly tractable. The intuition is that one can effectively transform a many-body interaction problem into a two-body interaction problem (i.e., agent vs the population mean) via mean-field approximation.

The idea of mean-field approximation, which considers the behaviour of large numbers of particles where individual particles have a negligible impact on the system, originated from physics. Important applications include solving Ising models²⁷ (Kadanoff, 2009; Weiss, 1907), or more recently, understanding the learning dynamics of over-parameterised deep neural networks (Hu et al., 2019; Lu et al., 2020b; Sirignano and Spiliopoulos, 2020; Song et al., 2018). In the game theory and MARL context, mean-field approximation essentially enables one to think of the interactions between every possible permutation of agents as an interaction between each agent itself and the aggregated mean effect of the population of the other agents, such that the N -player game ($N \rightarrow +\infty$) turns into a “two”-player game. Moreover, under *the law of large numbers* and *the theory of propagation of chaos* (Gärtner, 1988; McKean, 1967; Sznitman, 1991), the aggregated version of the optimisation problem in Eq. (2.46) asymptotically approximates the original N -player game.

The assumption in the mean-field regime that each agent responds only to the mean effect of the population may appear rather limited initially; however, for many

²⁷An Ising model is a model used to study magnetic phase transitions under different system temperatures. In a 2D Ising model, one can imagine the magnetic spins are laid out on a lattice, and each spin can have one of two directions, either up or down. When the system temperature is high, the direction of the spins is chaotic, and when the temperature is low, the directions of the spins tend to be aligned. Without the mean-field approximation, computing the probability of the spin direction is a combinatorial hard problem; for example, in a 5×5 2D lattice, there are 2^{25} possible spin configurations. A successful approach to solving the Ising model is to observe the phase change under different temperatures and compare it against the ground truth.

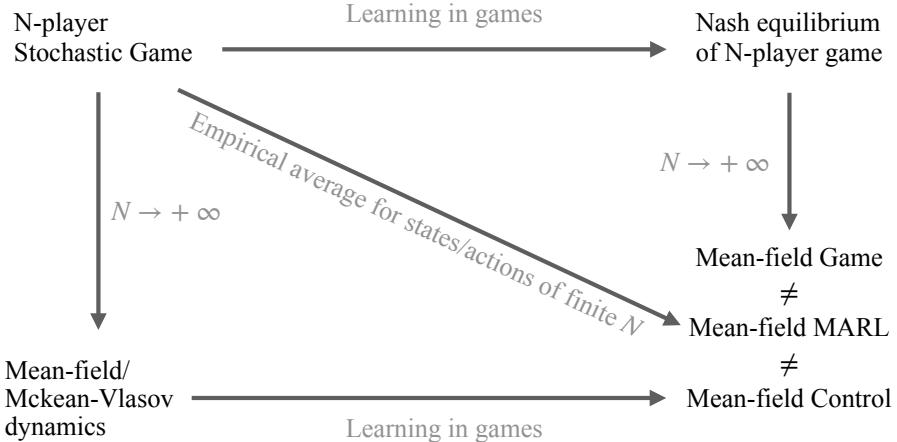


Figure 2.3: Relations of mean-field learning algorithms in games with large N .

real-world applications, agents often cannot access the information of all other agents but can instead know the global information about the population. For example, in high-frequency trading in finance (Cardaliaguet and Lehalle, 2018; Lehalle and Mouzouni, 2019), each trader cannot know every other trader's position in the market, although they have access to the aggregated order book from the exchange. Another example is real-time bidding for online advertisements (Guo et al., 2019; Iyer et al., 2014), in which participants can only observe, for example, the second-best prize that wins the auction but not the individual bids from other participants.

There is a subtlety associated with types of games in which one applies the mean-field theory. If one applies the mean-field type theory in non-cooperative²⁸ games, in which agents act independently to maximise their own individual reward, and the solution concept is NE, then the scenario is usually referred to as a *mean-field game (MFG)* (Guéant et al., 2011; Huang et al., 2006; Jovanovic and Rosenthal, 1988; Lasry and Lions, 2007). If one applies mean-field theory in cooperative games in which there exists a central controller to control all agents cooperatively to reach some Pareto optima, then the situation is usually referred to as *mean-field control (MFC)* (Andersson and Djehiche, 2011; Bensoussan et al., 2013), or *McKean-Vlasov dynamics (MKV)* control. If one applies the mean-field approximation to solve a standard SG through MARL, specifically, to factorise each agent's reward

²⁸Note that the word “non-cooperative” does not mean agents cannot collaborate to complete a task, it means agents cannot collude to form a coalition: they have to behave independently.

function or the joint-Q function, such that they depend only on the agent’s local state and the mean action of others, then it is called *mean-field MARL (MF-MARL)* ([Subramanian et al., 2020](#); [Yang et al., 2018b](#); [Zhou et al., 2019](#)).

Despite the difference in the applicable game types, technically, the differences among MFG/MFC/MF-MARL can be elaborated from the perspective of the order in which the equilibrium is learned (optimised) and the limit as $N \rightarrow +\infty$ is taken ([Carmona et al., 2013](#)). MFG learns the equilibrium of the game first and then takes the limit as $N \rightarrow +\infty$, while MFC takes the limit first and optimises the equilibrium later. MF-MARL is somewhat in between. The mean-field in MF-MARL refers to the empirical average of the states and/or actions of a *finite* population; N does not have to reach infinity, though the approximation converges asymptotically to the original game when N is large. This result is in contrast to the mean-field in MFG and MFC, which is essentially a probability distribution of states and/or actions of an *infinite* population (i.e., the McKean-Vlasov dynamics). Before providing more details, we summarise the relationships of MFG, MFC, and MF-MARL in Figure 2.3. Readers are recommended to revisit their differences after finishing reading the below subsections.

2.5.1 Non-cooperative Mean-Field Game

MFGs have been widely studied in different domains, including physics, economics, and stochastic control ([Carmona et al., 2018](#); [Guéant et al., 2011](#)). An intuitive example to quickly illustrate the idea of MFG is the problem of *when does the meeting start* ([Guéant et al., 2011](#)). For a meeting in the real world, people often schedule a calendar time t in advance, and the actual start time T depends on when the majority of participants (e.g., 90%) arrive. Each participant plans to arrive at τ^i , and the actual arrival time, $\tilde{\tau}^i = \tau^i + \sigma^i \varepsilon^i$, is often influenced by some uncontrolled factors $\sigma^i \varepsilon^i, \varepsilon^i \sim \mathcal{N}(0, 1)$, such as weather or traffic. Assuming all players are rational, they do not want to be later than either t or T ; moreover, they do not want to arrive too early and have to wait. The cost function of each individual can be written as $c^i(t, T, \tilde{\tau}^i) = \mathbb{E}[\alpha |\tilde{\tau}^i - t|_+ + \beta |\tilde{\tau}^i - T|_+ + \gamma |T - \tilde{\tau}^i|_+]$, where α, β, γ are constants. The key question to ask is when is the best time for an agent to arrive,

as a result, when will the meeting actually start, i.e., what is T ?

The challenge of the above problem lies in the coupled relationship between T and τ^i ; that is, in order to compute T , we need to know τ^i , which is based on T itself. Therefore, solving the time T is essentially equivalent to finding the fixed point, if it exists, of the stochastic process that generates T . In fact, T can be effectively computed through a two-step iterative process, and we denote as Γ^1 and Γ^2 . At Γ^1 , given the current²⁹ value of T , each agent solves their optimal arrival time τ^i by minimising their cost $R^i(t, T, \tilde{\tau}^i)$. At Γ^2 , agents calibrate the new estimate of T based on all τ^i values that were computed in Γ^1 . Γ^1 and Γ^2 continue iterating until T converges to a fixed point, i.e., $\Gamma^2 \circ \Gamma^1(T^*) = T^*$. The key insight is that the interaction with other agents is captured simply by the mean-field quantity. Since the meeting starts only when 90% of the people arrive, if one considers a continuum of players with $N \rightarrow +\infty$, T becomes the 90th quantile of a distribution, and each agent can easily find the best response. This result contrasts to the cases of a finite number of players, in which the ordered statistic is intractable, especially when N is large (but still finite).

Approximating an N -player SG by letting $N \rightarrow +\infty$ and letting each player choose an optimal strategy in response to the population's macroscopic information (i.e., the mean field), though analytically friendly, is not cost-free. In fact, MFG makes two major assumptions: 1) the impact of each player's action on the outcome is infinitesimal, resulting in all agents being identical, interchangeable, and indistinguishable; 2) each player maintains *weak interactions* with others only through a mean field, denoted by $L^i \in \Delta^{|\mathbb{S}||\mathbb{A}|}$, which is essentially a population state-action joint distribution

$$L^i = (\mu^{-i}(\cdot), \alpha^{-i}(\cdot)) = \lim_{N \rightarrow +\infty} \left(\frac{\sum_{j \neq i} \mathbf{1}(s^j = \cdot)}{N-1}, \frac{\sum_{j \neq i} \mathbf{1}(a^j = \cdot)}{N-1} \right) \quad (2.44)$$

where s^j and a^j player j 's local state³⁰ and local action. Therefore, for SGs that do

²⁹At time step 0, it can be a random guess. Since the fixed point exists, the final convergence result is irrelevant to the initial guess.

³⁰Note that in mean-field learning in games, the state is not assumed to be global. This is different from Dec-POMDP, in which there exists an observation function that maps the global state to the

not share the homogeneity assumption³¹ and weak interaction assumption, MFG is not an effective approximation. Furthermore, since agents have no identity in MFG, one can choose a representative agent (the agent index is thus omitted) and write the formulation³² of the MFG as

$$V(s, \pi, \{L_t\}_{t=0}^{\infty}) := \mathbb{E} \left[\sum_{t=0}^{\infty} \gamma^t R(s_t, a_t, L_t) \mid s_0 = s \right]$$

subject to $s_{t+1} \sim P(s_t, a_t, L_t), a_t \sim \pi_t(s_t)$. (2.45)

Each agent applies a local policy³³ $\pi_t : \mathbb{S} \rightarrow \Delta(\mathbb{A})$, which assumes the population state is not observable. Note that both the reward function and the transition dynamics depend on the sequence of the mean-field terms $\{L_t\}_{t=0}^{\infty}$. From each agent's perspective, the MDP is time-varying and is determined by all other agents.

The solution concept in MFG is a variant of the (Markov perfect) NE named the mean-field equilibrium, which is a pair of $\{\pi_t^*, L_t^*\}_{t \geq 0}$ that satisfies two conditions: 1) for fixed $L^* = \{L_t^*\}$, $\pi^* = \{\pi_t^*\}$ is the optimal policy, that is, $V(s, \pi^*, L^*) \geq V(s, \pi, L^*), \forall \pi, s$; 2) L^* matches with the generated mean field when agents follow π^* . The two-step iteration process in the meeting start-time example applied in MFG is then expressed as $\Gamma^1(L_t) = \pi_t^*$ and $\Gamma^2(L_t, \pi_t^*) = L_{t+1}$, and it terminates when $\Gamma^2 \circ \Gamma^1(L) = L = L^*$. Mean-field equilibrium is essentially a fixed point of MFG, its existence for discrete-time³⁴ discounted MFGs has been verified by [Saldi et al. \(2018\)](#) in the infinite-population limit $N \rightarrow +\infty$ and also in the partially observable setting ([Saldi et al., 2019](#)). However, these works consider the case

local observation for each agent.

³¹In fact, the homogeneity in MFG can be relaxed to allow agents to have (finite) different types ([Lacker and Zariphopoulou, 2019](#)), though within each type, agents must be homogeneous.

³²MFG is more commonly formulated in a continuous-time setting in the domain of optimal control, where it is typically composed by a backward *Hamilton-Jacobi-Bellman equation* (e.g., the Bellman equation in RL is its discrete-time counterpart) that describes the optimal control problem of an individual agent and a forward *Fokker-Planck equation* that describes the dynamics of the aggregate distribution (i.e., the mean field) of the population.

³³A general non-local policy $\pi(s, L) : \mathbb{S} \times \Delta^{|\mathbb{S}| \times |\mathbb{A}|} \rightarrow \Delta(\mathbb{A})$ is also valid for MFG, and it makes the learning easier by assuming L is fully observable.

³⁴The existence of equilibrium in continuous-time MFGs is widely studied in the area of stochastic control ([Cardaliaguet et al., 2015; Carmona and Delarue, 2013; Carmona et al., 2016, 2015b; Fischer et al., 2017; Huang et al., 2006; Lacker, 2015, 2018; Lasry and Lions, 2007](#)), though it may be of less interest to RL researchers.

where the mean field in MFG includes only the population state. Recently, [Guo et al. \(2019\)](#) demonstrated the existence of NE in MFG, taking into account both the population states and actions distributions. In addition, they proved that if Γ^1 and Γ^2 meet *small parameter conditions* ([Huang et al., 2006](#)), then the NE is unique in the sense of L^* . In terms of uniqueness, a common result is based on assuming monotonic cost functions ([Lasry and Lions, 2007](#)). In general, MFGs admit multiple equilibria ([Nutz et al., 2020](#)); the reachability of multiple equilibria is studied when the cost functions are anti-monotonic ([Cecchin et al., 2019](#)) or quadratic ([Delarue and Tchuendom, 2020](#)).

Based on the two-step fixed-point iteration in MFGs, various model-free RL algorithms have been proposed for learning the NE. The idea is that in the step Γ^1 , one can approximate the optimal π_t given L_t through single-agent RL algorithms³⁵ such as (deep) Q-learning ([Anahtarcı et al., 2019; Anahtarcı et al., 2020; Guo et al., 2019](#)), (deep) policy-gradient methods ([Elie et al., 2020; Guo et al., 2020; Subramanian and Mahajan, 2019; uz Zaman et al., 2020](#)), and actor-critic methods ([Fu et al., 2019; Yang et al., 2019b](#)). Then, in step Γ^2 , one can compute the forward L_{t+1} by sampling the new π_t directly or via fictitious play ([Cardaliaguet and Hadikhanloo, 2017; Elie et al., 2019; Hadikhanloo and Silva, 2019](#)). A surprisingly good result is that the sample complexity of both value-based and policy-based learning methods for MFG in fact shares the same order of magnitude as those of single-agent RL algorithms ([Guo et al., 2020](#)). However, one major subtlety of these learning algorithms for MFGs is how to obtain stable samples for L_{t+1} . For example, [Guo et al. \(2020\)](#) discovered that applying a softmax policy for each agent and projecting the mean-field quantity on an ε -net with finite cover help to significantly stabilise the forward propagation of L_{t+1} .

2.5.2 Cooperative Mean-Field Control

MFC maintains the same homogeneity assumption and weak interaction assumption as MFG. However, unlike MFG, in which each agent behaves independently, there is

³⁵Since agents in MFG are homogeneous, if the representative agent reaches convergence, then the joint policy is the NE. Additionally, given L_t , the MDP to the representative agent is stationary.

a central controller that coordinates all agents' behaviours in the context of MFC. In cooperative multi-agent learning, assuming each agent observes only a local state, the central controller maximises the aggregated accumulative reward:

$$\sup_{\boldsymbol{\pi}} \frac{1}{N} \sum_{i=1}^N \mathbb{E}_{\mathbf{s}_{t+1} \sim P, \mathbf{a}_t \sim \boldsymbol{\pi}} \left[\sum_t \gamma^t R^i(\mathbf{s}_t, \mathbf{a}_t) \mid \mathbf{s}_0 = \mathbf{s} \right]. \quad (2.46)$$

Solving Eq. (2.46) is a combinatorial problem. Clearly, the sample complexity of applying the Q-learning algorithm grows exponentially in N (Even-Dar and Mansour, 2003). To avoid the curse of dimensionality in N , MFC (Carmona et al., 2018; Gu et al., 2019) pushes $N \rightarrow +\infty$, and under the law of large numbers and the theory of propagation of chaos (Gärtner, 1988; McKean, 1967; Sznitman, 1991), the optimisation problem in Eq. (2.46), in the view of a representative agent, can be equivalently written as

$$\begin{aligned} & \sup_{\boldsymbol{\pi}} \mathbb{E} \left[\sum_t \gamma \tilde{R}(s_t, a_t, \mu_t, \alpha_t) \mid s_0 \sim \mu \right] \\ & \text{subject to } s_{t+1} \sim P(s_t, a_t, \mu_t, \alpha_t), a_t \sim \boldsymbol{\pi}_t(s_t, \mu_t). \end{aligned} \quad (2.47)$$

in which (μ_t, α_t) is the respective state and action marginal distribution of the mean-field quantity, $\mu_t(\cdot) = \lim_{N \rightarrow +\infty} \sum_{i=1}^N \mathbb{1}(s_t^i = \cdot)/N$, $\alpha_t(\cdot) = \sum_{s \in \mathbb{S}} \mu_t(s) \cdot \boldsymbol{\pi}_t(s, \mu_t)(\cdot)$, and $\tilde{R} = \lim_{N \rightarrow +\infty} \sum_i R^i/N$. The MFC approach is attractive not only because the dimension of MFC is independent of N , but also because MFC has shown to approximate the original cooperative game in terms of both game values and optimal strategies (Lacker, 2017; Motte and Pham, 2019).

Although the MFC formulation in Eq. (2.47) appears similar to the MFG formulation in Eq. (2.45), their underlying physical meaning is fundamentally different. As is illustrated in Figure 2.3, the difference is which operation is performed first: learning the equilibrium of the N -player game or taking the limit as $N \rightarrow +\infty$. In the fixed-point iteration of MFG, one first assumes L_t is given and then lets the (infinite) number of agents find the best response to L_t , while in MFC, one assumes an infinite number of agents to avoid the curse of dimensionality in cooperative

MARL and then finds the optimal policy for each agent from a central controller perspective. In addition, compared to mean-field NE in MFG, the solution concept of the central controller in MFC is the Pareto optimum³⁶, an equilibrium point where no individual can be better off without making others worse off. Finally, other differences between MFG and MFC can be found in [Carmona et al. \(2013\)](#).

In MFC, since the marginal distribution of states serves as an input in the agent's policy and is no longer assumed to be known in each iteration (in contrast to MFG), the dynamic programming principle no longer holds in MFC due to its non-Markovian nature ([Andersson and Djehiche, 2011](#); [Buckdahn et al., 2011](#); [Carmona et al., 2015a](#)). That is, MFC problems are inherently time-inconsistent. A counter-example of the failure of standard Q-learning in MFC can be found in [Gu et al. \(2019\)](#). One solution is to learn MFC by adding common noise to the underlying dynamics such that all existing theory on learning MDP with stochastic dynamics can be applied, such as Q-learning ([Carmona et al., 2019b](#)). In the special class of linear-quadratic MFCs, [Carmona et al. \(2019a\)](#) studied the policy-gradient method and its convergence, and [Luo et al. \(2019\)](#) explored an actor-critic algorithm. However, this approach of adding common noise still suffers from high sample complexity and weak empirical performance ([Gu et al., 2019](#)). Importantly, applying dynamic programming in this setting lacks rigorous verifications, leaving aside the measurability issues and the existence of a stationary optimal policy.

Another way to address the time inconsistency in MFCs is to consider an **enlarged** state-action space ([Djete et al., 2019](#); [Gu et al., 2019](#); [Laurière and Pironneau, 2014](#); [Pham and Wei, 2016, 2017, 2018](#)). This technique is also called “lift up”, which essentially means to lift up the state space and the action space into their corresponding probability measure spaces in which dynamic programming principles hold. For example, [Gu et al. \(2019\)](#); [Motte and Pham \(2019\)](#) proposed to lift the finite state-action space \mathbb{S} and \mathbb{A} to a compact state-action space embedded in Euclidean space denoted by $\mathcal{C} := \Delta(\mathbb{S}) \times \mathcal{H}$ and $\mathcal{H} := \{h : \mathbb{S} \rightarrow \Delta(\mathbb{A})\}$, and the

³⁶The Pareto optimum is a subset of NE.

optimal Q-function associated with the MFC problem in Eq. (2.47) is

$$Q_{\mathcal{C}}(\mu, h) = \sup_{\pi} \mathbb{E} \left[\sum_{t=0}^{\infty} \gamma^t \tilde{R}(s_t, a_t, \mu_t, \alpha_t) \middle| s_0 \sim \mu, u_0 \sim \alpha, a_t \sim \pi_t \right], \forall (\mu, h) \sim \mathcal{C}. \quad (2.48)$$

The physical meaning of \mathcal{H} is the set of all possible local policies $h : \mathbb{S} \rightarrow \Delta(\mathbb{A})$ over all different states. Note that after lift up, the mean-field term μ_t in π_t of Eq. (2.47) no longer exists as an input to h . Although the support of each h is $|\Delta(\mathbb{A})|^{\mathbb{S}}$, it proves to be the minimum space under which the Bellman equation can hold. The Bellman equation for $Q_{\mathcal{C}} : C \rightarrow \mathbb{R}$ is

$$Q_{\mathcal{C}}(\mu, h) = R(\mu, h) + \gamma \sup_{\tilde{h} \in \mathcal{H}} Q_{\mathcal{C}}(\Phi(\mu, h), \tilde{h}) \quad (2.49)$$

where R and Φ are the reward function and transition dynamics written as

$$R(\mu, h) = \sum_{s \in \mathbb{S}} \sum_{a \in \mathcal{A}} \tilde{R}(s, a, \mu, \alpha(\mu, h)) \cdot \mu(s) \cdot h(s)(a) \quad (2.50)$$

$$\Phi(\mu, h) = \sum_{s \in \mathbb{S}} \sum_{a \in \mathbb{A}} P(s, a, \mu, \alpha(\mu, h)) \cdot \mu(s) \cdot h(s)(a) \quad (2.51)$$

with $\alpha(\mu, h)(\cdot) := \sum_{s \in \mathbb{S}} \mu(s) \cdot h(s)(\cdot)$ representing the marginal distribution of the mean-field quantity in action. The optimal value function is $V^*(\mu) = \max_{h \in \mathcal{H}} Q_{\mathcal{C}}(\mu, h)$. Since both μ and h are probability distributions, the difficulty of learning MFC then changes to how to deal with continuous state and continuous action inputs to $Q_{\mathcal{C}}(\mu, h)$, which is still an open research question. Gu et al. (2020) tried to discretise the lifted space \mathcal{C} through ε -net and then adopted the kernel regression on top of the discretisation; impressively, the sample complexity of the induced Q-learning algorithm is independent of the number of agents N .

2.5.3 Mean-Field MARL

The scalability issue of multi-agent learning in non-cooperative general-sum games can also be alleviated by applying the mean-field approximation directly to each agent's Q-function (Subramanian et al., 2020; Yang et al., 2018b; Zhou et al., 2019). In fact, Yang et al. (2018b) was the first to combine mean-field theory with the

MARL algorithm. The full treatment will be presented in Chapter 5. The idea is to first factorise the Q-function using only the local pairwise interactions between agents (see Eq. (2.52)) and then apply the mean-field approximation; specifically, one can write the neighbouring agent's action a^k as the sum of the mean action \bar{a}^j and a fluctuation term $\delta a^{j,k}$, i.e., $a^k = \bar{a}^j + \delta a^{j,k}$, $\bar{a}^j = \frac{1}{N^j} \sum_k a^k$, in which $\mathcal{N}(j)$ is the set of neighbouring agents of the learning agent j with its size being $N^j = |\mathcal{N}(j)|$. With the above two processes, we can reach the mean-field Q-function $Q^j(s, a^j, \bar{a}^j)$ that approximates $Q^j(s, \mathbf{a})$ as follows

$$Q^j(s, \mathbf{a}) = \frac{1}{N^j} \sum_k Q^j(s, a^j, a^k) \quad (2.52)$$

$$\begin{aligned} &= \frac{1}{N^j} \sum_k \left[Q^j(s, a^j, \bar{a}^j) + \nabla_{\bar{a}^j} Q^j(s, a^j, \bar{a}^j) \cdot \delta a^{j,k} \right. \\ &\quad \left. + \frac{1}{2} \delta a^{j,k} \cdot \nabla_{\delta a^{j,k}}^2 Q^j(s, a^j, \bar{a}^{j,k}) \cdot \delta a^{j,k} \right] \end{aligned} \quad (2.53)$$

$$\begin{aligned} &= Q^j(s, a^j, \bar{a}^j) + \nabla_{\bar{a}^j} Q^j(s, a^j, \bar{a}^j) \cdot \left[\frac{1}{N^j} \sum_k \delta a^{j,k} \right] \\ &\quad + \frac{1}{2N^j} \sum_k \left[\delta a^{j,k} \cdot \nabla_{\delta a^{j,k}}^2 Q^j(s, a^j, \bar{a}^{j,k}) \cdot \delta a^{j,k} \right] \end{aligned} \quad (2.54)$$

$$\begin{aligned} &= Q^j(s, a^j, \bar{a}^j) + \frac{1}{2N^j} \sum_k R_{s,a^j}^j(a^k) \\ &\approx Q^j(s, a^j, \bar{a}^j). \end{aligned} \quad (2.55)$$

The second term in Eq. (2.54) is zero by definition, and the third term can be bounded if the Q-function is smooth, and it is neglected on purpose. The mean-field action \bar{a}^j can be interpreted as the empirical distribution of the actions taken by agent j 's neighbours. However, unlike the mean-field quantity in MFG or MFC, this quantity does not have to assume an infinite population of agents, which is more friendly for many real-world tasks, although a large N can reduce the approximation error between a^k and \bar{a}^j due to the law of large numbers. In addition, the mean-field term in MF-MARL does not include the state distribution, unlike MFG or MFC.

Based on the mean-field Q-function, one can write the Q-learning update as

$$\begin{aligned} Q_{t+1}^j(s, a^j, \bar{a}^j) &= (1 - \alpha) Q_t^j(s, a^j, \bar{a}^j) + \alpha \left[R^j + \gamma v_t^{j, \text{MF}}(s') \right] \\ v_t^{j, \text{MF}}(s') &= \sum_{a^j} \pi_t^j(a^j | s', \bar{a}^j) \cdot \mathbb{E}_{\bar{a}^j(\mathbf{a}^{-j}) \sim \pi_t^{-j}} [Q_t^j(s', a^j, \bar{a}^j)]. \end{aligned} \quad (2.56)$$

The mean action \bar{a}^j depends on $a^j, j \in \mathcal{N}(j)$, which itself depends on the mean action. The chicken-and-egg problem is essentially the time inconsistency that also occurs in MFC. To avoid coupling between a^j and \bar{a}^j , Yang et al. (2018b) proposed a filtration such that in each stage game $\{\mathbf{Q}_t\}$, the mean action \bar{a}_{\perp}^j is computed first using each agents' current policies, i.e., $\bar{a}_{\perp}^j = \frac{1}{N^j} \sum_k a^k, a^k \sim \pi_t^k$, and then given \bar{a}_{\perp}^j , each agent finds the best response by

$$\pi_t^j(a^j | s, \bar{a}_{\perp}^j) = \frac{\exp(\beta Q_t^j(s, a^j, \bar{a}_{\perp}^j))}{\sum_{a^j \in \mathbb{A}^j} \exp(\beta Q_t^j(s, a^j, \bar{a}_{\perp}^j))}. \quad (2.57)$$

For large β , the Boltzmann policy in Eq. (2.57) proves to be a contraction mapping, which means the optimal action a^j is unique given \bar{a}_{\perp}^j ; therefore, the chicken-and-egg problem is resolved³⁷.

MF-Q can be regarded as a modification of the Nash-Q learning algorithm (Hu and Wellman, 2003), with the solution concept changed from NE to mean-field NE (see the definition in MFG). As a result, under the same conditions, which include the strong assumption that there exists a unique NE at every stage game encountered, $\mathbf{H}^{\text{MF}} \mathbf{Q}(s, \mathbf{a}) = \mathbb{E}_{s' \sim p} [\mathbf{R}(s, \mathbf{a}) + \gamma \mathbf{v}^{\text{MF}}(s')]$ proves to be a contraction operator. Furthermore, the asymptotic convergence of the MF-Q learning update in Eq. (2.56) has also been established.

Considering only pairwise interactions in MF-Q may appear rather limited. However, it has been noted that the pairwise approximation of the agent and its neighbours, while significantly reducing the complexity of the interactions among agents, can still preserve global interactions between any pair of agents (Blume, 1993). In fact, such an approach is widely adopted in other machine learning do-

³⁷Coincidentally, the techniques of fixing the mean-field term first and adopting the Boltzmann policy for each agent were discovered by Guo et al. (2019) in learning MFGs at the same time.

mains, for example, factorisation machines (Rendle, 2010) and learning to rank (Cao et al., 2007). Based on MF-Q, Li et al. (2019a) solved the real-world taxi order dispatching task for Uber China and demonstrated strong empirical performance against humans. Subramanian and Mahajan (2019) extended MF-Q to include multiple types of agents and applied the method to a large-scale predator-prey simulation scenario. Ganapathi Subramanian et al. (2020) further relaxed the assumption that agents have access to exact cumulative metrics regarding the mean-field behaviour of the system, and proposed partially observable MF-Q that maintains a distribution to model the uncertainty regarding the mean field of the system.

Chapter 3

Emergent Population Dynamics from Million-Agent RL

Before introducing any novel developments on many-agent RL techniques, in this chapter, I choose first to demonstrate an application of studying emergent population dynamics from a group of intelligent agents powered by MARL. I scale the size of AI populations to millions. This study intends to show readers the potentials of many-agent RL techniques in biologies. Specifically, I put intelligent agents into a simulated predator-prey world and verify if the principles discovered in nature could also be used in understanding an artificially-created intelligent population, and vice versa. To achieve this, I simulate a large-scale predator-prey world, where the laws of the world are designed by only the findings or logical equivalence that have been discovered in nature. I endow the agents with intelligence based on deep MARL methods. In order to scale the population size up to millions of agents, a large-scale deep MARL training platform with redesigned experience buffer is proposed. The results show that the population dynamics of AI agents, driven only by each agent's individual self-interest, reveals a systematic pattern that is similar to the *Lotka-Volterra* model studied in population biology. I further disclose the emergent behaviours of collective adaptations in studying how the agents' grouping behaviours will change with the environmental resources. Both of the two findings could be explained by the *self-organisation* theory in nature.

3.1 Background and Motivation

By employing the modelling power of deep learning, single-agent reinforcement learning (RL) has started to display, even surpass, human-level intelligence on a wide variety of tasks, ranging from playing the games of Labyrinth (Mnih et al., 2016), Atari (Mnih et al., 2015), and Go (Silver et al., 2016) to other tasks such as continuous control on locomotions (Lillicrap et al., 2015), text generation (Yu et al., 2017), and neural architecture design (Zoph and Le, 2016). Very recently, multi-agent RL algorithms have further broadened the use of RL and demonstrated their potentials in the setting where both of the agents' incentives and economical constraints exist. For example, the studies (de Vries et al., 2016; Lazaridou et al., 2016; Mordatch and Abbeel, 2017; Wang et al., 2016) have shown that with different multi-agent cooperative learning environments, the compositional language naturally emerges. Researchers (Foerster et al., 2017b; Peng et al., 2017a; Usunier et al., 2016) have also demonstrated that multiple agents can be trained to play the combat game in StarCraft, and the agents have mastered collaborative strategies that are similar to those of experienced human players. Nonetheless, all of the aforementioned RL systems so far have been limited to less than tens of agents, and the focuses of their studies are rather in the optimisation of a micro and individual level policy. Macro-level studies about the resulting collective behaviours and dynamics emerging from a large population of AI agents remain untouched.

Yet, on the other hand, real-world populations exhibit specific orders and regularity on collective behaviours: honey bees use specific waggle dances to transmit signals, a trail of ants transfers food by leaving chemical marks on the routes, V-shaped formations of bird flocks during migration, or particular sizes of fish schools in the deep ocean. Even human beings can easily show ordered macro-dynamics, for example, the rhythmical audience applause after the concerts, or the periodical human waves in the fanatical football game. A stream of research on the theory of *self-organisation* (Ashby, 1991) explores a new approach to explaining the emergence of order in nature. In fact, the self-organising dynamics appear in many other disciplines of natural sciences (Bak, 2013). The theory of *self-organisation* sug-

gests that the ordered global dynamics, no matter how complex, are induced from repeated interactions between local individual parts of a initially disordered system, without external supervisions or interventions. Such a concept has proven critical in many fields of natural science (Camazine, 2003; Kauffman, 1993; Sumpter, 2006).

As once the ancient philosopher *Lucretius* (Palmer, 2014) said:

“A designing intelligence is necessary to create orders in nature”;

an interesting question for us is to understand what kinds of ordered macro-dynamics, if any, that a community of artificially-created agents would possess when they are together put into the natural context. In this chapter, we fill the research gap by conducting an empirical study on the above questions. We aim to understand whether the principles, *e.g.*, self-organisation theory (Ashby, 1991), that are developed in the real world could also be applied to understanding an AI population. In order to achieve this, we argue that the key to this study is to have a clear methodology of introducing the micro-level intelligence; therefore, we simulate a predator-prey world where each individual AI agent is endowed with intelligence through a large-scale deep RL framework. The population size is scaled up to a million level. To maximise the generality, the laws of the predator-prey world are designed by only incorporating the natural findings or logic equivalence; miscellaneous potential dynamics can thus be studied.

We first study the macro-dynamics of the population size for both the predators and preys, and then we investigate the emergence of one of the most fundamental collective behaviours – grouping. In particular, we compare the statistics and dynamics of the intelligent population with the theories and models from real-world biological studies. Interestingly, we find that the artificial predator-prey ecosystem, with individual intelligence incorporated, reaches an ordered pattern on dynamics that is similar to what the *Lotka-Volterra* model (Lotka, 1925) indicates in population biology. Also, we discover the emergence of the collective adaptations on grouping behaviours when the environment changes. Both of the two findings can be well explained based on the *self-organisation* theory. Moreover, the proposed million-agent RL platform could serve as the initial steps to understand the be-

haviours of large-scale AI populations, driven by deep RL algorithms. It could potentially open up an exciting research direction of understanding AI population by linking the findings from the AI world with the natural science principles from the real world, thus contributing to the research frontiers, for example, smart city (Nam and Pardo, 2011) and swarm intelligence (Kennedy, 2006).

While our subject is a computerised artifact, our work is also related to the research conducted in natural sciences. The theory of *self-organisation* proposed in (Ashby, 1991) serves as a fundamental way of thinking to understand the emergence of orders in nature (even though Physicists tend to challenge this theory because *The Second Law of Thermodynamics* (Boltzmann, 1974) states that the total level of disorders in an isolated system can never decrease over time). *Self-organisation* theory believers think that the global ordered dynamics of a system can originate from numerous interactions between local individuals that are initially disordered, with no needs for external interventions. The theory predicts the existence of the ordered dynamics in the population.

In fact, the self-organising phenomena have been observed in multiple fields in natural sciences (Camazine, 2003; Kauffman, 1993; Sumpter, 2006). For example, in population biology, one important discovery is the ordered harmonic dynamics of the population sizes between predators and preys (*e.g.*, the lynx and snowshoe hare (Gilpin, 1973)), which is summarised into the *Lotka-Volterra* model (Lotka, 1925). It describes the fact that there is a 90° lag in the phase space between the population sizes of predators and preys (more details are discussed later in Section 3.4.1). Even though explainable via the *self-organisation* theory, the *Lotka-Volterra* models are summarised based on the statistics from the ecological field studies. There is essentially no learning process or individual intelligence involved. In this work, we chose a different approach by incorporating the individual intelligence into the population dynamics studies. Each agent is endowed with the intelligence to make its own decision rather than is considered homogeneous and rule-based. Our intention is to find out whether an AI population still creates ordered dynamics such as the *Lotka-Volterra* equations, if so, whether the dynamics is explainable

from the perspective of the *self-organisation* theory.

In multiple disciplines of natural sciences spanning from zoology, psychology, to economy (Dunbar, 2016; Guillen, 2000; Sumpter, 2006), one of the most fundamental thus important collective behaviours to study is: grouping – a population of units aggregate together for collective decision-making. Grouping is believed to imply the emergence of sociality and to induce other collective behaviours (Javarone and Marinazzo, 2016). In studying the grouping behaviours, traditional approaches include setting up a game with rigid and reductive predefined interactive rules for each agent and then conduct simulations based on the ad-hoc game (Fryxell et al., 2007; Inada and Kawachi, 2002; Niwa, 1994). Rule-based games might work well on biological organisms that inherit the same characteristics from their ancestors; however, they show limits on studying the large-scale heterogeneous agents (Boccaletti et al., 2006). In contrast to rule-based games with no learning process at all, here we investigate the formation of grouping behaviours on a million-level AI populations driven by RL algorithms. Our intention is to find out how the grouping behaviours in AI population emerge and change *w.r.t.* the environmental factors such as the food resources, and if any other collective behaviours emerge from grouping.

3.2 Design of the Predator-Prey World

In this chapter, we try to understand: 1) whether the AI population creates any ordered patterns on population dynamics, and 2) the dynamics of the collective grouping behaviours. Predator-prey interaction is one fundamental relationship observed in nature. Here we intend to simulate a predator-prey world with million-level agents (shown in Figure 3.1). The world is designed to be easily adaptable to incorporate other environmental complexity to investigate the miscellaneous dynamics as well as collective behaviours of AI population where each individual agent is driven by pure self-interest.

3.2.1 The Axioms Observed in Nature

To avoid introducing any specific rules that could harm the generality of the observed results, we design the laws of the world by only considering those fac-

tual findings or logical equivalence that have been observed in the natural system ([Sumpter, 2006](#), Section 5). We regard those laws as the *axioms* of studying population dynamics and collective behaviours. Here we briefly review the axioms accepted, and refer the corresponding natural evidence to the Table 3.1. Note that these axioms should not be treated separately; instead, we consider how the combination of these different axioms could produce and affect collective dynamics.

- (i). **Positive Feedback.** Positive feedback enhances particular behaviours through reinforcement. It helps spread the information of a meaningful action quickly between individuals.
- (ii). **Negative Feedback.** Negative feedback leads to homeostasis. It helps stabilise the collective behaviours produced in favour of the positive feedback from going to extremes.
- (iii). **Individual Variation.** Individual variation is essential to guarantee the continued explorations of new solutions to the same problem within a population.
- (iv). **Response Threshold.** Response threshold is the threshold beyond which individuals will change their behaviours as a response to the stimulus.
- (v). **Redundancy.** Redundancy ensures functional continuity of the whole population even when a catastrophic event happens.
- (vi). **Synchronisation.** Synchronisation is a special kind of positive feedback in time rather than space. An example would be how individuals with unique frequency of applause affect the crowd's frequency in a concert.
- (vii). **Selfishness.** Individuals always tend to maximise their own utility. One will not behave altruistically for others until it can benefit more from behaving collectively than acting alone.

Table 3.1: The axioms adopted in the simulated predator-prey world (also see [Sumpter \(2006\)](#)[Section 5]).

Axiom	Description	Example in Nature
Positive Feedback	Positive feedback enhances particular behaviours through reinforcement. It helps spread the information of a meaningful action quickly between the individuals.	One backup in nature comes from the observations from ants (Bonabeau et al., 1997 ; Wilson et al., 1971). When an ant discovers a food source through a particular search trail, the path to the food will soon serve as the trigger for positive feedback through which other ants start to follow.
Negative Feedback	Negative feedback leads to homeostasis. It helps stabilise the collective behaviours produced in favour of the positive feedback from going to extremes.	In the ant's case (Bonabeau et al., 1997 ; Wilson et al., 1971), as the population size of ants is limited, with an increasing number of ants forage the food from outside, the distribution of ants between food sources will be stable.
Individual Variation	Individual variation is of the essence to guarantee the continued explorations of new solutions to the same problem within a population.	One evidence comes from the honey bees (Jeanne, 1988 ; Pankiw and Page Jr, 2000). Social insects as honey bees have evolved to be highly variable in the directional sense, response to sucrose, level of focus in food collection in order to ensure the diversification in ways of food collection, otherwise one single food resource will be depleted quickly.
Response Threshold	Response threshold is the threshold beyond which individuals will change their behaviours as a response to the stimulus.	An evidence found in nature is that bumblebees will start to fan so as to cool down the hive when the temperature inside goes above a threshold level (Weidenmüller, 2004).
Redundancy	Redundancy ensures functional continuity of the whole population even when a catastrophic event happens.	In the kingdom of bees, if the community suffers a drastic reduction in the number of worker bees, younger bees will soon replace their positions to guarantee that the whole community functions well (Seeley, 2009).
Synchronisation	Synchronisation is a special kind of positive feedback in time rather than space.	An example would be how individuals with unique frequency of applause affect the crowd's frequency in a concert. Empirical evidence includes audience applause that is often achieved through adjustments by individuals having a unique frequency among the local average.
Selfishness	Selfishness means that agents should always maximise their own utility.	Easily observable in nature. For example, the decision making of humans in most times tend to be selfish.

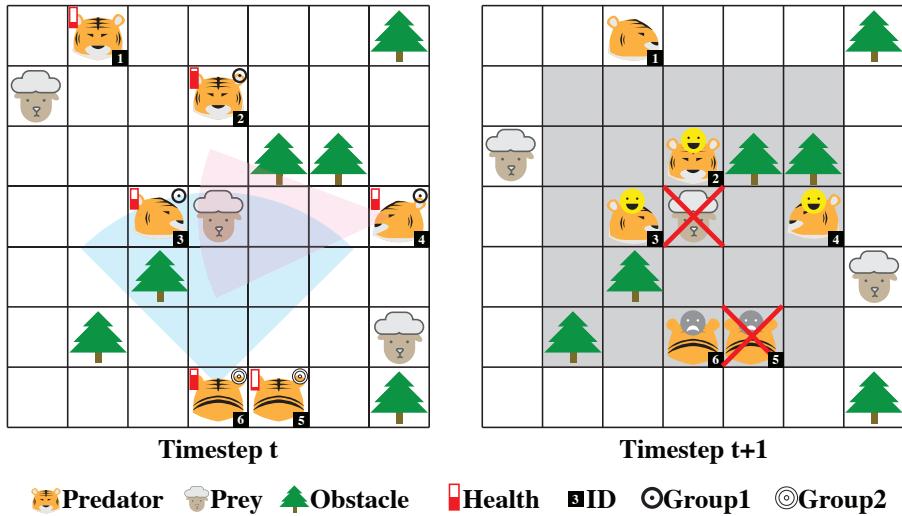


Figure 3.1: Illustration of the predator-prey world. In the 2D world, there exist preys, predators, and obstacles. Predators hunt the prey so as to survive from starvation. The reward is the proportion of prey that the predator obtains. Each predator has its own health bar and limited eyesight view. Predators can form a group to hunt the prey so that the chance of capturing can increase, but this also means that the captured prey will be shared among all group members. When there are multiple groups targeting the same prey, the largest group within the capture radius will win. In this example, predators $\{2, 3, 4\}$ form a group and win the prey over the group $\{5, 6\}$. Predator 5 soon dies due to starvation.

3.2.2 Realisation of the Axioms

We realise the predator-prey world via designing a *Stochastic Game*. We list the detailed rules of the game and its corresponding axiom.

3.2.2.1 Studying Population Dynamics

In the predator-prey world (see Figure 3.1), the goal for the predator species is to survive in the ecosystem and procreate their next generations (Axiom (vii)). Positions of predator/prey/obstacles in the world are all initialised randomly at the beginning. The environment is considered under an infinite horizon. While the population of both preys and predators can be boosted by breeding offsprings (Axiom (v)), they, however, face the hazards of either being hunted as preys, or dying of starvation as predators (Axiom (ii)). To realise the idea of starvation, we make the health status of predator decrease with time by a constant factor, which can also be restored by capturing and eating preys (Axiom (i)). Predators are assumed to have an infinite appetite; the logical equivalence in nature is that a predator normally

can store food resource for future survival. Each predator can have unique characteristics, *e.g.*, identity vector, eyesight and health status (Axiom (iii)). The unique characteristics of each agent represent the diversity of the population. Each individual agent makes independent decisions, and can behave differently even given the same scenario. Predators can form a group to increase the chance of capturing a prey (Axiom (i) & (iv)). Group members are visible to predators within its view. Suppose a single agent chooses the action of “join a group”. In that case, the environment will randomly select a group within its view, and the agent will become a member of that group until it decides to “leave the current group” afterwards. Note that a single predator may hunt for the prey alone as well as hunt as a group member. As illustrated in Figure 3.1, each prey is assigned a square capture area with a capture radius ρ , which reflects the difficulty of being hunted (Axiom (iv)). Groups of predators, or singles, will only be able to hunt the prey if they manage to stay within the capture radius. Apart from the capture radius, another parameter, the capture threshold k ($k=0,1,2,\dots$), also reflects the capturing difficulty of each prey (Axiom (iv)). Within the capture area, only meeting the threshold will a group of predators become a valid candidate. When there are multiple valid candidate groups targeting the same prey, the group with the largest group size will be the winner, which mimics jungle law. When a group wins over other candidates, all the members in that group will share the prey equally (Axiom (ii)). The trade-off here is, in the pursuit of preys, grouping is encouraged as large group can help increase the probability of capturing a prey; however, huge group size will also be inhibited due to the lower proportion of prey each group member obtains from the sharing (Axiom (vii)).

3.2.2.2 Studying Grouping Behaviours

Considering *synchronisation* of Axiom (vi) and *selfishness* of Axiom (vii), we incorporate the second type of prey that can be captured by an individual predator alone, which means we set the capture threshold k to 1 for that species. An analogy here is to think of tigers as the predators, sheep as the prey that can be hunted by a group of predators, and rabbits as the preys that can be captured by a single predator.

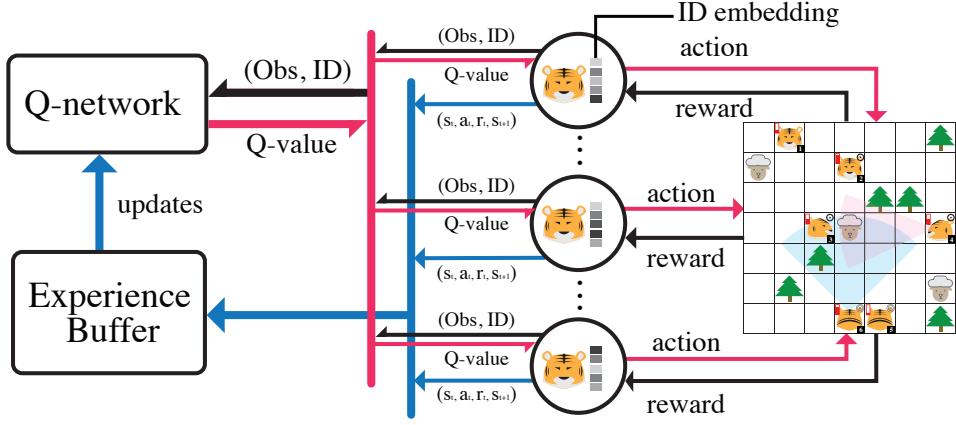


Figure 3.2: Million-agent Q-learning System in the Predator-prey World.

These two kinds of preys can be considered as an abstraction of individual reward and grouping reward, respectively. Predators have to make a decision to either join a group for hunting the sheep or conduct hunting the rabbit by itself in order to maximise its long-term reward and the probability of survival (Axiom (i),(ii),(vii)), which introduces a trade-off between acting alone and collaborating with others. We keep alternating the environments by feeding these two kinds of preys one after another (Axiom (vi)) and examine the dynamics of grouping behaviours. To emphasise the dynamics of grouping behaviours and also to avoid the influences from the systematic preferences for grouping as a result of the changing population size, we keep the population size of predators fixed by endowing them with eternal longevity, which can also be considered as a short-term observation during which there is little change of the predator population size. Under the environment, each agent's optimal strategy continuously varies over time, and the predator population has to learn to adapt their collective strategy correspondingly.

3.3 The AI Population Driven by MARL

In the designed predator-prey world, we build AI population under the multi-agent deep RL setting. Formally, the multi-agent Markov decision process (or, stochastic game) is denoted by $\{\mathcal{S}, \mathcal{A}, \mathcal{T}, \mathcal{R}, \mathcal{O}, \gamma, \rho_0, N\}$. \mathcal{S} denotes the set of true environmental states, and $\rho_0(\mathcal{S})$ denotes the initial state distribution. At each time step, each agent $i \in \{1, \dots, N\}$ in the predators population (they have to hunt preys to sur-

vive) takes an action $a^i \in \mathcal{A}$ where \mathcal{A} is the valid action space. The joint actions $\mathbf{a} \in \mathcal{A}^N$ induce a transition of the environment based on the transition function between states, $\mathcal{T} : \mathcal{S} \times \mathcal{A}^N \rightarrow \mathcal{S}$. The reward function is defined by $\mathcal{R} : \mathcal{S} \times \mathcal{A}^N \rightarrow \mathbb{R}^N$, and $\gamma \in [0, 1)$ denotes the discount factor. The environment is partially-observed; each agent can observe $o^i \in \mathcal{O}(s, a^i)$. An agent gains “intelligence” by learning a stochastic policy $\pi_\theta^i(a^i|s^i = o^i)$ that could maximise its expected cumulative reward in the predator-prey environment, *i.e.*, $\theta^* := \text{argmax}_\theta \mathbb{E}_{(s, \mathbf{a})} [\sum_{t=0}^{\infty} \gamma^t \mathcal{R}_t^i]$. The action-value function is defined by $Q^{\pi^i}(s_t, a_t) = \mathbb{E}_{s_{t+1:\infty}, a_{t+1:\infty}} [\sum_{l=0}^{\infty} \gamma^l \mathcal{R}_{t+l}^i | s_t, a_t]$. Considering the exploration in the action space, ε -greedy methods can be applied on selecting the action, $\pi_\theta^i(a^i|s^i) = \varepsilon\text{-greedy}(Q^{\pi^i}(s^i, a^i))$.

The action space \mathcal{A} includes {forward, backward, left, right, rotate left, rotate right, stand still, join a group, and leave a group}. It is considered invalid if a predator takes the “join a group” action as a group member already, takes the “leave a group” action as a single individual, or tries to cross the map borders. The environment will not settle invalid actions. Within the horizon of each individual agent, there are five channels for the observation o^i . The observation $O_t^i \in \mathbb{R}^{m \times n \times 5}$ is dependent on the agent’s current position and orientation. The agent’s eyesight ranges up to a distance limit towards the grids ahead and the grids to the left and right. The type of object (predators/preys/obstacles/blank areas) on the map occupy the first three channels: raw RGB pixels. The fourth channel is an indicator of whether or not that object is a group member. The fifth channel is the health status $h \in \mathbb{R}$ if the object is an agent, otherwise padded with zero. Each agent is assigned with a unique identity embedding $v^i \in \mathbb{R}^5$, together with the local observation, it makes up the state for each agent $s^i = (o^i, v^i)$. Individual agent is supposed to make independent decisions, and behave differently based on its local observation as well as ID embeddings as the inputs of policy π_θ^i .

3.3.1 The Implementation of Intelligent Agents

We designed a multi-agent RL platform with environmental optimisations in TensorFlow ([Abadi et al., 2016](#)) to make the training of million-agent learning feasible. To the best of our knowledge, back in 2017, this platform was the first to introduce

Algorithm 2 Million-agent Q-learning (in the case of population dynamics in Section 3.2.2)

```

1: Initialise agent's Q-network  $\pi^i$ , agent's identity  $v^i$ .
2: Randomly initialise the environment  $s \sim \rho_0(\mathcal{S})$ .
3: for time step=1,2,..., do
4:   Procreate predators/preys offsprings with random positions.
5:   for agent i=1,2,...,n do
6:     Compute the local observation features  $\mathcal{O}(i)$ .
7:     Compute the identity embedding  $\mathcal{I}(i)$ .
8:     Compute the current state for agent:  $s_t^i = (\mathcal{O}(i), \mathcal{I}(i))$ .
9:     Take action  $a_t^i \sim \pi_\theta^i(a^i|s^i) = \epsilon\text{-greedy}(Q^{\pi^i}(s_t^i, a_t^i))$ .
10:    Apply action  $a_t^i$ , and get reward  $r_t^i$ , next state  $s_{t+1}^i$ .
        Within the capture radius of each prey, the group of predators meeting the threshold will become valid candidate, the candidate with largest group size will be the final winner, and the reward are shared among the group members equally.
11:    Store tuple  $< s_t^i, a_t^i, s_{t+1}^i, r_t^i >$  in the experience buffer.
12:   end for
13:   if  $|\mathcal{B}| \geq$  batch size then
14:     Sample a mini-batch from  $\mathcal{B}$ .
15:     Update the parameters of Q-function w.r.t. the loss:
16:     
$$(r_t^j + \gamma \max_{a' \in \mathcal{A}} Q^\pi(s_{t+1}^j, a') - Q^\pi(s_t^j, a_t^j))^2$$

17:   end if
18:   Clear experience buffer  $\mathcal{B}$ .
19:   Decay the health of predators who starve.
20:   Reward (the group of) predator(s) who win the preys.
21:   Remove the dead predators and preys from the map.
22: end for
  
```

the training environment that enables simulating millions of agents driven by deep RL algorithms.

In particular, our setting is implemented through “centralised training with independent execution”. This is a natural paradigm for a large set of computationally tractable multi-agent problems. In the training stage, agents update the centralised Q-value function approximated by a deep neural network: $Q^\pi(s^i, a^i) = Q((o^i, v^i), a^i)$. However, each individual agent must rely on its local observation and unique identity to make independent decisions during the execution time. Apart from the standard setting of Q-learning (Watkins and Dayan, 1992) and deep Q-learning (Mnih et al., 2013), here we introduce a special experience buffer that aims to maximise the GPU efficiency and also mitigate the non-stationary issue in the

off-policy learning. At each time step, all agents contribute their experienced transitions $(s_t^i, a_t^i, r_t^i, s_{t+1}^i)$ to the buffer, as shown in Figure 3.2. We collect all the agents' experience of one time step in parallel and then update the Q-network using the experience simultaneously. This significantly increases the utilisation of the GPU memory, and is essential to the million-agent training. Based on the experience from the buffer, the Q-network is updated as:

$$Q^\pi(s_t^i, a_t^i) \leftarrow Q^\pi(s_t^i, a_t^i) + \alpha \left[r_t^i + \gamma \max_{a' \in \mathcal{A}} Q^\pi(s_{t+1}^i, a') - Q^\pi(s_t^i, a_t^i) \right]. \quad (3.1)$$

It is worth mentioning that the experience buffer in Figure 3.2 stores the experience from the agents only for the current time step; this is markedly different from the replay buffer that is commonly used in the traditional DQN where the buffer maintains a first-in-first-out queue across different time steps. Using the off-policy replay buffer will typically lead to the non-stationarity issue for multi-agent learning tasks (Lowe et al., 2017a). On the other hand, (Mnih et al., 2015) introduced the replay buffer to disrupt the auto-correlations between consecutive examples. In our million-agent RL setting, the experiences are sampled concurrently from millions of agents. Each individual agent has different states and policies; therefore, there are naturally no strong auto-correlations between the training examples. Moreover, it is unlikely that the unwanted feedback loops arise since one single agent's decision will hardly dominate the sampled experiences. The results further testify the robustness of our design of the experience buffer. See Algorithm 2 for the pesudo-code of the population dynamics example described in Section 3.2.2.

3.4 Experiments and Findings

Two sets of experiments – understanding population dynamics & collective behaviours – have been conducted. In particular, we intend to understand the kinds of ordered dynamics and emergent collective behaviours that the AI population could create in the predator-prey world similar to the natural context. The detailed parameter settings of the predator-prey world can be found in Table 3.2. The codes are published at <https://github.com/geek-ai/1m-agents>.

Table 3.2: Parameters settings for the predator-prey world. Code is released at <https://github.com/geek-ai/1m-agents>.

SETTINGS	VALUE	DESCRIPTION
WIDTH DEFAULT	1000	THE WIDTH OF THE MAP.
HEIGHT DEFAULT	1000	THE HEIGHT OF THE MAP.
BATCH SIZE DEFAULT	32	THE BATCH SIZE OF THE PROCESS OF TRAINING.
AGENT NUMBER DEFAULT	10000	THE INITIAL NUMBER OF AGENTS.
PIG MAX NUMBER DEFAULT	5000	THE INITIAL NUMBER OF PREY-PIG.
RABBIT MAX NUMBER DEFAULT	3000	THE INITIAL NUMBER OF PREY-RABBIT.
AGENT INCREASE RATE DEFAULT	0.001	THE BIRTH RATE OF THE AGENT.
PIG INCREASE RATE DEFAULT	0.001	THE BIRTH RATE OF THE PREY-PIG.
RABBIT INCREASE RATE DEFAULT	0.001	THE BIRTH RATE OF THE PREY-RABBIT.
REWARD RADIUS PIG DEFAULT	7	THE REWARD RADIUS THRESHOLD OF THE PREY-PIG.
REWARD RADIUS RABBIT DEFAULT	2	THE REWARD RADIUS THRESHOLD OF THE PREY-RABBIT.
REWARD THRESHOLD PIG DEFAULT	3	THE REWARD THRESHOLD OF THE PREY-PIG.
AGENT EMBEDING DIM DEFAULT	5	THE DIMENSION OF THE AGENT EMBEDDING.
DAMAGE PER STEP DEFAULT	0.01	THE DECREASE HEALTH OF AGENT PER STEP.

3.4.1 The Emergent Population Dynamics

We first study the population dynamics with a community of predators and preys by tracking the population size of each species over time. Specifically, we initialise 10,000 predators and 5,000 preys randomly scattered over a map of size $1,000 \times 1,000$. All predators' health status is set to 1.0 initially and decays by 0.01 at each time step. In two comparing settings, preys' birth rates are set to 0.006 and 0.01, respectively. The Q-network has two hidden layers, each with 32 hidden units, interleaved with sigmoid non-linear layers, which then project to 9-dimensional outputs, one for each potential action. During training, the predators learn in an off-policy RL scheme, with exploratory parameter $\epsilon = 0.1$.

Surprisingly, we find that the AI population reveals a systematic pattern when measuring the population dynamics. As shown in Figure 3.3, the population sizes of both predators and preys reach a dynamic equilibrium where both curves present a wax-and-wane shape, but with a 90° lag in the phase, *i.e.*, the crest of one is aligned with the trough of the other. The underlying logic of such ordered dynamics could be that when the predators' population grows because they learn to know how to hunt efficiently, as a consequence of more preys being captured, the preys' population shrinks, which will later cause the predators' population to also shrink due to the lack of food supply, and with the help of fewer predators, the population of preys will recover from the shrinkage and start to regrow. Such logic drives the

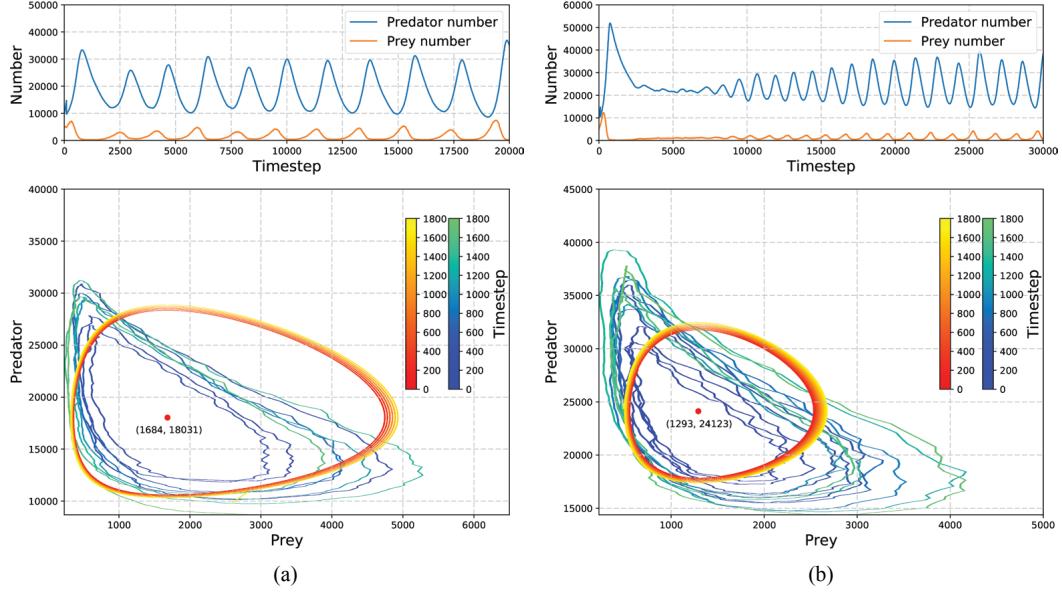


Figure 3.3: Population dynamics in both the time space (1st row) and the phase space (2nd row). The orange circles denote the theoretical solutions to the *Lotka-Volterra* equation, with the red spot as the equilibrium. The green-blue circles denote the simulation results. **a)**: The simulated birth rate of preys is 0.006. Fitted LV model: $\alpha = 0.0067, \beta = 3.75 \times 10^{-7}, \delta = 6.11 \times 10^{-7}, \gamma = 0.001$. **b)**: The simulated birth rate of preys is 0.01. Fitted LV model: $\alpha = 0.0086, \beta = 3.57 \times 10^{-7}, \delta = 9.47 \times 10^{-7}, \gamma = 0.0012$, where α in the LV model represents the birth rate.

2-D contour of population sizes (see the green-blue traits in the 2nd row in Figure 3.3) into harmonic cycles, and the circle patterns become stable with the increasing level of intelligence agents acquired from the RL algorithm. As shown later in the ablation study, enabling individual intelligence is the key to observing these ordered patterns in the population dynamics.

In fact, the population dynamics possessed by AI agents are consistent with the *Lotka-Volterra* (LV) model studied in biology (shown by the orange traits in Figure 3.3). In population biology, the LV model ([Lotka, 1925](#)) describes a *Hamiltonian* system with two-species interactions, *e.g.*, predators and preys. In the LV model, the population size of predators q and of preys p change over time based on the following pair of nonlinear differential equations:

$$\frac{1}{p} \frac{dp}{dt} = \alpha - \beta q, \quad \frac{1}{q} \frac{dq}{dt} = \delta p - \gamma. \quad (3.2)$$

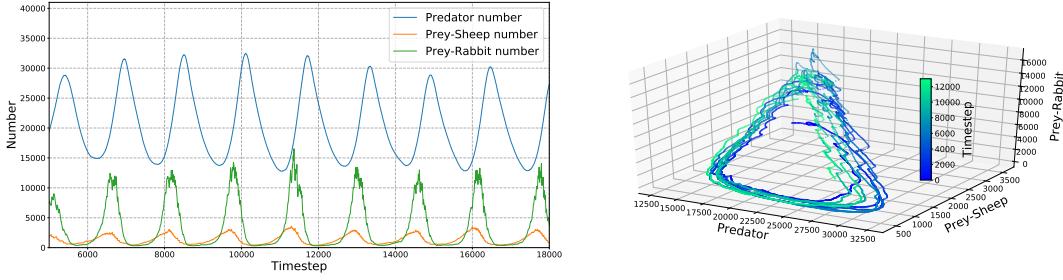


Figure 3.4: Population dynamics in the time space and the phase space. A new type of prey (green line) is introduced, which can be captured by a single agent. The AI population shows ordered dynamics in the 3-D phase space.

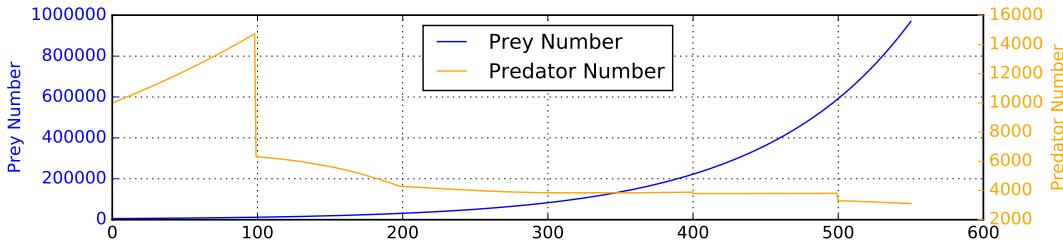


Figure 3.5: Population dynamics with the learning function of AI population disabled. The simulation follows the same setting as Figure 3.3(b). No ordered dynamics are found any more.

The preys are assumed to have an affluent food resource and thus can reproduce exponentially with rate α , until meeting predation, which is proportional to the rate at which the predators and the prey meet, represented by βq . The predators have an exponential decay in the population due to natural death denoted by γ . Meanwhile, they can also boost the population by hunting the prey, represented by δp . The solution to the equations is a harmonic function (wax-and-wane shaped) with the population size of predators lagging that of preys by 90° in the phase. On the phase space plot, it shows as a series of periodical circle $V = -\delta p + \gamma \ln(p) - \beta q + \alpha \ln(q)$, with V dependent on initial conditions. In other words, which equilibrium cycle to reach depends on where the ecosystem starts. Similar patterns on the population dynamics might indicate that an AI population's orders are induced from the same logic as the ecosystem that LV model describes. However, the critical difference here is that, unlike the LV equations that model the observed macro-dynamics directly, we start from a microcosmic point of view – the AI population is only driven by the self-interest (powered by RL) of the individual agent, and then reaching the macroscopic principles. To further test the robustness of our findings, we per-

form an ablation study on three of the most important factors that we think are critical to the generation of the ordered dynamics. First, we analyse whether the observed pattern is restricted by the specific settings of the predator-prey world. We expose the predator models, which are trained in the environment where the birth rate of preys is 0.006 in Figure 3.3(a), into a new environment where the birth rate of preys is 0.01. Figure 3.3(b) shows that after a period of time for adjustment, the predators adapt to the new environment, and the AI agents as a whole manage to maintain the patterns. Second, we break the binary predator-prey relationships by introducing a second type of prey that does not require group hunting. As shown in Figure 3.4, in the case of three species which the LV model may find challenging to analyse, we can still observe the ordered harmonic circles in 3-D space. Third, we investigate the role of individual intelligence by disabling the learning function in the setting of Figure 3.3(b). Figure 3.5 shows that the AI population does not possess any ordered dynamics anymore if each individual agent's intelligence is disabled. As such, the whole ecosystem explodes with an exponentially-increasing amount of preys and the extinction of predators. The reason why predator goes extinct is that the increased birth rate of preys leads to new distributions on the states, thus the observations; consequently, the predators' original optimal policy becomes suboptimal in the new environment. Given that the number of preys increases exponentially, and the map size is limited, the sheep will soon cover all the blank spaces, and the predators can barely aggregate any valid groups for hunting and finally die of starvation.

3.4.2 The Emergent Grouping Behaviours

Next, we investigate the dynamics of the collective grouping behaviours. In particular, we intend to determine the relationship between environmental food resources and the proportion of the predators that participate in the group hunting, which we refer to as the “group proportion”. In the face of two kinds of preys (one requires group hunting and the other does not), the predators have to decide to either join a group for hunting a sheep or hunt a rabbit itself alone. We conduct two experiments with the predator population size equalling ten thousand and two million, the map

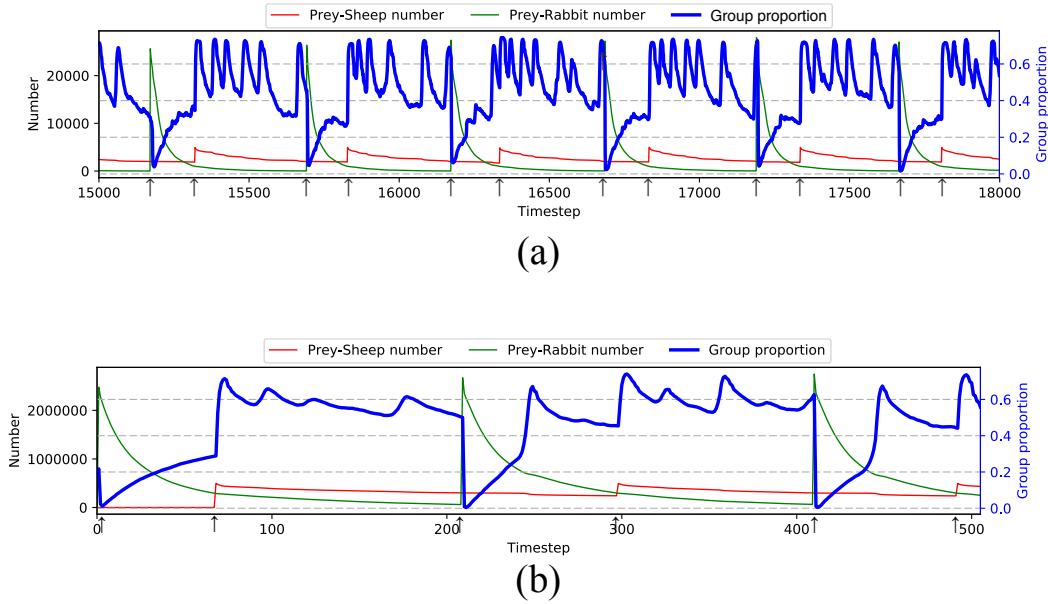


Figure 3.6: **a)** Grouping proportion in the predator-prey world where two kinds of preys are fed alternatively. ↑ points out the time step that preys are fed. It indicates that when the number of the prey sheep (that requires group hunting) increases, the proportion of groups in the AI population increases, and adapting to grouping becomes a collective behaviour. Vice versa for the case when the prey rabbit are fed. **b)** The same experiment on a two-million AI population.

size equalling $10^3 \times 10^3$ and $10^4 \times 10^4$ respectively. Acting like a “zoo-keeper”, we supplement the number of preys to a fixed amount if the number drops below a certain threshold. For each supplement, we alternate the types of preys to feed in. Suppose the number of species A is below the threshold, we supply species B. The setting of Q-network is the same as in the study on population dynamics.

As the preys are alternatively fed, the predator’s policy needs to react correspondingly to the new environment so as to survive. As shown in Figure 3.6(a) and 3.6(b), the moment right after the rabbits are fed into the environment, the proportion of groups drastically drop down to nearly 0. Predators collectively behave to be selfish rather than to be altruistic to the group. With the number of rabbits being captured, the proportion of grouping behaviours increases mildly again, and meets a spike soon after the sheep are fed into the environment, and reaches another dynamic equilibrium. In a highly-variable environment, the population of predators show the intelligence of adapting their hunting strategies collectively without any external supervisions or controls.

3.4.3 Connection to Self-Organising Theory

Judging from the ordered patterns of the AI population in the predator-prey world, we have reasons to agree with *Lucretius* that a designing intelligence is necessary to create orders in nature. In fact, in understanding the emergence of orders in a system, the theory of *self-organisation* proposed in (Ashby, 1991) considers that the global ordered dynamics of a system can spontaneously originate from numerous interactions between local individuals that are initially disordered, with no need of external interventions. The theory predicts the existence of the ordered dynamics from numerous local interactions between the individuals and the system. This could potentially explain the ordered patterns observed on our AI population that has been tested. Meanwhile, according to the theory, the created order is independent of the complexity of the individual involved. For example, the *Lotka-Volterra* dynamics also hold for other natural systems such as herbivores and plants, or parasites and hosts. Even though the LV models are based on a set of equations with fixed interaction terms, while our findings depend on intelligent agents driven by the consistent learning process, the generalisation of the resulting dynamics onto an AI population still leads us to expect a more general law that could unify the artificially created agents with the population we have studied in the natural sciences for a long time.

Arguably, in contrast to the *self-organisation* theory, reductionist scientists hold a different view that order can only be created by transferring it from external systems. A typical example is *The Second Law of Thermodynamics* (Boltzmann, 1974) stating that the total entropy (the level of disorder) will always increase over time in a *closed* system. Such an idea has widely been accepted, particularly in physics where quantitative analysis is feasible. However, we argue that our findings from the AI population do not go against this law. RL-based agents are not exceptions simply because the environment they “live” in are not *closed*. Whenever a system can exchange matter with its environment, an entropy decrease of that system (orders emerge) is still compatible with the second law. Further discussions on *entropy and life* (Schrodinger, 1943) go beyond this work, and deserve future work.

3.5 Chapter Summary

In this chapter, I conducted an empirical study on an AI population by simulating a predator-prey world where each agent was empowered by deep RL techniques. The number of agents is scaled up to millions. I found that the AI population possessed the ordered population dynamics consistent with the *Lotka-Volterra* model in ecology. I also discovered the emergent collective adaptations when the environmental resources changed over time. Importantly, both of the findings could be well explained by the *self-organisation* theory from natural sciences.

In the future, I will conduct further experiments on our million-agent RL platform by involving the ideas of leadership, cannibalism, and irrationality for discovering other profound natural principles in the full deep MARL-driven population. In return, I expect my findings could also enlighten an interesting research direction of interpreting the RL-based AI population using the natural science principles developed in the real world, and apply the AI population driven by RL for applications like smart cities or swarm intelligence.

Chapter 4

Many-Agent Policy Evaluation: α^α -Rank

In this chapter, I study the policy evaluation problem in the many-agent settings. Since computing the classic solution concept, such as Nash equilibrium, is *PPAD*-hard in even two-player cases and the solution may not be unique, a critical problem left is designing a new solution concept and tractable algorithm that can evaluate many-player general-sum games. Specifically, I introduce a new type of solution concept called α^α -Rank. α^α -Rank is built on α -Rank, a recently introduced graph-based solution concept that has been proposed as an effective alternative solution to evaluating/ranking joint policy profiles in large scale multi-agent systems. α -Rank claimed tractability through a polynomial-time implementation with respect to the total number of pure strategy profiles. In the following sections, I first re-investigate such claim and conjecture that solving α -Rank is in fact *NP*-hard. Specifically, I demonstrate that due to the need of constructing an exponentially large Markov chain, α -Rank is infeasible beyond a small finite number of agents. I substantiate these claims by adopting the amount of dollars spent as a non-refutable evaluation metric. Realising such scalability issue, I present α^α -Rank: a stochastic implementation of α -Rank with a double oracle mechanism allowing for reductions in joint strategy spaces. Our α^α -Rank does not need to save an exponentially-large transition matrix, and can terminate early under required precision. Although theoretically, our method exhibits similar worst-case complexity guarantees compared to

α -Rank, it allows us, for the first time, to practically conduct large-scale multi-agent evaluations. On $10^4 \times 10^4$ random matrices, we achieve a $1000\times$ speed reduction. Furthermore, we also show successful results on large joint strategy profiles with a maximum size in the order of $\mathcal{O}(2^{25})$ (≈ 33 million joint strategies) – a setting not evaluable by α -Rank using a reasonable computational budget.

4.1 Background and Motivation

Scalable policy evaluation and learning have been long-standing challenges in many-agent reinforcement learning (MARL) with two difficulties obstructing progress. First, joint-strategy spaces exponentially explode when many strategic decision-makers are considered. Second, the underlying game dynamics may exhibit cyclic behaviour (e.g., Rock-Paper-Scissor game) rendering appropriate evaluation criteria non-trivial. Focusing on the second challenge, much work in multi-agent systems followed a game-theoretic treatment proposing fixed-points, e.g., Nash (Nash et al., 1950) equilibrium, as potentially valid evaluation metrics (Yang et al., 2018b; Zhang et al., 2019a). Though appealing, such measures are normative only when prescribing behaviours of perfectly rational agents – an assumption rarely met in reality (Grau-Moya et al., 2018; Leibfried et al., 2017; Wen et al., 2019a,c). In fact, many game dynamics have been proven not to converge to any fixed-point equilibria (Hart and Mas-Colell, 2003; Viossat, 2007; Yang et al., 2018c), but rather to limit cycles (Bowling and Veloso, 2001a; Palaiopanos et al., 2017). Apart from these challenges, solving for a Nash equilibrium even for “simple” settings, e.g., two-player games are known to be *PPAD*-complete (Chen and Deng, 2005) – a demanding complexity class when it comes to computational requirements.

To address some of the above limitations, Omidshafiei et al. (2019a) recently proposed α -Rank as a graph-based game-theoretic solution to multi-agent evaluation. α -Rank adopts Markov Conley Chains to highlight the presence of cycles in game dynamics, and attempts to compute stationary distributions as a mean for strategy profile ranking. In a novel attempt, the authors reduce multi-agent evaluation

to computing a stationary distribution of a Markov chain. Namely, consider a set of N agents each having a strategy pool of size k , a Markov chain is, first, defined over the graph of joint strategy profiles with a transition matrix $\mathbf{T} \in \mathbb{R}^{k^N \times k^N}$, and then a stationary distribution $\mathbf{v} \in \mathbb{R}^{k^N}$ is computed solving: $\mathbf{T}\mathbf{v} = \mathbf{v}$. The probability mass in \mathbf{v} then represents the ranking of joint-strategy profile.

Extensions of α -Rank have been developed in various instances. [Rowland et al. \(2019\)](#) adapted α -Rank to model games with incomplete information. [Muller et al. \(2019\)](#) combined α -Rank with the policy search space oracle (PSRO) ([Lanc-tot et al., 2017](#)) and claimed their method to be a *generalised* training approach for multi-agent learning. Unsurprisingly, these work inherit the same claim of tractability from α -Rank. For example, the abstract in [Muller et al. \(2019\)](#) reads “ α -Rank, which is unique (thus faces no equilibrium selection issues, unlike Nash) and *tractable* to compute in general-sum, *many-player* settings.”

In this work, we contribute to refining the claims made in α -Rank dependent on its input type. We thoroughly argue that α -Rank exhibits a prohibitive computational and memory bottleneck that is hard to remedy even if payoff matrices were provided as inputs. We measure such a restriction using money spent as a non-refutable metric to assess α -Rank’s validity scale. With this in mind, we then present a stochastic solver that we title α^α -Rank as a scalable and memory-efficient alternative. Our method reduces memory constraints, and makes use of the oracle mechanism for reductions in joint strategy spaces. This, in turn, allows us to run large-scale multi-player experiments, including evaluations on self-driving cars and Ising models where the maximum size involves tens of millions of joint strategies.

4.2 Preliminary: α -Rank

In α -Rank, strategy profiles of N agents are evaluated through an evolutionary process of mutation and selection. Initially, agent populations are constructed by creating multiple copies of each learner $i \in \{1, \dots, N\}$ assuming that all agents (in one population) execute the same unified policy. With this, α -Rank then simulates a multi-agent game played by randomly sampled learners from each population.

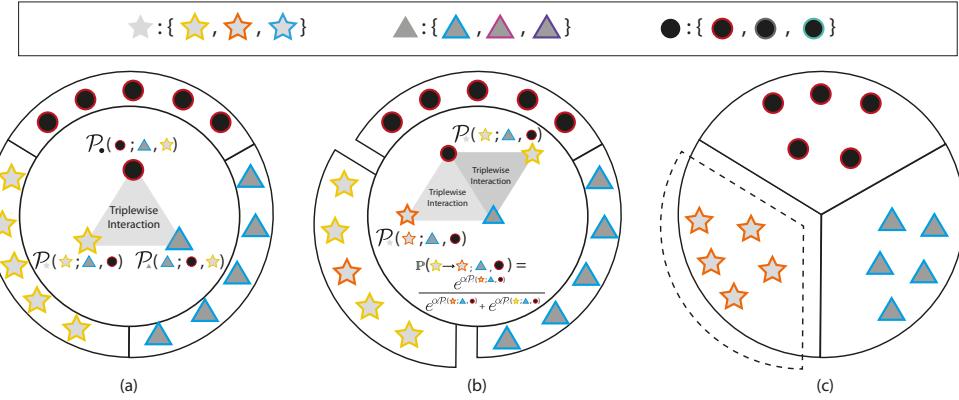


Figure 4.1: Example of α -Rank evaluation on $N = 3$ players (star, triangle, circle) each with $|s| = 3$ strategies (denoted by the colours) and $m = 5$ copies. a) Each population obtains a fitness value \mathcal{P}_i depending on the strategies chosen, b) one mutation strategy (red star) occurs, and c) the population either selects the original strategy, or being fixated by the mutation strategy.

Upon game termination, each participating agent receives a payoff to be used in policy mutation and selection after its return to the population. Here, the agent is faced with a probabilistic choice between switching to the mutation policy, continuing to follow its current policy, or randomly selecting a novel policy (other than the previous two) from the pool. This process repeats with the goal of determining an evolutionary dominant profile that spreads across the population of agents. Figure 4.1 demonstrates a simple example of a three-player game, and each player plays three strategies.

Mathematical Formulation: To formalise α -Rank, we consider N agents each, denoted by i , having access to a set of strategies of size k_i . We refer to the strategy set for agent i by $\mathcal{S}_i = \{\pi_{i,1}, \dots, \pi_{i,k_i}\}$, $k_i = |\mathcal{S}_i|$, with $\pi_{i,j} : \mathcal{X} \times \mathcal{A}_i \rightarrow [0, 1]$ representing the j^{th} allowed policy of the learner. \mathcal{X} represents the set of states, and \mathcal{A}_i is the set of actions for agent i . A joint strategy profile is a set of policies for all participating agents in the joint strategy set, i.e., $\mathcal{S}_{\text{joint}} = \mathcal{S}_1 \times \mathcal{S}_2 \times \dots \times \mathcal{S}_N$: $\pi_{\text{joint}} = \{\pi_{1,j_1}, \dots, \pi_{N,j_N}\}$, with $\pi_{i,j_i} \in \mathcal{S}_i$ and $j_i \in \{1, \dots, k_i\}$. We assume $k = k_1 = \dots = k_N$ hereafter.

To evaluate performance, we assume each agent is additionally equipped with a payoff (reward) function $\mathcal{P}_i : \mathcal{S}_{\text{joint}} \rightarrow \mathbb{R}_+$. Crucially, the domain of \mathcal{P}_i is the pool of *joint strategies* so as to accommodate the effect of other learners on the

i^{th} player's performance. Finally, given a joint profile π_{joint} , we define the corresponding joint payoff to be the collection of all individual payoff functions, i.e., $\mathcal{P}_{\text{joint}} = \{\mathcal{P}_1(\pi_{\text{joint}}), \dots, \mathcal{P}_N(\pi_{\text{joint}})\}$. After attaining payoffs from the environment, each agent returns to its population and faces a choice between switching the whole population to a mutation policy, exploring a novel policy, or sticking to the current one. Such a choice is probabilistic and defined proportional to rewards by

$$\begin{aligned}\mathbb{P}(\pi_{i,a} \rightarrow \pi_{i,b}, \boldsymbol{\pi}_{-i}) &= \frac{e^{\alpha \mathcal{P}_i(\pi_{i,b}, \boldsymbol{\pi}_{-i})}}{e^{\alpha \mathcal{P}_i(\pi_{i,a}, \boldsymbol{\pi}_{-i})} + e^{\alpha \mathcal{P}_i(\pi_{i,b}, \boldsymbol{\pi}_{-i})}} - \frac{\mu}{2} \\ &\quad \text{for } (\pi_{i,a}, \pi_{i,b}) \in \mathcal{S}_i \times \mathcal{S}_i, \\ \mathbb{P}(\pi_{i,a} \rightarrow \pi_{i,c}, \boldsymbol{\pi}_{-i}) &= \frac{\mu}{k_i - 2}, \quad \forall \pi_{i,c} \in \mathcal{S}_i \setminus \{\pi_{i,a}, \pi_{i,b}\},\end{aligned}$$

with $\mu \in \mathbb{R}_+$ being an exploration parameter¹, $\boldsymbol{\pi}_{-i} = \boldsymbol{\pi} \setminus \pi_i$ representing policies followed by other agents, and $\alpha \in \mathbb{R}_+$ a ranking-intensity parameter. A large α ensures that the probability of a sub-optimal strategy overtaking a better strategy is close to zero.

As noted in [Omidshafiei et al. \(2019a\)](#), one can relate the above switching process to a random walk on a Markov chain with states defined as elements in $\mathcal{S}_{\text{joint}}$. Essentially, the Markov chain models the *sink strongly connected components* (SSCC) of the *response graph* associated with the game. The response graph of a game is a directed graph where each node corresponds to each joint strategy profile, and directed edges if the deviating player's new strategy is a better response to that player, and the SSCC of a directed graph are the (group of) nodes with no outgoing edges.

Each entry in the transition probability matrix $\mathbf{T} \in \mathbb{R}^{|\mathcal{S}_{\text{joint}}| \times |\mathcal{S}_{\text{joint}}|}$ of the Markov chain refers to the probability of one agent switching from one policy in a relation to attained payoffs. Consider any two joint strategy profiles π_{joint} and $\hat{\pi}_{\text{joint}}$ that differ in only *one* individual strategy for the i^{th} agent, i.e., there exists a unique agent such that $\pi_{\text{joint}} = \{\pi_{i,a}, \boldsymbol{\pi}_{-i}\}$ and $\hat{\pi}_{\text{joint}} = \{\hat{\pi}_{i,b}, \boldsymbol{\pi}_{-i}\}$ with $\pi_{i,a} \neq \hat{\pi}_{i,b}$,

¹In α -Rank, μ is heuristically set to a small positive constant to ensure at maximum two varying policies per-each population. Theoretical justification can be found in ([Fudenberg and Imhof, 2006](#)).

we set $[T]_{\boldsymbol{\pi}_{\text{joint}}, \hat{\boldsymbol{\pi}}_{\text{joint}}} = \frac{1}{\sum_{l=1}^N (k_l - 1)} \rho_{\pi_{i,a}, \hat{\pi}_{i,b}}(\boldsymbol{\pi}_{-i})$, with $\rho_{\pi_{i,a}, \hat{\pi}_{i,b}}(\boldsymbol{\pi}_{-i})$ defining the probability that one copy of agent i with strategy $\pi_{i,a}$ invades the population with all other agents (in that population) playing $\hat{\pi}_{i,b}$. Following [Pinsky and Karlin \(2010\)](#), for $\mathcal{P}_i(\pi_{i,a}, \boldsymbol{\pi}_{-i}) \neq \mathcal{P}_i(\hat{\pi}_{i,b}, \boldsymbol{\pi}_{-i})$, such a probability is formalised as

$$\rho_{\pi_{i,a}, \hat{\pi}_{i,b}}(\boldsymbol{\pi}_{-i}) = \frac{1 - e^{-\alpha(\mathcal{P}_i(\pi_{i,a}, \boldsymbol{\pi}_{-i}) - \mathcal{P}_i(\hat{\pi}_{i,b}, \boldsymbol{\pi}_{-i}))}}{1 - e^{-m\alpha(\mathcal{P}_i(\pi_{i,a}, \boldsymbol{\pi}_{-i}) - \mathcal{P}_i(\hat{\pi}_{i,b}, \boldsymbol{\pi}_{-i}))}}, \quad (4.1)$$

and $\frac{1}{m}$ otherwise, with m being the size of the population. So far, we presented relevant derivations for the $(\boldsymbol{\pi}_{\text{joint}}, \hat{\boldsymbol{\pi}}_{\text{joint}})$ entry of the state transition matrix when exactly the i^{th} agent differs in exactly one strategy. Having one policy change, however, only represents a subset of allowed variations; two more cases need to be considered. Now we restrict our attention to variations in joint policies involving more than two individual strategies, i.e., $|\boldsymbol{\pi}_{\text{joint}} \setminus \hat{\boldsymbol{\pi}}_{\text{joint}}| \geq 2$. Here, we set² $[T]_{\boldsymbol{\pi}_{\text{joint}}, \hat{\boldsymbol{\pi}}_{\text{joint}}} = 0$. Consequently, the remaining event of self-transitions can be thus written as $[T]_{\boldsymbol{\pi}_{\text{joint}}, \hat{\boldsymbol{\pi}}_{\text{joint}}} = 1 - \sum_{\hat{\boldsymbol{\pi}} \neq \boldsymbol{\pi}_{\text{joint}}} [T]_{\boldsymbol{\pi}_{\text{joint}}, \hat{\boldsymbol{\pi}}}$. Summarising the above three cases, we can write the $(\boldsymbol{\pi}_{\text{joint}}, \hat{\boldsymbol{\pi}}_{\text{joint}})$'s entry of the Markov chain's transition matrix as:

$$[T]_{\boldsymbol{\pi}_{\text{joint}}, \hat{\boldsymbol{\pi}}_{\text{joint}}} = \begin{cases} \frac{1}{\sum_{l=1}^N (k_l - 1)} \rho_{\pi_{i,a}, \hat{\pi}_{i,b}}(\boldsymbol{\pi}_{-i}), & \text{if } |\boldsymbol{\pi}_{\text{joint}} \setminus \hat{\boldsymbol{\pi}}_{\text{joint}}| = 1, \\ 1 - \sum_{\hat{\boldsymbol{\pi}} \neq \boldsymbol{\pi}_{\text{joint}}} [T]_{\boldsymbol{\pi}_{\text{joint}}, \hat{\boldsymbol{\pi}}}, & \text{if } \boldsymbol{\pi}_{\text{joint}} = \hat{\boldsymbol{\pi}}_{\text{joint}}, \\ 0, & \text{if } |\boldsymbol{\pi}_{\text{joint}} \setminus \hat{\boldsymbol{\pi}}_{\text{joint}}| \geq 2, \end{cases} \quad (4.2)$$

The goal in α -Rank is to establish an ordering in policy profiles dependent on evolutionary stability of each joint strategy. In other words, higher ranked strategies are those that are prevalent in populations with higher average time of survival. Formally, such a notion can be easily derived as the limiting vector $\mathbf{v} = \lim_{t \rightarrow \infty} [[T]^T]^t \mathbf{v}_0$ of our Markov chain when evolving from an initial distribution \mathbf{v}_0 . Knowing that the limiting vector is a stationary distribution, one can

²This assumption significantly reduces the analysis complexity as detailed in [Fudenberg and Imhof \(2006\)](#).

Algorithm 3 α -Rank (see Section 3.1.1 in [Omidshafiei et al. \(2019a\)](#))

-
- 1: **(Unspecified) Inputs:** $\mathcal{S}_{\text{joint}}$, Multi-agent Simulator
 - 2: Listing all possible joint-strategy profiles, for each profile, run the multi-agent simulator to get the payoff values for all players $\mathcal{P}_{\text{joint}} = \{\mathcal{P}_1(\pi_{\text{joint}}), \dots, \mathcal{P}_N(\pi_{\text{joint}})\}, \forall \pi_{\text{joint}} \in \mathcal{S}_{\text{joint}}$.
 - 3: Construct Markov chain's transition matrix \mathbf{T} by Eq. (4.2).
 - 4: Compute the stationary distribution \mathbf{v} by Eq. (4.3).
 - 5: Rank all π_{joint} in \mathbf{v} based on their probability masses.
 - 6: **Outputs:** The ranked list of \mathbf{v} (each element refers to the time that players spend in playing that π_{joint} during evolution).
-

calculate the solution to the following eigenvector problem:

$$[\mathbf{T}]^T \mathbf{v} = \mathbf{v}. \quad (4.3)$$

We summarised the pseudo-code of α -Rank in Algorithm 3. As the input to α -Rank is unclear and turns out to be controversial later, we point the readers to the original description in Section 3.1.1 of [Omidshafiei et al. \(2019a\)](#), and the practical implementation of α -Rank from [Lanctot et al. \(2019\)](#) for self-judgement. In what comes next, we demonstrate that the tractability claim of α -Rank needs to be relaxed as the algorithm exhibits exponential time and memory complexities in number of players dependent on the input type considered. This, consequently, renders α -Rank inapplicable to large-scale multi-agent systems contrary to the original presentation.

4.3 Reconsidering α -Rank's Complexity

The original presentation of α -Rank claims to be tractable in the sense that it runs in polynomial time with respect to the total number of joint-strategy profiles. Unfortunately, such a claim is not clear without a formal specification of the inputs to Algorithm 3.1.1 in [Omidshafiei et al. \(2019a\)](#). In fact, we, next, demonstrate that α -Rank's can easily exhibit exponential complexity under the input of an $N \times k$ table, rendering it inapplicable beyond a finite small number of players. We also present a conjecture stating that determining the top-rank joint strategy profile in α -Rank may in fact be NP -hard.

4.3.1 Conjecture: Computing α -Rank is NP-Hard

Before diving into the details of our arguments, it is first instructive to note that *tractable* algorithms are those that exhibit a worst-case polynomial running time in the size of their input (Papadimitriou, 2003). Mathematically, for a size \mathcal{I} input, a polynomial time algorithm adheres to an $\mathcal{O}(\mathcal{I}^d)$ complexity for some constant d independent of \mathcal{I} .

Following the presentation in Section 3.1.1 in Omidshafiei et al. (2019a), α -Rank assumes the availability of a game simulator to construct a payoff matrix quantifying the performance of joint strategy profiles. As such, we deem that necessary inputs for such construction are of the size $\mathcal{I} = N \times k$, where N is the total number of agents and k is the total number of strategies per agent, where we assumed $k = k_i = k_j$ for simplicity.

Following the definition above, if α -Rank possesses polynomial complexity then it should attain a time proportional to $\mathcal{O}((N \times k)^d)$ with d being a constant independent of N and k . As the algorithm requires to compute a stationary distribution of a Markov chain described by a transition matrix \mathbf{T} with k^N rows and columns, the time complexity of α -Rank amounts to $\mathcal{O}(k^N)$. Clearly, this result demonstrates exponential, thus intractable, complexity in the number of agents N . In fact, we conjecture that determining top rank joint strategy profile using α -Rank with an $N \times k$ input is **NP-hard**.

Proposition 1 (Conjecture that α -Rank is **NP-hard**). Consider N agents each with k strategies. Computing top-rank joint strategy profile with respect to the stationary distribution of the Markov chain's transition matrix, \mathbf{T} , is **NP-hard**.

Proof. To illustrate the point of the conjecture above, imagine N agents each with k strategies. Following the certificate argument for determining complexity classes, we ask the question:

“Assume we are given a joint strategy profile π_{joint} , is π_{joint} top rank w.r.t the stationary distribution of the Markov chain?”

To determine an answer to the above question, one requires an evaluation mecha-

Table 4.1: Time and space complexity comparison given N (number of agents) \times k (number of strategies) as inputs.

Method	Time	Memory
Power Method	$\mathcal{O}(k^{N+1}N)$	$\mathcal{O}(k^{N+1}N)$
PageRank	$\mathcal{O}(k^{N+1}N)$	$\mathcal{O}(k^{N+1}N)$
Eig. Decomp.	$\mathcal{O}(k^{N\omega})$	$\mathcal{O}(k^{N+1}N)$
Mirror Descent	$\mathcal{O}(k^{N+1} \log k)$	$\mathcal{O}(k^{N+1}N)$

nism of some sort. If the time complexity of this mechanism is polynomial with respect to the input size, i.e., $N \times k$, then one can claim that the problem belongs to the NP complexity class. However, if the mechanism mentioned above exhibits an exponential time complexity, then the problem belongs to the *NP-hard* class. When it comes to α -Rank, we believe a mechanism answering the above question would require computing a holistic solution of the problem, which, unfortunately, is exponential (i.e., $\mathcal{O}(k^N)$). Crucially, if our conjecture proves correct, we do not see how α -Rank can handle more than a finite small number of agents. \square

4.3.2 Approximation Solutions to α -Rank

Given exponential complexity as derived above, we can resort to approximations of stationary distributions that aim at determining ε -close solution for some precision parameter $\varepsilon > 0$. Here, we note that a problem of this type is a long-standing classical problem from linear algebra. Various techniques including Power method, PageRank, eigenvalue decomposition, and mirror descent can be utilised. Briefly surveying this literature, we demonstrate that any such implementation (unfortunately) scales exponentially in the number of players. For a quick summary, please consult Table 4.1.

- **Power Method**

One of the most common approaches to computing a stationary distribution is the Power Method that computes the stationary vector \mathbf{v} by constructing a sequence $\{\mathbf{v}_j\}_{j \geq 0}$ from a non-zero initialisation \mathbf{v}_0 by applying $\mathbf{v}_{j+1} =$

$\frac{1}{\|\mathbf{T}^\top \mathbf{v}_j\|} \mathbf{T}^\top \mathbf{v}_j$. Though viable, we first note that the Power Method exhibits an exponential memory complexity in terms of the number of agents. To formally derive the bound, define n to represent the total number of joint strategy profiles, i.e., $n = k^N$, and m the total number of transitions between the states of the Markov chain. By construction, one can easily see that $m = n(kN - N + 1)$ as each row and column in \mathbf{T} contains $kN - N + 1$ non-zero elements. Hence, memory complexity of such implementation is in the order of

$$\mathcal{O}(m) = \mathcal{O}(n[kN - N + 1]) \approx \mathcal{O}(k^N kN).$$

The time complexity of the Power method, furthermore, is given by $\mathcal{O}(m \times \mathcal{T})$, where \mathcal{T} is the total number of iterations. Since m is of the order $n \log n$, such an implementation's total complexity is also exponential.

Additionally, one can look into the complexity of the Power method through the following lemma.

Lemma 1 (Second-Smallest Eigenvalue). *Consider the Markov chain defined in Section 4.2 with states in $\mathcal{S}_{\text{joint}}$ and transition probability matrix \mathbf{T} . The second-smallest eigenvalue of the normalised Laplacian of the graph associated with the Markov chain is given by: $\mu_2(\mathcal{L}_{\mathbb{G}}) = \frac{\min_i k_i - 1}{\sum_{i=1}^N k_i - N + 1}$, with k_i denoting the number of strategies of agent i .*

Proof. Note that this lemma is proved by co-author Rasul Tutunov in [Yang et al. \(2019a\)](#). For simplicity we drop round index k in the below derivation. Notice, the underlying graph for the constructed Markov Chain can be represented as a Cartesian product of N complete graphs \mathcal{K}_{k_i} (Here \mathcal{K}_p denotes a complete graph with p nodes.):

$$\mathbb{G} = \mathcal{K}_{k_1} \times \mathcal{K}_{k_2} \times \cdots \times \mathcal{K}_{k_N} \quad (4.4)$$

Indeed, two vertices $\boldsymbol{\pi}, \hat{\boldsymbol{\pi}} \in \mathbb{G}$ are connected by the edge if and if only these joint strategy profiles differ in at most one individual strategy, i.e $\exists! i \in$

$\{1, \dots, N\} : \boldsymbol{\pi} = \{\pi_{ia}, \boldsymbol{\pi}_{-i}\}$, $\hat{\boldsymbol{\pi}} = \{\hat{\pi}_{ib}, \boldsymbol{\pi}_{-i}\}$. Hence, the spectral properties of \mathbb{G} can be described in terms of spectral properties of \mathcal{K}_{k_i} (Barik et al., 2015):

$$\text{Spectr}(\mathbb{G}) = \{\mu_{i_1}(\mathbb{L}_{\mathcal{K}_{k_1}}^{(\text{un})}) + \dots + \mu_{i_N}(\mathbb{L}_{\mathcal{K}_{k_N}}^{(\text{un})})\}_{i_1=1, i_2=1, \dots, i_N=1}^{k_1, k_2, \dots, k_N}$$

$$\text{Eigenvectors of } \mathbb{G} = \{\boldsymbol{\vartheta}_{i_1, 1} \otimes \boldsymbol{\vartheta}_{i_2, 2} \otimes \dots \otimes \boldsymbol{\vartheta}_{i_N, N}\}_{i_1=1, i_2=1, \dots, i_N=1}^{k_1, k_2, \dots, k_N}.$$

where $\mu_i(\mathbb{L}_{\mathcal{K}_{k_j}}^{(\text{un})})$ is the i^{th} eigenvalue of the unnormalised Laplacian of the complete graph \mathcal{K}_{k_j} and $\boldsymbol{\vartheta}_{i,j}$ is the corresponding eigenvector. In other words, $\mathbb{L}_{\mathcal{K}_{k_j}} \boldsymbol{\vartheta}_{i,j} = \mu_i(\mathbb{L}_{\mathcal{K}_{k_j}}) \boldsymbol{\vartheta}_{i,j}$ for all $i = 1, \dots, k_j$ and $j = 1, \dots, N$. The spectrum of unnormalised Laplacian of the complete graph \mathcal{K}_{k_i} is given by $\text{Spectr}(\mathcal{K}_{k_i}) = \{0, k_i - 1\}$ and the only eigenvector corresponding to zero eigenvalue is $\mathbf{1} \in \mathbb{R}^{k_i}$. Therefore, the minimum non-zero eigenvalue of unnormalised Laplacian of \mathbb{G} is given by $\min_i k_i - 1$. Finally, due to the fact that \mathbb{G} is a regular graph (with degree of each node is equal to $\sum_{i=1}^N k_i - N + 1$), the smallest non-zero eigenvalue of the normalised Laplacian of \mathbb{G} is given by $\frac{\min_i k_i - 1}{\sum_{i=1}^N k_i - N + 1}$. \square

Given this result, the overall time complexity of the Power Method is bounded by $\mathcal{O}\left(n \times \log \left[\frac{\sum_{i=1}^N k_i - N + 1}{\min_i k_i - 1} \right] \right) = \mathcal{O}(n \log n)$. Indeed, notice that $\frac{\min_i k_i - 1}{\sum_{i=1}^N k_i - N + 1} \leq \frac{\min_i k_i - 1}{\log n} = \Omega\left(\frac{1}{\log n}\right)$, hence, $\frac{\sum_{i=1}^N k_i - N + 1}{\min_i k_i - 1} = \mathcal{O}(\log n)$. As for the memory complexity, the Power Method has the same requirements as the PageRank algorithm due to the necessity to store matrix \mathbf{T} . These results imply that the Power Method scales exponentially with the number of agents N , and therefore, is inapplicable when N is large.

• PageRank

Inspired by ranking web-pages on the internet, one can consider PageRank (Page et al., 1999) for computing the solution to the eigenvalue problem presented above. Applied to our setting, we first realise that the memory is analogous to the Power Method, that is $\mathcal{O}(m) = \mathcal{O}(K^{N+1}N)$, and the time complexity are in the order of $\mathcal{O}(m+n) \approx \mathcal{O}(K^{N+1}N)$.

- **Eigenvalue Decomposition**

Apart from the above, we can also consider the problem as a standard eigenvalue decomposition task (also what is used to implement α -Rank in [Lanctot et al. \(2019\)](#)) and adopt the method in [Coppersmith and Winograd \(1990\)](#) to compute the stationary distribution. Unfortunately, state-of-the-art techniques for eigenvalue decomposition also require exponential memory ($\mathcal{O}(K^{N+1}N)$) and exhibit a time complexity of the form $\mathcal{O}(n^\omega) = \mathcal{O}(k^{N\omega})$ with $\omega \in [2, 2.376]$ ([Coppersmith and Winograd, 1990](#)). Clearly, these bounds restrict α -Rank to small agent number N .

- **Mirror Descent**

Another optimisation-based alternative is the ordered subsets mirror descent algorithm ([Ben-Tal et al., 2001](#)). This is an iterative procedure requiring projection step on the standard n -dimensional simplex on every iteration: $\Delta_n = \{\mathbf{x} \in \mathbb{R}^n : \mathbf{x}^\top \mathbf{1} = 1 \text{ & } \mathbf{x} \succeq \mathbf{0}\}$. As mentioned in ([Ben-Tal et al., 2001](#)), computing this projection requires $\mathcal{O}(n \log n)$ time. Hence, the projection step is exponential in the number of agents N . This makes mirror descent inapplicable to α -Rank when N is large.

- **Other Methods**

Apart from the methods listed above, we are aware of other approaches that could solve the leading eigenvector for big matrices, for example, the online learning approach ([Garber et al., 2015](#)), the sketching methods ([Tropp et al., 2017](#)), and the subspace iteration with Rayleigh-Ritz acceleration ([Golub and Van der Vorst, 2000](#)). The trade-off of these methods is that they usually assume special structure of the matrix, such as being Hermitian or at least positive semi-definite, which α -Rank however does not fit. Importantly, they can not offer any advantages on the time complexity either.

4.3.3 On the Definition of Agents

Having discussed our results with the authors, we were suggested that “inputs” to α -Rank are **exponentially-sized** payoff matrices, i.e., assuming line 2 in Algorithm 3 as an input. Though polynomial in an exponentially-sized input, this consideration does not resolve the problems mentioned above. In this section, we further demonstrate additional theoretical and practical problems when considering the advised “input” by the authors.

α -Rank redefines a strategy to correspond to the agents under evaluation differentiating them from players in the game (see line 4 in Section 3.1.1 and also Figure 2a in [Omidshafiei et al. \(2019a\)](#)). Complexity results are then given in terms of these “agents”, where tractability is claimed. We want to clarify that such definitions do not necessarily reflect the real underlying time complexity, whereby without formal input definitions, it is challenging to claim tractability.

To illustrate, consider solving a travelling salesman problem in which a traveller needs to visit a set of cities while returning to the origin following the shortest route. Although it is well-known that the travelling salesman problem is *NP-hard*, following the line of thought presented in α -Rank, one can show that such a problem reduces to a polynomial-time (linear, i.e., tractable) problem in the size of “meta-cities”, which is not a valid claim.

So what are the “meta-cities”, and what is wrong with the above argument?

A strategy in the travelling salesman problem corresponds to a permutation in the order of cities. Rather than operating with the number of cities, following α -Rank, we can construct the space of all permutations and calling each permutation a “meta-city” (or agent)³. Having enumerated all permutations, somehow, searching for the shortest route can be performed in polynomial time. Even though one can state that solving the travelling salesman problem is polynomial in the size of permutations, it is incorrect to claim that any such algorithm is tractable. The same exact argument can be made for α -Rank, whereby having a polynomial-time algorithm in an

³How to enumerate all these permutations of cities is analogous to enumerating an exponentially sized matrix in α -Rank if $N \times k$ was not the input to α -Rank.

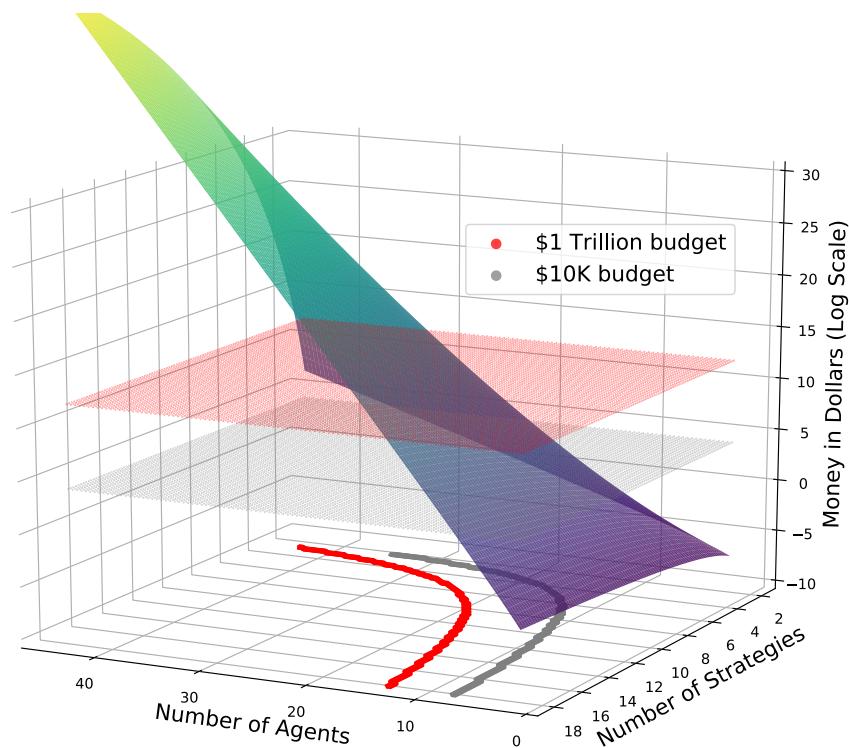


Figure 4.2: Money cost of constructing the transition matrix \mathbf{T} in computing α -Rank (line 3 in Algorithm 3). Note that one trillion dollar is the world’s total hardware budget (Department, 2020). The projected contours show that due to the exponentially-growing size of α -Rank’s “input”, under reasonable budget, α -Rank is unable to handle more than ten agents.

exponentially-sized space does not at all imply tractability⁴. For this reason, reporting complexity results needs to be done concerning the size of the input without any redefinition (i.e. agents in multi-agent systems, and cities in the travelling salesman problem).

As is clear so-far, inputs to α -Rank lack clarity. Confused on the form of the input, we realise that we are left with two choices: 1) list of all joint strategy profiles, or 2) a table of the size $N \times k$ – a collection of all of the players’ strategy pools. If we are to follow the first direction, the paper’s claims are of course correct; however, this by no means resolves the problem as it is not clear how one would construct such input in a tractable manner. Precisely, given an $N \times k$ table (collection of all of the players’ strategy pools) as input, constructing the aforemen-

⁴Note that this claim does not apply to the complexity of solving Nash equilibrium. For example, in solving zero-sum games, polynomial tractability is never claimed on the number of players, whereas α -Rank claims to be tractable in the number of players.

Table 4.2: Cost of getting the payoff table $\mathcal{P}_{\text{joint}}$ (line 2 in Algorithm 3) for the experiments conducted in Omidshafiei et al. (2019a). We list the numbers by the cost of running one joint-strategy profile \times the number of joint-strategy profiles considered. A detailed breakup can be found in Figure 4.7.

Game Env.	PetaFlop/s-days	Cost (\$)	Time (days)
AlphaZero Go (Silver et al., 2017)	$1,413 \times 7$	$207M$	$1.9M$
AlphaGo Zero (Silver et al., 2016)	$1,181 \times 7$	$172M$	$1.6M$
AlphaZero Chess (Silver et al., 2017)	17×1	$352K$	$3.2K$
MuJoCo Soccer (Liu et al., 2019b)	0.053×10	$4.1K$	72
Leduc Poker (Lanctot et al., 2017)	0.006×9	420	7
Kuhn Poker (Heinrich et al., 2015)	$< 10^{-4} \times 256$	< 1	—
AlphaStar (Vinyals et al., 2019a)	52,425	$244M$	$1.3M$

tioned list requires exponential time (k^N). In other words, providing α -Rank with such a list only hides the exponential complexity burden in a pre-processing step. Analogously, applying this idea to the travelling salesman problem described above would hide the exponential complexity under a pre-processing step used to construct all possible permutations. Provided as inputs, the travelling salesman problem can now be solved in linear time, i.e., transforming an intractable problem to a tractable one by a mere redefinition.

4.3.4 Dollars Spent: A Non-Refutable Metric

Admittedly, our arguments have been mostly theoretical and can become controversial dependent on the setting one considers. To abolish any doubts, we followed the authors' advice and considered the input of α -Rank to be exponentially-sized payoff matrices. We then conducted an experiment measuring dollars spent to evaluate the scalability of running just line 3 in Algorithm 3, while considering the tasks reported in Omidshafiei et al. (2019a).

Assuming $\mathcal{P}_{\text{joint}} = \{\mathcal{P}_1(\pi_{\text{joint}}), \dots, \mathcal{P}_N(\pi_{\text{joint}})\}$, $\forall \pi_{\text{joint}} \in \mathcal{S}_{\text{joint}}$ is given at no cost, the total amount of floating point operations (FLOPS) needed for constructing \mathbf{T} given in Eq. (4.2) is $9k^N N(k-1) + k^N N(k-1) + 0 = 10k^N N(k-1)$. In terms of money cost needed for just building \mathbf{T} , we plot the dollar amount in Figure 4.2 considering the Nvidia Tesla K80 GPU⁵ which can process under single precision

⁵https://en.wikipedia.org/wiki/Nvidia_Tesla

at maximum 5.6 TFlop/s at a price of 0.9 \$/hour⁶. Clearly, Figure 4.2 shows that due to the fact that α -Rank needs to construct a Markov chain with an exponential size in the number of agents, it is only “money” feasible on tasks with at most tens of agents. It is also worth noting that our analysis is optimistic in the sense that we have not considered costs of storing \mathbf{T} nor computing Eq. (4.3).

Conclusion I:

*Given exponentially-sized payoff matrices, constructing transition matrices in α -Rank for about 20 agents each with 8 strategies requires about **one trillion dollars** in budget.*

Though assumed given, in reality, the payoff values $\mathcal{P}_{\text{joint}}$ come at a non-trivial cost themselves, which is particularly true in RL tasks (Silver et al., 2016). Here, we take a closer look at the amount of money it takes to attain payoff matrices for the experiments listed in Omidshafiei et al. (2019a) that we present in Table 4.2. Following the methodology in the blog of <https://openai.com/blog/ai-and-compute/>, we first count the total FLOPS each model uses under the unit of PetaFlop/s-day that consists of performing 10^{20} operations per second in one day. For each experiment, if the answer to “how many GPUs were trained and for how long” was not available, we then traced back to the neural architecture used and counted the operations needed for both forward and backward propagation. The cost in time was then transformed from PetaFlop/s-day using Tesla K80 as discussed above. In addition, we also list the cost of attaining payoff values from the most recent AlphaStar model (Vinyals et al., 2019a). It is evident that although α -Rank could take the payoff values as “input” at a hefty price, the cost of acquiring such values is not negligible, e.g., payoff values from GO cost about 207M \$, and require a single GPU to run for more than five thousand years⁷!

⁶<https://aws.amazon.com/ec2/instance-types/p2/>

⁷In practice, the game outcomes are noisy, multiple runs are often needed (check Theorem 3.2 in Rowland et al. (2019)), which will turn the numbers in Table 4.2 to an even larger scale.

Conclusion II:

Acquiring necessary inputs to α -Rank easily becomes intractable giving credence to our arguments in Section 4.3.3.

4.4 α^α -Rank: A Scalable Solution to α -Rank

One can consider approximate solutions to the problem in Eq. (4.3). As briefly surveyed in Section 4.3, most current methods, unfortunately, require exponential *time and memory* complexities. We believe achieving a solution that aims to reduce time complexity is an interesting and open question in linear algebra in general, and leave such a study to future work. Here, we rather contribute by a stochastic optimisation method that can attain a solution through random sampling of payoff matrices without the need to store exponential-size input. Contrary to memory requirements reported in Table 4.1, our method requires a linear (in number of agents) per-iteration complexity of the form $\mathcal{O}(Nk)$. It is worth noting that most other techniques need to store exponentially-sized matrices before commencing with any numerical instructions. Though we do not theoretically contribute to reductions in time complexities, we do, however, augment our algorithm with a double-oracle heuristic for joint strategy space reduction. In fact, our experiments reveal that α^α -Rank can converge to the correct top-rank strategies in hundreds of iterations in large strategy spaces, i.e., spaces with ≈ 33 million profiles.

4.4.1 Solution by Stochastic Optimisation

Computing the stationary distribution can be rewritten as an optimisation problem:

$$\min_{\mathbf{x}} \frac{1}{n} \|\mathbf{T}^T \mathbf{x} - \mathbf{x}\|_2^2 \quad \text{s.t. } \mathbf{x}^T \mathbf{1} = 1, \text{ and } \mathbf{x} \succeq \mathbf{0}, \quad (4.5)$$

where the constrained objective in Eq. (4.5) simply seeks a vector \mathbf{x} minimising the distance between \mathbf{x} , itself, and $\mathbf{T}^T \mathbf{x}$ while ensuring that \mathbf{x} lies on an n -dimensional probability simplex. To handle exponential complexities needed for acquiring exact solutions, we pose a relaxation of the problem in Eq. (4.5) and focus on computing

an approximate solution vector $\tilde{\mathbf{x}}$ instead, where $\tilde{\mathbf{x}}$ solves:

$$\min_{\mathbf{x}} \frac{1}{n} \|\mathbf{T}^T \mathbf{x} - \mathbf{x}\|_2^2 \text{ s.t. } |\mathbf{x}^T \mathbf{1} - 1| \leq \delta \text{ for } 0 < \delta < 1, \text{ and } \mathbf{x} \succeq \mathbf{0}. \quad (4.6)$$

Before proceeding, however, it is worth investigating the relation between the solutions of the original Eq. (4.5) and relaxed Eq. (4.6) problems. We summarise such a relation in the following proposition that shows that determining $\tilde{\mathbf{x}}$ suffices for computing a stationary distribution of α -Rank's Markov chain:

Proposition 2 (Connections to Markov Chain). Let $\tilde{\mathbf{x}}$ be a solution to the relaxed optimisation problem in Eq. (4.6). Then, $\frac{\tilde{\mathbf{x}}}{\|\tilde{\mathbf{x}}\|_1} = \mathbf{v}$ is the stationary distribution of Eq. (4.3) in Section 4.2.

Importantly, the above proposition, additionally, allows us to focus on solving the problem in Eq. (4.6) that only exhibits inequality constraints. Problems of this nature can be solved by considering a barrier function leading to an unconstrained finite sum minimisation problem. By denoting \mathbf{b}_i to be the i^{th} row of $\mathbf{T}^T - \mathbf{I}$, we can, thus, write: $\frac{1}{n} \|\mathbf{T}^T \mathbf{x} - \mathbf{x}\|_2^2 = \frac{1}{n} \sum_{i=1}^n (\mathbf{x}^T \mathbf{b}_i)^2$. Introducing logarithmic barrier-functions, with $\lambda > 0$ being a penalty parameter, we have:

$$\min_{\mathbf{x} \in \mathbb{R}^n} \frac{1}{n} \sum_{i=1}^n (\mathbf{x}^T \mathbf{b}_i)^2 - \lambda \log(\delta^2 - [\mathbf{x}^T \mathbf{1} - 1]^2) - \frac{\lambda}{n} \sum_{i=1}^n \log(x_i). \quad (4.7)$$

Eq. (4.7) represents a standard finite minimisation problem, which can be solved using any off-the-shelf stochastic optimisation methods, e.g., stochastic gradients, ADAM ([Kingma and Ba, 2014](#)). A stochastic gradient execution involves sampling a strategy profile $i_t \sim [1, \dots, n]$ at iteration t , and then executing a descent step: $\mathbf{x}_{t+1} = \mathbf{x}_t - \eta_t \nabla_{\mathbf{x}} f_{i_t}(\mathbf{x}_t)$, with $\nabla_{\mathbf{x}} f_{i_t}(\mathbf{x}_t)$ being a sub-sampled gradient of Eq. (4.7), and λ being a scheduled penalty parameter with $\lambda_{t+1} = \frac{\lambda_t}{\gamma}$ for some $\gamma > 1$:

$$\nabla_{\mathbf{x}} f_{i_t}(\mathbf{x}_t) = 2 \left(\mathbf{b}_{i_t}^T \mathbf{1} \right) \mathbf{b}_{i_t} + \frac{2\lambda_t (\mathbf{x}_t^T \mathbf{1} - 1)}{\delta^2 - (\mathbf{x}_t^T \mathbf{1} - 1)^2} - \frac{\lambda_t}{n} \left[\frac{1}{[\mathbf{x}_t]_1}, \dots, \frac{1}{[\mathbf{x}_t]_n} \right]^T. \quad (4.8)$$

Algorithm 4 α^α -Rank (Log-Barrier Stochastic Gradient Descent)

1: **Input:** $\mathbf{x}_0 = \frac{1}{n}\mathbf{1}$, $\mathcal{T}, \lambda, \delta, \varepsilon, \{\eta_t\}_{t=1}^{\mathcal{T}}, \gamma > 1$.

2: **Output:** $\hat{\mathbf{v}}$

3: Set $\delta_0 = \delta, \lambda_0 = \lambda$.

4: **for** $t = 0$ to $\mathcal{T} - 1$ **do**:

5: Sample $i_t \sim [1, \dots, n]$ and compute:

$$\nabla_{\mathbf{x}} f_{i_t}(\mathbf{x}_t) = 2(\mathbf{b}_{i_t}^T \mathbf{1}) \mathbf{b}_{i_t} + \frac{2\lambda_t (\mathbf{x}_t^T \mathbf{1} - 1)}{\delta^2 - (\mathbf{x}_t^T \mathbf{1} - 1)^2} \mathbf{1} - \frac{\lambda_t}{n} \begin{bmatrix} \frac{1}{[\mathbf{x}_t]_1} \\ \vdots \\ \frac{1}{[\mathbf{x}_t]_n} \end{bmatrix}$$

6: Update $\mathbf{x}_{t+1} = \mathbf{x}_t - \eta_t \nabla_{\mathbf{x}} f_{i_t}(\mathbf{x}_t)$.

7: Update $\lambda_{t+1} = \frac{\lambda_t}{\gamma}$.

8: **end for**

9: Set $\hat{\mathbf{v}} = \mathbf{x}_{\mathcal{T}}$

To avoid any confusion, we name the above stochastic approach of solving α -Rank via Eq. (4.7) & (4.8) as **α^α -Rank** and present its pseudo-code in Algorithm 4. When comparing our algorithm to these reported in Table 4.1, it is worth highlighting that computing updates using Eq. (4.8) requires no storage of the full transition or payoff matrices as updates are performed by only using sub-sampled columns as shown in line 10 of Algorithm 5.

4.4.2 Time and Memory Complexity of α^α -Rank

We now investigate the convergence rate and time and space complexity of the proposed α^α -Rank algorithm, whose pseudocode can be further simplified into Algorithm 4.

Theorem 1 (Convergence Rate of α^α -Rank). *Let $\tilde{\mathbf{x}}_\lambda$ be the output of applying gradient descent on the objective in Eq. (4.7), after \mathcal{T} iterations, then*

$$\mathbb{E} \left[\left\| \left[\mathbf{T}^{[k], \top} \right] \tilde{\mathbf{x}}_\lambda - \tilde{\mathbf{x}}_\lambda \right\|_2^2 \right] \leq \mathcal{O} \left(\lambda + \frac{n}{\mathcal{T}} + \frac{n^2}{\gamma \mathcal{T} \lambda} \right),$$

where expectation is taken w.r.t. all randomness of a stochastic gradient implementation, and $\gamma > 1$ is a decay-rate for λ , i.e., $\lambda^{[t+1]} = \lambda^{[t]} / \gamma$, and n is the total number of profiles $n = \prod_{i=1}^N k_i$. See the Algorithm 4.

Proof. Let $\tilde{\mathbf{x}}$ and $\tilde{\mathbf{x}}_\lambda^*$ be the solutions of Eq. (4.6) and Eq. (4.7) respectively. Conver-

gence guarantees for logarithmic barrier method (Nocedal and Wright, 2006) with penalty parameter λ and barrier parameter γ gives:

$$\left\| [\mathbf{T}]^T \tilde{\mathbf{x}}_\lambda^* - \tilde{\mathbf{x}}_\lambda^* \right\|_2^2 - \left\| [\mathbf{T}]^T \tilde{\mathbf{x}} - \tilde{\mathbf{x}} \right\|_2^2 = \mathcal{O} \left(\frac{n^2}{\gamma^\mathcal{T} \lambda} \right) \quad (4.9)$$

and using $[\mathbf{T}]^T \tilde{\mathbf{x}} = \tilde{\mathbf{x}}$ in Eq. (4.9) gives:

$$\left\| [\mathbf{T}]^T \tilde{\mathbf{x}}_\lambda^* - \tilde{\mathbf{x}}_\lambda^* \right\|_2^2 = \mathcal{O} \left(\frac{n^2}{\gamma^\mathcal{T} \lambda} \right) \quad (4.10)$$

Applying the convergence guarantees Rakhlin et al. (2011) of stochastic gradient descent method to strongly convex function $F(\mathbf{x})$ gives:

$$\mathbb{E} [F(\tilde{\mathbf{x}}_\lambda) - F(\tilde{\mathbf{x}}_\lambda^*)] \leq \mathcal{O} \left(\frac{1}{\mathcal{T}} \right)$$

Using the definition of function $F(\mathbf{x})$:

$$n \mathbb{E} [F(\tilde{\mathbf{x}}_\lambda) - F(\tilde{\mathbf{x}}_\lambda^*)] \geq \mathbb{E} \left[\left\| [\mathbf{T}]^T \tilde{\mathbf{x}}_\lambda - \tilde{\mathbf{x}}_\lambda \right\|_2^2 \right] - \left\| [\mathbf{T}]^T \tilde{\mathbf{x}}_\lambda^* - \tilde{\mathbf{x}}_\lambda^* \right\|_2^2 - \mathcal{O}(\lambda)$$

Combining the last two results:

$$\mathbb{E} \left[\left\| [\mathbf{T}]^T \tilde{\mathbf{x}}_\lambda - \tilde{\mathbf{x}}_\lambda \right\|_2^2 \right] - \left\| [\mathbf{T}]^T \tilde{\mathbf{x}}_\lambda^* - \tilde{\mathbf{x}}_\lambda^* \right\|_2^2 \leq \mathcal{O}(\lambda) + \mathcal{O} \left(\frac{n}{\mathcal{T}} \right)$$

Finally, using (4.10) gives:

$$\mathbb{E} \left[\left\| [\mathbf{T}]^T \tilde{\mathbf{x}}_\lambda - \tilde{\mathbf{x}}_\lambda \right\|_2^2 \right] \leq \mathcal{O} \left(\lambda + \frac{n}{\mathcal{T}} + \frac{n^2}{\gamma^\mathcal{T} \lambda} \right)$$

By choosing $\lambda = \frac{\varepsilon}{3}$ and $\mathcal{T} = \max \left\{ \frac{3n}{\varepsilon}, \frac{2 \log \frac{3n}{\varepsilon}}{\log \gamma} \right\}$, we have $\lambda + \frac{n}{\mathcal{T}} + \frac{n^2}{\gamma^\mathcal{T} \lambda} \leq \frac{\varepsilon}{3} + \frac{\varepsilon}{3} + \frac{\varepsilon}{3} = \varepsilon$. Hence,

$$\mathbb{E} \left[\left\| [\mathbf{T}]^T \tilde{\mathbf{x}}_\lambda - \tilde{\mathbf{x}}_\lambda \right\|_2^2 \right] \leq \mathcal{O}(\varepsilon)$$

□

Theorem 2 (Complexities of α^α -Rank). *Let ε be the precision parameter, α^α -Rank has $\mathcal{O}\left(\frac{n^2}{\varepsilon^2} \sum_{i=1}^N k_i\right)$ for time and $\mathcal{O}\left(\frac{n}{\varepsilon} \sum_{i=1}^N k_i\right)$ for space complexity.*

Proof. The above result of Theorem 1.2 implies given precision parameter $\varepsilon > 0$, after $\mathcal{T} = \mathcal{O}\left(\frac{n}{\varepsilon}\right)$ iterations, Algorithm 4 outputs vector $\tilde{\mathbf{x}}_\lambda \succ \mathbf{0}$ such that:

$$|\tilde{\mathbf{x}}_\lambda^T \mathbf{1} - 1| \leq \delta, \text{ and } \mathbb{E} \left[\left\| [\mathbf{T}]^T \tilde{\mathbf{x}}_\lambda - \tilde{\mathbf{x}}_\lambda \right\|_2 \right] \leq \mathcal{O}(\varepsilon)$$

Hence, by tuning δ and ε one can approximate a stationary distribution vector \mathbf{v} .

Algorithm 4 starts with uniform distribution vector $\mathbf{x}_0 = \frac{1}{n} \mathbf{1}$ and at step t it updates the previous iterative \mathbf{x}_t by a rule given in line 5. Let $\mathcal{N}_t = \{j \in [n] : \exists r \in [t] \text{ with } [\mathbf{b}_{i_r}]_j \neq 0\}$ (note that here $[n] = \{1, \dots, n\}$ and $[t] = \{0, \dots, t\}$). Then, for $j \notin \mathcal{N}_t$ all entries $[\mathbf{x}_{t+1}]_j$ are equal to each other and updated as:

$$[\mathbf{x}_{t+1}]_j = [\mathbf{x}_t]_j + \frac{2\eta_t \lambda_t (\mathbf{x}_t^T \mathbf{1} - 1)}{\delta^2 - (\mathbf{x}_t^T \mathbf{1} - 1)^2} - \frac{\eta_t \lambda_t}{n [\mathbf{x}_t]_j}$$

Given value $\mathbf{x}_t^T \mathbf{1}$, the above computation takes $\mathcal{O}(1)$ time and space. For entries $j \in \mathcal{N}_t$, all entries $[\mathbf{x}_{t+1}]_j$ might be different (worst case) and, therefore, update

$$[\mathbf{x}_{t+1}]_j = [\mathbf{x}_t]_j - 2\mathbf{1}^T \mathbf{b}_{i_t} [\mathbf{b}_{i_t}]_j + \frac{2\eta_t \lambda_t (\mathbf{x}_t^T \mathbf{1} - 1)}{\delta^2 - (\mathbf{x}_t^T \mathbf{1} - 1)^2} - \frac{\eta_t \lambda_t}{n [\mathbf{x}_t]_j}$$

need to be performed at most $\mathcal{O}(|\mathcal{N}_t|)$ times. Notice, only $\mathcal{O}(\sum_{i=1}^N k_i - N + 1)$ of these entries require the same computation of $\mathbf{1}^T \mathbf{b}_{i_t}$ and all other entries can be updated in constant time and space. Hence, the computation of all these entries can be done in $\mathcal{O}(|\mathcal{N}_t|)$ time and space. Finally, term $\mathbf{x}_{t+1}^T \mathbf{1}$ can be computed in $\mathcal{O}(|\mathcal{N}_t|)$ time and space. Hence, storing and updating of all entries $[\mathbf{x}_{t+1}]_j$ require $\mathcal{O}(t \sum_{i=1}^N k_i)$ time and space. Therefore, after $\mathcal{O}(\mathcal{T})$ steps, the overall time and memory complexity of Algorithm 4 are given by $\mathcal{O}(\mathcal{T}^2 \sum_{i=1}^N k_i)$ and $\mathcal{O}(\mathcal{T} \sum_{i=1}^N k_i)$ respectively. Using $\mathcal{T} = \mathcal{O}\left(\frac{n}{\varepsilon}\right)$ eventually gives $\mathcal{O}\left(\frac{n^2}{\varepsilon^2} \sum_{i=1}^N k_i\right)$ for time and $\mathcal{O}\left(\frac{n}{\varepsilon} \sum_{i=1}^N k_i\right)$ for space complexity (note that $n = \prod_{i=1}^N k_i$). \square

Algorithm 5 α^α -Oracle: Practical Multi-Agent Evaluation

```

1: Inputs: Number of trails  $\mathcal{N}$ , total number of iterations  $T$ , decaying learning
   rate  $\{\eta_t\}_{t=1}^T$ , penalty parameter  $\lambda$ ,  $\lambda$  decay rate  $\gamma > 1$ , and a constraint relax-
   ation term  $\delta$ , initialise  $p = 0$ .
2: while  $p \leq \mathcal{N}$  do:
3:   Initialise the strategy set  $\{\mathcal{S}_i^{[0]}\}$  by sub-sampling from  $\{\mathcal{S}_i\}$ 
4:   while  $\{\mathcal{S}_i^{[k]}\} \neq \{\mathcal{S}_i^{[k-1]}\}$  do:
5:     Compute total number of joint profiles  $n = \prod_{i=1}^N |\mathcal{S}_i^{[k]}|$ 
6:     Initialise a vector  $\mathbf{x}_0 = \frac{1}{n} \mathbf{1}$ 
7:     for  $t = 0 \rightarrow T - 1$  do:      //  $\alpha^\alpha$ -Rank update
8:       Uniformly sample one strategy profile  $i_t^{[k]} \sim \{1, \dots, n\}$ 
9:       Construct  $b_{i_t^{[k]}}^{[k]}$  as the  $i_t^{[k]}$  row of  $\mathbf{T}^{[k],T} - \mathbf{I}$ 
10:      Update  $\mathbf{x}_{t+1}^{[k]} = \mathbf{x}_t^{[k]} - \eta_t \nabla_{\mathbf{x}} f_{i_t^{[k]}}(\mathbf{x}_t^{[k]})$  by Eq. (4.8)
11:      Set  $\lambda_{t+1} = \lambda_t / \gamma$ 
12:      Get  $\boldsymbol{\pi}_{\text{joint}}^{[p],\text{top}}$  by ranking the prob. mass of  $\mathbf{v}^{[k]} = \frac{\mathbf{x}_T^{[k]}}{\|\mathbf{x}_T^{[k]}\|_1}$ 
13:      for each agent  $i$  do:      // The oracles (Section 4.4.3)
14:        Compute the best response  $\pi_i^*$  to  $\boldsymbol{\pi}_{\text{joint}}^{[p],\text{top}}$  by Eq. (4.11)
15:        Update the strategy set by  $\mathcal{S}_i^{[k]} = \mathcal{S}_i^{[k-1]} \cup \{\pi_i^*\}$ 
16:      Set  $p = p + 1$ 
17: Return: The best performing joint-strategy profile  $\boldsymbol{\pi}_{\text{joint},*}$  among  $\{\boldsymbol{\pi}_{\text{joint}}^{[1:\mathcal{N}],\text{top}}\}$ .

```

4.4.3 Efficient Exploration via Oracles

Stochastic sampling enables to solve α -Rank with no need to store the transition matrix \mathbf{T} ; however, the size of the column \mathbf{b}_i (i.e., $\prod_{i=1}^N k_i$) can still be prohibitively large. Here we further boost scalability of our method by introducing an *oracle* mechanism. The heuristic of oracles was first proposed in solving large-scale zero-sum matrix games ([McMahan et al., 2003](#)). The idea is to first create a sub-game in which all players are only allowed to play a restricted number of strategies, which are then expanded by adding each of the players' best-responses to their opponents; the sub-game will be replayed with agents' augmented strategy sets before a new round of best responses is computed.

The best response is assumed to be given by an *oracle* that can be simply implemented by a grid search, where given the top-rank profile $\boldsymbol{\pi}_{-i}^{\text{top}}$ at iteration k , the goal for agent i is to select the optimal π_i^* from a pre-defined strategy set \mathcal{S}_i to

maximise its reward:

$$\pi_i^* = \arg \max_{\pi_i \in \mathcal{S}_i} \mathbb{E}_{\pi_i, \pi_{-i}^{\text{top}}} \left[\sum_{h \geq 0} \gamma_i^h \mathcal{P}_i(s_h, u_{i,h}, u_{-i,h}) \right], \quad (4.11)$$

with s_h denoting the state, $u_{i,h} \sim \pi_i(\cdot | s_{i,h})$, $u_{-i,h} \sim \pi_{-i}^{\text{top}}(\cdot | s_{-i,h})$ denoting the actions from agent i and the opponents, respectively. Though the worse-case scenario of introducing oracles would require solving the original evaluation problem, our experimental results on large-scale systems demonstrate efficiency by converging early, with no need to recover the full games.

For a complete exposition, we summarise the pseudocode of our proposed method, named **α^α -Oracle**, in Algorithm 5. α^α -Oracle degenerates to α^α -Rank (lines 5 – 12) if one initialises strategy sets of agents by the full size at the beginning, i.e., $\{\mathcal{S}_i^{[0]}\} \triangleq \{\mathcal{S}_i\}$.

Providing valid convergence guarantee for α^α -Oracle is an exciting direction for future work. In fact, recently [Muller et al. \(2019\)](#) proposed a close idea of adopting an oracle mechanism into α -Rank without any stochastic solver however. Interestingly, it is reported that bad initialisation can lead to failures in recovering top-rank strategies. Contrary to the results reported in [Muller et al. \(2019\)](#), we rather demonstrate the effectiveness of our approach through running multiple trials of initialisation for $\{\mathcal{S}_i^{[0]}\}$. In addition, we also believe the stochastic nature of α^α -Oracle potentially prevents from being trapped by the poor local minimal from sub-games.

4.5 Experiments and Results

In this section, we demonstrate the scalability of α^α -Rank in successfully recovering optimal policies in self-driving car simulations and the Ising model – a setting with tens-of-millions of possible strategies. We note that these sizes are far beyond the capability of state-of-the-art methods; α -Rank ([Omidshafiei et al., 2019a](#)) considers at maximum 4 agents with 4 strategies. All of our experiments were run only on a single machine with 64 GB memory and 10-core Intel i9 CPU.

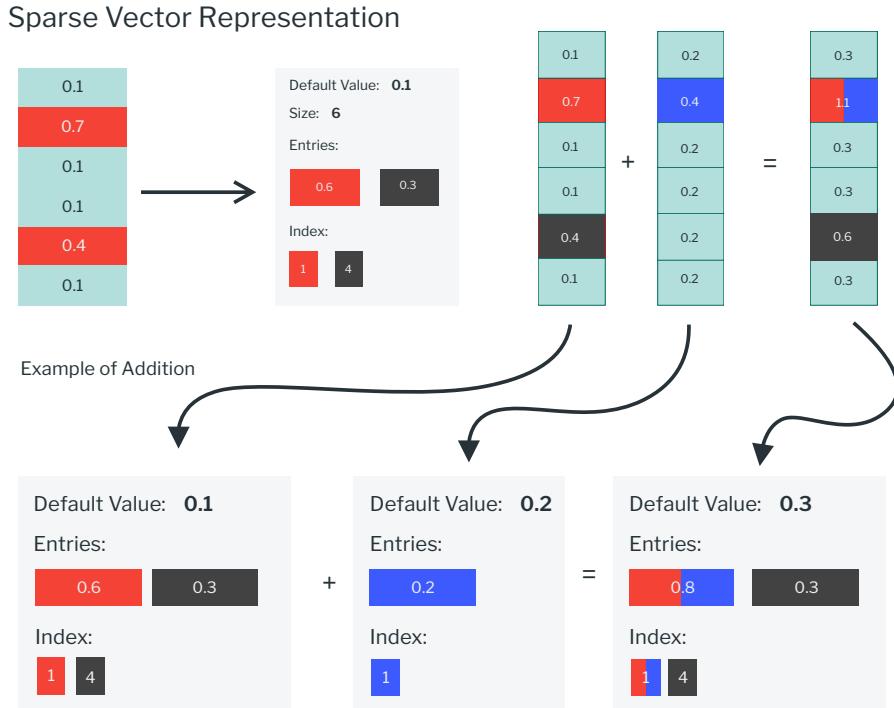


Figure 4.3: Sparse vector implementation in α^α -Rank with an example on addition.

4.5.1 Implementations of Sparse Vectors

During the implementation phase, we realised that the transition probability, $T^{[k]}$, of the Markov chain induces a sparsity pattern (each row and column in $T^{[k]}$ contains $\sum_{i=1}^N k_i^{[k]} - N + 1$ non-zero elements, check Section 4.4) that if exploited can lead to significant speed-up. To fully leverage such sparsity, we tailored a novel data structure for sparse storage and computations needed by Algorithm 5.

To fully leverage such sparsity, we design a new data structure (see Figure 4.3) for storage and computation. Compared to standard techniques (e.g., COO, CSR, and CRS⁸) that store (row, column, value) of a sparse vector, our data structure adopts a more efficient protocol that stores (defaults, positions, biases) leading to improvements in computational efficiency, which gives us additional advantages in computational efficiency. We reload the operations for such data structure, including *addition, scalar multiplication, dot product, element-wise square root, L1 norm*, see Figure 4.3.

⁸<https://docs.scipy.org/doc/scipy/reference/sparse.html>

Table 4.3: Hyper-parameter settings for the experiments. m : Population size, α : Ranking intensity, η : Learning rate

Experiments	m	α	η	Max. Iter.	β_1	λ	δ
NFG (w/o self-play)	50	$10^{-4} \sim 10^2$	1.0	1000	0.9	0.5	0.1
NFG (self-play)	50	$10^{-4} \sim 10^2$	0.03	1000	0.9	0.5	0.1
Random Matrix	n/a	n/a	0.01	1000	n/a	0.1	0.01
Highway Exp. (SGD)	40	1.0	15.0	2000	0.999	0.5	0.1
Highway Exp. (Oracle)	40	1.0	1.0	200	0.999	0.5	0.1
Ising Model	40	90.0	0.01	4000	0.999	0.5	0.1

4.5.2 Hyper-Parameter Settings

Before introducing the experimental results, I first list the hyper-parameter settings for each task in Table 4.3. For all of our experiments, the gradient updates include two phases: warm-up phase and Adam ([Kingma and Ba, 2014](#)) phase. In the warm-up phase, we used standard stochastic gradient descent; after that, we replace SGD with Adam until convergence. In practice, we find this yields faster convergence than normal stochastic gradient descent. As our algorithm does column sampling for the stochastic matrix (i.e. batch size equals one), adding a momentum term intuitively stabilises the learning. The warm-up step is 100 for all experiments. We also implement infinite α in [Lanctot et al. \(2019\)](#), when calculating transition matrix (or its column), where our noise term is set to be 0.01. For most of our experiments involving α^α -Rank, we set the terminating condition when the gradient norm is less than 10^{-9} . However, for Random Matrix experiment, we set the terminating gradient norm to be 10^{-2} , learning rate to be in between $15 - 17$, α (ranking intensity) to be in between $1 - 2.5$, number of population to be between $25 - 55$ (in integer). For all of the Adam experiments, after the warmup-step, we choose to decay δ and λ by 0.999 for each time steps, where we have δ always to be 0.1. Similarly, λ starts at the value 0.5. In the speed and memory experiments, the decay is set to be 0.9.

4.5.3 Analysis on Normal-Form Games

As Algorithm 5 is a generalisation (in terms of scalability) of α -Rank, it is instructive to validate the correctness of our results on three simple matrix games. Our algorithm provides the expected ranking in all three normal-form games shown in Figure 4.4, which is consistent with α -Rank’s results ([Omidshafiei et al., 2019a](#)).

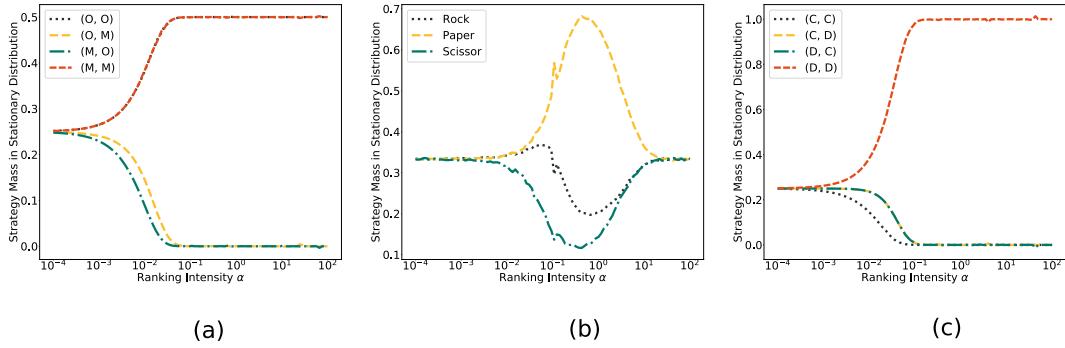


Figure 4.4: Ranking intensity sweep on (a) Battle of Sexes (b) Biased RPS (c) Prisoner’s Dilemma.

Battle of sexes. Battle of sexes is an asymmetric game $R_{OM} = \begin{bmatrix} 3,2 & 0,0 \\ 0,0 & 2,3 \end{bmatrix}$. α^α -Rank suggests that populations would spend an equal amount of time on the profile (O,O) and (M,M) during the evolution. The distribution mass of (M,O) drops to 0 faster than that of (O,M), this is because deviating from (M,O) for either player has a larger gain (from 0 to 3) than deviating from (O,M) (from 0 to 2).

Biased Rock-Paper-Scissor. We consider the biased RPS game $R_{RPS} = \begin{bmatrix} 0 & -0.5 & 1 \\ 0.5 & 0 & -0.1 \\ -1 & 0.1 & 0 \end{bmatrix}$. As it is a single-population game, we adopt the transitional probability matrix of Eqn. 11 in [Omidshafiei et al. \(2019a\)](#). Such game has the inherent structure that Rock/Paper/Scissor is equally likely to be invaded by a mutant, e.g., the scissor population will always be fixated by the rock population; therefore, our method suggests the long-term survival rate for all three strategies are the same $(\frac{1}{3}, \frac{1}{3}, \frac{1}{3})$. Note this is different from the Nash equilibrium solution that is $(\frac{1}{16}, \frac{5}{8}, \frac{5}{16})$.

Prison’s Dilemma. In prison’s dilemma $R_{CD} = \begin{bmatrix} -1,-1 & -3,0 \\ 0,-3 & -2,-2 \end{bmatrix}$, cooperation is an evolutionary transient strategy since the cooperation player can always be exploited by the defection player. Our method thus yields (D,D) as the only strategy profile that could survive in the long-term evolution.

4.5.4 Analysis on Random Matrices

To further assess scalability, we measured the time and memory needed by our method for computing stationary distributions of varying sizes of simulated random matrices. Baselines included eigenvalue decomposition from Numpy, optimisation tools from PyTorch, and α -Rank from OpenSpiel ([Lanctot et al., 2019](#)). We terminated execution of α^α -Rank when gradient norms fell short of a predefined

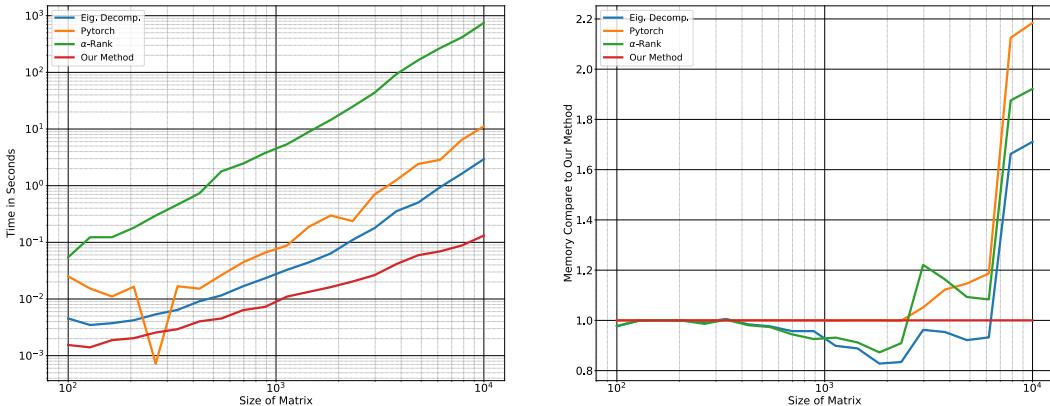


Figure 4.5: Comparisons of time and memory complexities on varying sizes of random matrices.

threshold of **0.01**. According to Figure 4.5, α^α -Rank can achieve three orders of magnitude reduction in time (i.e. 1000x faster) compared to the default α -Rank implementation from [Lanctot et al. \(2019\)](#). Memory-wise, our method uses only half of the space when considering, for instance, $10^4 \times 10^4$ matrices.

4.5.5 Autonomous Driving on Highway

Having assessed correctness and scalability, we now present novel application domains on large-scale *multi-agent/multi-player* systems. For that, we made use of high-way ([Leurent, 2018](#)); an environment for simulating self-driving scenarios with social vehicles designed to mimic real-world traffic flow. We conducted a ranking experiment involving 5 agents each with 5 strategies, i.e., a strategy space in the order of $\mathcal{O}(5^5)$ (3125 possible strategy profiles). Agent strategies varied between “rational” and “dangerous” drivers, which we encoded using different reward functions during training. Specifically, we have Table 4.4. By setting this, we can

Table 4.4: Reward settings in Self-driving Car Simulation.

	Collision Reward	Speed Reward
Rational driver	-2.0	0.4
Risky driver 1	10.0	10.0
Risky driver 2	20.0	10.0
Risky driver 3	30.0	10.0
Risky driver 4	40.0	10.0

make sure, at upfront, the best joint-strategy strategy is for all cars to drive ratio-

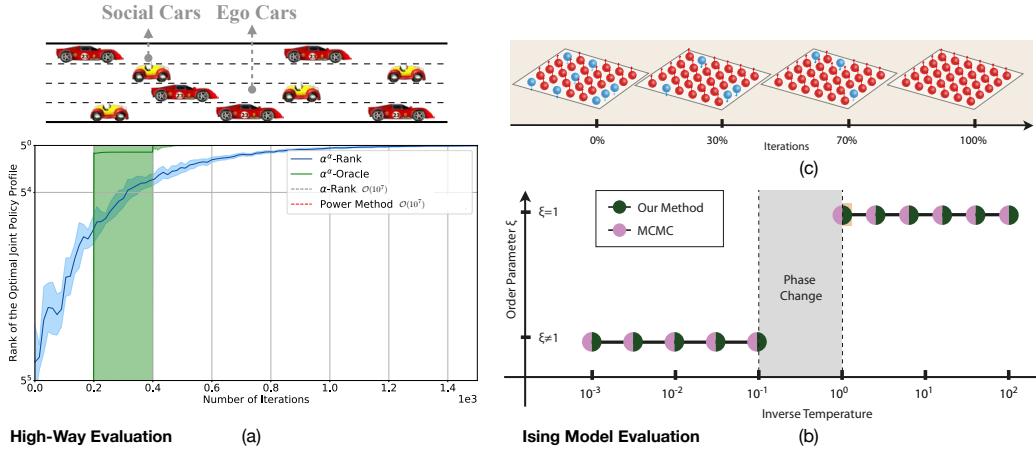


Figure 4.6: Large-scale multi-agent evaluations. (a) Convergence of the optimal joint-strategy profile in self-driving simulation. (b) Status of the Ising-model equilibrium measured by $\xi = \frac{|N_\uparrow - N_\downarrow|}{|N|}$. (c) Change of the top-rank profile from α^α -Oracle.

nally.

The environmental reward given to each agent is calculated by:

$$\text{Collision Reward} + \frac{\text{Current Velocity}}{\text{Default Velocity}} \times \text{Proportional Speed Reward}$$

Collision Reward is calculated when an agent collided with either social car or other agents. Note that although we define five types of driving behaviours (one rational + four dangerous) by letting each controlled car have a different ego reward function during training, the reward we report is the unchangeable environmental reward.

All of our value iteration agents are based on the default environment discretisation in Leurent (2018), which represents the environment in terms of time to collision MDP, taking into account that the other agents are moving in constant speed. For all experiments, we run value-iteration for 200 steps with the discounting factor of 0.99. For each controllable car, the default speed is randomised to be between 10 to 25, while the social cars are randomised to be between 23 to 25.

We considered both α^α -Rank and α^α -Oracle, and reported the results by running 1000 random seeds. We set α^α -Oracle to run 200 iterations of gradient updates in solving the top-rank strategy profile (lines 8 – 12 in Algorithm 5). Results depicted in Figure 4.6(a) clearly demonstrate that both our proposed methods are

capable of recovering the correct highest-ranking strategy profile. α^α -Oracle converges faster than α^α -Rank, which we believe is due to the oracle mechanism saving time in inefficiently exploring “dangerous” drivers upon one observation. We also note that although such a problem is feasible using α -Rank and the Power Method, our results achieve four orders of reduction in the number of iterations.

4.5.6 Ising Model

We repeated the above experiment on the Ising model ([Ising, 1925](#)) that is typically used for describing ferromagnetism in statistical mechanics. It assumes a system of magnetic spins, where each spin a^j is either an up-spin, \uparrow , or down-spin, \downarrow . The system energy is defined by $E(\mathbf{a}, h) = -\sum_j(h^j a^j + \frac{\lambda}{2} \sum_{k \neq j} a^j a^k)$ with h^j and λ being constant coefficients. The probability of one spin configuration is $P(\mathbf{a}) = \frac{\exp(-E(\mathbf{a}, h)/\tau)}{\sum_a \exp(-E(\mathbf{a}, h)/\tau)}$ where τ is the environmental temperature. Finding the equilibrium of the system is notoriously hard because it is needed to enumerate all possible configurations in computing $P(\mathbf{a})$. Traditional approaches include Markov Chain Monte Carlo (MCMC). An interesting phenomenon is the *phase change*, i.e., the spins will reach an equilibrium in the low temperatures. With the increasing τ , such equilibrium will suddenly break, and the system becomes chaotic.

Here we try to observe the phase change through multi-agent evaluation methods. We treat each spin as an agent, and the reward is $r^j = h^j a^j + \frac{\lambda}{2} \sum_{k \neq j} a^j a^k$. We consider the top-rank strategy profile from α^α -Oracle as the system equilibrium and compare it against the ground truth from MCMC. We consider a 5×5 2D model which induces a prohibitively-large strategy space of the size 2^{25} (≈ 33 million strategies) to which existing methods are inapplicable. Figure 4.6(b) illustrates that our method identifies the same phase change as that of MCMC. We also show an example of how α^α -Oracle’s top-ranked profile finds the system’s equilibrium when $\tau = \alpha = 1$ in Figure 4.6(c). Note that the problem of 25 agent with 2 strategies goes far beyond the capability of α -Rank on one single machine (billions of elements in \mathbf{T}).

4.6 Chapter Appendix

	AlphaGo Zero	AlphaZero	AlphaZero Chess	AlphaStar
Reported in paper	3-Day	40-Day	13-Day	9 hours
Day	3	40	13	0.375
Hours of training	72	960	312	9
Matches played	4,900,000	29,000,000	140,000,000	44,000,000
Network	20 blocks	40 blocks	20 blocks	21 blocks
MCTS simulations	1600	1600	800	800
Seconds per move	0.4	0.8	0.2	0.04
Training: # GPUs	64	64	64	64
Training: # CPUs	19	19	19	19
Self-play: # TPUs/player	4	4	4	4
# TPUs (v1) required	6382.098765	5665.740741	21039.88604	8,691
# TPUs (v3) required				28,800
GCP Data				
TPU \$/h				
GPU \$/h (best)				
CPU \$/h (best)	0.01	0.01	0.01	0.01
Estimates				
# moves in a Go match	211	211	211	40
How many TPUs ?				
Seconds/match/player	84.4	168.8	42.2	1.6
Matches/hour/player	42.65402844	21.32701422	85.30805687	2250
Matches/total time/player	3071.090047	20473.93365	26616.11374	20250
# Players required	1595.524691	1416.435185	5259.97151	2172.839506
# TPUs required	6382.098765	5665.740741	21039.88604	8691.358025
Total compute (PF)	0	0	0	0
pfs-days	0	0	0	0
Costs				
Training total cost (GPU) \$	\$0.00	\$0.00	\$0.00	\$0.00
Training total cost (CPU) \$	\$13.68	\$182.40	\$59.28	\$1.71
Self-play: TPU total cost \$	\$0.00	\$0.00	\$0.00	\$0.00
Final Estimate \$	\$13.68	\$182.40	\$59.28	\$1.71
Money cost to alpha-rank	\$95.76	\$1,276.80	\$414.96	\$1.71
per GPU day	19,338	229,190	274,351	3,283
per GPU day cost to alpha-rank	135,368	1,604,327	1,920,454	3,283

Figure 4.7: Cost breakup for Table 4.2. The full spread sheet can also be found at https://docs.google.com/spreadsheets/d/1XJM4C9WWRJTBrIzSuBkpAnleoo3B9TanMC17nWVIn_s.

4.7 Chapter Summary

In this chapter, I presented major bottlenecks prohibiting α -Rank from scaling beyond tens of agents. Dependent on the type of input, α -Rank's time and memory complexities can quickly become exponential. I further argued that the input that α -Rank needs can lead to notoriously difficult NP -hard problems. To substantiate my claims, I empirically validated my claims by presenting dollars spent as a non-refutable metric. Realising these problems, I proposed a scalable alternative for many-agent evaluation based on stochastic optimisation and double oracles, along with rigorous scalability results on a variety of benchmarks. For future work, I plan to understand the relation between α -Rank's solution and that of a Nash equilibrium. Second, I will attempt to conduct a theoretical study on the convergence of our proposed α^α -Oracle algorithm.

Chapter 5

Many-Agent Policy Learning: A Mean-Field Approach

In the previous chapter, I have looked into the problem of policy evaluation in the many-agent context. In this chapter, I will move onwards to the core problem of many-agent policy learning. Existing multi-agent RL methods are limited typically to a small number of agents. When the agent number increases vastly, the learning becomes intractable due to the curse of dimensionality and the exponential growth of agent interactions. As a potential solution, here I present *Mean-Field MARL*, where the interactions within the population of agents are approximated by those between a single agent and the average effect from the overall population or neighbouring agents; the interplay between the two entities is mutually reinforced: the learning of the individual agent's optimal policy depends on the dynamics of the population, while the dynamics of the population change according to the collective patterns of the individual policies. I develop practical mean-field Q-learning and mean-field Actor-Critic algorithms and analyse the convergence of the solution to Nash equilibrium. Experiments on Gaussian squeeze, the Ising model, and many-agent battle games justify the learning effectiveness of my mean-field approaches. In addition, I report the first result of solving the Ising model through a MARL approach.

5.1 Background and Motivation

MARL is concerned with a set of autonomous agents that share a common environment (Busoniu et al., 2008). Learning in MARL is fundamentally difficult since agents not only interact with the environment but also with each other. Independent Q -learning (Tan, 1993) that considers other agents as a part of the environment often fails as the multi-agent setting breaks the theoretical convergence guarantee and makes the learning unstable, that is, changes in the policy of one agent will affect those of the others, and vice versa (Matignon et al., 2012).

Instead, accounting for the extra information from *conjecturing* the policies of other agents is beneficial to each single learner (Foerster et al., 2017c; Lowe et al., 2017a). Studies show that an agent who learns the effect of joint actions has better performance than those who do not in many scenarios, including cooperative games (Panait and Luke, 2005), zero-sum stochastic games (Littman, 1994), and general-sum stochastic games (Hu and Wellman, 2003; Littman, 2001a).

The existing equilibrium-solving approaches, although principled, are only capable of solving a handful of agents (Bowling and Veloso, 2002; Hu and Wellman, 2003). The computational complexity of directly solving (Nash) equilibrium would prevent them from applying to the situations with a large group or even a population of agents. Yet, in practice, many cases do require strategic interactions among a large number of agents, such as the gaming bots in Massively Multiplayer Online Role-Playing Game (Jeong et al., 2015), the trading agents in stock markets (Troy, 1997), or the online advertising bidding agents (Wang et al., 2017a).

In this paper, we tackle MARL when a large number of agents co-exist. We consider a setting where each agent is directly interacting with a finite set of other agents; through a chain of direct interactions, any pair of agents are interconnected globally (Blume, 1993). The scalability is solved by employing Mean-Field theory (Stanley, 1971) – the interactions within the population of agents are approximated by that of a single agent played with the average effect from the overall (local) population. The learning is mutually reinforced between two entities rather than many entities: the learning of the individual agent’s optimal policy is based on the dynam-

ics of the agent population; meanwhile, the dynamics of the population is updated according to the individual policies. Based on such formulation, we develop practical mean-field Q -learning and mean-field Actor-Critic algorithms, and discuss the convergence of our solution under certain assumptions. Our experiment on a simple multi-agent resource allocation problem shows that our mean-field MARL is capable of learning over many-agent interactions when others fail. We also demonstrate that mean-field MARL manages to learn and solve the Ising model with temporal-difference learning without even explicitly knowing the energy function. Finally, in a mixed cooperative-competitive battle game, we show that the mean-field MARL achieves high winning rates against other baselines previously reported for many agent systems.

5.2 Related Work

We continue our discussion on related work from the previous section and make comparisons with existing techniques in a greater scope. Our work follows the same direction as [Bowling and Veloso \(2002\)](#); [Hu and Wellman \(2003\)](#); [Littman \(1994\)](#) on adapting a Stochastic Game ([van der Wal et al., 1981](#)) into the MARL formulation. Specifically, [Littman \(1994\)](#) addressed two-player zero-sum stochastic games by introducing a “minimax” operator in Q -learning, whereas [Hu and Wellman \(2003\)](#) extended it to the general-sum case by learning a Nash equilibrium in each stage game and considering a mixed strategy. Nash-Q learning is guaranteed to converge to Nash strategies under the (strong) assumption that there exists an equilibrium for every stage game. In the situation where agents can be identified as either “friends” or “foes” ([Littman, 2001a](#)), one can simply solve it by alternating between fully cooperative and zero-sum learning. Considering the convergence speed, [Littman and Stone \(2005\)](#) and [de Cote and Littman \(2008\)](#) draw on the *folk theorem* and acquired a polynomial-time Nash equilibrium algorithm for repeated stochastic games, while [Bowling and Veloso \(2002\)](#) tweaked the learning rate to improve the convergence.

The recent treatment of MARL was using deep neural networks as function

approximator. In addressing the non-stationary issue in MARL, various solutions have been proposed including neural-based opponent modelling (He and Boyd-Graber, 2016), policy parameter sharing (Gupta et al., 2017), etc. Researchers have also adopted the paradigm of *centralised training with decentralised execution* for multi-agent policy-gradient learning: BICNET (Peng et al., 2017b), COMA (Foerster et al., 2018b) and MADDPG (Lowe et al., 2017a), which allows the centralised critic Q -function to be trained with the actions of other agents, while the actor needs only local observation to optimise the agent’s policy.

The above MARL approaches limit their studies mostly to tens of agents. As the number of agents grows larger, not only the input space of Q grows exponentially, but most critically, the accumulated noises by the exploratory actions of other agents make the Q -function learning no longer feasible. Our work addresses the issue by employing the mean-field approximation (Stanley, 1971) over the joint action space. The parameters of the Q -function are independent of the number of agents as it transforms the interactions of multiple agents into the interactions between single agent *v.s.* the distribution of the neighbouring agents. This effectively alleviates the problem of the exploratory noise (Colby et al., 2015) caused by many other agents, and allows each agent to determine which actions are beneficial to itself.

Our work is also closely related to the recent development of mean-field games (MFG) (Huang et al., 2006; Lasry and Lions, 2007; Weintraub et al., 2006). MFG studies population behaviours resulting from the aggregations of decisions taken from individuals. Mathematically, the dynamics are governed by a set of two stochastic differential equations that model the backward dynamics of an individual’s value function, and the forward dynamics of the aggregate distribution of agent population. Although backward equation equivalently describes what the Bellman equation indicates in the MDP, the primary goal for MFG is rather for model-based planning and to infer the movements of the individual density through time. The mean-field approximation (Stanley, 1971) is also employed in physics, but our work is different in that we focus on a model-free solution of learning optimal actions

when the dynamics of the system and the reward function are unknown. Very recently, [Yang et al. \(2017\)](#) built a connection between MFG and RL. Their focus is, however, on applying inverse RL algorithms to learn both the reward function and the forward dynamics of the MFG from the policy data. In contrast, our goal is to form a computable Q -learning algorithm under the framework of temporal difference learning.

5.3 Preliminary and Notations

MARL intersects between reinforcement learning and game theory. The marriage of the two gives rise to the general framework of *stochastic game* ([Shapley, 1953](#)).

5.3.1 Stochastic Game

An N -agent (or, N -player) stochastic game Γ is formalised by the tuple $\Gamma \triangleq (\mathcal{S}, \mathcal{A}^1, \dots, \mathcal{A}^N, r^1, \dots, r^N, p, \gamma)$, where \mathcal{S} denotes the state space, and \mathcal{A}^j is the action space of agent $j \in \{1, \dots, N\}$. The reward function for agent j is defined as $r^j : \mathcal{S} \times \mathcal{A}^1 \times \dots \times \mathcal{A}^N \rightarrow \mathbb{R}$, determining the immediate reward. The transition probability $p : \mathcal{S} \times \mathcal{A}^1 \times \dots \times \mathcal{A}^N \rightarrow \Omega(\mathcal{S})$ characterises the stochastic evolution of states in time, with $\Omega(\mathcal{S})$ being the collection of probability distributions over the state space \mathcal{S} . The constant $\gamma \in [0, 1)$ represents the reward discount factor across time. At time step t , all agents take actions simultaneously, each receives the immediate reward r_t^j as a consequence of taking the previous joint actions.

The agents choose actions according to their policies, also known as strategies. For agent j , the corresponding policy is defined as $\pi^j : \mathcal{S} \rightarrow \Omega(\mathcal{A}^j)$, where $\Omega(\mathcal{A}^j)$ is the collection of probability distributions over agent j 's action space \mathcal{A}^j .

Let $\boldsymbol{\pi} \triangleq [\pi^1, \dots, \pi^N]$ denote the joint policy of all agents; we assume, as one usually does, $\boldsymbol{\pi}$ to be time-independent, which is referred to be *stationary*. Provided an initial state s , the value function of agent j under the joint policy $\boldsymbol{\pi}$ is written as the expected cumulative discounted future reward:

$$v_{\boldsymbol{\pi}}^j(s) = v^j(s; \boldsymbol{\pi}) = \sum_{t=0}^{\infty} \gamma^t \mathbb{E}_{\boldsymbol{\pi}, p} [r_t^j | s_0 = s, \boldsymbol{\pi}] \quad (5.1)$$

The Q -function (or, the action-value function) can then be defined within the framework of N -agent game based on the Bellman equation given the value function in Eq. (5.1) such that the Q -function $Q_{\pi}^j : \mathcal{S} \times \mathcal{A}^1 \times \dots \times \mathcal{A}^N \rightarrow \mathbb{R}$ of agent j under the joint policy π can be formulated as

$$Q_{\pi}^j(s, \mathbf{a}) = r^j(s, \mathbf{a}) + \gamma \mathbb{E}_{s' \sim p} [v_{\pi}^j(s')] , \quad (5.2)$$

where s' is the state at the next time step. The value function v_{π}^j can be expressed in terms of the Q -function in Eq. (5.2) as

$$v_{\pi}^j(s) = \mathbb{E}_{\mathbf{a} \sim \pi} [Q_{\pi}^j(s, \mathbf{a})] . \quad (5.3)$$

The Q -function for N -agent game in Eq. (5.2) extends the formulation for single-agent game by considering the joint action taken by all agents $\mathbf{a} \triangleq [a^1, \dots, a^N]$, and by taking the expectation over the joint action in Eq. (5.3).

We formulate MARL by the stochastic game with a discrete-time non-cooperative setting, *i.e.* no explicit coalitions are considered. The game is assumed to be incomplete but to have perfect information (Littman, 1994), *i.e.* each agent knows neither the game dynamics nor the reward functions of others. Still, it can observe and react to the previous actions and the others' immediate rewards.

5.3.2 Nash Q-Learning

In MARL, the objective of each agent is to learn an optimal policy to maximise its value function. Optimising the v_{π}^j for agent j depends on the joint policy π of all agents, the concept of *Nash equilibrium* in stochastic games is therefore of great importance (Hu and Wellman, 2003). It is represented by a particular joint policy $\pi_* \triangleq [\pi_*^1, \dots, \pi_*^N]$ such that for all $s \in \mathcal{S}$, $j \in \{1, \dots, N\}$ and all valid π^j , it satisfies

$$v^j(s; \pi_*) = v^j(s; \pi_*^j, \pi_*^{-j}) \geq v^j(s; \pi^j, \pi_*^{-j}).$$

Here we adopt a compact notation for the joint policy of all agents except j as $\pi_*^{-j} \triangleq [\pi_*^1, \dots, \pi_*^{j-1}, \pi_*^{j+1}, \dots, \pi_*^N]$.

In a Nash equilibrium, each agent acts with the *best response* π_*^j to others, provided that all other agents follow the policy π_*^{-j} . It has been shown that, for a N -agent stochastic game, there is at least one Nash equilibrium with stationary policies (Fink et al., 1964). Given a Nash policy π_* , the Nash value function $v^{\text{Nash}}(s) \triangleq [v_{\pi_*}^1(s), \dots, v_{\pi_*}^N(s)]$ is calculated with all agents following π_* from s_0 onwards.

Nash Q -learning (Hu and Wellman, 2003) defines an iterative procedure with two alternating steps for computing the Nash policy: 1) solving the Nash equilibrium of the current stage game defined by $\{\mathbf{Q}_t\}$ using the Lemke-Howson algorithm (Lemke and Howson, 1964), 2) improving the estimation of the Q -function with the new Nash equilibrium value. It can be proved that under certain assumptions, the Nash operator $\mathcal{H}^{\text{Nash}}$ in the following expression forms a contraction mapping:

$$\mathcal{H}^{\text{Nash}} \mathbf{Q}(s, \mathbf{a}) = \mathbb{E}_{s' \sim p} [\mathbf{r}(s, \mathbf{a}) + \gamma v^{\text{Nash}}(s')] \quad (5.4)$$

where $\mathbf{Q} \triangleq [Q^1, \dots, Q^N]$, and $\mathbf{r}(s, \mathbf{a}) \triangleq [r^1(s, \mathbf{a}), \dots, r^N(s, \mathbf{a})]$. The Q -function will eventually converge to the value received in a Nash equilibrium of the game, referred to as the *Nash Q-value*.

5.4 Mean-Field MARL

The dimension of joint action \mathbf{a} grows proportionally w.r.t. the number of agents N . As all agents act strategically and evaluate simultaneously their value functions based on the joint actions, it becomes infeasible to learn the standard Q -function $Q^j(s, \mathbf{a})$. To address this issue, we factorise the Q -function as follows:

$$Q^j(s, \mathbf{a}) = \frac{1}{N^j} \sum_{k \in \mathcal{N}(j)} Q^j(s, a^j, a^k), \quad (5.5)$$

where $\mathcal{N}(j)$ is the index set of the neighbouring agents of agent j with the size $N^j = |\mathcal{N}(j)|$ determined by the settings of different applications. It is worth noting that the pairwise approximation of the agent and its neighbours, while significantly reducing the complexity of the interactions among agents, still preserves global interactions between any pair of agents implicitly (Blume, 1993). This approach

is commonly used in recommendation (e.g., factorisation machine) (Rendle, 2012) and learning to rank (Cao et al., 2007).

5.4.1 Mean-Field Approximation in RL

The pairwise interaction $Q^j(s, a^j, a^k)$ as in Eq. (5.5) can be approximated using the mean-field theory (Stanley, 1971). Here we consider discrete action spaces, where the action a^j of agent j is a discrete categorical variable represented as the one-hot encoding with each component indicating one of the D possible actions: $a^j \triangleq [a_1^j, \dots, a_D^j]$. We calculate the *mean* action \bar{a}^j based on the neighbourhood $\mathcal{N}(j)$ of agent j , and express the one-hot action a^k of each neighbour k in terms of the sum of \bar{a}^j and a small fluctuation $\delta a^{j,k}$ as

$$a^k = \bar{a}^j + \delta a^{j,k}, \quad \text{where } \bar{a}^j = \frac{1}{N^j} \sum_k a^k, \quad (5.6)$$

where $\bar{a}^j \triangleq [\bar{a}_1^j, \dots, \bar{a}_D^j]$ can be interpreted as the empirical distribution of the actions taken by agent j 's neighbours. By Taylor's theorem, the pairwise Q -function $Q^j(s, a^j, a^k)$, if twice-differentiable w.r.t. the action a^k taken by neighbour k , can be expanded and expressed as

$$\begin{aligned} Q^j(s, \mathbf{a}) &= \frac{1}{N^j} \sum_k Q^j(s, a^j, a^k) \\ &= \frac{1}{N^j} \sum_k \left[Q^j(s, a^j, \bar{a}^j) + \nabla_{\bar{a}^j} Q^j(s, a^j, \bar{a}^j) \cdot \delta a^{j,k} \right. \\ &\quad \left. + \frac{1}{2} \delta a^{j,k} \cdot \nabla_{\bar{a}^{j,k}}^2 Q^j(s, a^j, \bar{a}^{j,k}) \cdot \delta a^{j,k} \right] \\ &= Q^j(s, a^j, \bar{a}^j) + \nabla_{\bar{a}^j} Q^j(s, a^j, \bar{a}^j) \cdot \left[\frac{1}{N^j} \sum_k \delta a^{j,k} \right] \\ &\quad + \frac{1}{2N^j} \sum_k \left[\delta a^{j,k} \cdot \nabla_{\bar{a}^{j,k}}^2 Q^j(s, a^j, \bar{a}^{j,k}) \cdot \delta a^{j,k} \right] \end{aligned} \quad (5.8)$$

$$\begin{aligned} &= Q^j(s, a^j, \bar{a}^j) + \frac{1}{2N^j} \sum_k R_{s,a^j}^j(a^k) \\ &\approx Q^j(s, a^j, \bar{a}^j) . \end{aligned} \quad (5.9)$$

where $R_{s,a}^j(a^k) \triangleq \delta a^{j,k} \cdot \nabla_{\tilde{a}^{j,k}}^2 Q^j(s, a^j, \tilde{a}^{j,k}) \cdot \delta a^{j,k}$ denotes the Taylor polynomial's remainder with $\tilde{a}^{j,k} = \bar{a}^j + \varepsilon^{j,k} \delta a^{j,k}$ and $\varepsilon^{j,k} \in [0, 1]$. In Eq. (5.8), $\sum_k \delta a^k = 0$ by Eq. (5.6) such that the first-order term is dropped. From the perspective of agent j , the action a^k in the second-order remainders $R_{s,a}^j(a^k)$ is chosen based on the external action distribution of agent k , $R_{s,a}^j(a^k)$ is thus essentially a random variable. In fact, one can further prove that the remainder $R_{s,a}^j(a^k)$ is bounded within a symmetric interval $[-2M, 2M]$ under the mild condition of the Q -function $Q^j(s, a^j, a^k)$ being M -smooth (e.g. the linear function); as a result, $R_{s,a}^j(a^k)$ acts as a small fluctuation near zero. Specifically, we have the following lemma.

Lemma 2. *Suppose that Q is M -smooth, where its gradient ∇Q is Lipschitz-continuous with constant M such that for all a, \bar{a}*

$$\|\nabla Q(a) - \nabla Q(\bar{a})\|_2 \leq M\|a - \bar{a}\|_2, \quad (5.10)$$

where $\|\cdot\|_2$ indicates the ℓ_2 -norm, then each single remainder term $R_{s,a}^j(a^k)$ in Eq. (5.9) is bounded in $[-2M, 2M]$.

Proof. Recall Eq. (5.9) that we approximate the action a^k taken by the neighbouring agent k with the mean action \bar{a} calculated from the neighbourhood $\mathcal{N}(j)$. The state s and the action a^j of the central agent j can be considered as fixed parameters; the indices j, k of agents are essentially irrelevant to the derivation. With those omitted for simplicity, we rewrite the expression of the pairwise Q -function as $Q(a) \triangleq Q^j(s, a^j, a^k)$. With the Lagrange's mean value theorem, we have

$$\begin{aligned} \nabla Q(a) - \nabla Q(\bar{a}) &= \nabla Q(\bar{a} + 1 \cdot (a - \bar{a})) - \nabla Q(\bar{a}) \\ &= \nabla^2 Q(\bar{a} + \tau \cdot (a - \bar{a})) \cdot (a - \bar{a}), \quad \text{where } \tau \in [0, 1]. \end{aligned}$$

Take the ℓ_2 -norm on the both sides of the above equation, it follows from the smoothness condition that

$$\|\nabla Q(a) - \nabla Q(\bar{a})\|_2 = \|\nabla^2 Q(\bar{a} + \tau \cdot (a - \bar{a})) \cdot (a - \bar{a})\|_2 \leq M\|a - \bar{a}\|_2.$$

Define $\delta a \triangleq a - \bar{a}$ and the normalised vector $\delta \hat{a} \triangleq a - \bar{a} / \|a - \bar{a}\|_2$ with $\|\delta \hat{a}\|_2 = 1$, it follows from the above inequality

$$\|\nabla^2 Q(a + \tau \cdot \delta a) \cdot \delta \hat{a}\|_2 \leq M.$$

By arbitrary choice of (the unnormalised vector) δa such that the magnitude $\|\delta a\|_2 \rightarrow 0$, it follows from above that

$$\|\nabla^2 Q(a) \cdot \delta \hat{a}\|_2 \leq M.$$

By aligning (the normalised vector) $\delta \hat{a}$ in the direction of the eigenvectors of the Hessian matrix $\nabla^2 Q$, we can obtain for any eigenvalue λ of $\nabla^2 Q$ that

$$\|\nabla^2 Q(a) \cdot \delta \hat{a}\|_2 = \|\lambda \cdot \delta \hat{a}\|_2 = |\lambda| \cdot \|\delta \hat{a}\|_2 \leq M,$$

which indicates that all eigenvalues of $\nabla^2 Q$ can be bounded in the symmetric interval $[-M, M]$.

As the Hessian matrix $\nabla^2 Q$ is real symmetric and hence diagonalisable, there exists an orthogonal matrix U such that $U^\top [\nabla^2 Q] U = \Lambda \triangleq \text{diag}[\lambda_1, \dots, \lambda_D]$. It then follows that

$$\begin{aligned} \delta a \cdot \nabla^2 Q \cdot \delta a &= [U \delta a]^\top \Lambda [U \delta a] = \sum_{i=1}^D \lambda_i [U \delta a]_i^2, \\ \text{with } -M \|U \delta a\|_2 &\leq \sum_{i=1}^D \lambda_i [U \delta a]_i^2 \leq M \|U \delta a\|_2 \end{aligned}$$

Recall the definition $\delta a = a - \bar{a}$ in Eq. (5.6), where a is the one-hot encoding for D actions, and \bar{a} is a D -dimensional multinomial distribution. We can know that

$$\|U \delta a\|_2 = \|\delta a\|_2 = (a - \bar{a})^\top (a - \bar{a}) = a^\top a + \bar{a}^\top \bar{a} - \bar{a}^\top a - a^\top \bar{a} = 2(1 - \bar{a}_i) \leq 2,$$

where i represents the specific action a has represented such that $a_{i'} = 0$ for $i' \neq i$.

With all elements assembled, we have proved that each single remainder term

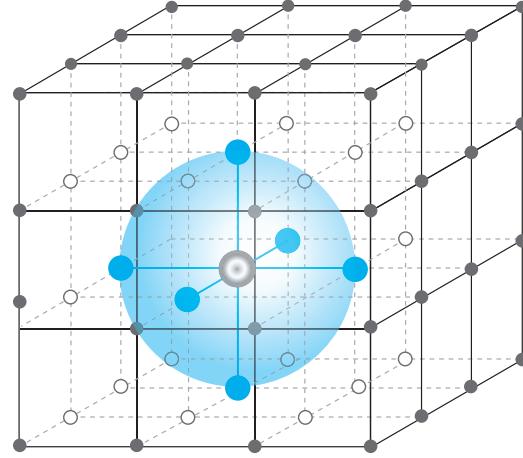


Figure 5.1: Mean-field approximation. Each agent is represented as a node in the grid, which is only affected by the **mean** effect from its **neighbours** (the blue area). Many-agent interactions are effectively converted into two-agent interactions.

$R_{s,a}^j(a^k)$ in Eq. (5.9) is bounded in $[-2M, 2M]$. Furthermore, with the assumptions of homogeneity and locality on all agents within the neighbourhood, the remainders tend to cancel each other, leading to only $Q^j(s, a^j, \bar{a}^j)$ left in Eq. (5.9). \square

As illustrated in Figure 5.1, with the mean-field approximation, the pairwise interactions $Q^j(s, a^j, a^k)$ between agent j and each neighbouring agent k are simplified as that between j , the *central* agent, and the virtual *mean* agent, that is abstracted by the mean effect of all neighbours within j 's neighbourhood. The interaction is thus simplified and expressed by the mean-field Q -function $Q^j(s, a^j, \bar{a}^j)$ in Eq. (5.9). During the learning phase, given an experience $e = (s, \{a^k\}, \{r^j\}, s')$, the Q -function is updated in a recurrent manner:

$$Q_{t+1}^j(s, a^j, \bar{a}^j) = (1 - \alpha) Q_t^j(s, a^j, \bar{a}^j) + \alpha [r^j + \gamma v_t^j(s')], \quad (5.11)$$

where α_t denotes the learning rate, and \bar{a}^j is the mean action of all neighbours of agent j as defined in Eq. (5.6). The mean-field value function $v_t^j(s')$ for agent j at time t in Eq. (5.11) is

$$v_t^j(s') = \sum_{a^j} \pi_t^j(a^j | s', \bar{a}^j) \mathbb{E}_{\bar{a}^j(\mathbf{a}^{-j}) \sim \pi_t^{-j}} [Q_t^j(s', a^j, \bar{a}^j)], \quad (5.12)$$

As shown in Eqs.(5.11) and (5.12), with the mean-field approximation, the MARL problem is converted into that of solving for the central agent j 's best response π_t^j w.r.t. the mean action \bar{a}^j of all j 's neighbours, which represents the action distribution of all neighbouring agents of the central agent j .

We introduce an iterative procedure in computing the best response π_t^j of each agent j . In the stage game $\{\mathbf{Q}_t\}$, the mean action \bar{a}^j of all j 's neighbours is first calculated by averaging the actions a^k taken by j 's N^j neighbours from the policies π_t^k parametrised by their previous mean actions \bar{a}^k_- .

$$\bar{a}^j = \frac{1}{N^j} \sum_k a^k, \quad a^k \sim \pi_t^k(\cdot | s, \bar{a}^k_-), \quad (5.13)$$

With each \bar{a}^j calculated as in Eq. (5.13), the policy π_t^j changes consequently due to the dependence on the current \bar{a}^j . The new Boltzmann policy is then determined for each j that

$$\pi_t^j(a^j | s, \bar{a}^j) = \frac{\exp(\beta Q_t^j(s, a^j, \bar{a}^j))}{\sum_{a^{j'} \in \mathcal{A}^j} \exp(\beta Q_t^j(s, a^{j'}, \bar{a}^j))}. \quad (5.14)$$

By iterating Eqs. (5.13) and (5.14), the mean actions \bar{a}^j and the corresponding policies π_t^j for all agents improves alternatively. In spite of lacking an intuitive impression of being stationary, in the following subsections, we will show that the mean action \bar{a}^j will be equilibrated at a unique point after several iterations. Hence, the policy π_t^j converges.

To distinguish from the Nash value function $v^{\text{Nash}}(s)$ in Eq. (5.4), we denote the mean-field value function in Eq. (5.12) as $\mathbf{v}^{\text{MF}}(s) \triangleq [v^1(s), \dots, v^N(s)]$. With \mathbf{v}^{MF} assembled, we now define the mean-field operator \mathcal{H}^{MF} in the form of

$$\mathcal{H}^{\text{MF}} \mathbf{Q}(s, \mathbf{a}) = \mathbb{E}_{s' \sim p} [\mathbf{r}(s, \mathbf{a}) + \gamma \mathbf{v}^{\text{MF}}(s')]. \quad (5.15)$$

In fact, we can prove that \mathcal{H}^{MF} forms a contraction mapping; that is, one updates \mathbf{Q} by iteratively applying the mean-field operator \mathcal{H}^{MF} , the mean-field Q -function will eventually converge to the Nash Q -value under certain assumptions.

5.4.2 Mean-Field Q-Learning and Actor-Critic

Algorithm 6 Mean-Field Q -learning (MF- Q)

```

1: Initialise  $Q_{\phi^j}$ ,  $Q_{\phi_-^j}$ , and  $\bar{a}^j$  for all  $j \in \{1, \dots, N\}$ 
2: while training not finished do
3:   for  $m = 1, \dots, M$  do
4:     For each agent  $j$ , sample action  $a^j$  from  $Q_{\phi^j}$  by Eq. (5.14), with the
       current mean action  $\bar{a}^j$  and the exploration rate  $\beta$ 
5:     For each agent  $j$ , compute the new mean action  $\bar{a}^j$  by Eq. (5.13)
6:   end for
7:   Take the joint action  $\mathbf{a} = [a^1, \dots, a^N]$  and observe the reward  $\mathbf{r} = [r^1, \dots, r^N]$ 
       and the next state  $s'$ 
8:   Store  $\langle s, \mathbf{a}, \mathbf{r}, s', \bar{\mathbf{a}} \rangle$  in replay buffer  $\mathcal{D}$ , where  $\bar{\mathbf{a}} = [\bar{a}^1, \dots, \bar{a}^N]$ 
9:   for  $j = 1$  to  $N$  do
10:    Sample a minibatch of  $K$  experiences  $\langle s, \mathbf{a}, \mathbf{r}, s', \bar{\mathbf{a}} \rangle$  from  $\mathcal{D}$ 
11:    Sample action  $a_-^j$  from  $Q_{\phi_-^j}$  with  $\bar{a}_-^j \leftarrow \bar{a}^j$  for each experience.
12:    Set  $y^j = r^j + \gamma v_{\phi_-^j}^{\text{MF}}(s')$  by Eq. (5.12) for each experience.
13:    Update the  $Q$ -network by minimising the loss

$$\mathcal{L}(\phi^j) = \frac{1}{K} \sum (y^j - Q_{\phi^j}(s^j, a^j, \bar{a}^j))^2$$

14:  end for
15:  Update the target network for each agent  $j$  with learning rate  $\tau$ :

$$\phi_-^j \leftarrow \tau \phi^j + (1 - \tau) \phi_-^j$$

16: end while

```

We can implement the mean-field Q -function in Eq. (5.9) by universal function approximators such as neural networks, where the Q -function is parameterised with the weights ϕ . The update rule in Eq. (5.11) can be reformulated as weights adjustment. For off-policy learning, we exploit either standard Q -learning ([Watkins and Dayan, 1992](#)) for discrete action spaces or DPG ([Silver et al., 2014](#)) for continuous action spaces. Here we focus on the former, which we call MF- Q .

In MF- Q , agent j is trained by minimising the loss function

$$\mathcal{L}(\phi^j) = (y^j - Q_{\phi^j}(s, a^j, \bar{a}^j))^2,$$

where $y^j = r^j + \gamma v_{\phi_-^j}^{\text{MF}}(s')$ is the target mean-field value calculated with the weights

Algorithm 7 Mean-Field Actor-Critic (MF-AC)

1: Initialise Q_{ϕ^j} , $Q_{\phi_-^j}$, π_{θ^j} , $\pi_{\theta_-^j}$, and \bar{a}^j for all $j \in \{1, \dots, N\}$
2: **while** training not finished **do**
3: For each agent j , sample action $a^j = \pi_{\theta^j}(s)$; compute the new mean action
 $\bar{a} = [\bar{a}^1, \dots, \bar{a}^N]$
4: Take the joint action $\mathbf{a} = [a^1, \dots, a^N]$ and observe the reward $\mathbf{r} = [r^1, \dots, r^N]$
and the next state s'
5: Store $\langle s, \mathbf{a}, \mathbf{r}, s', \bar{a} \rangle$ in replay buffer \mathcal{D}
6: **for** $j = 1$ to N **do**
7: Sample a minibatch of K experiences $\langle s, \mathbf{a}, \mathbf{r}, s', \bar{a} \rangle$ from \mathcal{D}
8: Set $y^j = r^j + \gamma v_{\phi_-^j}^{\text{MF}}(s')$ by Eq. (5.12)
9: Update the critic by minimising the loss

$$\mathcal{L}(\phi^j) = \frac{1}{K} \sum (y^j - Q_{\phi^j}(s, a^j, \bar{a}^j))^2$$

10: Update the actor using the sampled policy gradient:

$$\nabla_{\theta^j} \mathcal{J}(\theta^j) \approx \frac{1}{K} \sum \nabla_{\theta^j} \log \pi_{\theta^j}(s') Q_{\phi_-^j}(s', a_-^j, \bar{a}_-^j) \Big|_{a_-^j = \pi_{\theta_-^j}(s')}$$

11: **end for**
12: Update the target network for each agent j with learning rates τ_ϕ and τ_θ :

$$\begin{aligned} \phi_-^j &\leftarrow \tau_\phi \phi^j + (1 - \tau_\phi) \phi_-^j \\ \theta_-^j &\leftarrow \tau_\theta \theta^j + (1 - \tau_\theta) \theta_-^j \end{aligned}$$

13: **end while**

ϕ_-^j . Differentiating $\mathcal{L}(\phi^j)$ gives us the following equation:

$$\nabla_{\phi^j} \mathcal{L}(\phi^j) = (y^j - Q_{\phi^j}(s, a^j, \bar{a}^j)) \nabla_{\phi^j} Q_{\phi^j}(s, a^j, \bar{a}^j). \quad (5.16)$$

Instead of setting up Boltzmann policy using the Q -function as in MF- Q , we can explicitly model the policy by neural networks with the weights θ , which leads to the on-policy actor-critic method (Konda and Tsitsiklis, 2000b) that we call MF-AC. The policy network π_{θ^j} , *i.e.* the actor, of MF-AC is trained by the sampled policy gradient:

$$\nabla_{\theta^j} \mathcal{J}(\theta^j) \approx \nabla_{\theta^j} \log \pi_{\theta^j}(s) Q_{\phi^j}(s, a^j, \bar{a}^j) \Big|_{a=\pi_{\theta^j}(s)}.$$

The critic of MF-AC follows the same setting for MF- Q with Eq. (5.16). During the training of MF-AC, one needs to alternatively update ϕ and θ until convergence. We illustrate the MF- Q iterations in Figure 5.2, and present the pseudocode for both MF- Q and MF-AC as follows. We published the code at <https://github.com/mlii/mfrl>.

5.5 Convergence Analysis of MF-Q

We now prove the convergence of $\mathbf{Q}_t \triangleq [Q_t^1, \dots, Q_t^N]$ to the Nash Q -value $\mathbf{Q}_* = [Q_*^1, \dots, Q_*^N]$ as the iterations of MF- Q are applied.

5.5.1 MF-Q with Tabular Form Representation

The first proof is presented by showing that the mean-field operator \mathcal{H}^{MF} in Eq. (5.15) forms a contraction mapping with the fixed point at \mathbf{Q}_* under the main assumptions. We start by introducing the assumptions:

Assumption 1. *Each action-value pair is visited infinitely often, and the reward is bounded by some constant K .*

Assumption 2. *Agent's policy is Greedy in the Limit with Infinite Exploration (GLIE). In the case with the Boltzmann policy, the policy becomes greedy w.r.t. the Q -function in the limit as the temperature decays asymptotically to zero.*

Assumption 3. *For each stage game $[Q_t^1(s), \dots, Q_t^N(s)]$ at time t and in state s in training, for all $t, s, j \in \{1, \dots, N\}$, the Nash equilibrium $\boldsymbol{\pi}_* = [\pi_*^1, \dots, \pi_*^N]$ is recognised either as 1) the global optimum or 2) a saddle point expressed as:*

1. $\mathbb{E}_{\boldsymbol{\pi}_*}[Q_t^j(s)] \geq \mathbb{E}_{\boldsymbol{\pi}}[Q_t^j(s)], \forall \boldsymbol{\pi} \in \Omega(\prod_k \mathcal{A}^k);$
2. $\mathbb{E}_{\boldsymbol{\pi}_*}[Q_t^j(s)] \geq \mathbb{E}_{\boldsymbol{\pi}^j} \mathbb{E}_{\boldsymbol{\pi}_{*}^{-j}}[Q_t^j(s)], \forall \boldsymbol{\pi}^j \in \Omega(\mathcal{A}^j)$ and
 $\mathbb{E}_{\boldsymbol{\pi}_*}[Q_t^j(s)] \leq \mathbb{E}_{\boldsymbol{\pi}^j} \mathbb{E}_{\boldsymbol{\pi}_{*}^{-j}}[Q_t^j(s)], \forall \boldsymbol{\pi}^{-j} \in \Omega(\prod_{k \neq j} \mathcal{A}^k).$

Note that Assumption 3 imposes a strong constraint on every single stage game encountered in training. In practice, however, we find this constraint appears not to be a necessary condition for the learning algorithm to converge. This is in line with the empirical findings in [Hu and Wellman \(2003\)](#).

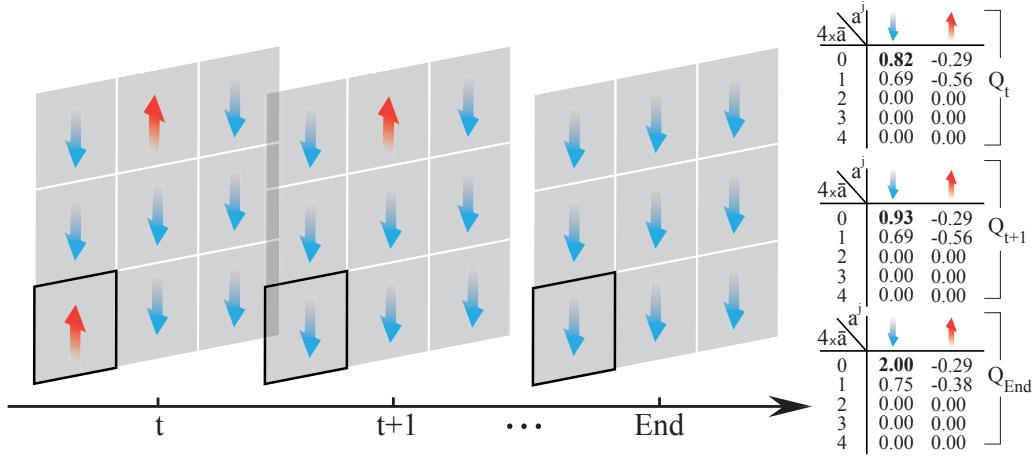


Figure 5.2: MF-Q iterations on a 3×3 stateless toy example. The goal is to coordinate the agents to an agreed direction. Each agent has two choices of actions: *up* \uparrow or *down* \downarrow . The reward of each agent's staying in the same direction as its $[0, 1, 2, 3, 4]$ neighbours are $[-2.0, -1.0, 0.0, 1.0, 2.0]$, respectively. The neighbours are specified by the four directions on the grid with cyclic structure on all directions, e.g. the first row and the third row are adjacent. The reward for the highlighted agent j on the bottom left at time $t + 1$ is 2.0, as all neighbouring agents stay down in the same time. We listed the Q-tables for agent j at three time steps where \bar{a}^j is the percentage of neighbouring ups. Following Eq. (5.11), we have $Q_{t+1}^j(\uparrow, \bar{a}^j = 0) = Q_t^j(\uparrow, \bar{a}^j = 0) + \alpha[r^j - Q_t^j(\uparrow, \bar{a}^j = 0)] = 0.82 + 0.1 \times (2.0 - 0.82) = 0.93$. The rightmost plot shows the convergent scenario where the Q -value of staying down is 2.0, which is the largest reward in the environment.

Our proof is also built upon the two lemmas as follows:

Lemma 3. Under Assumption 3, the Nash operator $\mathcal{H}^{\text{Nash}}$ in Eq. (5.4) forms a contraction mapping on the complete metric space from \mathcal{Q} to \mathcal{Q} with the fixed point being the Nash Q -value of the entire game, i.e. $\mathcal{H}_t^{\text{Nash}} \mathbf{Q}_* = \mathbf{Q}_*$.

Proof. See Theorem 17 in [Hu and Wellman \(2003\)](#). \square

Lemma 4. The random process $\{\Delta_t\}$ defined in \mathbb{R} as

$$\Delta_{t+1}(x) = (1 - \alpha_t(x))\Delta_t(x) + \alpha_t(x)F_t(x) \quad (5.17)$$

converges to zero with probability 1 (w.p.1) when

1. $0 \leq \alpha_t(x) \leq 1$, $\sum_t \alpha_t(x) = \infty$, $\sum_t \alpha_t^2(x) < \infty$;
2. $x \in \mathcal{X}$, the set of possible states, and $|\mathcal{X}| < \infty$;

3. $\|\mathbb{E}[F_t(x)|\mathcal{F}_t]\|_W \leq \gamma \|\Delta_t\|_W + c_t$, where $\gamma \in [0, 1)$ and c_t converges to zero w.p.1;
4. $\text{var}[F_t(x)|\mathcal{F}_t] \leq K(1 + \|\Delta_t\|_W^2)$ with constant $K > 0$.

Here \mathcal{F}_t denotes the filtration of an increasing sequence of σ -fields including the history of processes; $\alpha_t, \Delta_t, F_t \in \mathcal{F}_t$ and $\|\cdot\|_W$ is a weighted maximum norm (Bertsekas, 2012).

Proof. See Theorem 1 in Jaakkola et al. (1994) and Corollary 5 Szepesvári and Littman (1999) for detailed derivation. We include it here to stay self-contained.

□

By subtracting $\mathbf{Q}_*(s, \mathbf{a})$ on both sides of Eq. (5.11), we present the relation from the comparison with Eq. (5.17) such that

$$\begin{aligned} \Delta_t(x) &= \mathbf{Q}_t(s, \mathbf{a}) - \mathbf{Q}_*(s, \mathbf{a}), \\ \mathbf{F}_t(x) &= \mathbf{r}_t + \gamma \mathbf{v}_t^{\text{MF}}(s_{t+1}) - \mathbf{Q}_*(s_t, \mathbf{a}_t), \end{aligned} \quad (5.18)$$

where $x \triangleq (s_t, \mathbf{a}_t)$ denotes the visited state-action pair at time t . In Eq. (5.17), $\alpha(t)$ is interpreted as the learning rate with $\alpha_t(s', \mathbf{a}') = 0$ for any $(s', \mathbf{a}') \neq (s_t, \mathbf{a}_t)$; this is because each agent only updates the Q -function with the state s_t and actions \mathbf{a}_t visited at time t . Lemma 4 suggests $\Delta_t(x)$'s convergence to zero, which means, if it holds, the sequence of Q 's will asymptotically tend to the Nash Q -value \mathbf{Q}_* .

One last piece to establish the main theorem is the below:

Proposition 3. *Let the metric space be \mathbb{R}^N and the metric be $d(\mathbf{a}, \mathbf{b}) = \sum_j |a^j - b^j|$, for $\mathbf{a} = [a^j]_1^N, \mathbf{b} = [b^j]_1^N$. If the Q -function is K -Lipschitz continuous w.r.t. a^j , then the operator $\mathcal{B}(a^j) \triangleq \pi^j(a^j|s, \bar{a}^j)$ in Eq. (5.14) forms a contraction mapping under sufficiently low temperature β .*

Proof. Following the contraction mapping theorem (Kreyszig, 1978), in order to be a contraction, the operator has to satisfy:

$$d(\mathcal{B}(\mathbf{a}), \mathcal{B}(\mathbf{b})) \leq \alpha d(\mathbf{a}, \mathbf{b}), \quad \forall \mathbf{a}, \mathbf{b}$$

where $0 \leq \alpha < 1$ and $\mathcal{B}(\mathbf{a}) \triangleq [\mathcal{B}(a^1), \dots, \mathcal{B}(a^N)]$.

Here we start from the binomial case and then adapt to the multinomial case in general. We first rewrite $\mathcal{B}(a^j)$ as

$$\begin{aligned}\mathcal{B}(a^j) = \pi^j(a^j|s, \bar{a}^j) &= \frac{\exp(-\beta Q_t^j(s, a^j, \bar{a}^j))}{\exp(-\beta Q_t^j(s, a^j, \bar{a}^j)) + \exp(-\beta Q_t^j(s, \neg a^j, \bar{a}^j))} \\ &= \frac{1}{1 + \exp(-\beta \cdot \Delta Q(s, a^j, \bar{a}))},\end{aligned}\quad (5.19)$$

where $\Delta Q(s, a^j, \bar{a}) = Q(s, a^{\neg j}, \bar{a}) - Q(s, a^j, \bar{a})$.

Then we have

$$\begin{aligned}|\mathcal{B}(a^j) - \mathcal{B}(b^j)| &= \left| \frac{1}{1 + e^{-\beta \cdot \Delta Q(s, a^j, \bar{a})}} - \frac{1}{1 + e^{-\beta \cdot \Delta Q(s, b^j, \bar{a})}} \right| \\ &= \left| \frac{\beta e^{-\beta \Delta Q_0}}{(1 + e^{-\beta \Delta Q_0})^2} \right| |\Delta Q(s, a^j, \bar{a}) - \Delta Q(s, b^j, \bar{a})| \\ &\leq \frac{1}{4T} \cdot \left| Q(s, a^{\neg j}, \bar{a}) - Q(s, b^{\neg j}, \bar{a}) + Q(s, b^j, \bar{a}) - Q(s, a^j, \bar{a}) \right| \\ &\leq \frac{1}{4T} \cdot (K \cdot |1 - a^j - (1 - b^j)| + K \cdot |a^j - b^j|) \\ &\leq \frac{1}{4T} \cdot 2K \cdot \sum_j |a^j - b^j|.\end{aligned}\quad (5.20)$$

In the second equation, we apply the mean value theorem in calculus: $\exists x_0 \in [x_1, x_2]$, s.t., $f(x_1) - f(x_2) = f'(x_0)(x_1 - x_2)$. In the third equation we use the maximum value for $e^{-\beta \Delta Q_0}/(1 + e^{-\beta \Delta Q_0})^2 = 1/4$ when $Q_0 = 0$. In the last equation we apply the Lipschitz constraint in the assumption where constant $K \geq 0$. Finally, we have:

$$\begin{aligned}d(\mathcal{B}(\mathbf{a}), \mathcal{B}(\mathbf{b})) &\leq \frac{1}{4T} \cdot 2K \cdot \sum_j |a^j - b^j| \\ &= \frac{K}{2T} d(\mathbf{a}, \mathbf{b})\end{aligned}\quad (5.21)$$

In order for the contraction to hold, $T > \frac{K}{2}$. In other words, when the action space is binary for each agent, and the temperature is sufficiently large, the mean-field procedure converges.

This proposition can be easily extended to the multinomial case by replacing

binary variable a^j by a multi-dimensional binary indicator vector \mathbf{a}^j , on each dimension, the rest of the derivations would remain essentially the same. \square

Theorem 3. *In a finite-state stochastic game, the \mathbf{Q}_t values computed by the update rule of MF-Q in Eq. (5.11) converge to the Nash Q-value $\mathbf{Q}_* = [Q_*^1, \dots, Q_*^N]$, if Assumptions 1, 2 & 3, and Lemma 4's first and second conditions are met.*

Proof. Let \mathcal{F}_t denote the σ -field generated by all random variables in the history of the stochastic game up to time t : $(s_t, \alpha_t, \mathbf{a}_t, r_{t-1}, \dots, s_1, \alpha_1, \mathbf{a}_1, \mathbf{Q}_0)$. Note that \mathbf{Q}_t is a random variable derived from the historical trajectory up to time t . Given the fact that all \mathbf{Q}_τ with $\tau < t$ are \mathcal{F}_t -measurable, both \mathbf{A}_t and \mathbf{F}_{t-1} are therefore also \mathcal{F}_t -measurable, which satisfies the measurability condition of Lemma 4.

To apply Lemma 4, we need to show that the mean-field operator \mathcal{H}^{MF} meets Lemma 4's third and fourth conditions. For Lemma 4's third condition, we begin with Eq. (5.18) that

$$\begin{aligned} \mathbf{F}_t(s_t, \mathbf{a}_t) &= \mathbf{r}_t + \gamma \mathbf{v}_t^{\text{MF}}(s_{t+1}) - \mathbf{Q}_*(s_t, \mathbf{a}_t) \\ &= \mathbf{r}_t + \gamma \mathbf{v}_t^{\text{Nash}}(s_{t+1}) - \mathbf{Q}_*(s_t, \mathbf{a}_t) \\ &\quad + \gamma [\mathbf{v}_t^{\text{MF}}(s_{t+1}) - \mathbf{v}_t^{\text{Nash}}(s_{t+1})] \\ &= [\mathbf{r}_t + \gamma \mathbf{v}_t^{\text{Nash}}(s_{t+1}) - \mathbf{Q}_*(s_t, \mathbf{a}_t)] + \mathbf{C}_t(s_t, \mathbf{a}_t) \\ &= \mathbf{F}_t^{\text{Nash}}(s_t, \mathbf{a}_t) + \mathbf{C}_t(s_t, \mathbf{a}_t). \end{aligned} \quad (5.22)$$

Note the fact that $\mathbf{F}_t^{\text{Nash}}$ in Eq. (5.22) is essentially the \mathbf{F}_t in Lemma 4 in proving the convergence of the Nash Q-learning algorithm. From Lemma 3, it is straightforward to show that $\mathbf{F}_t^{\text{Nash}}$ forms a contraction mapping with the norm $\|\cdot\|_\infty$ being the maximum norm on \mathbf{a} . We thus have for all t that

$$\|\mathbb{E}[\mathbf{F}_t^{\text{Nash}}(s_t, \mathbf{a}_t)|\mathcal{F}_t]\|_\infty \leq \gamma \|\mathbf{Q}_t - \mathbf{Q}_*\|_\infty = \gamma \|\mathbf{A}_t\|_\infty.$$

In meeting the third condition, we obtain from Eq. (5.22) that

$$\begin{aligned} \|\mathbb{E}[\mathbf{F}_t(s_t, \mathbf{a}_t)|\mathcal{F}_t]\|_\infty &\leq \|\mathbf{F}_t^{\text{Nash}}(s_t, \mathbf{a}_t)|\mathcal{F}_t\|_\infty + \|\mathbf{C}_t(s_t, \mathbf{a}_t)|\mathcal{F}_t\|_\infty \\ &\leq \gamma \|\mathbf{A}_t\|_\infty + \|\mathbf{C}_t(s_t, \mathbf{a}_t)|\mathcal{F}_t\|_\infty. \end{aligned} \quad (5.23)$$

We are left to prove that $c_t = \|\mathbf{C}_t(s_t, \mathbf{a}_t)|\mathcal{F}_t\|$ converges to zero *w.p.1.* With Assumption 3, for each stage game, all the globally optimal equilibrium(s) share the same Nash value, so does the saddle-point equilibrium(s). Each of the two following results is essentially associated with one of the two mutually exclusive scenarios in Assumption 3:

1. For globally optimal equilibria, all players obtain the joint maximum values that are unique and identical for all equilibria according to the definition;
2. Suppose that the stage game $\{\mathbf{Q}_t\}$ has two saddle-point equilibria, $\boldsymbol{\pi}$ and $\boldsymbol{\rho}$. It holds for agent j that

$$\begin{aligned} \mathbb{E}_{\boldsymbol{\pi}^j} \mathbb{E}_{\boldsymbol{\pi}^{-j}} [\mathbf{Q}_t^j(s)] &\geq \mathbb{E}_{\boldsymbol{\rho}^j} \mathbb{E}_{\boldsymbol{\pi}^{-j}} [\mathbf{Q}_t^j(s)], \\ \mathbb{E}_{\boldsymbol{\rho}^j} \mathbb{E}_{\boldsymbol{\rho}^{-j}} [\mathbf{Q}_t^j(s)] &\leq \mathbb{E}_{\boldsymbol{\rho}^j} \mathbb{E}_{\boldsymbol{\pi}^{-j}} [\mathbf{Q}_t^j(s)]. \end{aligned}$$

By combining the above inequalities, we obtain

$$\mathbb{E}_{\boldsymbol{\pi}^j} \mathbb{E}_{\boldsymbol{\pi}^{-j}} [\mathbf{Q}_t^j(s)] \geq \mathbb{E}_{\boldsymbol{\rho}^j} \mathbb{E}_{\boldsymbol{\rho}^{-j}} [\mathbf{Q}_t^j(s)].$$

By the definition of saddle points, the above inequality still holds by reversing the order of $\boldsymbol{\pi}$ and $\boldsymbol{\rho}$; hence, the equilibria for agent i at both saddle points are the same such that $\mathbb{E}_{\boldsymbol{\pi}^j} \mathbb{E}_{\boldsymbol{\pi}^{-j}} [\mathbf{Q}_t^j(s)] = \mathbb{E}_{\boldsymbol{\rho}^j} \mathbb{E}_{\boldsymbol{\rho}^{-j}} [\mathbf{Q}_t^j(s)]$.

Given Proposition 3 that the policy based on the mean-field Q -function forms a contraction mapping, and that all optimal/saddle points share the same Nash value in each stage game, with the homogeneity of agents, \mathbf{v}^{MF} will asymptotically converge to \mathbf{v}^{Nash} , the third condition is thus satisfied.

For the fourth condition, we exploit the conclusion that is proved above that \mathcal{H}^{MF} forms a contraction mapping, *i.e.* $\mathcal{H}^{\text{MF}}\mathbf{Q}_* = \mathbf{Q}_*$, and it follows that

$$\begin{aligned}\mathbf{var}[\mathbf{F}_t(s_t, \mathbf{a}_t) | \mathcal{F}_t] &= \mathbb{E}[(\mathbf{r}_t + \gamma \mathbf{v}_t^{\text{MF}}(s_{t+1}) - \mathbf{Q}_*(s_t, \mathbf{a}_t))^2] \\ &= \mathbb{E}[(\mathbf{r}_t + \gamma \mathbf{v}_t^{\text{MF}}(s_{t+1}) - \mathcal{H}^{\text{MF}}(\mathbf{Q}_*))^2] \\ &= \mathbf{var}[\mathbf{r}_t + \gamma \mathbf{v}_t^{\text{MF}}(s_{t+1}) | \mathcal{F}_t] \\ &\leq K(1 + \|\mathbf{A}_t\|_W^2).\end{aligned}\tag{5.24}$$

In the last step of Eq. (5.24), we employ Assumption 1 that the reward \mathbf{r}_t is always bounded by some constant. Finally, with all conditions met, it follows Lemma 4 that \mathbf{A}_t converges to zero *w.p.1*, *i.e.* \mathbf{Q}_t converges to \mathbf{Q}_* *w.p.1*. \square

5.5.2 MF-Q with Linear Function Approximation

Previous convergence results in Theorem 3 have shown that the Mean-Field Q-learning algorithm will converge when the Q function is in tabular cases. We now move onto the proof that the MF-Q algorithm will converge when the Q function is represented by some function approximations.

An N -agent (or, N -player) stochastic game Γ is formalised by the tuple $\Gamma \triangleq (\mathcal{S}, \mathcal{A}^1, \dots, \mathcal{A}^N, r^1, \dots, r^N, p, \gamma)$. The state-space \mathcal{S} is finite. Let (\mathcal{S}, p_π) be the Markov chain induced by the joint policy π , and we assume it to be uniformly ergodic.

Let $\mathcal{Q} = \{\mathbf{Q}_\theta\}$ be a family of real-valued functions defined on $\mathcal{S} \times \mathcal{A} \times \bar{\mathcal{A}}$, where $\bar{\mathcal{A}}$ is the action space for the mean actions computed from the neighbours. Assuming that the function class is linearly parameterised, for each agent j , Q can be expressed as the linear span of a fixed set of P linearly independent functions $\omega_p^j : \mathcal{S} \times \mathcal{A} \times \bar{\mathcal{A}} \rightarrow \mathbb{R}$. Given the parameter vector $\phi^j \in \mathbb{R}^P$, for each agent, the function Q_{ϕ^j} is thus defined as

$$Q_{\phi^j}(s, a^j, \bar{a}^j) = \sum_{p=1}^P \omega_p^j(s, a^j, \bar{a}^j) \phi^j(p) = \boldsymbol{\omega}^j(s, a^j, \bar{a}^j)^\top \phi^j\tag{5.25}$$

In the functional approximation setting, we can apply the update rules:

$$\begin{aligned}\phi_{t+1}^j &= \phi_t^j + \alpha_t \Delta_t \nabla_{\phi^j} Q_{\phi^j}(s, a^j, \bar{a}^j) \\ &= \phi_t^j + \alpha_t \Delta_t \omega^j(s, a_t^j, \bar{a}_t^j)\end{aligned}\quad (5.26)$$

In the above, Δ_t is the temporal difference at time t .

$$\Delta_t = r^j + \gamma v_{\phi^j}^{\text{MF}}(s') - Q_{\phi^j}(s, a^j, \bar{a}^j) \quad (5.27)$$

$$= r^j + \gamma \mathbb{E}_{a \sim \pi^{\text{MF}}} [Q_{\phi^j}(s', a^j, \bar{a}^j)] - Q_{\phi^j}(s, a^j, \bar{a}^j). \quad (5.28)$$

And the goal is to derive the parameter vector $\phi = \{\phi^j\}$ such that $\omega^\top \phi$ approximates the (local) Nash Q-values. At each time step, the learning policy π_{ϕ_t} is the Boltzmann policy with respect to $\omega^\top \phi$. Give **Proposition 1**, we know that the policy π_{ϕ_t} is $\frac{K}{2T}$ Lipschitz continuous with respect to ϕ_t .

Similar to the framework used in the convergence proof of Q-learning with function approximation (Melo et al., 2008), we establish the convergence of Eq. (5.26) by adopting an ordinary differentiable equation (ODE) with a globally asymptotically stable equilibrium point where the trajectories closely follow.

Theorem 4. *Given the MDP Γ , π_{ϕ_t} , $\{\omega_p, p = 1, \dots, P\}$, and the learning policy π_{ϕ_t} that is $\frac{K}{2T}$ Lipschitz continuous with respect to ϕ_t , if the Assumptions 1, 2 & 3, and Lemma 4's first and second conditions are met, then there exists C_0 such that the algorithm in Eq. (5.26) converges w.p.1 if $\frac{K}{2T} < C_0$.*

Proof. We first re-write the Eq. (5.26) as on ODE:

$$\begin{aligned}\frac{d\phi}{dt} &= \mathbb{E}_{\phi} \left[\omega_s^\top \left(\mathbf{r}(s, a, s') + \gamma \omega_{s'}^\top \phi - \omega_s^\top \phi \right) \right] \\ &= \mathbb{E}_{\phi} \left[\omega_s^\top (\gamma \omega_{s'}^\top - \omega_s^\top) \right] \phi + \mathbb{E}_{\phi} \left[\omega_s^\top (\mathbf{r}(s, a, s')) \right] \\ &= A_\phi \phi + b_\phi\end{aligned}\quad (5.29)$$

Notice that we use a vector for considering the updating rule for the Q function of each agent. We can easily know that necessity condition of the equilibrium is that it

must follow $\boldsymbol{\phi}^* = \mathbf{A}_{\phi^*}^{-1} \mathbf{b}_{\phi^*}$. The existence of the such equilibrium has been restricted in the scenario that meets Assumption 3. In the proof of Theorem 1 we have already pointed out that under the Assumption 3, the existing equilibrium, either in the form of global equilibrium or in the form of saddle-point equilibrium, is unique.

Let $\tilde{\boldsymbol{\phi}} = \boldsymbol{\phi}_t - \boldsymbol{\phi}^*$, we have:

$$\begin{aligned}
\frac{d}{dt} \|\tilde{\boldsymbol{\phi}}\|_2 &= 2\boldsymbol{\phi} \cdot \frac{d\boldsymbol{\phi}}{dt} - 2\boldsymbol{\phi}^* \cdot \frac{d\boldsymbol{\phi}}{dt} \\
&= 2\tilde{\boldsymbol{\phi}}^\top (\mathbf{A}_\phi \boldsymbol{\phi} + \mathbf{b}_\phi - \mathbf{A}_{\phi^*} \boldsymbol{\phi}^* - \mathbf{b}_{\phi^*}) \\
&= 2\tilde{\boldsymbol{\phi}}^\top (\mathbf{A}_{\phi^*} \boldsymbol{\phi} - \mathbf{A}_{\phi^*} \boldsymbol{\phi} + \mathbf{A}_\phi \boldsymbol{\phi} + \mathbf{b}_\phi - \mathbf{A}_{\phi^*} \boldsymbol{\phi}^* - \mathbf{b}_{\phi^*}) \\
&= 2\tilde{\boldsymbol{\phi}}^\top \mathbf{A}_{\phi^*} \tilde{\boldsymbol{\phi}} + 2\tilde{\boldsymbol{\phi}}^\top (\mathbf{A}_\phi - \mathbf{A}_{\phi^*}) \boldsymbol{\phi} + 2\tilde{\boldsymbol{\phi}}^\top (\mathbf{b}_\phi - \mathbf{b}_{\phi^*}) \\
&\leq 2\tilde{\boldsymbol{\phi}}^\top \left(\mathbf{A}_{\phi^*} + \sup_{\boldsymbol{\phi}} \|\mathbf{A}_\phi - \mathbf{A}_{\phi^*}\|_2 + \sup_{\boldsymbol{\phi}} \frac{\|\mathbf{b}_\phi - \mathbf{b}_{\phi^*}\|_2}{\|\boldsymbol{\phi} - \boldsymbol{\phi}^*\|_2} \right) \tilde{\boldsymbol{\phi}}
\end{aligned} \tag{5.30}$$

As we know that the policy π_{ϕ_t} is Lipschitz w.r.t ϕ_t , this implies that \mathbf{A}_ϕ and \mathbf{v}_ϕ are also Lipschitz continuous w.r.t to ϕ . In other words, if $\frac{K}{2T} \leq C_0$ is sufficiently small and close to zero, then the norm term of $\left(\sup_{\boldsymbol{\phi}} \|\mathbf{A}_\phi - \mathbf{A}_{\phi^*}\|_2 + \sup_{\boldsymbol{\phi}} \frac{\|\mathbf{b}_\phi - \mathbf{b}_{\phi^*}\|_2}{\|\boldsymbol{\phi} - \boldsymbol{\phi}^*\|_2} \right)$ goes to zero. Considering near the equilibrium point $\boldsymbol{\phi}^*$, \mathbf{A}_{ϕ^*} is a negative definite matrix, the Eq. (5.30) tends to be negative definite as well, so the ODE in Eq. (5.29) is globally asymptotically stable and the conclusion of the theorem follows. \square

5.5.3 MF-Q is Rational

Apart from being convergent to the Nash Q -value, MF-Q is also **rational** (Bowling and Veloso, 2001b, 2002). In line with (Bowling and Veloso, 2001b, 2002), we argue that to better evaluate a multi-agent learning algorithm, on top of the convergence guarantee, discussion on the property of Rationality is also needed.

Definition 11 (Definition of Rationality). (*also see (Bowling and Veloso, 2001b, 2002)*) In an N -agent stochastic game defined in this paper, given all agents converge to stationary policies, if the learning algorithm converges to a policy that is a **best response** to the other agents' policies, then the algorithm is Rational.

Our mean-field Q -learning is rational in that Eq. (5.5) converts many agents

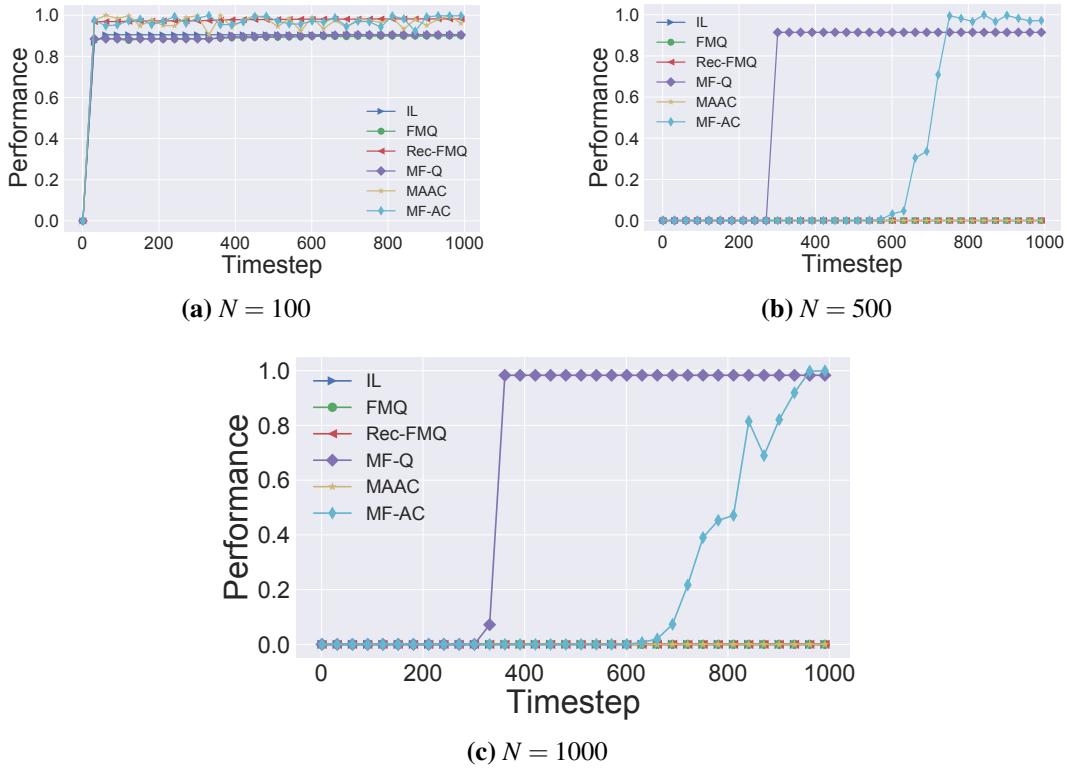


Figure 5.3: Learning with N agents in the GS environment with $\mu = 400$ and $\sigma = 200$. With the increasing number of agents, MF-Q methods demonstrate remarkable performance improvement.

interactions into two-body interactions between a single agent and the distribution of other agents' actions. When all agents follow stationary policies, their policy distribution would be stationary too. As such the two-body stochastic game becomes an MDP, and the agent would choose a policy (Assumption 2) which is the best response to the distribution of other stationary policies. As agents are symmetric in our case, they all show the best response to the distributions, and are thus rational.

5.6 Experiments and Results

We analyse and evaluate our algorithms in three different scenarios, including two stage games: the Gaussian Squeeze and the Ising Model, and the *mixed cooperative-competitive* battle game.

5.6.1 Gaussian Squeeze

Environment. In the Gaussian Squeeze (GS) task (HolmesParker et al., 2014), N homogeneous agents determine their individual action a^j to jointly optimise the

most appropriate summation $x = \sum_{j=1}^N a^j$. Each agent has ten action choices – integers 0 to 9. The system objective is defined as $G(x) = xe^{-\frac{(x-\mu)^2}{\sigma^2}}$, where μ and σ are the pre-defined mean and variance of the system. In the scenario of traffic congestion, each agent is one traffic controller trying to send a^j vehicles into the main road. Controllers are expected to coordinate with each other to make the full use of the main route while avoiding congestions. Each agent’s goal is to learn to allocate system resources efficiently, avoiding either over-use or under-use. The GS problem here sits ideally as an ablation study on the impact of multi-agent exploratory noises on learning (Colby et al., 2015).

Model Settings. We implement MF- Q and MF-AC following the framework of centralised training (shared critic) with decentralised execution (independent actor). We compare against four baseline models: (1) Independent Learner (IL), a traditional Q -Learning algorithm that does not consider the actions performed by other agents and treat them as part of the environment; (2) Frequency Maximum Q -value (FMQ) (Kapetanakis and Kudenko, 2002), a modified IL which increases the Q -values of actions that frequently produced good rewards in the past; (3) Recursive Frequency Maximum Q -value (Rec-FMQ) (Matignon et al., 2012), an improved version of FMQ that recursively computes the occurrence frequency to evaluate and then choose actions; (4) Multi-agent Actor-Critic (MAAC), a variant of MADDPG architecture for the discrete action space (see Eq. (4) in Lowe et al. (2017b)). All models use multi-layer neural networks as the function approximator.

Hyper-parameter Settings. IL, FMQ, Rec-FMQ and MF- Q all use a three-layer MLP to approximate the Q -value. All agents share the same Q -network for each experiment. The shared Q -network takes an agent embedding as input and computes the Q -value for each candidate action. For MF- Q , we also feed in the action approximation \bar{a} . We use Adam optimiser with a learning rate of 0.00001 and ϵ -greedy exploration unless otherwise specified. For FMQ, we set the exponential decay rate $s = 0.006$, start temperature $max_temp=1000$ and FMQ heuristic $c = 5$. For Rec-FMQ, we set the frequency learning rate $\alpha_f = 0.01$. MAAC and MF-AC use Adam optimiser with a learning rate of 0.001 and 0.0001 for Critics and

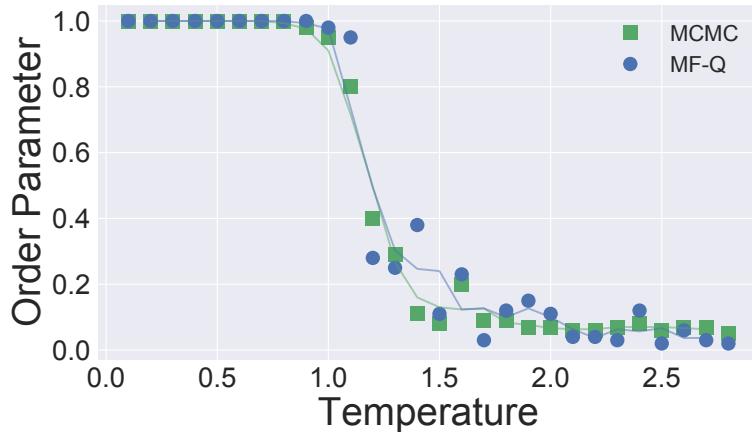


Figure 5.4: The *order parameter* at equilibrium v.s. temperature in the Ising model with 20×20 grid.

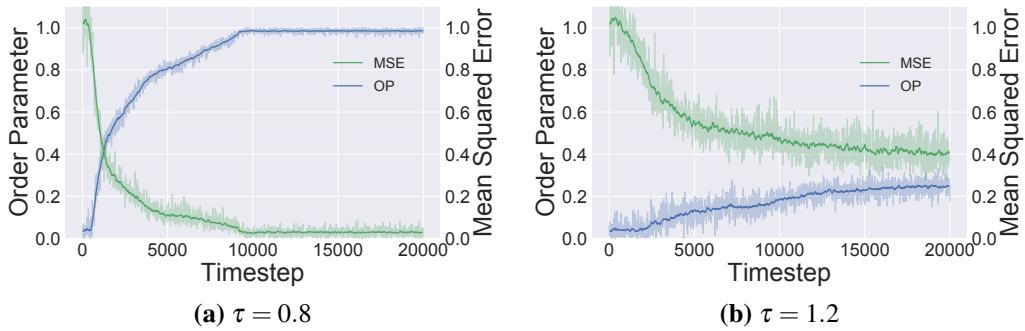


Figure 5.5: Training performance of MF-Q in the Ising model with 20×20 grid.

Actors respectively, and $\tau = 0.01$ for updating the target networks. We share the critic among all agents in each experiment and feed in an agent embedding as an extra input. Actors are kept separate. The discounted factor γ is set to be 0.95, and the mini-batch size is set to be 200. The size of the replay buffer is 10^6 , and we update the network parameters after every 500 samples added to the replay buffer. For all models, we use the performance of the joint-policy learned up to that point if learning and exploration were turned off (*i.e.*, take the greedy action w.r.t. the learned policy) to compare our method with the above baseline models.

Results. Figure. 5.3 illustrates the results for the GS environment of $\mu = 400$ and $\sigma = 200$ with three different numbers of agents ($N = 100, 500, 1000$) that stand for 3 levels of congestions. In the smallest GS setting of Figure 5.3a, all models show excellent performance. As the agent number increases, Figure 5.3b and 5.3c show MF-Q and MF-AC's capabilities of learning the optimal allocation effectively after a few iterations, whereas all four baselines fail to learn at all. We believe

this advantage is due to the awareness of other agents' actions under the mean-field framework; such mechanism keeps the interactions among agents manageable while reducing the noisy effect of the exploratory behaviours from the other agents. Between MF- Q and MF-AC, MF- Q converges faster; this is probably because MF- Q does not need to learn the policy explicitly. Both FMQ and Rec-FMQ fail to reach excellent performance, it might be because agents are essentially unable to distinguish the rewards received for the same actions, and are thus unable to update their own Q -values *w.r.t.* the actual contributions. It is worth noting that MAAC is surprisingly inefficient in learning when the number of agents becomes large; it simply fails to handle the non-aggregated noises due to the agents' explorations.

5.6.2 MARL Solutions to Ising Models

Environment. In statistical mechanics, the Ising model is a mathematical framework to describe ferromagnetism ([Ising, 1925](#)). It also has wide applications in socio-physics ([Galam and Walliser, 2010](#)). With the energy function explicitly defined, mean-field approximation ([Stanley, 1971](#)) is a typical way to solve the Ising model for every spin j , *i.e.* $\langle a^j \rangle = \sum_a a^j P(a)$.

An Ising model is defined as a stateless system with N homogeneous sites on a finite square lattice. Each site determines their individual spin a^j to interact with each other and aims to minimise the system energy for a more stable environment. The system energy is defined as $E(a, h) = -\sum_j (h^j a^j + \frac{\lambda}{2} \sum_{k \in \mathcal{N}(j)} a^j a^k)$ where $\mathcal{N}(j)$ is the set of nearest neighbours of site j , $h^j \in \mathbb{R}$ is the external field affecting site j , and $\lambda \in \mathbb{R}$ is an interaction term determines how much the sites tend to align in the same direction. The system is said to reach an equilibrium point when the system energy is minimised, with the probability $P(a) = \frac{\exp(-E(a,h)/\tau)}{\sum_a \exp(-E(a,h)/\tau)}$, where τ is the system temperature. When the temperature rises beyond a certain point (the Curie temperature), the system can no longer keep a stable form, and a phase transition happens. As the ground-truth is known, we would be able to evaluate the correctness of the Q -function learning when there is a large body of interacting agents.

The mean-field theory provides an approximate solution to $\langle a^j \rangle = \sum_a a^j P(a)$

Algorithm 8 MCMC in Ising Model

```

1: initialise spin state  $a \in \{-1, 1\}^N$  for  $N$  sites
2: while training not finished do
3:   randomly choose site  $j \in \mathcal{N}(j)$ 
4:   flip the spin state for site  $j$ :  $a_-^j \leftarrow -a^j$ 
5:   compute energy  $E(a, h) = -\sum_j (h^j a^j + \frac{\lambda}{2} \sum_{k \in \mathcal{N}(j)} a^j a^k)$  for  $a^j$  and  $a_-^j$ 
6:   randomly choose  $\varepsilon \sim U(0, 1)$ 
7:   if  $\exp((E(a^j, h) - E(a_-^j, h)) / \tau) > \varepsilon$  then
8:      $a^j \leftarrow a_-^j$ 
9:   end if
10: end while

```

through a set of self-consistent mean-field equations $\langle a^j \rangle = \frac{\exp(-[h^j a^j + \lambda \sum_{k \in \mathcal{N}(j)} \langle a^k \rangle] / \tau)}{1 + \exp(-[h^j a^j + \lambda \sum_{k \in \mathcal{N}(j)} \langle a^k \rangle] / \tau)}$, which can be solved iteratively by

$$\langle a^j \rangle^{(t+1)} = \frac{\exp(-[h^j a^j + \lambda \sum_{k \in \mathcal{N}(j)} \langle a^k \rangle^{(t)}] / \tau)}{1 + \exp(-[h^j a^j + \lambda \sum_{k \in \mathcal{N}(j)} \langle a^k \rangle^{(t)}] / \tau)},$$

where t represents the number of iterations.

To fit into the MARL setting, we transform the Ising model into a stage game where the reward for each spin/agent is defined by $r^j = h^j a^j + \frac{\lambda}{2} \sum_{k \in \mathcal{N}(j)} a^j a^k$; here $\mathcal{N}(j)$ is the set of nearest neighbours of spin j , $h^j \in \mathbb{R}$ is the external field affecting the spin j , and $\lambda \in \mathbb{R}$ is an interaction coefficient that determines how much the spins are motivated to stay aligned. Unlike the typical setting in physics, here each spin does not know the energy function, but aims to understand the environment, and to maximise its reward by learning the optimal policy of choosing the spin state: up or down. To learn an optimal joint policy π^* for the Ising model, we use the stateless Q -learning with mean-field approximation (MF- Q), defined as

$$Q^j(a^j, \bar{a}^j) \leftarrow Q^j(a^j, \bar{a}^j) + \alpha [r^j - Q^j(a^j, \bar{a}^j)],$$

where the mean \bar{a}^j is given as the mean $\langle a^j \rangle$ from the last time step, and the individual reward is $r^j = h^j a^j + \frac{\lambda}{2} \sum_{k \in \mathcal{N}(j)} a^j a^k$. To balance the trade-off between exploration and exploitation under low-temperature settings, we use a policy with Boltzmann exploration and a decayed exploring temperature. The temperature for

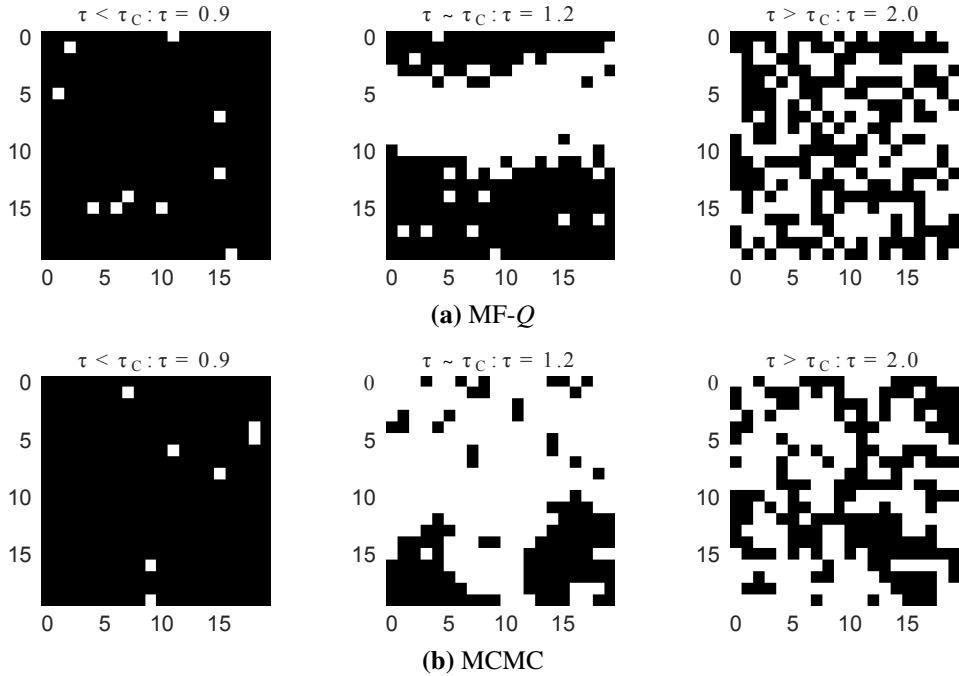


Figure 5.6: The spins of the Ising model at equilibrium under different temperatures.

Boltzmann exploration of MF- Q is multiplied by an exponential decay factor.

Without the loss of generality, we assume $\lambda > 0$, so neighbouring sites with the same action result in lower energy (observe higher reward) and are more stable. Each site should also align with the sign of external field h^j to reduce the system energy. For simplification, we eliminate the effect of external fields and assume the model to be discrete, *i.e.*, $\forall j \in N, h^j = 0, a^j \in \{-1, 1\}$.

In addition to the reward, the *order parameter* (OP) (Stanley, 1971) is a traditional measure of purity for the Ising model. OP is defined as $\xi = \frac{|N_\uparrow - N_\downarrow|}{N}$, where N_\uparrow represents the number of up spins, and N_\downarrow for the down spins. The closer the OP is to 1, the more orderly the system is.

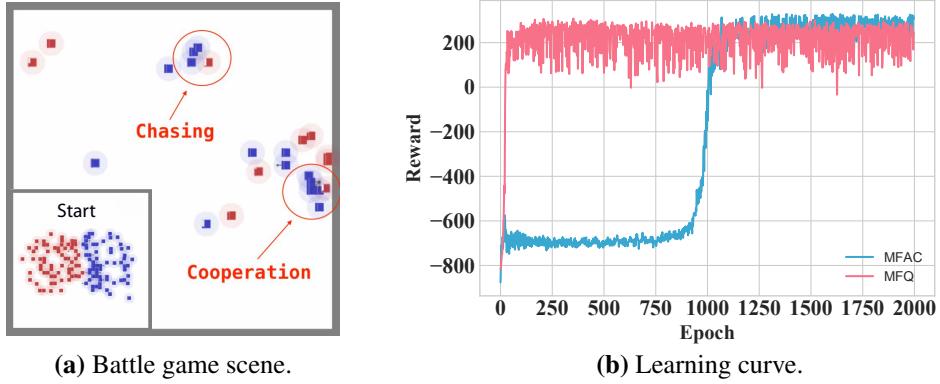
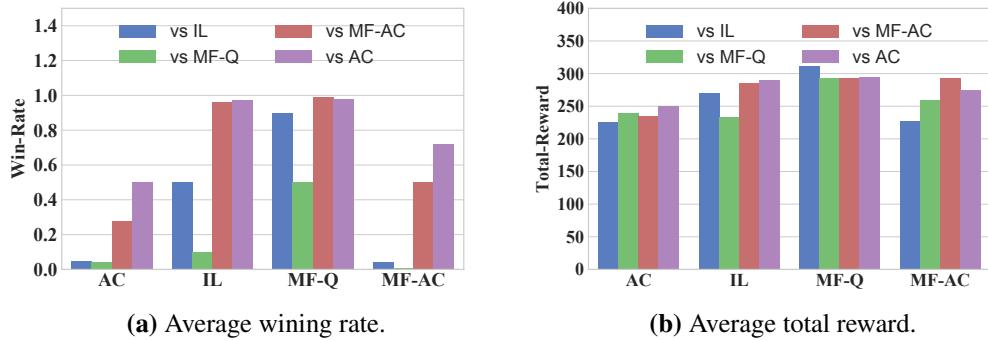
Model Settings. To validate the correctness of MF- Q learning, we implement MCMC methods (Binder et al., 1993) to simulate the same Ising model and provide the ground truth for comparison. Specifically, we simulate the Ising model using Metropolis Monte Carlo methods (MCMC) in Algorithm 8. After initialisation, we randomly change a site's spin state and calculate the energy change, select a random number between 0 and 1, and accept the state change only if the number is less than $e^{\frac{(E^j - E_{-}^j)}{\tau}}$. This is called the Metropolis technique, which saves computation time by selecting the more probable spin states.

One of the learning goals is to obtain the accurate approximation of $\langle a^j \rangle$. Notice that agents here do not precisely know the energy function, but instead use temporal difference learning to approximate $\langle a^j \rangle$ during the learning procedure. Once this is accurately approximated, the Ising model as a whole should be able to converge to the same simulation result suggested by MCMC.

Correctness of MF- Q . Figure. 5.4 illustrates the relationship between the order parameter at equilibrium under different system temperatures. MF- Q converges nearly to the same plot as MCMC, which justifies our algorithms' correctness. Critically, MF- Q finds a similar Curie temperature (the phase change point) as MCMC, that is $\tau = 1.2$. As far as we know, this is the first work that manages to solve the Ising model via model-free RL methods. Figure. 5.5 illustrates the mean squared error between the learned Q -value and the reward target. MF- Q is shown in Figure 5.5a to be able to learn the target well under low-temperature settings. When it comes to the Curie temperature, the environment enters into the phase change when the stochasticity dominates, resulting in a lower OP and higher MSE observed in Figure 5.5b. We visualise the equilibrium in Figure 5.6. The equilibrium points from MF- Q in fact match MCMC's results under three types of temperatures. The spins tend to stay aligned under a low temperature ($\tau = 0.9$). As the temperature rises ($\tau = 1.2$), some spins become volatile, and patches start to form as spontaneous magnetisation. This phenomenon is mostly observed around the Curie temperature. After passing the Curie temperature, the system becomes unstable and disordered due to the large thermal fluctuations, resulting in random spinning patterns.

5.6.3 Many-Agent Battle Games

Environment. The Battle game in the Open-source MAgent system (Zheng et al., 2018) is a *Mixed Cooperative-Competitive* scenario with two armies fighting against each other in a grid world, each empowered by a different RL algorithm. In the setting of Figure 5.7a, each army consists of 64 homogeneous agents. Each army's goal is to get more rewards by collaborating with teammates to destroy all the opponents. An agent can take actions to either move to or attack nearby grids. Agents are expected to learn skills such as chasing to hunt. We adopt the default reward

**Figure 5.7:** The battle game: 64 v.s. 64.**Figure 5.8:** Performance comparisons in the battle game. ‘‘IL’’ stands for the independent learning method.

setting: -0.005 for every move, 0.2 for attacking an enemy, 5 for killing an enemy, -0.1 for attacking an empty grid, and -0.1 for being attacked or killed.

Model Settings. Our MF-*Q* and MF-AC are compared against the baselines that are proved successful on the MAgent platform. We focus on the battles between mean-field methods (MF-*Q*, MF-AC) and their non-mean field counterparts, independent *Q*-learning (IL) and advantageous actor critic (AC). We exclude MADDPG/MAAC as baselines, as the framework of centralised critic cannot deal with the varying number of agents for the battle (simply because agents could die in the battle). Also, as we demonstrated in the previous experiment of Figure 5.3, MAAC tends to scale poorly and fail when N is in hundreds.

Hyper-parameter Settings. IL and MF-*Q* have almost the same hyper-parameter settings. The learning rate is $\alpha = 10^{-4}$ and the dynamic exploration rate linearly decays from $\gamma = 1.0$ to $\gamma = 0.05$ during the 2000 rounds training. The discounted factor γ is set to be 0.95 , and the mini-batch size is 128. The size of the

replay buffer is 5×10^5 . AC and MF-AC also have almost the same hyper-parameter settings. The learning rate is $\alpha = 10^{-4}$, the temperature of soft-max layer in *actor* is $\tau = 0.1$. And the coefficient of entropy in the total loss is 0.08, the coefficient of value in the total loss is 0.1.

Results and Discussion. We train all four models by 2000 rounds *self-plays*, and then use them for comparative battles. During the training, agents can quickly pick up chasing and cooperation skills to kill in Figure 5.7a. Figure 5.8 shows the result of winning rate and the total reward over 2000 rounds cross-comparative experiments. It is evident that on all the metrics mean-field methods, MF-*Q* vastly outperforms the corresponding baselines, *i.e.* IL and AC respectively, which shows the effectiveness of the mean-field MARL algorithms. Interestingly, IL performs far better than AC and MF-AC (check the 2nd block in Figure 5.8a), although it is worse than the mean-field counterpart MF-*Q*. This might imply the effectiveness of off-policy learning with shuffled buffer replay in many-agent RL towards a more stable learning process. Also, the *Q*-learning family tends to introduce a positive bias (Hasselt, 2010) by using the maximum action value as an approximation for the maximum expected action value, and such overestimation can be beneficial for every single agent to find the best response to others even though the environment itself is still changing. On the other hand, On-policy methods need to comply with the GLIE assumption (Assumption 2 in Sec 3.3) so as to converge properly to the optimal value (Singh et al., 2000a), which is, in the end, a greedy policy as off-policy methods. Figure. 5.7b further shows the self-play learning curves of MF-AC and MF-*Q*. MF-*Q* presents a faster convergence speed than MF-AC, which is consistent with the findings in the Gaussian Squeeze task (see Figure 5.3b & 5.3c).

5.7 Chapter Summary

In this chapter, we developed mean-field MARL methods to model the dynamics of interactions in the many-agent systems. Leveraging on the mean-field approximation, I proposed a practical answer to the many-agent policy learning problem. Specifically, my method of MF-*Q* iteratively learns each agent’s best response to the

mean effect from its neighbours; this effectively transforms the many-body problem into a two-body problem. Theoretical analysis on the convergence of the MF- Q algorithm to Nash Q -value was provided, covering both tabular cases and linear function approximations. Three types of tasks have justified the effectiveness of our approaches. In particular, I report the first result to solve the Ising model in physics using model-free RL methods.

Chapter 6

Many-Agent Learning in Open-Ended Meta-Games

In this chapter¹, I look into the problem of conducting many-agent learning at the meta-game level. Unlike previous chapters in which an agent by default represents a player in a game, in the meta-game, a player is assumed to have a population of many policies (e.g., different tactics in playing Poker), and each policy in the population is regarded as an “agent”. As a result, many-agent learning in meta-games essentially deals with the interactions between different policies of one player, such as finding a new better-performing policy (i.e., “agent”) that can exploit other policies in that player’s population.

Meta-game is an essential notion towards solving real-world games (see Chapter 2.3.6 for a reminder), especially for games in which enumerating all possible pure strategies is infeasible. For example, in Poker games, the total number of pure strategies, which is the cartesian product of all atomic actions (e.g., “fold”) under all information states, can easily be an astronomical number, and therefore directly solving the Poker game itself is challenging. However, an alternative approach can be to construct a higher-level meta-game where an “agent” corresponds to a policy (e.g., the tactic of “play defensively”), and then tackle the task of learning a

¹Please note that an early work on the similar topic of using determinantal point process to model behavioural diversity in the policy space was studied in [Slumbers \(2020\)](#) under my supervision. On top of such early effort, significant amount of developments, both in terms of theory proofs and experiments, have been further achieved. Necessary demarcation has been made for those results that are derived in [Slumbers \(2020\)](#) but not my contributions.

population of effective policies (i.e., “agents”) that can solve the meta-game (i.e., having zero exploitability). Since the number of “agents” is usually far smaller than the number of pure strategies in the underlying games, solving the meta-game is considered as more effective.

In this chapter, I focus on how to promote behavioural diversity for many-agent learning in meta-games. In fact, promoting behavioural diversity is critical for solving real-world games. In games with non-transitive dynamics (e.g., Rock-Paper-Scissors game) where strategy cycles exist, and there is no consistent winner, acquiring a diverse set of effective strategies helps an agent keep exploiting opponents that are not encountered during training. Despite the importance, there is a lack of rigorous treatment for measuring behavioural diversity in games; as a result, it is unclear how to construct sequences of diversity-aware objectives that yield effective open-ended learning processes.

This chapter offers a new geometric interpretation of behavioural diversity in (meta-)games. Specifically, I introduce a diversity metric based on *determinantal point processes (DPP)*. A DPP defines a probability measure that models the likelihood of choosing a random subset from a ground set where diverse subsets are preferred. I adapt DPP for learning in games. In particular, I consider the expected cardinality as the diversity measure such that only the strategies that have diverse performance in terms of payoffs will be selected. Our diversity measure is a general metric, from which I, respectively, develop *diverse fictitious play*, and *diverse policy-space response oracle* algorithms for solving normal-form games, where the total number of strategies is finite and known, and open-ended meta-games, where the total number of strategies is infinite and unknown. Theoretically, I prove the uniqueness of the *diverse best response* as well as the convergence of our algorithms on two-player games. Importantly, I show that maximising the expected cardinality of DPP is guaranteed to increase the *gamescapes* – convex polytopes that encode agents’ mixtures of strategies. To validate our diversity-aware solvers, I apply them on a series of games that involve both transitive and non-transitive dynamics, including randomised Games of Skill, 2D-Rock Paper Scissors, and Colonel Blotto,

against strong baselines. Results suggest that our methods consistently outperform the existing state-of-the-art.

6.1 Background and Motivation

Nature exhibits a remarkable tendency towards *diversity* (Holland et al., 1992). Over the past billions of years, natural evolution has discovered a vast assortment of unique and diversified species. Each of them is capable of orchestrating the complex biological processes necessary to survive. Equally, in computer science, machine intelligence can be considered the ability to adapt to a diverse set of complex environments (Hernández-Orallo, 2017), which suggests that the ceiling of intelligence may be raised by providing environments of increasing diversity and complexity. Recent successes of AIs that achieved superhuman performance on the game of GO (Silver et al., 2016) and StarCraft (Vinyals et al., 2019a) provides justification for considering behavioural diversity in the design of learning protocols, in that it plays a pivotal role in not only preventing AI agents from checking the same policies again and again, but more importantly, boosting the performance of the whole population of strategies when encountering unseen opponents.

Biodiversity in ecosystems is built up by the cyclic non-transitive interactions among competing populations (Kerr et al., 2002; Reichenbach et al., 2007). Such non-transitive relations can be thought of as a Rock-Paper-Scissors (RPS) game, in which Rock crushes Scissors, Scissors cuts Paper, and Paper wraps Rock. The natural law indicates that *diversity* and *non-transitivity* are somehow intertwined. Interestingly, in game theory, the necessity of pursuing behavioural diversity is deeply rooted in the non-transitive nature of games (Balduzzi et al., 2019, 2018b; Lanctot et al., 2017). In general, an arbitrary game, of either the normal-form type (Candogan et al., 2011) or the differential type (Balduzzi et al., 2018a), can always be decomposed into a sum of two components: a transitive part plus a non-transitive part. The transitive part of a game represents the component in which the law of winning is transitive (e.g., if strategy A beats B, B beats C, then we know A will beat C), and the non-transitive part refers to the component in which the set of strategies follow

the conservation law (e.g., the endless cycles between Rock, Paper, and Scissors). Diversity matters especially for these non-transitive parts simply because there is no consistent winner in such games, and a player who only plays a small subset of strategies can easily be exploited. Real-world games often consist of a mixture of both transitive and non-transitive components (Czarnecki et al., 2020), therefore, it is critical to design diversity-aware objectives inside the learning protocols.

Despite the importance of diversity, there is very limited work that models diversity in a mathematically rigorous way. The majority of work so far has followed on a biological approach, borrowing ideas directly from natural evolution. The idea of *co-evolution* (Durham, 1991; Paredis, 1995) has drawn forth a series of effective heuristics in designing learning protocols in games. The goal of co-evolution is to automatically generate an endless procession of increasingly better-performing agents by exerting selection pressure on multiple self-optimising agents. Following this idea, techniques such as open-ended evolution (Banzhaf et al., 2016; Lehman and Stanley, 2008; Standish, 2003), population based training methods (Jaderberg et al., 2019; Liu et al., 2018), and auto-curricula (Baker et al., 2019a; Leibo et al., 2019) have shown great empirical success in solving complicated real-world games. Although these methods are related to fundamental game theoretical ideas such as fictitious play (Brown, 1951) and self-play (Fudenberg et al., 1998), the underlying principle that induces diversity is yet unclear, and theoretical justifications are missing. The lack of a rigorous solution for measuring diversity further hinders one from designing a principled approach to generating diversity during training. For example, it has been found that if diversity is promoted in the wrong way, then the idea of “*diversity trumps ability*” will not hold (Hong and Page, 2004; Ostrom, 2009), especially when improving diversity is in the opposite direction to improving *quality* (measured by agent ability). Additionally, there is empirical evidence showing that diversity is only helpful when a minimum of quality is guaranteed, otherwise it will be detrimental (Pugh et al., 2016).

In this work, we introduce a rigorous way of modelling diversity for learning in games. Our approach offers a new geometric interpretation of behavioural di-

versity; it is built upon *determinantal point processes (DPP)* that originate in the modelling of repulsive quantum particles in physics (Macchi, 1977). A DPP is a special type of point process, which measures the probability of selecting a random subset from a ground set where only diverse subsets are desired. We adapt DPPs to learning in games by proposing the expected cardinality of DPPs as the diversity metric. Based on this new metric, we further design diversity-aware extensions of fictitious play (FP) (Brown, 1951) and policy-space response oracles (PSRO) (Balduzzi et al., 2019; Lanctot et al., 2017; Muller et al., 2019). The main advantage of our methods is that promoting diversity during training consequently leads to more effective learning and lower exploitability. We show that maximising the expected cardinality guarantees an expansion of the gamescape spanned by agents' mixtures of policies, whilst at the same time, our learning algorithms maintain the same convergence properties as FP and PSRO. We evaluate our methods on three types of games, covering both normal-form games and open-ended meta-games, and each of these games involves both transitive and non-transitive dynamics. Results show that our diversity-aware learning algorithms achieve state-of-the-art performance against a series of strong baselines.

6.2 Related Work

Modelling diversity has been extensively studied in evolutionary computation (EC) (Bäck et al., 1997; Fogel, 2006). EC methods mimic the selection/mutation processes of natural evolution. One classical idea of EC is *novelty search* (Lehman and Stanley, 2008, 2011a) which aims to search only for behavioural diversity. Quality-diversity (QD) methods hybridise novelty search with fitness competitions under the notion of *survival of the fittest* (Pugh et al., 2016); two representatives are *Novelty Search with Local Competition* (Lehman and Stanley, 2011b) and *MAP-Elites* (Cully et al., 2015; Mouret and Clune, 2015). Solving a game can be regarded as an open-ended system in which a continual production of policies is needed, and an adaptive objective guides agents to keep improving endlessly (i.e., co-evolution) (Balduzzi et al., 2019; Leibo et al., 2019). In co-evolution for games, QD meth-

ods are applied to continually optimise for better-performing and diverse policies for each learning agent (Banzhaf et al., 2016; Lehman and Stanley, 2008; Taylor et al., 2016). Although remarkable successes have been demonstrated in tackling complicated games (Baker et al., 2019a; Cully et al., 2015; Jaderberg et al., 2019), defining diversity in EC methods is often task-dependent, and a universal measure is lacking. Furthermore, due to the biological nature of EC methods, developing a theoretical understanding of how diversity is generated is non-trivial (Brown et al., 2005).

Searching for behavioural diversity is also a common topic in RL, which is often studied in the context of skill discovery (Eysenbach et al., 2018; Florensa et al., 2017; Hausman et al., 2018), intrinsic rewards (Barto, 2013; Bellemare et al., 2016; Gregor et al., 2017), or maximum-entropy learning (Haarnoja et al., 2017, 2018; Levine, 2018). These RL algorithms can still be regarded as QD methods, in the sense that quality means maximising cumulative reward, and diversity means different things depending on the context, e.g., visiting a new state (Eysenbach et al., 2018), or, obtaining a policy with larger entropy (Levine, 2018). One related work in RL, yet with a different focus, is Q-DPP (Yang et al., 2020), a multi-agent RL algorithm that adopts DPP as a function approximator to factorise agents' joint Q-functions in the context of solving cooperative MARL tasks.

Our work is also related to smooth fictitious play (Fudenberg and Levine, 1995; Hofbauer and Sandholm, 2002), a variant of fictitious play (FP) (Brown, 1951) that perturbs the best response dynamics with an entropy term to account for diversity. For solving two-player zero-sum games, Double Oracle (DO) was proposed; its idea is to best respond to the opponent's Nash Equilibrium (NE) policy at each iteration (McMahan et al., 2003). *Policy Space Response Oracle (PSRO)* generalises FP and DO by allowing an RL subroutine powered by deep neural networks to approximate the best response (Lanctot et al., 2017). Considering behavioural diversity, Balduzzi et al. (2019) introduced *Rectified PSRO (PSRO_{rN})* that adaptively encourages agents to amplify their strengths and ignore their weaknesses in the PSRO protocol. However, PSRO_{rN} suffers from poor empirical performance (McAleer

et al., 2020; Muller et al., 2019). In solving general-sum games, since computing NE is *PPAD*-hard (Chen et al., 2009; Daskalakis et al., 2009), Muller et al. (2019) developed α -*PSRO* that replaces NE with α -*Rank* (Omidshafiei et al., 2019a; Yang et al., 2019a), a new type of solution concept that has polynomial-time solutions on multi-player general-sum games. Yet, how to promote behavioural diversity in the context of α -*PSRO* is unclear. In this work, based on a geometric interpretation of games, we offer a rigorous behavioural diversity measure. We show that our diversity measure can be easily incorporated into FP, PSRO, and α -*PSRO*. Notably, on three challenging games, our methods outperform all these baselines by a significant margin.

6.3 Preliminary and Notations

We consider normal-form games (NFG), denoted by $\langle \mathcal{N}, \mathbb{S}, \mathbf{G} \rangle$, where each player $i \in \mathcal{N}$ has a finite set of pure strategies \mathbb{S}^i . Let $\mathbb{S} = \prod_{i \in \mathcal{N}} \mathbb{S}^i$ denote the space of joint pure-strategy profiles, and \mathbb{S}^{-i} denote the set of joint strategy profiles except the i -th player. A mixed strategy of player i is written by $\boldsymbol{\pi}^i \in \Delta_{\mathbb{S}^i}$ where Δ is a probability simplex. A joint mixed-strategy profile is $\boldsymbol{\pi} \in \Delta_{\mathbb{S}}$, and $\boldsymbol{\pi}(S) = \prod_{i \in \mathcal{N}} \boldsymbol{\pi}^i(S^i)$ represents the probability of joint strategy profile S . For each $S \in \mathbb{S}$, let $\mathbf{G}(S) = (\mathbf{G}^1(S), \dots, \mathbf{G}^N(S)) \in \mathbb{R}^N$ denote the vector of payoff values for each player. The expected payoff of player i under a joint mixed-strategy profile $\boldsymbol{\pi}$ is thus written as $\mathbf{G}^i(\boldsymbol{\pi}) = \sum_{S \in \mathbb{S}} \boldsymbol{\pi}(S) \mathbf{G}^i(S)$, which we also denote as $\mathbf{G}^i(\boldsymbol{\pi}^i, \boldsymbol{\pi}^{-i})$.

6.3.1 Solution Concepts of Games

A Nash equilibrium (NE) is proved to exist in all finite games (Nash et al., 1950); it is a special kind of joint mixed-strategy profile $\boldsymbol{\pi}$ in which each player $i \in \mathcal{N}$ is acting in their best response to the rest of agents, written as $\boldsymbol{\pi}^i \in \mathbf{BR}^i(\boldsymbol{\pi}^{-i}) = \arg \max_{\boldsymbol{\pi} \in \Delta_{\mathbb{S}^i}} [\mathbf{G}^i(\boldsymbol{\pi}, \boldsymbol{\pi}^{-i})]$. $\mathbf{BR}_\varepsilon^i(\boldsymbol{\pi}^{-i}) = \{ \boldsymbol{\pi}^i : \mathbf{G}^i(\boldsymbol{\pi}^i, \boldsymbol{\pi}^{-i}) \geq \mathbf{G}^i(\mathbf{BR}^i(\boldsymbol{\pi}^{-i}), \boldsymbol{\pi}^{-i}) - \varepsilon \}$ defines the set of ε -best responses ($\varepsilon > 0$) to the strategy profile $\boldsymbol{\pi}^{-i}$, and an ε -NE is a joint profile $\boldsymbol{\pi}$ s.t. $\boldsymbol{\pi}^i \in \mathbf{BR}_\varepsilon^i(\boldsymbol{\pi})$, $\forall i \in \mathcal{N}$. *Exploitability* (Lanctot et al.,

2017) measures the distance of a joint strategy $\boldsymbol{\pi}$ to a NE, written as

$$\text{Exploitability}(\boldsymbol{\pi}) = \sum_{i \in \mathcal{N}} \left[\mathbf{G}^i(\mathbf{BR}^i(\boldsymbol{\pi}^{-i}), \boldsymbol{\pi}^{-i}) - \mathbf{G}^i(\boldsymbol{\pi}) \right]. \quad (6.1)$$

When the exploitability reaches zero, then $\boldsymbol{\pi}$ is one NE.

Computing NE in multi-player general-sum games is hard (Daskalakis et al., 2009). No polynomial-time solution is available even in two-player cases (Chen et al., 2009). Additionally, NE may not be unique. α -Rank (Omidshafiei et al., 2019a; Yang et al., 2019a) is a new type of solution concept that serves as a promising replacement for NE. The key associated benefits are its uniqueness, and its polynomial-time solvability in multi-player general-sum games.

The idea of α -Rank is built on the *response graph* of a game. On the response graph, each joint pure-strategy profile is a node, and a directed edge points from node $\sigma \in \mathbb{S}$ to node $S \in \mathbb{S}$ if 1) σ and S differ in only one single player's strategy, and 2) that deviating player, denoted by i , benefits from deviating from S to σ such that $\mathbf{G}^i(\sigma) > \mathbf{G}^i(S)$. We are interested in the so-called *sink strongly-connected components (SSCC)* nodes that have only incoming edges but no outgoing edges on the response graph. To find those SSCC nodes, α -Rank constructs a random walk along the directed response graph, which can be equivalently described by a Markov chain, with the transition probability matrix \mathbf{C} being:

$$\mathbf{C}_{S,\sigma} = \begin{cases} \eta \frac{1 - \exp(-\alpha(\mathbf{G}^k(\sigma) - \mathbf{G}^k(S)))}{1 - \exp(-\alpha m(\mathbf{G}^k(\sigma) - \mathbf{G}^k(S)))} & \text{if } \mathbf{G}^k(\sigma) \neq \mathbf{G}^k(S) \\ \frac{\eta}{m} & \text{otherwise} \end{cases},$$

$$\mathbf{C}_{S,S} = 1 - \sum_{i \in \mathcal{N}} \mathbf{C}_{S,\sigma} \quad (6.2)$$

$\eta = (\sum_{i \in \mathcal{N}} (|S^i| - 1))^{-1}$, $m \in \mathbb{N}$, $\alpha > 0$ are three constants. Large α ensures the Markov chain is irreducible, and thus guarantees the existence and uniqueness of α -Rank solution, which is the resulting unique stationary distribution $\boldsymbol{\pi}$ of the Markov chain, $\mathbf{C}^\top \boldsymbol{\pi} = \boldsymbol{\pi}$. The probability mass of each joint strategy in $\boldsymbol{\pi}$ can be interpreted as the longevity of that strategy throughout the evolution (Omidshafiei et al., 2019a).

6.3.2 Formulation of Open-Ended Meta-Game

The framework of NFGs is often limited in describing real-world games. In games like Poker or GO, it is impossible to list all atomic pure strategies; instead, we are more interested in “higher-level” strategic policies (e.g., “bluff”), and the resulting game is a *meta-game*, denoted by $\langle \mathcal{N}, \mathbb{S}, \mathcal{M} \rangle$. A meta-game payoff table \mathcal{M} is constructed by simulating games that cover all combinations of “high-level” policies, with entries corresponding to the players’ empirical payoffs under a certain joint “high-level” policy profile. With slight abuse of notation², we respectively use \mathbb{S}^i to denote the “high-level” policy set (e.g., “bluff” and “semi-bluff”), and use $\boldsymbol{\pi}^i \in \Delta_{\mathbb{S}^i}$ to denote the meta-policy on the “high-level” policy set (e.g., the i -th player bluffs 30% of the time and semi-bluffs 70% of the time). $\boldsymbol{\pi} = (\boldsymbol{\pi}^1, \dots, \boldsymbol{\pi}^N)$ is a joint meta-policy profile. In meta-game analysis (*a.k.a.* *empirical game-theoretic analysis* (EGTA)) (Tuyls et al., 2018; Wellman, 2006)), traditional game-theoretical concepts (e.g., NE, α -Rank) can still be computed based on \mathcal{M} , even in a more scalable manner, this is because the number of “higher-level” strategies in the meta-game is usually far smaller than the number of atomic actions of the underlying game. Furthermore, it has been shown that an ε -NE of the meta-game is in fact a 2ε -NE of the underlying game (Tuyls et al., 2018).

Many real-world games (e.g., Poker, GO, and StarCraft) can be described by an open-ended zero-sum meta-game (Balduzzi et al., 2019, 2018b; Lanctot et al., 2017). In such a game, the game engine is represented by an evaluation function $\phi : \mathbb{S}^1 \times \mathbb{S}^2 \rightarrow \mathbb{R}$ such that if $S^1 \in \mathbb{S}^1$ beats $S^2 \in \mathbb{S}^2$, then $\phi(S^1, S^2) > 0$, and $\phi < 0, \phi = 0$ refers to losses and ties. The sets of \mathbb{S}^1 and \mathbb{S}^2 can be regarded as two populations of deep neural networks (DNNs) and each S^1, S^2 is a DNN with independent weights. The meta-game payoff is $\mathcal{M} = \{\phi(S^1, S^2) : (S^1, S^2) \in \mathbb{S}^1 \times \mathbb{S}^2\}$, and it is *symmetric* if $\mathbb{S}^1 = \mathbb{S}^2$, and $\phi(S^1, S^2) = -\phi(S^2, S^1), \forall S^1, S^2 \in \mathbb{S}^1$. A game is **transitive** if the ϕ can be represented by a monotonic rating function f such that performance on the game is the difference in ratings: $\phi(S^1, S^2) = f(S^1) - f(S^2)$, and it is **non-transitive** if ϕ satisfies $\int_{S^2 \in \mathbb{S}^2} \phi(S^1, S^2) \cdot dS^2 = 0$, meaning that winning against some strategies

²The difference between NFGs and meta-games can be told by the payoff of \mathbf{G} vs. \mathcal{M} .

Algorithm 9 A General Solver for Open-Ended Meta-Games

```

1: Initialise: the "high-level" policy set  $\mathbb{S} = \prod_{i \in \mathcal{N}} \mathbb{S}^i$ , the meta-game payoff
    $M, \forall S \in \mathbb{S}$ , and meta-policy  $\boldsymbol{\pi}^i = \text{UNIFORM}(\mathbb{S}^i)$ .
2: for iteration  $t \in \{1, 2, \dots\}$  do:
3:   for each player  $i \in \mathcal{N}$  do:
4:     Compute the meta-policy  $\boldsymbol{\pi}_t$  by meta-game solver  $\mathcal{S}(M_t)$ .
5:     Find a new policy against others by Oracle:  $S_t^i = \mathcal{O}^i(\boldsymbol{\pi}_t^{-i})$ .
6:     Expand  $\mathbb{S}_{t+1}^i \leftarrow \mathbb{S}_t^i \cup \{S_t^i\}$  and update meta-payoff  $M_{t+1}$ .
7:   terminate if:  $\mathbb{S}_{t+1}^i = \mathbb{S}_t^i, \forall i \in \mathcal{N}$ .
8: Return:  $\boldsymbol{\pi}$  and  $\mathbb{S}$ .

```

will be counterbalanced with losses against other strategies in the population.

Meta-games are often **open-ended**³ because in general there exists an infinite number of pure strategies in real-world games, and, as new strategies will be discovered and added to agents' strategy sets during training, the dimension of the meta-game M will also be expanded. The *gamescape* (Balduzzi et al., 2019) of a population of strategies is defined as the convex hull of payoff vectors of strategies in \mathbb{S} , written as

$$\text{Gamescape}(\mathbb{S}) := \left\{ \sum_i \alpha_i \cdot \mathbf{m}_i : \boldsymbol{\alpha} \geq 0, \boldsymbol{\alpha}^\top \mathbf{1} = 1, \mathbf{m}_i = M_{[i,:]} \right\}. \quad (6.3)$$

6.3.3 Solving Open-Ended Meta-Games

Fictitious play (FP) (Brown, 1951) is a standard game-theoretic model of learning in NFGs. In the process of FP, at each iteration, each player chooses a best response to their opponents' time-average strategies. The resulting average strategies of all players in the limit of time will converge to a NE in games such as two-player zero-sum games, or, potential games (Monderer and Shapley, 1996). *Generalised weakened fictitious play (GWFP)* (Leslie and Collins, 2006) is a generalised version of FP. It maintains the same convergence property but allows for approximate best responses and perturbed average strategy updates. Formally, it is written as

Definition 12 (Generalised weakened fictitious play). *A GWFP is a process of*

³When the underlying game is an NFG, the meta-game will not be open-ended because it cannot go beyond the size of NFG. When "high-level" policies of a meta-game overlap with the pure strategies of an NFG, that meta-game recovers the underlying NFG.

Table 6.1: Variations of Different Meta-Game Solvers

Method	\mathcal{S}	\mathcal{O}	Game type
Self-play (Fudenberg et al., 1998)	$[0, \dots, 0, 1]^N$	$\mathbf{Br}(\cdot)$	multi-player potential
GWFP (Leslie and Collins, 2006)	UNIFORM	$\mathbf{Br}_\varepsilon(\cdot)$	two-player zero-sum / potential
Double Oracle (McMahan et al., 2003)	NE	$\mathbf{Br}(\cdot)$	two-player zero-sum
PSRO _N (Lanctot et al., 2017)	NE	$\mathbf{Br}_\varepsilon(\cdot)$	two-player zero-sum
PSRO _{rN} (Balduzzi et al., 2019)	NE	Rectified Br _{ε} (\cdot)	symmetric zero-sum
α -PSRO (Muller et al., 2019)	α -Rank	PBR (\cdot)	multi-player general-sum
Our methods	NE/ α -Rank	Eq. (6.20) / (6.22)	two-player general-sum

$\{\boldsymbol{\pi}_t\}_{t \geq 0}$ with $\boldsymbol{\pi}_t \in \prod_{i \in \mathcal{N}} \Delta_{\mathbb{S}^i}$, such that

$$\boldsymbol{\pi}_{t+1}^i \in (1 - \alpha_{t+1}) \boldsymbol{\pi}_t^i + \alpha_{t+1} (\mathbf{BR}_{\varepsilon_t}^i(\boldsymbol{\pi}_t^{-i}) + M_{t+1}^i) \quad (6.4)$$

with $\alpha_t \rightarrow 0$ and $\varepsilon_t \rightarrow 0$ as $t \rightarrow \infty$, $\sum_{t \geq 1} \alpha_t = \infty$, and $\{M_t\}_{t \geq 1}$ is a sequence of perturbations that satisfies, $\forall T > 0$,

$$\lim_{t \rightarrow \infty} \sup_k \left\{ \left\| \sum_{i=t}^{k-1} \alpha_{i+1} M_{i+1} \right\| \text{ s.t. } \sum_{i=n}^{k-1} \alpha_i \leq T \right\} = 0. \quad (6.5)$$

The original FP is recovered with $\alpha_t = 1/t$, $\varepsilon_t = 0$ and $M_t = 0, \forall t$.

Similar to GWFP for solving NFGs, a general solver for open-ended meta-games involves an iterative two-step process of solving the meta-policy first, and then based on the meta-policy, finding a new better-performing policy to augment the existing population (see Algorithm 9). The meta-policy solver, denoted as $\mathcal{S}(\cdot)$, computes a joint meta-policy profile $\boldsymbol{\pi}$ based on the current payoff M where different solution concepts can be adopted (e.g., NE, α -Rank). Finding a new policy is

equivalent to solving a single-player optimisation problem given opponents' policy sets \mathbb{S}^{-i} and meta-policies $\boldsymbol{\pi}^{-i}$, which are fixed and known. One can regard a new policy as given by an *Oracle*, denoted by \mathcal{O} . In two-player zero-sum cases, an oracle represents $\mathcal{O}^1(\boldsymbol{\pi}^2) = \{S^1 : \sum_{S^2 \in \mathbb{S}^2} \boldsymbol{\pi}^2(S^2) \cdot \phi(S^1, S^2) > 0\}$. Generally, Oracles can be implemented through optimisation subroutines such as RL algorithms or zero-order methods. Finally, after a new policy is found, the payoff table M is expanded, and the missing entries are filled by running new game simulations. The above two-step process loops over each player at each iteration, and it terminates if no new policies can be found for any players.

Algorithm 9 is a general framework, with appropriate choices of meta-game solver \mathcal{S} and Oracle \mathcal{O} , it can represent solvers for different types of meta-games. For example, it is trivial to see that FP is recovered when $\mathcal{S} = \text{UNIFORM}(\cdot)$ and $\mathcal{O}^i = \text{BR}^i(\cdot)$. The Double Oracle (McMahan et al., 2003) and PSRO methods (Balduzzi et al., 2019) refer to the cases when the meta-solver computes NE. We summarise variations of meta-game solvers in Table 6.1. It is worth mentioning that when $\mathcal{S} = \alpha$ -Rank, the design of the Oracle is non-trivial. Muller et al. (2019) showed that a simple best response fails to converge to the SSCC of a response graph before termination, instead, they proposed *Preference-based Best Response* (*PBR*) Oracle that suits α -Rank, written by

$$\text{PBR}^i(\boldsymbol{\pi}^{-i}) \subseteq \arg \max_{\sigma \in \mathbb{S}^i} \mathbf{E}_{S^{-i} \sim \boldsymbol{\pi}^{-i}} \left[\mathbf{1}[M^i(\sigma, S^{-i}) > M^i(S^i, S^{-i})] \right]. \quad (6.6)$$

6.3.4 Effective Diversity

Promoting behavioural diversity is important particularly in games with non-transitive dynamics (e.g., RPS game). Since there is no consistent winner, exploring diverse strategies expands the gamescape, and ultimately helps a player to keep exploiting different types of opponents. On the other hand, there are typically far more ways of acting incompetently than acting effectively (Czarnecki et al., 2020); it is therefore necessary to consider the quality factor as well as diversity.

Considering both quality and diversity for solving NFGs, the smooth FP method (Fudenberg and Levine, 1995) incorporates an entropy term $\mathcal{H}(\cdot)$ in addition to the best response to advocate behavioural diversity, written as $\pi^i \in \mathbf{BR}_\epsilon^i(\boldsymbol{\pi}^{-i}) = \arg \max_{\pi \in \Delta_{S^i}} [\mathbf{G}^i(\pi, \boldsymbol{\pi}^{-i}) + \tau \cdot \mathcal{H}(\pi)]$ where τ is a temperature constant. In the case of $\tau_t \rightarrow 0$ as $t \rightarrow \infty$, smooth FP converges to the GWFP process almost surely (Leslie and Collins, 2006).

Entropy measures the diversity of a policy in terms of its **randomness**; however, when it comes to solving open-ended meta-games, measuring diversity against peer models in the population becomes critical. Towards this end, *effective diversity* (*ED*) (Balduzzi et al., 2019) is proposed to quantify the diversity for a population of policies \mathbb{S} by

$$\text{ED}(\mathbb{S}) = \boldsymbol{\pi}^{*\top} [M]_+ \boldsymbol{\pi}^*, \quad (6.7)$$

where $[x]_+ := x$ if $x \geq 0$, and 0 otherwise, M is the meta-payoff table of \mathbb{S} , and $\boldsymbol{\pi}^*$ is the NE of M . Using the Nash distribution ensures that the diversity is only related to the best-responding models, and the rectifier $[x]_+$ further quantifies the variety of ways of how these “winner” models (those within the support of NE) beat each other. It is clear to see that $\text{ED}(\mathbb{S}) = 0$ if there is only one dominant policy in \mathbb{S} . To promote ED during training, an Oracle that suits rectified NE is introduced in the PSRO process, namely PSRO_{rN} , written as

$$\mathcal{O}^1(\boldsymbol{\pi}^2) = \left\{ S^1 : \sum_{S^2 \in \mathbb{S}^2} \boldsymbol{\pi}^{2,*}(S^2) \cdot [\phi(S^1, S^2)]_+ > 0 \right\}.$$

PSRO_{rN} designs a learning protocol that encourages player 1 to amplify its strengths and ignore its weaknesses in finding a new policy. Importantly, it has been proved that if player 2 plays the Nash strategy, then such an Oracle guarantees to enlarge the gamescape. Nonetheless, focusing on the winners only can be problematic. Since weak agents may still hold the promise of tackling niche tasks, they can serve as stepping stones for discovering stronger policies in the future. For example, when training StarCraft AIs, overcoming agents’ weaknesses was found to be more critical than amplifying strengths (Vinyals et al., 2019a). Another counter-example

that fails PSRO_{rN} is the RPS-X game with $\mathbf{g} = \begin{bmatrix} 0 & -1 & 1 & -2/5 \\ 1 & 0 & -1 & -2/5 \\ -1 & 1 & 0 & -2/5 \\ 2/5 & 2/5 & 2/5 & 0 \end{bmatrix}$ (McAfee et al., 2020).

It is straightforward to see that if the initial strategy population of PSRO_{rN} starts from either {Rock}, {Paper}, or {Scissors}, then the algorithm will terminate without exploring the fourth strategy as the best response to the population of {R,P,S} is still in {R,P,S}; however, the unseen fourth strategy can exploit any meta-policies from the population of {R,P,S} by achieving a positive payoff of 2/5.

6.4 A New Measurement for Diversity

Instead of choosing between amplifying strengths or overcoming weaknesses, we take an altogether different approach to modelling the behavioural diversity in games. Specifically, we introduce a new diversity measure based on a geometric interpretation of meta-games modelled by a determinantal point process (DPP).

6.4.1 Determinantal Point Process

Originating in quantum physics for modelling repulsive Fermion particles (Kulesza et al., 2012; Macchi, 1977), a DPP is a probabilistic framework that characterises how likely a subset of diverse items is going to be sampled from a ground set where diverse subsets are preferred.

Definition 13 (Determinantal Point Process (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 \mathcal{L} that measures similarity for any pairs of items in \mathcal{Y} , let \mathbf{Y} be a random subset drawn according to \mathbb{P} , then we have, $\forall Y \subseteq \mathcal{Y}$, the probability of sampling Y is*

$$\text{DPP}(\mathcal{L}) := \mathbb{P}_{\mathcal{L}}(\mathbf{Y} = Y) \propto \det(\mathcal{L}_Y) = \text{Volume}^2 \left(\{\mathbf{w}_i\}_{i \in Y} \right), \quad (6.8)$$

where $\mathcal{L}_Y := [\mathcal{L}_{i,j}]_{i,j \in Y}$ denotes a submatrix of \mathcal{L} whose entries are indexed by the items included in Y . Given a PSD kernel $\mathcal{L} = \mathcal{W}\mathcal{W}^\top$, $\mathcal{W} \in \mathbb{R}^{M \times P}$, $P \leq M$, each row \mathbf{w}_i represents a P -dimensional feature vector of item $i \in \mathcal{Y}$, then the geometric

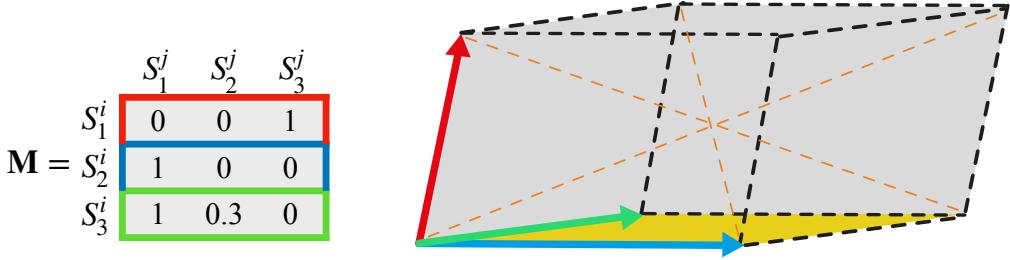


Figure 6.1: Game-DPP. The squared volume of the grey cube equals to $\det(\mathcal{L}_{\{S_1^i, S_2^i, S_3^i\}})$. Since S_2^i, S_3^i share similar payoff vectors, this leads to a smaller yellow area, and thus the probability of these two strategies co-occurring is low. The diversity (expected cardinality) of the population $\{S_1^i\}, \{S_1^i, S_2^i\}, \{S_1^i, S_2^i, S_3^i\}$ are 0, 1, 1.21 respectively.

meaning of $\det(\mathcal{L}_Y)$ is the squared volume of the parallelepiped spanned by the rows of \mathcal{W} that correspond to the sampled items in Y .

A PSD matrix ensures all principal minors of \mathcal{L} are non-negative (i.e., $\det(\mathcal{L}_Y) \geq 0, \forall Y \subseteq \mathcal{Y}$), which suffices to be a proper probability distribution. The normaliser of $\mathbb{P}_{\mathcal{L}}(Y = Y)$ can be computed by $\sum_{Y \subseteq \mathcal{Y}} \det(\mathcal{L}_Y) = \det(\mathcal{L} + \mathbf{I})$, where \mathbf{I} is the $M \times M$ identity matrix.

The entries of \mathcal{L} are pairwise inner products between item vectors. The kernel \mathcal{L} can intuitively be thought of as representing dual effects – the diagonal elements $\mathcal{L}_{i,i}$ aim to capture the quality of item i , whereas the off-diagonal elements $\mathcal{L}_{i,j}$ capture the similarity between the items i and j . A DPP models the **repulsive** connections among the items in a sampled subset. For example, in a two-item subset, since $\mathbb{P}_{\mathcal{L}}(\{i, j\}) \propto \begin{vmatrix} \mathcal{L}_{i,i} & \mathcal{L}_{i,j} \\ \mathcal{L}_{j,i} & \mathcal{L}_{j,j} \end{vmatrix} = \mathcal{L}_{i,i}\mathcal{L}_{j,j} - \mathcal{L}_{i,j}\mathcal{L}_{j,i}$, we know that if item i and item j are perfectly similar such that $\mathbf{w}_i = \mathbf{w}_j$ and thus $\mathcal{L}_{i,j} = \sqrt{\mathcal{L}_{i,i}\mathcal{L}_{j,j}}$, then these two items will not co-occur, hence such a subset of $Y = \{i, j\}$ will never be sampled.

6.4.2 Expected Cardinality

Our target is to find a population of diverse policies, with each of them performing differently from other policies due to their unique characteristics. Therefore, when modelling the behavioural diversity in games, we can naturally set the payoff matrix to be the kernel of a DPP so that the similarity between two policies depends on their

performance against the opponents in terms of payoffs.

Definition 14 (Game-DPP (G-DPP)). *A G-DPP for each player (see an example in Figure 6.1) is a DPP in which the ground set is the strategy population $\mathcal{Y} = \mathbb{S}$, and the kernel of the DPP is represented by the normalised payoff table M of a game, that is,*

$$\mathcal{L}_{\mathbb{S}} = MM^T. \quad (6.9)$$

For learning in open-ended games, we want to keep adding diverse policies to the population. This is equivalent to say, at each iteration, if we take a random sample from the G-DPP that consists of all existing policies, we hope the cardinality of such a random sample is large (since policies with similar payoff vectors will be unlikely to co-occur). In this sense, we design a diversity measure based on the expected cardinality of random samples from a G-DPP, i.e., $\mathbb{E}_{\mathbf{Y} \sim \mathbb{P}_{\mathcal{L}_{\mathbb{S}}}}[|\mathbf{Y}|]$. By the following lemma, we show that such a diversity measure can be computed in a tractable way.

Lemma 5 (Diversity of G-DPP). *The diversity of a G-DPP in terms of expected cardinality can be computed by*

$$\text{Diversity}(\mathbb{S}) = \mathbb{E}_{\mathbf{Y} \sim \mathbb{P}_{\mathcal{L}_{\mathbb{S}}}}[|\mathbf{Y}|] = \text{Tr}(\mathbf{I} - (\mathcal{L}_{\mathbb{S}} + \mathbf{I})^{-1}). \quad (6.10)$$

Proof. We can calculate the expected cardinality of a DPP sample by an exponential sum over all $2^{|\mathcal{Y}|}$ subsets of $\mathcal{Y} = \mathbb{S}$ as follows:

$$\begin{aligned} \mathbb{E}_{\mathbf{Y} \sim \mathbb{P}_{\mathcal{L}_{\mathbb{S}}}}[|\mathbf{Y}|] &= \sum_{\mathbf{Y} \subseteq \mathcal{Y}} |\mathbf{Y}| \mathbb{P}_{\mathcal{L}_{\mathcal{Y}}}(\mathbf{Y}) \\ &= \sum_{\mathbf{Y} \subseteq \mathcal{Y}} |\mathbf{Y}| \frac{\det(\mathcal{L}_{\mathbf{Y}})}{\det(\mathcal{L}_{\mathcal{Y}} + \mathbf{I})} \end{aligned}$$

Based on the following Lemma from [Rising \(2013\)](#),

Lemma 6 (Theorem 2.3.9 in [Rising \(2013\)](#)). *Let $\mathbf{Y} \sim \text{DPP}(\mathcal{K})$ where $\mathcal{K} = \mathbf{I} - (\mathcal{L} + \mathbf{I})^{-1}$, and let $\{\lambda_i\}_{i=1}^n$ be the eigenvalues of \mathcal{K} . Then $|\mathbf{Y}| = \sum_{i=1}^n Z_i$ where $\{Z_i\}_{i=1}^n$ are independent Bernoulli trials with $\mathbb{E}[Z_i] = \lambda_i$.*

we can write expectation over the random variable $|\mathbf{Y}|$ by

$$\mathbb{E}_{\mathbf{Y} \sim \mathbb{P}_{\mathcal{L}_S}}[|\mathbf{Y}|] = \mathbb{E}\left[\sum_{i=1}^n Z_i\right] = \sum_{i=1}^n \mathbb{E}[Z_i] = \sum_{i=1}^n \lambda_i = \text{Tr}(\mathcal{K}). \quad (6.11)$$

Since $\mathcal{K} = \mathcal{L}(\mathcal{L} + \mathbf{I})^{-1} = \mathbf{I} - (\mathcal{L} + \mathbf{I})^{-1}$, we can relate the eigenvalues of \mathcal{L} and \mathcal{K} as follows,

$$\lambda_n^{\mathcal{K}} = \frac{\lambda_n^{\mathcal{L}}}{\lambda_n^{\mathcal{L}} + 1}, \quad \forall n$$

where the superscript references which matrix the eigenvalues belong to. Finally, we have,

$$\mathbb{E}_{\mathbf{Y} \sim \mathbb{P}_{\mathcal{L}_S}}[|\mathbf{Y}|] = \sum_{i=1}^n \frac{\lambda_i^{\mathcal{L}}}{\lambda_i^{\mathcal{L}} + 1} = \text{Tr}(\mathbf{I} - (\mathcal{L} + \mathbf{I})^{-1}) \quad (6.12)$$

and we have the expected cardinality of a sample from our DPP based upon the kernel matrix \mathcal{L} . \square

An excellent property of our diversity measure is that it is well-defined even in the case when \mathcal{Y} has duplicated policies, as dealing with redundant policies turns out to be a critical challenge for game evaluation (Balduzzi et al., 2018b). In fact, redundancy also prevents us from directly using $\det(\mathcal{L}_S)$ as the diversity measure because the determinant value becomes zero with duplicated entries.

6.4.3 Expected Cardinality vs. Matrix Rank.

There is a fundamental difference between using expected cardinality and the matrix rank as the diversity measure. The rank of a matrix is the maximal number of linearly independent columns, though it can measure the *distinctiveness* between the columns, it cannot model the *diversity*. For example, in the RPS game, a strategy that plays [99% Rock, 1% Scissors] and a strategy that plays [98% Rock, 2% Scissors] are distinctive strategies since they have linearly independent payoff vectors against other strategies; however, they are not diverse as they both favour playing Rock. If one strategy is added into the population whilst the other already exists,

the rank of the payoff matrix will increase by one, but the increment on expected cardinality is minor. In the example of Figure 6.1, adding the green strategy only contributes to the expected cardinality by 0.21. This property is particularly important for modelling behavioural diversity in games, especially when it comes to the end of training as it becomes increasingly hard to find a diverse policy that can exploit a different aspect of opponents, finding a policy that has different payoff values from existing ones is still trivial. To conclude, we show by the following proposition that the expected cardinality of G-DPP is upper bounded by the rank of the payoff matrix, and the diversity only reaches the maxima when the payoff vectors are **orthogonal**.

Proposition 4 (The Upper Bound of Diversity). *For a population of policies \mathbb{S} and its payoff matrix M , the diversity of the population will be bounded by $\text{Diversity}(\mathbb{S}) \leq \text{rank}(M)$, and when M is normalised, we have $\text{Diversity}(\mathbb{S}) \leq \text{rank}(M)/2$. In both cases, the maximal diversity is only reached when the rows of M are orthogonal.*

Proof. Please note that an early version of this proposition is originally proved by [Slumbers \(2020\)](#) [Proposition 20] in a master project under my supervision.

We start from the case of un-normalised meta-game. We can always do the SVD decomposition of M and get

$$M = U\Sigma V^\top$$

Then, the kernel \mathcal{L} of G-DPP can be written as

$$\mathcal{L} = MM^\top = (U\Sigma V^\top)(U\Sigma V^\top)^\top = U\Sigma^2 U^\top$$

This means that the eigenvalues of \mathcal{L} are $\lambda_n = \sigma_n^2$ where σ_n are the entries of the diagonal of Σ . Thus, based on Lemma 5, we can write the expected cardinality

given kernel \mathcal{L} as

$$\mathbb{E}_{\mathbf{Y} \sim \mathbb{P}_{\mathcal{L}_S}} [|\mathbf{Y}|] = \sum_n \frac{\lambda_n}{\lambda_n + 1} = \sum_n \frac{\sigma_n^2}{\sigma_n^2 + 1} \quad (6.13)$$

A larger cardinality can only be achieved by either adding more eigenvalues or making the eigenvalues larger. Since we can only get a maximum of $n = \text{rank}(\mathbf{G})$ non-zero eigenvalues, making the eigenvalues larger will increase the terms of the summation by $\frac{\lambda_n}{\lambda_n + 1} < 1$ which means that

$$\sup \mathbb{E}_{\mathbf{Y} \sim \mathbb{P}_{\mathcal{L}_S}} [|\mathbf{Y}|] = \text{rank}(\mathbf{M}) \quad (6.14)$$

Thus, the maximum achievable diversity is obtained when population S makes M full rank and $\lambda_n \rightarrow \infty, \forall n$.

We now prove the case for a normalised meta-game. We first show that a DPP is maximised over orthogonal feature vectors. To do this, we show that the probability of sampling the whole set of agents is maximised for orthogonal vectors. Starting by noting Hadamard's Inequality, it states that if N is the matrix having columns v_i , then

$$\left| \det(N) \right| \leq \prod_{i=1}^n \|v_i\|, \quad (6.15)$$

where notably equality holds if and only if the vectors v_i are orthogonal. Therefore, by this inequality we also know that the determinant of M is maximised when the payoffs are orthogonal for each agent, as this allows the equality to hold.

Additionally, as we define our M to be a square matrix of size $M \times M$, we can decompose the determinant of the kernel \mathcal{L} as,

$$\det(\mathcal{L}_{\mathcal{Y}}) = \det(MM^\top) = \det(M)\det(M^\top) = \det(M)^2 \quad (6.16)$$

and therefore we know that $\det(\mathcal{L}_{\mathcal{Y}})$ is maximised whenever $\det(M)$ is maximised. By the following equation of the subset probability,

$$\mathbb{P}_{\mathcal{L}}(\mathbf{Y} = \mathcal{Y}) \propto \det(\mathcal{L}_{\mathcal{Y}}) \quad (6.17)$$

we have that the probability of sampling the whole ground set is maximised when the $\det(\mathcal{L}_{\mathcal{Y}})$ is maximised, which coincides with the orthogonality of M .

Finally, based on the fact that orthogonal feature vectors maximise a population's diversity, we show that a meta-game with orthogonal feature vectors will receive an expected cardinality value of $\frac{n}{2}$.

We know that the entries of our kernel \mathcal{L} are simply $L_{ij} = \mathbf{w}_i \mathbf{w}_j^\top$. As long as $\|\mathbf{w}_i\| = 1, \forall i$ it will be the case that $L_{ii} = 1$. Additionally, as we have stated that the rows of the meta-game are orthogonal, $\mathbf{w}_i \mathbf{w}_j^\top = 0, \forall i, j$, so all off-diagonal entries $L_{ij} = 0$. For example, the kernel \mathcal{L} for a meta-game of size 3 would be $\mathcal{L} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}$, which is obviously the identity matrix of size M. The identity matrix has M eigenvalues of 1, and therefore we have

$$\mathbb{E}_{\mathbf{Y} \sim \mathbb{P}_{\mathcal{L}_{\mathcal{S}}}(|\mathbf{Y}|)} = \sum_{i=1}^n \frac{\lambda_i^{\mathcal{L}}}{\lambda_i^{\mathcal{L}} + 1} = \sum_{i=1}^n \frac{1}{2} = \frac{n}{2}.$$

As the expected cardinality is maximised for orthogonal feature vectors, and we know that orthogonal feature vectors will return an expected cardinality of $\frac{n}{2}$, we know that when our feature vectors are normalised this is the max achievable expected cardinality. \square

6.4.4 Expected Cardinality vs. Effective Diversity.

The principle that underpins Eq. (6.7) and Eq. (6.10) are different. Here we illustrate their difference from the perspective of matrix norm. Notably, maximising the effective diversity in Eq. (6.7) is equivalent to maximising a matrix norm in that $ED(\mathcal{S}) = \frac{1}{2} \|\boldsymbol{\pi}^* \odot M \odot \boldsymbol{\pi}^*\|_{1,1}$ where \odot is Hadamard product and $\|\mathbf{A}\|_{1,1} := \sum_{i,j} |a_{ij}|$. In comparison, the proposition below shows that maximising the diversity in terms of expected cardinality is maximising the Frobenius norm of the meta-payoff M .

Proposition 5 (Matrix Norm). *Maximising the diversity in Eq. (6.10) also maximises the Frobenius norm of the payoff table M .*

Proof. Please note that an early version of this proposition is originally proved by [Slumbers \(2020\)](#) [Proposition 21] in a master project under my supervision.

Based on Proposition 4, the maximum achievable diversity is:

$$\sup \mathbb{E}_{\mathbf{Y} \sim \mathbb{P}_{\mathcal{L}_S}} [|\mathbf{Y}|] = \sup \sum_n \frac{\lambda_n}{\lambda_n + 1} = \text{rank}(\mathbf{M}) = n$$

Since the terms in the above summation are $0 \leq \frac{\lambda_n}{\lambda_n + 1} < 1$, the expected cardinality is also maximised for $\lambda_n \rightarrow \infty, \forall n$. Therefore, we have that the maximum achievable diversity is attained when the eigenvalues of \mathcal{L} are maximised and that $\text{rank}(\mathbf{M}) = n$. In G-DPP, we have that the kernel matrix is defined as $\mathcal{L} = \mathbf{M}\mathbf{M}^\top$, and the Frobenius norm can be written as

$$\begin{aligned} \|\mathbf{M}\|_F &= \sqrt{\sum_{i=1}^n \sum_{j=1}^n |M_{ij}|^2} = \sqrt{\text{Tr}(\mathbf{M}\mathbf{M}^\top)} \\ &= \sqrt{\text{Tr}(\mathcal{L})} = \sqrt{\sum_{i=1}^n \lambda_i} \end{aligned}$$

We have shown above that maximising the expected cardinality of \mathcal{L} is achieved by maximising the eigenvalues of \mathcal{L} and it directly follows that:

$$\sup \|\mathbf{M}\|_F = \sup \sqrt{\sum_{i=1}^n \lambda_i}$$

is achieved as $\lambda_i \rightarrow \infty, \forall i$. □

Proposition 6 (Matrix Norm). *The opposite direction of Proposition 5 is not correct, that is, simply maximising $\|\mathbf{M}\|_F$ will not necessarily lead to a large diversity of agent populations.*

Proof. We also allude to why maximising $\|\mathbf{M}\|_F$ will not necessarily lead to a large diversity of agents (i.e., the opposite direction of this proposition is not necessarily correct) and why our measure based upon expected cardinality will. Firstly, note that given

$$\|\mathbf{M}\|_F = \sqrt{\sum_{i=1}^n \lambda_i},$$

an optimiser may go about increasing this function in two manners. One manner may be to continually add new eigenvalues, which will improve the overall diversity

of M . However, an optimiser may also increase this norm by focusing on one eigenvalue, or a subset of eigenvalues, and increase those to be as large as possible. Our major worry here would be that by focusing on a subset of eigenvalues, new agents that are added to the population may be redundant as they are added to increase the current set of eigenvalues – not to bring in new non-zero eigenvalues. This would lead to a lack of diversity in the population. On the other hand, note the formula for expected cardinality is,

$$\mathbb{E}_{\mathbf{Y} \sim \mathbb{P}_{\mathcal{L}_S}} [|\mathbf{Y}|] = \sum_n \frac{\lambda_n}{\lambda_n + 1}$$

and note how each eigenvalue can at most contribute a value of 1 to the expected cardinality value. The importance of this is that an optimiser will gain less marginal benefit from increasing an already large eigenvalue than what it would gain from adding a new non-zero eigenvalue to the meta-game. Therefore, whilst we can guarantee that our expected cardinality measure will always search for new non-zero eigenvalues (and by that notion, more diverse agents), we can not guarantee the same for the Frobenius norm measure. \square

Geometrically, for a given matrix M , considering the box which is the image of a unit cube (in the 3D case) that is stretched by M , the Frobenius norm represents the sum of lengths of all diagonals in that box regardless of their directions (see the orange lines in Figure 6.1). Therefore, whilst the $\|\cdot\|_{1,1}$ norm reflects the idea that $\text{ED}(\mathbb{S})$ accounts for the winners within the Nash support only, the Frobenius norm considers all strategies equally. In later experiments, we show that this in fact results in a significant improvement in performance in games with non-transitive dynamics. Lastly, it is worth highlighting that the opposite direction of Proposition 5 is not correct, that is, maximising $\|M\|_F$ will **NOT** necessarily lead to large diversity. A counter-example in Figure 6.1 is that, if one of the orange lines is long, but the rest are short, though the Frobenius norm is large, the expected cardinality is still small.

6.5 Diversity Aware Learning Algorithms

With the newly proposed diversity measure of Eq. (6.10), we can now design diversity-aware learning algorithms. In this section, we incorporate the new di-

versity measurement into fictitious play, PSRO, and α -PSRO respectively, and turn them into diversity aware learning algorithms.

6.5.1 Diverse Fictitious Play

We start by extending the classical FP to a diverse version such that at each iteration, the player not only considers a best response, but also considers how this new strategy can help enrich the existing strategy pool after the update. Formally, our *diverse FP* method maintains the same update rule as Eq. (6.4), but with the best response changing into

$$\mathbf{BR}_\varepsilon^i(\boldsymbol{\pi}^{-i}) = \arg \max_{\boldsymbol{\pi} \in \Delta_{S^i}} \left[\mathbf{G}^i(\boldsymbol{\pi}, \boldsymbol{\pi}^{-i}) + \tau \cdot \text{Diversity}(\mathbb{S}^i \cup \{\boldsymbol{\pi}\}) \right] \quad (6.18)$$

where τ is a tunable constant. An excellent property of diverse FP is that the expected cardinality is guaranteed to be a strictly concave function; therefore, Eq. (6.18) has a unique solution at each iteration. We prove the following proposition.

Proposition 7. *The expected cardinality is a strictly concave function. The related best response in Eq. (6.18) has a unique optimiser.*

Proof. We study the sign of the second derivative of the diversity measure (Eq.(11)) in a neighbourhood of the positive semidefinite symmetric matrix \mathcal{L} . We apply a perturbation to \mathcal{L} such that $\tilde{\mathcal{L}} = \mathcal{L} + \varepsilon \mathbf{A}$ with \mathbf{A} being a symmetric matrix and $\varepsilon \in \mathbb{R}$, as a result, what we need to show is:

$$\frac{\partial^2}{\partial \varepsilon^2} \text{Tr}(\mathbf{I} - (\tilde{\mathcal{L}} + \mathbf{I})^{-1}) \Big|_{\varepsilon=0} < 0$$

First, we notice that

$$\mathbf{I} - (\tilde{\mathcal{L}} + \mathbf{I})^{-1} = \mathbf{I} - \sum_{n=0}^{\infty} (-1)^n (\tilde{\mathcal{L}})^n = - \sum_{n=1}^{\infty} (-1)^n (\tilde{\mathcal{L}})^n = \sum_{n=0}^{\infty} (-1)^n (\mathcal{L} + \varepsilon \mathbf{A})^{n+1}$$

where we used the matrix expansion $(\mathbf{I} + \tilde{\mathcal{L}})^{-1} = \sum_{n=0}^{\infty} (-1)^n \tilde{\mathcal{L}}^n$. We can always ensure $\|\tilde{\mathcal{L}}\|_F < 1$ by choosing ε small enough and redefining $\mathcal{L} := \frac{\mathcal{L}}{\|\mathcal{L}\|_F + \beta}$, $\beta > 0 \in \mathbb{R}$ if necessary. We note that after this modification \mathcal{L} re-

mains a legitimate matrix for a DPP due to being positive semidefinite, and do not affect the rankings of agents. Then,

$$\begin{aligned}
\frac{\partial^2}{\partial \varepsilon^2} (\mathbf{I} - (\tilde{\mathcal{L}} + \mathbf{I})^{-1}) \Big|_{\varepsilon=0} &= \frac{\partial^2}{\partial \varepsilon^2} \sum_{n=0}^{\infty} (-1)^n (\mathcal{L} + \varepsilon \mathbf{A})^{n+1} \Big|_{\varepsilon=0} \\
&= \sum_{n=0}^{\infty} (-1)^n \frac{\partial^2}{\partial \varepsilon^2} (\mathcal{L} + \varepsilon \mathbf{A})^{n+1} \Big|_{\varepsilon=0} \\
&= \frac{1}{2} \mathbf{A}^2 \sum_{n=0}^{\infty} (-1)^n (n+1)(n) \mathcal{L}^{n-1} + \frac{1}{2} \sum_{n=0}^{\infty} (-1)^n (n+1)(n) \mathcal{L}^{n-1} \mathbf{A}^2 \quad (6.19)
\end{aligned}$$

The series $\sum_{n=0}^{\infty} (-1)^n (n+1)(n) \mathcal{L}^{n-1}$ can be written as

$$\begin{aligned}
\sum_{n=0}^{\infty} (-1)^n (n+1)(n) \mathcal{L}^{n-1} &= \sum_{n=0}^{\infty} (-1)^{n+1} (n+2)(n+1) \mathcal{L}^n \\
&= - \sum_{n=0}^{\infty} (-1)^{n+2} \frac{\partial^2}{\partial \mathcal{L}^2} \mathcal{L}^{n+2} \\
&= - \frac{\partial^2}{\partial \mathcal{L}^2} \sum_{n=0}^{\infty} (-1)^{n+2} \mathcal{L}^{n+2} \\
&= - \frac{\partial^2}{\partial \mathcal{L}^2} (\mathcal{L}^2 - \mathcal{L}^3 + \mathcal{L}^4 - \dots) \\
&= - \frac{\partial^2}{\partial \mathcal{L}^2} ((\mathbf{I} + \mathcal{L})^{-1} + \mathcal{L} - \mathbf{I}) \\
&= -2(\mathbf{I} + \mathcal{L})^{-3}
\end{aligned}$$

where we used the matrix expansion $(\mathbf{I} + \mathcal{L})^{-1} = \sum_{n=0}^{\infty} (-1)^n \mathcal{L}^n$ which holds under the modifications specified before. Finally, we obtain that

$$\begin{aligned}
\frac{\partial^2}{\partial \varepsilon^2} \text{Tr} (\mathbf{I} - (\tilde{\mathcal{L}} + \mathbf{I})^{-1}) \Big|_{\varepsilon=0} &= \text{Tr} \left(\frac{\partial^2}{\partial \varepsilon^2} (\mathbf{I} - (\tilde{\mathcal{L}} + \mathbf{I})^{-1}) \right) \Big|_{\varepsilon=0} \\
&= - \text{Tr} (\mathbf{A}^2 (\mathbf{I} + \mathcal{L})^{-3} + (\mathbf{I} + \mathcal{L})^{-3} \mathbf{A}^2) \\
&< 0
\end{aligned}$$

where the last inequality comes from the fact that both $(\mathbf{I} + \mathcal{L})^{-3}$ and \mathbf{A}^2 are positive definite and the trace of the product of two positive definite matrices is positive. We therefore proved that the expected cardinality is a strictly concave function. \square

Intuitively, the diverse FP process will almost surely converge to a GWFP process as long as $\tau \rightarrow 0$ and thus will enjoy the same convergence guarantees as GWFP (e.g., a NE in two-player zero-sum or potential games). However, in order to prove such connection rigorously, we need to show that the sequence of expected changes in strategy, which is induced by finding a strategy that maximises the expected cardinality at each iteration, is actually a uniformly bounded martingale sequence that satisfies Eq. (6.5). Formally, we prove the following theorem.

Theorem 5 (Convergence of Diverse FP). *The perturbation sequence induced by a diverse FP process is a uniformly bounded martingale difference sequence; therefore, diverse FP shares the same convergence properties as GWFP, e.g., on two-player zero-sum games.*

Proof. We begin by proving that our perturbation sequence is a martingale sequence w.r.t. the normalised expected cardinality,

$$X_n = \frac{\mathbb{E}_{\mathbf{Y} \sim \mathbb{P}_{\mathcal{L}_S}} [|\mathbf{Y}|]}{|\mathcal{Y}|}$$

Based on Proposition 4, we know $0 \leq X_n \leq 0.5$. We are to show that $S_n = \frac{1}{n} \sum_{i=1}^n [X_i - \mu]$ where $\mu = \mathbb{E}[X_n]$ is a martingale sequence by proving that it meets the three conditions.

- (i). S_n is a measurable function as it is a partial sum of $\{X_i\}_{i=1}^\infty$.
- (ii). $\mathbb{E}[|S_n|] < \infty, \forall n$ because $\mathbb{E}[|S_n|] \leq \frac{1}{n} \sum_{i=1}^n \mathbb{E}[|X_i - \mu|] < \infty$
- (iii). $\mathbb{E}[S_{n+1}|X_1, \dots, X_n] = S_n$

$$\begin{aligned} \mathbb{E}[S_{n+1}|X_1, \dots, X_n] &= \mathbb{E}[S_n + \frac{1}{n}X_{n+1} - \frac{1}{n}\mu] \\ &= S_n + \mathbb{E}[\frac{1}{n}X_{n+1} - \frac{1}{n}\mu | X_1, \dots, X_n] \\ &= S_n - \frac{1}{n}\mu + \mathbb{E}[\frac{1}{n}X_{n+1} | X_1, \dots, X_n] \\ &= S_n - \frac{1}{n}\mu + \frac{1}{n}\mu \\ &= S_n \end{aligned}$$

and therefore we have that $\{S_k\}_{k=1}^{\infty}$ is a martingale sequence w.r.t. $\{X_k\}_{k=1}^{\infty}$. Now we let $M_k = S_k - S_{k-1}, k = 2, 3, \dots$ and show that this is a martingale **difference** sequence under the same three conditions.

- (i). M_k is measurable as both S_k and S_{k-1} are.
- (ii). $\mathbb{E}[|M_k|] < \infty, \forall k$ because $\mathbb{E}[|M_k|] \leq \mathbb{E}[|S_k|] + \mathbb{E}[|S_{k-1}|] < \infty$
- (iii). $\mathbb{E}[M_{k+1}|X_1, \dots, X_n] = 0$

$$\begin{aligned}\mathbb{E}[M_{k+1}|X_1, \dots, X_n] &= \mathbb{E}[S_{k+1} - S_k | X_1, \dots, X_n] \\ &= \mathbb{E}[S_{k+1} | X_1, \dots, X_n] - S_k \\ &= S_k - S_k \\ &= 0\end{aligned}$$

and we have shown that $\{M_k\}_{k=1}^{\infty}$ is a martingale difference sequence w.r.t. $\{X_k\}_{k=1}^{\infty}$. Next, we show that this martingale sequence is bounded uniformly in $L2$. M_n is said to be bounded in $L2$ if $\sup_n[M_n^2] < \infty$, which can be shown due to the following:

$$\begin{aligned}\sup_n[M_n^2] &= \sup_n \left[\frac{1}{n} \sum_{i=1}^n [X_i - \mu]^2 \right] < \infty \\ \text{as } \frac{1}{n} \sum_{i=1}^n [X_i - \mu]^2 &\leq 0.25, \forall n\end{aligned}$$

Therefore, if $M_n \in L2$, then the martingale sequence M is bounded in $L2$ if and only if $\sum_{k \geq 1} \mathbb{E}[(M_k - M_{k-1})^2] < +\infty$, which we can show in the following manner:

$$\begin{aligned}\sum_{k \geq 1} \mathbb{E}[(M_k - M_{k-1})^2] &= \sum_{k \geq 1} [\mathbb{E}[M_k^2] - 2\mathbb{E}[M_k M_{k-1}] + \mathbb{E}[M_{k-1}^2]] \\ &= \sum_{k \geq 1} [\mathbb{E}[M_k^2] + \mathbb{E}[M_{k-1}^2] - \mathbb{E}[2M_k M_{k-1}]] \\ &= \sum_{k \geq 1} [\mathbb{E}[M_k^2] + \mathbb{E}[M_{k-1}^2]] \\ &< \infty, \quad \text{as } M_k \in L2, \forall k.\end{aligned}$$

Finally, as we satisfy both of the conditions of GWFP, namely that our expected cardinality function is strictly concave as shown in Proposition 7 and the perturbation meets the condition of Eq. (6.5) we have that as $\tau \rightarrow 0$ our diverse FP process will almost surely result in a GWFP process, which is known to converge in two-player zero-sum games and potential games (Leslie and Collins, 2006). \square

6.5.2 Diverse Policy Space Response Oracle

When solving NFGs, the total number of strategies is known, and thus a best response in Eq. (6.18) can be computed through a direct search, and the uniqueness of the solution is guaranteed by Proposition 7. When it comes to solving open-ended meta-games, the total number of policies is unknown and often infinitely many. Therefore, a best response must be computed through optimisation subroutines such as gradient-based methods or RL algorithms. In this section, we extend our diversity measure to the policy space and develop diversity-aware solvers for open-ended meta-games.

In the general solver for meta-games in Algorithm 9, at the t -th iteration, the algorithm maintains a collection of policies \mathbb{S}_t^i that have been learned so far by player i . Our goal here is to design an Oracle to train a new strategy $S_{\boldsymbol{\theta}}$, parameterised by $\boldsymbol{\theta} \in \mathbb{R}^d$ (e.g., a deep neural net), which on one hand maximises player i 's payoff, and, on the other hand is diverse from all strategies in the current population \mathbb{S}_t^i . Therefore, we define the ground set of the G-DPP at iteration t to be the union of the existing population \mathbb{S}_t^i and the new model that we are going to add, that is,

$$\mathcal{Y}_t = \mathbb{S}_t^i \cup \{S_{\boldsymbol{\theta}}\}.$$

With the proper ground set at each iteration, we can have the diversity measure in terms of expected cardinality. As a result, in symmetric zero-sum meta-games, the objective of Oracles can be written as

$$\mathcal{O}^1(\boldsymbol{\pi}^2) = \arg \max_{\boldsymbol{\theta} \in \mathbb{R}^d} \sum_{S^2 \in \mathbb{S}^2} \boldsymbol{\pi}^2(S^2) \cdot \phi(S_{\boldsymbol{\theta}}, S^2) + \tau \cdot \text{Diversity}(\mathbb{S}^1 \cup \{S_{\boldsymbol{\theta}}\}) \quad (6.20)$$

where $\pi^2(\cdot)$ is the meta-policy of player two, and depending on the meta-game solvers, it can represent NE, UNIFORM, etc.

Based on Eq. (6.20), we can tell that the diversity of policies during training is expected to come from two sources. The obvious aspect is from the expected cardinality of G-DPP that forces agents to find new solutions that are different from existing policies against a fixed opponent. The less obvious aspect is from how the opponents are treated. Although the meta-policy of player 2 is determined by $\pi^2(\cdot)$, the learning player will have to focus on exploiting certain diverse aspects of $\pi^2(\cdot)$ in order to acquire new diverse skills. This is similar in manner to selecting a diverse set of opponents. In fact, we are able to show that our diversity-driven oracle can strictly enlarge the gamescape defined in Eq. (6.3). Importantly, unlike PSRO_{rN} (Balduzzi et al., 2019), here we do not need to assume the opponents are playing only a Nash strategy. Formally, we have:

Proposition 8 (Gamescape Enlargement). *Training a new strategy S_θ via Eq. (6.20) strictly enlarges the gamescape, i.e.,*

$$\text{Gamescape}(\mathbb{S}) \subsetneq \text{Gamescape}(\mathbb{S} \cup \{S_\theta\}).$$

Proof. According to Proposition 4 and Proposition 7, maximising the expected cardinality of the DPP implies there exists a unique solution that makes the meta-game payoff table M full rank and increases the eigenvalues. If M is full rank, this means that the new row in the meta-game M corresponding to the new policy S_θ must be linearly independent to the other rows and thus it must not be a convex combination of the other rows; as a result, the gamescape is strictly enlarged. Such a result is also expected because the diversity in terms of expected cardinality promotes orthogonality, which is a stronger property than linear independency. \square

6.5.3 Implementation of Oracles.

Optimising Eq. (6.20) is more challenging than Eq. (6.18). Except for cases when ϕ is differentiable and we can directly apply gradient-based methods to solve Eq. (6.20), in general, we have to seek for gradient-free solutions or RL algorithms

since for many real-world games, the elements in M are not differentiable w.r.t θ . To tackle this, we provide zero-order methods and RL-based approaches as approximation solutions to Eq. (6.20) and list the pseudocode of those Oracles in Algorithm 10 and Algorithm 11.

The code can be found in: https://github.com/diversepsro/diverse_psro

Algorithm 10 Diverse Best Response Oracle

```

1: Inputs:
2: Populations  $\mathbb{S}_t = \prod_{i \in \mathcal{N}} \mathbb{S}_t^i$ , with  $S_t^i \in \mathbb{S}_t^i$  parametrised by  $\theta_{S_t^i}$ 
3: Metapolicies  $\boldsymbol{\pi}_t = \prod_{i \in \mathcal{N}} \boldsymbol{\pi}_t^i$ 
4: Learning rate  $\mu$ 
5: Diversity probability  $\lambda$ 
6:
7: function oracle( $\mathbb{S}_t, \boldsymbol{\pi}_t, \mu, \lambda$ )
8:   Compute  $BR_{qual} = BR^i(\mathbb{S}_t^{-i}, \boldsymbol{\pi}_t^{-i})$ 
9:   for each pure strategy  $P_j$  do:
10:    Update meta-payoff  $M_j = M(\mathbb{S}_t^i \cup \{P_j\})$ 
11:    Compute  $BR_{div} = \arg \max_{P_j} \left( \text{Tr}(\mathbf{I} - (M_j M_j^\top + \mathbf{I})^{-1}) \right)$ .
12:    Choose  $BR = BR_{div}$  with probability  $\lambda$  else  $BR = BR_{qual}$ 
13:    Update  $\theta_{S_t^i} = \mu \theta_{S_t^i} + (1 - \mu) \theta_{BR}$ 
14: Return:  $S_t^i$ 

```

Algorithm 11 Diverse Gradient Ascent Oracle

```

1: Inputs:
2: Populations  $\mathbb{S}_t = \prod_{i \in \mathcal{N}} \mathbb{S}_t^i$ , with  $S_t^i \in \mathbb{S}_t^i$  parametrised by  $\theta_{S_t^i}$ 
3: Metapolicies  $\boldsymbol{\pi}_t = \prod_{i \in \mathcal{N}} \boldsymbol{\pi}_t^i$ 
4: Number of training updates  $N_{train}$ 
5: Diversity weight  $\lambda$ 
6:
7: function oracle( $\mathbb{S}_t, \boldsymbol{\pi}_t, N_{train}, \lambda$ )
8:   Choose  $S^{train} \in \mathbb{S}_t^i$  with probability  $\boldsymbol{\pi}_t^i$ 
9:   for  $j = 1, \dots, N_{train}$ :
10:    Compute payoff  $p_j$  of  $S^{train}$ 
11:    Compute meta-payoff  $M_j = M(\mathbb{S}_t^i \cup \{S^{train}\})$ 
12:    Compute diversity  $d_j = \text{Tr}(\mathbf{I} - (M_j M_j^\top + \mathbf{I})^{-1})$ 
13:    Compute loss  $l_j = -(1 - \lambda)p_j - \lambda d_j$ 
14:    Update  $\theta_{S^{train}}$  to minimise  $l_j$  using SGD or ADAM
15: Return:  $S^{train}$ 

```

Algorithm 12 Diverse Zero-order Oracle

```

1: Inputs:
2: Populations  $\mathbb{S}_t = \prod_{i \in \mathcal{N}} \mathbb{S}_t^i$ , with  $S_t^i \in \mathbb{S}_t^i$  parametrised by  $\theta_{S_t^i}$ 
3: Metapolicies  $\boldsymbol{\pi}_t = \prod_{i \in \mathcal{N}} \boldsymbol{\pi}_t^i$ 
4: Learning rate  $\mu$ 
5: Noise parameter  $\sigma$ 
6: Diversity weight  $\lambda$ 
7:
8: function oracle( $\mathbb{S}_t, \boldsymbol{\pi}_t, \mu, \sigma, \lambda$ )
9:   Sample a single  $S_t^i \in \mathbb{S}_t^i$  with probability  $\boldsymbol{\pi}_t^i$ 
10:  for perturbation  $j \in 1, 2, \dots$  do:
11:     $S_j = \text{RandomPerturbation}(S_t^i, \mu, \sigma)$ 
12:    Update meta-payoff  $M_j = M(\mathbb{S}_t^i \cup \{S_j\})$ 
13:    Compute payoff  $p_j$  of  $S_j$ 
14:    Compute diversity  $d_j = \text{Tr}(\mathbf{I} - (M_j M_j^\top + \mathbf{I})^{-1})$ .
15:  Return:  $\arg \max_{S_j} (1 - \lambda) p_j + \lambda d_j$ 
16: function RandomPerturbation( $S_t^i, \mu, \sigma$ )
17:   Generate noise  $\varepsilon \sim \text{Lognormal}(0, \sigma^2)$ 
18:   Mutate  $\theta_{S_t^i} = \mu \theta_{S_t^i} + (1 - \mu) \varepsilon$ 
19: Return  $S_t^i$ 

```

6.5.4 Diverse Oracle for α -Rank

We can additionally design diverse oracles that are compatible with α -Rank (Muller et al., 2019). Note that α -Rank is a replacement solution concept for NE on multi-player general-sum games; therefore, the goal of learning algorithms that use α -Rank as the meta-game solver, such as α -PSRO (Muller et al., 2019), is no longer converging to NE, but rather finding all SSCCs on the response graph. Since a naive best response fails to converge to SSCCs, a diverse Oracle suggested by Eq. (6.20) cannot be applied, instead, we have to account for the behavioural diversity based on the Preference-based Best Response (PBR) in Eq. (6.6). We conduct a quality-diversity kernel decomposition to unify PBR with our diversity measure. Specifically, given a DPP kernel $\mathcal{L} = \mathcal{W}\mathcal{W}^\top$, we can rewrite the i -th row of \mathcal{W} to be the product of both a quality term and diversity features. We denote the quality term as $q_i \in \mathbb{R}^+$ and the diversity feature vector as $\mathbf{w}_i \in \mathbb{R}^P$. The entries of \mathcal{L} are thus $\mathcal{L}_{ij} = q_i \mathbf{w}_i \mathbf{w}_j^\top q_j$, and the DPP measure is written as $\mathbb{P}_{\mathcal{L}}(Y) \propto (\prod_{i \in Y} q_i^2) \det(\mathcal{W}_Y)$. To design diverse Oracles that fit α -Rank, we define the quality term of each strategy to be the exponent of the PBR value, and the diversity features \mathbf{w}_i to be the

corresponding rows in the normalised payoff table \mathbf{M} , so that for each strategy, i.e.,

$$q_i = \exp\left(\sum_{S^{-i} \sim \boldsymbol{\pi}^{-i}} [\mathbf{1}[M^i(\sigma, S^{-i}) > M^i(S^i, S^{-i})]]\right), \mathbf{w}_i = \frac{\mathbf{M}_{[i,:]}}{\|\mathbf{M}\|_F} \quad (6.21)$$

The payoff vectors have to be normalised (i.e., $\|\mathbf{w}_i\| = 1$) so that feature vectors have no influence over the quality term. Finally, the diversity-aware Oracle that suits α -Rank is written as

$$\mathcal{O}_t^i(\boldsymbol{\pi}^{-i}) = \arg \max_{\boldsymbol{\pi} \in \Delta_{\mathbb{S}_t^i}} \text{Tr}\left(\mathbf{I} - (\mathcal{L}_{\mathbb{S}_t^i \cup \{\boldsymbol{\pi}\}} + \mathbf{I})^{-1}\right). \quad (6.22)$$

We are able to show the convergence results of our diverse Oracle with α -Rank on two-player symmetric NFGs. We first define the precise definition of convergence to SSCCs on the response graph, and then present the main theorem.

Theorem 6 (Convergence of Diverse α -PSRO). *Diverse α -PSRO with the Oracle of Eq. (6.22) converges to the sub-cycle of the unique SSCC in the two-player symmetric games.*

Proof. Please note that an early version of this proposition is originally proved by [Slumbers \(2020\)](#) [Proposition 25] in a master project under my supervision.

Intuitively, our proof follows the same argument as [Muller et al. \(2019\)](#) [Proposition 3], which can be adapted in the context of our diverse PBR oracle in Eq. (14). The uniqueness of the SSCC follows from the fact that in the single-population case, the response graph is fully-connected. Suppose at termination of α -DPP, the α -population contains no strategy within the SSCC, and let s be a strategy in the α -DPP population. We claim that s attains a higher value for the objective defining the α -DPP oracle than any strategy in the α -DPP population, which contradicts the fact that α -DPP has terminated. By virtue of being in the SSCC we have that $\mathbf{M}^1(s, s') > \mathbf{M}^2(s', s)$ for all s' outside the SSCC, and in particular for all $s' \in \mathbb{S}_t^i$, thus the PBR objective for s is 1.

If a member of the underlying game's SSCC appears in the α -DPP population, this member will induce its own meta-SSCC in the meta game's response graph

which will have positive probability under the α -Rank distribution for the meta-game, and the DPP oracle for this meta-SSCC will always return a member of the underlying game’s SSCC. This is because the only strategies that receive a non-zero quality score are those that are members of the underlying game’s SSCC, and we ignore strategies with a quality of zero. If the only strategy that returns a non-zero quality score, which must be a member of the SSCC, is already in the population, the corresponding meta-SSCC already contains a cycle of the underlying SSCC. Note that if the meta-SSCC does not contain a cycle, it must be a singleton. Either this singleton is equal to the full SSCC of the underlying game (in which we have α -fully converged), or it is not, in which case the DPP oracle must return a new strategy from the underlying SSCC, contradicting our previous assumption. \square

6.6 Experiments and Results

We compare our diversity-aware solvers with state-of-the-art baseline algorithms including self-play, PSRO (Balduzzi et al., 2019), pipeline PSRO (a parallel implementation of PSRO) (McAleer et al., 2020), rectified PSRO (Balduzzi et al., 2019), and α -PSRO (Muller et al., 2019). We investigate the performance of these algorithms on both NFGs where the total number of strategies is known and thus a best response is computed by direct search, and meta-games where the total number of policies is infinite and a best response needs computing by optimising subroutines.

Our selected games are challenging in the sense that they involve both transitive and non-transitive dynamics. If an algorithm fails to discover a diverse set of policies, it will be trapped in some localised strategy cycles which makes it easily exploitable (similar to the RPS-X example in Section 6.3.4). Therefore, we focus on the evaluation metrics of exploitability in Eq. (6.1) and how extensively the gamescapes are explored. One exception is the comparison between α -PSRO and diverse α -PSRO, since the solution concept is no longer NE, instead of exploitability, we use the metric of

$$\text{PCS-Score} = \frac{\text{\# of underlying SSCC strategies found in } \alpha\text{-PSRO population}}{\text{\# of underlying SSCC strategies}}$$

Table 6.2: Hyper-parameter settings for our diverse-PSRO method *vs.* other baselines on four experiments.

SETTINGS	VALUE	DESCRIPTION
RANDOM GAMES OF SKILL		
ORACLE METHOD	DIVERSE BEST RESPONSE	SUBROUTINE OF GETTING ORACLES
LEARNING RATE	0.5	LEARNING RATE FOR AGENTS
IMPROVEMENT THRESHOLD	0.03	CONVERGENCE CRITERIA
METASOLVER	FICTITIOUS PLAY	METASOLVER METHOD
METASOLVER ITERATIONS	1000	# ITERATIONS FOR METASOLVER
# OF THREADS IN PIPELINE	2	# LEARNERS IN PIPELINE PSRO
# OF SEEDS	10	# TRIALS
DPP WEIGHTING	0.15	WEIGHT OF THE DPP BEST RESPONSE
2D ROCK PAPER SCISSORS		
ORACLE	GRADIENT ASCENT	SUBROUTINE OF GETTING ORACLES
OPTIMIZER	ADAM	GRADIENT ASCENT OPTIMIZER
LEARNING RATE	0.1	LEARNING RATE FOR OPTIMIZER
BETAS	(0.9, 0.99)	BETAS PARAMETER FOR OPTIMIZER
π^i	$\pi_k^i = e^{(-(x_i - \mu_k)^\top \Sigma (x_i - \mu_k)/2)}$	STRATEGY FROM 2D COORDINATES
Σ	0.5I	COVARIANCE MATRIX FOR GAUSSIANS
μ_1	(2.872, -0.025)	POSITION OF THE FIRST GAUSSIAN
μ_2	(-1.458, -2.475)	POSITION OF THE SECOND GAUSSIAN
μ_3	(-1.414, -2.500)	POSITION OF THE THIRD GAUSSIAN
STRATEGY INITIALISATION	0.1	UNIFORM DISTRIBUTION
METASOLVER	FICTITIOUS PLAY	METASOLVER METHOD
METASOLVER ITERATIONS	1000	# ITERATIONS FOR METASOLVER
ITERATIONS	50	# TRAINING ITERATIONS
DPP WEIGHT	$\frac{0.7}{1+e^{(-0.25(t-25))}}$	WEIGHT OF THE DPP BEST RESPONSE
# OF THREADS IN PIPELINE	4	# LEARNERS IN PIPELINE PSRO
# OF SEEDS	10	# OF TRIALS
COLONEL BLOOTTO		
ORACLE METHOD	ZERO-ORDER	SUBROUTINE OF GETTING ORACLES
ORACLE ITERATIONS	50	NEW STRATEGIES PER ITERATION
μ	0.1	LEARNING RATE
METASOLVER	LINEAR PROGRAMMING	METASOLVER METHOD
# OF SEEDS	10	# OF TRIALS
DPP WEIGHT	0.15	WEIGHT OF THE DPP BEST RESPONSE
# OF AREAS	3	# OF AREAS TO DISTRIBUTE COINS
# OF COINS	10	# OF COINS TO DISTRIBUTE
α-PSRO		
ORACLE METHOD	PBR / DPP-PBR	SUBROUTINE OF GETTING ORACLES
METASOLVER	α -RANK	METASOLVER METHOD
ITERATIONS	50	# OF TRAINING ITERATIONS
# OF SEEDS	20	# OF TRIALS
α	INFINITE	THE α IN α -RANK

for fair comparison (Muller et al., 2019). Finally, the hyper-parameter settings for each experiment are listed in Table 6.2.

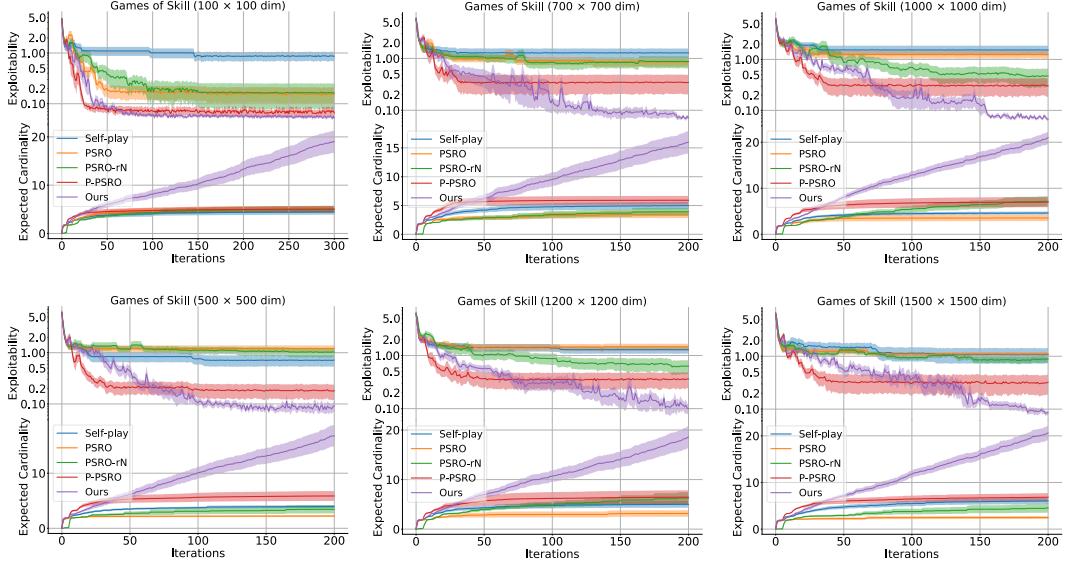


Figure 6.2: Results on Games of Skill with payoff matrices sizes ranging from 100×100 to 1500×1500 . For each subplot, the upper half shows the exploitability versus the population size (i.e., the PSRO iteration), and lower half shows the diversity in terms of expected cardinality.

6.6.1 Random Games of Skill

The payoff values of Games of Skill (Czarnecki et al., 2020) are defined by

$$\mathbf{G}_{i,j} := \frac{1}{2} (W_{i,j} - W_{j,i}) + S_i - S_j \text{ with } W_{i,j}, S_i \stackrel{i.i.d.}{\sim} \mathcal{N}(0, \sigma_W^2 \text{ or } \sigma_S^2).$$

The intuition behind Games of Skill is that S_i represents the transitive strength (e.g., ranking/Elo score) of a strategy π_i , and $W_{i,j}$ encodes all possible interactions of π_i against π_j , which often includes non-transitive cycles. Many real-world games exhibit the Game of Skill geometry in terms of payoffs (Czarnecki et al., 2020). We used the Diverse Best Response Oracle (see Algorithm 10)

In Figure 6.2, we report the results on 10 randomised games, each with different number of strategies ranging from 100×100 to 1500×1500 . The results show that our method outperforms all baselines in terms of both exploitability and diversity. While the baselines have saturated after 50 iterations, our method keeps finding novel effective strategies, leading to almost zero exploitability. Additionally, we have also observed consistent results on games with either larger or smaller dimensions, see Figure 6.2.

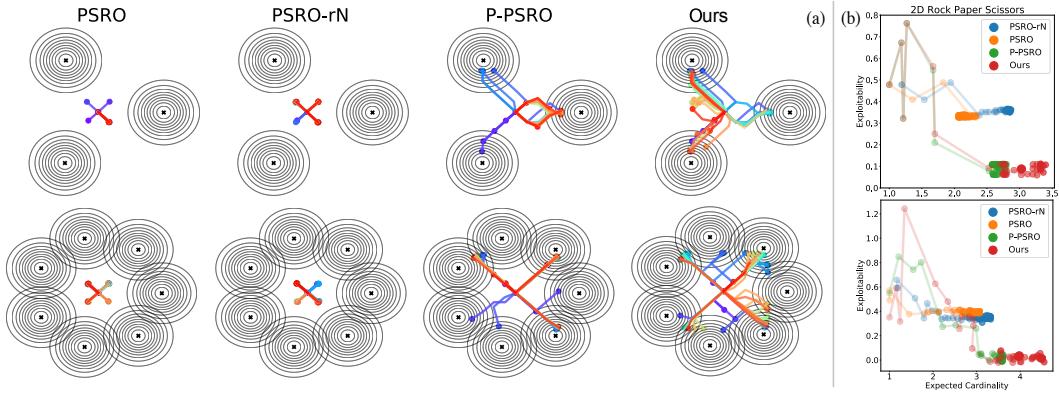


Figure 6.3: 2D-Rock Paper Scissors. a): Exploration trajectories during training. b): Performance vs. Diversity comparisons.

6.6.2 2D-Rock Paper Scissors

In our 2D-RPS game⁴, R/P/S are represented by three equally-distanced Gaussian humps on a 2D plane. Each strategy corresponds to a tuple of 2D coordinates, which can be translated into the weights on R/P/S by measuring the likelihood under each Gaussian. Unlike classical RPS, 2D-RPS contains transitive dynamics, for example, a strong Rock that is closer to the centre of Rock hump can beat a weak Rock that is relatively far away. Precisely, the payoff matrix of 2D-RPS is given by

$$M_{i,j} = \boldsymbol{\pi}^{i,\top} \begin{bmatrix} 0 & -1 & 1 \\ 1 & 0 & -1 \\ -1 & 1 & 0 \end{bmatrix} \boldsymbol{\pi}^j + \frac{1}{2} \sum_{k=1}^3 (\boldsymbol{\pi}_k^i - \boldsymbol{\pi}_k^j). \quad (6.23)$$

Note that since there is an infinite number of points on the 2D plane, 2D-RPS is an open-ended meta-game. Players' coordinates are initialised uniformly. They must learn to *climb* up the Gaussians to maximise the transitive payoff whilst also exploring all three Gaussians to stay un-exploitable. We used the Gradient Ascent Oracle (see Algorithm 11). Figure 6.3 shows the exploration trajectories for different algorithms along with the plot of performance vs. diversity. Results suggest that both PSRO and PSRO_{rN} fail to complete the task, whilst DPP-PSRO manages to

⁴The design of the 2D-RPS game, though inspired by multiple sources including the Disc game in [Balduzzi et al. \(2019\)](#) and Gaussian-RPS game in an unpublished work named “Pick your battles: interaction graphs as population-level objectives for strategic diversity. author unknown”, is however novel in terms of combining both transitive components and intransitive components in the RPS game, which leads to Eq. 6.23

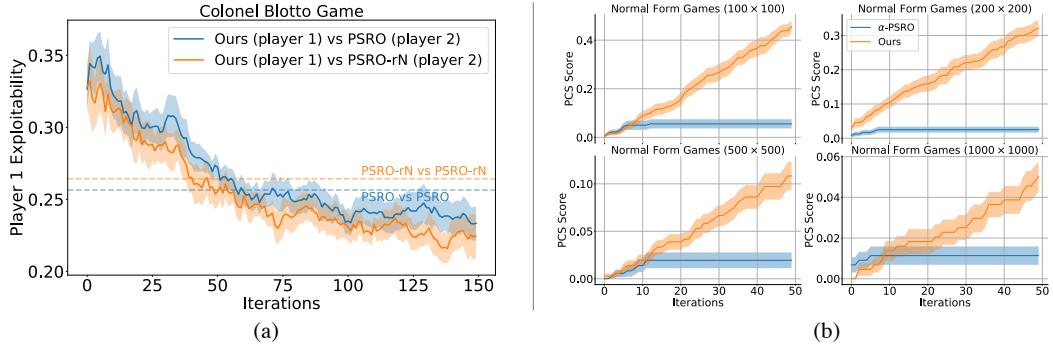


Figure 6.4: Results of a) exploitability on the Blotto Game , b) PCS-Score against α -PSRO on NFGs

solve the task almost perfectly by generating a population of diverse and effective strategies. Consistent results can be observed when we extend the 2D-RPS game to include seven Gaussian humps.

6.6.3 Colonel Blotto

Blotto is a classical resource allocation game that is widely analysed for election campaigns (Roberson, 2006). In this game, two players have a budget of coins which they simultaneously distribute over a fixed number of areas. An area is won by the player who puts the most coins, and the player that wins the most areas wins the game. We report the game results with 3 areas and 10 coins over 10 games. Here, we used the Diverse Zero-Order Oracle (see Algorithm 12). We test how a DPP-PSRO player performs in terms of exploitability against a PSRO and a $PSRO_{rN}$ player respectively. We benchmark the performance against the cases of PSRO vs PSRO and $PSRO_{rN}$ vs $PSRO_{rN}$. Figure 4a shows that our method consistently exploits PSRO and $PSRO_{rN}$ methods.

6.6.4 Diverse α -PSRO

As the PBR in Eq. (6.6) requires looping through all strategies, we test our diverse α -PSRO method on randomly generated zero-sum NFGs with varying dimensions. As our evaluation metric we measure the PCS-Score – the number of underlying SSCC elements that have been found in the population – at every iteration, and compare this to original α -PSRO. We note that we do not employ the *novelty-bound* suggested in Muller et al. (2019) to illustrate how the original α -PSRO has a strong

cyclic nature, which stops it from finding even a few underlying SSCC elements. Results in Figure 6.4b suggest that introducing diversity into α -PSRO can effectively prevent the learner from exploring the same strategic cycles during training, and is therefore able to find SSCCs and continuously improve the PCS-Score.

6.7 Chapter Summary

In this chapter, I look into the many-agent learning problems in open-ended meta-games. I have summarised existing meta-game solvers, and highlighted the necessity to promote behavioural diversity during the learning process. Specifically, I offer a geometric interpretation of behavioural diversity for learning in meta-games by introducing a new diversity measure built upon the expected cardinality of a determinantal point process. Based on the diversity measure, I propose diversity-aware general solvers for open-ended meta-games. I prove the convergence of our methods, and show that they are guaranteed to expand the gamescape in training. On Games of Skill, 2D-RPS, and Colonel Blotto, our methods outperform strong state-of-the-art baselines by a large margin including pipeline PSRO ([McAleer et al., 2020](#)), PSRO_{rN} ([Balduzzi et al., 2019](#)), and α -PSRO ([Muller et al., 2019](#)).

Chapter 7

Modelling Diverse Interactions in Autonomous Driving

Multi-agent RL has achieved a remarkable amount of success in solving various types of video games. A cornerstone of this success is the auto-curriculum framework, which essentially shapes the learning process by continually creating new challenging tasks for agents to adapt to, thereby facilitating the acquisition of new skills. In this chapter, I extend MARL methods to real-world domains, outside of purely video games, and introduce one novel application, which is about autonomous driving (AD).

Multi-agent interaction is a fundamental aspect of AD in the real world. Despite more than a decade of research and development, the problem of how to competently interact with diverse road users in diverse scenarios remains mostly unsolved. Existing solutions to modelling interactions mainly use rule-based methods, for example, to maintain a hard-coded minimum safe distance between vehicles. Due to their conservativeness, an urgent need from the AD community is to model realistic and diverse interactive skills. Learning-based methods, MARL techniques in particular, have much to offer towards solving this problem. Yet, a multi-agent AD simulator that can support MARL auto-curriculum training and induce diverse driving interactions is still lacking.

In the following sections, I will first present a dedicated simulation platform

called SMARTS¹ (Scalable Multi-Agent RL Training School). SMARTS supports the training, accumulation, and use of diverse behaviour models of road users, especially those that are powered by RL and MARL algorithms. Through SMARTS, researchers can, for the first time, apply truly learning-based techniques to induce increasingly more realistic and diverse interactions in AD simulations, which in turn ignites deeper and broader research interest on understanding multi-agent interactions in AD for MARL researchers.

At last, I present a blue sky idea that maintaining behavioural diversity in the policy space is essential for extending the successes of MARL techniques in the real physical world, outside the domain of purely video games. Towards achieving this goal, I emphasise that the framework of diversity aware auto-curricula could serve as one promising direction. Specifically, I first highlight the necessity of promoting behavioural diversity in multi-agent systems, and then present four open challenges to be solved for creating diversity aware auto-curricula, and finally discuss why AD via SMARTS serves as an excellent testbed for relevant studies on this topic.

7.1 Background of Autonomous Driving

Autonomous driving (AD) (a.k.a self-driving cars) (Badue et al., 2020) refers to the technique that enables a vehicle to sense its environment and to move safely to the destination with little or no human interventions. Typically, the architecture of a self-driving car contains two main components: the *perception* system, and the *decision-making* system (Paden et al., 2016). The perception system is responsible for answering the questions of “where am I ?” and “what is around me ?”; it contains subroutines such as car localisation, road mapping, and obstacle detection. Outputs from the perception system are then leveraged by the decision-making system to answer the questions of “what will happen next?”, and, finally, “what should

¹Note that SMARTS is a massive collaborative project that involves dozens of researchers and engineers. My contribution is at the side of proposing features that enable MARL training in SMARTS, and applying MARL techniques in SMARTS. The simulation engine of SMARTS, including its novelty in design principles (e.g., traffic provider, domain-specific language, the “bubble” mechanism) and their corresponding implementations, should not be taken as my contribution towards this chapter. The full description of SMARTS can be found in Zhou et al. (2020) and <https://github.com/huawei-noah/SMARTS>.

I do”; this includes solving tasks such as path planning, behaviour selection, and obstacle avoidance. For example, the behaviour selector subsystem is responsible for choosing the current driving behaviour including lane change, merging, traffic light handling, and meanwhile, avoiding collisions with static or moving obstacles within the decision horizon. The scope of the discussion in this chapter stays within the context of decision-making since that is where RL techniques naturally fit.

To gauge the level of autonomy of self-driving techniques, the Society of Automotive Engineers (SAE) International has published a category of levels based on the amount of human driver input and attentiveness that are required, in which the level 0 stands for the full-time performance by a human driver covering all aspects of the dynamic driving task, and the level 5 stands for the fully-automated driving system that monitors the driving environment under all roadway conditions that can be managed by human drivers ([International, 2014](#)).

An iconic event that spurs the development of self-driving techniques is probably the DARPA Grand Challenge. Since the first DARPA competition organised in 2004 where all competing cars failed the mission of navigating a 142-mile course throughout desert trails within a 10-hour time limit, and the best performer only managed to drive 7.3 miles, remarkable progress has been made in the past sixteen years. For example, the commercial company—Waymo—alone has driven more than 20 million miles on public roads under the SAE level-4 setting ([International, 2014; Waymo, 2020](#)). The vision of deploying AD techniques for better safety and efficiency in transportation is gradually getting substantiated.

7.2 Challenges in Modelling Diverse Interactions

In spite of such groundbreaking achievement, fundamental aspects of autonomous driving, especially a behavioural model that delivers realistic interactions with diverse road users, remain underexplored. Unfortunately, rather than embracing inter-driver interaction, current mainstream level-4 AD solutions restrict it. When encountering complex interactive scenarios, autonomous cars tend to slow down and wait for the situation to get clear. They rarely cut in front of another car or force

their way in at a merge, even when human drivers routinely do so. In California in 2018, 86% of crashes involving autonomous vehicles were attributable to the AD car’s conservative behaviours (Stewart, 2020), with 57% rear endings and 29% sideswipes by other vehicles on the AD car. Trial AD cars in Arizona and California are often targets of complaints about blocking other cars (Siddiqui, 2020), excessive hard braking (Efrati, 2020a), hesitant highway merging, and inflexible pick-up or drop-off locations (Efrati, 2020b,c). While rarely illegal, the overly conservative driving style frustrates human drivers as peer road users, and sometimes even causes hazards. It also restricts AD technologies from being applied on special-purpose vehicles such as ambulances or police cars.

Recently, data-driven approaches for training behaviour models capable of diverse and realistic interaction in simulators is receiving attention. Imitation learning is an obvious option (Bansal et al., 2019). Yet, a suitable data set is crucial. Publicly available AD datasets such as KITTI (Geiger et al., 2013), Oxford RobotCar (Maddern et al., 2017), and BDD100K (Yu et al., 2018) focus on perception and provide little data regarding interaction. There are driving behaviour-related datasets that are collected from bird’s eye view by cameras mounted on tall buildings or by drones, such as the NGSIM dataset (Alexiadis et al., 2004), highD dataset (Krajewski et al., 2018), and INTERACTION dataset (Zhan et al., 2019b). However, these datasets are limited in that the majority of filmed intersections are controlled by traffic lights; therefore, the interactions involved are trivial, while urban scenarios which contain genuine dense interactions, such as roundabouts and unprotected left turns, are missing. In addition to motion datasets in bird’s eye view, there are onboard-sensor-based types of datesets collected from LiDARS, front-view cameras, or GPS, including Argoverse (Chang et al., 2019), HDD dataset (Ramanishka et al., 2018), and 100-car study (Neale et al., 2005), but they either suffer from the incompleteness of interaction entities or there are very few repetitions at the same location with similar features, and are limited to only the ego vehicle’s perspective. Moreover, all these data sets involve large scale human labelling or at a minimum require near perfect detection and tracking modules to generate the labels, hindering

scaling and resource-constrained open research. Lastly, all of the above interaction data do not have the underlying physical models of social vehicles, which could be crucial for generating realistic behaviours in highly dynamic situations.

Apart from the limitations in data-driven approaches, another critical reason for the lack of realistic interactions is that existing AD *simulators* have very limited capacity for modelling realistic interactions with diverse driving behaviours. Simulators are crucial for validating the AI software controlling the autonomous vehicle (also called *ego vehicle*). For validating the ego vehicle’s interactive behaviour with *social vehicles* (i.e., other vehicles that share the same driving environment), we need diverse *social agents* capable of realistic and competent interaction. Conventional AD simulators ([Dosovitskiy et al., 2017](#); [Simulator, 2020](#); [Virtual Test Drive, VTD](#)) focus on modelling sensory inputs and control dynamics, rather than interaction. Behaviours of social vehicles thus end up being controlled by simple scripts or rule-based models (e.g., IDM for longitudinal control ([Treiber et al., 2000](#)) or MOBIL for lateral control ([Treiber and Kesting, 2009](#))). The resulting simulated interaction between ego and social vehicles falls far short of the richness and diversity of interaction in the real world. AD companies also heavily use replay of historical data collected from real-world trials to validate ego vehicle behaviour ([Atlantic, 2017](#)). However, such a data-replay approach to simulation does not allow true interaction between vehicles because the social vehicles are not controlled by intelligent agents but merely stick to historical trajectories ([Anguelov, 2020](#); [Vogt, 2020](#)). In short, it is still an open question how to create a population of diverse, intelligent social agents that can be adopted in simulation to provide traffic with realistic interaction.

7.3 When MARL Meets Autonomous Driving

To address the challenge of inducing realistic and diverse interactions in AD, we take the view that driving in a shared public space with multiple road users is fundamentally a multi-agent problem. To illustrate this perspective, consider the so-called “double merge” (Figure 7.1) studied in [Shalev-Shwartz et al. \(2016b\)](#). There

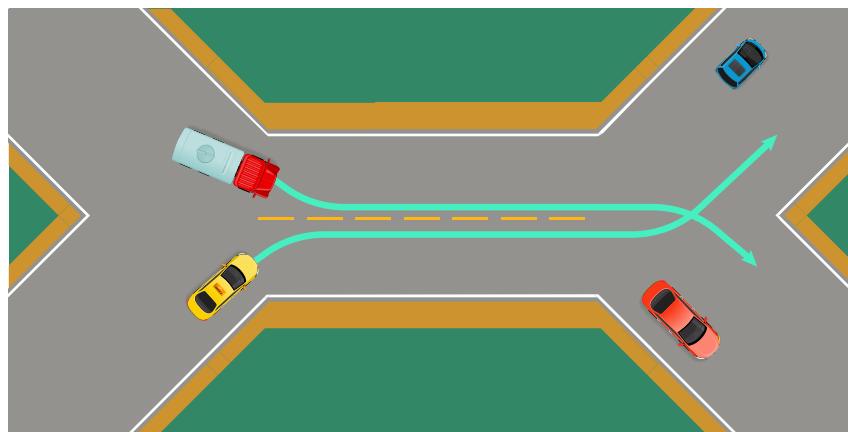


Figure 7.1: The double merge scenario in autonomous driving.

are vehicles coming into a shared road section from one side and aiming to exit this section on the opposite side. In the process, they need to change lane by weaving through cars that may also be changing lane in the opposite direction. Failing that, they will be forced to either go out on the wrong road or wait somewhere before the road splits until a usable gap emerges². Double merge is a sequential decision making setting that involves complex interactions among multiple agents. Questions one can ask about include: Should a car drive further or wait here when looking for a gap? Should it force a lane change even when there is no adequate gap? Will the other cars give the gap? Is this other car coming over to my lane so that I can trade places with it? Furthermore, handwritten rules are unlikely to scale up to the full complexity of interactive scenarios such as the double merge ([Shalev-Shwartz et al., 2016b](#)). We think learning with consideration of interaction is necessary and MARL is especially promising.

[Zhou et al. \(2020\)](#) (see their Table 1) proposed to categorise the expected learning outcomes of using MARL techniques for AD tasks into six levels (denoted by **M**), similar to the level of autonomy defined by SAE. **M0** agents are designed to follow specific rules and stick to them regardless of how the environment dynamics may have shifted. After driving many times through a dense intersection, for example, the agents have no improvements except through tweaks by human engineers.

²Such predicament does happen to autonomous cars. See a case in which a vehicle was forced to wait right before the split: <https://youtu.be/HjiiGCe1pE>, and a related case: <https://youtu.be/spw176TZ7-8?t=90>.

M1 agents can learn to adapt from their online experiences, and such a learning process is expected to change their behaviours for future runs. **M2** agents not only learn to adapt, but also learn to model other road users. However, there is still no direct information exchange among the learning agents during such a decentralised learning process. **M3** further requires information exchange among the agents during training so as to coordinate their learning, but at execution time there is neither centralised control nor direct information exchange. At **M4**, agents are required to coordinate their learning at the local group level (e.g., at an intersection) in a way such that the Nash equilibrium or other equilibrium variant is ensured at execution time. At **M5**, agents start to focus on solving how their local actions in the scenario (e.g., “how I change into the other lane here and now”) may impact global welfare (e.g., “may decide whether there is going to be a congestion on the highway five minutes later”) so as to minimise *the price of anarchy* (Roughgarden, 2005).

To date, AD research has mostly focused on **M0** with highly limited attempts at **M1** and **M2**. We believe that a key reason is the lack of suitable AD simulation of interaction among heterogeneous traffic participants on the road³. When it comes to MARL for AD interaction, the behaviour of the simulated traffic participants must be both realistic and diverse, especially for the *social vehicles* (i.e. vehicles sharing the driving environment with the autonomous vehicle) interacting with the *ego vehicles* (i.e. vehicles controlled by the AD software).

7.4 SMARTS: A Bespoke AD Testbed for MARL

In AD, we believe that a MARL approach powered by diversity-aware auto-curriculum can help solve the challenging problem of generating high-quality behaviour models that can approach human-level sophistication. This is because driving in a shared public space with many other road users is fundamentally a multi-agent problem wherein agents’ behaviour co-evolve. Co-evolved diverse and competent behaviours can allow AD simulation to encompass sophisticated

³Interestingly, multi-agent learning research at M3 through M5 happens more often for *traffic management* (Wei et al., 2019; Wu et al., 2017; Zhang et al., 2019b), for which simulation has been under heavy development for about twenty years (Barceló et al., 2005; Dresner and Stone, 2004; Krajzewicz et al., 2002).

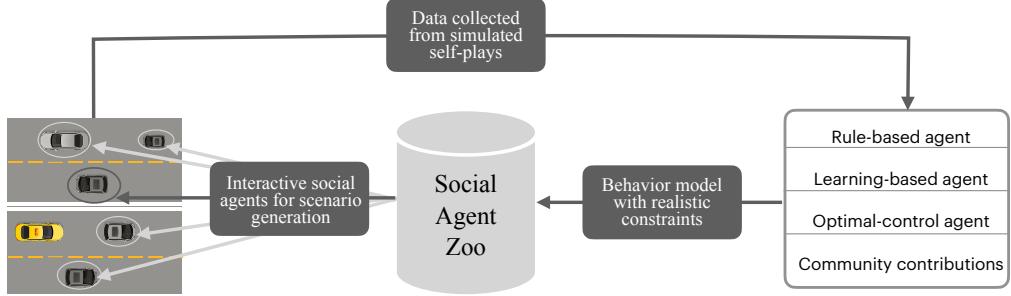


Figure 7.2: Bootstrapping interaction realism and diversity.

interactions seen among human drivers and thus help alleviate the conservativeness issue of existing AD solutions. However, for such a plan to work, we need an appropriate simulator that supports MARL auto-curriculum for diversity. Fortunately, the SMARTS AD simulator (Zhou et al., 2020) provides a well-suited platform. In this section, to stay self-contained, I will briefly describe the design goals of SMARTS, highlight its basic architecture, and the key features. I refer to <https://github.com/huawei-noah/SMARTS> for full details of the SMARTS platform.

Unlike many existing simulators, SMARTS is *natively multi-agent* in that social agents use the same APIs as the ego agent to control vehicles, and thus may use arbitrarily complex computational models, either rule-based or (MA)RL-driven (see Figure 7.2). To be sure, there are some excellent open-source simulators, such as CARLA (Dosovitskiy et al., 2017) and SUMO (Krajzewicz et al., 2002), and even environments explicitly designed for RL and multi-agent research, such as Flow (Wu et al., 2017) and AIM4 (AI Laboratory, 2006). Despite their respective successes, none of them comes close to addressing the specific need for in-depth MARL research for AD tasks. The most specifically relevant efforts are probably highway-env (Leurent, 2018), which is architected as a set of independently hand-crafted interaction scenarios, and BARK (Bernhard et al., 2020), which emphasises the development of behaviour models without explicit support for multi-agent research. Moreover, since there are only a small number of agent behaviour models to choose from in the aforementioned solutions, it is still challenging to configure realistic interaction scenarios. Finally, the research community still lacks standard benchmarking suites that are comparable to MuJoCo (Todorov et al., 2012) in how it

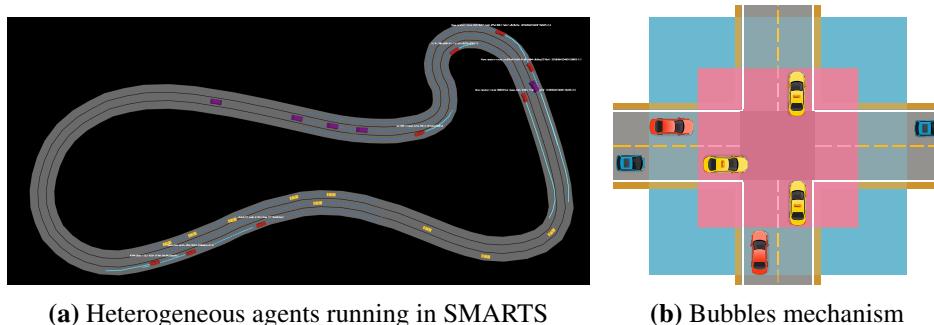


Figure 7.3: The SMARTS platform proposed by Zhou et al. (2020). **(a):** Multiple ego agents in training (red vehicles) can run simultaneously within a shared SMARTS instance. Each agent runs in a separate process and can also run remotely. **(b):** Bubbles are regions in which social vehicles (yellow) may be controlled by agents from the Social Agent Zoo and interact meaningfully with ego vehicles (red). Inside blue Zone, social vehicles are controlled by the traffic provider. Inside red Zone, they are controlled by agents from the Social Agent Zoo by MARL algorithms. Blue areas are transition zones.

brought physics simulation to RL research and to StarCraft II (Vinyals et al., 2017) in how it brought large scale games to multi-agent research, but remain faithful to the AD reality. Consequently, papers were published with ad-hoc, one-off small scale simulations that are typically concerned with just a single task and typically not maintained or reusable. SMARTS is in fact developed in response to such a situation. The goal of SMARTS is to provide an industrial-strength platform that helps to bring MARL research for AD to the next level, and both empowers and challenges it for many years to come.

7.4.1 Key Features

There are many research-friendly features in SMARTS, see screenshot in Figure 7.3, to name a few:

- SMARTS is 3D simulator with a realistic physics engine. Researchers can easily study the effects of wet road surfaces or flat tires.
- SMARTS is *natively multi-agent*, by which we mean that social agents could be as intelligent as ego agents and as heterogeneous as need be (see Figure 7.3a).
- For scalable integration, in addition to distributed training of ego agents supported through Ray (Moritz et al., 2018), SMARTS also manages its own *dis-*

tributed social agent computing.

- To provide the best possible *researcher experience*, we follow the standard Open-AI Gym APIs, provide a web-streaming visualisation solution, which allows the ongoing simulation to be viewed from anywhere, such as on a smartphone, and offer a comprehensive set of observations and action spaces that can easily fit various research designs with minimum pre-processing and post-processing.
- Given our emphasis on support for MARL research, SMARTS integrates with popular libraries such as PyMARL ([Samvelyan et al., 2019](#)) and MALib ([Wen, 2019](#)), provides the implementation of a few new algorithms, and supports a *comprehensive set of MARL algorithms*. (A most comprehensive set to our knowledge. See Table 7.1 for specifics.)
- SMARTS offers a *MARL benchmarking suite with AD-specific evaluation metrics*, which both poses challenging research questions and remains true to the AD reality. More benchmarking suites based on SMARTS are also pending release.

The most relevant to MARL research is probably the *social agent zoo* feature and the *bubble* mechanism (see Figure 7.3b). The *social agent zoo* maintained in SMARTS hosts a growing number of behaviour models readily usable as agents in simulation, regardless of their divergence in model architectures, observation/action spaces, or computational requirements. The key component that makes such computations possible is the built-in “bubble” mechanism, which defines a spatiotemporal region so that intensive computing is only activated inside the bubbles where fine-grained interaction is required, such as at unprotected left-turns, roundabouts, and highway double merges. In the meantime, agents, either ego or social, can learn from the interactive trajectories collected within the bubbles, and adapt their behaviours accordingly. These features together make SMARTS highly suitable for multi-agent auto-curriculum studies.

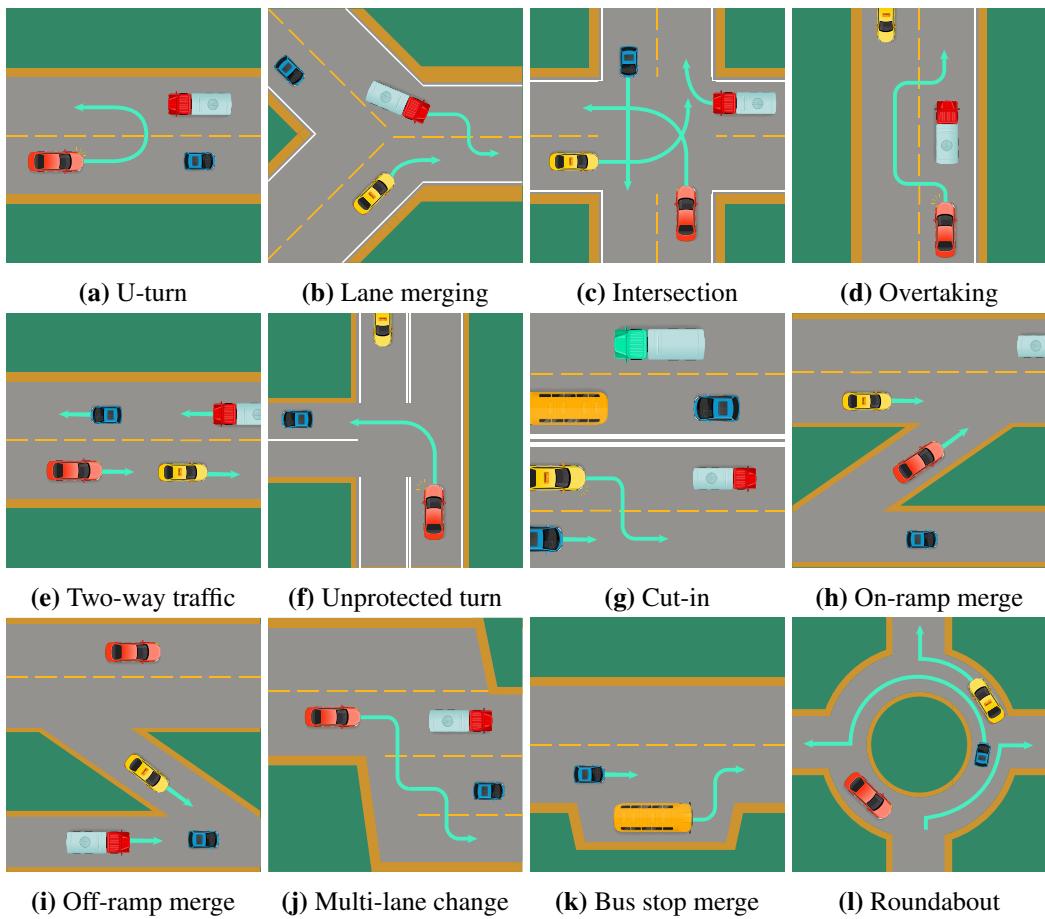


Figure 7.4: Scenarios of driving interaction specifiable in SMARTS

7.4.2 Support for RL Study

SMARTS supports training RL agents in numerous scenarios that vary in road structure and traffic (see Figure 7.4), along with different combinations of those scenarios reaching the scale of a city level. To train and evaluate the ego agents with specific parameters in these scenarios, we have implemented a benchmark runner based on Ray (Moritz et al., 2018) and RLlib (Liang et al., 2018). With Ray’s support, SMARTS enables users to conduct scalable, parallelisable training and evaluations.

When it comes to RL training, the **observation space** for an agent is specified as a configurable subset of available sensor types that include dynamic object list, bird’s-eye view occupancy grid maps and RGB images, ego vehicle states, and road structure. For example, a default observation in Zhou et al. (2020) is a stack of three consecutive frames covering the dynamic objects and key events. For each frame, it contains 1) relative position of goal; 2) distance to the centre of the lane; 3) speed; 4)

steering; 5) a list of heading errors; 6) at most eight neighbouring vehicles' driving states (relative distance, speed and position); 7) a bird's-eye view grey-scale image with the agent at the centre.

The **actions space** usually includes both the longitudinal control aspect (e.g., slow down, or speed up) and the lateral control aspect (e.g., turn left or right). The controls are executed based on the signal the RL agent sends to the environment, represented by the throttle/brake and the steering angle, just as what a human driver would output to a real car. SMARTS also provides a discrete action space, including if or not to change lane or to accelerate, to accommodate to Q-learning type methods. The **reward** can be customised by the user based on many statistics that SMARTS returns. By default, SMARTS provides distance travelled along the mission route per time step as the raw reward signal, which may be used in reward customisation that combines feedback on key events and other observations on the road. For example, an appropriate reward setting can be a combination of factors including whether the agent reached its goal, the distance to destination if the agent goes into the wrong direction, and whether collision happens. Furthermore, under the multi-agent setting, an agent's performance is not only a matter of how it behaves, but also depends on how other agents interact with it. Therefore, on top of the aforementioned reward setting, which essentially treats AD as a driving game, SMARTS also provides metrics in terms of behavioural analysis, reflecting the agents' interaction skills. For example, it computes *aggressiveness*, *safety*, *agility*, *stability*, *maneuver diversity* for a population of policies, and provides radar plots for representing these capabilities (Osband et al., 2019). We refer to Table 3 in Zhou et al. (2020) for the detailed mathematical definitions.

7.4.3 MARL Benchmarking Suite

One of the most significant advantages of SMARTS is that it is designed to support MARL researchs. In fact, SMARTS has the most comprehensive in-built MARL algorithms, which are listed in Table 7.1. Based on the supported MARL model zoo, SMARTS provides a benchmarking suite for AD tasks in different scenarios, which is comparable to the series of MuJoCo tasks (Todorov et al., 2012) in single-agent

Paradigm (Figure 2.1)	Algorithms	On/Off-Policy	Game	Critic Type
#4	BiCNet (Peng et al., 2017a)	Off	Coop.	Central
	CommNet (Sukhbaatar et al., 2016)	Off	Coop.	Central
#2	Independent Q (Tan, 1993)	Off	/	Independent
	Independent PG (Sutton et al., 1999)	On	/	Independent
	Independent AC (Mnih et al., 2016)	Both	/	Independent
	PR2 / GR2 (Wen et al., 2019b, 2018)	Off	Mixed	Central
#5	ROMMEÖ (Tian et al., 2019)	Off	Mixed	Central
	MAAC (Lowe et al., 2017a)	Both	Mixed	Central
	MADDPG (Lowe et al., 2017a)	Off	Mixed	Central
	MF-AC/MF-Q (Yang et al., 2018b)	Both	Coop.	Central
	COMA (Foerster et al., 2018b)	Off	Coop.	Central
	VDN (Sunehag et al., 2018)	Off	Coop.	Central
	QMIX (Rashid et al., 2018)	Off	Coop.	Central
	QTRAN (Son et al., 2019)	Off	Coop.	Central
	MAVEN (Mahajan et al., 2019)	Off	Coop.	Central
	Q-DPP (Yang et al., 2020)	Off	Coop.	Central
#6	Networked Q (Zhang et al., 2018b)	Off	Mixed	Network
	Networked AC (Zhang et al., 2018c)	On	Mixed	Network
	Value Propagation (Qu et al., 2019)	Off	Mixed	Network

Table 7.1: MARL algorithm suite available in SMARTS. The training paradigm of each algorithm is referred to Figure 2.1. The game refers to the type of game that can be solved. Critic types refer to what level of information it needs during training.

Algorithm	Scenario - No Social Vehicle			Scenario - Random Social Vehicle		
	Two-Way	Double Merge	Intersection	Two-Way	Double Merge	Intersection
DQN	0/0.97	0.77/0.23	0.83/0.20	0.40/0.60	0.60/0.23	0.92/0.05
PPO	0/1	0/1	0.1/0.07	0.25/0.75	0.02/0.98	0.50/0.45
MAAC	0/1	0.42/0.58	0/1	0.25/0.75	0.42/0.6	0.32/0.68
MF-AC	0/0.8	0.6/0.4	0.54/0.4	0.45/0.5	0.7/0.3	0.62/0.37
Network Q	0/0.3	0.7/0.25	0.4/0.23	0.4/0.2	0.8/0.2	0.75/0.2
CommNet	0/0.96	0.46/0.45	0.3/0.7	0.25/0.65	0.5/0.5	0.5/0.45
MADDPG	0/1	0.1/0.9	0/1	0.13/0.87	0.17/0.8	0.30/0.7

Table 7.2: Results reported in Zhou et al. (2020) on the average collision rate (the lower the better) / mission completion rate (the higher the better) in the MARL benchmarking suite of SMARTS. The left three and right three columns differ in whether there are social vehicles on the road provided controlled by a traffic provider such as SUMO.

RL research, and to StarCraft II combats (Vinyals et al., 2017) in multiple-player video games.

In the section, I show the example result of MARL benchmarks on three increasingly challenging driving scenarios that require non-trivial interactive capabilities. They are the two-way driving in Figure 7.4e, double merge in Figure 7.1, and unprotected intersections in Figure 7.4c. Each agent in each scenario has a different mission to complete: driving from a specific start to a specific goal. The traffic on the road consists of the agents in training as well as the background traffic provided by SUMO (Krajzewicz et al., 2002). The chosen baselines in the benchmark include two independent learning algorithms, DQN (Mnih et al., 2013) and PPO (Schulman et al., 2017), four centralised training methods, MAAC, MF-AC (Yang et al., 2018b), MADDPG (Lowe et al., 2017a), and Networked Fitted-Q (Zhang et al., 2018a), and one fully centralised method, CommNet (Sukhbaatar et al., 2016). Their driving performance is shown in Table 7.2 and the behavioural statistics are shown in Figure 7.5. A general conclusion we get by running the MARL benchmarking suite so far is that as the difficulty of the scenarios increases (two way driving is the easiest, and the intersection is the hardest), interaction among traffic participants become inevitable and more frequent; this poses great challenges for existing RL algorithms that do not model interactions explicitly. We hope the

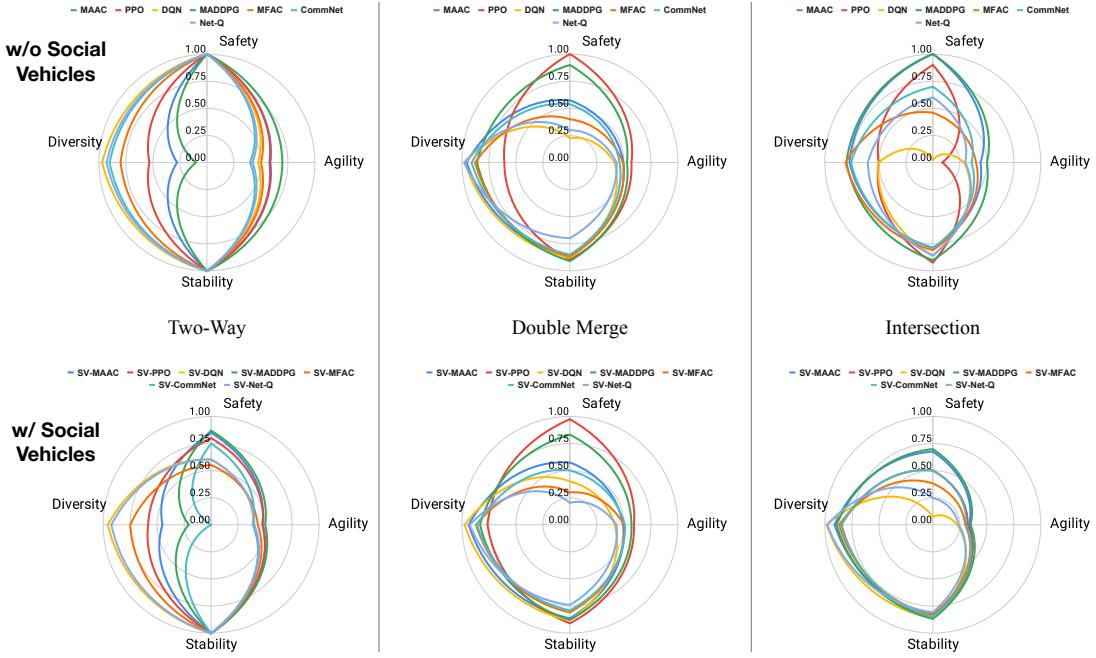


Figure 7.5: Results reported in Zhou et al. (2020) on four behaviour metrics in the MARL benchmarking suite of SMARTS. The larger the coverage, the more desirable the behaviour. The wider scattered the curves, the more diverse the behaviours. The upper three and bottom three plots differ in whether there are social vehicles on the road provided controlled by a traffic provider such as SUMO.

performance can be continuously improved by the upcoming models that the community is currently developing.

7.5 Blue Sky Idea: Diverse Auto-Curriculum is the Key to AD

MARL extends RL to cover the setting where there are multiple learning entities in the environment (Hernandez-Leal et al., 2019; Yang and Wang, 2020). This technique has also had remarkable successes especially on multi-player video games such as StarCraft (Vinyals et al., 2019b), Dota2 (Pachocki et al., 2018), and Hide and Seek (Baker et al., 2019a). Nonetheless, MARL so far has had relatively few successes in solving real-world problems.

In fact, the challenges of deploying RL in the real world are frequently discussed in both workshops (Workshop, 2019, 2020a,b) and papers (Dulac-Arnold et al., 2020, 2019). These can include a lack of an accurate simulator, a high cost of environmental interaction, and difficulty in learning both effective and diverse poli-

cies. Off-policy RL (Levine et al., 2020) or imitation learning (Hussein et al., 2017) methods could be used for policy evaluation or policy improvement in an offline manner; this would allow agents to learn good initial behaviour before ever interacting with an environment. However, these methods are only applicable if there are existing data sets. When the training data are limited, for example in the AD domain (Zhan et al., 2019a), offline methods are insufficient for robust performance in the real world due to a lack of *diversity* in agents’ behaviours (Dasari et al., 2019; Wang et al., 2017b). In fact, even in cases when a simulator is available, the issue of lacking behavioural diversity could still exist due to the sim-to-real gap (Matas et al., 2018; OpenAI et al., 2018; Rusu et al., 2017). Unfortunately, MARL suffers from all of these concerns. Moreover, things only worsen when one considers the additional complexity of multi-agent problems that arise from the cross product of multiple agents’ state and action spaces induced by social interactions. Developing frameworks that can deal with the underlying complexities of the MARL domain is crucial. We believe that the development of effective, yet diverse, behaviours is critical for MARL to impact real-world AD problems.

Towards such a goal, the framework of *auto-curricula* (Leibo et al., 2019; Portelas et al., 2020) serves as a promising direction. In natural evolution, species with stronger adaptability can flourish when nature alters the environment and previously well-adapted species can no longer survive in the new regime. Through this *co-evolution process* (Durham, 1991; Paredis, 1995; Rausher, 2001), the diversity of life on Earth has been maintained over billions of years. Inspired by the mechanism of biodiversity in nature, a series of MARL learning frameworks have been proposed recently and demonstrated remarkable empirical successes. These include open-ended evolution (Banzhaf et al., 2016; Lehman and Stanley, 2008; Standish, 2003), population-based training (Jaderberg et al., 2019; Liu et al., 2018), and training by emergent curricula (Baker et al., 2019a; Leibo et al., 2019; Portelas et al., 2020). In general, these frameworks can be unified under the idea of an *auto-curriculum* that automatically generates an endless procession of better-performing agents by exerting selection pressure among the multiple self-optimising agents.

The underlying principle of auto-curricula is that, since any adaptation an agent makes will have a cascading effect to which other agents must adapt in order to survive, there exists an intrinsic mechanism, arising from agents' social interactions, that provides an automatic curriculum that continually facilitate agents' acquisition of new skills. In order to extend MARL methods to real-world domains outside of purely video games, we envision that maintaining a diversity-aware auto-curriculum is critical for successful MARL applications.

7.5.1 The Necessity of A Diverse Auto-Curriculum

Nature exhibits a remarkable tendency towards *diversity* (Holland et al., 1992). Over the past billions of years, natural evolution has discovered a variety of unique species. Each of them is capable of orchestrating, in different ways, the complex biological processes that are necessary to survive. Equally, in computer science, machine intelligence can be considered the ability to adapt to a diverse set of complex environments (Hernández-Orallo, 2017). This suggests that the ceiling of intelligence rises when environments of increasing diversity and complexity are provided. In fact, recent successes in developing AI that can achieve super-human performance on complicated multi-player video games, such as StarCraft (Vinyals et al., 2019a), Hide and Seek (Baker et al., 2019a), and Dota2 (Pachocki et al., 2018), have provided factual justifications for emphasising behavioural diversity when designing learning protocols in multi-agent systems. Specifically, promoting behavioural diversity is pivotal for MARL methods in that it not only prevents AI agents from checking the same policies repeatedly, but more importantly, helping the population of AI agents discover niche skills and maintain robust performance when encountering unfamiliar types of opponents at test time. In the example of training StarCraft AI (Han et al., 2020; Vinyals et al., 2019a), adding a diverse set of strategies, which aims to exploit the existing policy pool, substantially helps agents avoid systematic weaknesses when playing against professional human players.

Behavioural diversity and the *non-transitivity* of many environments are intertwined. In biological systems, it has been found that biodiversity is promoted by the non-transitive interactions among many competing populations (Kerr et al., 2002;

Reichenbach et al., 2007). The central feature of such non-transitive relations can be thought of as analogous to a Rock-Paper-Scissors (RPS) game, in which Rock crushes Scissors, Scissors cuts Paper, and Paper wraps Rock. Coincidentally, in game theory, the necessity of pursuing behavioural diversity is also deeply rooted in the non-transitive structure of games (Balduzzi et al., 2019, 2018b; Lanctot et al., 2017). In general, an arbitrary game, of either the normal-form type (Candogan et al., 2011) or the differential type (Balduzzi et al., 2018a), can always be decomposed into a sum of two components: a *transitive part* plus a *non-transitive part*. The transitive part of a game represents the structure in which the rule of winning is transitive (i.e., if strategy A beats B, B beats C, then A can surely beat C), and the non-transitive part refers to the game structure in which the set of strategies follow a cyclic rule (e.g., the endless cycles among Rock, Paper, and Scissors). Diversity matters especially for the non-transitive part simply because there is no consistent winner in such sub-games: if a player only plays Rock, it can be exploited by Paper, but not so if it is diverse in playing Rock and Scissor. In fact, real-world problems often consist of a mixture of both parts (Czarnecki et al., 2020), therefore it is critical to design objectives in the learning framework that can lead to behavioural diversity.

Effective MARL performance often requires diversity in two aspects. The first aspect is about the training player using diversified strategies against a fixed type of opponents—since most games involve aforementioned non-transitivity in the policy space, it is necessary for each player to acquire a diverse population of winning strategies to achieve high unexploitability. The second aspect is the ability to pick a diverse set of opponents⁴ during training. In playing cooperative card games like Hanabi (Bard et al., 2020), one player may or may not understand indirect signalling when choosing a card to play. If an agent has not learned to play diversely with both mindsets, it will fail to accurately model the collaborator and play sub-optimally. Similarly, in real-world driving, many different locales have different conventions. The UK and the US drive on different sides of the road, or, even within the same

⁴We use the term “opponent” for presentation purposes in this section, acknowledging that agents may be teammates, opponents, or something in between for non-zero-sum games.

country, different cities can have different conventions. For example, the *Pittsburgh Left* assumes that a few cars will turn left in front of traffic at the beginning of a green light (Wikipedia, 2020), while other locales assume cars will turn left in front of traffic during a yellow or the beginning of a red light (School, 2018). As a result, it is expected that an autonomous agent without a diverse mindset could easily create hazards on roads (Waymo, 2020).

7.5.2 Open Challenges of Applying Auto-Curricula in AD

Auto-curricula (Baker et al., 2019a; Leibo et al., 2019; Portelas et al., 2020) provide a framework that automatically shapes the learning procedures for AI agents by consistently challenging them with new tasks that are adapting to their capabilities. If the challenges generated by an auto-curriculum become increasingly diverse and complex over time, AI agents will accumulate more diverse and effective skills.

In fact, recent successes in training AIs that achieve super-human performance and acquire surprisingly diverse behaviours on complex video games (Baker et al., 2019a; Silver et al., 2016; Vinyals et al., 2019a) provide substantial justification for adopting auto-curricula as an effective learning protocol to account for diversity. However, in order to serve as a general framework in tackling more real-world problems beyond video games, auto-curriculum still faces four open challenges.

Open Challenge I

How do we measure diversity in an auto-curriculum?

The first challenge is to define the correct objective to measure and promote diversity in the generated auto-curriculum. In the single-agent setting, diversity can be defined through a different reward function (Lehman and Stanley, 2008, 2011a), visiting a new state (Eysenbach et al., 2018; Such et al., 2017; Yang et al., 2020) or acquiring a new skill (Florensa et al., 2017; Hausman et al., 2018). However, in the MARL setting, since there are multiple players, and each player has a population of strategies, diversity should be considered in the joint policy space, considering all agents' existing strategies. Yet, there is minimal work that tries to quantify the behavioural diversity at the population level. Existing definitions are often ad hoc. For

example, behavioural diversity can be defined as the variance in rewards (Lehman and Stanley, 2008, 2011a), the convex hull of a *gamescape* (Balduzzi et al., 2019), whether or not visiting a new environmental state (Eysenbach et al., 2018; Such et al., 2017; Yang et al., 2020), or, acquiring new types of skills in a task (Florensa et al., 2017; Hausman et al., 2018). Although there are no straightforward answers, we believe one promising direction could be to leverage the *determinantal point process* (Kulesza et al., 2012) from quantum physics where the diversity is measured through the determinant value in a vector space, thus the level of orthogonality among the input vectors, which can be represented by agents' different joint-strategy profiles in terms of rewards (see Chapter 6).

Open Challenge II

How do we generate diversity-aware auto-curricula, especially in non-zero-sum settings?

The second challenge is about the applicability on *non-zero-sum games*. The curricula in the examples of StarCraft (Vinyals et al., 2019b) or Hide and Seek (Baker et al., 2019a) are generated by competitive self-play from the players in zero-sum games. However, many real-world tasks, such as autonomous driving, are not zero-sum—in fact, they tend to be a mixed setting where cooperation outweighs competition. Therefore, creating an auto-curriculum in non-zero-sum games is an open problem. Interestingly, recent studies have shown that adapting in social dilemmas can also create an effective auto-curriculum for the emergence of collective cooperation (Hughes et al., 2018; Leibo et al., 2017; Perolat et al., 2017), that is, through sequences of new challenges in addressing social dilemmas, agents eventually learn to achieve a socially-beneficial outcome. This in fact resembles tasks such as discovering collective driving strategies that can mitigate congestion, for example, consider the case of solving Braess's paradox (Braess, 1968) (i.e., a typical example in modelling road network and traffic flow) where agents progressively learn to sanction those who tend to over-exploit the shared resources, thus creating new curriculum. Nonetheless, creating curricula for collective cooperation

is still under-developed relative to auto-curricula induced by zero-sum games. Importantly, as pointed out by Leibo et al. (2019), auto-curricula induced by social dilemmas could be cursed by the “no-free-lunch” property: once you resolve a social dilemma in one place, then another one crops up somewhere else, a problem also known as higher-order social dilemmas (Mathew, 2017; Ostrom, 2000).

Open Challenge III

How do we shape an auto-curriculum to induce diverse yet effective behaviours?

Thirdly, although RL techniques offer insights about how a desirable behaviour can be learned in a fixed environment, it is still unclear how complex and useful behaviours can be best developed, while such behaviours influence the environment. In fact, it is often the case that the more complex the behaviour, the less likely it is generated directly from scratch (Leibo et al., 2019). An example is that it is highly unlikely a policy at the world-champion level is quickly generated by the curriculum once the training of AI agents to drive in F1 competitions starts. Moreover, this issue is only exacerbated when multiple agents ($N \gg 2$) are involved to explore the joint-strategy space. Fortunately, initial progress has been made by works on *Policy Space Response Oracle (PSRO)* (Balduzzi et al., 2019; Lanctot et al., 2017; Muller et al., 2019) where different kinds of *rectifiers* have been proposed to shape the auto-curricula so that effective behaviours with high quality can be emphasised. For example, PSRO with a *Nash rectifier* (Balduzzi et al., 2019) explores only strategies that have positive Nash support so as to preserve the strategy strength. Despite the empirical success in generating diverse yet effective strategies, PSRO methods only work in solving symmetric zero-sum games, a limitation highlighted in Open Challenge II.

Open Challenge IV

How do we deal with the non-transitivity when learning in a diversity-aware auto-curricula?

Lastly, results on game decomposition suggests that a game (Balduzzi et al., 2019; Candogan et al., 2011) generally consists of both *transitive* and *non-transitive* structures. In fact, the topological structure of real-world tasks often resembles a spinning top if projected onto a 2D space (Czarnecki et al., 2020), with the x-axis being the non-transitive dimension and y-axis being the transitive dimension. And the non-transitive part can in fact harm the effectiveness of auto-curriculum. For example, an auto-curriculum generated by self-play in zero-sum games (Gilpin, 1975; Samothrakis et al., 2012) could make a learning agent endlessly chase its own tail by creating the same tasks repetitively without breaking out. Things become even worse when the non-transitivity issue couples with the catastrophic-forgetting property of the model itself, such as with deep neural networks (Kirkpatrick et al., 2017). As a result, agents may end up with just acquiring mediocre solutions or getting trapped in limited cycles in the strategy space forever (Balduzzi et al., 2019, 2018b). Although memorising a library of all possible policies can help prevent cycling (e.g., three strategies in the toy example of Rock Paper Scissor), for many real-world tasks, the dimension of the non-transitive cycles can be huge, as a result, building such a library itself becomes an endless task (Czarnecki et al., 2020).

7.6 Chapter Summary

In this chapter, I elaborate on the great potentials of applying MARL techniques to model realistic interactions in autonomous driving. In substantiating my claims, I first introduce SMARTS, an open-source platform that is dedicated to bringing together scalable multi-agent learning and scalable simulation of realistic driving interactions in autonomous driving. The SMARTS platform can support investigations into how various MARL algorithms perform in the AD context, and in turn, MARL algorithms can help offer new types of high-quality interactions for AD research. Furthermore, I present a blue sky idea that creating diversity-aware auto-curricula is critical to support MARL situated in the real physical world, outside the classic domain of purely video games. I present four open research challenges to be solved for creating diversity-aware auto-curricula, and suggest AD via SMARTS as

an important testbed in which solutions to those challenges can be explored. Overall, I believe that the pressing need for high-quality behaviour models in AD simulation is an excellent opportunity for the MARL community to make a unique contribution by (1) theoretically addressing the modelling challenges on behavioural diversity and, (2) experimentally training generations of increasingly diverse and competent agents to provide the right kind of interaction that is currently missing in the AD research.

Chapter 8

Conclusions and Future Work

Multi-agent interaction is a common aspect of many real-world problems. The ubiquity of optimal decision-making process with multiple agents involved in a stochastic environment has made MARL a significant and fast-developing research area in machine learning/AI. In this thesis, I have provided dedicated treatments for MARL problems with many many agents. Specifically, I offered algorithmic developments tackling three aspects of MARL problems, including scalable policy evaluation (Chapter 4), scalable policy learning (Chapter 5), and learning in open-ended meta-games (Chapter 6). As a cherry on the cake, I demonstrate two applications of MARL on two real-world problems (Chapter 3 and Chapter 7), which show the great potential of this technique. Finally, I hope that my research could provide a stimulus for future studies on MARL topics and identify the following four research directions for future research.

- **Deep MARL Theory:** In contrast to the remarkable empirical success of MARL methods, developing theoretical understandings of MARL techniques are very much under-explored in the literature. Although many early works have been conducted on understanding the convergence property and the finite-sample bound of single-agent RL algorithms ([Bertsekas and Tsitsiklis, 1996](#)), extending those results into multi-agent, even many-agent, settings seem to be non-trivial. Furthermore, it has become a common practice nowadays to use DNNs to represent value functions in RL and multi-agent RL. In fact, many recent remarkable successes of multi-agent RL benefit from the success of deep

learning techniques (Baker et al., 2019b; Pachocki et al., 2018; Vinyals et al., 2019b). Therefore, there are pressing needs to develop theories that could explain and offer insights into the effectiveness of deep MARL methods. Overall, I believe there is an approximate ten-year gap between the theoretical developments of single-agent RL and multi-agent RL algorithms. Learning the lessons from single-agent RL theories and extending them into multi-agent settings, especially understanding the incurred difficulty due to involving multiple agents, and then generalising the theoretical results to include DNNs could probably act as a practical road map in developing MARL theories. Along this thread, I recommend the work of Zhang et al. (2019c) for a comprehensive summary of existing MARL algorithms that come with theoretical convergence guarantee.

- **Safe and Robust MARL:** Although RL provides a general framework for optimal decision making, it has to incorporate certain types of constraints when RL models are truly to be deployed in the real-world environment. I believe it is critical to firstly account for MARL with robustness and safety constraints; one direct example is on autonomous driving. At a very high level, robustness refers to the property that an algorithm can generalise and maintain robust performance in settings that are different from the training environment (Abdullah et al., 2019; Morimoto and Doya, 2005). And safety refers to the property that an algorithm can only act in a pre-defined safety zone with minimum times of violations even during training time (Garcia and Fernández, 2015). In fact, the community is still at the early stage of developing theoretical frameworks to encompass either robust or safe constraint in single-agent settings. In the multi-agent setting, the problem could only become more challenging because the solution now requires to take into account the coupling effect between agents, especially those agents that have conflict interests (Li et al., 2019b). In addition to opponents, one should also consider robustness towards the uncertainty of environmental dynamics (Zhang et al., 2020), which in turn will change the behaviours of opponents and pose a more significant challenge.

- **Model-Based MARL:** Most of the algorithms I have introduced in this thesis are *model-free*, in the sense that the RL agent does not need to know how the environment works and it can learn how to behave optimally through purely interacting with the environment. In the classic control domain, *model-based* approaches have been extensively studied in which the learning agent will first build an explicit state-space “model” to understand how the environment works in terms of state-transition dynamics and reward function, and then learn from the “model”. The benefit of model-based algorithms lies in the fact that they often require much fewer data samples from the environment ([Deisenroth and Rasmussen, 2011](#)). The MARL community has initially come up with model-based approaches, for example the famous R-MAX algorithm ([Brafman and Tennenholz, 2002](#)), nearly two decades ago. Surprisingly, the developments along the model-based thread halted ever since. Given the impressive results that model-based approaches have demonstrated on single-agent RL tasks ([Hafner et al., 2019a,b](#); [Schrittwieser et al., 2020](#)), model-based MARL approaches deserves more attention from the community.
- **Multi-Agent Meta-RL:** Throughout this thesis, I have introduced many MARL applications; each task needs a bespoke MARL model to solve. A natural question to ask is whether we can use one model that can generalise across multiple tasks. For example, [Terry et al. \(2020\)](#) has put together almost one hundred MARL tasks, including Atari, robotics, and various kinds of board games and pokers into a Gym API. An ambitious goal is to develop algorithms that can solve all of the tasks in one or a few shots. This requires multi-agent meta-RL techniques. Meta-learning aims to train a generalised model on a variety of learning tasks, such that it can solve new learning tasks with few or without additional training samples. Fortunately, [Finn et al. \(2017\)](#) has proposed a general meta-learning framework – MAML – that is compatible with any model trained with gradient-descent based methods. Although MAML works well on supervised learning tasks, developing meta-RL algorithms seems to be highly non-trivial ([Rothfuss et al., 2018](#)), and introducing the meta-learning frame-

work on top of MARL is even an uncharted territory. I expect multi-agent meta-RL to be a challenging yet fruitful research topic, since making a group of agents master multiple games necessarily requires agents to automatically discover their identities and roles when playing different games; this itself is a hot research idea. Besides, the meta-learner in the outer loop would need to figure out how to compute the gradients with respect to the entire inner-loop subroutine, which must be a MARL algorithm such as multi-agent policy gradient method or mean-field Q-learning, and, this would probably lead to exciting enhancements to the existing meta-learning framework.

Bibliography

- Abadi, M., Agarwal, A., Barham, P., Brevdo, E., Chen, Z., Citro, C., Corrado, G. S., Davis, A., Dean, J., Devin, M., et al. (2016). Tensorflow: Large-scale machine learning on heterogeneous distributed systems. *arXiv preprint arXiv:1603.04467*.
- Abbasi-Yadkori, Y., Bartlett, P., Bhatia, K., Lazic, N., Szepesvari, C., and Weisz, G. (2019). Politex: Regret bounds for policy iteration using expert prediction. In *International Conference on Machine Learning*, pages 3692–3702.
- Abdallah, S. and Lesser, V. (2008). A multiagent reinforcement learning algorithm with non-linear dynamics. *Journal of Artificial Intelligence Research*, 33:521–549.
- Abdullah, M. A., Ren, H., Ammar, H. B., Milenkovic, V., Luo, R., Zhang, M., and Wang, J. (2019). Wasserstein robust reinforcement learning. *arXiv preprint arXiv:1907.13196*.
- Adler, I. (2013). The equivalence of linear programs and zero-sum games. *International Journal of Game Theory*, 42(1):165–177.
- Adler, J. L. and Blue, V. J. (2002). A cooperative multi-agent transportation management and route guidance system. *Transportation Research Part C: Emerging Technologies*, 10(5-6):433–454.
- Adolphs, L., Daneshmand, H., Lucchi, A., and Hofmann, T. (2019). Local saddle point optimization: A curvature exploitation approach. In *The 22nd International Conference on Artificial Intelligence and Statistics*, pages 486–495.
- AI Laboratory, U. o. T. a. A. (2006). AIM: Autonomous intersection management. <http://www.cs.utexas.edu/~aim/>.
- Al-Tamimi, A., Lewis, F. L., and Abu-Khalaf, M. (2007). Model-free q-learning designs for linear discrete-time zero-sum games with application to h-infinity control. *Automatica*, 43(3):473–481.
- Alexiadis, V., Colyar, J., Halkias, J., Hranac, R., and McHale, G. (2004). The next generation simulation program. *Institute of Transportation Engineers. ITE Journal*, 74(8):22.

- Amato, C., Bernstein, D. S., and Zilberstein, S. (2010). Optimizing fixed-size stochastic controllers for pomdps and decentralized pomdps. *Autonomous Agents and Multi-Agent Systems*, 21(3):293–320.
- Anahtarci, B., Karıksız, C. D., and Saldi, N. (2019). Fitted q-learning in mean-field games. *arXiv preprint arXiv:1912.13309*.
- Anahtarci, B., Kariksiz, C. D., and Saldi, N. (2020). Q-learning in regularized mean-field games. *arXiv preprint arXiv:2003.12151*.
- Andersson, D. and Djehiche, B. (2011). A maximum principle for sdes of mean-field type. *Applied Mathematics & Optimization*, 63(3):341–356.
- Anguelov, D. (2020). Presentation at MIT course on self-driving cars. https://www.youtube.com/watch?v=Q0nGo2-y0xY&feature=youtu.be&t=1920&ab_channel=LexFridman.
- Arslan, G. and Yüksel, S. (2016). Decentralized q-learning for stochastic teams and games. *IEEE Transactions on Automatic Control*, 62(4):1545–1558.
- Ashby, W. R. (1991). Principles of the self-organizing system. In *Facets of Systems Science*, pages 521–536. Springer.
- Atlantic, T. (2017). Inside Waymo’s secret world for training self-driving cars. <https://www.theatlantic.com/technology/archive/2017/08/inside-waymos-secret-testing-and-simulation-facilities/537648/>.
- Auer, P., Cesa-Bianchi, N., Freund, Y., and Schapire, R. E. (1995). Gambling in a rigged casino: The adversarial multi-armed bandit problem. In *Proceedings of IEEE 36th Annual Foundations of Computer Science*, pages 322–331. IEEE.
- Auer, P., Cesa-Bianchi, N., Freund, Y., and Schapire, R. E. (2002). The nonstochastic multiarmed bandit problem. *SIAM journal on computing*, 32(1):48–77.
- Auer, P., Jaksch, T., and Ortner, R. (2009). Near-optimal regret bounds for reinforcement learning. In *Advances in neural information processing systems*, pages 89–96.
- Azar, M. G., Osband, I., and Munos, R. (2017). Minimax regret bounds for reinforcement learning. In *International Conference on Machine Learning*, pages 263–272.
- Bäck, T., Fogel, D. B., and Michalewicz, Z. (1997). Handbook of evolutionary computation. *Release*, 97(1):B1.
- Badue, C., Guidolini, R., Carneiro, R. V., Azevedo, P., Cardoso, V. B., Forechi, A., Jesus, L., Berriel, R., Paixão, T. M., Mutz, F., et al. (2020). Self-driving cars: A survey. *Expert Systems with Applications*, page 113816.

- Bak, P. (2013). *How nature works: the science of self-organized criticality*. Springer Science & Business Media.
- Baker, B., Kanitscheider, I., Markov, T., Wu, Y., Powell, G., McGrew, B., and Mordatch, I. (2019a). Emergent tool use from multi-agent autocurricula. In *International Conference on Learning Representations*.
- Baker, B., Kanitscheider, I., Markov, T. M., Wu, Y., Powell, G., McGrew, B., and Mordatch, I. (2019b). Emergent tool use from multi-agent autocurricula. *CoRR*, abs/1909.07528.
- Balduzzi, D., Garnelo, M., Bachrach, Y., Czarnecki, W., Pérolat, J., Jaderberg, M., and Graepel, T. (2019). Open-ended learning in symmetric zero-sum games. In *ICML*, volume 97, pages 434–443. PMLR.
- Balduzzi, D., Racaniere, S., Martens, J., Foerster, J., Tuyls, K., and Graepel, T. (2018a). The mechanics of n-player differentiable games. In *ICML*, volume 80, pages 363–372. JMLR.org.
- Balduzzi, D., Tuyls, K., Perolat, J., and Graepel, T. (2018b). Re-evaluating evaluation. In *Advances in Neural Information Processing Systems*, pages 3268–3279.
- Bansal, M., Krizhevsky, A., and Ogale, A. (2019). Chauffeurnet: Learning to drive by imitating the best and synthesizing the worst. In *Proceedings of Robotics: Science and Systems*, FreiburgimBreisgau, Germany.
- Banzhaf, W., Baumgaertner, B., Beslon, G., Doursat, R., Foster, J. A., McMullin, B., De Melo, V. V., Miconi, T., Spector, L., Stepney, S., et al. (2016). Defining and simulating open-ended novelty: requirements, guidelines, and challenges. *Theory in Biosciences*, 135(3):131–161.
- Barcelí, J., Codina, E., Casas, J., Ferrer, J. L., and García, D. (2005). Microscopic traffic simulation: A tool for the design, analysis and evaluation of intelligent transport systems. *J. Intell. Robotics Syst.*, 41(2–3):173–203.
- Bard, N., Foerster, J. N., Chandar, S., Burch, N., Lanctot, M., Song, H. F., Parisotto, E., Dumoulin, V., Moitra, S., Hughes, E., et al. (2020). The hanabi challenge: A new frontier for ai research. *Artificial Intelligence*, 280:103216.
- Barik, S., Bapat, R., and Pati, S. (2015). On the laplacian spectra of product graphs. *Applicable Analysis and Discrete Mathematics*, 9.
- Barto, A. G. (2013). Intrinsic motivation and reinforcement learning. In *Intrinsically motivated learning in natural and artificial systems*, pages 17–47. Springer.
- Bellemare, M., Srinivasan, S., Ostrovski, G., Schaul, T., Saxton, D., and Munos, R. (2016). Unifying count-based exploration and intrinsic motivation. In *Advances in neural information processing systems*, pages 1471–1479.

- 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.
- Ben-Tal, A., Margalit, T., and Nemirovski, A. (2001). The ordered subsets mirror descent optimization method with applications to tomography. *SIAM Journal on Optimization*, 12(1):79–108.
- Benaïm, M. and Faure, M. (2013). Consistency of vanishingly smooth fictitious play. *Mathematics of Operations Research*, 38(3):437–450.
- Benaïm, M. and Hirsch, M. W. (1999). Mixed equilibria and dynamical systems arising from fictitious play in perturbed games. *Games and Economic Behavior*, 29(1-2):36–72.
- Benders, J. (1962). Partitioning procedures for solving mixed-variable programming problems, numerische matkematic 4.
- Bengio, Y. (2009). *Learning deep architectures for AI*. Now Publishers Inc.
- Bensoussan, A., Frehse, J., Yam, P., et al. (2013). *Mean field games and mean field type control theory*, volume 101. Springer.
- Berger, U. (2007). Brown’s original fictitious play. *Journal of Economic Theory*, 135(1):572–578.
- Bernhard, J., Esterle, K., Hart, P., and Kessler, T. (2020). BARK: Open behavior benchmarking in multi-agent environments. *arXiv preprint arXiv:2003.02604*.
- Bernstein, D. S., Amato, C., Hansen, E. A., and Zilberstein, S. (2009). Policy iteration for decentralized control of markov decision processes. *Journal of Artificial Intelligence Research*, 34:89–132.
- Bernstein, D. S., Givan, R., Immerman, N., and Zilberstein, S. (2002). The complexity of decentralized control of markov decision processes. *Mathematics of operations research*, 27(4):819–840.
- Bertsekas, D. P. (2005). The dynamic programming algorithm. *Dynamic Programming and Optimal Control; Athena Scientific: Nashua, NH, USA*, pages 2–51.
- Bertsekas, D. P. (2012). Weighted sup-norm contractions in dynamic programming: A review and some new applications. *Dept. Elect. Eng. Comput. Sci., Massachusetts Inst. Technol., Cambridge, MA, USA, Tech. Rep. LIDS-P-2884*.
- Bertsekas, D. P. and Tsitsiklis, J. N. (1996). *Neuro-dynamic programming*. Athena Scientific.
- Billings, D., Burch, N., Davidson, A., Holte, R., Schaeffer, J., Schauenberg, T., and Szafron, D. (2003). Approximating game-theoretic optimal strategies for full-scale poker. In *IJCAI*, volume 3, page 661.

- Binder, K., Heermann, D., Roelofs, L., Mallinckrodt, A. J., and McKay, S. (1993). Monte carlo simulation in statistical physics. *Computers in Physics*, 7(2):156–157.
- Blackwell, D. et al. (1956). An analog of the minimax theorem for vector payoffs. *Pacific Journal of Mathematics*, 6(1):1–8.
- Blei, D. M., Kucukbir, A., and McAuliffe, J. D. (2017). Variational inference: A review for statisticians. *Journal of the American statistical Association*, 112(518):859–877.
- Bloembergen, D., Tuyls, K., Hennes, D., and Kaisers, M. (2015). Evolutionary dynamics of multi-agent learning: A survey. *Journal of Artificial Intelligence Research*, 53:659–697.
- Blume, L. E. (1993). The statistical mechanics of strategic interaction. *Games and economic behavior*, 5(3):387–424.
- Boccaletti, S., Latora, V., Moreno, Y., Chavez, M., and Hwang, D.-U. (2006). Complex networks: Structure and dynamics. *Physics reports*, 424(4):175–308.
- Boltzmann, L. (1974). The second law of thermodynamics. In *Theoretical physics and philosophical problems*, pages 13–32. Springer.
- Bonabeau, E., Theraulaz, G., Deneubourg, J.-L., Aron, S., and Camazine, S. (1997). Self-organization in social insects. *Trends in Ecology & Evolution*, 12(5):188–193.
- Borkar, V. S. (1997). Stochastic approximation with two time scales. *Systems & Control Letters*, 29(5):291–294.
- Borkar, V. S. (2002). Reinforcement learning in markovian evolutionary games. *Advances in Complex Systems*, 5(01):55–72.
- Boutilier, C., Dean, T., and Hanks, S. (1999). Decision-theoretic planning: Structural assumptions and computational leverage. *Journal of Artificial Intelligence Research*, 11:1–94.
- Bowling, M. (2000). Convergence problems of general-sum multiagent reinforcement learning. In *ICML*, pages 89–94.
- Bowling, M. (2005). Convergence and no-regret in multiagent learning. In *Advances in neural information processing systems*, pages 209–216.
- Bowling, M. and Veloso, M. (2000). An analysis of stochastic game theory for multiagent reinforcement learning. Technical report, Carnegie-Mellon Univ Pittsburgh Pa School of Computer Science.
- 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.
- Braess, D. (1968). Über ein paradoxon aus der verkehrsplanung. *Unternehmensforschung*, 12(1):258–268.
- Brafman, R. I. and Tennenholz, M. (2002). R-max-a general polynomial time algorithm for near-optimal reinforcement learning. *Journal of Machine Learning Research*, 3(Oct):213–231.
- Breton, M., Filar, J. A., Haurle, A., and Schultz, T. A. (1986). On the computation of equilibria in discounted stochastic dynamic games. In *Dynamic games and applications in economics*, pages 64–87. Springer.
- Brown, G., Wyatt, J., Harris, R., and Yao, X. (2005). Diversity creation methods: a survey and categorisation. *Information Fusion*, 6(1):5–20.
- Brown, G. W. (1951). Iterative solution of games by fictitious play. *Activity analysis of production and allocation*, 13(1):374–376.
- Brown, N., Kroer, C., and Sandholm, T. (2017). Dynamic thresholding and pruning for regret minimization. In *AAAI*, pages 421–429.
- Brown, N., Lerer, A., Gross, S., and Sandholm, T. (2019). Deep counterfactual regret minimization. In *International Conference on Machine Learning*, pages 793–802.
- Brown, N. and Sandholm, T. (2015). Regret-based pruning in extensive-form games. In *Advances in Neural Information Processing Systems*, pages 1972–1980.
- Brown, N. and Sandholm, T. (2017). Reduced space and faster convergence in imperfect-information games via pruning. In *International conference on machine learning*, pages 596–604.
- Brown, N. and Sandholm, T. (2018). Superhuman ai for heads-up no-limit poker: Libratus beats top professionals. *Science*, 359(6374):418–424.
- Brown, N. and Sandholm, T. (2019). Superhuman ai for multiplayer poker. *Science*, 365(6456):885–890.
- Brown, T. B., Mann, B., Ryder, N., Subbiah, M., Kaplan, J., Dhariwal, P., Neelakantan, A., Shyam, P., Sastry, G., Askell, A., et al. (2020). Language models are few-shot learners. *arXiv preprint arXiv:2005.14165*.

- Browne, C. B., Powley, E., Whitehouse, D., Lucas, S. M., Cowling, P. I., Rohlfshagen, P., Tavener, S., Perez, D., Samothrakis, S., and Colton, S. (2012). A survey of monte carlo tree search methods. *IEEE Transactions on Computational Intelligence and AI in games*, 4(1):1–43.
- Bu, J., Ratliff, L. J., and Mesbahi, M. (2019). Global convergence of policy gradient for sequential zero-sum linear quadratic dynamic games. *arXiv*, pages arXiv–1911.
- Buckdahn, R., Djehiche, B., and Li, J. (2011). A general stochastic maximum principle for sdes of mean-field type. *Applied Mathematics & Optimization*, 64(2):197–216.
- Burch, N., Lanctot, M., Szafron, D., and Gibson, R. G. (2012). Efficient monte carlo counterfactual regret minimization in games with many player actions. In *Advances in Neural Information Processing Systems*, pages 1880–1888.
- Busoniu, L., Babuska, R., and De Schutter, B. (2008). A comprehensive survey of multiagent reinforcement learning. *IEEE Trans. Systems, Man, and Cybernetics, Part C*, 38(2):156–172.
- Bušoniu, 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.
- Cai, Q., Yang, Z., Jin, C., and Wang, Z. (2019a). Provably efficient exploration in policy optimization. *arXiv preprint arXiv:1912.05830*.
- Cai, Q., Yang, Z., Lee, J. D., and Wang, Z. (2019b). Neural temporal-difference learning converges to global optima. In *Advances in Neural Information Processing Systems*, pages 11315–11326.
- Camazine, S. (2003). *Self-organization in biological systems*. Princeton University Press.
- Camerer, C. F., Ho, T.-H., and Chong, J.-K. (2002). Sophisticated experience-weighted attraction learning and strategic teaching in repeated games. *Journal of Economic theory*, 104(1):137–188.
- Camerer, C. F., Ho, T.-H., and Chong, J.-K. (2004). A cognitive hierarchy model of games. *The Quarterly Journal of Economics*.
- Campos-Rodriguez, R., Gonzalez-Jimenez, L., Cervantes-Alvarez, F., Amezcu-Garcia, F., and Fernandez-Garcia, M. (2017). Multiagent systems in automotive applications. *Multi-agent Systems*, page 43.
- Candogan, O., Menache, I., Ozdaglar, A., and Parrilo, P. A. (2011). Flows and decompositions of games: Harmonic and potential games. *Mathematics of Operations Research*, 36(3):474–503.

- Cao, Z., Qin, T., Liu, T.-Y., Tsai, M.-F., and Li, H. (2007). Learning to rank: from pairwise approach to listwise approach. In *Proceedings of the 24th international conference on Machine learning*, pages 129–136. ACM.
- Cardaliaguet, P., Delarue, F., Lasry, J.-M., and Lions, P.-L. (2015). The master equation and the convergence problem in mean field games. *arXiv preprint arXiv:1509.02505*.
- Cardaliaguet, P. and Hadikhanloo, S. (2017). Learning in mean field games: the fictitious play. *ESAIM: Control, Optimisation and Calculus of Variations*, 23(2):569–591.
- Cardaliaguet, P. and Lehalle, C.-A. (2018). Mean field game of controls and an application to trade crowding. *Mathematics and Financial Economics*, 12(3):335–363.
- Carmona, R. and Delarue, F. (2013). Probabilistic analysis of mean-field games. *SIAM Journal on Control and Optimization*, 51(4):2705–2734.
- Carmona, R., Delarue, F., et al. (2015a). Forward–backward stochastic differential equations and controlled mckean–vlasov dynamics. *The Annals of Probability*, 43(5):2647–2700.
- Carmona, R., Delarue, F., et al. (2018). *Probabilistic Theory of Mean Field Games with Applications I-II*. Springer.
- Carmona, R., Delarue, F., and Lachapelle, A. (2013). Control of mckean–vlasov dynamics versus mean field games. *Mathematics and Financial Economics*, 7(2):131–166.
- Carmona, R., Delarue, F., Lacker, D., et al. (2016). Mean field games with common noise. *The Annals of Probability*, 44(6):3740–3803.
- Carmona, R., Lacker, D., et al. (2015b). A probabilistic weak formulation of mean field games and applications. *The Annals of Applied Probability*, 25(3):1189–1231.
- Carmona, R., Laurière, M., and Tan, Z. (2019a). Linear-quadratic mean-field reinforcement learning: convergence of policy gradient methods. *arXiv preprint arXiv:1910.04295*.
- Carmona, R., Laurière, M., and Tan, Z. (2019b). Model-free mean-field reinforcement learning: mean-field mdp and mean-field q-learning. *arXiv preprint arXiv:1910.12802*.
- Cecchin, A., Pra, P. D., Fischer, M., and Pelino, G. (2019). On the convergence problem in mean field games: a two state model without uniqueness. *SIAM Journal on Control and Optimization*, 57(4):2443–2466.

- Cesa-Bianchi, N. and Lugosi, G. (2006). *Prediction, learning, and games*. Cambridge university press.
- Chalmers, C. (2020). Is reinforcement learning worth the hype? 2020. URL <https://www.capgemini.com/gb-en/2020/05/is-reinforcement-learning-worth-the-hype/>.
- Chang, M.-F., Lambert, J., Sangkloy, P., Singh, J., Bak, S., Hartnett, A., Wang, D., Carr, P., Lucey, S., Ramanan, D., et al. (2019). Argoverse: 3d tracking and forecasting with rich maps. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 8748–8757.
- Chasnov, B., Ratliff, L. J., Mazumdar, E., and Burden, S. A. (2019). Convergence analysis of gradient-based learning with non-uniform learning rates in non-cooperative multi-agent settings. *arXiv preprint arXiv:1906.00731*.
- Chen, X. and Deng, X. (2005). Settling the complexity of 2-player nash equilibrium. eccc. Technical report, Report.
- Chen, X. and Deng, X. (2006). Settling the complexity of two-player nash equilibrium. In *2006 47th Annual IEEE Symposium on Foundations of Computer Science (FOCS'06)*, pages 261–272. IEEE.
- Chen, X., Deng, X., and Teng, S.-H. (2009). Settling the complexity of computing two-player nash equilibria. *Journal of the ACM (JACM)*, 56(3):1–57.
- Cheung, W. C., Simchi-Levi, D., and Zhu, R. (2020). Reinforcement learning for non-stationary markov decision processes: The blessing of (more) optimism. *ICML*.
- Claus, C. and Boutilier, C. (1998a). The dynamics of reinforcement learning in cooperative multiagent systems. *AAAI/IAAI*, 1998:746–752.
- Claus, C. and Boutilier, C. (1998b). The dynamics of reinforcement learning in cooperative multiagent systems. *AAAI/IAAI*, 1998:746–752.
- Colby, M. K., Kharaghani, S., HolmesParker, C., and Tumer, K. (2015). Counterfactual exploration for improving multiagent learning. In *Proceedings of the 2015 International Conference on Autonomous Agents and Multiagent Systems*, pages 171–179. International Foundation for Autonomous Agents and Multiagent Systems.
- Conitzer, V. and Sandholm, T. (2002). Complexity results about nash equilibria. *arXiv preprint cs/0205074*.
- Conitzer, V. and Sandholm, T. (2007). Awesome: A general multiagent learning algorithm that converges in self-play and learns a best response against stationary opponents. *Machine Learning*, 67(1-2):23–43.

- Conitzer, V. and Sandholm, T. (2008). New complexity results about nash equilibria. *Games and Economic Behavior*, 63(2):621–641.
- Coppersmith, D. and Winograd, S. (1990). Matrix multiplication via arithmetic progressions. *Journal of symbolic computation*, 9(3):251–280.
- 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.
- Cowling, P. I., Powley, E. J., and Whitehouse, D. (2012). Information set monte carlo tree search. *IEEE Transactions on Computational Intelligence and AI in Games*, 4(2):120–143.
- Cully, A., Clune, J., Tarapore, D., and Mouret, J.-B. (2015). Robots that can adapt like animals. *Nature*, 521(7553):503–507.
- Czarnecki, W. M., Gidel, G., Tracey, B., Tuyls, K., Omidshafiei, S., Balduzzi, D., and Jaderberg, M. (2020). Real world games look like spinning tops. *arXiv*, pages arXiv–2004.
- Da Silva, F. L. and Costa, A. H. R. (2019). A survey on transfer learning for multiagent reinforcement learning systems. *Journal of Artificial Intelligence Research*, 64:645–703.
- Dall’Anese, E., Zhu, H., and Giannakis, G. B. (2013). Distributed optimal power flow for smart microgrids. *IEEE Transactions on Smart Grid*, 4(3):1464–1475.
- Dantzig, G. (1951). A proof of the equivalence of the programming problem and the game problem, in “activity analysis of production and allocation”(ed. tc koopmans), cowles commission monograph, no. 13.
- Dasari, S., Ebert, F., Tian, S., Nair, S., Bucher, B., Schmeckpeper, K., Singh, S., Levine, S., and Finn, C. (2019). Robonet: Large-scale multi-robot learning. *arXiv*, pages arXiv–1910.
- 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.
- Daskalakis, C., Ilyas, A., Syrgkanis, V., and Zeng, H. (2017). Training gans with optimism. *arXiv*, pages arXiv–1711.
- Daskalakis, C. and Panageas, I. (2018). The limit points of (optimistic) gradient descent in min-max optimization. In *Advances in Neural Information Processing Systems*, pages 9236–9246.
- Daskalakis, C. and Papadimitriou, C. H. (2005). Three-player games are hard. In *Electronic colloquium on computational complexity*, volume 139, pages 81–87.

- de Cote, E. M. and Littman, M. L. (2008). A polynomial-time nash equilibrium algorithm for repeated stochastic games. In McAllester, D. A. and Myllymäki, P., editors, *UAI 2008*, pages 419–426. AUAI Press.
- de Vries, H., Strub, F., Chandar, S., Pietquin, O., Larochelle, H., and Courville, A. (2016). GuessWhat?! Visual object discovery through multi-modal dialogue.
- Deisenroth, M. and Rasmussen, C. E. (2011). Pilco: A model-based and data-efficient approach to policy search. In *Proceedings of the 28th International Conference on machine learning (ICML-11)*, pages 465–472.
- Delarue, F. and Tchuendom, R. F. (2020). Selection of equilibria in a linear quadratic mean-field game. *Stochastic Processes and their Applications*, 130(2):1000–1040.
- Department, S. R. (2020). Forecast information technology (it) hardware spending worldwide from 2013 to 2019 (in billion u.s. dollars).
- Derakhshan, F. and Yousefi, S. (2019). A review on the applications of multiagent systems in wireless sensor networks. *International Journal of Distributed Sensor Networks*, 15(5):1550147719850767.
- Dermed, L. M. and Isbell, C. L. (2009). Solving stochastic games. In *Advances in Neural Information Processing Systems*, pages 1186–1194.
- Devlin, J., Chang, M.-W., Lee, K., and Toutanova, K. (2018). Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*.
- Dibangoye, J. and Buffet, O. (2018). Learning to act in decentralized partially observable mdps. In *International Conference on Machine Learning*, pages 1233–1242.
- Dibangoye, J. S., Amato, C., Buffet, O., and Charpillet, F. (2016). Optimally solving dec-pomdps as continuous-state mdps. *Journal of Artificial Intelligence Research*, 55:443–497.
- Dick, T., Gyorgy, A., and Szepesvari, C. (2014). Online learning in markov decision processes with changing cost sequences. In *International Conference on Machine Learning*, pages 512–520.
- Djete, M. F., Possamaï, D., and Tan, X. (2019). McKean-vlasov optimal control: the dynamic programming principle. *arXiv preprint arXiv:1907.08860*.
- Dosovitskiy, A., Ros, G., Codevilla, F., Lopez, A., and Koltun, V. (2017). Carla: An open urban driving simulator. *arXiv preprint arXiv:1711.03938*.
- Dresner, K. and Stone, P. (2004). Multiagent traffic management: a reservation-based intersection control mechanism. In *AAMAS 2004.*, pages 530–537.

- Dulac-Arnold, G., Levine, N., Mankowitz, D. J., Li, J., Paduraru, C., Gowal, S., and Hester, T. (2020). An empirical investigation of the challenges of real-world reinforcement learning. *arXiv preprint arXiv:2003.11881*.
- Dulac-Arnold, G., Mankowitz, D., and Hester, T. (2019). Challenges of real-world reinforcement learning. *arXiv preprint arXiv:1904.12901*.
- Dunbar, R. (2016). Group size, vocal grooming and the origins of language. *Psychonomic Bulletin & Review*, pages 1–4.
- Durham, W. H. (1991). *Coevolution: Genes, culture, and human diversity*. Stanford University Press.
- Efrati, A. (2020a). Waymo riders describe experiences on the road.
- Efrati, A. (2020b). Waymo's backseat drivers: Confidential data reveals self-driving taxi hurdles.
- Efrati, A. (2020c). Waymo's big ambitions slowed by tech trouble. <https://bit.ly/311Kwgt>.
- Elie, R., Pérolat, J., Laurière, M., Geist, M., and Pietquin, O. (2019). Approximate fictitious play for mean field games. *arXiv preprint arXiv:1907.02633*.
- Elie, R., Pérolat, J., Laurière, M., Geist, M., and Pietquin, O. (2020). On the convergence of model free learning in mean field games. In *AAAI*, pages 7143–7150.
- Erev, I. and Roth, A. E. (1998). Predicting how people play games: Reinforcement learning in experimental games with unique, mixed strategy equilibria. *American economic review*, pages 848–881.
- Even-Dar, E., Kakade, S. M., and Mansour, Y. (2005). Experts in a markov decision process. In *Advances in neural information processing systems*, pages 401–408.
- Even-Dar, E., Kakade, S. M., and Mansour, Y. (2009). Online markov decision processes. *Mathematics of Operations Research*, 34(3):726–736.
- Even-Dar, E. and Mansour, Y. (2003). Learning rates for q-learning. *Journal of machine learning Research*, 5(Dec):1–25.
- Eysenbach, B., Gupta, A., Ibarz, J., and Levine, S. (2018). Diversity is all you need: Learning skills without a reward function. In *International Conference on Learning Representations*.
- Fazel, M., Ge, R., Kakade, S., and Mesbahi, M. (2018). Global convergence of policy gradient methods for the linear quadratic regulator. In *International Conference on Machine Learning*, pages 1467–1476.
- Feinberg, E. A. (2010). Total expected discounted reward mdps: existence of optimal policies. *Wiley Encyclopedia of Operations Research and Management Science*.

- Fiez, T., Chasnov, B., and Ratliff, L. J. (2019). Convergence of learning dynamics in stackelberg games. *arXiv*, pages arXiv–1906.
- Filar, J. and Vrieze, K. (2012). *Competitive Markov decision processes*. Springer Science & Business Media.
- Fink, A. M. et al. (1964). Equilibrium in a stochastic n -person game. *Journal of science of the hiroshima university, series ai (mathematics)*, 28(1):89–93.
- Finn, C., Abbeel, P., and Levine, S. (2017). Model-agnostic meta-learning for fast adaptation of deep networks. In *ICML*.
- Fischer, M. et al. (2017). On the connection between symmetric n -player games and mean field games. *The Annals of Applied Probability*, 27(2):757–810.
- Florensa, C., Duan, Y., and Abbeel, P. (2017). Stochastic neural networks for hierarchical reinforcement learning. *arXiv*, pages arXiv–1704.
- 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.
- Foerster, J., Farquhar, G., Afouras, T., Nardelli, N., and Whiteson, S. (2017a). Counterfactual multi-agent policy gradients. *arXiv preprint arXiv:1705.08926*.
- Foerster, J., Nardelli, N., Farquhar, G., Afouras, T., Torr, P. H., Kohli, P., and Whiteson, S. (2017b). Stabilising experience replay for deep multi-agent reinforcement learning. In *Proceedings of the 34th International Conference on Machine Learning-Volume 70*, pages 1146–1155.
- Foerster, J. N., Chen, R. Y., Al-Shedivat, M., Whiteson, S., Abbeel, P., and Mordatch, I. (2017c). Learning with opponent-learning awareness. *CoRR*, abs/1709.04326.
- Foerster, J. N., Farquhar, G., Afouras, T., Nardelli, N., and Whiteson, S. (2018b). Counterfactual multi-agent policy gradients. In [McIlraith and Weinberger \(2018\)](#).
- Fogel, D. B. (2006). *Evolutionary computation: toward a new philosophy of machine intelligence*, volume 1. John Wiley & Sons.
- Freund, Y. and Schapire, R. E. (1997). A decision-theoretic generalization of on-line learning and an application to boosting. *Journal of computer and system sciences*, 55(1):119–139.
- Fryxell, J. M., Mosser, A., Sinclair, A. R., and Packer, C. (2007). Group formation stabilizes predator–prey dynamics. *Nature*, 449(7165):1041–1043.
- Fu, Z., Yang, Z., Chen, Y., and Wang, Z. (2019). Actor-critic provably finds nash equilibria of linear-quadratic mean-field games. *arXiv preprint arXiv:1910.07498*.

- Fudenberg, D., Drew, F., Levine, D. K., and Levine, D. K. (1998). *The theory of learning in games*, volume 2. MIT press.
- Fudenberg, D. and Imhof, L. A. (2006). Imitation processes with small mutations. *Journal of Economic Theory*, 131(1):251–262.
- Fudenberg, D. and Kreps, D. M. (1993). Learning mixed equilibria. *Games and economic behavior*, 5(3):320–367.
- Fudenberg, D. and Levine, D. (1995). Consistency and cautious fictitious play. *Journal of Economic Dynamics and Control*.
- Galam, S. and Walliser, B. (2010). Ising model versus normal form game. *Physica A: Statistical Mechanics and its Applications*, 389(3):481–489.
- Ganapathi Subramanian, S., Taylor, M. E., Crowley, M., and Poupart, P. (2020). Partially observable mean field reinforcement learning. *arXiv e-prints*, pages arXiv–2012.
- Garber, D., Hazan, E., and Ma, T. (2015). Online learning of eigenvectors. In *ICML*, pages 560–568.
- García, J. and Fernández, F. (2015). A comprehensive survey on safe reinforcement learning. *Journal of Machine Learning Research*, 16(1):1437–1480.
- Gärtner, J. (1988). On the mckean-vlasov limit for interacting diffusions. *Mathematische Nachrichten*, 137(1):197–248.
- Gasser, R. and Huhns, M. N. (2014). *Distributed Artificial Intelligence: Volume II*, volume 2. Morgan Kaufmann.
- Geiger, A., Lenz, P., Stiller, C., and Urtasun, R. (2013). Vision meets robotics: The kitti dataset. *The International Journal of Robotics Research*, 32(11):1231–1237.
- Gibson, R. G., Lanctot, M., Burch, N., Szafron, D., and Bowling, M. (2012). Generalized sampling and variance in counterfactual regret minimization. In *AAAI*.
- Gigerenzer, G. and Selten, R. (2002). *Bounded rationality: The adaptive toolbox*. MIT press.
- Gilpin, A., Hoda, S., Pena, J., and Sandholm, T. (2007). Gradient-based algorithms for finding nash equilibria in extensive form games. In *International Workshop on Web and Internet Economics*, pages 57–69. Springer.
- Gilpin, A. and Sandholm, T. (2006). Finding equilibria in large sequential games of imperfect information. In *Proceedings of the 7th ACM conference on Electronic commerce*, pages 160–169.
- Gilpin, M. E. (1973). Do hares eat lynx? *The American Naturalist*, 107(957):727–730.

- Gilpin, M. E. (1975). Limit cycles in competition communities. *The American Naturalist*, 109(965):51–60.
- Golub, G. H. and Van der Vorst, H. A. (2000). Eigenvalue computation in the 20th century. *Journal of Computational and Applied Mathematics*, 123(1-2):35–65.
- González-Sánchez, D. and Hernández-Lerma, O. (2013). *Discrete-time stochastic control and dynamic potential games: the Euler–Equation approach*. Springer Science & Business Media.
- Goodfellow, I., Pouget-Abadie, J., Mirza, M., Xu, B., Warde-Farley, D., Ozair, S., Courville, A., and Bengio, Y. (2014a). Generative adversarial nets. In *NIPS*, pages 2672–2680.
- Goodfellow, I. J., Shlens, J., and Szegedy, C. (2014b). Explaining and harnessing adversarial examples. *arXiv preprint arXiv:1412.6572*.
- Gordon, G. J. (2007). No-regret algorithms for online convex programs. In *Advances in Neural Information Processing Systems*, pages 489–496.
- 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 20, page 242.
- Gregor, K., Rezende, D. J., and Wierstra, D. (2017). Variational intrinsic control.
- Gu, H., Guo, X., Wei, X., and Xu, R. (2019). Dynamic programming principles for learning mfcfs. *arXiv preprint arXiv:1911.07314*.
- Gu, H., Guo, X., Wei, X., and Xu, R. (2020). Q-learning for mean-field controls. *arXiv preprint arXiv:2002.04131*.
- Guéant, O., Lasry, J.-M., and Lions, P.-L. (2011). Mean field games and applications. In *Paris-Princeton lectures on mathematical finance 2010*, pages 205–266. Springer.
- Guestrin, C., Koller, D., and Parr, R. (2002a). Multiagent planning with factored mdps. In *Advances in neural information processing systems*, pages 1523–1530.
- Guestrin, C., Lagoudakis, M., and Parr, R. (2002b). Coordinated reinforcement learning. In *ICML*, volume 2, pages 227–234. Citeseer.
- Guillen, M. F. (2000). Business groups in emerging economies: A resource-based view. *Academy of Management Journal*, 43(3):362–380.
- Guo, X., Hu, A., Xu, R., and Zhang, J. (2019). Learning mean-field games. In *Advances in Neural Information Processing Systems*, pages 4966–4976.

- Guo, X., Hu, A., Xu, R., and Zhang, J. (2020). A general framework for learning mean-field games. *arXiv preprint arXiv:2003.06069*.
- Gupta, J. K., Egorov, M., and Kochenderfer, M. (2017). Cooperative multi-agent control using deep reinforcement learning. In *AAMAS*, pages 66–83. Springer.
- Haarnoja, T., Tang, H., Abbeel, P., and Levine, S. (2017). Reinforcement learning with deep energy-based policies. In *ICML*.
- 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 *International Conference on Machine Learning*, pages 1861–1870.
- Hadikhanloo, S. and Silva, F. J. (2019). Finite mean field games: fictitious play and convergence to a first order continuous mean field game. *Journal de Mathématiques Pures et Appliquées*, 132:369–397.
- Hafner, D., Lillicrap, T., Ba, J., and Norouzi, M. (2019a). Dream to control: Learning behaviors by latent imagination. In *International Conference on Learning Representations*.
- Hafner, D., Lillicrap, T., Fischer, I., Villegas, R., Ha, D., Lee, H., and Davidson, J. (2019b). Learning latent dynamics for planning from pixels. In *International Conference on Machine Learning*, pages 2555–2565. PMLR.
- Han, L., Xiong, J., Sun, P., Sun, X., Fang, M., Guo, Q., Chen, Q., Shi, T., Yu, H., and Zhang, Z. (2020). Tstarbot-x: An open-sourced and comprehensive study for efficient league training in starcraft ii full game. *arXiv preprint arXiv:2011.13729*.
- Hannan, J. (1957). Approximation to bayes risk in repeated play. *Contributions to the Theory of Games*, 3:97–139.
- 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.
- Hansen, T. D., Miltersen, P. B., and Zwick, U. (2013). Strategy iteration is strongly polynomial for 2-player turn-based stochastic games with a constant discount factor. *Journal of the ACM (JACM)*, 60(1):1–16.
- Hart, S. (2013). *Simple adaptive strategies: from regret-matching to uncoupled dynamics*, volume 4. World Scientific.
- Hart, S. and Mas-Colell, A. (2001). A reinforcement procedure leading to correlated equilibrium. In *Economics Essays*, pages 181–200. Springer.
- Hart, S. and Mas-Colell, A. (2003). Uncoupled dynamics do not lead to nash equilibrium. *American Economic Review*, 93(5):1830–1836.
- Hasselt, H. V. (2010). Double q-learning. In *NIPS*, pages 2613–2621.

- Hausman, K., Springenberg, J. T., Wang, Z., Heess, N., and Riedmiller, M. (2018). Learning an embedding space for transferable robot skills. In *International Conference on Learning Representations*.
- He, H. and Boyd-Graber, J. L. (2016). Opponent modeling in deep reinforcement learning. In Balcan, M. and Weinberger, K. Q., editors, *ICML*, volume 48, pages 1804–1813. JMLR.org.
- Heinrich, J., Lanctot, M., and Silver, D. (2015). Fictitious self-play in extensive-form games. In *ICML*, pages 805–813.
- Heinrich, J. and Silver, D. (2016). Deep reinforcement learning from self-play in imperfect-information games. *arXiv preprint arXiv:1603.01121*.
- Hennes, D., Morrill, D., Omidshafiei, S., Munos, R., Perolat, J., Lanctot, M., Gruslys, A., Lespiau, J.-B., Parmas, P., Duenez-Guzman, E., et al. (2019). Neural replicator dynamics. *arXiv preprint arXiv:1906.00190*.
- Herings, P. J.-J. and Peeters, R. (2010). Homotopy methods to compute equilibria in game theory. *Economic Theory*, 42(1):119–156.
- Herings, P. J.-J., Peeters, R. J., et al. (2004). Stationary equilibria in stochastic games: Structure, selection, and computation. *Journal of Economic Theory*, 118(1):32–60.
- 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). A survey and critique of multiagent deep reinforcement learning. *Autonomous Agents and Multi-Agent Systems*, 33(6):750–797.
- Hernández-Orallo, J. (2017). *The measure of all minds: evaluating natural and artificial intelligence*. Cambridge University Press.
- Heusel, M., Ramsauer, H., Unterthiner, T., Nessler, B., and Hochreiter, S. (2017). Gans trained by a two time-scale update rule converge to a local nash equilibrium. In *Advances in neural information processing systems*, pages 6626–6637.
- Hirsch, M. W. (2012). *Differential topology*, volume 33. Springer Science & Business Media.
- Hofbauer, J. and Sandholm, W. H. (2002). On the global convergence of stochastic fictitious play. *Econometrica*, 70(6):2265–2294.
- Holland, J. H. et al. (1992). *Adaptation in natural and artificial systems: an introductory analysis with applications to biology, control, and artificial intelligence*. MIT press.

- HolmesParker, C., Taylor, M., Zhan, Y., and Tumer, K. (2014). Exploiting structure and agent-centric rewards to promote coordination in large multiagent systems. In *Adaptive and Learning Agents Workshop*.
- Hong, L. and Page, S. E. (2004). Groups of diverse problem solvers can outperform groups of high-ability problem solvers. *Proceedings of the National Academy of Sciences*, 101(46):16385–16389.
- Hu, J. and Wellman, M. P. (2003). Nash q-learning for general-sum stochastic games. *Journal of Machine learning research*, 4(Nov):1039–1069.
- Hu, J., Wellman, M. P., et al. (1998). Multiagent reinforcement learning: theoretical framework and an algorithm. In *ICML*, volume 98, pages 242–250.
- Hu, K., Ren, Z., Siska, D., and Szpruch, L. (2019). Mean-field langevin dynamics and energy landscape of neural networks. *arXiv preprint arXiv:1905.07769*.
- Huang, M., Malhamé, R. P., Caines, P. E., et al. (2006). Large population stochastic dynamic games: closed-loop mckean-vlasov systems and the nash certainty equivalence principle. *Communications in Information & Systems*, 6(3):221–252.
- Hughes, E., Leibo, J. Z., Phillips, M., Tuyls, K., Dueñez-Guzman, E., Castañeda, A. G., Dunning, I., Zhu, T., McKee, K., Koster, R., et al. (2018). Inequity aversion improves cooperation in intertemporal social dilemmas. In *Advances in neural information processing systems*, pages 3326–3336.
- Huhns, M. N. (2012). *Distributed Artificial Intelligence: Volume I*, volume 1. Elsevier.
- Hussein, A., Gaber, M. M., Elyan, E., and Jayne, C. (2017). Imitation learning: A survey of learning methods. *ACM Computing Surveys (CSUR)*, 50(2):1–35.
- Inada, Y. and Kawachi, K. (2002). Order and flexibility in the motion of fish schools. *Journal of theoretical Biology*, 214(3):371–387.
- International, S. (2014). Automated driving levels of driving automation are defined in new sae international standard j3016.
- Ising, E. (1925). Beitrag zur theorie des ferromagnetismus. *Zeitschrift für Physik*, 31(1):253–258.
- Iyer, K., Johari, R., and Sundararajan, M. (2014). Mean field equilibria of dynamic auctions with learning. *Management Science*, 60(12):2949–2970.
- Jaakkola, T., Jordan, M. I., and Singh, S. P. (1994). Convergence of stochastic iterative dynamic programming algorithms. In *NIPS*, pages 703–710.
- Jaderberg, M., Czarnecki, W. M., Dunning, I., Marrs, L., Lever, G., Castaneda, A. G., Beattie, C., Rabinowitz, N. C., Morcos, A. S., Ruderman, A., et al. (2019). Human-level performance in 3d multiplayer games with population-based reinforcement learning. *Science*, 364(6443):859–865.

- Jan't Hoen, P., Tuyls, K., Panait, L., Luke, S., and La Poutre, J. A. (2005). An overview of cooperative and competitive multiagent learning. In *International Workshop on Learning and Adaption in Multi-Agent Systems*, pages 1–46. Springer.
- Javarone, M. A. and Marinazzo, D. (2016). Evolutionary dynamics of group formation. *arXiv preprint arXiv:1612.03834*.
- Jeanne, R. L. (1988). *Interindividual behavioral variability in social insects*. Westview Press.
- Jeong, S. H., Kang, A. R., and Kim, H. K. (2015). Analysis of game bot's behavioral characteristics in social interaction networks of mmorpg. In *ACM SIGCOMM Computer Communication Review*, volume 45, pages 99–100. ACM.
- Jia, Z., Yang, L. F., and Wang, M. (2019). Feature-based q-learning for two-player stochastic games. *arXiv*, pages arXiv–1906.
- Jin, C., Netrapalli, P., and Jordan, M. I. (2019). What is local optimality in nonconvex-nonconcave minimax optimization? *arXiv*, pages arXiv–1902.
- Jin, P., Keutzer, K., and Levine, S. (2018). Regret minimization for partially observable deep reinforcement learning. In *International Conference on Machine Learning*, pages 2342–2351.
- Johanson, M., Bard, N., Burch, N., and Bowling, M. (2012a). Finding optimal abstract strategies in extensive-form games. In *AAAI*. Citeseer.
- Johanson, M., Bard, N., Lanctot, M., Gibson, R. G., and Bowling, M. (2012b). Efficient nash equilibrium approximation through monte carlo counterfactual regret minimization. In *AAMAS*, pages 837–846.
- Jovanovic, B. and Rosenthal, R. W. (1988). Anonymous sequential games. *Journal of Mathematical Economics*, 17(1):77–87.
- Kadanoff, L. P. (2009). More is the same; phase transitions and mean field theories. *Journal of Statistical Physics*, 137(5-6):777.
- Kaelbling, L. P., Littman, M. L., and Moore, A. W. (1996). Reinforcement learning: A survey. *Journal of artificial intelligence research*, 4:237–285.
- Kaniovski, Y. M. and Young, H. P. (1995). Learning dynamics in games with stochastic perturbations. *Games and economic behavior*, 11(2):330–363.
- Kapetanakis, S. and Kudenko, D. (2002). Reinforcement learning of coordination in cooperative multi-agent systems. In *NCAI*, pages 326–331, Menlo Park, CA, USA.
- Kauffman, S. A. (1993). *The origins of order: Self-organization and selection in evolution*. Oxford University Press, USA.

- Kearns, M. (2007). Graphical games. *Algorithmic game theory*, 3:159–180.
- Kearns, M., Littman, M. L., and Singh, S. (2013). Graphical models for game theory. *arXiv preprint arXiv:1301.2281*.
- Kennedy, J. (2006). Swarm intelligence. In *Handbook of nature-inspired and innovative computing*, pages 187–219. Springer.
- Kerr, B., Riley, M. A., Feldman, M. W., and Bohannan, B. J. (2002). Local dispersal promotes biodiversity in a real-life game of rock–paper–scissors. *Nature*, 418(6894):171–174.
- Keynes, J. M. (1936). *The General Theory of Employment, Interest and Money*. Macmillan. 14th edition, 1973.
- Kingma, D. P. and Ba, J. (2014). Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*.
- Kirkpatrick, J., Pascanu, R., Rabinowitz, N., Veness, J., Desjardins, G., Rusu, A. A., Milan, K., Quan, J., Ramalho, T., Grabska-Barwinska, A., et al. (2017). Overcoming catastrophic forgetting in neural networks. *Proceedings of the national academy of sciences*, 114(13):3521–3526.
- Klopf, A. H. (1972). *Brain function and adaptive systems: a heterostatic theory*. Number 133. Air Force Cambridge Research Laboratories, Air Force Systems Command, United
- Kober, J., Bagnell, J. A., and Peters, J. (2013). Reinforcement learning in robotics: A survey. *The International Journal of Robotics Research*, 32(11):1238–1274.
- Kocsis, L. and Szepesvári, C. (2006). Bandit based monte-carlo planning. In *European conference on machine learning*, pages 282–293. Springer.
- Kok, J. R. and Vlassis, N. (2004). Sparse cooperative q-learning. In *Proceedings of the twenty-first international conference on Machine learning*, page 61.
- Koller, D. and Megiddo, N. (1992). The complexity of two-person zero-sum games in extensive form. *Games and economic behavior*, 4(4):528–552.
- Koller, D. and Megiddo, N. (1996). Finding mixed strategies with small supports in extensive form games. *International Journal of Game Theory*, 25(1):73–92.
- Konda, V. R. and Tsitsiklis, J. N. (2000a). Actor-critic algorithms. In *Advances in neural information processing systems*, pages 1008–1014.
- Konda, V. R. and Tsitsiklis, J. N. (2000b). Actor-critic algorithms. In Solla, S. A., Leen, T. K., and Müller, K., editors, *Advances in Neural Information Processing Systems 12*, pages 1008–1014. MIT Press.

- Kong, W. and Monteiro, R. D. (2019). An accelerated inexact proximal point method for solving nonconvex-concave min-max problems. *arXiv*, pages arXiv–1905.
- Kovařík, V., Schmid, M., Burch, N., Bowling, M., and Lisý, V. (2019). Rethinking formal models of partially observable multiagent decision making. *arXiv preprint arXiv:1906.11110*.
- Krajewski, R., Bock, J., Kloeker, L., and Eckstein, L. (2018). The highd dataset: A drone dataset of naturalistic vehicle trajectories on german highways for validation of highly automated driving systems. In *2018 21st International Conference on Intelligent Transportation Systems (ITSC)*, pages 2118–2125. IEEE.
- Krajzewicz, D., Hertkorn, G., Rössel, C., and Wagner, P. (2002). SUMO (Simulation of Urban MObility)—an open-source traffic simulation. In *MESM*, pages 183–187.
- Kreps, D. M. and Wilson, R. (1982). Reputation and imperfect information. *Journal of economic theory*, 27(2):253–279.
- Kreyszig, E. (1978). *Introductory functional analysis with applications*, volume 1. wiley New York.
- 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.
- Kuhn, H. W. (1950a). Extensive games. *Proceedings of the National Academy of Sciences of the United States of America*, 36(10):570.
- Kuhn, H. W. (1950b). A simplified two-person poker. *Contributions to the Theory of Games*, 1:97–103.
- Kulesza, A., Taskar, B., et al. (2012). Determinantal point processes for machine learning. *Foundations and Trends® in Machine Learning*, 5(2–3):123–286.
- Lăă, Q. D., Chew, Y. H., and Soong, B.-H. (2016). *Potential Game Theory*. Springer.
- Lacker, D. (2015). Mean field games via controlled martingale problems: existence of markovian equilibria. *Stochastic Processes and their Applications*, 125(7):2856–2894.
- Lacker, D. (2017). Limit theory for controlled mckean–vlasov dynamics. *SIAM Journal on Control and Optimization*, 55(3):1641–1672.
- Lacker, D. (2018). On the convergence of closed-loop nash equilibria to the mean field game limit. *arXiv preprint arXiv:1808.02745*.
- Lacker, D. and Zariphopoulou, T. (2019). Mean field and n-agent games for optimal investment under relative performance criteria. *Mathematical Finance*, 29(4):1003–1038.

- Lagoudakis, M. G. and Parr, R. (2003). Learning in zero-sum team markov games using factored value functions. In *Advances in Neural Information Processing Systems*, pages 1659–1666.
- Lanctot, M., Lockhart, E., Lespiau, J.-B., Zambaldi, V., Upadhyay, S., Pérolat, J., Srinivasan, S., Timbers, F., Tuyls, K., Omidshafiei, S., et al. (2019). Open-spiel: A framework for reinforcement learning in games. *arXiv preprint arXiv:1908.09453*.
- Lanctot, M., Waugh, K., Zinkevich, M., and Bowling, M. (2009). Monte carlo sampling for regret minimization in extensive games. In *Advances in neural information processing systems*, pages 1078–1086.
- Lanctot, M., Zambaldi, V., Gruslys, A., Lazaridou, A., Tuyls, K., Pérolat, J., Silver, D., and Graepel, T. (2017). A unified game-theoretic approach to multiagent reinforcement learning. In *Advances in neural information processing systems*, pages 4190–4203.
- Lange, S., Gabel, T., and Riedmiller, M. (2012). Batch reinforcement learning. In *Reinforcement learning*, pages 45–73. Springer.
- Lasry, J.-M. and Lions, P.-L. (2007). Mean field games. *Japanese journal of mathematics*, 2(1):229–260.
- Lauer, M. and Riedmiller, M. (2000). An algorithm for distributed reinforcement learning in cooperative multi-agent systems. In *ICML*. Citeseer.
- Laurière, M. and Pironneau, O. (2014). Dynamic programming for mean-field type control. *Comptes Rendus Mathematique*, 352(9):707–713.
- Lazaridou, A., Peysakhovich, A., and Baroni, M. (2016). Multi-agent cooperation and the emergence of (natural) language. *CoRR*, abs/1612.07182.
- LeCun, Y., Bengio, Y., and Hinton, G. (2015). Deep learning. *nature*, 521(7553):436–444.
- Lehalle, C.-A. and Mouzouni, C. (2019). A mean field game of portfolio trading and its consequences on perceived correlations. *arXiv preprint arXiv:1902.09606*.
- Lehman, J. and Stanley, K. O. (2008). Exploiting open-endedness to solve problems through the search for novelty. In *ALIFE*, pages 329–336.
- Lehman, J. and Stanley, K. O. (2011a). Abandoning objectives: Evolution through the search for novelty alone. *Evolutionary computation*, 19(2):189–223.
- Lehman, J. and Stanley, K. O. (2011b). Evolving a diversity of virtual creatures through novelty search and local competition. In *Proceedings of the 13th annual conference on Genetic and evolutionary computation*, pages 211–218.

- Leibfried, F., Grau-Moya, J., and Bou-Ammar, H. (2017). An information-theoretic optimality principle for deep reinforcement learning. *CoRR*, abs/1708.01867.
- Leibo, J. Z., Hughes, E., Lanctot, M., and Graepel, T. (2019). Autocurricula and the emergence of innovation from social interaction: A manifesto for multi-agent intelligence research. *arXiv*, pages arXiv–1903.
- Leibo, J. Z., Zambaldi, V., Lanctot, M., Marecki, J., and Graepel, T. (2017). Multi-agent reinforcement learning in sequential social dilemmas. *arXiv preprint arXiv:1702.03037*.
- Lemke, C. E. and Howson, Jr, J. T. (1964). Equilibrium points of bimatrix games. *Journal of the Society for Industrial and Applied Mathematics*, 12(2):413–423.
- 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.
- Leurent, E. (2018). An environment for autonomous driving decision-making. <https://github.com/eleurent/highway-env>.
- Levine, S. (2018). Reinforcement learning and control as probabilistic inference: Tutorial and review. *arXiv preprint arXiv:1805.00909*.
- Levine, S., Kumar, A., Tucker, G., and Fu, J. (2020). Offline reinforcement learning: Tutorial, review, and perspectives on open problems. *arXiv preprint arXiv:2005.01643*.
- Leyton-Brown, K. and Tennenholtz, M. (2005). Local-effect games. In *Dagstuhl Seminar Proceedings*. Schloss Dagstuhl-Leibniz-Zentrum für Informatik.
- Li, M., Qin, Z., Jiao, Y., Yang, Y., Wang, J., Wang, C., Wu, G., and Ye, J. (2019a). Efficient ridesharing order dispatching with mean field multi-agent reinforcement learning. In *The World Wide Web Conference*, pages 983–994.
- Li, S., Wu, Y., Cui, X., Dong, H., Fang, F., and Russell, S. (2019b). Robust multi-agent reinforcement learning via minimax deep deterministic policy gradient. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 33, pages 4213–4220.
- Li, Y. (2017). Deep reinforcement learning: An overview. *arXiv preprint arXiv:1701.07274*.

- Li, Z. and Tewari, A. (2018). Sampled fictitious play is hannan consistent. *Games and Economic Behavior*, 109:401–412.
- Liang, E., Liaw, R., Nishihara, R., Moritz, P., Fox, R., Goldberg, K., Gonzalez, J., Jordan, M., and Stoica, I. (2018). Rllib: Abstractions for distributed reinforcement learning. In *ICML*, pages 3053–3062.
- 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., Jin, C., and Jordan, M. I. (2019). On gradient descent ascent for nonconvex-concave minimax problems. *arXiv preprint arXiv:1906.00331*.
- Lisy, V., Kovarik, V., Lanctot, M., and Bosansky, B. (2013). Convergence of monte carlo tree search in simultaneous move games. In *Advances in Neural Information Processing Systems*, pages 2112–2120.
- Littlestone, N. and Warmuth, M. K. (1994). The weighted majority algorithm. *Information and computation*, 108(2):212–261.
- Littman, M. L. (1994). Markov games as a framework for multi-agent reinforcement learning. In *Proceedings of the eleventh international conference on machine learning*, volume 157, pages 157–163.
- Littman, M. L. (2001a). Friend-or-foe q-learning in general-sum games. In *ICML*, volume 1, pages 322–328.
- Littman, M. L. (2001b). Value-function reinforcement learning in markov games. *Cognitive Systems Research*, 2(1):55–66.
- Littman, M. L. and Stone, P. (2005). A polynomial-time nash equilibrium algorithm for repeated games. *Decision Support Systems*, 39(1):55–66.
- Liu, B., Cai, Q., Yang, Z., and Wang, Z. (2019a). Neural proximal/trust region policy optimization attains globally optimal policy. *arXiv preprint arXiv:1906.10306*.
- Liu, S., Lever, G., Merel, J., Tunyasuvunakool, S., Heess, N., and Graepel, T. (2018). Emergent coordination through competition. In *International Conference on Learning Representations*.
- Liu, S., Lever, G., Merel, J., Tunyasuvunakool, S., Heess, N., and Graepel, T. (2019b). Emergent coordination through competition. *arXiv preprint arXiv:1902.07151*.
- Lockhart, E., Lanctot, M., Pérolat, J., Lespiau, J.-B., Morrill, D., Timbers, F., and Tuyls, K. (2019). Computing approximate equilibria in sequential adversarial games by exploitability descent. *arXiv preprint arXiv:1903.05614*.

- Lotka, A. J. (1925). Elements of physical biology.
- Lowe, R., Wu, Y., Tamar, A., Harb, J., Abbeel, O. P., and Mordatch, I. (2017a). Multi-agent actor-critic for mixed cooperative-competitive environments. In *NIPS*, pages 6382–6393.
- Lowe, R., Wu, Y., Tamar, A., Harb, J., Abbeel, P., and Mordatch, I. (2017b). Multi-agent actor-critic for mixed cooperative-competitive environments. In Guyon, I., von Luxburg, U., Bengio, S., Wallach, H. M., Fergus, R., Vishwanathan, S. V. N., and Garnett, R., editors, *NIPS*, pages 6382–6393.
- Lu, S., Tsaknakis, I., Hong, M., and Chen, Y. (2020a). Hybrid block successive approximation for one-sided non-convex min-max problems: algorithms and applications. *IEEE Transactions on Signal Processing*.
- Lu, Y., Ma, C., Lu, Y., Lu, J., and Ying, L. (2020b). A mean-field analysis of deep resnet and beyond: Towards provable optimization via overparameterization from depth. *arXiv preprint arXiv:2003.05508*.
- Luo, Y., Yang, Z., Wang, Z., and Kolar, M. (2019). Natural actor-critic converges globally for hierarchical linear quadratic regulator. *arXiv preprint arXiv:1912.06875*.
- 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.
- Macua, S. V., Zazo, J., and Zazo, S. (2018). Learning parametric closed-loop policies for markov potential games. In *International Conference on Learning Representations*.
- Maddern, W., Pascoe, G., Linegar, C., and Newman, P. (2017). 1 year, 1000 km: The oxford robotcar dataset. *The International Journal of Robotics Research*, 36(1):3–15.
- Mahadevan, S. (1996). Average reward reinforcement learning: Foundations, algorithms, and empirical results. *Machine learning*, 22(1-3):159–195.
- Mahajan, A., Rashid, T., Samvelyan, M., and Whiteson, S. (2019). Maven: Multi-agent variational exploration. In *Advances in Neural Information Processing Systems*, pages 7613–7624.
- Mannor, S. and Shimkin, N. (2003). The empirical bayes envelope and regret minimization in competitive markov decision processes. *Mathematics of Operations Research*, 28(2):327–345.
- Maskin, E. and Tirole, J. (2001). Markov perfect equilibrium: I. observable actions. *Journal of Economic Theory*, 100(2):191–219.

- Matas, J., James, S., and Davison, A. J. (2018). Sim-to-real reinforcement learning for deformable object manipulation. *arXiv preprint arXiv:1806.07851*.
- Mathew, S. (2017). How the second-order free rider problem is solved in a small-scale society. *American Economic Review*, 107(5):578–81.
- Matignon, L., Laurent, G. J., and Le Fort-Piat, N. (2012). Independent reinforcement learners in cooperative markov games: a survey regarding coordination problems. *The Knowledge Engineering Review*, 27(1):1–31.
- Matousek, J. and Gärtner, B. (2007). *Understanding and using linear programming*. Springer Science & Business Media.
- Maynard Smith, J. (1972). On evolution.
- Mazumdar, E. and Ratliff, L. J. (2018). On the convergence of gradient-based learning in continuous games. *arXiv*, pages arXiv–1804.
- Mazumdar, E., Ratliff, L. J., Sastry, S., and Jordan, M. I. (2019a). Policy gradient in linear quadratic dynamic games has no convergence guarantees. Smooth Games Optimization and Machine Learning Workshop: Bridging Game
- Mazumdar, E. V., Jordan, M. I., and Sastry, S. S. (2019b). On finding local nash equilibria (and only local nash equilibria) in zero-sum games. *arXiv preprint arXiv:1901.00838*.
- McAleer, S., Lanier, J., Fox, R., and Baldi, P. (2020). Pipeline psro: A scalable approach for finding approximate nash equilibria in large games. *arXiv preprint arXiv:2006.08555*.
- McIlraith, S. A. and Weinberger, K. Q., editors (2018). *Proceedings of the Thirty-Second AAAI Conference on Artificial Intelligence, New Orleans, Louisiana, USA, February 2-7, 2018*. AAAI Press.
- McKean, H. P. (1967). Propagation of chaos for a class of non-linear parabolic equations. *Stochastic Differential Equations (Lecture Series in Differential Equations, Session 7, Catholic Univ., 1967)*, pages 41–57.
- McMahan, H. B., Gordon, G. J., and Blum, A. (2003). Planning in the presence of cost functions controlled by an adversary. In *Proceedings of the 20th International Conference on Machine Learning (ICML-03)*, pages 536–543.
- Melo, F. S., Meyn, S. P., and Ribeiro, M. I. (2008). An analysis of reinforcement learning with function approximation. In *Proceedings of the 25th international conference on Machine learning*, pages 664–671. ACM.
- Mertikopoulos, P., Lecouat, B., Zenati, H., Foo, C.-S., Chandrasekhar, V., and Pilipouras, G. (2018). Optimistic mirror descent in saddle-point problems: Going the extra (gradient) mile. In *International Conference on Learning Representations*.

- Mertikopoulos, P. and Zhou, Z. (2019). Learning in games with continuous action sets and unknown payoff functions. *Mathematical Programming*, 173(1-2):465–507.
- Mescheder, L., Geiger, A., and Nowozin, S. (2018). Which training methods for gans do actually converge? *arXiv preprint arXiv:1801.04406*.
- Mescheder, L., Nowozin, S., and Geiger, A. (2017). The numerics of gans. In *Advances in Neural Information Processing Systems*, pages 1825–1835.
- Mguni, D. (2020). Stochastic potential games. *arXiv preprint arXiv:2005.13527*.
- Michael, D. (2020). Algorithmic game theory lecture notes. <http://www.cs.jhu.edu/~mdinitz/classes/AGT/Spring2020/Lectures/lecture6.pdf>.
- Minsky, M. (1961). Steps toward artificial intelligence. *Proceedings of the IRE*, 49(1):8–30.
- Minsky, M. L. (1954). *Theory of neural-analog reinforcement systems and its application to the brain model problem*. Princeton University.
- Mnih, V., Badia, A. P., Mirza, M., Graves, A., Lillicrap, T., Harley, T., Silver, D., and Kavukcuoglu, K. (2016). Asynchronous methods for deep reinforcement learning. In *International Conference on Machine Learning*, pages 1928–1937.
- Mnih, V., Kavukcuoglu, K., Silver, D., Graves, A., Antonoglou, I., Wierstra, D., and Riedmiller, M. (2013). Playing atari with deep reinforcement learning. *arXiv preprint arXiv:1312.5602*.
- 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):529–533.
- Monderer, D. and Shapley, L. S. (1996). Potential games. *Games and economic behavior*, 14(1):124–143.
- Moravčík, M., Schmid, M., Burch, N., Lisý, V., Morrill, D., Bard, N., Davis, T., Waugh, K., Johanson, M., and Bowling, M. (2017). Deepstack: Expert-level artificial intelligence in heads-up no-limit poker. *Science*, 356(6337):508–513.
- Mordatch, I. and Abbeel, P. (2017). Emergence of Grounded Compositional Language in Multi-Agent Populations.
- Morimoto, J. and Doya, K. (2005). Robust reinforcement learning. *Neural computation*, 17(2):335–359.
- Moritz, P., Nishihara, R., Wang, S., Tumanov, A., Liaw, R., Liang, E., Elibol, M., Yang, Z., Paul, W., Jordan, M. I., et al. (2018). Ray: A distributed framework for emerging AI applications. In *OSDI*, pages 561–577.

- Motte, M. and Pham, H. (2019). Mean-field markov decision processes with common noise and open-loop controls. *arXiv preprint arXiv:1912.07883*.
- Mouret, J.-B. and Clune, J. (2015). Illuminating search spaces by mapping elites. *arXiv*, pages arXiv–1504.
- Müller, J. P. and Fischer, K. (2014). Application impact of multi-agent systems and technologies: A survey. In *Agent-oriented software engineering*, pages 27–53. Springer.
- Muller, P., Omidshafiei, S., Rowland, M., Tuyls, K., Perolat, J., Liu, S., Hennes, D., Marris, L., Lanctot, M., Hughes, E., et al. (2019). A generalized training approach for multiagent learning. In *International Conference on Learning Representations*.
- Munos, R. and Szepesvári, C. (2008). Finite-time bounds for fitted value iteration. *Journal of Machine Learning Research*, 9(May):815–857.
- Nagarajan, V. and Kolter, J. Z. (2017). Gradient descent gan optimization is locally stable. In *Advances in neural information processing systems*, pages 5585–5595.
- Nam, T. and Pardo, T. A. (2011). Conceptualizing smart city with dimensions of technology, people, and institutions. In *Proceedings of the 12th annual international digital government research conference: digital government innovation in challenging times*, pages 282–291. ACM.
- Nash, J. (1951). Non-cooperative games. *Annals of mathematics*, pages 286–295.
- Nash, J. F. et al. (1950). Equilibrium points in n-person games. *Proceedings of the national academy of sciences*, 36(1):48–49.
- Nayyar, A., Mahajan, A., and Teneketzis, D. (2013). Decentralized stochastic control with partial history sharing: A common information approach. *IEEE Transactions on Automatic Control*, 58(7):1644–1658.
- Neale, V. L., Dingus, T. A., Klauer, S. G., Sudweeks, J., and Goodman, M. (2005). An overview of the 100-car naturalistic study and findings. *National Highway Traffic Safety Administration, Paper*, 5:0400.
- Nedic, A., Olshevsky, A., and Shi, W. (2017). Achieving geometric convergence for distributed optimization over time-varying graphs. *SIAM Journal on Optimization*, 27(4):2597–2633.
- Nedic, A. and Ozdaglar, A. (2009). Distributed subgradient methods for multi-agent optimization. *IEEE Transactions on Automatic Control*, 54(1):48–61.
- Nemirovsky, A. S. and Yudin, D. B. (1983). Problem complexity and method efficiency in optimization.

- Neu, G., Antos, A., György, A., and Szepesvári, C. (2010). Online markov decision processes under bandit feedback. In *Advances in Neural Information Processing Systems*, pages 1804–1812.
- Neu, G., Gyorgy, A., Szepesvari, C., and Antos, A. (2014). Online markov decision processes under bandit feedback. *IEEE Transactions on Automatic Control*, 59(3):676–691.
- Neu, G., Jonsson, A., and Gómez, V. (2017). A unified view of entropy-regularized markov decision processes. *NIPS*.
- Neumann, J. v. (1928). Zur theorie der gesellschaftsspiele. *Mathematische annalen*, 100(1):295–320.
- Nguyen, T. T., Nguyen, N. D., and Nahavandi, S. (2020). Deep reinforcement learning for multiagent systems: A review of challenges, solutions, and applications. *IEEE transactions on cybernetics*.
- Niwa, H.-S. (1994). Self-organizing dynamic model of fish schooling. *Journal of theoretical Biology*, 171(2):123–136.
- Nocedal, J. and Wright, S. J. (2006). *Numerical Optimization*. Springer, New York, NY, USA, second edition.
- Nouiehed, M., Sanjabi, M., Huang, T., Lee, J. D., and Razaviyayn, M. (2019). Solving a class of non-convex min-max games using iterative first order methods. In *Advances in Neural Information Processing Systems*, pages 14934–14942.
- Nowé, A., Vrancx, P., and De Hauwere, Y.-M. (2012). Game theory and multi-agent reinforcement learning. In *Reinforcement Learning*, pages 441–470. Springer.
- Nutz, M., San Martin, J., Tan, X., et al. (2020). Convergence to the mean field game limit: a case study. *The Annals of Applied Probability*, 30(1):259–286.
- Oliehoek, F. A., Amato, C., et al. (2016). *A concise introduction to decentralized POMDPs*, volume 1. Springer.
- Omidshafiei, S., Papadimitriou, C., Piliouras, G., Tuyls, K., Rowland, M., Lespiau, J.-B., Czarnecki, W. M., Lanctot, M., Perolat, J., and Munos, R. (2019a). α -rank: Multi-agent evaluation by evolution. *Scientific reports*, 9(1):1–29.
- Omidshafiei, S., Papadimitriou, C., Piliouras, G., Tuyls, K., Rowland, M., Lespiau, J.-B., Czarnecki, W. M., Lanctot, M., Perolat, J., and Munos, R. (2019b). α -rank: Multi-agent evaluation by evolution.
- Omidshafiei, S., Tuyls, K., Czarnecki, W. M., Santos, F. C., Rowland, M., Connor, J., Hennes, D., Muller, P., Perolat, J., De Vylde, B., et al. (2020). Navigating the landscape of games. *arXiv preprint arXiv:2005.01642*.

- OpenAI, Andrychowicz, M., Baker, B., Chociej, M., Józefowicz, R., McGrew, B., Pachocki, J. W., Pachocki, J., Petron, A., Plappert, M., Powell, G., Ray, A., Schneider, J., Sidor, S., Tobin, J., Welinder, P., Weng, L., and Zaremba, W. (2018). Learning dexterous in-hand manipulation. *CoRR*, abs/1808.00177.
- OroojlooyJadid, A. and Hajinezhad, D. (2019). A review of cooperative multi-agent deep reinforcement learning. *arXiv preprint arXiv:1908.03963*.
- Ortner, R., Gajane, P., and Auer, P. (2020). Variational regret bounds for reinforcement learning. In *Uncertainty in Artificial Intelligence*, pages 81–90. PMLR.
- Osband, I., Doron, Y., Hessel, M., Aslanides, J., Sezener, E., Saraiva, A., McKinney, K., Lattimore, T., Szepesvari, C., Singh, S., et al. (2019). Behaviour suite for reinforcement learning. *arXiv preprint arXiv:1908.03568*.
- Osborne, M. J. and Rubinstein, A. (1994). *A course in game theory*. MIT press.
- Ostrom, E. (2000). Collective action and the evolution of social norms. *Journal of economic perspectives*, 14(3):137–158.
- Ostrom, E. (2009). *Understanding institutional diversity*. Princeton university press.
- Pachocki, J., Brockman, G., Raiman, J., Zhang, S., Pondé, H., Tang, J., Wolski, F., Dennison, C., Jozefowicz, R., Debiak, P., et al. (2018). Openai five, 2018. URL <https://blog.openai.com/openai-five>.
- Paden, B., Čáp, M., Yong, S. Z., Yershov, D., and Frazzoli, E. (2016). A survey of motion planning and control techniques for self-driving urban vehicles. *IEEE Transactions on intelligent vehicles*, 1(1):33–55.
- Page, L., Brin, S., Motwani, R., and Winograd, T. (1999). The pagerank citation ranking: Bringing order to the web. Technical report, Stanford InfoLab.
- Palaiopanos, G., Panageas, I., and Pilipouras, G. (2017). Multiplicative weights update with constant step-size in congestion games: Convergence, limit cycles and chaos. In *Advances in Neural Information Processing Systems*, pages 5872–5882.
- Palmer, A. (2014). *Reading Lucretius in the Renaissance*, volume 16. Harvard University Press.
- 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.
- Pankiw, T. and Page Jr, R. E. (2000). Response thresholds to sucrose predict foraging division of labor in honeybees. *Behavioral Ecology and Sociobiology*, 47(4):265–267.
- Papadimitriou, C. H. (2003). *Computational complexity*. John Wiley and Sons Ltd.

- Papadimitriou, C. H. and Tsitsiklis, J. N. (1987). The complexity of markov decision processes. *Mathematics of operations research*, 12(3):441–450.
- Papoudakis, G., Christianos, F., Schäfer, L., and Albrecht, S. V. (2020). Comparative evaluation of multi-agent deep reinforcement learning algorithms. *arXiv preprint arXiv:2006.07869*.
- Paredis, J. (1995). Coevolutionary computation. *Artificial life*, 2(4):355–375.
- Peng, P., Yuan, Q., Wen, Y., Yang, Y., Tang, Z., Long, H., and Wang, J. (2017a). Multiagent bidirectionally-coordinated nets for learning to play starcraft combat games. *arXiv preprint arXiv:1703.10069*.
- Peng, P., Yuan, Q., Wen, Y., Yang, Y., Tang, Z., Long, H., and Wang, J. (2017b). Multiagent bidirectionally-coordinated nets for learning to play starcraft combat games. *CoRR*, abs/1703.10069.
- Perkins, S., Mertikopoulos, P., and Leslie, D. S. (2015). Mixed-strategy learning with continuous action sets. *IEEE Transactions on Automatic Control*, 62(1):379–384.
- Perolat, J., Leibo, J. Z., Zambaldi, V., Beattie, C., Tuyls, K., and Graepel, T. (2017). A multi-agent reinforcement learning model of common-pool resource appropriation. In *Advances in Neural Information Processing Systems*, pages 3643–3652.
- Perolat, J., Piot, B., and Pietquin, O. (2018). Actor-critic fictitious play in simultaneous move multistage games. In *International Conference on Artificial Intelligence and Statistics*, pages 919–928. PMLR.
- Peters, J. and Schaal, S. (2008). Natural actor-critic. *Neurocomputing*, 71(7-9):1180–1190.
- Pham, H. and Wei, X. (2016). Discrete time mckean–vlasov control problem: a dynamic programming approach. *Applied Mathematics & Optimization*, 74(3):487–506.
- Pham, H. and Wei, X. (2017). Dynamic programming for optimal control of stochastic mckean–vlasov dynamics. *SIAM Journal on Control and Optimization*, 55(2):1069–1101.
- Pham, H. and Wei, X. (2018). Bellman equation and viscosity solutions for mean-field stochastic control problem. *ESAIM: Control, Optimisation and Calculus of Variations*, 24(1):437–461.
- Pinsky, M. and Karlin, S. (2010). *An introduction to stochastic modeling*. Academic press.
- Portelas, R., Colas, C., Weng, L., Hofmann, K., and Oudeyer, P.-Y. (2020). Automatic curriculum learning for deep rl: A short survey. *arXiv preprint arXiv:2003.04664*.

- Powers, R. and Shoham, Y. (2005a). Learning against opponents with bounded memory. In *IJCAI*, volume 5, pages 817–822.
- Powers, R. and Shoham, Y. (2005b). New criteria and a new algorithm for learning in multi-agent systems. In *Advances in neural information processing systems*, pages 1089–1096.
- Prasad, H., LA, P., and Bhatnagar, S. (2015). Two-timescale algorithms for learning nash equilibria in general-sum stochastic games. In *Proceedings of the 2015 International Conference on Autonomous Agents and Multiagent Systems*, pages 1371–1379.
- Pugh, J. K., Soros, L. B., and Stanley, K. O. (2016). Quality diversity: A new frontier for evolutionary computation. *Frontiers in Robotics and AI*, 3:40.
- Qu, C., Mannor, S., Xu, H., Qi, Y., Song, L., and Xiong, J. (2019). Value propagation for decentralized networked deep multi-agent reinforcement learning. In *Advances in Neural Information Processing Systems*, pages 1184–1193.
- Rafique, H., Liu, M., Lin, Q., and Yang, T. (2018). Non-convex min-max optimization: Provable algorithms and applications in machine learning. *arXiv*, pages arXiv–1810.
- Rakhlin, A., Shamir, O., and Sridharan, K. (2011). Making gradient descent optimal for strongly convex stochastic optimization. *arXiv preprint arXiv:1109.5647*.
- Ramanishka, V., Chen, Y.-T., Misu, T., and Saenko, K. (2018). Toward driving scene understanding: A dataset for learning driver behavior and causal reasoning. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 7699–7707.
- 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 *International Conference on Machine Learning*, pages 4295–4304.
- Ratliff, L. J., Burden, S. A., and Sastry, S. S. (2013). Characterization and computation of local nash equilibria in continuous games. In *2013 51st Annual Allerton Conference on Communication, Control, and Computing (Allerton)*, pages 917–924. IEEE.
- Ratliff, L. J., Burden, S. A., and Sastry, S. S. (2014). Genericity and structural stability of non-degenerate differential nash equilibria. In *2014 American Control Conference*, pages 3990–3995. IEEE.
- Rausher, M. D. (2001). Co-evolution and plant resistance to natural enemies. *Nature*, 411(6839):857–864.
- Reichenbach, T., Mobilia, M., and Frey, E. (2007). Mobility promotes and jeopardizes biodiversity in rock–paper–scissors games. *Nature*, 448(7157):1046–1049.

- Rendle, S. (2010). Factorization machines. In *2010 IEEE International Conference on Data Mining*, pages 995–1000. IEEE.
- Rendle, S. (2012). Factorization machines with libfm. *ACM Transactions on Intelligent Systems and Technology (TIST)*, 3(3):57.
- Riedmiller, M. (2005). Neural fitted q iteration–first experiences with a data efficient neural reinforcement learning method. In *European Conference on Machine Learning*, pages 317–328. Springer.
- Rising, J. K. (2013). Advances in the theory of determinantal point processes. *PhD Dissertation*.
- Roberson, B. (2006). The colonel blotto game. *Economic Theory*, 29(1):1–24.
- Ruijters, D. M., Vamplew, P., Whiteson, S., and Dazeley, R. (2013). A survey of multi-objective sequential decision-making. *Journal of Artificial Intelligence Research*, 48:67–113.
- Rosenberg, A. and Mansour, Y. (2019). Online convex optimization in adversarial markov decision processes. In *International Conference on Machine Learning*, pages 5478–5486.
- Rothfuss, J., Lee, D., Clavera, I., Asfour, T., and Abbeel, P. (2018). Promp: Proximal meta-policy search. In *International Conference on Learning Representations*.
- Roughgarden, T. (2005). *Selfish routing and the price of anarchy*, volume 174. MIT press Cambridge.
- Rowland, M., Omidshafiei, S., Tuyls, K., Perolat, J., Valko, M., Piliouras, G., and Munos, R. (2019). Multiagent evaluation under incomplete information.
- Rusu, A. A., Večerík, M., Rothörl, T., Heess, N., Pascanu, R., and Hadsell, R. (2017). Sim-to-real robot learning from pixels with progressive nets. In *Conference on Robot Learning*, pages 262–270. PMLR.
- Saldi, N., Basar, T., and Raginsky, M. (2018). Markov–nash equilibria in mean-field games with discounted cost. *SIAM Journal on Control and Optimization*, 56(6):4256–4287.
- Saldi, N., Başar, T., and Raginsky, M. (2019). Approximate nash equilibria in partially observed stochastic games with mean-field interactions. *Mathematics of Operations Research*, 44(3):1006–1033.
- Samothrakis, S., Lucas, S., Runarsson, T., and Robles, D. (2012). Coevolving game-playing agents: Measuring performance and intransitivities. *IEEE Transactions on Evolutionary Computation*, 17(2):213–226.

- Samvelyan, M., Rashid, T., de Witt, C. S., Farquhar, G., Nardelli, N., Rudner, T. G., Hung, C.-M., Torr, P. H., Foerster, J., and Whiteson, S. (2019). The StarCraft multi-agent challenge. *arXiv preprint arXiv:1902.04043*.
- Schaeffer, M. S. N. S. J., Shafiei, N., et al. (2009). Comparing uct versus cfr in simultaneous games.
- Schmid, M., Burch, N., Lanctot, M., Moravcik, M., Kadlec, R., and Bowling, M. (2019). Variance reduction in monte carlo counterfactual regret minimization (vr-mccfr) for extensive form games using baselines. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 33, pages 2157–2164.
- Schmidhuber, J. (2015). Deep learning in neural networks: An overview. *Neural networks*, 61:85–117.
- Schoemaker, P. J. (2013). *Experiments on decisions under risk: The expected utility hypothesis*. Springer Science & Business Media.
- School, V. D. (2018). 6 times you can proceed on a red light. <https://www.valleydrivingschool.com/blog/main/6-times-you-can-proceed-on-a-red-light>.
- Schriftwieser, J., Antonoglou, I., Hubert, T., Simonyan, K., Sifre, L., Schmitt, S., Guez, A., Lockhart, E., Hassabis, D., Graepel, T., et al. (2020). Mastering atari, go, chess and shogi by planning with a learned model. *Nature*, 588(7839):604–609.
- Schrodinger, E. (1943). *What is life?* University Press: Cambridge.
- Schulman, J., Levine, S., Abbeel, P., Jordan, M., and Moritz, P. (2015). Trust region policy optimization. In *International conference on machine learning*, pages 1889–1897.
- Schulman, J., Wolski, F., Dhariwal, P., Radford, A., and Klimov, O. (2017). Proximal policy optimization algorithms. *arXiv preprint arXiv:1707.06347*.
- Seeley, T. D. (2009). *The wisdom of the hive: the social physiology of honey bee colonies*. Harvard University Press.
- Selten, R. (1965). Spieltheoretische behandlung eines oligopolmodells mit nachfrageträgheit: Teil i: Bestimmung des dynamischen preisgleichgewichts. *Zeitschrift für die gesamte Staatswissenschaft/Journal of Institutional and Theoretical Economics*, (H. 2):301–324.
- Shakshuki, E. M. and Reid, M. (2015). Multi-agent system applications in health-care: current technology and future roadmap. In *ANT/SEIT*, pages 252–261.
- Shalev-Shwartz, S. and Ben-David, S. (2014). *Understanding machine learning: From theory to algorithms*. Cambridge university press.

- Shalev-Shwartz, S. et al. (2011). Online learning and online convex optimization. *Foundations and trends in Machine Learning*, 4(2):107–194.
- Shalev-Shwartz, S., Shammah, S., and Shashua, A. (2016a). Safe, multi-agent, reinforcement learning for autonomous driving. *arXiv preprint arXiv:1610.03295*.
- Shalev-Shwartz, S., Shammah, S., and Shashua, A. (2016b). Safe, multi-agent, reinforcement learning for autonomous driving.
- Shapley, L. S. (1953). Stochastic games. *Proceedings of the national academy of sciences*, 39(10):1095–1100.
- Shapley, L. S. (1974). A note on the lemke-howson algorithm. In *Pivoting and Extension*, pages 175–189. Springer.
- Shi, W., Song, S., and Wu, C. (2019). Soft policy gradient method for maximum entropy deep reinforcement learning. In *Proceedings of the 28th International Joint Conference on Artificial Intelligence*, pages 3425–3431. AAAI Press.
- Shoham, Y. and Leyton-Brown, K. (2008). *Multiagent systems: Algorithmic, game-theoretic, and logical foundations*. Cambridge University Press.
- Shoham, Y., Powers, R., and Grenager, T. (2007). If multi-agent learning is the answer, what is the question? *Artificial intelligence*, 171(7):365–377.
- Shub, M. (2013). *Global stability of dynamical systems*. Springer Science & Business Media.
- Siddiqui, F. (2020). Some of the biggest critics of Waymo and other self-driving cars are the silicon valley residents who know how they work. <https://wapo.st/30UAX6R>.
- Sidford, A., Wang, M., Wu, X., Yang, L., and Ye, Y. (2018). Near-optimal time and sample complexities for solving markov decision processes with a generative model. In *Advances in Neural Information Processing Systems*, pages 5186–5196.
- Sidford, A., Wang, M., Yang, L., and Ye, Y. (2020). Solving discounted stochastic two-player games with near-optimal time and sample complexity. In *International Conference on Artificial Intelligence and Statistics*, pages 2992–3002.
- Silver, D., Huang, A., Maddison, C. J., Guez, A., Sifre, L., Van Den Driessche, G., Schrittwieser, J., Antonoglou, I., Panneershelvam, V., Lanctot, M., et al. (2016). Mastering the game of go with deep neural networks and tree search. *nature*, 529(7587):484–489.
- Silver, D., Hubert, T., Schrittwieser, J., Antonoglou, I., Lai, M., Guez, A., Lanctot, M., Sifre, L., Kumaran, D., Graepel, T., et al. (2018). A general reinforcement learning algorithm that masters chess, shogi, and go through self-play. *Science*, 362(6419):1140–1144.

- Silver, D., Lever, G., Heess, N., Degris, T., Wierstra, D., and Riedmiller, M. (2014). Deterministic policy gradient algorithms. In *ICML*, pages 387–395.
- Silver, D., Schrittwieser, J., Simonyan, K., Antonoglou, I., Huang, A., Guez, A., Hubert, T., Baker, L., Lai, M., Bolton, A., et al. (2017). Mastering the game of go without human knowledge. *Nature*, 550(7676):354–359.
- Simon, H. A. (1972). Theories of bounded rationality. *Decision and organization*, 1(1):161–176.
- Simulator, L. S. A. V. (2020). LGSVL Simulator. <https://www.lgsvlsimulator.com/docs/>.
- Singh, S., Jaakkola, T., Littman, M. L., and Szepesvári, C. (2000a). Convergence results for single-step on-policy reinforcement-learning algorithms. *Machine learning*, 38(3):287–308.
- Singh, S. P., Kearns, M. J., and Mansour, Y. (2000b). Nash convergence of gradient dynamics in general-sum games. In *UAI*, pages 541–548.
- Sirignano, J. and Spiliopoulos, K. (2020). Mean field analysis of neural networks: A law of large numbers. *SIAM Journal on Applied Mathematics*, 80(2):725–752.
- Slumbers, O. (2020). Introducing diversity into population-based multi-agent learning. *UCL MSc Computational Statistics and Machine Learning Thesis*.
- Smith, J. M. and Price, G. R. (1973). The logic of animal conflict. *Nature*, 246(5427):15–18.
- 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 *International Conference on Machine Learning*, pages 5887–5896.
- Song, M., Montanari, A., and Nguyen, P. (2018). A mean field view of the landscape of two-layers neural networks. *Proceedings of the National Academy of Sciences*, 115:E7665–E7671.
- Srebro, N., Sridharan, K., and Tewari, A. (2011). On the universality of online mirror descent. In *Advances in neural information processing systems*, pages 2645–2653.
- Srinivasan, S., Lanctot, M., Zambaldi, V., Pérolat, J., Tuyls, K., Munos, R., and Bowling, M. (2018). Actor-critic policy optimization in partially observable multiagent environments. In *Advances in neural information processing systems*, pages 3422–3435.
- Standish, R. K. (2003). Open-ended artificial evolution. *International Journal of Computational Intelligence and Applications*, 3(02):167–175.

- Stanley, H. E. (1971). Phase transitions and critical phenomena. *Clarendon, Oxford*, 9.
- Stewart, J. (2020). Humans just can't stop rear-ending self-driving cars – let's figure out why. <https://bit.ly/3jfcffs>.
- Stone, P. (2007). Multiagent learning is not the answer. it is the question. *Artificial Intelligence*, 171(7):402–405.
- Stone, P. and Veloso, M. (2000). Multiagent systems: A survey from a machine learning perspective. *Autonomous Robots*, 8(3):345–383.
- Subramanian, J. and Mahajan, A. (2019). Reinforcement learning in stationary mean-field games. In *Proceedings of the 18th International Conference on Autonomous Agents and MultiAgent Systems*, pages 251–259.
- Subramanian, S. G., Poupart, P., Taylor, M. E., and Hegde, N. (2020). Multi type mean field reinforcement learning. *arXiv preprint arXiv:2002.02513*.
- Such, F. P., Madhavan, V., Conti, E., Lehman, J., Stanley, K. O., and Clune, J. (2017). Deep neuroevolution: Genetic algorithms are a competitive alternative for training deep neural networks for reinforcement learning. *arXiv preprint arXiv:1712.06567*.
- Sukhbaatar, S., Fergus, R., et al. (2016). Learning multiagent communication with backpropagation. In *Advances in Neural Information Processing Systems*, pages 2244–2252.
- Sumpter, D. (2006). The principles of collective animal behaviour. *Phil. Trans. R. Soc. B*, 361:5–22.
- Sunehag, P., Lever, G., Gruslys, A., Czarnecki, W. M., Zambaldi, V. F., Jaderberg, M., Lanctot, M., Sonnerat, N., Leibo, J. Z., Tuyls, K., et al. (2018). Value-decomposition networks for cooperative multi-agent learning based on team reward. In *AAMAS*, pages 2085–2087.
- Suttle, W., Yang, Z., Zhang, K., Wang, Z., Basar, T., and Liu, J. (2019). A multi-agent off-policy actor-critic algorithm for distributed reinforcement learning. *arXiv preprint arXiv:1903.06372*.
- Sutton, R. S. (1988). Learning to predict by the methods of temporal differences. *Machine learning*, 3(1):9–44.
- Sutton, R. S. and Barto, A. G. (1998). *Reinforcement learning: An introduction*, volume 1. MIT press Cambridge.
- 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.

- Sutton, R. S., McAllester, D. A., Singh, S. P., Mansour, Y., et al. (1999). Policy gradient methods for reinforcement learning with function approximation. In *NIPS*, volume 99, pages 1057–1063.
- Swenson, B. and Poor, H. V. (2019). Smooth fictitious play in $n \times 2$ potential games. In *2019 53rd Asilomar Conference on Signals, Systems, and Computers*, pages 1739–1743. IEEE.
- Syed, U., Bowling, M., and Schapire, R. E. (2008). Apprenticeship learning using linear programming. In *Proceedings of the 25th international conference on Machine learning*, pages 1032–1039.
- Szepesvári, C. (2010). Algorithms for reinforcement learning. *Synthesis lectures on artificial intelligence and machine learning*, 4(1):1–103.
- Szepesvári, C. and Littman, M. L. (1999). A unified analysis of value-function-based reinforcement-learning algorithms. *Neural computation*, 11(8):2017–2060.
- Szer, D., Charpillet, F., and Zilberstein, S. (2005). Maa*: A heuristic search algorithm for solving decentralized pomdps.
- Sznitman, A.-S. (1991). Topics in propagation of chaos. In *Ecole d’été de probabilités de Saint-Flour XIX—1989*, pages 165–251. Springer.
- Tammelin, O., Burch, N., Johanson, M., and Bowling, M. (2015). Solving heads-up limit texas hold’em. In *Twenty-Fourth International Joint Conference on Artificial Intelligence*.
- Tan, M. (1993). Multi-agent reinforcement learning: Independent vs. cooperative agents. In *Proceedings of the tenth international conference on machine learning*, pages 330–337.
- Taylor, M. E. and Stone, P. (2009). Transfer learning for reinforcement learning domains: A survey. *Journal of Machine Learning Research*, 10(7).
- Taylor, T., Bedau, M., Channon, A., Ackley, D., Banzhaf, W., Beslon, G., Dolson, E., Froese, T., Hickinbotham, S., Ikegami, T., et al. (2016). Open-ended evolution: Perspectives from the oee workshop in york. *Artificial life*, 22(3):408–423.
- Terry, J. K., Black, B., Jayakumar, M., Hari, A., Santos, L., Dieffendahl, C., Williams, N. L., Lokesh, Y., Sullivan, R., Horsch, C., and Ravi, P. (2020). Pettingzoo: Gym for multi-agent reinforcement learning. *arXiv preprint arXiv:2009.14471*.
- Tesauro, G. (1995). Temporal difference learning and td-gammon. *Communications of the ACM*, 38(3):58–68.

- Thekumparampil, K. K., Jain, P., Netrapalli, P., and Oh, S. (2019). Efficient algorithms for smooth minimax optimization. In *Advances in Neural Information Processing Systems*, pages 12680–12691.
- Thorndike, E. L. (1898). Animal intelligence: an experimental study of the associative processes in animals. *The Psychological Review: Monograph Supplements*, 2(4):i.
- Tian, Z., Wen, Y., Gong, Z., Punakkath, F., Zou, S., and Wang, J. (2019). A regularized opponent model with maximum entropy objective. In *Proceedings of the 28th International Joint Conference on Artificial Intelligence*, pages 602–608. AAAI Press.
- Todorov, E., Erez, T., and Tassa, Y. (2012). MuJoCo: A physics engine for model-based control. In *2012 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 5026–5033. IEEE.
- Toussaint, M., Charlin, L., and Poupart, P. (2008). Hierarchical pomdp controller optimization by likelihood maximization. In *UAI*, volume 24, pages 562–570.
- Treiber, M., Hennecke, A., and Helbing, D. (2000). Congested traffic states in empirical observations and microscopic simulations. *Physical review E*, 62(2):1805.
- Treiber, M. and Kesting, A. (2009). Modeling lane-changing decisions with mobil. In *Traffic and Granular Flow'07*, pages 211–221. Springer.
- Tropp, J. A., Yurtsever, A., Udell, M., and Cevher, V. (2017). Practical sketching algorithms for low-rank matrix approximation. *SIAM Journal on Matrix Analysis and Applications*, 38(4):1454–1485.
- Troy, C. A. (1997). Envisioning stock trading where the brokers are bots. *New York Times*, 16.
- Tsaknakis, H. and Spirakis, P. G. (2007). An optimization approach for approximate nash equilibria. In *International Workshop on Web and Internet Economics*, pages 42–56. Springer.
- Tuyls, K. and Nowé, A. (2005). Evolutionary game theory and multi-agent reinforcement learning.
- Tuyls, K. and Parsons, S. (2007). What evolutionary game theory tells us about multiagent learning. *Artificial Intelligence*, 171(7):406–416.
- 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.
- Tuyls, K. and Weiss, G. (2012). Multiagent learning: Basics, challenges, and prospects. *Ai Magazine*, 33(3):41–41.

- Usunier, N., Synnaeve, G., Lin, Z., and Chintala, S. (2016). Episodic exploration for deep deterministic policies: An application to starcraft micromanagement tasks. *arXiv preprint arXiv:1609.02993*.
- uz Zaman, M. A., Zhang, K., Miehling, E., and Başar, T. (2020). Approximate equilibrium computation for discrete-time linear-quadratic mean-field games. In *2020 American Control Conference (ACC)*, pages 333–339. IEEE.
- van der Wal, J., van der Wal, J., van der Wal, J., Mathématicien, P.-B., van der Wal, J., and Mathematician, N. (1981). *Stochastic Dynamic Programming: successive approximations and nearly optimal strategies for Markov decision processes and Markov games*. Mathematisch centrum.
- Van Otterlo, M. and Wiering, M. (2012). Reinforcement learning and markov decision processes. In *Reinforcement Learning*, pages 3–42. Springer.
- Vinyals, O., Babuschkin, I., Czarnecki, W. M., Mathieu, M., Dudzik, A., Chung, J., Choi, D. H., Powell, R., Ewalds, T., Georgiev, P., et al. (2019a). Grandmaster level in starcraft ii using multi-agent reinforcement learning. *Nature*, 575(7782):350–354.
- Vinyals, O., Babuschkin, I., Czarnecki, W. M., Mathieu, M., Dudzik, A., Chung, J., Choi, D. H., Powell, R., Ewalds, T., Georgiev, P., et al. (2019b). Grandmaster level in StarCraft II using multi-agent reinforcement learning. *Nature*, 575(7782):350–354.
- Vinyals, O., Ewalds, T., Bartunov, S., Georgiev, P., Vezhnevets, A. S., Yeo, M., Makhzani, A., Küttler, H., Agapiou, J., Schrittwieser, J., et al. (2017). Starcraft ii: A new challenge for reinforcement learning. *arXiv preprint arXiv:1708.04782*.
- Viossat, Y. (2007). The replicator dynamics does not lead to correlated equilibria. *Games and Economic Behavior*, 59(2):397–407.
- Viossat, Y. and Zapechelnyuk, A. (2013). No-regret dynamics and fictitious play. *Journal of Economic Theory*, 148(2):825–842.
- Virtual Test Drive (VTD) (2020). Complete Tool-Chain for Driving Simulation. <https://www.mscsoftware.com/product/virtual-test-drive>.
- Vogt, K. (2020). The disengagement myth. <https://medium.com/cruise/the-disengagement-myth-1b5cbdf8e239>.
- Von Neumann, J. and Morgenstern, O. (1945). *Theory of games and economic behavior*. Princeton University Press Princeton, NJ.
- Von Neumann, J. and Morgenstern, O. (2007). *Theory of games and economic behavior (commemorative edition)*. Princeton university press.

- Wang, J., Zhang, W., Yuan, S., et al. (2017a). Display advertising with real-time bidding (rtb) and behavioural targeting. *Foundations and Trends® in Information Retrieval*, 11(4-5):297–435.
- Wang, L., Cai, Q., Yang, Z., and Wang, Z. (2019). Neural policy gradient methods: Global optimality and rates of convergence. In *International Conference on Learning Representations*.
- Wang, S. I., Liang, P., and Manning, C. D. (2016). Learning language games through interaction. *arXiv preprint arXiv:1606.02447*.
- Wang, X. and Sandholm, T. (2003). Reinforcement learning to play an optimal nash equilibrium in team markov games. In *Advances in neural information processing systems*, pages 1603–1610.
- Wang, Z., Merel, J. S., Reed, S. E., de Freitas, N., Wayne, G., and Heess, N. (2017b). Robust imitation of diverse behaviors. *Advances in Neural Information Processing Systems*, 30:5320–5329.
- Watkins, C. J. and Dayan, P. (1992). Q-learning. *Machine learning*, 8(3-4):279–292.
- Waugh, K., Morrill, D., Bagnell, J. A., and Bowling, M. (2014). Solving games with functional regret estimation. *arXiv preprint arXiv:1411.7974*.
- Waymo (2020). Waymo safety report. <https://bit.ly/2T4vRl4>.
- Weaver, L. and Tao, N. (2001). The optimal reward baseline for gradient-based reinforcement learning. In *Proceedings of the Seventeenth conference on Uncertainty in artificial intelligence*, pages 538–545.
- Wei, C.-Y., Hong, Y.-T., and Lu, C.-J. (2017). Online reinforcement learning in stochastic games. In *Advances in Neural Information Processing Systems*, pages 4987–4997.
- Wei, H., Xu, N., Zhang, H., Zheng, G., Zang, X., Chen, C., Zhang, W., Zhu, Y., Xu, K., and Li, Z. (2019). Colight: Learning network-level cooperation for traffic signal control. In *CIKM*, pages 1913–1922.
- Weidenmüller, A. (2004). The control of nest climate in bumblebee (*bombus terrestris*) colonies: interindividual variability and self reinforcement in fanning response. *Behavioral Ecology*, 15(1):120–128.
- Weintraub, G. Y., Benkard, L., and Van Roy, B. (2006). Oblivious equilibrium: A mean field approximation for large-scale dynamic games. In *Advances in neural information processing systems*, pages 1489–1496.
- Weiss, G. (1999). *Multiagent systems: a modern approach to distributed artificial intelligence*. MIT press.

- Weiss, P. (1907). L'hypothèse du champ moléculaire et la propriété ferromagnétique.
- Wellman, M. P. (2006). Methods for empirical game-theoretic analysis. In *AAAI*, pages 1552–1556.
- Wen, Y. (2019). Malib: A multi-agent learning framework. <https://github.com/ying-wen/malib>.
- Wen, Y., Yang, Y., Lu, R., and Wang, J. (2019a). Multi-agent generalized recursive reasoning. *arXiv preprint arXiv:1901.09216*.
- Wen, Y., Yang, Y., Luo, R., and Wang, J. (2019b). Modelling bounded rationality in multi-agent interactions by generalized recursive reasoning. *IJCAI*, pages arXiv–1901.
- Wen, Y., Yang, Y., Luo, R., Wang, J., and Pan, W. (2018). Probabilistic recursive reasoning for multi-agent reinforcement learning. In *International Conference on Learning Representations*.
- Wen, Y., Yang, Y., Luo, R., Wang, J., and Pan, W. (2019c). Probabilistic recursive reasoning for multi-agent reinforcement learning. *arXiv preprint arXiv:1901.09207*.
- Wikipedia (2020). Pittsburgh left. https://en.wikipedia.org/wiki/Pittsburgh_left.
- Williams, R. J. (1992). Simple statistical gradient-following algorithms for connectionist reinforcement learning. *Machine learning*, 8(3-4):229–256.
- Wilson, E. O. et al. (1971). The insect societies. *The insect societies*.
- Wooldridge, M. (2009). *An introduction to multiagent systems*. John Wiley & Sons.
- Workshop, I. (2019). Reinforcement learning for real life. <https://icml.cc-Conferences/2019/ScheduleMultitrack?event=3515>.
- Workshop, N. (2020a). Challenges of real-world rl. <https://sites.google.com/view/neurips2020rwrl>.
- Workshop, R. (2020b). RL for real life 2020. <https://sites.google.com/view/RL4RealLife>.
- Wu, C., Kreidieh, A., Parvate, K., Vinitsky, E., and Bayen, A. M. (2017). Flow: Architecture and benchmarking for reinforcement learning in traffic control. *ArXiv*, abs/1710.05465.
- Wu, F., Zilberstein, S., and Chen, X. (2010). Rollout sampling policy iteration for decentralized pomdps. In *Proceedings of the Twenty-Sixth Conference on Uncertainty in Artificial Intelligence*, pages 666–673.

- Wu, F., Zilberstein, S., and Jennings, N. R. (2013). Monte-carlo expectation maximization for decentralized pomdps. In *Twenty-Third International Joint Conference on Artificial Intelligence*.
- Yabu, Y., Yokoo, M., and Iwasaki, A. (2007). Multiagent planning with trembling-hand perfect equilibrium in multiagent pomdps. In *Pacific Rim International Conference on Multi-Agents*, pages 13–24. Springer.
- Yadkori, Y. A., Bartlett, P. L., Kanade, V., Seldin, Y., and Szepesvári, C. (2013). Online learning in markov decision processes with adversarially chosen transition probability distributions. In *Advances in neural information processing systems*, pages 2508–2516.
- Yang, J., Ye, X., Trivedi, R., Xu, H., and Zha, H. (2017). Deep mean field games for learning optimal behavior policy of large populations. *CoRR*, abs/1711.03156.
- Yang, J., Ye, X., Trivedi, R., Xu, H., and Zha, H. (2018a). Learning deep mean field games for modeling large population behavior. In *International Conference on Learning Representations*.
- Yang, Y., Luo, R., Li, M., Zhou, M., Zhang, W., and Wang, J. (2018b). Mean field multi-agent reinforcement learning. In *International Conference on Machine Learning*, pages 5571–5580.
- Yang, Y., Tutunov, R., Sakulwongtana, P., Ammar, H. B., and Wang, J. (2019a). Alpha-alpha-rank: Scalable multi-agent evaluation through evolution.
- Yang, Y. and Wang, J. (2020). An overview of multi-agent reinforcement learning from game theoretical perspective. *arXiv preprint arXiv:2011.00583*.
- Yang, Y., Wen, Y., Chen, L., Wang, J., Shao, K., Mguni, D., and Zhang, W. (2020). Multi-agent determinantal q-learning. *ICML*.
- Yang, Y., Yu, L., Bai, Y., Wen, Y., Zhang, W., and Wang, J. (2018c). A study of ai population dynamics with million-agent reinforcement learning. In *Proceedings of the 17th International Conference on Autonomous Agents and MultiAgent Systems*, pages 2133–2135. International Foundation for Autonomous Agents and Multiagent Systems.
- Yang, Z., Chen, Y., Hong, M., and Wang, Z. (2019b). Provably global convergence of actor-critic: A case for linear quadratic regulator with ergodic cost. In *Advances in Neural Information Processing Systems*, pages 8353–8365.
- Yang, Z., Xie, Y., and Wang, Z. (2019c). A theoretical analysis of deep q-learning. *arXiv preprint arXiv:1901.00137*.
- Ye, Y. (2005). A new complexity result on solving the markov decision problem. *Mathematics of Operations Research*, 30(3):733–749.

- Ye, Y. (2010). The simplex method is strongly polynomial for the markov decision problem with a fixed discount rate.
- Yongacoglu, B., Arslan, G., and Yüksel, S. (2019). Learning team-optimality for decentralized stochastic control and dynamic games. *arXiv preprint arXiv:1903.05812*.
- Young, H. P. (1993). The evolution of conventions. *Econometrica: Journal of the Econometric Society*, pages 57–84.
- Yu, F., Xian, W., Chen, Y., Liu, F., Liao, M., Madhavan, V., and Darrell, T. (2018). Bdd100k: A diverse driving video database with scalable annotation tooling. *arXiv preprint arXiv:1805.04687*.
- Yu, J. Y., Mannor, S., and Shimkin, N. (2009). Markov decision processes with arbitrary reward processes. *Mathematics of Operations Research*, 34(3):737–757.
- Yu, L., Zhang, W., Wang, J., and Yu, Y. (2017). Seqgan: Sequence generative adversarial nets with policy gradient. In *AAAI*, pages 2852–2858.
- Zazo, S., Valcarcel Macua, S., Sánchez-Fernández, M., and Zazo, J. (2015). Dynamic potential games in communications: Fundamentals and applications. *arXiv*, pages arXiv–1509.
- Zermelo, E. and Borel, E. (1913). On an application of set theory to the theory of the game of chess. In *Congress of Mathematicians*, pages 501–504. CUP.
- Zhan, W., Sun, L., Wang, D., Shi, H., Clausse, A., Naumann, M., Kummerle, J., Königshof, H., Stiller, C., de La Fortelle, A., et al. (2019a). Interaction dataset: An international, adversarial and cooperative motion dataset in interactive driving scenarios with semantic maps. *arXiv preprint arXiv:1910.03088*.
- Zhan, W., Sun, L., Wang, D., Shi, H., Clausse, A., Naumann, M., Kümmeler, J., Königshof, H., Stiller, C., de La Fortelle, A., and Tomizuka, M. (2019b). INTERACTION Dataset: An INTERNATIONAL, Adversarial and Cooperative moTION Dataset in Interactive Driving Scenarios with Semantic Maps. *arXiv:1910.03088 [cs, eess]*.
- Zhang, C. and Lesser, V. (2010). Multi-agent learning with policy prediction. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 24.
- Zhang, G. and Yu, Y. (2019). Convergence of gradient methods on bilinear zero-sum games. *arXiv e-prints*, pages arXiv–1908.
- Zhang, H., Chen, W., Huang, Z., Li, M., Yang, Y., Zhang, W., and Wang, J. (2019a). Bi-level actor-critic for multi-agent coordination. *arXiv preprint arXiv:1909.03510*.

- Zhang, H., Feng, S., Liu, C., Ding, Y., Zhu, Y., Zhou, Z., Zhang, W., Yu, Y., Jin, H., and Li, Z. (2019b). CityFlow: A multi-agent reinforcement learning environment for large scale city traffic scenario. In *WWW*, page 3620–3624. ACM.
- Zhang, K., Sun, T., Tao, Y., Genc, S., Mallya, S., and Basar, T. (2020). Robust multi-agent reinforcement learning with model uncertainty. *Advances in Neural Information Processing Systems*, 33.
- Zhang, K., Yang, Z., and Basar, T. (2018a). Networked multi-agent reinforcement learning in continuous spaces. In *2018 IEEE Conference on Decision and Control (CDC)*, pages 2771–2776. IEEE.
- Zhang, K., Yang, Z., and Başar, T. (2019c). Multi-agent reinforcement learning: A selective overview of theories and algorithms. *arXiv preprint arXiv:1911.10635*.
- Zhang, K., Yang, Z., and Basar, T. (2019d). Policy optimization provably converges to nash equilibria in zero-sum linear quadratic games. In *Advances in Neural Information Processing Systems*, pages 11602–11614.
- Zhang, K., Yang, Z., Liu, H., Zhang, T., and Başar, T. (2018b). Finite-sample analysis for decentralized batch multi-agent reinforcement learning with networked agents. *arXiv preprint arXiv:1812.02783*.
- Zhang, K., Yang, Z., Liu, H., Zhang, T., and Basar, T. (2018c). Fully decentralized multi-agent reinforcement learning with networked agents. In *International Conference on Machine Learning*, pages 5872–5881.
- Zhang, Y. and Zavlanos, M. M. (2019). Distributed off-policy actor-critic reinforcement learning with policy consensus. In *2019 IEEE 58th Conference on Decision and Control (CDC)*, pages 4674–4679. IEEE.
- Zhao, T., Hachiya, H., Niu, G., and Sugiyama, M. (2011). Analysis and improvement of policy gradient estimation. In *Advances in Neural Information Processing Systems*, pages 262–270.
- Zheng, L., Yang, J., Cai, H., Zhou, M., Zhang, W., Wang, J., and Yu, Y. (2018). Magent: A many-agent reinforcement learning platform for artificial collective intelligence. In [McIlraith and Weinberger \(2018\)](#).
- 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.
- Zhou, M., Luo, J., Villela, J., Yang, Y., Rusu, D., Miao, J., Zhang, W., Alban, M., Fadakar, I., Chen, Z., et al. (2020). Smarts: Scalable multi-agent reinforcement learning training school for autonomous driving. *arXiv preprint arXiv:2010.09776*.

- Zimin, A. and Neu, G. (2013). Online learning in episodic markovian decision processes by relative entropy policy search. In *Advances in neural information processing systems*, pages 1583–1591.
- Zinkevich, M. (2003). Online convex programming and generalized infinitesimal gradient ascent. In *Proceedings of the 20th international conference on machine learning (icml-03)*, pages 928–936.
- Zinkevich, M., Greenwald, A., and Littman, M. L. (2006). Cyclic equilibria in markov games. In *Advances in Neural Information Processing Systems*, pages 1641–1648.
- Zinkevich, M., Johanson, M., Bowling, M., and Piccione, C. (2008). Regret minimization in games with incomplete information. In *Advances in neural information processing systems*, pages 1729–1736.
- Zoph, B. and Le, Q. V. (2016). Neural architecture search with reinforcement learning. *arXiv preprint arXiv:1611.01578*.