

Towards Efficient Collaboration via Graph Modeling in Reinforcement Learning

Wenzhe Fan¹ †, Zishun Yu¹, Chengdong Ma², Changye Li³, Yaodong Yang², Xinhua Zhang¹

¹ University of Illinois Chicago

² Institute for Artificial Intelligence, Peking University

³ Yuanpei College, Peking University

wfan23@uic.edu, zyu32@uic.edu, chengdong.ma@stu.pku.edu.cn, antoine031106@gmail.com,
yaodong.yang@pku.edu.cn, zhangx@uic.edu

Abstract

In multi-agent reinforcement learning, a commonly considered paradigm is centralized training with decentralized execution. However, in this framework, decentralized execution restricts the development of coordinated policies due to the local observation limitation. In this paper, we consider the cooperation among *neighboring* agents during execution and formulate their interactions as a graph. Thus, we introduce a novel encoder-decoder architecture named Factor-based Multi-Agent Transformer (*f*-MAT) that utilizes a transformer to enable communication between neighboring agents during both training and execution. By dividing agents into different overlapping groups and representing each group with a *factor*, *f*-MAT achieves efficient message passing and *parallel* action generation through factor-based attention layers. Empirical results in networked systems such as traffic scheduling and power control demonstrate that *f*-MAT achieves superior performance compared to strong baselines, thereby paving the way for handling complex collaborative problems.

1 Introduction

The intricate nature of collaboration and the demand for time efficiency render multi-agent reinforcement learning (MARL) a challenging problem. First, the joint action space grows exponentially with the number of agents, resulting in a complex scenario for making cooperative decisions. Second, it requires effective information exchange throughout the system to help agents learn the state of the environment and other agents. For example, traffic light control at multiple intersections needs a coordination mechanism that allows each signal to act based on traffic conditions not only at its own intersection but also at nearby neighbors, even at distant signals. Therefore, developing an efficient collaboration approach is crucial for decision-making in multi-agent systems.

A commonly considered paradigm in MARL is centralized training with decentralized execution (CTDE) (Sunehag et al. 2017; Rashid et al. 2020; Son et al. 2019; Foerster et al. 2018; Lowe et al. 2017; Yu et al. 2022), where each agent acts independently according to its own observation.

However, these methods only focus on stabilizing training with advanced value estimation and do not finely model

the relationship among agents during execution. As a result, some of them may fail in the simplest cooperative tasks (Kuba et al. 2022). Our method retains the actor-critic architecture in the CTDE framework but extends independent action to neighborhood-based action, capturing the interactions between agents during execution.

There are literatures (Boehmer, Kurin, and Whiteson 2020; Li et al. 2020) modeling the relations among the agents in multi-agent systems with graph, in which each agent is represented by a node and the agent interaction is represented by an edge. To facilitate communication between agents, Foerster et al. (2016) and Sukhbaatar, Szlam, and Fergus (2016) utilize the averaged encoded hidden states of other agents, while Zhang et al. (2018) and Zhang, Yang, and Başar (2021) seek consensus to achieve the optimal common reward. Das et al. (2019) and Jiang and Lu (2018) implemented attention mechanisms to determine the optimal time and target for communication. Further research employs graph neural networks (GNNs) and graph attention networks (GANs) to enhance agent interactions (Jiang et al. 2018; Hoshen 2017; Das et al. 2019; Singh, Jain, and Sukhbaatar 2019; Niu, Paleja, and Gombolay 2021; Kim et al. 2019). Additionally, MARL methods capitalize on networked topologies (Chu, Chinchali, and Katti 2020; Zhang et al. 2018; Gupta, Hazra, and Dukkupati 2020; Guestrin, Lagoudakis, and Parr 2002; Zhang, Aberdeen, and Vishwanathan 2007). Unfortunately, many of these methods suffer from the time complexity of $O(n^2)$ for n number of agents (Hao et al. 2023). Moreover, agent-level communication limits the efficiency of learning cooperative policies.

To address these challenges, we propose Factor-based Multi-Agent Transformer (*f*-MAT), which enables efficient collaboration during *both* training *and* execution through all agents via graph modeling within the CTDE framework. First, to enable flexible communication, we propose a hypernode called *factor*. By modeling the collaboration structure as a graph, we organize agents into different groups and represent each of them as a factor. Serving as the intermediary, each factor can include multiple agents and each agent can belong to multiple factors. Therefore, we allow an agent’s observation and action to propagate and to influence other agents, while also facilitating communication at the group level.

Second, to capture the interactions between neighboring agents, we utilize the transformer model, which shifts the

search in the joint action space from a multiplicative to an additive size (Wen et al. 2022), effectively addressing the problem of exponential growth of the action space. Furthermore, the mask mechanism ensures that messages are only passed between factors and their constituent agents.

Third, to address the $O(n^2)$ time complexity in GAN-based methods, we propose a factor-based attention that reduces the complexity to $O(m \cdot S_f \cdot L)$, where m is the number of factors, S_f is the maximum size of the factors (number of agents in it), and L is the number of layers. Generally, this leads to significant savings as our experiments show.

Finally, to further reduce the computation cost, we propose a **parallel decoding** in transformer inference, which significantly improves the efficiency upon the conventional autoregressive decoding. This is particularly useful for MARL because the agents do not generally employ an order.

We evaluate the performance and efficiency of f -MAT in grid alignment, traffic scheduling, and power control. Empirical results demonstrate that f -MAT fulfills the efficient collaboration compared to other baselines, paving the way for efficient collaboration in multi-agent systems.

2 Related Work

Cooperative MARL presents a challenging problem as it is difficult for each agent to deduce its individual contribution to the global reward while cooperating with other agents.

A substantial amount of research has focused on the CTDE framework. VDN (Sunehag et al. 2017) directly factorizes the joint action-value function into the summation of independent Q-value functions. QMIX (Rashid et al. 2020) enforces the summation of action-value functions to a monotonic function. QTRAN (Son et al. 2019) generalizes factorization by learning a state-value function, dispensing with the additivity and monotonicity assumptions. COMA (Foerster et al. 2018) introduces a counterfactual baseline in the advantage function and effectively isolates each agent’s contribution. MADDPG (Lowe et al. 2017) has access not only to the actions and observations of its corresponding agent but also to all other agents in the environment. MAPPO (Yu et al. 2022) is the first to apply PPO (Schulman et al. 2017) to the multi-agent setting with parameter sharing. These methods focus on learning a centralized critic, downplaying the interactions among agents, especially during execution. Recently, MAT (Wen et al. 2022) approaches MARL in a fully centralized fashion. However, in practice, centralized execution is not feasible or is overly expensive in computation and communication.

To better model the relationship between agents, graphs have been commonly leveraged. DCG (Boehmer, Kurin, and Whiteson 2020) considers a pre-specified coordination graph to enable message passing between agents and their neighbors. DICG (Li et al. 2020) improves this approach by inferring the dynamic coordination graph structure which is subsequently used by a GNN. HAMA (Ryu, Shin, and Park 2020) and MAGIC (Niu, Paleja, and Gombolay 2021) utilize GANs to deal with communication between agents. Some of them suffer from the time complexity of $O(n^2)$ problem for n agents, especially in a dense graph or attention-based structure. Nonetheless, the graph modeling approach greatly

inspires us to leverage it for capturing more effective relationships among agents.

Further extensions allow agents to exchange messages during execution. DIAL (Foerster et al. 2016) enables discrete communication via the limited-bandwidth channel. CommNet (Sukhbaatar, Szlam, and Fergus 2016) extends to a continuous communication channel. TarMAC (Das et al. 2019) achieves targeted communication with a signature-based soft attention mechanism. ATOC (Jiang and Lu 2018) employs an attention mechanism to decide whether an agent should communicate in its observable field. NeurComm (Gupta, Hazra, and Dukkupati 2020) proposes a neural communication protocol for networked system control. However, these methods focus primarily on communication between individual agents and overlook communications at the group level, limiting their effectiveness in managing large-scale distributed systems. Yet, using attention mechanism to control when and with whom to communicate inspired us to employ transformers to ensure that information flows only to relevant agents.

In this paper, we begin with CTDE framework and explore along graph modeling perspective, aiming to find an efficient message-passing mechanism using the transformer model.

3 Preliminaries

We follow Littman (1994) to model cooperative MARL as a Markov game $\langle \mathcal{N}, \mathcal{O}, \mathcal{A}, P, R, \gamma \rangle$. $\mathcal{N} = \{1, 2, 3, \dots, n\}$ is the set of agents. $\mathcal{O} = \prod_{i=1}^n \mathcal{O}^i$ is the product of local observation spaces of the agents, namely, the agent observation space. $\mathcal{A} = \prod_{i=1}^n \mathcal{A}^i$ is the product of the agents’ action spaces, i.e., the joint action space. $P : \mathcal{O} \times \mathcal{A} \times \mathcal{O} \rightarrow [0, 1]$ is the transition probability function. $R : \mathcal{O} \times \mathcal{A} \rightarrow \mathbb{R}$ is the joint reward function. $\gamma \in [0, 1)$ is the discount factor. Let i_1, \dots, i_n be a random permutation of $1, \dots, n$, and we abbreviate $i_{t:s} := \{i_t, i_{t+1}, \dots, i_s\}$ if $t \leq s$, and \emptyset otherwise.

3.1 Multi-Agent Transformer

Multi-Agent Transformer (MAT, Wen et al. 2022) casts cooperative MARL as a sequential modeling problem wherein one maps a sequence of observations to a sequence of actions, through the multi-agent advantage decomposition theorem (Kuba et al. 2021). This decomposition reveals the insight that the joint advantage $A_\pi^{i_1:n}$ can be decomposed into the sum of individual ones, allowing one to reduce the search space of multiplicative size $\prod_{i=1}^n |\mathcal{A}_i|$ to additive size $\sum_{i=1}^n |\mathcal{A}_i|$. In addition, the definition of individual advantage function $A_\pi^{i_m}(o, a^{i_1:m-1}, a^{i_m})$ naturally reveals a causal sequential structure, where a^{i_m} depends on the set of preceding actions $a^{i_1:m-1}$ (and the joint observation). MAT leverages this decomposed causal structure, using encoder-decoder transformers with causal masked self-attention, by the following encoder and decoder training:

$$L_{\text{enc}}(\phi) = \frac{1}{Tn} \sum_{i=1}^n \sum_{t=0}^{T-1} \left[R(o_t, \mathbf{a}_t) + \gamma V_\phi(\hat{o}_{t+1}^i) - V_\phi(\hat{o}_t^i) \right]$$

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

$$r_t^{i_m}(\theta) = \pi_\theta^{i_m}(a_t^{i_m} | \hat{o}_t^{i_1:n}, \hat{\mathbf{a}}_t^{i_1:m-1}) / \pi_{\theta_{\text{old}}}^{i_m}(a_t^{i_m} | \hat{o}_t^{i_1:n}, \hat{\mathbf{a}}_t^{i_1:m-1})$$

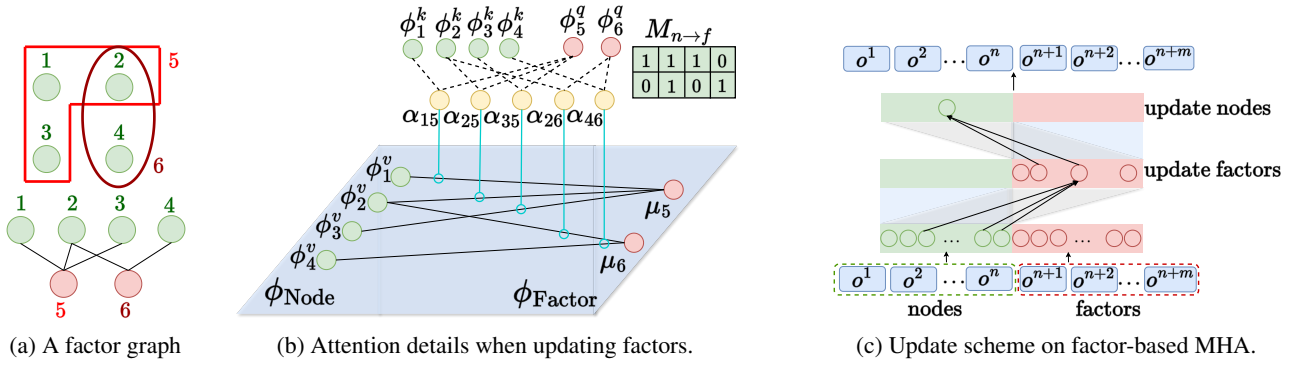


Figure 1: Factor-based attention layer. **Green** represents nodes, **Red** represents factors. (a) Factor graph: divide all nodes (1, 2, 3, 4) into two overlapping groups (1, 2, 3) and (2, 4); define two hypernodes, factor 5 and factor 6 to represent each group; transform a general graph to a bipartite graph. (b) Attention details when updating factors: to update factor observation o_5, o_6 , we set o_5, o_6 as query, $o_1 \dots o_4$ as key and value. Query o_5^q only take attention to related agents' observations o_1, o_2, o_3 . Similar operation to factor observation o_6 . \tilde{o}_5 and \tilde{o}_6 are updated factors. (c) Update scheme on factor-based MHA: It is a two-way message passing, which first updates factors and keep nodes unchanged, then update nodes and keep factors unchanged.

The policy of agent i_m is $\pi_{\theta}^{i_m}(a_t^{i_m} | \hat{o}_t^{i_1:n}, \hat{a}_t^{i_1:m-1})$, requiring observation representations of all agents $\hat{o}_t^{i_1:n}$ and all preceding agents' actions $\hat{a}_t^{i_1:m-1}$. So the execution is centralized. In practice, decision-making may not require complete information from the system. Often, only observations from nearby or related neighbors are relevant. Pulling information from all agents can introduce redundant details, wasting computation and communication. That said, MAT inspires a transformer-based structure in our message-passing mechanism.

4 Factor-based Multi-Agent Transformer

The goal of f -MAT is to address the challenge of multi-agent collaboration in execution for centralized training decentralized execution (CTDE) algorithms, aiming to generate more cooperative policies. In this paper, we focus on exploring a message-passing mechanism from a graph modeling perspective, seeking to enhance cooperation through a more efficient and expansive communication approach.

4.1 Factor Representation of Coordination Graph

To enable the broader message passing, we divide agents into different groups and represent each group with a virtual hypernode named *factor*, thereby using factors as the intermediary to fulfill the group-level communication. Pooling several agents into one group instills the prior that they tend to influence each other. For example, cooperation is particularly necessary for them to achieve optimal actions, or an agent's optimal policy should draw on a factor-mate's inputs (not necessarily the raw observations), or agents in this factor collectively define some situations that impact other agents. We will pass low-cost messages to agents within the same factor to share information. Therefore, a larger factor promotes collaboration between more agents. Since an agent can belong to multiple factors, larger factors also allow information such as actions and observations to be propagated more efficiently to other agents.

We use the toy example in Fig. 1a to illustrate our idea, and it can be easily extended to general networks. Factors can be defined flexibly, accounting for multiple inductive biases and practical constraints. For example, it can be any group of agents such as (1, 2), (2, 3), (3, 4) or (1, 2, 3, 4). Here, we divide the graph into two groups: (1, 2, 3) and (2, 4), represented by factors 5 and 6, respectively.

Following the standard practice in the graphical model literature (Bishop 2006), we characterize the agent-factor membership with a bipartite graph $\mathcal{G} = \langle \mathcal{N}, \mathcal{F} \rangle$, where $\mathcal{F} = \{f_j : j = 1, \dots, m\}$ is a set of factors. An undirected edge is placed between a node $i \in \mathcal{N}$ and a factor $f \in \mathcal{F}$ if and only if i is a member of f , denoted as $i \in f$. Denote the set of edges as \mathcal{E} . Fig. 1a shows the resulting factor graph.

The factor representation simplifies the complex collaboration among agents on a general graph into a group-level message passing framework represented by a bipartite graph, which can be easily utilized by the transformer architecture. In the next section, we will overcome the $O(n^2)$ complexity of transformer and explore an efficient message passing mechanism across the entire graph via factors.

4.2 Factor-based Attention Layer

A crucial property of f -MAT is passing messages between agents and factors. We propose factor-based attention layer to fulfill the efficient message passing by combining factor and multi-head attention layers (MHA) in transformer through proper masking.

Assume we have n nodes, m factors, the input sequence matrix is $O \in \mathbb{R}^{(n+m) \times D}$, and D is the embedding dimension. As illustrated in Fig. 1c, $O[\mathcal{N}, :]$ with $\mathcal{N} \in [1, \dots, n]$ is the raw observation of nodes, and $O[\mathcal{F}, :]$ with $\mathcal{F} \in [n+1, \dots, m]$ is the observation of factors initialized by averaging the observations of constituent nodes. We calculate factor embeddings using nodes, followed by computing node embeddings using factors. During message passing, masks are employed to ensure each factor only pays attention to its connected nodes, and each node only pays attention to its

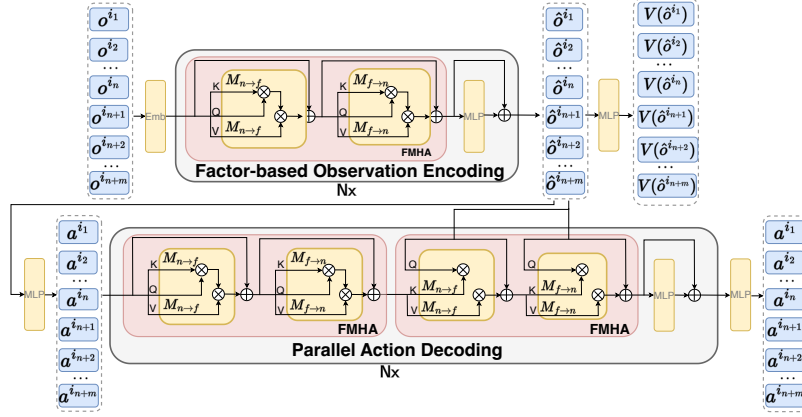


Figure 2: Architecture of f -MAT. At each time step, the encoder takes the observation of nodes and factors as the input and outputs the factor-based observation representation. The factor observation is initialized by the average of related agent’s observations. In decoder, we initialized the actions by the learned observation representation and generate actions in parallel. All attention layers utilized are factor-based attention layers. The pseudo code of f -MAT can be found in Appendix A.

connected factors. To maintain the fixed length observation $n + m$, we keep the nodes unchanged when updating the factors, and the factors unchanged when updating the nodes.

For example, in Fig. 1a, nodes are defined as $\mathcal{N} = \{1, \dots, 4\}$, factors are defined as $\mathcal{F} = \{5, 6\}$, and the input of the factor-based model is $O[\mathcal{N} \cup \mathcal{F}, :] = (o^1, \dots, o^4, o^5, o^6)^\top$. In the attention layer that sends messages from nodes to factors, we first keep nodes (o^1, \dots, o^4) unchanged, then update factors by masking out all unrelated nodes. Assume the updated factors are denoted as $(\tilde{o}^5, \tilde{o}^6)$, and the output of the attention layer with mask $M_{n \rightarrow f}$ (meaning node to factor) is $(o^1, \dots, o^4, \tilde{o}^5, \tilde{o}^6)$. In the attention layer that sends messages from factors to nodes, we first keep factors $(\tilde{o}^5, \tilde{o}^6)$ unchanged, then update nodes by masking out all unrelated factors. Assume the updated nodes are denoted as $(\tilde{o}^1, \dots, \tilde{o}^4)$, and the output of the attention layer with mask $M_{f \rightarrow n}$ is $(\tilde{o}^1, \dots, \tilde{o}^4, \tilde{o}^5, \tilde{o}^6)$.

Here, we defined two mask matrices of $(n+m) \times (n+m)$: $M_{n \rightarrow f}$ for message passing from nodes/agents to factors/-groups, and its *transpose* as $M_{f \rightarrow n}$ for the opposite direction:

$$M_{n \rightarrow f}(i, j) = \begin{cases} 1 & \text{if } i \in \mathcal{N} \text{ and } j \in \mathcal{F} \text{ and } i \in f_j \\ 0 & \text{else} \end{cases}. \quad (1)$$

Both matrices are sparse, rendering optimization in implementation. Fig. 1b shows the top-right corner of $M_{n \rightarrow f}$ for the running example, i.e., rows 1 to n and columns $n+1$ to $n+m$; the other elements are 0. The queries Q are no longer represented by $O \in \mathbb{R}^{(n+m) \times D}$. Instead, in factor-based attention, queries Q are set to $O[\mathcal{F}, :]$, keys K to $O[\mathcal{N}, :]$, and values V to $O[\mathcal{N}, :]$. Similar operations apply to $M_{f \rightarrow n}$.

Applying the factor-based attention mechanism to our running example, the queries are $Q = \{5, 6\}$, with keys K and values V being $\{1, 2, 3, 4\}$. When updating factors, each factor only attends to its constituent nodes. Thus, o^5_q only attends to $\{o^1_k, o^2_k, o^3_k\}$, and o^6_q only attends to $\{o^2_k, o^4_k\}$. Once factors o^5 and o^6 are updated, we maintain the fixed-length observation $n + m$ without changing those for the nodes.

The update of nodes and factors is completed after passing through two layers of factor-based attention (node-to-factor and factor-to-node). We define it as factor-based multi-head attention (f -MHA), and multiple f -MHA layers allow longer-distance propagation between agents through factors, so that all the necessary information is spread throughout the graph.

The time and space complexity of the factor-based attention model with L layers is $O(L \cdot |\mathcal{E}|)$, which is bounded by $O(m \cdot S_f \cdot L)$. Typically, $L \leq 3$. Depending on the factor topology, it can be much more efficient than the traditional attention models, which cost $O(n^2)$. We will demonstrate the savings via the three experiments in Sec. 5.

4.3 Encoder and Decoder Implementations

The overall architecture of f -MAT can be found in Fig. 2. Our implementation is based on MAT, where we replaced conventional MHA with f -MHA, leading to a few nuances in the encoder/decoder updates and inference.

Encoder In conventional encoder-decoder transformers as well as in MAT, attention is not masked in encoders, leading to centralized policy w.r.t. observations, i.e., requiring joint observation \mathbf{o} as the input. In contrast, f -MAT applies factor-based masks to enable local message passing, so that a policy only needs to draw upon **local observations** instead of global ones. Equation (2) below formalizes the encoder objective of f -MAT, using the Bellman error of the value function V that is defined locally. The pseudocode is given in Algorithm 2.

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

Decoder Similarly, decoders are composed of our factor-based attention layers, and training is similar to conventional decoder training, except that our attention is local. **The major difference lies in inference**, the computational bottleneck of MAT due to the auto-regression. Furthermore, since there is generally no natural order among agents, MAT manually introduces a random order and regenerates it every iteration.

In contrast, we propose **parallel inference** for f -MAT which is by itself novel for transformers. The method is detailed in Algorithm 3 in Appendix A, where message passing alternates between node-to-factor and factor-to-node in a **synchronized** fashion. This resembles Gibbs sampling, and we sketch the connection in Appendix B.

We observe that a small number of f -MHA layers ($L \leq 3$) can already produce good performance in our experiments. In addition, we consider using action distributions (directly maps the factored observations to individual actions through linear layers) as initialization to bypass the slow mixing issue of Gibbs, as it is known that good initialization improves efficiency of samplers (Boland, Friel, and Maire 2018).

For training, we update the decoder using PPO loss (Schulman et al. 2017) with a factorized policy:

$$L_{\text{dec}}(\theta) = \frac{-1}{Tn} \sum_{i=0}^n \sum_{t=0}^{T-1} \min \left(r_t^i(\theta) \hat{A}_t, \text{clip}(r_t^i(\theta), 1 \pm \epsilon) \hat{A}_t \right) \quad (3)$$

$$r_t^i(\theta) = \pi_{\theta}^i(a_t^i | \hat{o}_t^{\text{rf}(i)}) / \pi_{\theta_{\text{old}}}^i(a_t^i | \hat{o}_t^{\text{rf}(i)})$$

The policy π^i for agent i draws on the observations from its **reception field** $\text{rf}(i)$. It consists of all the agents that can be reached from i with at most $2L$ hops on the factor graph.

The complete pseudocode for f -MAT’s encoder and decoder can be found in Algorithm 1 in Appendix A. f -MAT seeks a balance between centralized and decentralized execution, offering a novel solution to cooperative MARL. The key insight is the factor-based mechanism, which facilitates efficient and extensive message passing among agents from graph modeling perspective. Additionally, f -MAT enables parallel action generation within the transformer model, further reducing computational time and making it better suited for environments that require agents to take cooperative actions simultaneously.

5 Experiments

We evaluate f -MAT in three environments, each presenting different challenges. The first environment is a strong cooperation scenario named grid alignment, where the agents prefer to align their actions with the neighbors of the same row or column. The second environment is traffic light control with heterogeneous agents. The third environment is power control, emphasizing local control where an agent’s actions have a limited effect on those distant from it. We choose MAT, a fully centralized method, as the performance upper bound. Then, we primarily utilize MAPPO and MAT-dec* under the CTDE framework as our baseline competitors. We evaluate all methods in terms of performance and training efficiency.

5.1 Grid Alignment

Environment Our first experiment is on a simplified domain of traffic flow (Zhang, Aberdeen, and Vishwanathan 2007), called **GridSim**, where all agents coordinate their actions to maximize the global reward. The traffic flow is an $s \times s$ grid shown in Fig. 7a in Appendix C. Each row

*MAT-dec is a more decentralized variant of MAT, but it still has a centralized component at execution, namely the encoder.

and column includes a buffer to hold traffic units that arrive with a probability ($\text{Pr} = 0.5$) at each timestep. At each grid intersection, an agent controls a gate, which can be chosen from two actions: keeping the gate aligned horizontally or vertically. Traffic can only flow through a column if **all** its gates are vertically aligned, and similarly through a row if all its gates are horizontally aligned. When this happens, all waiting traffic for that line propagates instantly, and each unit of traffic contributes $+1$ to a global reward. The optimal reward is the grid size s . So each agent should ideally choose the same actions as its directly preceding neighbor, especially if that neighbor is in the lane with the most waiting traffic in the buffer. Hence, this environment emphasizes the collaboration of direct neighbors.

Results It is natural to form groups/factors along each row and column. Figure 3 illustrates the resulting performance and training efficiency. The ‘gs’ in legends refers to the group size (S_f) - ‘gs4’ means that each factor involves $S_f = 4$ agents located consecutively along each row or column. This leads to $O(s(s - S_f)S_fL)$ complexity, which is much lower than $O(n^2) = O(s^4)$.

MAT sets an upper bound in GridSim where an agent’s action depends on the actions of preceding neighbors. Comprehensive knowledge of all preceding agents aids decision-making. In the 8×8 grid shown in Fig. 3a, while all CTDE methods achieve optimal results, f -MAT is the closest to MAT in performance and converges the fastest. As we move towards a 12×12 grid, the differences in performance among the four methods become more pronounced. f -MAT continues to approach the optimal performance, whereas MAT-dec and MAPPO fall short, achieving 75% and 65% of f -MAT’s performance on 10×10 (Fig. 3b) and 12×12 grids (Fig. 3c), respectively. Interestingly, Fig. 3b and 3c show that the group size significantly impacts the global reward. We will delve into the impact of group size in Sec. 5.4.

To compare the training efficiency, Fig. 3d shows that f -MAT achieves optimal performance among all CTDE methods, just slightly slower than MAT. f -MAT requires only 1/2 of the training time taken by MAPPO and MAT-dec to reach the similar performance. This confirms that f -MAT is significantly more efficient in learning. We note that this subplot is based on a single seed, because it is difficult to plot the average over multiple seeds. We hence present the results of two more seeds in Figure 8a in Appendix D. Similar results with additional seeds for traffic control and power control are in Fig. 9a and Fig. 10, respectively.

5.2 Traffic Light Control

Environment The second environment adapted the Simulation of Urban Mobility (SUMO, Chen et al. 2020; Ault and Sharon 2021), which is widely recognized in the transportation community. We chose as our testbed an area with $n = 28$ traffic lights in Monaco (Chu, Chinchali, and Katti 2020), illustrated in Fig. 7b in Appendix C. We used the average queue length at intersections to measure the level of traffic congestion. Traffic light control is a challenging application in MARL because each traffic light needs to observe a wider area to make decisions, and the actions of each agent impact a broader region beyond just adjacent agents. Another

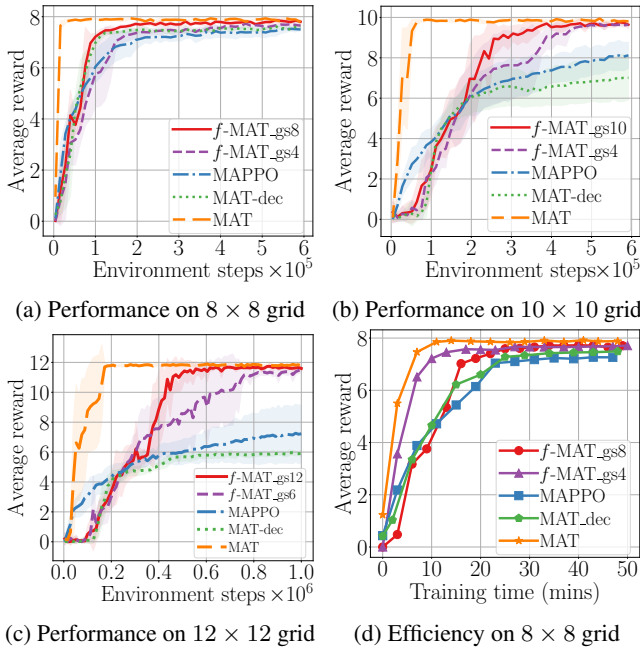


Figure 3: The performance results for **GridSim** with three different number of agents ($n = 64, 100, 144$) and the training efficiency on grid 8×8 . All performance results are presented as mean \pm std. ‘Efficiency’ in subplot (d) refers to training efficiency, i.e., evaluation reward vs. training time.

challenge arises from the *heterogeneity* of agents in terms of observation and action.

Results It is less clear here how to form the groups based on the problem formulation or the definition of the reward. We randomly divided the agents into two groups or four groups, and then we added each agent’s directly connected neighbors. This leads to f -MAT-f2 with $S_f = 20$ and f -MAT-f4 with $S_f = 15$ approximately. The complexity of $O(S_f L)$ is much lower than O^2 . We will discuss group selection later in Sec. 5.4.

Fig. 4 shows that under different numbers of factors, f -MAT produces performance comparable to MAT and achieves a higher training efficiency than all other methods.

Furthermore, f -MAT maintains its effectiveness even in a heterogeneous environment. In contrast, MAPPO, being a shared parameter approach, is inherently susceptible to failure in an inhomogeneous setting. MAT, MAT-dec, and f -MAT incorporate an embedding layer that aligns the diverse observation and action dimensions, thereby enabling their adaptability and success in heterogeneous environments.

Given the need for collaboration over a broader area and communication between agents in this environment, f -MAT outperforms MAT-dec in both performance and training efficiency. f -MAT learns faster than MAT, requiring 3/5 training time of MAPPO and MAT-dec to achieve comparable results. This highlights the need for cooperation during execution and further validates the advantage of f -MAT.

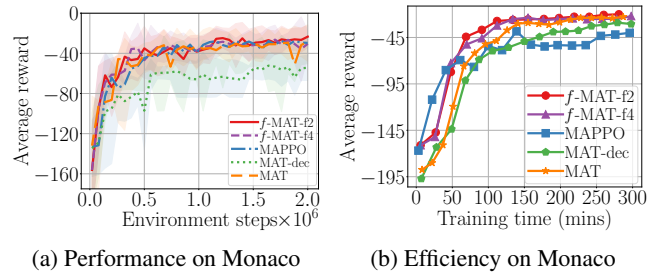


Figure 4: The performance and training efficiency results for **traffic light control**, an area in Monaco with 28 traffic lights. f -MAT is compatible with *heterogeneous* environments.

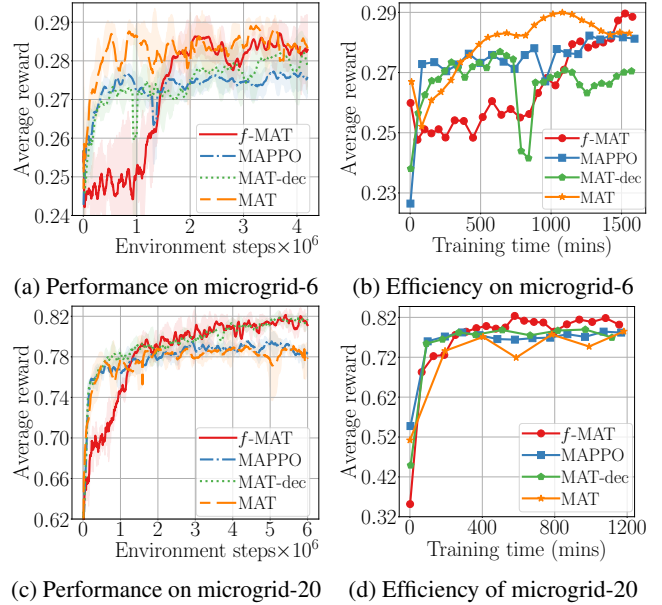


Figure 5: The performance and training efficiency results for **power grid control**. f -MAT outperforms more evidently in complex environments.

5.3 Power Control

Environment The voltage control problem in distributed generators (DGs) can be viewed as a cooperative MARL problem. We have two microgrid systems (Chen et al. 2021): one with 6 distributed DGs (microgrid-6) and a larger-scale microgrid system with 20 DGs (microgrid-20), both shown in Fig. 7c in Appendix C. Power control is an environment where communication among all agents is not necessary, as control infrastructures are typically dispersed across a large area. It is a widely used environment for communication-based methods. We reference their results and compare them with our methods in the Table 2 in Appendix D.

Results We manually divided groups with only one overlapping agent. In microgrid-6, we set the number of factors to $m = 2$ and $S_f = 4$. In microgrid-20, we set the number of factors to $m = 3$ and $S_f = 6$. This results in a complexity of $O(S_f L)$. Fig. 5 shows that f -MAT is one of the best performing methods in the two microgrid systems, achieving

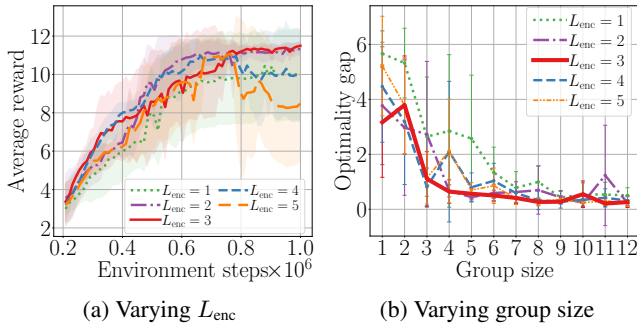


Figure 6: Ablation on GridSim: (a) Average reward under different L_{enc} with group size = 6. (b) Comparison on optimality gap under different L_{enc} and group sizes.

the highest training efficiency.

One observation from this environment is that MAT outperforms other methods in microgrid 6, but suffers the poorest performance in microgrid 20, probably due to the local control nature of the setting, where an agent does not require information from all others to make decisions. Utilizing fully centralized observations during execution in the decentralized environment may introduce irrelevant information, potentially impairing performance. This can also explain that MAT-dec yields results close to MAT and f -MAT in the microgrid 6 system, and achieves one of the best performance in the microgrid 20.

Another observation is that f -MAT demonstrates its superior training efficiency more slowly in microgrid 6 compared to microgrid 20, suggesting that f -MAT exhibits its advantages more easily in complex environments.

In this section, we conducted experiments on three environments with different communication scopes: direct preceding neighborhoods (grid alignment), local neighborhoods (power control) and broader areas (traffic light control). The selection of factors ranges from a clearly defined formulation (grid alignment) and limited scope (power control) to random choices on a general graph (traffic light control). In addition, the observation and action spaces include both homogeneous and heterogeneous settings.

5.4 Ablation

Choice of L_{enc} and L_{dec} We select a 12×12 grid from GridSim, fix the group size to $S_f = 6$, and then vary the value of L_{enc} from 1 to 4. As shown in Fig. 6a, $L_{enc} = 3$ produces the most stable trend and achieves the highest reward. The results of group sizes $S_f = 9, 12$ in Appendix Fig. 8b and 8c further support this observation. To explore the relationship between L_{enc} and group size, we use the optimality gap, the value between the true optimal reward and the learned reward achieved by the algorithm, to illustrate the variations. In Fig. 6b, $L_{enc} = 3$ generally yields good performance among various group sizes. We notice an initial improvement in performance as L_{enc} increases. However, further increases do not consistently enhance results. As the number of layers grows, the information received by agents becomes more homogeneous, making it challenging to distinguish between

individual agent features. Such an over-smoothing effect is also observed in GNNs (Kipf and Welling 2017).

Furthermore, the results for group size = 12 demonstrate that selecting the appropriate group can lead to optimal rewards with a smaller L_{enc} . The results for group sizes from 6 to 11 show that slightly increasing L_{enc} can also offset a less ideal group choice, but increasing it to 4 or 5 leads to limited benefits, as the occasional performance boost cannot compensate for the significantly increased computation time incurred by a larger value of L_{enc} .

Based on the above experiments, we recommend setting $L_{enc} = 3$, which we used to produce our main results.

Choice of group size It is challenging to theoretically analyze the impact of group selection and size. Here, we quote some relevant results from Lemma 2 in Qu, Wierman, and Li (2022) and Theorem 3 in Ma et al. (2024) to support our results, although their setting only considers direct links between agents rather than through factors. As Ma et al. (2024) shows, the optimality gap decays exponentially with increasing k , where k is the number of hops used in learning. But when k is larger than a threshold, other aspects such as sample efficiency, computational cost, and the representational capacity of the neural network must be considered, leading to the increase of the optimality gap.

In our setting, the group size S_f can be considered a counterpart of k . So, the above theoretical results (for k) are consistent with our experiments with varied values of S_f across different L_{enc} shown in Fig. 6b. To select S_f , an economical way is to choose the "elbow" value. Or, one can pick S_f that achieves the best performance, provided that computational cost and efficiency are manageable.

Choice of group selection The group selection in f -MAT is flexible. In GridSim, we designated groups based on the definition of reward. We can also first choose the number of groups and then randomly pick agents and its related neighbor (directed neighbor, k -nearest neighbor, k -hop neighbor), similar to our approach with traffic lights. From Fig. 9b, we selected 2 and 4 as number of factors discussed in Sec. 5.2.

Computational time Recall the complexity of MHA in f -MAT is $O(m \cdot S_f \cdot L)$. We conducted experiments on the inference time cost to demonstrate this big-O complexity, and the result in 12×12 GridSim is shown in the Table 1 in Appendix D. The group size of f -MAT is 12. f -MAT runs in a comparable computation time to MAPPO during inference, benefiting from its parallel action generation mechanism. MAT generates actions autoregressively during inference, making it the slowest method during inference. MAT-dec modifies the decoder of MAT by replacing the attention block with an MLP, yet it still generates actions autoregressively. As a result, MAT-dec is only slightly faster than MAT.

Conclusion In this paper, we propose Factor-based Multi-Agent-Transformer that enables efficient collaborations in both training and execution through all agents via graph modeling within the CTDE framework. This approach enriches CTDE framework by incorporating neighborhood interactions during execution. Empirical results demonstrate that f -MAT achieves strong performance across diverse environments. Future work will concentrate on dynamic graphs and the development of learnable factors.

References

- Ault, J.; and Sharon, G. 2021. Reinforcement Learning Benchmarks for Traffic Signal Control. In *Proceedings of the Thirty-fifth Conference on Neural Information Processing Systems (NeurIPS 2021) Datasets and Benchmarks Track*.
- Bishop, C. 2006. *Pattern Recognition and Machine Learning*. Springer.
- Boehmer, W.; Kurin, V.; and Whiteson, S. 2020. Deep Coordination Graphs. In *International Conference on Machine Learning (ICML)*.
- Boland, A.; Friel, N.; and Maire, F. 2018. Efficient MCMC for Gibbs random fields using pre-computation. *Electronic Journal of Statistics*, 12(2): 4138–4179.
- Casella, G.; and George, E. I. 1992. Explaining the Gibbs sampler. *The American Statistician*, 46(3): 167–174.
- Chen, C.; Wei, H.; Xu, N.; Zheng, G.; Yang, M.; Xiong, Y.; Xu, K.; and Li, Z. 2020. Toward a thousand lights: Decentralized deep reinforcement learning for large-scale traffic signal control. In *Proceedings of the AAAI conference on artificial intelligence*, 3414–3421.
- Chen, D.; Chen, K.; Li, Z.; Chu, T.; Yao, R.; Qiu, F.; and Lin, K. 2021. Powernet: Multi-agent deep reinforcement learning for scalable powergrid control. *IEEE Transactions on Power Systems*, 37(2): 1007–1017.
- Chu, T.; Chinchali, S.; and Katti, S. 2020. Multi-agent Reinforcement Learning for Networked System Control. In *International Conference on Learning Representations (ICLR)*.
- Das, A.; Gervet, T.; Romoff, J.; Batra, D.; Parikh, D.; Rabbat, M.; and Pineau, J. 2019. Tarmac: Targeted multi-agent communication. In *International Conference on Machine Learning (ICML)*.
- Foerster, J. N.; Assael, Y. M.; de Freitas, N.; and Whiteson, S. 2016. Learning to communicate with Deep multi-agent reinforcement learning. In *Advances in Neural Information Processing Systems (NeurIPS)*.
- Foerster, J. N.; Farquhar, G.; Afouras, T.; Nardelli, N.; and Whiteson, S. 2018. Counterfactual multi-agent policy gradients. In *National Conference of Artificial Intelligence (AAAI)*.
- Geman, S.; and Geman, D. 1984. Stochastic relaxation, Gibbs distributions, and the Bayesian restoration of images. *IEEE Transactions on pattern analysis and machine intelligence*, PAMI-6(6): 721–741.
- Gonzalez, J.; Low, Y.; Gretton, A.; and Guestrin, C. 2011. Parallel gibbs sampling: From colored fields to thin junction trees. In *Proceedings of the Fourteenth International Conference on Artificial Intelligence and Statistics*, 324–332. JMLR Workshop and Conference Proceedings.
- Guestrin, C.; Lagoudakis, M.; and Parr, R. 2002. Coordinated reinforcement learning. In *International Conference on Machine Learning (ICML)*.
- Gupta, S.; Hazra, R.; and Dukkipati, A. 2020. Networked Multi-Agent Reinforcement Learning with Emergent Communication. In *Proceedings of the 19th International Conference on Autonomous Agents and MultiAgent Systems*.
- Hao, Q.; Huang, W.; Feng, T.; Yuan, J.; and Li, Y. 2023. GAT-MF: Graph Attention Mean Field for Very Large Scale Multi-Agent Reinforcement Learning. In *Proceedings of the 29th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, 685–697.
- Hoshen, Y. 2017. VAIN: Attentional Multi-agent Predictive Modeling. In *Advances in Neural Information Processing Systems (NeurIPS)*.
- Jiang, J.; Dun, C.; Huang, T.; and Lu, Z. 2018. Graph convolutional reinforcement learning. *arXiv preprint arXiv:1810.09202*.
- Jiang, J.; and Lu, Z. 2018. Learning attentional communication for multi-agent cooperation. In *Advances in Neural Information Processing Systems (NeurIPS)*.
- Kim, D.; Moon, S.; Hostallero, D.; Kang, W. J.; Lee, T.; Son, K.; and Yi, Y. 2019. Learning to schedule communication in multi-agent reinforcement learning. In *International Conference on Learning Representations (ICLR)*.
- Kipf, T. N.; and Welling, M. 2017. Semi-Supervised Classification with Graph Convolutional Networks. In *International Conference on Learning Representations (ICLR)*.
- Kuba, J. G.; Chen, R.; Wen, M.; Wen, Y.; Sun, F.; Wang, J.; and Yang, Y. 2022. Trust Region Policy Optimisation in Multi-Agent Reinforcement Learning. In *International Conference on Learning Representations (ICLR)*.
- Kuba, J. G.; Wen, M.; Meng, L.; Zhang, H.; Mguni, D.; Wang, J.; Yang, Y.; et al. 2021. Settling the variance of multi-agent policy gradients. *Advances in Neural Information Processing Systems*, 34: 13458–13470.
- Li, S.; Gupta, J. K.; Morales, P.; Allen, R.; and Kochenderfer, M. J. 2020. Deep implicit coordination graphs for multi-agent reinforcement learning. *arXiv preprint arXiv:2006.11438*.
- Littman, M. L. 1994. Markov games as a framework for multi-agent reinforcement learning. In *Machine learning proceedings 1994*, 157–163. Elsevier.
- Lowe, R.; Wu, Y. I.; Tamar, A.; Harb, J.; Abbeel, P.; OpenAI; and Mordatch, I. 2017. Multi-agent actor-critic for mixed cooperative-competitive environments. In *Advances in Neural Information Processing Systems (NeurIPS)*.
- Ma, C.; Li, A.; Du, Y.; Dong, H.; and Yang, Y. 2024. Efficient and scalable reinforcement learning for large-scale network control. *Nature Machine Intelligence*, 6(9): 1006–1020.
- Newman, D.; Smyth, P.; Welling, M.; and Asuncion, A. 2007. Distributed inference for latent dirichlet allocation. *Advances in neural information processing systems*, 20.
- Niu, Y.; Paleja, R.; and Gombolay, M. 2021. Multi-Agent Graph-Attention Communication and Teaming. In *Proceedings of the 20th International Conference on Autonomous Agents and MultiAgent Systems*.
- Qu, G.; Wierman, A.; and Li, N. 2022. Scalable reinforcement learning for multiagent networked systems. *Operations Research*, 70(6): 3601–3628.
- 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. *Journal of Machine Learning Research (JMLR)*, 21(1).

Ryu, H.; Shin, H.; and Park, J. 2020. Multi-agent actor-critic with hierarchical graph attention network. In *Proceedings of the AAAI Conference on Artificial Intelligence*, 7236–7243.

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

Singh, A.; Jain, T.; and Sukhbaatar, S. 2019. Individualized Controlled Continuous Communication Model for Multiagent Cooperative and Competitive Tasks. In *International Conference on Learning Representations (ICLR)*.

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.

Sukhbaatar, S.; Szlam, A.; and Fergus, R. 2016. Learning multiagent communication with backpropagation. In *Advances in Neural Information Processing Systems (NeurIPS)*.

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*.

Wen, M.; Kuba, J.; Lin, R.; Zhang, W.; Wen, Y.; Wang, J.; and Yang, Y. 2022. Multi-agent reinforcement learning is a sequence modeling problem. In *Advances in Neural Information Processing Systems (NeurIPS)*.

Yu, C.; Velu, A.; Vinitzky, E.; Gao, J.; Wang, Y.; Bayen, A.; and Wu, Y. 2022. The Surprising Effectiveness of PPO in Cooperative Multi-Agent Games. In *Thirty-sixth Conference on Neural Information Processing Systems Datasets and Benchmarks Track*.

Zhang, K.; Yang, Z.; and Başar, T. 2021. Decentralized multi-agent reinforcement learning with networked agents: Recent advances. *Frontiers of Information Technology & Electronic Engineering*, 22(6): 802–814.

Zhang, K.; Yang, Z.; Liu, H.; Zhang, T.; and Basar, T. 2018. Fully Decentralized Multi-Agent Reinforcement Learning with Networked Agents. In *International Conference on Machine Learning (ICML)*.

Zhang, X.; Aberdeen, D.; and Vishwanathan, S. V. N. 2007. Conditional random fields for multi-agent reinforcement learning. In *International Conference on Machine Learning (ICML)*.