PMAT: Optimizing Action Generation Order in Multi-Agent Reinforcement Learning

Kun Hu
National University of Defense Technology
Changsha, China
khu@nudt.edu.cn

Muning Wen
Shanghai Jiao Tong University
Shanghai, China
muningwen@sjtu.edu.cn

Xihuai Wang
Shanghai Jiao Tong University
Shanghai, China
leoxhwang@sjtu.edu.cn

Shao Zhang
Shanghai Jiao Tong University
Shanghai, China
shaozhang@sjtu.edu.cn

Yiwei Shi
University of Bristol
Bristol, United Kingdom
yiwei.shi@bristol.ac.uk

Minne Li
Intelligent Game and Decision Lab
Beijing, China
lmn.2011@tsinghua.org.cn

Minglong Li*
National University of Defense Technology
Changsha, China
liminglong10@nudt.edu.cn

Ying Wen*
Shanghai Jiao Tong University
Shanghai, China
ying.wen@sjtu.edu.cn

ABSTRACT

Multi-Agent Reinforcement Learning (MARL) faces challenges in coordinating agents due to complex interdependencies within multi-agent systems. Most MARL algorithms use the simultaneous decision-making paradigm but ignore the action-level dependencies among agents, which reduces coordination efficiency. In contrast, the sequential decision-making paradigm provides finer-grained supervision for agent decision order, presenting the potential for handling dependencies via better decision order management. However, determining the optimal decision order remains a challenge. In this paper, we introduce **Action Generation with Plackett-Luce Sampling (AGPS)**, a novel mechanism for agent decision order optimization. We model the order determination task as a Plackett-Luce sampling process to address issues such as ranking instability and vanishing gradient during the network training process. AGPS realizes credit-based decision order determination by establishing a bridge between the significance of agents' local observations and their decision credits, thus facilitating order optimization and dependency management. Integrating AGPS with the Multi-Agent Transformer, we propose the **Prioritized Multi-Agent Transformer (PMAT)**, a sequential decision-making MARL algorithm with decision order optimization. Experiments on benchmarks including StarCraft Multi-Agent Challenge, Google Research Football, and Multi-Agent MuJoCo show that PMAT outperforms state-of-the-art algorithms, greatly enhancing coordination efficiency.

*Correspondence to Minglong Li <liminglong10@nudt.edu.cn> and Ying Wen <ying.wen@sjtu.edu.cn>.

Proc. of the 24th International Conference on Autonomous Agents and Multiagent Systems (AAMAS 2025), Y. Vorobeychik, S. Das, A. Nowé (eds.), May 19 – 23, 2025, Detroit, Michigan, USA. © 2025 International Foundation for Autonomous Agents and Multiagent Systems (www.ifaamas.org).

KEYWORDS

Multi-agent reinforcement learning; Action generation order

ACM Reference Format:
Kun Hu, Muning Wen, Xihuai Wang, Shao Zhang, Yiwei Shi, Minne Li, Minglong Li, and Ying Wen. 2025. PMAT: Optimizing Action Generation Order in Multi-Agent Reinforcement Learning. In *Proc. of the 24th International Conference on Autonomous Agents and Multiagent Systems (AAMAS 2025), Detroit, Michigan, USA, May 19 – 23, 2025*, IFAAMAS, 9 pages.

1 INTRODUCTION

In a multi-agent system (MAS), the optimal action of one agent is often affected by the behavior of others [39], creating complex inter-agent dependency relationships [4, 12]. Therefore, a key challenge of Multi-Agent Reinforcement Learning (MARL) [33, 38] algorithms is handling the inter-agent dependencies to manage coordination [27, 33]. The dependency relationships among agents necessitate optimizing the decision-making order to achieve optimal team strategies in multi-agent cooperation tasks [4]. As illustrated in Figure 1, MARL algorithms typically employ two decision-making paradigms that generate agents' actions either simultaneously [32, 37] or sequentially [34]. Although simultaneously generating the actions of all agents can facilitate collective learning, it overlooks the potential action-level order dependencies within an MAS. Consequently, the newly generated action of the concurrent agent may offset the overall performance improvement established by previous agents, resulting in degradation of coordination efficiency [1].

Fortunately, the recent incorporation of sequence models (SMs) in reinforcement learning effectively facilitates sequential decision-making [6, 13]. In MARL, the utilization of the auto-regressive token generalization mechanism of SMs allows each agent to leverage prior agents' actions during its decision-making process [18, 35], which enables handling dependencies of agents in a sequential manner. While the sequential paradigm holds the potential for effective dependency management, identifying the optimal decision order

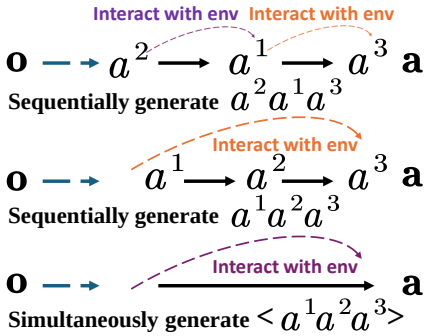


Figure 1: The simultaneous action generation paradigm generates agents' actions concurrently and interacts with the environment once. The sequential action generation paradigm generates agents' actions in an agent-by-agent manner, providing finer-grained supervision for the action generation order. Agents can interact with the environment once per decision or once per iteration under this paradigm.

presents a formidable challenge. Typically, a fixed or randomized action generation order [34] is adopted in this paradigm. The fixed-order decision-making scheme is limited by its inability to adapt to agents' dynamically evolving action-level dependencies, which leads to sub-optimal algorithm performance. For instance, in a football match, the possession of the ball frequently shifts between players as the game progresses. If decisions are made following a fixed order within the team, players might not respond effectively to real-time changes, resulting in poor coordination.

Although the randomized-order decision-making scheme enables dynamic ordering, learning algorithms may converge to local optima due to random sampling that inadequately represents the solution space or fails to converge to optimal solutions under sub-optimal settings [29]. Consequently, sequential decision-making in the MARL domain remains fundamentally constrained and less effective without a robust mechanism to determine the optimal decision order. Hence, there is a pressing need to develop an adaptive decision-ordering mechanism that can effectively manage the dynamic action-level order dependencies among agents.

While deep learning-based methods have shown remarkable progress in addressing ordering issues [8, 22], learning the optimal agent decision order still faces several key challenges. On the one hand, directly ordering agents according to their preference scores learned by deep neural networks presents a vanishing gradient issue since the variations of network outputs may not alter the ordering results. On the other hand, when agents declare similar ranking scores, even minor fluctuations in scoring can significantly change the final orderings, thus introducing instability issues.

To tackle these challenges, we introduce **Action Generation with Plackett-Luce Sampling (AGPS)**, a Plackett-Luce (P-L) model-based sequential decision-making scheme in MARL. Specifically, we formulate the order determination task as an agent-by-agent sampling process and utilize P-L sampling [17, 23] in decision order optimization, which facilitates robust and adaptive decision-ordering in multi-agent cooperation tasks. Facilitated by AGPS, we propose **Prioritized Multi-Agent Transformer (PMAT)**, a sequential decision-making MARL algorithm with decision order optimization. We evaluate the proposed algorithm on popular MARL benchmarks including StarCraft Multi-Agent Challenge [24], Google Research Football [15], and Multi-Agent MuJoCo [9], where PMAT consistently demonstrates superior task performance compared with several state-of-the-art MARL algorithms.

2 RELATED WORK

In this section, we introduce several representative state-of-the-art MARL algorithms, covering both the simultaneous and the sequential decision-making paradigms. We also discuss the difference between several types of order optimization in MARL.

Simultaneous Decision-Making MARL Algorithms. The vast majority of *Centralized Training Decentralized Execution* (CTDE) [10, 19, 20] algorithms in MARL adopt a simultaneous decision-making paradigm. Here we introduce two representative ones. MAPPO [37] is a straightforward policy-based approach that endows the policy network of all agents with a shared set of parameters and utilizes agents' aggregated trajectories to facilitate policy optimization. HAPPO [14] is a heterogeneous-agent trust-region method that employs a sequential policy update paradigm. During an update in HAPPO, the agents randomly choose an update order and update their own policies over the newly updated policies of previous agents. Due to the adoption of the simultaneous decision-making paradigm, both MAPPO and HAPPO suffer from potential action conflicts and lack coordination efficiency guarantee.

Sequential Decision-Making MARL Algorithm. To alleviate potential action conflicts and further enhance multi-agent coordination, Wen et al. [34] proposed Multi-Agent Transformer (MAT), which presents an auto-regressive sequential decision-making MARL algorithm based on the *Transformer* [31] architecture. MAT successfully transforms multi-agent joint policy optimization into a sequential decision-making process by generating actions in an agent-by-agent manner, which holds the potential for finer-grained supervision and management of inter-agent dependencies.

Different Order Optimization in MARL. The current MARL algorithms have started to focus on the impact of "order" on agent-level or batch-level updates, and several solutions have been proposed. Wang et al. [32] proposed an agent-by-agent policy optimization method, A2PO, which adopts a semi-greedy agent selection rule to determine agent update order within a single rollout. Furthermore, B2MAPO [40] establishes update batches, further enhancing algorithm efficiency to facilitate joint policy optimization in larger-scale agent clusters. These studies on sequential update MARL algorithms offer valuable insights, suggesting that decision order optimization can be achieved incrementally on an item-by-item basis. Unlike these works, this paper focuses on optimizing agent decision order under the sequential decision-making paradigm, which remains an underexplored area in MARL.

3 PRELIMINARIES & BACKGROUND

3.1 Cooperative MARL Problem Formulation

Cooperative MARL problems can usually be modeled as *Markov games* $< \mathcal{N}, \boldsymbol{O}, \boldsymbol{\mathcal{A}}, R, P, \gamma >$ [16], where $\mathcal{N} = \{1, \ldots, n\}$ is the set of agents, $\boldsymbol{O} = \prod_{i=1}^{n} O^i$ is the joint observation space, $\boldsymbol{\mathcal{A}} = \prod_{i=1}^{n} \mathcal{A}^i$ is the joint action space, $R : \boldsymbol{O} \times \boldsymbol{\mathcal{A}} \longrightarrow \mathbb{R}$ is the joint reward function, $P : \boldsymbol{O} \times \boldsymbol{\mathcal{A}} \times \boldsymbol{O} \longrightarrow [0, 1]$ is the transition probability function and $\gamma \in [0, 1)$ is the discount factor. Within each time

step, all agents act simultaneously based on their observations. At time step $t \in \mathbb{N}$, each agent i ($i \in \mathcal{N}$) obtains its individual local observation $\mathrm{o}_t^i \in \boldsymbol{O}^i$ and takes an action $\mathrm{a}_t^i \in \mathcal{A}^i$ according to its own policy π_i, which represents a component of the joint policy $\boldsymbol{\pi}$.

We consider a fully cooperative setting where all agents share the same reward function. When time step t ends, the whole team receives a joint reward $R(\mathbf{o}_t, \mathbf{a}_t)$ and observes $\mathbf{o}_{t+1}$ whose probability distribution is $P(\cdot|\mathbf{o}_t, \mathbf{a}_t)$. Following infinitely long times of this process, the multi-agent team finally gains a cumulative return of $R^\gamma \triangleq \sum_{t=0}^{\infty} \gamma^t R(\mathbf{o}_t, \mathbf{a}_t)$. The observation value and the observation-action value can then be defined as

$$V_{\boldsymbol{\pi}}(\boldsymbol{o}) \triangleq \mathbb{E}_{\mathbf{o}_{1:\infty} \sim P, \mathbf{a}_{0:\infty} \sim \boldsymbol{\pi}}[R^\gamma | \mathbf{o}_0 = \boldsymbol{o}] \tag{1}$$

and

$$Q_{\boldsymbol{\pi}}(\boldsymbol{o}, \boldsymbol{a}) \triangleq \mathbb{E}_{\mathbf{o}_{1:\infty} \sim P, \mathbf{a}_{1:\infty} \sim \boldsymbol{\pi}}[R^\gamma | \mathbf{o}_0 = \boldsymbol{o}, \mathbf{a}_0 = \boldsymbol{a}] \tag{2}$$

respectively. And the advantage value is defined as

$$A_{\boldsymbol{\pi}}(\boldsymbol{o}, \boldsymbol{a}) \triangleq Q_{\boldsymbol{\pi}}(\boldsymbol{o}, \boldsymbol{a}) - V_{\boldsymbol{\pi}}(\boldsymbol{o}). \tag{3}$$

3.2 Multi-Agent Advantage Decomposition

In this work, we pay close attention to the impact of action generation order on multi-agent joint advantage improvement in MARL. Before proceeding to our methods, in this section we first introduce existing definitions and theorems as follows:

Definition 1 (Multi-Agent Advantage Function [14]). *Let $i_{1:m}$ denote an ordered subset $\{i_1, \ldots, i_m\}$ of $\mathcal{N}$ and $-i_{1:m}$ denote its complement. The multi-agent observation-action value function is defined as*

$$Q_{\boldsymbol{\pi}}^{i_{1:m}}(\boldsymbol{o}, \boldsymbol{a}^{i_{1:m}}) \triangleq \mathbb{E}_{\boldsymbol{a}^{-i_{1:m}} \sim \boldsymbol{\pi}^{-i_{1:m}}} \left[Q_{\boldsymbol{\pi}}(\boldsymbol{o}, \boldsymbol{a}^{i_{1:m}}, \boldsymbol{a}^{-i_{1:m}}) \right].$$

Let $j_{1:k}$ denote another ordered subset of $\mathcal{N}$, such that $i_{1:m} \cap j_{1:k} = \emptyset$. Then, the multi-agent advantage function is defined as

$$A_{\boldsymbol{\pi}}^{i_{1:m}}(\boldsymbol{o}, \boldsymbol{a}^{j_{1:k}}, \boldsymbol{a}^{i_{1:m}}) \triangleq Q_{\boldsymbol{\pi}}^{j_{1:k}, i_{1:m}}(\boldsymbol{o}, \boldsymbol{a}^{j_{1:k}}, \boldsymbol{a}^{i_{1:m}}) - Q_{\boldsymbol{\pi}}^{j_{1:k}}(\boldsymbol{o}, \boldsymbol{a}^{j_{1:k}}).$$

Definition 1 describes the contribution of agents $i_{1:m}$ taking actions $\boldsymbol{a}^{i_{1:m}}$ once agents $j_{1:k}$ have taken actions $\boldsymbol{a}^{j_{1:k}}$, thus facilitating multi-agent joint policy optimization via the following theorem of

Theorem 3.1 (Multi-Agent Advantage Decomposition [34]). *Let $i_{1:m}$ be a permutation of agents and i_k denote the k^{th} agent within $i_{1:m}$. Then, for joint observation $\boldsymbol{o} = \boldsymbol{o} \in \boldsymbol{O}$ and joint action $\boldsymbol{a} = \boldsymbol{a}^{i_{1:m}} \in \mathcal{A}$, the following equation always holds,*

$$A_{\boldsymbol{\pi}}^{i_{1:m}}(\boldsymbol{o}, \boldsymbol{a}^{i_{1:m}}) = \sum_{k=1}^{m} A_{\boldsymbol{\pi}}^{i_k}(\boldsymbol{o}, \boldsymbol{a}^{i_{1:k-1}}, a^{i_k}).$$

Theorem 3.1 provides an intuitive guide for joint policy optimization within a multi-agent team. It suggests that a sequential optimization of each agent's action contingent upon the actions of preceding agents can finally improve the joint advantage. Hence, one major strength of MARL algorithms adopting the sequential action generation paradigm lies in the potential to ensure that each agent i_j achieves a positive advantage upon the actions $\boldsymbol{a}^{i_{1:j-1}}$ of its predecessors via sequential decision-making. As an example, MAT [34] utilizes an auto-regressive token generation mechanism to ensure that each agent achieves a positive advantage based on previous agents' actions during the decision-making process.

3.3 Multi-Agent Transformer

Multi-Agent Transformer (MAT) [34] is a successful implementation of the **encoder-decoder** architecture of the *Transformer* [31] in MARL. The attention mechanism of MAT first encodes agents' observations and actions with a weight matrix calculated by multiplying the embedded queries and keys. Subsequently, representations are calculated by multiplying the weight matrix with the embedded values. In general, the encoder of MAT takes a sequence of observations $(o^{i_1}, \ldots, o^{i_n})$ as input and passes them through several computational blocks to generate the corresponding observation representations $(\hat{o}^{i_1}, \ldots, \hat{o}^{i_n})$. Each of these computational blocks consists of an unmasked self-attention mechanism and a multi-layer perceptron (MLP) to extract the interrelationship among agents. The encoder is trained to approximate the value functions by minimizing the empirical Bellman error of

$$L_{\text{Encoder}}(\phi) = \frac{1}{Tn} \sum_{m=1}^{n} \sum_{t=0}^{T-1} \left[R(\mathbf{o}_t, \mathbf{a}_t) + \gamma V_{\bar{\phi}}(\hat{\mathrm{o}}_{t+1}^{i_m}) - V_\phi(\hat{\mathrm{o}}_t^{i_m}) \right]^2, \tag{4}$$

where ϕ denotes the network parameter and $\bar{\phi}$ denotes the target network parameter. The decoder of MAT receives the observation representations output by the encoder. It sequentially generates and passes the embedded actions of agents $\boldsymbol{a}^{i_{0:m-1}}$ $(m = 1, \ldots n)$ through a sequence of decoding blocks, where a^{i_0} is an arbitrary symbol indicating the start of decoding. Every decoding block is equipped with a masked self-attention mechanism utilizing triangular matrices to ensure that for each agent i_j attention is computed between the i_r^{th} and the i_j^{th} action heads ($r < j$) so that the sequential scheme can be maintained. The decoding block finally finishes with an MLP and skipping connections, generating a sequence of multi-agent joint action. Parameterized by θ, the decoder is trained to minimize the following clipping PPO [26] objective of

$$L_{\text{Decoder}}(\theta) = -\frac{1}{Tn} \sum_{m=1}^{n} \sum_{t=0}^{T-1} \min \left(\mathrm{r}_t^{i_m}(\theta) \hat{A}_t, \text{clip}(\mathrm{r}_t^{i_m}(\theta), 1 \pm \epsilon) \hat{A}_t \right), \tag{5}$$

where

$$\mathrm{r}_t^{i_m}(\theta) = \frac{\pi_\theta^{i_m}(\mathrm{a}_t^{i_m} | \hat{\mathbf{o}}_t^{i_{1:n}}, \hat{\mathbf{a}}_t^{i_{1:m-1}})}{\pi_{\theta_{\text{old}}}^{i_m}(\mathrm{a}_t^{i_m} | \hat{\mathbf{o}}_t^{i_{1:n}}, \hat{\mathbf{a}}_t^{i_{1:m-1}})}, \tag{6}$$

and $\hat{A}_t$ represents an estimate of the joint advantage function. To estimate the joint value function, *generalized advantage estimation* (GAE) [25] can be applied with $\hat{V}_t = \frac{1}{n} \sum_{m=1}^{n} V(\hat{\mathrm{o}}_t^{i_m})$.

4 DECISION ORDER MATTERS

Although the positive-advantage decision-making scheme confers a monotonic improvement guarantee upon MAT, it fails to maximize the joint advantage achieved in each iteration. This stems from lacking effective management of the action-level dependencies among agents. Specifically, if the optimal action of agent i_j depends upon the action of agent i_k who plays a pivotal role, enabling i_k to decide prior to i_j provides essential decision-making information for i_j, thus holding the potential to enhance the overall team performance (which can also be evidenced by *Example 3*, [1]).

As an illustrative example, in Figure 2, we take two frames from the *academy pass and shoot with keeper* scenario of Google Research

Figure 2: A multi-agent cooperation scenario taken from Google Research Football. Player JOHNSON passes the ball to his partner TURING who has a favorable shooting angle (left), and TURING converts the shot into a goal (right).

Football [15], where the local observation of Player TURING exhibits prior significance due to his advantageous positioning for scoring a goal. In this case, TURING is allowed to decide first and he decides to take a shot as illustrated in Figure 2, where decisions are specially plotted in dashed arrows. Subsequently, TURING's teammate JOHNSON decides to pass the ball, considering both TURING's decision (shoot) and his own observation (the position of TURING and the opponent player etc.). In such a prioritized sequential decision-making manner, subsequent players hold the potential to recognize the intentions of preceding players who possess more significant local observations and align their actions with these predecessors in an efficient way, thus facilitating the emergence of collaborative behavior among agents and enhancing the overall task performance of the whole team. Later, we will build upon this insight to introduce a sequential action generation scheme that optimizes the agent decision order according to the significance of their local observations to the joint advantage.

To further discuss the impact of decision order, we define the *action generation order* σ as a permutation of $\mathcal{N}$, which induces the multi-agent joint policy (contingent upon this order) as

$$\boldsymbol{\pi}^{\sigma}_{i_{1:n}} = \pi_{i_1}(a^{i_1}|\boldsymbol{o}, \sigma) \cdot \pi_{i_2}(a^{i_2}|\boldsymbol{o}, \sigma, a^{i_1}) \cdot \ldots \cdot \pi_{i_n}(a^{i_n}|\boldsymbol{o}, \sigma, a^{i_1}, \ldots, a^{i_{n-1}}). \tag{7}$$

Then, we define the *optimal action generation order* as

Definition 2 (Optimal Action Generation Order). *Given a group of agents marked as* $\{1, \ldots, n\}$*, if an action generation order* $\sigma^* = \{i_1, i_2, \ldots, i_n\}$ *satisfies* $A^{1:n}_{\boldsymbol{\pi}^{\sigma^*}}(\boldsymbol{o}, \boldsymbol{a}^{1:n}_{\sigma^*}) \geq A^{1:n}_{\boldsymbol{\pi}^{\sigma}}(\boldsymbol{o}, \boldsymbol{a}^{1:n}_{\sigma})$ *for any* $\sigma \neq \sigma^*$*, then we define* σ^* *as the Optimal Action Generation Order.*

Applying different action generation orders can result in distinct joint action, ultimately affecting the multi-agent joint advantage achieved in each iteration. Hence, optimizing agent decision order is significant to joint advantage optimization in MARL. Definition 2 defines the optimal agent decision order, utilizing the joint advantage value as feedback signal. Our goal is to identify and utilize an optimized decision-making order as the action generation order in each round of the multi-agent sequential decision-making process. By utilizing this refined order, we aim to optimize the joint advantage achieved in each iteration of multi-agent joint action, thereby enhancing the efficiency and task performance of MARL algorithms.

5 DECISION ORDER OPTIMIZATION

In this section, we first discuss the challenges of learning the optimal decision order and highlight the advantages of utilizing the Plackett-Luce (P-L) model for decision order optimization. We then introduce AGPS, a P-L model-based sequential decision-making mechanism. Additionally, we also present a practical MARL algorithm that serves as an application instantiation of AGPS.

5.1 Decision Ordering as a Ranking Task

A straightforward approach to optimizing the agent decision order involves evaluating all permutations of n agents within each iteration of the sequential action generation process, with the objective of identifying the specific permutation that can maximize the joint advantage value. While this method exhibits favorable interpretability and can guarantee optimal orderings, the primary limitation stems from its factorial search complexity ($O(n!)$), which significantly increases the computational cost and limits its applicability in large-scale multi-agent systems. Inspired by the football case depicted in Section 4, we propose to leverage the potential correlation between the optimal decision order and the preference scoring of agents' local observations to address this challenge. Specifically, we model the decision-ordering task as a label ranking problem, enabling the application of parametric probabilistic models [7] and deep learning-based optimization techniques.

Learning to rank is a fundamental problem in the domain of machine learning [3, 36], where deep learning-based methods have witnessed widespread applications [28]. By establishing a scoring network to evaluate the preference scores of agents' local observations and ranking them accordingly, an optimized decision sequence can be derived in a computationally efficient manner. However, this approach also exhibits several limitations when implemented in multi-agent systems. Firstly, the output variations of the scoring network do not necessarily convert into adjustments in the final rankings, which can potentially induce the vanishing gradient issue during the neural network training process. Secondly, generating agent decision order in a deterministic manner can introduce instability in that, when the individual scores are similar, minor discrepancies in scoring can produce substantial alterations in the final rankings. To address these challenges, we model the decision-ordering task as a multi-step sampling process and propose a P-L model-based approach that facilitates decision-credit allocation and decision-order optimization in multi-agent sequential decision-making.

5.2 Plackett-Luce Sampling

The Plackett-Luce model derives its name from independent work by Plackett [23] and Luce [17], which has found extensive applications in various real-world tasks like horse-racing [23], document ranking [3] and information retrieval [11], etc. A P-L model is parameterized by an n-length vector $\boldsymbol{v} = (v_1, \ldots, v_n)$ where $v_i > 0$ represents the preference score associated with each object i. The probability of sampling an ordered permutation $\sigma^{(n)} = (\sigma(1), \ldots, \sigma(n))$ from a P-L distribution can be written as

$$\mathrm{P}(\sigma^{(n)} \mid \boldsymbol{v}) = \prod_{i=1}^{n} \frac{v_{\sigma(i)}}{v_{\sigma(i)} + v_{\sigma(i+1)} + \ldots + v_{\sigma(n)}}. \tag{8}$$

The P-L model extends the Bradley-Terry (BT) model suggested by Bradley and Terry [2], which is renowned for its application in the domain of pairwise comparisons, to model item preferences as sampling probabilities. Specifically, the BT model specifies the probability that "$\sigma(i)$ wins against $\sigma(j)$" in terms of

$$\mathrm{P}(\sigma(i) \succ \sigma(j)) = \frac{v_{\sigma(i)}}{v_{\sigma(i)} + v_{\sigma(j)}}, \tag{9}$$

where $\succ$ denotes the asymmetric relation which indicates that $\sigma(i)$ precedes $\sigma(j)$ in ordering. Derived from the BT model, the expected

ordering generated from a P-L sampling process satisfies

$$\begin{aligned} \sigma^{(n)} &= [\sigma(1), \sigma(2), \ldots, \sigma(n)], \\ \text{s.t.} \quad & \forall(\sigma(i), \sigma(j)), i < j \Rightarrow v_i \geq v_j. \end{aligned} \tag{10}$$

P-L sampling provides a probabilistic understanding of preference structures, addressing key challenges such as instability and vanishing gradient associated with the ordering process. Specifically, P-L sampling decomposes the ranking task of n objects as a sequence of $n-1$ independent selection stages, wherein each stage involves choosing the next top-scoring item from the remaining alternatives. This sequential mechanism ensures that objects with similar preference scores are assigned comparable selection probabilities, thereby reducing the sensitivity to minor score fluctuations and enhancing the robustness of ranking. Additionally, this method effectively converts the variations in preference scores output by neural networks into adjustments in the final rankings, thus mitigating the vanishing gradient issue in the neural network training process. Furthermore, P-L sampling has been shown to be computationally efficient [21] and offers various optimization opportunities, indicating potential scalability in large-scale multi-agent systems.

5.3 Action Generation with P-L Sampling

We utilize P-L sampling to optimize the action generation order within an MAS. To handle non-negative constraints, we parameterize the multi-agent P-L distribution using logarithmic parameters $\boldsymbol{z} = (z_1, \ldots, z_n)$. The probability of obtaining the optimal action generation order σ^* can then be derived as

$$P(\sigma^*|\boldsymbol{z}) = \prod_{i=1}^{n-1} \frac{\exp z_{\sigma^*(i)}}{\sum_{j=i}^{n} \exp z_{\sigma^*(j)}}. \tag{11}$$

Utilizing the sequence-related joint advantage value $A_{\boldsymbol{\pi}^\sigma}^{i_{1:n}}$ (abbreviated as $A(\sigma)$) as feedback signal, the parameters $\boldsymbol{z}$ are learned through a scoring network that outputs the preference scores associated with agents' local observations. Specifically, let $\mathbb{S}_N$ denote the space of n-agent permutations, the expectation-form objective function of the agent decision order optimization problem can be formulated as

$$J(\boldsymbol{z}) = \mathbb{E}_{\sigma \in \mathbb{S}_N}\left[A(\sigma)\right] = \sum_{\sigma \in \mathbb{S}_N} A(\sigma) P(\sigma|\boldsymbol{z}). \tag{12}$$

For numerical computation, the gradient of Equation (12) can be estimated via *Monte Carlo approximation* as

$$\nabla_{\boldsymbol{z}} J(\boldsymbol{z}) = \mathbb{E}_{\sigma \in \mathbb{S}_N}\left[A(\sigma) \nabla_{\boldsymbol{z}} \log P(\sigma|\boldsymbol{z})\right] \approx \frac{1}{N} \sum_{i=1}^{N} A(\sigma_i) \nabla_{\boldsymbol{z}} \log P(\sigma_i|\boldsymbol{z}). \tag{13}$$

The i^{th} partial derivative of the log-likelihood $\log P(\sigma|\boldsymbol{z})$ with respect to $z_{\sigma(i)}$ in Equation (13) can be calculated by

$$\frac{\partial \log P(\sigma|\boldsymbol{z})}{\partial z_{\sigma(i)}} = 1 - \exp(z_{\sigma(i)}) \sum_{k=1}^{i} \frac{1}{\sum_{j=k}^{n} \exp(z_{\sigma(j)})}, \tag{14}$$

and the full gradient $\nabla_{\boldsymbol{z}} \log P(\sigma|\boldsymbol{z})$ can be generated within $O(n)$ timesteps [5], thus demonstrating superior efficiency.

We designate the proposed sequential action generation mechanism as **Action Generation with P-L Sampling (AGPS)**. AGPS allows agents whose observations contribute more significantly to the multi-agent joint advantage to be granted higher decision-making priority. In this manner, subsequent agents can effectively perceive the decisions of their predecessors and offer proactive cooperation during their decision-making process, thereby enhancing the overall coordination within a multi-agent team.

5.4 Practical Algorithm

As illustrated in Figure 1, sequential decision-making MARL algorithms can be implemented via single or multiple interactions with the environment. While the latter paradigm benefits from timely feedback, it suffers from degradation in computational efficiency due to numerous interactions within each iteration of multi-agent joint action. Fortunately, recent advancements in auto-regressive SMs have provided fresh insights into the single-interaction paradigm, facilitating efficient sequential decision-making MARL algorithms based on the *Transformer* [31] architecture.

Prioritized Multi-Agent Transformer. We demonstrate that AGPS can be effectively integrated with MAT for performance enhancement. On the one hand, the observation representations output by the encoder of MAT synthesize not only the local observations of individual agents but also the high-level inter-agent relationships [34], thus providing informative signals for preference scoring. On the other hand, the masked self-attention mechanism of MAT's decoder inherently promotes efficient sequential action generation. Hence, via integrating P-L sampling with the encoder-decoder architecture of MAT, an effective bridge between the local observation representations (as outputs of the encoder) and the decision orderings (as inputs of the decoder) can be established, which facilitates auto-regressive sequential action generation and ultimately leads to the proposed **Prioritized Multi-Agent Transformer (PMAT)** as illustrated in Figure 3.

As shown in Figure 3, the scoring block consists of an MLP Φ parameterized by φ, which takes agents' local observation representations $(\hat{o}^{i_1}, \ldots, \hat{o}^{i_n})$ as input and yields the corresponding preference scores $(c_{i_1}, \ldots, c_{i_n})$. Specifically, the scoring network is trained to minimize the following clipping objective of

$$L_{\text{Ranking}}(\varphi) = -\frac{1}{T} \sum_{t=0}^{T-1} \min\left(\mathrm{r}_t^\sigma(\varphi)\hat{A}_t, \mathrm{clip}(\mathrm{r}_t^\sigma(\varphi), 1 \pm \epsilon)\hat{A}_t\right), \tag{15}$$

where

$$\mathrm{r}_t^\sigma(\varphi) = \frac{P(\sigma|\Phi_\varphi(\hat{\mathbf{o}}_t^{i_{1:n}}))}{P(\sigma|\Phi_{\varphi_{old}}(\hat{\mathbf{o}}_t^{i_{1:n}}))}, \tag{16}$$

$\hat{A}_t$ estimates the sequence-related joint advantage, and $P(\sigma|\cdot)$ represents the probability of obtaining the current order $\sigma = (j_1, \ldots, j_n)$ via P-L sampling. Before being passed into the decoder, the original observation representations $(\hat{o}^{i_1}, \ldots, \hat{o}^{i_n})$ are reordered according to this order as $(\hat{o}^{j_1}, \ldots, \hat{o}^{j_n})$. Meanwhile, $\sigma = (j_1, \ldots, j_n)$ also serves as the auto-regressive action generation order within the decoder, thus realizing agent decision order optimization.

Notably, the sampling paradigm differs slightly between the training and inference stages. During the training stage, P-L sampling is carried out in a non-deterministic manner since introducing randomness can facilitate generalization capability and mitigate the risk of overfitting specific training instances. During the inference

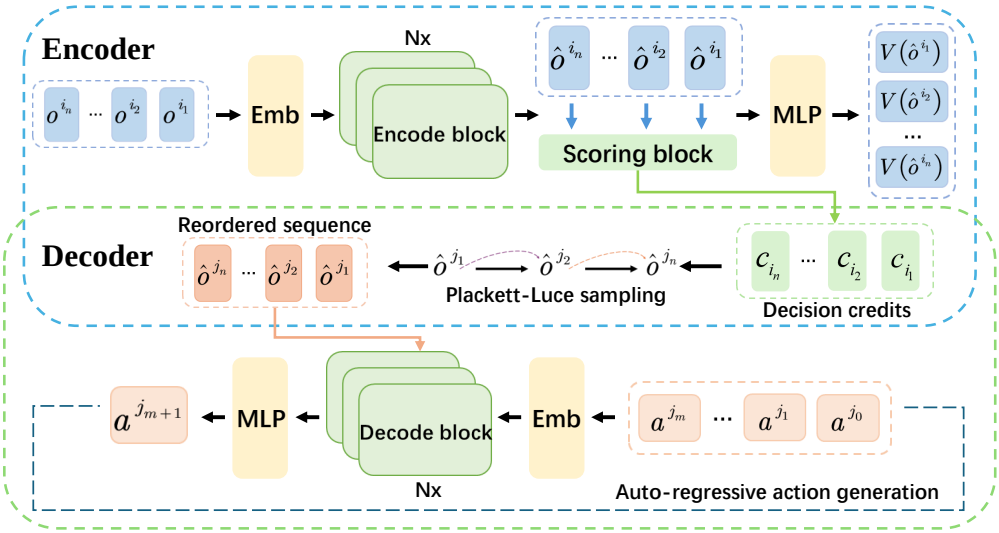


Figure 3: The overall framework of the proposed *Prioritized Multi-Agent Transformer*. The encoder processes agents' local observations at each time step, transforming them into high-level representations. The observation representations are subsequently fed into the scoring block to generate individual preference scores, referred to as decision credits. P-L sampling is then conducted based on the scoring to compute the action generation order. The representations are reordered prior to being fed into the decoder which sequentially generates agents' actions in accordance with this reordered sequence.

stage, however, deterministic sampling is carried out for performance enhancement. Besides, in the training stage, the output of all actions $\boldsymbol{a}^{i_{1:n}}$ can be computed with parallel acceleration in the sense that $\boldsymbol{a}^{i_{1:n-1}}$ have already been collected and stored in the replay buffer. In contrast, during the inference stage, each action a^{i_m} has to be inserted back into the decoder auto-regressively to generate the following action $a^{i_{m+1}}$ in a sequential manner.

6 EXPERIMENTS

In this section, we evaluate the effectiveness of the proposed AGPS and its application instantiation, PMAT, within popular MARL benchmarks. We compare PMAT with advanced MARL methods including MAT [34], MAPPO [37], and HAPPO [14].

6.1 Experimental Environments

In this work, we evaluate our method within the following three MARL benchmarks: StarCraft Multi-Agent Challenge (referred to as SMAC) [24], Google Research Football (referred to as GRF) [15] and Multi-Agent MuJoCo (referred to as MA MuJoCo) [9].

StarCraft Multi-Agent Challenge. SMAC [24] is an open-source research environment designed to evaluate MARL algorithms based on the StarCraft II game engine. SMAC simulates complex scenarios and varying unit types with real-time multi-agent interactions, enabling comprehensive benchmarking of cooperation strategies. Specifically, we conduct comparison experiments on two challenging maps, *10m vs 11m* (*Hard*, *homogeneous* and *asymmetric*) and *MMM2* (*Super Hard*, *heterogeneous* and *asymmetric*).

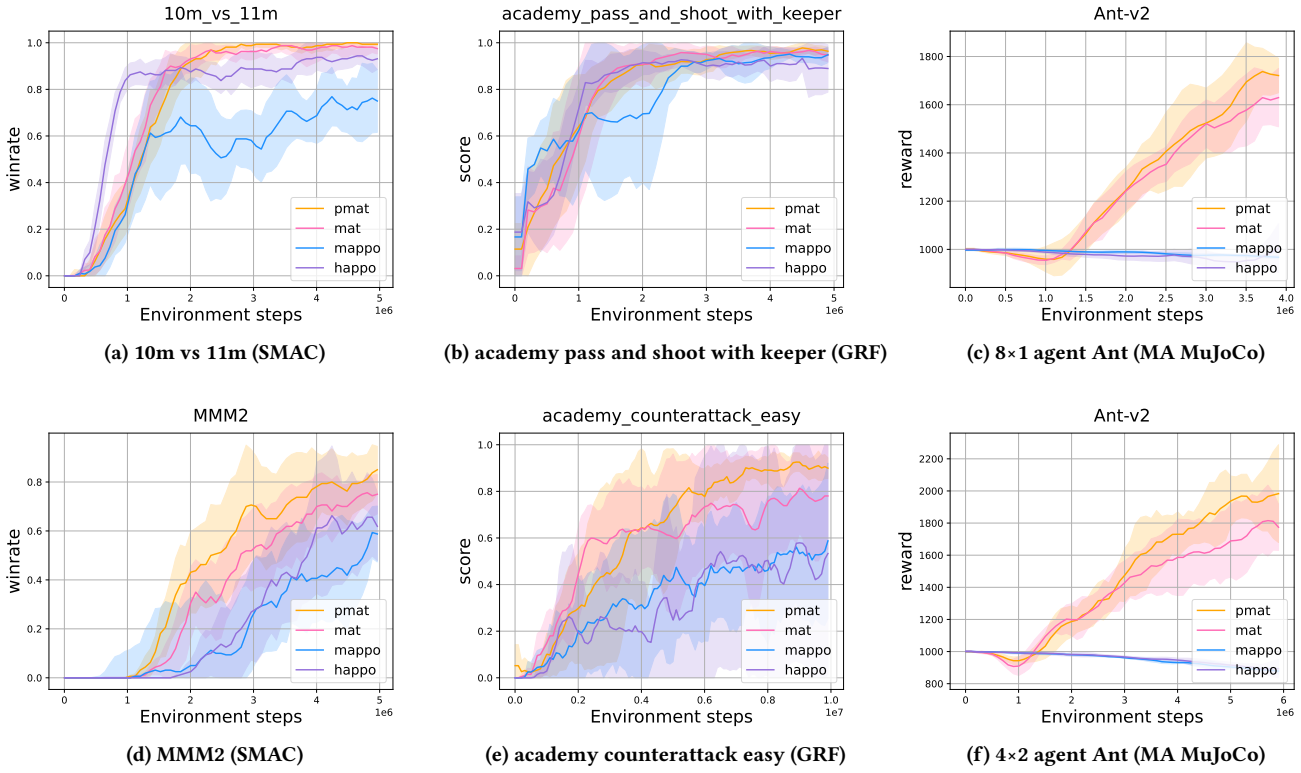
Google Research Football. GRF [15] is an open-source research environment designed for MARL algorithm evaluation in a simulated football setting. In GRF, agents play different roles within a football team (e.g., forwards, wingers, etc.), demonstrating evident role heterogeneity. We utilize the *academy pass and shoot with keeper*, *academy counterattack easy*, and *academy 3 vs 1 with keeper* scenarios for algorithm evaluation.

Multi-Agent MuJoCo. MA MuJoCo [9] contains a variety of multi-agent continuous control tasks where individual agents control the joints of biomimetic robot entities and coordinate to facilitate specific behavior. Built on the MuJoCo physics engine [30], MA MuJoCo's modular architecture allows for easy customization of the environments and agents' behavior, facilitating the simulation of complex interactions among agents trained by MARL algorithms. We evaluate the proposed method using the *Ant-v2* scenario with two different configurations: *8×1 agent Ant* and *4×2 agent Ant*.

6.2 Experimental Results

StarCraft Multi-Agent Challenge. The comparison results of all methods in the SMAC domain are shown in Figure 4a and Figure 4d. Specifically, for the *10m vs 11m* map, PMAT achieves a winrate of 99.4%, outperforming the baseline methods MAT (97.5%), MAPPO (75.0%), and HAPPO (93.1%). For the *MMM2* map, PMAT achieves a winrate of 85.0%, consistently surpassing the baseline methods MAT (75.0%), MAPPO (58.8%), and HAPPO (61.9%). The experimental results indicate that the performance enhancement of PMAT over MAT tends to be more pronounced in heterogeneous scenarios (e.g., *MMM2*) than in homogeneous scenarios (e.g., *10m vs 11m*). This observation broadly corroborates our hypothesis in Section 1. Specifically, as the heterogeneity intensifies the sequential dependencies among agents within such tasks, effective management of these dependencies significantly improves the coordination efficiency. In addition, given that both of the selected maps contain ten ally agents, the experimental results in the SMAC domain can also validate the scalability of the proposed method.

Google Research Football. The comparison results of all methods in the GRF domain are illustrated in Figure 4b and Figure 4e. PMAT



(a) 10m vs 11m (SMAC) (b) academy pass and shoot with keeper (GRF) (c) 8×1 agent Ant (MA MuJoCo)

(d) MMM2 (SMAC) (e) academy counterattack easy (GRF) (f) 4×2 agent Ant (MA MuJoCo)

Figure 4: Experimental results in StarCraft Multi-Agent Challenge, Google Research Football, and Multi-Agent MuJoCo.

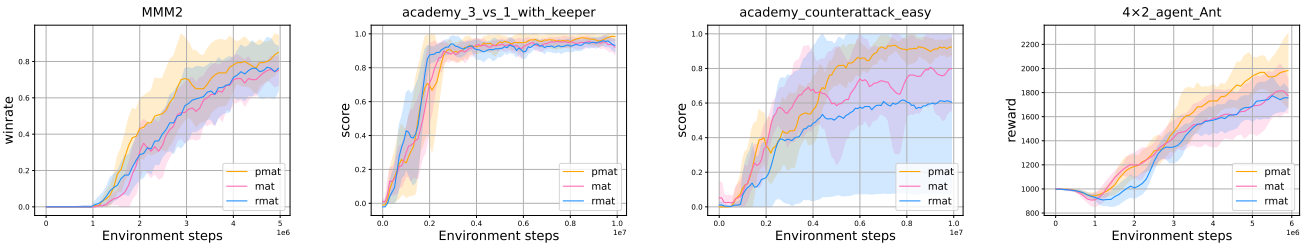
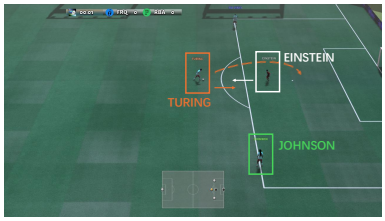


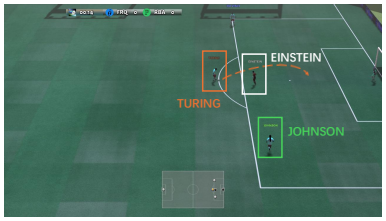
Figure 5: Ablation results in StarCraft Multi-Agent Challenge, Google Research Football, and Multi-Agent MuJoCo.

demonstrates 0.964 in average episode scores in the *academy pass and shoot with keeper* scenario, which outperforms MAT (0.947), MAPPO (0.948), and HAPPO (0.890). In the *academy counterattack easy* scenario, PMAT demonstrates 0.899 in average episode scores, consistently surpassing MAT (0.780), MAPPO (0.588), and HAPPO (0.533). As illustrated in Figure 4, while introducing the scoring network elevates the training cost and marginally degrades its performance in the early stages, PMAT ultimately surpasses MAT with the advancement of training. The experimental results in the GRF domain demonstrate that PMAT achieves superior performance in cooperation tasks that exhibit evident role heterogeneity. Moreover, the performance curve of MAT exhibits more pronounced fluctuations than PMAT in some scenarios (e.g., *academy counterattack easy*), further validating the efficacy of the proposed AGPS mechanism in enhancing the stability of MARL algorithms.

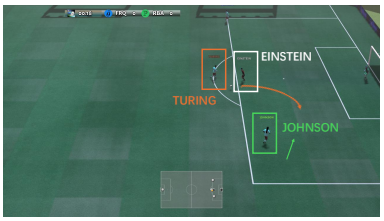
Multi-Agent MuJoCo. The comparison results of all methods in the MA MuJoCo domain are presented in Figure 4c and Figure 4f. It can be observed that, equipped with AGPS, PMAT consistently outperforms the baselines in both the *8×1 agent Ant* and the *4×2 agent Ant* scenarios. As illustrated in Figure 4, while PMAT initially exhibits comparable task performance to MAT, it progressively outperforms MAT as training proceeds, which demonstrates the effectiveness of the proposed method. Additionally, it can be observed that in both of the *Ant-v2* scenarios, the sequential decision-making MARL algorithms like MAT and PMAT exhibit distinct advantages in task performance over the simultaneous decision-making MARL algorithms like MAPPO and HAPPO. We attribute this phenomenon to the lack of essential information regarding previous agents' decisions, which exacerbates coordination challenges as the torques



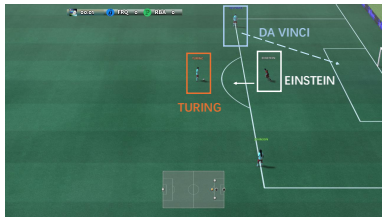
(a) TURING attempts to bypass EINSTEIN.



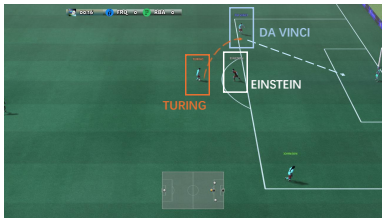
(b) EINSTEIN mounts a frontal defense.



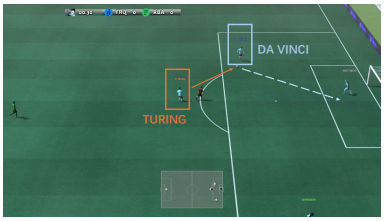
(c) TURING opts for a pass to JOHNSON.



(d) TURING retains possession and waits.



(e) TURING passes the ball to DA VINCI.



(f) DA VINCI gets the ball and shoots.

Figure 6: Different cooperation strategies of agents trained by MAT and PMAT in a Google Research Football setting. Figures 6a to 6c illustrate the behavior of agents trained by MAT. Figures 6d to 6f illustrate the behavior of agents trained by PMAT. The solid lines represent the current action, whereas the dashed lines denote the predicted intention of the following action.

applied by different agents may counteract each other, thus leading to degradation in overall task performance.

6.3 Ablation Study

In this section, we present an ablation study to further evaluate the advantage-based scoring mechanism adopted in AGPS across the SMAC, GRF, and MA MuJoCo benchmarks. Specifically, in addition to MAT, we introduce another baseline, namely the randomized MAT (abbreviated as rMAT in this section), which adopts a fully randomized ordering strategy to determine the action generation order based on MAT. We summarize the experimental results in Figure 5, where PMAT exhibits superior performance compared with both MAT and rMAT in all displayed tasks. It can be observed that the randomized ordering strategy introduces instability since rMAT exhibits pronounced variance in some scenarios (e.g., *academy counterattack easy*), resulting in performance degradation. In contrast, the integration of AGPS effectively enhances both stability and monotonicity in task performance improvement, validating the effectiveness of the advantage-based scoring mechanism.

6.4 Case Study

In this section, we aim to analyze the distinct coordination strategies of agents trained by MAT and PMAT. Specifically, we conduct a fine-grained case study on agents' coordination behavior in the *academy 3 vs 1 with keeper* scenario of GRF, as illustrated in Figure 6. It can be observed that given identical initial conditions (our player TURING holds the ball, directly facing the defense of the opponent player EINSTEIN), agents trained by MAT adopt a strategy wherein TURING attempts to dribble past EINSTEIN and shoot (Figures 6a to 6c). In contrast, agents trained by PMAT adopt a different strategy (Figures 6d to 6f), wherein TURING directly awaits an opportune moment to pass the ball to his teammate DA VINCI, who enjoys a superior shooting angle unimpeded by direct opposition. Compared with the former strategy wherein agents make decisions more based on their own observations, the latter strategy evaluates the significance of local observations within a multi-agent team and allows agents who hold advantageous observations to make decisions first (DA VINCI, shoot), followed by proactive coordination of other agents (TURING, pass the ball to DA VINCI). Generally, the comparison between the aforementioned coordination strategies in this section offers an intuitive validation for the necessity of agent decision order optimization in MARL, demonstrating its effectiveness in promoting cooperative behavior among agents.

7 CONCLUSION

In this work, we propose **Action Generation with Plackett-Luce Sampling (AGPS)**, a sequential decision-making mechanism in MARL. AGPS assigns decision credits to individual agents within a multi-agent team and significantly facilitates joint policy improvement by providing finer-grained supervision for decision order optimization. Integrating AGPS with the Multi-Agent Transformer, we propose the **Prioritized Multi-Agent Transformer (PMAT)**, a sequential decision-making MARL algorithm with optimized decision-ordering. Extensive experiments across various benchmarks showcase the effectiveness of AGPS as well as the superiority of PMAT in learning efficiency and task performance over several strong baselines. For future work, we plan to further investigate the effectiveness of AGPS by integrating it with a broader context of MARL algorithms. Besides, we will also evaluate the applicability of P-L sampling in larger-scale multi-agent systems.

ACKNOWLEDGMENTS

This work is supported by the National Natural Science Foundation of China (62106278, 91948303-1, 611803375, 12002380, 62101575) and the National Key R&D Program of China (2021ZD0140301). Team from Shanghai Jiao Tong University is supported by the National Key R&D Program of China (2022ZD0114804). Muning Wen and Xihuai Wang are supported by the Wen-Tsun Wu AI Honorary Doctoral Scholarship from AI Institute, Shanghai Jiao Tong University.

REFERENCES

[1] Dimitri Bertsekas. 2021. Multiagent reinforcement learning: Rollout and policy iteration. *IEEE/CAA Journal of Automatica Sinica* 8, 2 (2021), 249–272.

[2] Ralph Allan Bradley and Milton E Terry. 1952. Rank analysis of incomplete block designs: I. The method of paired comparisons. *Biometrika* 39, 3/4 (1952), 324–345.

[3] Zhe Cao, Tao Qin, Tie-Yan Liu, Ming-Feng Tsai, and Hang Li. 2007. Learning to rank: from pairwise approach to listwise approach. In *Proceedings of the 24th international conference on Machine learning*. 129–136.

[4] Cristiano Castelfranchi, Maria Miceli, and Amedeo Cesta. 1992. Dependence relations among autonomous agents. *Decentralized AI* 3 (1992), 215–227.

[5] Josu Ceberio and Valentino Santucci. 2023. Model-based gradient search for permutation problems. *ACM Transactions on Evolutionary Learning and Optimization* 3, 4 (2023), 1–35.

[6] Lili Chen, Kevin Lu, Aravind Rajeswaran, Kimin Lee, Aditya Grover, Misha Laskin, Pieter Abbeel, Aravind Srinivas, and Igor Mordatch. 2021. Decision transformer: Reinforcement learning via sequence modeling. *Advances in neural information processing systems* 34 (2021), 15084–15097.

[7] Weiwei Cheng, Eyke Hüllermeier, and Krzysztof J Dembczynski. 2010. Label ranking methods based on the Plackett-Luce model. In *Proceedings of the 27th International Conference on Machine Learning (ICML-10)*. 215–222.

[8] William W Cohen, Robert E Schapire, and Yoram Singer. 1997. Learning to order things. *Advances in neural information processing systems* 10 (1997).

[9] Christian Schroeder de Witt, Bei Peng, Pierre-Alexandre Kamienny, Philip Torr, Wendelin Böhmer, and Shimon Whiteson. 2020. Deep multi-agent reinforcement learning for decentralized continuous cooperative control. *arXiv preprint arXiv:2003.06709* 19 (2020).

[10] Jakob Foerster, Gregory Farquhar, Triantafyllos Afouras, Nantas Nardelli, and Shimon Whiteson. 2018. Counterfactual multi-agent policy gradients. In *Proceedings of the AAAI conference on artificial intelligence*, Vol. 32.

[11] John Guiver and Edward Snelson. 2009. Bayesian inference for Plackett-Luce ranking models. In *proceedings of the 26th annual international conference on machine learning*. 377–384.

[12] Mahdi Hannoun, Jaime Simao Sichman, Olivier Boissier, and Claudette Sayettat. 1998. Dependence relations between roles in a multi-agent system: Towards the detection of inconsistencies in organization. In *International Workshop on Multi-Agent Systems and Agent-Based Simulation*. Springer, 169–182.

[13] Michael Janner, Qiyang Li, and Sergey Levine. 2021. Offline reinforcement learning as one big sequence modeling problem. *Advances in neural information processing systems* 34 (2021), 1273–1286.

[14] JG Kuba, R Chen, M Wen, Y Wen, F Sun, J Wang, and Y Yang. 2022. Trust Region Policy Optimisation in Multi-Agent Reinforcement Learning. In *ICLR 2022-10th International Conference on Learning Representations*. The International Conference on Learning Representations (ICLR), 1046.

[15] Karol Kurach, Anton Raichuk, Piotr Stańczyk, Michał Zając, Olivier Bachem, Lasse Espeholt, Carlos Riquelme, Damien Vincent, Marcin Michalski, Olivier Bousquet, et al. 2020. Google research football: A novel reinforcement learning environment. In *Proceedings of the AAAI conference on artificial intelligence*, Vol. 34. 4501–4510.

[16] Michael L Littman. 1994. Markov games as a framework for multi-agent reinforcement learning. In *Machine learning proceedings 1994*. Elsevier, 157–163.

[17] R Duncan Luce. 1959. *Individual choice behavior*. Vol. 4. Wiley New York.

[18] Linghui Meng, Muning Wen, Chenyang Le, Xiyun Li, Dengpeng Xing, Weinan Zhang, Ying Wen, Haifeng Zhang, Jun Wang, Yaodong Yang, et al. 2023. Offline pre-trained multi-agent decision transformer. *Machine Intelligence Research* 20, 2 (2023), 233–248.

[19] Frans A Oliehoek, Christopher Amato, et al. 2016. *A concise introduction to decentralized POMDPs*. Vol. 1. Springer.

[20] Frans A Oliehoek, Matthijs TJ Spaan, and Nikos Vlassis. 2008. Optimal and approximate Q-value functions for decentralized POMDPs. *Journal of Artificial Intelligence Research* 32 (2008), 289–353.

[21] Harrie Oosterhuis. 2021. Computationally efficient optimization of plackett-luce ranking models for relevance and fairness. In *Proceedings of the 44th International ACM SIGIR Conference on Research and Development in Information Retrieval*. 1023–1032.

[22] Liang Pang, Yanyan Lan, Jiafeng Guo, Jun Xu, Jingfang Xu, and Xueqi Cheng. 2017. Deeprank: A new deep architecture for relevance ranking in information retrieval. In *Proceedings of the 2017 ACM on Conference on Information and Knowledge Management*. 257–266.

[23] Robin L Plackett. 1975. The analysis of permutations. *Journal of the Royal Statistical Society Series C: Applied Statistics* 24, 2 (1975), 193–202.

[24] Mikayel Samvelyan, Tabish Rashid, Christian Schroeder De Witt, Gregory Farquhar, Nantas Nardelli, Tim GJ Rudner, Chia-Man Hung, Philip HS Torr, Jakob Foerster, and Shimon Whiteson. 2019. The starcraft multi-agent challenge. *arXiv preprint arXiv:1902.04043* (2019).

[25] John Schulman, Philipp Moritz, Sergey Levine, Michael Jordan, and Pieter Abbeel. 2015. High-dimensional continuous control using generalized advantage estimation. *arXiv preprint arXiv:1506.02438* (2015).

[26] John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. 2017. Proximal policy optimization algorithms. *arXiv preprint arXiv:1707.06347* (2017).

[27] Michael Schumacher. 2001. Multi-agent systems. *Objective Coordination in Multi-Agent System Engineering: Design and Implementation* (2001), 9–32.

[28] Aliaksei Severyn and Alessandro Moschitti. 2015. Learning to rank short text pairs with convolutional deep neural networks. In *Proceedings of the 38th international ACM SIGIR conference on research and development in information retrieval*. 373–382.

[29] Shai Shalev-Shwartz and Shai Ben-David. 2014. *Understanding machine learning: From theory to algorithms*. Cambridge university press.

[30] Emanuel Todorov, Tom Erez, and Yuval Tassa. 2012. Mujoco: A physics engine for model-based control. In *2012 IEEE/RSJ international conference on intelligent robots and systems*. IEEE, 5026–5033.

[31] A Vaswani. 2017. Attention is all you need. *Advances in Neural Information Processing Systems* (2017).

[32] Xihuai Wang, Zheng Tian, Ziyu Wan, Ying Wen, Jun Wang, and Weinan Zhang. 2023. Order matters: Agent-by-agent policy optimization. *arXiv preprint arXiv:2302.06205* (2023).

[33] Xihuai Wang, Zhicheng Zhang, and Weinan Zhang. 2022. Model-based multi-agent reinforcement learning: Recent progress and prospects. *arXiv preprint arXiv:2203.10603* (2022).

[34] Muning Wen, Jakub Kuba, Runji Lin, Weinan Zhang, Ying Wen, Jun Wang, and Yaodong Yang. 2022. Multi-agent reinforcement learning is a sequence modeling problem. *Advances in Neural Information Processing Systems* 35 (2022), 16509–16521.

[35] Muning Wen, Runji Lin, Hanjing Wang, Yaodong Yang, Ying Wen, Luo Mai, Jun Wang, Haifeng Zhang, and Weinan Zhang. 2023. Large sequence models for sequential decision-making: a survey. *Frontiers of Computer Science* 17, 6 (2023), 176349.

[36] Fen Xia, Tie-Yan Liu, Jue Wang, Wensheng Zhang, and Hang Li. 2008. Listwise approach to learning to rank: theory and algorithm. In *Proceedings of the 25th international conference on Machine learning*. 1192–1199.

[37] Chao Yu, Akash Velu, Eugene Vinitsky, Jiaxuan Gao, Yu Wang, Alexandre Bayen, and Yi Wu. 2022. The surprising effectiveness of ppo in cooperative multi-agent games. *Advances in Neural Information Processing Systems* 35 (2022), 24611–24624.

[38] Kaiqing Zhang, Zhuoran Yang, and Tamer Başar. 2021. Multi-agent reinforcement learning: A selective overview of theories and algorithms. *Handbook of reinforcement learning and control* (2021), 321–384.

[39] Weinan Zhang, Xihuai Wang, Jian Shen, and Ming Zhou. 2021. Model-based multi-agent policy optimization with adaptive opponent-wise rollouts. *arXiv preprint arXiv:2105.03363* (2021).

[40] Wenjing Zhang, Wei Zhang, Wenqing Hu, and Yifan Wang. 2024. B2MAPO: A Batch-by-Batch Multi-Agent Policy Optimization to Balance Performance and Efficiency. *arXiv preprint arXiv:2407.15077* (2024).