Multi-Agent Reinforcement Learning Toward Mixed-Motivation Collaboration

Haoyuan Qin[*], **Ziwei Deng**[*], **Mian Deng**[*]
Department of Computer Science and Technology, Class 2
School of Informatics, Xiamen University, Xiamen, China
23020231154219@stu.xmu.edu.cn
23020231154177@stu.xmu.edu.cn
23020231154137@stu.xmu.edu.cn

Abstract

In this paper, we address the challenge of multi-agent mixed-motivation cooperation within the diverse scenarios presented by the Melting Pot Contest at NeurIPS 2023. The contest's innovative environments test agents' adaptability to new partners and unforeseen situations, requiring a display of authentic cooperative intelligence for success. We introduce a deep reinforcement learning framework combining system design and algorithms. Our method involves a ResNet for visual processing, a GRU layer for temporal analysis, and an MLP for estimating action values. This setup allows agents to construct policies delivering Q values for informed action choices, guided by a pre-set strategy during societal game scenarios. We adopt Centralized Training with Decentralized Execution (CTDE) and employ the Value Decomposition Network (VDN) for cooperative policy development. To mitigate competition's negative impacts in non-competitive settings, we integrate NPC agents in training, which undertake tasks to enrich the learning environment and boost our agents' performance. Tested in the Melting Pot Contest, our framework proved effective, achieving nuanced cooperation and earning second place overall.

Introduction

Multi-agent system is a distributed decision-making system consisting of multiple intelligences interacting with the environment. It has a wide range of applications in real life, such as in military, industrial, transportation and other fields, multi-agent system can efficiently complete the group autonomy decision-making task. For example, in many fields such as military, industry, and transportation, MAS can efficiently accomplish group autonomous decision-making tasks.(Wooldridge 2009). In addition, some complex social problems, such as resource scheduling, business competition, financial analysis, group psychology, etc., can also be abstracted into multi-agent models to solve their agent optimization(Sutton and Barto 2018).

Multi-agent reinforcement learning has become the focus of MAS researchers, and it greatly facilitates the optimization of MAS models. With the deepening of research, MARL improves the level of group decision-making on the one hand, and enriches the types of tasks performed by multiple agents on the other. At present, according to the different optimization objectives of multi-intelligence, the learning tasks of multi-intelligence in MARL can be classified into three types: fully collaborative, fully competitive, and mixed environments.Multi-Agent cooperation is often more widely used in real production and operation processes, where learning cooperation in a narrow sense means that agents learn to collaborate only through local observation to achieve common goals.

Melting Pot is composed of a set of test scenarios and a protocol for using them. A scenario is a multi-agent body environment that evaluates the ability of a centralized group of intelligences to generalize to new social situations. Each scenario is formed by a substrate and a background population. The substrate is the physical part of the world: its spatial layout, where objects are, how they move, the rules of physics, and so on. The background population is the part of the simulation that is full of energies, excluding the focal population (that is, the multi-agent body population that we train). Although the focal population was in the environmental matrix during training, it never experienced any of the individuals in the background population. Thus, the performance of the focal population in the test scenarios measures the generalization ability of its agents in a social context to which they were not directly exposed during training. The focus at the time of assessment was therefore on measuring generalization to a new social partner in a set of normative test scenarios.

To address these challenges, we designed a deep reinforcement learning framework and a set of algorithms. First, we modified ResNet(He et al. 2016) for the problem scenario to realize the conversion from visual RL to state RL; we used VDN (Sunehag et al. 2017)to solve the problem of credit allocation for multiple intelligences; and we used auxiliary rewards in combination with action mask filtering to realize the macro-statute learning of mixed-motivation cooperation in a melting pot environment. The model constructed by our team is currently in the top 5 in the overall score list and ranked 1st in the cleanup task.

Related Work

There are two types of environments for multi-agent control with reinforcement learning: cooperative and competitive

[*]These authors contributed equally.

environments. Our work belongs to the latter. Most of the existing studies have used games as testbeds for RL progress. At the time of training intelligences are allowed unlimited access to the background substrate. This means that at the time of testing the intelligences are usually already familiarized with their physical environment. This is because the kind of generalization explored in MeltingPot is primarily along the social dimension, rather than the physical dimension. This contrasts with most work on generalization in reinforcement learning.

We use the term Multi-Agent Population Learning Algorithm (MAPLA) to refer to any training process that produces a decentralized population of agents capable of interacting with each other at the same time.There is a wide variety of MAPLA algorithms. Most multi-agent reinforcement learning methods are capable of generating populations. For example, AlphaGo(Silver et al. 2017), AlphaZero (Silver et al. 2018), Hide-and-Seek(Baker et al. 2019), and AlphaStar (Vinyals et al. 2019) etc. use self-gaming schemes that fall into the MAPLA category, as do recent studies on DOTA(Berner et al. 2019) and MOBA (Ye et al. 2020) games, as well as MADDPG (Lowe et al. 2017), LOLA(Foerster et al. 2017), PSRO (Balduzzi et al. 2019), PSRO (Balduzzi et al. 2019), and Malthusian Reinforcement Learning (Leibo et al. 2018) algorithms all fall under the MAPLA. it includes both ”centralized training and decentralized execution” algorithms and ”fully decentralized” (decentralized at both training and testing times) algorithms. Melting Pot evaluates MAPLAs by assessing the degree of generalization of the population they produce.

Many classical reinforcement learning models use manually encoded information from humans as input to the model, e.g., for Go, we can define the difference in the number of remaining pieces between the two sides as an input so that we can indirectly capture the current probability of winning for both sides, and use this as a way to make a decision on the next move. However, not all environments are suitable for this, for example, in the field of robot control, where we are likely to have only one camera as input, in which case extracting useful information from the pixel-level input of the camera becomes somewhat difficult. This type of reinforcement learning that uses vision directly as input is called Vision-based RL, as opposed to a model training approach that uses low-dimensional features for training called State-based RL. In Melting Pot we face this same challenge, and our approach extracts the states and actions available in the reinforcement learning task from the image frames during training. available states and actions in the image frames during training.

Multi-intelligent Reinforcement Learning (MARL) typically requires the decomposition of a complex problem into multiple sub-problems, such that each intelligence focuses only on its own state and actions and cooperates to achieve the overall goal during training. In order to achieve efficient decentralized execution of MARL strategies, value factorization methods are widely used in MARL. (Schneider et al. 1999) consider optimizing the sum of individual reward functions by optimizing local combinations of individual value functions learned from individual value functions, and Russell and Zimdars (Russell and Zimdars 2003) sum the Q-functions of independently learned agents with individual rewards before greedily performing global action selection to optimize the total reward. VDN factorizes the value function as the sum of each agent's utility.QMIX (Rashid et al. 2020) models the monotonic relationship between individual utility and the value function. QAtten(Yang et al. 2020) and REFIL (Hu et al. 2021) use an attention mechanism to factorize the value function. qTran (Son et al. 2019) transforms the joint state-action value function Qjt into a form that is easy to factorize by means of linear constraints. ResQ (Shen et al. 2022) and ResZ transform the value function or the value distribution into a combination of the principal and the residual functions. Since QMIX and its successor algorithms require the use of global states, the introduction of global states incurs a significant time overhead. For simple tasks, VDN algorithms are both fast and efficient. Based on the task characteristics of MeltingPot, we choose a VDN algorithm that better meets our training needs for value decomposition.

Background

Markov decision process

In fully observed environments, stationary optimal policies exist. In partially observed environments, the policy usually incorporates past agent observations from the history $h_t = \{a_1, o_1, r_1, ..., a_{t-1}, o_{t-1}, r_{t-1}\}$ A practical approach utilized here, is to parameterize policies using recurrent neural networks.

We consider a partially observable scenario in which each agent draws individual observations $z \in Z$ according to observation function $O\left(s,a\right) : S \times A \rightarrow Z$. Each agent has an action-observation history which it conditions a stochastic policy π The joint policy π has a joint action-value function: $Q_\pi\left(s_t, ut\right) = E_{s_{t+1}:\infty, u_{t+1}:\infty}\left[\gamma R_t | s_t, u_t\right]$, where $R_t = \sum_{i=1}^{\infty} \gamma_i r_{t+i}$is the discounted reward.

Multi-Agent Reinforcement Learning

We consider problems where observations and actions are distributed across d agents, and are represented as d-dimensional tuples of primitive observations in O and actions in A. As is standard in MARL, the underlying environment is modeled as a Markov game where actions are chosen and executed simultaneously, and new observations are perceived simultaneously as a result of a transition to a new state

Although agents have individual observations and are responsible for individual actions, each agent only receives the joint reward, and we seek to optimize Rt as defined above. This is consistent with the Dec-POMDP framework (Oliehoek et al., 2008; Oliehoek and Amato, 2016)

Method

Approach Overview

Each agent's policy is implemented as a concatenation of a ResNet module, a GRU layer, and an MLP layer. The MLP layer outputs the Q value of each action. For each society

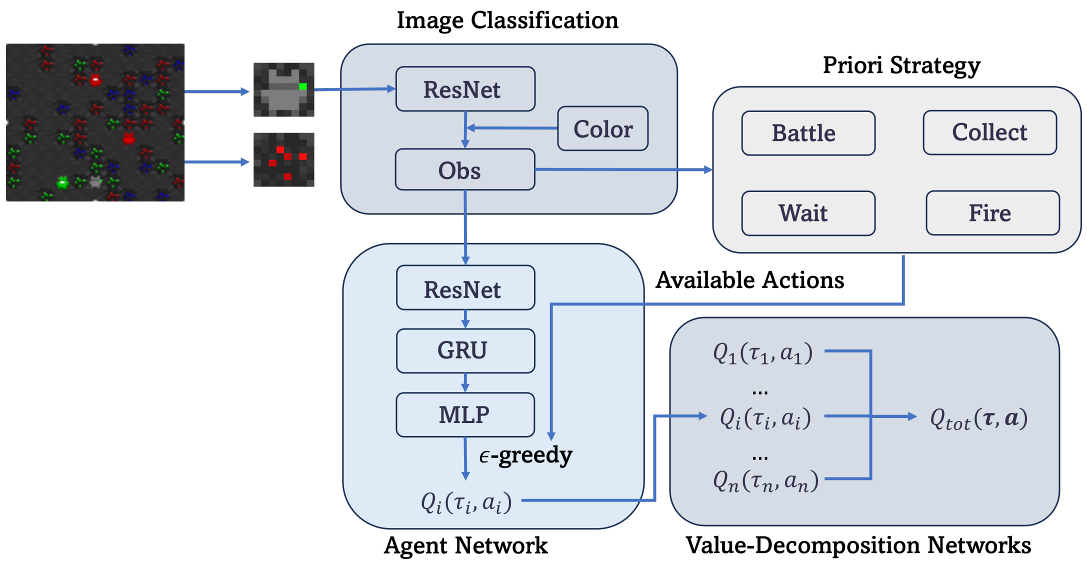

Figure 1: There are mainly three parts in our solution. Here is the image classification. We use ResNet to make every unit in the picture into a tensor. Then to make our agents more capable to generalize along social dimesnions, we set some priro strategies to regularize their actions. Finally, we use a value-decomposition network to solve the problem of credit assignment.

game, we use a pre-defined prior strategy to select the best action for each agent.

Regarding mixed motivation considerations, we assume that each agent will engage in cooperative behavior and then consider their competitive behavior. For cooperative behavior, adopt the popular Centralized Training with Decentralized Execution (CTDE) approach and use the VDN value factorization method.

In some environments, competitive behaviors could lead to negative effects because there are no competitors at all. To address this problem, we added some NPC agents during training; these NPC perform some collection activities, which lead to improved performance of the focused agents.

Our overall framework diagram is shown in Figure 1.

Value Decomposition Networks

By contrast, value decomposition networks (VDNs) (Sunehag et al., 2017) aim to learn a joint action-value function $Q_{tot}(\tau, u)$, where $\tau \in T.\tau_n$ is a joint action observation history and u is a joint action. It represents $Qtot$ as a sum of individual value functions one for each agent a, that condition only on individual action-observation histories:

$$Q_{tot}(\tau, u) = \sum_{i=1}^{n} Q_i(\tau^i, u^i; \theta^i)$$

The loss function in VDN training is:

$$L(\theta) = \sum_{i=1}^{b} \left[(y_i^{tot} - Q_{tot}(\tau, u, s; \theta))^2 \right]$$

where b is the batch size of transitions sampled from the replay buffer, $y^{tot} = r + \gamma \mathbf{max}_{u'} Q_{tot}(\tau', u', s'; \theta^-)$ and θ^- are the parameters of a target network as in DQN. Monotonicity can be enforced through a constraint on the relationship between Q_{tot} and each Q_a:

$$\frac{\partial Q_{tot}}{\partial Q_a} \geq 0, \forall a \in A$$

There are currently more multi-agent value decomposition methods such as QMIX, Since these algorithms needs the information of global state and the global state in this environments are relatively large. Therefore, it is complicated for us to deal with the relative position between agents, we use the VDN to decompose.

Agent Network

We use another ReNet to process the observation data, input them into the GRU for long-term memory, and use the epsilon-Greedy algorithm to select the action with the greatest value.

Due to the limitation of GPU space, we used ResNet18 and reduced some of the residual connection layers.

The epsilon-Greedy algorithm chooses to explore new possibilities part of the time and exploit the best known options part of the time. By adjusting the value of epsilon, the exploration and exploitation trade-offs can be balanced to suit different problems and environments.

$$Action\, at\, time(t) = \begin{cases} \mathbf{max}_a Q_t(a), & with\, prob\, 1-\epsilon \\ any\, action(a), & with\, prob\, \epsilon \end{cases}$$

Image classification

In order to facilitate the transformation from Vision-based Reinforcement Learning (Vision RL) to State-based Reinforcement Learning (State RL), the preprocessing of RGB images obtained from the environment is quintessential.

The 11x11 RGB images are methodically dissected into smaller blocks. Each block is then processed through a classification algorithm employing a Residual Neural Network (ResNet) architecture, which is adept at identifying the categorical nature of items depicted in the images.

To streamline this process, a dataset comprising 1000 timesteps randomly sampled from environmental interactions is compiled, with each timestep representing an individual RGB image. These images are then meticulously annotated by domain experts, providing accurately labeled data to underpin the supervised learning tasks. This work is instrumental in translating the complex visual inputs into a structured state representation that can be effectively utilized for state-based reinforcement learning algorithms.

Generalization training

In the MeltingPot environment, which features background players, it is imperative for agents to learn mixed-motivation strategies to navigate the complex interplay of cooperative and competitive dynamics. However, the prevalence of competitive behaviors in the absence of actual competitors can have deleterious effects on the cohesion and overall performance of the agents. To mitigate this issue and enhance the generalization capabilities of our agents, we introduced non-player characters governed by rudimentary artificial intelligence algorithms into the environment. Here are some settings for generalization training.

Environment	Focal	Background
Allelopathic Harvest	6	2
Clean Up	4	1
Prisoners Dilemma in the matrix	7	1
Territory:Rooms	1	0

Table 1: Training population configuration

These robots with simple intelligence serves as a catalyst for the agents to develop nuanced strategies that are not overly specialized to particular opponent types, thereby enhancing their versatility and efficacy in the MeltingPot environment.

Prior Strategies

For each environment, we apply different prior strategies for competitive/cooperative behaviors.

Allelopathic Harvest This environment is as shown in Figure 2. A challenging problem in this environment is how to distinguish the focal players from background players. So we considered a interesting strategy. When the berries are uniform in color, the focal player changes its color to an unpopular blue color, thereby distinguishing the focal players from the background players.

Clean Up This environment is as shown in Figure 3. We give certain penalties to players that stay in open space, empty orchard, and clean rivers and add a selfish background player during training to motivate the player's behavior. In this environment, we basically did not adopt any other prior strategies.

Prisoners Dilemma in the matrix This environment is as shown in Figure 4. We want the focal players to cooperate with other focal players but weakly cooperate or even strongly defect other background players. We want the player's strategy to be as pure as possible. when each player is born, its strategy is defined as either cooperation or defection, and it will only collect resources that are the same as its strategy. In order to prevent the focal players from defecting each other, we set the focal players that choose the defect strategy to be unable to actively interact.

Territory:Rooms This environment is as shown in Figure 5. In our opinion, the players in this environment should be extremely aggressive. We found a way to differentiate the focal players from the background players. At the beginning, players only color the walls in the middle of the four directions. During this process, we can identify focal players who behave similarly from observations. After completing this operation, we can proceed with the normal activities, trying to encroach on the territory of other players.

Experiment

The Melting Pot framework comprises test scenarios and a corresponding protocol designed to assess the generalization abilities of a specific group of agents in diverse social situations. Each scenario is composed of a substrate, representing the physical aspects of the environment, and a background population, which possesses agency but is distinct from the focal population undergoing evaluation. During training, the focal population interacts with the substrate, but it never encounters individuals from the background population. The goal is to evaluate how well the focal population's agents can adapt to and generalize in novel social situations that were not part of their direct training experiences.

To contextualize the range of agent returns, the focal returns are normalized to get a performance score that is between 0 (for the worst agent) and 1 (for the best agent).

Allelopathic Harvest

In Allelopathic Harvest, there are three berry varieties: red, green, and blue. The ripening rate of each variety depends on its proportion in the total. The goal for the sixteen players is to increase the overall ripening rate by adjusting the proportions, considering their individual preferences for a specific berry variety.

However, the red and blue serial numbers were reversed during the visualization process. We were not able to discover this problem until the last day of the competition, so we achieved quite poor performance in this environment.

Clean Up

Clean Up is a seven-player game where players earn rewards by collecting apples in an orchard. The orchard's apple growth rate is inversely related to river cleanliness, with pollution accumulating at a constant rate. There's a conflict between short-term individual incentives to collect apples

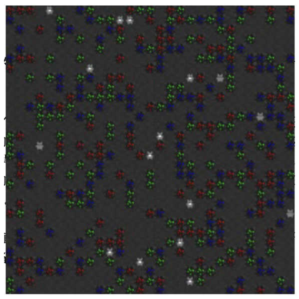
Figure 2: Allelopathic Harvest

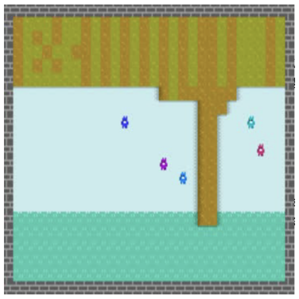
Figure 3: Clean Up

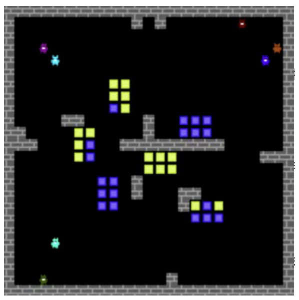
Figure 4: Pd in the matrix

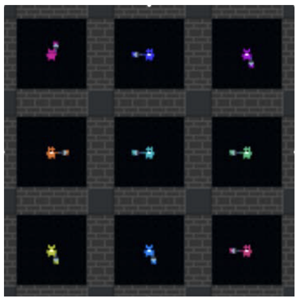
Figure 5: Territory:Rooms

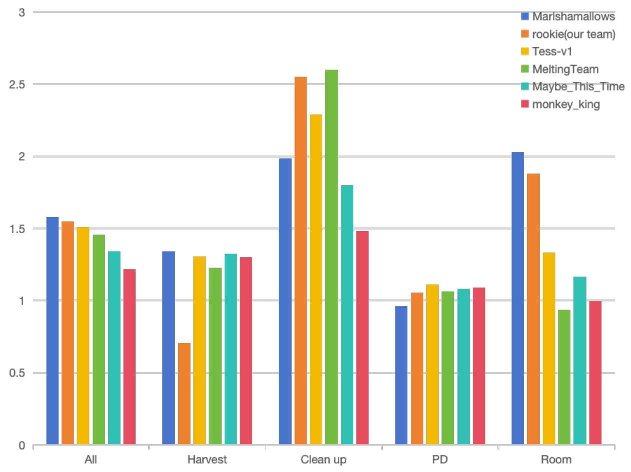

Figure 6: Our team achieved an overall second-place ranking, with standout performances in the "Clean up" and "Room" tasks, yet faced challenges in the "Harvest" task due to a color-coding error, affecting our scores. We performed comparably to top teams in the "PD" scenario.

and the long-term group interest in maintaining a clean river for continuous apple growth.

We achieve better performance because our multi-agent reinforcement learning method promotes cooperation between agents well.

Prisoners Dilemma in the matrix

In this scenario, each player's reward is based on the expected payoff from a mixed strategy in the matrix game, determined by their inventory objects. Resources correspond one-to-one to pure strategies, and the mixed strategy depends on the resources a player picks up. The environment is similar to the Prisoner's Dilemma, highlighting the tension between individual and group rewards in this eight-player game. The matrix for the interaction is

$$A_{row} = A_{col}^T = \begin{vmatrix} 3 & 0 \\ 5 & 1 \end{vmatrix}$$

In this environment, the difference between everyone's scores is not clear, and there should be better strategies waiting for us to discover.

Territory:Rooms

In this scenario, players can claim resources by touching them or using a claiming beam. Claimed resources provide stochastic rewards to the claiming player. Zapping a resource twice permanently destroys it, removing its function as a wall or resource. Players hit by a zapping beam are permanently removed from the game, with claimed resources reverting to an unclaimed state.

Our prior strategy can distinguish the background population and the focus population, avoid the battle of the focus population, and therefore achieve better results.

Conclusion

In this paper, we have investigated the complex landscape of multi-agent mixed-motivation cooperation within the context of the NeurIPS 2023 Melting Pot Contest. The deep reinforcement learning framework we proposed has demonstrated efficacy by securing the runner-up position in the final round, signifying a substantial contribution to the advancement of cooperative intelligence in AI. Technically, our integration of an enhanced ResNet with GRU and MLP layers to formulate agent policies, combined with the CTDE strategy and VDN value decomposition, has proven to be particularly effective in fostering adaptability and cooperative strategies among unknown entities.

Our approach has identifiable shortcomings in orientation perception, visualization configuration, and the accuracy of adversarial prediction, highlighting the limitations of our current research. Addressing these issues is a priority for our future work, as we strive to enhance the performance and adaptability of AI systems in complex environments through these improvements. Despite these challenges, we have made significant strides in the field of cooperative intelligence and look forward to further expanding upon these achievements in future research.

Achieving second place in the Melting Pot Contest underscores the robustness of our approach, demonstrating its capacity to handle dynamic cooperative contexts and to compete at an international level against state-of-the-art methodologies. This accomplishment not only reflects the depth and practicality of our research but also paves the way for future endeavours in effective multi-agent cooperation in unknown and dynamic settings.

References

Baker, B.; Kanitscheider, I.; Markov, T.; Wu, Y.; Powell, G.; McGrew, B.; and Mordatch, I. 2019. Emergent tool use from multi-agent autocurricula. *arXiv preprint arXiv:1909.07528*.

Balduzzi, D.; Garnelo, M.; Bachrach, Y.; Czarnecki, W.; Perolat, J.; Jaderberg, M.; and Graepel, T. 2019. Open-ended learning in symmetric zero-sum games. In *International Conference on Machine Learning*, 434–443. PMLR.

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

Foerster, J. N.; Chen, R. Y.; Al-Shedivat, M.; Whiteson, S.; Abbeel, P.; and Mordatch, I. 2017. Learning with opponent-learning awareness. *arXiv preprint arXiv:1709.04326*.

He, K.; Zhang, X.; Ren, S.; and Sun, J. 2016. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, 770–778.

Hu, J.; Jiang, S.; Harding, S. A.; Wu, H.; and Liao, S.-w. 2021. Rethinking the implementation tricks and monotonicity constraint in cooperative multi-agent reinforcement learning. *arXiv preprint arXiv:2102.03479*.

Leibo, J. Z.; Perolat, J.; Hughes, E.; Wheelwright, S.; Marblestone, A. H.; Duéñez-Guzmán, E.; Sunehag, P.; Dunning, I.; and Graepel, T. 2018. Malthusian reinforcement learning. *arXiv preprint arXiv:1812.07019*.

Lowe, R.; Wu, Y. I.; Tamar, A.; Harb, J.; Pieter Abbeel, O.; and Mordatch, I. 2017. Multi-agent actor-critic for mixed cooperative-competitive environments. *Advances in neural information processing systems*, 30.

Rashid, T.; Samvelyan, M.; De Witt, C. S.; Farquhar, G.; Foerster, J.; and Whiteson, S. 2020. Monotonic value function factorisation for deep multi-agent reinforcement learning. *The Journal of Machine Learning Research*, 21(1): 7234–7284.

Russell, S. J.; and Zimdars, A. 2003. Q-decomposition for reinforcement learning agents. In *Proceedings of the 20th International Conference on Machine Learning (ICML-03)*, 656–663.

Schneider, J.; Wong, W.-K.; Moore, A.; and Riedmiller, M. 1999. Distributed value functions.

Shen, S.; Qiu, M.; Liu, J.; Liu, W.; Fu, Y.; Liu, X.; and Wang, C. 2022. ResQ: A Residual Q Function-based Approach for Multi-Agent Reinforcement Learning Value Factorization. *Advances in Neural Information Processing Systems*, 35: 5471–5483.

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.; 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.

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*, 5887–5896. PMLR.

Sunehag, P.; Lever, G.; Gruslys, A.; Czarnecki, W. M.; Zambaldi, V.; Jaderberg, M.; Lanctot, M.; Sonnerat, N.; Leibo, J. Z.; Tuyls, K.; et al. 2017. Value-decomposition networks for cooperative multi-agent learning. *arXiv preprint arXiv:1706.05296*.

Sutton, R. S.; and Barto, A. G. 2018. *Reinforcement learning: An introduction*. MIT press.

Vinyals, O.; Babuschkin, I.; Czarnecki, W. M.; Mathieu, M.; Dudzik, A.; Chung, J.; Choi, D. H.; Powell, R.; Ewalds, T.; Georgiev, P.; et al. 2019. Grandmaster level in StarCraft II using multi-agent reinforcement learning. *Nature*, 575(7782): 350–354.

Wooldridge, M. 2009. *An introduction to multiagent systems*. John wiley & sons.

Yang, Y.; Hao, J.; Liao, B.; Shao, K.; Chen, G.; Liu, W.; and Tang, H. 2020. Qatten: A general framework for cooperative multiagent reinforcement learning. *arXiv preprint arXiv:2002.03939*.

Ye, D.; Chen, G.; Zhang, W.; Chen, S.; Yuan, B.; Liu, B.; Chen, J.; Liu, Z.; Qiu, F.; Yu, H.; et al. 2020. Towards playing full moba games with deep reinforcement learning. *Advances in Neural Information Processing Systems*, 33: 621–632.