RESEARCH-ARTICLE

DyPS: Dynamic Parameter Sharing in Multi-Agent Reinforcement Learning for Spatio-Temporal Resource Allocation

JINGWEI WANG, Beijing National Research Center for Information Science and Technology, Beijing, China

QIANYUE HAO, Beijing National Research Center for Information Science and Technology, Beijing, China

WENZHEN HUANG, Beijing National Research Center for Information Science and Technology, Beijing, China

XIAOCHEN FAN, Beijing National Research Center for Information Science and Technology, Beijing, China

ZHENTAO TANG, Huawei Technologies Noah's Ark Lab, Hong Kong, Hong Kong

BIN WANG, Huawei Technologies Noah's Ark Lab, Hong Kong, Hong Kong

View all

Open Access Support provided by:

Beijing National Research Center for Information Science and Technology

Huawei Technologies Noah's Ark Lab

Published: 25 August 2024

Citation in BibTeX format

KDD '24: The 30th ACM SIGKDD Conference on Knowledge Discovery and Data Mining
August 25 - 29, 2024
Barcelona, Spain

Conference Sponsors:
SIGMOD
SIGKDD

DyPS: Dynamic Parameter Sharing in Multi-Agent Reinforcement Learning for Spatio-Temporal Resource Allocation

Jingwei Wang*
Department of EE, BNRist,
Tsinghua University
Beijing, China

Qianyue Hao*
Department of EE, BNRist,
Tsinghua University
Beijing, China

Wenzhen Huang
Department of EE, BNRist,
Tsinghua University
Beijing, China

Xiaochen Fan
Institute for Electronics and
Information Technology in Tianjin,
Department of EE, BNRist,
Tsinghua University
Beijing, China

Zhentao Tang
Huawei Noah's Ark Lab
Beijing, China

Bin Wang
Huawei Noah's Ark Lab
Beijing, China

Jianye Hao
Tianjin University,
Huawei Noah's Ark Lab
Beijing, China

Yong Li
Department of EE, BNRist,
Tsinghua University
Beijing, China
liyong07@tsinghua.edu.cn

ABSTRACT

In large-scale metropolis, it is critical to efficiently allocate various resources such as electricity, medical care, and transportation to meet the living demands of citizens, according to the spatio-temporal distributions of resources and demands. Previous researchers have done plentiful work on such problems by leveraging Multi-Agent Reinforcement Learning (MARL) methods, where multiple agents cooperatively regulate and allocate the resources to meet the demands. However, facing the great number of agents in large cities, existing MARL methods lack efficient parameter sharing strategies among agents to reduce computational complexity. There remain two primary challenges in efficient parameter sharing: (1) during the RL training process, the behavior of agents changes significantly, limiting the performance of group parameter sharing based on fixed role division decided before training; (2) the behavior of agents forms complicated action trajectories, where their role characteristics are implicit, adding difficulty to dynamically adjusting agent role divisions during the training process. In this paper, we propose **Dy**namic **P**arameter **S**haring (**DyPS**) to solve the above challenges. We design self-supervised learning tasks to extract the implicit behavioral characteristics from the action trajectories of agents. Based on the obtained behavioral characteristics, we propose a hierarchical MARL framework capable of dynamically revising the agent role divisions during the training process and thus shares parameters among agents with the same role, reducing computational complexity. In addition, our framework can be combined with various typical MARL algorithms, including IPPO, MAPPO, etc. We conduct 7 experiments in 4 representative resource allocation scenarios, where extensive results demonstrate our method's superior performance, outperforming the state-of-the-art baseline methods by up to 31%. Our source codes are available at https://github.com/tsinghua-fib-lab/DyPS.

CCS CONCEPTS

• **Computing methodologies** → **Multi-agent reinforcement learning**; **Multi-agent planning**; • **Applied computing** → Supply chain management.

KEYWORDS

Multi-agent reinforcement learning, dynamic parameter sharing, spatio-temporal resource allocation.

ACM Reference Format:
Jingwei Wang*, Qianyue Hao*, Wenzhen Huang, Xiaochen Fan, Zhentao Tang, Bin Wang, Jianye Hao, and Yong Li. 2024. DyPS: Dynamic Parameter Sharing in Multi-Agent Reinforcement Learning for Spatio-Temporal Resource Allocation. In *Proceedings of the 30th ACM SIGKDD Conference on Knowledge Discovery and Data Mining (KDD '24), August 25–29, 2024, Barcelona, Spain.* ACM, New York, NY, USA, 12 pages. https://doi.org/10.1145/3637528.3672052

1 INTRODUCTION

In large-scale modern urban scenarios, it is a crucial decision-making problem to allocate a large number of various resources, *e.g.*, water [33, 42], electricity [51], medical care [17, 18], transportation [19, 49, 50], etc., across the city (Figure 1a). Proper and efficient allocation of resources meets the living and industrial demands of citizens, which is the basis for ensuring the normal functioning

*Equal contribution.

KDD '24, August 25–29, 2024, Barcelona, Spain

ACM ISBN 979-8-4007-0490-1/24/08.
https://doi.org/10.1145/3637528.3672052

and prosperity of the cities. However, due to the heterogeneous and time-varying distribution of the resources and demands, the optimal solution to such resource allocation problem must fully take the complicated spatio-temporal characteristics into consideration [3, 14, 36, 37]. Therefore, finding efficient resource allocation strategies is a challenging yet important problem.

Recent advancements in reinforcement learning (RL) inspire researchers to solve resource allocation problem within the framework of Markov decision process (MDP) [17, 18]. Especially, multi-agent reinforcement learning (MARL) is widely applied in such problems [19, 28, 30, 33, 50], where multiple RL agents work cooperatively to regulate and allocate resources. In large-scale cities, the amount of resources and demands tend to be enormous, which requires a large number of agents to be fully capable of allocating resources throughout the city, leading to an unacceptable huge number of learnable parameters and computational consumption (Figure 1b). Facing such situation, a typical solution is parameter sharing among agents [6, 10, 13, 35, 48]. Some rudimentary solutions simply share parameters among all agents [10, 13, 35, 48], which minimize the number of learnable parameters but discard the behavioral differences among agents with different roles (Figure 1c). Modified design models the spatio-temporal features of the agents before RL training and clusters agents into fixed groups according to their similarities, where agents in each group share parameters [6]. This design maintains the modeling of differences among the roles of agents while keep a small number of learnable parameters (Figure 1d).

Despite the prevalence of MARL with parameter sharing in solving resource allocation problems, there remain two major unsolved challenges in these methods, limiting the performance of existing parameter sharing strategies. (1) **Changing of agents' roles during training.** The majority of existing solutions fix the role grouping of agents before RL training, but during the training process, the roles of agents change significantly when their behaviors are updated. Therefore, the pre-fixed role grouping may not match the well-trained agents and limits the overall performance. (2) **Difficulty in identifying agents' roles.** The behavior of agents forms complicated action trajectories during the steps of the decision-making process, where their role characteristics are implicitly embedded in the trajectories. This hinders the identification of the agents' explicit roles and increases the difficulty of dynamically adjusting the roles during the training process.

Facing these challenges, we propose **Dy**namic **P**arameter **S**haring (**DyPS**) framework, which dynamically identifies and adjusts the agents' roles along with the RL training process (Figure 1e), solving large-scale resource allocation problems. Specifically, DyPS includes a self-supervised role modeling part and a hierarchical MARL part. The former part employs a Variational Long Short-term Memory (VLSTM) [15] and a Conditional Variational Auto Encoder (CVAE) [38] to respectively depict the behavioral characteristics of each agent and each group from the action trajectories. The latter part consists of two levels, the Group Selection Module and the Group-based Resource Allocation Module, where the Group Selection Module dynamically groups agents according to their roles based on the behavioral characteristics extracted by VLSTM and CVAE, and the Group-based Resource Allocation Module learns multi-agent resource allocation strategies with parameters shared among agents within the same group. Both modules are trained via Actor-Critic (AC) [24] manner according to the feedback in agent-environment interactions. We evaluate our framework across 7 experiments of 4 representative resource allocation scenarios, and extensive results indicate its superior performance, surpassing the state-of-the-art baseline methods by up to 31%. Also, our framework can be easily combined with multiple typical MARL algorithms and improve their performance on resource allocation problems.

In summary, the main contributions of this work include:

- We design a dynamic parameter sharing MARL framework for large-scale resource allocation problems. This framework is trained via environmental feedback, which dynamically groups the agents according to their changing roles during RL training. Then, agents within the same group share parameters, reducing the computational complexity while maintaining up-to-date modeling of agents' roles.
- We employ VLSTM and CVAE to extract behavioral characteristics of agents from their action trajectories via self-supervised learning. Therefore, we can identify the role of agents based on their behavioral characteristics and then dynamically adjust the grouping during RL training.
- Extensive experiments on various scenarios demonstrate that DyPS significantly outperforms the state-of-the-art baseline methods with up to 31% improvement on resource allocation tasks. Moreover, DyPS can be easily combined with multiple MARL algorithms and improve their performance on resource allocation.

2 PRELIMINARIES

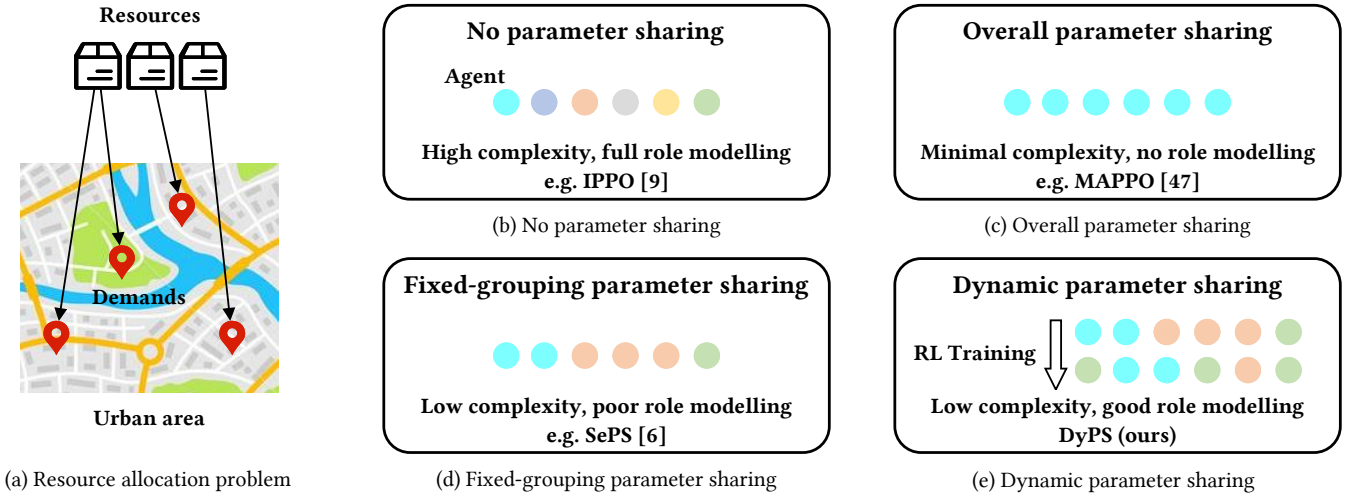
2.1 Problem Formulation

In this paper, we mainly consider a targeted urban area that is divided into a set of K disjoint uniform grids, and T is the total of number time steps considered in the task. Here, we provide a mathematical definition of resource, demand, and spatio-temporal resource allocation problems.

Definition 1 (Resource). *The total number of available resources at time step t is denoted as N_t. The available resources are distributed across the grids, where the number of available resources at time step t in grid k is denoted as N_t^k, and $\sum_k N_t^k = N_t$.*

Definition 2 (Demand). *Demands in the city are needs for resources, which are distributed across the grids. The number of demands at time step t in grid k is denoted as D_t^k, and $\sum_k D_t^k$ can be either smaller or larger than N_t, which corresponds to the situation with abundant or scarce resources.*

Definition 3 (Spatio-Temporal Resource Allocation). *Given a targeted urban area with certain resources and demands over T time steps, a resource allocation strategy is to move the resources among grids and meet the demands in each grid at each time step. The number of resources moved from grid i to j at step t is denoted as M_t^{ij}. Additionally, constraints may appear at specific tasks, where the movement of resources must follow the constrains. The goal of this problem is to find an effective resource allocation strategy to maximize the global benefit after utilizing the resources.*



(a) Resource allocation problem (b) No parameter sharing (c) Overall parameter sharing (d) Fixed-grouping parameter sharing (e) Dynamic parameter sharing

Figure 1: Comparison among resource allocation MARL methods with different parameter sharing designs. (a) Illustration of resource allocation problem in urban scenario. (b) None parameter sharing. (c) Overall parameter sharing. (d) Fixed-grouping parameter sharing. (e) Dynamic parameter sharing (proposed *DyPS*).

2.2 Multi-Agent Markov Decision Process

A spatio-temporal resource allocation problem can be solved iteratively following the framework of multi-agent Markov decision process (MAMDP), where multiple cooperative agents gradually allocate the resources to meet the needs step by step. A MAMDP [31] can be defined by the tuple $(n, \mathcal{S}, \mathcal{A}, P, R, \gamma)$, where n denotes the total number of agents. The global state $s = (s_1, \cdots, s_n) \in \mathcal{S} = \mathcal{S}^1 \times \cdots \times \mathcal{S}^N$ consists of local state of each agent. The joint action $a = (a_1, \cdots, a_n) \in \mathcal{A} = \mathcal{A}^1 \times \cdots \times \mathcal{A}^N$ consists of the local action of each agent and is produced by the global policy $\pi_\theta : \mathcal{S} \mapsto \mathcal{A}$. The global policy π_θ is parameterized with $\theta = (\theta_1, \cdots, \theta_N)$ and each agent i is assigned with a local policy π_{θ_i} to produce local action a_i. The state transition probability function $P : \mathcal{S} \times \mathcal{A} \mapsto \mathcal{S}'$ returns a distribution over successor states given a state and a joint action. The global one-step reward R is the sum of local one-step rewards from each agent, i.e., $R_t(s_t, a_t) = \sum_{i=1}^{n} r_i(s_t, a_t)$. The objective is to find a global policy π_θ to maximize the discounted return G: $G = \sum_{t=1}^{T} \gamma^t R_t(s_t, a_t)$, where γ is the discount factor.

In this work, we formulate the resource allocation problem as a MAMDP, where n agents cooperatively regulate and allocate the resources to meet the demands. In such problems, the global state is the distribution of resources and demands, and other intrinsic features of the target area; joint action is the allocation strategies decided jointly by all agents over the target area; and global reward is task-specific global benefit after utilizing the resources. Note that the number of agent does not necessarily equals the number of grids K, where the design of how n agents cover all grids can be task-specific.

2.3 Actor-Critic Algorithm

Actor-Critic (AC) methods [24], which leverages advantages from both value-based [32] and policy-based [39] methods, is a typical solution to MAMDP. AC methods include two estimators: a critic V_{π_θ} and an actor π_θ. The critic V_{π_θ} plays the role of the value-based method by estimating the value of the current state during training. It aims to minimize the TD error δ_t to precisely estimate the value of the current state:

$$\delta_t = (\gamma r(s_t, a_t) + V_{\pi_\theta}(s_{t+1}) - V_{\pi_\theta}(s_t))^2. \tag{1}$$

The actor π_θ plays the role of the policy-based method via interacting with the environment and generating actions according to the current policy. It utilizes an advantage function A_{π_θ} to make π_θ update more stable than policy gradient methods [43]:

$$A_{\pi_\theta}(s_t, a_t) = r(s_t, a_t) + V_{\pi_\theta}(s_{t+1}) - V_{\pi_\theta}(s_t), \tag{2}$$

and the actor π_θ is updated through $J(\theta)$:

$$\nabla J(\theta) = \mathbb{E}[\nabla_\theta log \pi_\theta(s_t, a_t) A_{\pi_\theta}(s_t, a_t)]. \tag{3}$$

In this work, we mainly proposed DyPS based on AC algorithms to solve the spatio-temporal resource allocation problem.

3 METHODS

3.1 System Overview

An overview of DyPS's architecture is presented in Figure 2. DyPS is comprised of three modules: Self-supervised Role Modeling, Group Selection, and Group-based Resource Allocation. For spatio-temporal resource allocation problems, agents demonstrate both similarities and differences. We employ VLSTM to capture the characteristics of individual agents and employ CVAE to encode the functionalities of different groups. Based on the behavioral characteristics extracted by VLSTM and CVAE, the Group Selection Module dynamically groups agents. The Group-based Resource Allocation Module contains multiple Resource Allocation Policy Networks, each corresponding to a specific resource allocation pattern. The hierarchical decision structure enables rich behavioral patterns for making decisions on different resources. Simultaneously, it achieves parameter sharing to reduce training costs by sharing behavioral patterns.

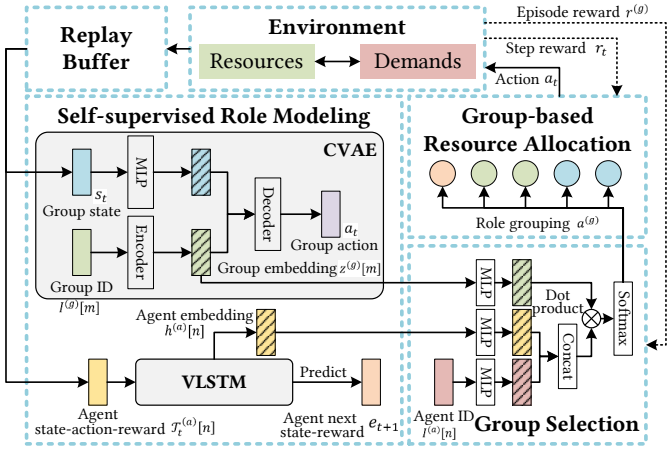


Figure 2: Overview of DyPS's architecture, including a Self-supervised Role modeling part and a hierarchical MARL part (Group Selection Module and Group-based Resource Allocation Module).

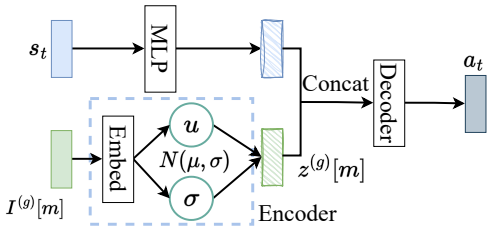


Figure 3: Structure of CVAE.

3.2 Self-supervised Role modeling

To effectively cluster agents into groups and determine which agents share parameters, role modeling for both groups and individual agents is essential. We employ two distinct self-supervised methods to model group roles and individual agent roles respectively, as follows:

3.2.1 CVAE for Group Role modeling. Agents within the same group share the same policy network. Modeling the role of each group is crucial for effectively clustering agents. Existing research [12, 41] demonstrated that the behavioral patterns of a policy network can be modeled through the state-action pairs obtained from the interaction between the agent and the environment. Considering that in some states, different policy networks may make identical decisions, therefore the same state-action pair could correspond to several policy networks. Therefore, we employ CVAE to encode the behavioral patterns of different policy networks, as shown in Figure 3. CVAE simultaneously learns a probabilistic encoder $q(z^{(g)}[m]|I^{(g)}[m];\theta^e)$ and a probabilistic decoder $p(a_t|z^{(g)}[m],s_t;\theta^d)$, where $I^{(g)}[m]$ denotes the identification of the Resource Allocation Policy Network m, s_t and a_t are the state of resources and the action output by the policy network, and $z^{(g)}[m]$ is the encoding of role of the group m. Different from the classical structure of autoencoders, s_t bypasses the encoder and can only be received by the decoder.

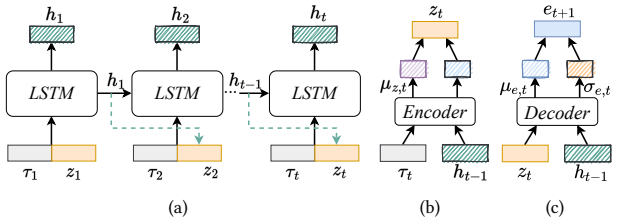


Figure 4: Structure of VLSTM. (a) is the feed-foward process of LSTM in VLSTM, (b) is the inference model and (c) is the generative model.

Based on this design, $z^{(g)}[m]$ only contains the information of the policy itself. We set the prior $p(z^{(g)}[m];\theta^p)$ to be the standard multivariate Gaussian, and the learning of CVAE is to maximize the evidence lower bound ELBO like VAE,

$$ELBO_c = \mathrm{E}_{q(z^{(g)}[m]|I^{(g)}[m];\theta^e)} \log p(a_t|z^{(g)}[m];\theta^d) - D_{KL}\left[q(z^{(g)}[m]|I^{(g)}[m];\theta^e)||p(z^{(g)}[m];\theta^p)\right]. \quad (4)$$

3.2.2 VLSTM for Agent Role modeling. In this section, we model the roles of agents from trajectory histories using VLSTM. Recent studies [11, 20, 29] have shown that the trajectories of agents contain spatio-temporal information. Extracting this information can help the Group Selection Agent precisely select groups for agents. Therefore, we utilize VLSTM [15] to extract the spatio-temporal features from the trajectory histories of resources, which combines the Variational Autoencoder (VAE) [23] with LSTM to enhance its robustness [8] in dynamic environments (*e.g.*, real-world ride-hailing) [15].

As shown Figure 4, the VLSTM model contains an inference model and a generative model. Trajectory history of agent n is denoted as $\mathcal{T}_t^{(a)}[n] = \{s_0, a_0, r_1, s_1, , \cdots, s_{t-1}, a_{t-1}, r_t\}$. In the following formulas, for simplicity, we omit agent identification. For example, we use $\mathcal{T}_t$ instead of $\mathcal{T}_t^{(a)}[n]$. VLSTM module can learn to encode complicated sequential features of $\mathcal{T}_t$ with a stochastic latent variable z_t. The generation model p_θ predicts the state-reward pair $e_{t+1} = (s_{t+1}, r_{t+1})$ of given the its internal states h_{t-1} as shown in Figure 4(c),

$$[\mu_{p,t}, \sigma_{p,t}^2] = f^p(h_{t-1}), \quad (5)$$

$$z_t \sim N(\mu_{p,t}, \sigma_{p,t}^2), \quad (6)$$

$$[\mu_{e,t}, \sigma_{e,t}^2] = f^d(z_t, h_{t-1}), \quad (7)$$

$$e_{t+1} \mid z_t \sim N(\mu_{e,t}, \sigma_{e,t}^2), \quad (8)$$

where f_p are parameterized feed-forward neural networks, and h_{t-1} is the hidden state variable of the LSTM contains historical spatial-temporal information, which can be recurrently updated with this formula and is shown in Figure 4(a),

$$h_t = f^e(h_{t-1}; z_t, s_{t-1}, a_{t-1}, r_t). \quad (9)$$

where f^e is a LSTM layer.

The inference model q_ϕ of VLSTM approximates the latent variable z_t given state s_t and hidden state variable h_{t-1} as shown in Figure 4(b),

$$[\mu_{z,t}, \sigma^2_{z,t}] = \phi^e(s_t, h_{t-1}), \tag{10}$$

$$z_t \mid s_t \sim N(\mu_{z,t}, \sigma^2_{z,t}). \tag{11}$$

The learning of VLSTM is to maximize the evidence lower bound ELBO like VAE [23],

$$ELBO_v = \sum_{t=1}^{T}[-D_{KL}(q_\phi(z_t|z_{1:t-1}, s_{1:t-1})||p_\theta(z_t|z_{1:t-1}, s_{1:t})) + E_{q_\phi(z_t|z_{1:t-1}, s_{1:t})}[\log(p_\theta(s_t|z_{1:t-1}, s_{1:t-1}))], \tag{12}$$

Finally, we append the identity id of each agent. we denote the hidden state encoded by complete trajectory histories $\mathcal{T}_T^{(a)}[n]$ of agent n as $h^{(a)}[n]$. Then $h^{(a)}[n]$ is further served as role representation of agent n to help the decision-making of Group Selection Agent.

3.3 Group Selection Module

This module is designed to determine the group of each agent, i.e., to decide its corresponding behavioral pattern by selecting the policy network for resource allocation. In this decision problem, the Group Selection Module needs to choose an appropriate resource allocation policy network based on the extracted role of resource allocation policy networks and the spatio-temporal behavior of agents. At the time step $t = 0$, once a resource allocation policy network is chosen, the agent is bound to the selected group until the end of the episode.

The detailed setting of the MAMDP of the Group Selection Module is as follows.

- **State:** The state of Group Selection Module is defined as $s^{(g)} = (I^{(a)}[n], h^{(a)}[n], z^{(g)}[m])$, which contains the identity of agent $I^{(a)}[n]$, extracted role $h^{(a)}[n]$ of agent n by VLSTM and role information $z^{(g)}[m]$ of group m modelled by CVAE.
- **Action:** The action of the Group Selection Module $a^{(g)}$ is determined by the state $s^{(g)}$ and is executed at the start of each episode. As shown in Figure 2, the $a^{(g)}$ can be calculated by

$$a^{(g)}[m, n] = \text{Softmax}(\left[I^{(a)}[n], h^{(a)}[n]\right] W z^{(g)}[m]), \tag{13}$$

  where $a^{(g)}$ denotes the probability matrix and the item in the m-th row and the n-th column of the matrix denotes the probability that the n-th agent is assigned to the m-th parameter group.
- **State Transition:** This is a one-step MAMDP. Once an action is executed, the next action decision is made only after the episode concludes.
- **Reward:** The reward $r^{(g)}$ is defined as the episode rewards among all agents in the Resource Allocation Module.

3.4 Group-based Resource Allocation Module

This module consists of multiple agents divided into several groups, with each group representing a resource allocation policy network. The network parameter set of group m is denoted as ψ_m. Then the parameter set of the Resource Allocation Module is $\{\psi_1, \psi_2 \cdots \psi_M\}$. For example, after the Group Selection Module takes action $a^{(g)}[m, n]$ and assigns a group m for agent n, this module supplies policy network ψ_m to the agent n. We denote $a_t^{(a)}[n]$ as action of agent n, which is obtained by $\pi(s_t^{(a)}[n]; \psi_m)$ with policy network ψ_m. Subsequently, agents address spatio-temporal resource allocation problems using Actor-Critic methods.

Algorithm 1 The training algorithm of DyPS

1: Initialize parameters for VLSTM and CVAE. Initialize number of Resource Allocation Agent with N and number of groups with M. Initialize parameters for Group Selection Module and policy networks in each group with φ and $\{\psi_1, \psi_2, \cdots, \psi_M\}$.
2: **for** *episodes* $k = 0, 1, 2, \cdots Max_{Epi}$ **do**
3: Initialize a set of transitions $D = \{\tau_i\}$.
4: Receive a state for Group Selection Module $s^{(g)}$ via VLSTM and CVAE for episode k.
5: Obtain action $a^{(g)}$ using Eq. (13).
6: Assign Group-based Resource Allocation Agents to resources based on $a^{(g)}$.
7: **for** $t = 0, 1, 2, ...T$ **do**
8: Obtain allocation action a_t.
9: Execute action a_t, obtain r_t and s_{t+1} from environment.
10: Store transition (s_t, a_t, s_{t+1}, r_t) into D.
11: Let $s_t = s_{t+1}$.
12: **end for**
13: Obtain reward $r^{(g)}$.
14: Update Group Selection Agent using the transition $(s^{(g)}, a^{(g)}, r^{(g)})$ via AC method.
15: Using set of trajectories D for batch gradient updating.
16: Update VLSTM by maximizing Eq. (12).
17: Update CVAE by maximizing Eq. (4).
18: Update Group-based Resource Allocation Agents via AC method.
19: **end for**

3.5 Training Algorithm

We summarize the training process of DyPS in Algorithm 1. As we can observe, firstly, DyPS interacts with the environment and collects a series of transitions by storing them in the set of transition D in preparation for training (lines 3-12). Then, these transition batches from D are utilized for training. VLSTM and CVAE are trained by maximizing ELBO (lines 15-17). The Group Selection Module and Group-based Resource Allocation Module are trained via AC method (line 14 and line 18). This process will be repeated for Max_{Epi} episodes until all the above modules of DyPS converge.

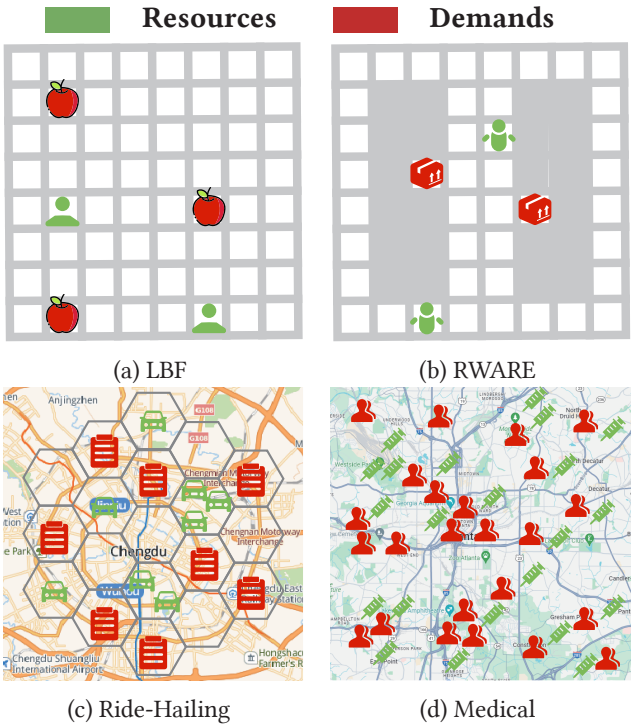
4 EXPERIMENTS

4.1 Experimental Scenarios

To thoroughly verify the effectiveness of DyPS, we set up 7 different experiments in 4 scenarios of spatio-temporal resource allocation by varying the number of resources and the number of needs in each environment. These scenarios cover several representative aspects of urban life and production, including transportation, medical care,

and labor resources. The total number of resources and demands over all steps of these scenarios are summarized in Table 1, and the details of each scenario are as follows:

- **Level-based foraging (LBF).** As shown in Figure 5a, LBF is a scenario where workers in a grid world are required to forage randomly scattered food [1, 2]. Here, resources are the workers, and demands are the food-foraging tasks. In our MARL solution, each resource is controlled by one agent, where the agents can move in four directions, trying to find food. Agents with different levels forage corresponding levels of foods. State of this task is the food distribution in adjacent grids, and we measure the performance via total return, i.e., the total number of foraged foods.
- **Multi-robot warehouse (RWARE).** As shown in Figure 5b, RWARE is a scenario that simulates robots moving goods in a warehouse [1, 44]. Here, resources are the robots and demands are the moving requests of goods. Various robots move different types of goods. In our MARL solution, each resource is controlled by one agent, where the agents can move in four directions, moving goods from the requested shelves to the goal posts. State of this task is the goods distribution in adjacent grids, and the current moving request. We measure the performance via total return, i.e., the total number of successfully delivered goods. Besides, constraint exists in this scenario that each robot can only move specific matched type of goods.
- **Order dispatch in on-demand ride-hailing services (Ride-Hailing).** As shown in Figure 5c, Ride-Hailing is a real-world scenario where drivers move to meet passengers' orders [21, 28, 40]. The environment is built based on the real-world data in Chengdu from DiDi [40], a famous online hailing platform. Here, resources refer to available drivers, while demands correspond to passenger ride orders. In our MARL solution, resources in each grid are controlled by one agent, where each agent can dispatch resources in its grid to target grids. Each agent controls one region, leading to heterogeneity due to diverse resource demands and supplies across cities. State of this task is the information of drivers and orders in the grids, and we measure the performance by the total value of orders served (GMV) and order response rate (ORR).
- **COVID-19 vaccines allocation (Medical)** As shown in Figure 5d, this is a real-world scenario that requires allocating a limited number of COVID-19 vaccines among urban population to minimize infection [5, 17]. The environment is built based on the real-world pandemic spreading data in Atlanta [5]. Here, resources are COVID-19 vaccines and demands are susceptible people around the city. In our MARL solution, resources in each grid are controlled by one agent, where each agent can dispatch resources in its grid to target grids. Each agent controls one region, leading to heterogeneity due to diverse resource demands and supplies across cities. State of this task is defined by the change in the number of infections within the grids, and we measure the performance by the reduced number of infections.



(a) LBF (b) RWARE

(c) Ride-Hailing (d) Medical

Figure 5: Illustration of experimental scenarios. (a) Level-based foraging (LBF). (b) Multi-robot warehouse (RWARE). (c) Order dispatch in on-demand ride-hailing services (Ride-Hailing). (d) COVID-19 vaccines allocation (Medical)

Table 1: Statistics of experimental scenarios.

Scenario	#Grids	#Resources	#Demands	#Agents
LBF (1)	100	6	4	6
LBF (2)	100	12	10	12
RWARE (1)	200	8	4	8
RWARE (2)	200	16	8	16
Ride-Hailing (1)	36	360	30000	36
Ride-Hailing (2)	100	1000	70000	100
Medical	3130	4.53M	7.19M	3130

4.2 Performance Evaluation

To verify the performance of DyPS we conduct spatio-temporal resource allocation experiments across 7 scenarios, comparing the following baseline parameter sharing methods:

- **NoPS.** In this baseline, all agents have their own parameters, and there is no overlap of gradients. This method retains the spatio-temporal differences of each agent but does not consider the spatio-temporal commonality of agents, which is common in the literature and usually compared with other parameter sharing methods, e.g., IPPO [9].
- **PS.** In this baseline, all agents share one set of parameters. It assumes that agents have similar spatio-temporal commonality but ignores the differences of each agent, which can be usually seen in the literature, e.g., MAPPO [47].

Table 2: Performance comparison over all baselines in six scenarios. Each value denotes the mean and standard deviation over 5 runs with different seeds. Bold numbers are the best in each column, and underlined numbers indicate sub-optimal value in each column.

Scenario	LBF (1)	LBF (2)	RWARE (1)	RWARE (2)	Ride-Hailing (1)		Ride-Hailing (2)		Medical
Metric	Return	Return	Return	Return	GMV	ORR	GMV	ORR	Infection Reduction
NoPS	0.31±0.02	0.91±0.05	2±0.15	0.35±0.12	230680±2550	0.40±0.02	519200±4250	0.20±0.01	253±24
PS	0.25±0.03	0.83±0.04	2.5±0.22	1.21±0.15	250940±2230	0.43±0.01	542500±4140	0.23±0.02	345±43
PS-id	0.52±0.02	0.91±0.05	4.6±0.20	10.45±1.15	260590±2150	0.45±0.03	556320±3970	0.24±0.01	378±55
PSA	0.41±0.03	0.90±0.03	4.5±0.18	12.11±2.12	253920±2080	0.42±0.03	547500±4270	0.22±0.03	365±68
SePS	0.61±0.04	0.96±0.03	6±0.15	33.21±2.50	257690±2210	0.43±0.02	545300±4210	0.23±0.02	452±72
DyPS (Ours)	**0.68±0.02**	**0.98±0.04**	**6.7±0.20**	**35.41±2.31**	**318200±2240**	**0.66±0.02**	**713520±3480**	**0.35±0.02**	**539±78**

- **PS-id.** Compared with the baseline PS, this method adds the id of the agent to the state of the agent to distinguish the spatio-temporal differences between different agents. It is encountered very often in the literature [10, 13, 35].
- **PSA [45].** This method is based on the framework of actor-critic [4]. It maintains each agent an actor and a critic to model the spatio-temporal differences of each agent and assigns a shared critic to model the spatio-temporal commonality of agents. These two critics will work together to train the actor of each agent.
- **SePS [6].**This method analyzes the spatio-temporal features of the agents by pre-sampling the trajectory in the environment before RL learning, and then groups them based on the analysis and keeps this grouping unchanged during the training process of RL.

Experimental results are summarized in Table 2, where the mean and standard deviation are obtained over 5 runs with different seeds. The results show that DyPS significantly outperforms all baseline methods in all scenarios in terms of the task-specific evaluation metrics, with at most 31% improvement achieved in Ride-Hailing (2) scenario, a large-scale real-world resource allocation task. It can also be observed that the SOTA parameter sharing method, SePS, performs well in the LBF and RWARE environments but does not perform well in the Ride-Hailing scenarios with a larger scale. This may be because agents in the first two scenarios have relatively clear role division, which can be defined before RL training. On the contrary, agents' roles change during RL learning in larger-scale scenarios, which makes the pre-defined role grouping mismatch the trained agents, harming the performance. This stresses the necessity of dynamical adjusting of role grouping in our method.

4.3 Ablation Studies

To provide a comprehensive understanding of the key components of DyPS, we conduct a series of experiments to investigate the effect of different components.

- **w/o CVAE.** This view evaluates the effectiveness CVAE in modeling the role of each group of agents. It removes CVAE from DyPS and directly decides the group of each agent via MLP, which takes in representations obtained from VLSTM.
- **w/o VLSTM.** This view evaluates the effectiveness VLSTM in modeling the role of each agent. It removes VLSTM from DyPS and decides the group of each agent via inner production of agent IDs and CVAE representations.
- **w/o Repr.** This view removes both CVAE and VLSTM from DyPS, and decides the group of each agent via MLP, which only takes in agent IDs.

We report the results in two scenarios in Table 3. It illustrates that removing VLSTM will make DyPS hard to model the spatio-temporal differences and commonality among agents, thus degrading the performance. Also, removing CVAE makes it difficult for the group selection module to distinguish the role of each group, which damages the performance of DyPS. In a nutshell, each component of DyPS improves the parameter sharing performance, and the full version achieves the best.

Table 3: Results of ablation studies. Each value denotes the mean and standard deviation over 5 runs with different seeds. Bold numbers are the best in each column.

Scenario	Ride-Hailing (1)		Ride-Hailing (2)	
Metric	GMV	ORR	GMV	ORR
w/o CVAE	305750±2250	0.65±0.01	859430±4220	0.48±0.02
w/o VLSTM	301200±2100	0.64±0.02	865270±4010	0.49±0.04
w/o Repr.	290990±2190	0.62±0.03	849260±4190	0.46±0.03
DyPS (Ours)	**318200±2240**	**0.66±0.02**	**873810±4080**	**0.50±0.01**

4.4 Adaptability to Multiple MARL Algorithms

To verify the general adaptability of DyPS, we integrate our framework with various on-policy MARL algorithms. Specifically, we substitute the role-based resource allocation module with various MARL algorithms.

- **IPPO [9].** This is the approach we employ in our previous experiments. In IPPO, each agent optimizes its own returns independently, and the network parameters between different agents are separate.
- **CoPO [34].** This approach employs meta-learning to adjust the relationship between the local and global levels of agents. It shares a common set of network parameters among all agents.
- **MAPPO [47].** This method features a decentralized actor and centralized critic structure, with all agents sharing the same reward function and a common set of parameters.

We report the results in the Ride-Hailing (2) scenario in Table 4. When combined with different MARL methods, DyPS consistently enhances performance by employing dynamic parameter sharing,

thereby highlighting its robust scalability. It's notable that MAPPO exhibits subpar performance attributed to reward sharing across all agents, exacerbating issues with credit assignment and rendering it unsuitable for tackling problems with large-scale agents.

Table 4: Results of adaptability to multiple MARL algorithms. Each value denotes the mean and standard deviation over 5 runs with different seeds.

Scenario	Ride-Hailing (2)	
Metric	GMV	ORR
IPPO	519200±4250	0.20±0.01
DyPS+IPPO	713520±3480	0.35±0.02
CoPO	633410±2100	0.34±0.02
DyPS+CoPO	873810±4080	0.50±0.01
MAPPO	536300±21040	0.22±0.01
DyPS+MAPPO	591460±4705	0.34±0.02

4.5 Role Utilization Comparison

To validate the effectiveness of DyPS's role-based dynamic parameter sharing design, we compare it with different methods of role utilization.

- **Role-As-State.** This approach enhances state augmentation by integrating the agents' role representation, extracted by VLSTM, into the states for decision-making.
- **Rode [41].** This method also incorporates role discovery and dynamic role assignment, using roles to decompose the action space into sub-tasks and couple these sub-tasks with specific roles.

We report the results for the Ride-Hailing (2) scenario in Table 5. The results indicate that DyPS is the most effective role utilization method. Regarding Role-As-State, with role-based state augmentation, Role-As-State outperforms IPPO but is not as effective as DyPS. Although Role-As-State can differentiate agents, the variance among roles' strategies prevents a single set of parameters from fitting all strategies. In contrast, DyPS employs different groups of parameters, enabling more heterogeneous strategies. As for Rode, it surpasses DyPS-IPPO due to the capability of its advanced submodule QMIX. However, due to the generalization of DyPS, DyPS demonstrates superior performance when combined with advanced cooperative MARL methods like CoPO.

Table 5: Performance comparison over different role utilization methods. Each value denotes the mean and standard deviation over 5 runs with different seeds.

Scenario	Ride-Hailing (2)	
Metric	GMV	ORR
Role-As-State	573740±5310	0.26±0.01
Rode	781540±7830	0.40±0.02
IPPO	519200±4250	0.20±0.01
CoPO	633410±2100	0.34±0.02
DyPS+IPPO	713520±3480	0.35±0.02
DyPS+CoPO	873810±4080	0.50±0.01

4.6 Strategies Visualization and Explainability

The number of agent roles is a critical hyperparameter in DyPS. We illustrate the performance across different numbers of roles in the Ride-Hailing (2) scenario in Figure 6 (a). It reveals that the optimal number of roles is 4, each corresponding to distinct resource-demand distribution patterns: grid areas with high car and order volumes, high car volumes with low order volumes, low car volumes with high order volumes, and low car and order volumes.

To delve deeper, we visualize the evolution of role grouping during training. As shown in Figure 6 (b), each grid represents an agent, and agents with each color are within the same role group. During DyPS training, we visualize the initial phase, mid-training phase, and convergence phase to showcase group selection changes. Initially, group assignment is random. During Middle stage, DyPS starts grouping grids with similar order demand distributions as depicted in Appendix Figure 7. By Late Stage, DyPS associates comparable roles with grids having similar order demand and vehicle supply as depicted in Appendix Figure 8. This late-stage selection is logical, as ride-hailing strategies should consider the dynamic interplay between order demand and vehicle supply, not just order distribution alone.

5 RELATED WORKS

5.1 Spatio-Temporal Resource Allocation

Spatio-temporal resource allocation [3, 14, 37] refers to the regulation and allocation of urban resources across time and space in cities to meet people's needs. Spatio-temporal resource allocation optimization is beneficial to the upgrading and development of industries such as transportation [49], logistics [28, 30], and energy [27, 42] in the city, which has received widespread attention from researchers. Conventional research often utilizes operational research algorithms to optimize spatio-temporal resource allocation. Examples include generic model for urban parking resource allocation [49], convex optimization algorithm for the allocation of delay-sensitive traffic resources [27], and integrated optimization approach to allocate water resource for sustainable urban development [42].

However, these methods have limited capability in solving large-scale problems and often simplify the problem settings, making it difficult to apply them in real-world scenarios. In recent years, with the development of artificial intelligence, multi-agent deep reinforcement learning (MARL) based optimization of spatio-temporal resource allocation has achieved remarkable success. For instance, some researchers solve the problem of express pickup and delivery via MARL [30], and others propose a mean field MARL to achieve efficient ride-hailing order dispatching [28]. As the scale of real-world cities continues to increase, researchers are concerned about how to scale up MARL algorithms to meet real-world demand. Therefore, in this paper, we propose a hierarchical MARL approach to dynamically share parameters among multiple agents, thereby reducing the computational consumption and enabling MARL algorithms to work in real-world scenarios.

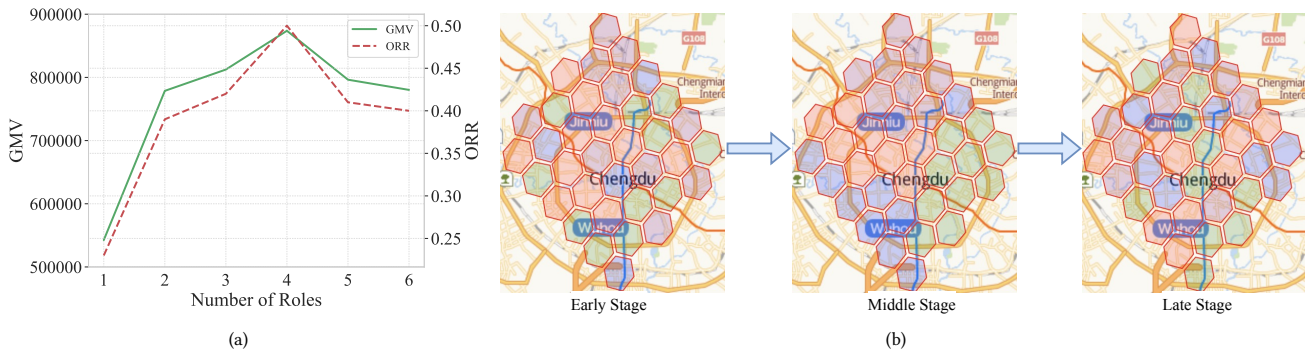


Figure 6: Strategies visualization and explainability of DyPS. (a) Performance with different numbers of roles. (b) Changing of role grouping during DyPS training.

5.2 Scaling up of MARL Algorithms

Typical MARL methods have limited performance when scaling up to real-world scenarios with a large number of agents. The main problem is that the joint action-observation space grows exponentially with the number of agents, which imposes high demand on the scalability of learning algorithms [26]. On the one hand, action decomposition is an alternative approach for reducing the complexity [16, 41, 46]. Mean-field algorithm [46] approximates the concatenation of actions with the unweighted average of these actions, which greatly reduces the dimension. However, such unweighted approximation discards the varying strengths of agent-agent interactions, losing precision in modeling the complex relations among the agents. To solve this problem, recent work develops the unweighted average into weighed one and utilizes graph attention mechanism to determine the weights, reaching better modeling of agent-agent interactions [16]. Rode [41] reduces complexity by decomposing the action space into sub-tasks and associating these sub-tasks with roles. However, the decomposition of the action space in Rode restricts it to discrete action spaces. Furthermore, it lacks a generalized design, making integration with other MARL algorithms challenging. In contrast, our proposed DyPS can handle both discrete and continuous action spaces and can seamlessly integrate with various on-policy MARL algorithms without requiring network structure modifications.

On the other hand, parameter sharing [7, 22, 25] among multi-agents is a widely used method to tackle the scalability challenge. It reduces learnable parameters of agents and improves sample efficiency, thereby facilitating the training of a large number of agents. However, naive parameter sharing [6] ignores the heterogeneity among agents, thus harming the optimization performance. Recent studies have proposed methods to enable selective parameter sharing among agents, thereby preserving the heterogeneous roles of agents. For example, SePS [6] analyzes the spatio-temporal features of the agents by pre-sampling the trajectory in the environment before RL learning, and then determines a fixed grouping of agents. As we mentioned before, agents' roles change during RL training, and pre-fixed role grouping may mismatch the well-trained agents, limiting the overall performance. Therefore, in this paper, we propose DyPS to dynamically adjust the parameter sharing among agents during RL training, achieving better performance than naive or pre-fixed parameter sharing strategies.

6 CONCLUSIONS

In this paper, we propose to solve the spatio-temporal resource allocation problem by an effective framework named DyPS. In this framework, we employ VLSTM and CVAE for self-supervised learning to extract the behavioral characteristics and thus identify the roles of agents from their action trajectories. We design a hierarchical MARL framework, which is trained according to the environmental feedback and can dynamically revise the parameter sharing among agents based on their roles. We conducted extensive experiments on various real-world scenarios, demonstrating that our method can effectively and dynamically share parameters among agents and achieve superior resource allocation performance compared with other baselines. Also, our framework can be adapted to various on-policy MARL algorithms, improving their performance on resource allocation tasks. Our design and in-depth experimental evaluations illustrate DyPS's great potential in practical applications, providing optimized solutions for the allocation of various resources in real-world scenarios. For future work, we aim to adapt DyPS to off-policy reinforcement learning algorithms, such as value decomposition methods.

ACKNOWLEDGMENTS

This work was supported in part by the National Natural Science Foundation of China under U22B2057 and U21B2036 and the National Key Research and Development Program of China under grant 2022ZD0116402.

REFERENCES

[1] Stefano V Albrecht and Subramanian Ramamoorthy. 2015. A game-theoretic model and best-response learning method for ad hoc coordination in multiagent systems. *arXiv preprint arXiv:1506.01170* (2015).

[2] Stefano V Albrecht and Peter Stone. 2019. Reasoning about hypothetical agent behaviours and their parameters. *arXiv preprint arXiv:1906.11064* (2019).

[3] Gowtham Atluri, Anuj Karpatne, and Vipin Kumar. 2018. Spatio-temporal data mining: A survey of problems and methods. *ACM Computing Surveys (CSUR)* 51, 4 (2018), 1–41.

[4] Mohammad Babaeizadeh, Iuri Frosio, Stephen Tyree, Jason Clemons, and Jan Kautz. 2016. Reinforcement learning through asynchronous advantage actor-critic on a gpu. *arXiv preprint arXiv:1611.06256* (2016).

[5] Lin Chen, Fengli Xu, Zhenyu Han, Kun Tang, Pan Hui, James Evans, and Yong Li. 2022. Strategic COVID-19 vaccine distribution can simultaneously elevate social utility and equity. *Nature Human Behaviour* 6, 11 (2022), 1503–1514.

[6] Filippos Christianos, Georgios Papoudakis, Muhammad A Rahman, and Stefano V Albrecht. 2021. Scaling multi-agent reinforcement learning with selective parameter sharing. In *International Conference on Machine Learning*. PMLR, 1989–1998.

[7] Xiangxiang Chu and Hangjun Ye. 2017. Parameter sharing deep deterministic policy gradient for cooperative multi-agent reinforcement learning. *arXiv preprint arXiv:1710.00336* (2017).

[8] Junyoung Chung, Kyle Kastner, Laurent Dinh, Kratarth Goel, Aaron Courville, and Yoshua Bengio. 2015. A recurrent latent variable model for sequential data. *arXiv preprint arXiv:1506.02216* (2015).

[9] Christian Schroeder de Witt, Tarun Gupta, Denys Makoviichuk, Viktor Makoviychuk, Philip HS Torr, Mingfei Sun, and Shimon Whiteson. 2020. Is independent learning all you need in the starcraft multi-agent challenge? *arXiv preprint arXiv:2011.09533* (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] Qiang Gao, Fan Zhou, Ting Zhong, Goce Trajcevski, Xin Yang, and Tianrui Li. 2022. Contextual Spatio-Temporal Graph Representation Learning for Reinforced Human Mobility Mining. *Information Sciences* (2022).

[12] Aditya Grover, Maruan Al-Shedivat, Jayesh Gupta, Yuri Burda, and Harrison Edwards. 2018. Learning policy representations in multiagent systems. In *International conference on machine learning*. PMLR, 1802–1811.

[13] Jayesh K Gupta, Maxim Egorov, and Mykel Kochenderfer. 2017. Cooperative multi-agent control using deep reinforcement learning. In *International conference on autonomous agents and multiagent systems*. Springer, 66–83.

[14] Ali Hamdi, Khaled Shaban, Abdelkarim Erradi, Amr Mohamed, Shakila Khan Rumi, and Flora D Salim. 2022. Spatiotemporal data mining: a survey on challenges and open problems. *Artificial Intelligence Review* 55, 2 (2022), 1441–1488.

[15] Dongqi Han, Kenji Doya, and Jun Tani. 2019. Variational recurrent models for solving partially observable control tasks. *arXiv preprint arXiv:1912.10703* (2019).

[16] Qianyue Hao, Wenzhen Huang, Tao Feng, Jian Yuan, and Yong Li. 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.

[17] Qianyue Hao, Wenzhen Huang, Fengli Xu, Kun Tang, and Yong Li. 2022. Reinforcement Learning Enhances the Experts: Large-scale COVID-19 Vaccine Allocation with Multi-factor Contact Network. In *Proceedings of the 28th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*. 4684–4694.

[18] Qianyue Hao, Fengli Xu, Lin Chen, Pan Hui, and Yong Li. 2021. Hierarchical reinforcement learning for scarce medical resource allocation with imperfect information. In *Proceedings of the 27th ACM SIGKDD Conference on Knowledge Discovery & Data Mining*. 2955–2963.

[19] Xiaohui Huang, Jiahao Ling, Xiaofei Yang, Xiong Zhang, and Kaiming Yang. 2023. Multi-Agent Mix Hierarchical Deep Reinforcement Learning for Large-Scale Fleet Management. *IEEE Transactions on Intelligent Transportation Systems* (2023).

[20] Shenggong Ji, Zhaoyuan Wang, Tianrui Li, and Yu Zheng. 2020. Spatio-temporal feature fusion for dynamic taxi route recommendation via deep reinforcement learning. *Knowledge-Based Systems* 205 (2020), 106302.

[21] Jiarui Jin, Ming Zhou, Weinan Zhang, Minne Li, Zilong Guo, Zhiwei Qin, Yan Jiao, Xiaocheng Tang, Chenxi Wang, Jun Wang, et al. 2019. Coride: joint order dispatching and fleet management for multi-scale ride-hailing platforms. In *Proceedings of the 28th ACM International Conference on Information and Knowledge Management*. 1983–1992.

[22] Meha Kaushik, K Madhava Krishna, et al. 2018. Parameter sharing reinforcement learning architecture for multi agent driving behaviors. *arXiv preprint arXiv:1811.07214* (2018).

[23] Diederik P Kingma, Max Welling, et al. 2019. An introduction to variational autoencoders. *Foundations and Trends® in Machine Learning* 12, 4 (2019), 307–392.

[24] Vijay R Konda and John N Tsitsiklis. 2000. Actor-critic algorithms. In *Advances in neural information processing systems*. 1008–1014.

[25] Fabian Konstantinidis, Ulrich Hofmann, Moritz Sackmann, Jörn Thielecke, Oliver De Candido, and Wolfgang Utschick. 2021. Parameter Sharing Reinforcement Learning for Modeling Multi-Agent Driving Behavior in Roundabout Scenarios. In *2021 IEEE International Intelligent Transportation Systems Conference (ITSC)*. IEEE, 1974–1981.

[26] Chenghao Li, Tonghan Wang, Chengjie Wu, Qianchuan Zhao, Jun Yang, and Chongjie Zhang. 2021. Celebrating diversity in shared multi-agent reinforcement learning. *Advances in Neural Information Processing Systems* 34 (2021), 3991–4002.

[27] Jian Li, Mugen Peng, Aolin Cheng, Yuling Yu, and Chonggang Wang. 2014. Resource allocation optimization for delay-sensitive traffic in fronthaul constrained cloud radio access networks. *IEEE Systems Journal* 11, 4 (2014), 2267–2278.

[28] Minne Li, Zhiwei Qin, Yan Jiao, Yaodong Yang, Jun Wang, Chenxi Wang, Guobin Wu, and Jieping Ye. 2019. Efficient ridesharing order dispatching with mean field multi-agent reinforcement learning. In *The world wide web conference*. 983–994.

[29] Yexin Li, Yu Zheng, and Qiang Yang. 2018. Dynamic bike reposition: A spatio-temporal reinforcement learning approach. In *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*. 1724–1733.

[30] Yexin Li, Yu Zheng, and Qiang Yang. 2020. Cooperative multi-agent reinforcement learning in express system. In *Proceedings of the 29th ACM International Conference on Information & Knowledge Management*. 805–814.

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

[32] Volodymyr Mnih, Koray Kavukcuoglu, David Silver, Alex Graves, Ioannis Antonoglou, Daan Wierstra, and Martin Riedmiller. 2013. Playing atari with deep reinforcement learning. *arXiv preprint arXiv:1312.5602* (2013).

[33] Jianjun Ni, Minghua Liu, Li Ren, and Simon X Yang. 2013. A multiagent Q-learning-based optimal allocation approach for urban water resource management system. *IEEE Transactions on Automation Science and Engineering* 11, 1 (2013), 204–214.

[34] Zhenghao Peng, Quanyi Li, Ka Ming Hui, Chunxiao Liu, and Bolei Zhou. 2021. Learning to simulate self-driven particles system with coordinated policy optimization. *Advances in Neural Information Processing Systems* 34 (2021), 10784–10797.

[35] Tabish Rashid, Mikayel Samvelyan, Christian Schroeder, Gregory Farquhar, Jakob Foerster, and Shimon Whiteson. 2018. Qmix: Monotonic value function factorisation for deep multi-agent reinforcement learning. In *International conference on machine learning*. PMLR, 4295–4304.

[36] Sijie Ruan, Jie Bao, Yuxuan Liang, Ruiyuan Li, Tianfu He, Chuishi Meng, Yanhua Li, Yingcai Wu, and Yu Zheng. 2020. Dynamic public resource allocation based on human mobility prediction. *Proceedings of the ACM on interactive, mobile, wearable and ubiquitous technologies* 4, 1 (2020), 1–22.

[37] Shashi Shekhar, Zhe Jiang, Reem Y Ali, Emre Eftelioglu, Xun Tang, Venkata MV Gunturi, and Xun Zhou. 2015. Spatiotemporal data mining: A computational perspective. *ISPRS International Journal of Geo-Information* 4, 4 (2015), 2306–2338.

[38] Kihyuk Sohn, Honglak Lee, and Xinchen Yan. 2015. Learning structured output representation using deep conditional generative models. *Advances in neural information processing systems* 28 (2015).

[39] Richard S Sutton, David A McAllester, Satinder P Singh, Yishay Mansour, et al. 1999. Policy gradient methods for reinforcement learning with function approximation.. In *NIPs*, Vol. 99. Citeseer, 1057–1063.

[40] Xiaocheng Tang, Fan Zhang, Zhiwei Qin, Yansheng Wang, Dingyuan Shi, Bingchen Song, Yongxin Tong, Hongtu Zhu, and Jieping Ye. 2021. Value Function is All You Need: A Unified Learning Framework for Ride Hailing Platforms. In *Proceedings of the 27th ACM SIGKDD Conference on Knowledge Discovery & Data Mining*. 3605–3615.

[41] Tonghan Wang, Tarun Gupta, Anuj Mahajan, Bei Peng, Shimon Whiteson, and Chongjie Zhang. 2020. Rode: Learning roles to decompose multi-agent tasks. *arXiv preprint arXiv:2010.01523* (2020).

[42] Fangliang Wei, Xiang Zhang, Jing Xu, Jianping Bing, and Guoyan Pan. 2020. Simulation of water resource allocation for sustainable urban development: An integrated optimization approach. *Journal of cleaner production* 273 (2020), 122537.

[43] Ronald J Williams. 1992. Simple statistical gradient-following algorithms for connectionist reinforcement learning. *Machine learning* 8, 3-4 (1992), 229–256.

[44] Peter R Wurman, Raffaello D'Andrea, and Mick Mountz. 2008. Coordinating hundreds of cooperative, autonomous vehicles in warehouses. *AI magazine* 29, 1 (2008), 9–9.

[45] Ning Yang, Bo Ding, PeiChang Shi, and Dawei Feng. 2022. Improving scalability of multi-agent reinforcement learning with parameters sharing. In *2022 IEEE International Conference on Joint Cloud Computing (JCC)*. IEEE, 37–42.

[46] Yaodong Yang, Rui Luo, Minne Li, Ming Zhou, Weinan Zhang, and Jun Wang. 2018. Mean field multi-agent reinforcement learning. In *International conference on machine learning*. PMLR, 5571–5580.

[47] 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.

[48] Chao Yu, Akash Velu, Eugene Vinitsky, Yu Wang, Alexandre Bayen, and Yi Wu. 2021. The surprising effectiveness of ppo in cooperative, multi-agent games. *arXiv preprint arXiv:2103.01955* (2021).

[49] Mahdi Zargayouna, Flavien Balbo, and Khadim Ndiaye. 2016. Generic model for resource allocation in transportation. Application to urban parking management.

Table 6: Hyper-parameters in experiments.

Hyper-parameter	LBF (1)	LBF (2)	RWARE (1)	RWARE (2)	Ride-Hailing (1)	Ride-Hailing (2)	Medical
Training steps	1M	1M	1M	1M	30k	30k	0.1M
Batch size	256	256	256	256	1024	1024	512
Optimizer	Adam	Adam	Adam	Adam	Adam	Adam	Adam
Actor learning rate (Group Selection)	1e-5	1e-5	1e-5	1e-5	1e-4	1e-4	1e-5
Critic learning rate (Group Selection)	1e-5	1e-5	1e-5	1e-5	1e-4	1e-4	1e-5
Actor learning rate (Resource Allocation)	5e-4	5e-4	5e-4	5e-4	1e-3	1e-3	1e-4
Critic learning rate (Resource Allocation)	5e-4	5e-4	5e-4	5e-4	1e-3	1e-3	1e-4
Discount factor	0.99	0.99	0.99	0.99	0.97	0.97	0.99
Clip threshold (PPO)	0.2	0.2	0.2	0.2	0.2	0.2	0.2
Lambda factor (PPO)	0.95	0.95	0.95	0.95	0.95	0.95	0.95

Figure 7: Order distribution heat map in the scenario of Ride-Hailing (1), the darker the color, the higher the order quantity.

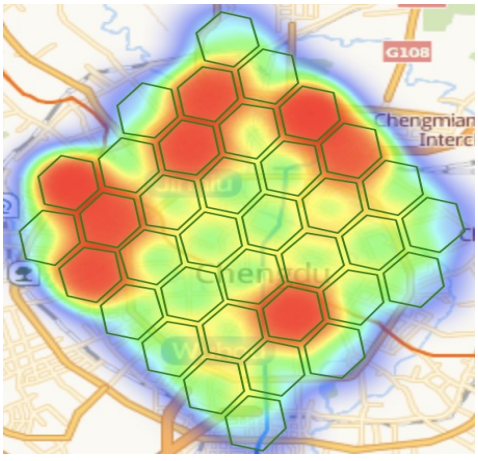


Figure 8: Resource-demand ratio (number of orders/number of drivers) heat map in the scenario of Ride-Hailing (1), the darker the color, the greater the supply-demand ratio.

Table 7: Results of rule-based methods in ride-hailing scenario.

Scenario	Ride-Hailing (1)		Ride-Hailing (2)	
Metric	GMV	ORR	GMV	ORR
Greedy	220770	0.37	539450	0.23
Nearest	162590	0.37	345180	0.24

Transportation Research Part C: Emerging Technologies 71 (2016), 538–554.

[50] Bolong Zheng, Lingfeng Ming, Qi Hu, Zhipeng Lü, Guanfeng Liu, and Xiaofang Zhou. 2022. Supply-demand-aware deep reinforcement learning for dynamic fleet management. *ACM Transactions on Intelligent Systems and Technology (TIST)* 13, 3 (2022), 1–19.

[51] Limei Zhou, Mingtian Fan, and Zuping Zhang. 2009. A study on the optimal allocation of emergency power supplies in urban electric network. In *CIRED 2009-20th International Conference and Exhibition on Electricity Distribution-Part 1*. IET, 1–4.

A APPENDIX

A.1 Details for Reproducibility

In this paper, we use a simple network structure to maintain the scalability and generality of our framework. Regarding the CVAE module, the encoder network is an MLP with a 64-dimensional hidden layer, and the decoder network is an MLP with two 64-dimensional hidden layers. The latent variable of the CVAE has a dimensionality of 16. For the VLSTM module, the hidden dimension of the LSTM is 64. The encoder network is an MLP with a 64-dimensional hidden layer, and the decoder network is an MLP with two 64-dimensional hidden layers.

We perform experiments using Python 3.9 and Pytorch 1.11 with NVIDIA GeForce RTX 3090 and NVIDIA A100 GPUs. Here, we provide detailed values of the hyper-parameters used in the experiments for reproducibility in Table 6.

A.2 Additional Information of the Ride-Hailing Scenario

The distributions of orders and drivers are significant in ride-hailing scenario. In Figure 7 and 8, we visualize order distribution and resource-demand distribution in the scenario of Ride-Hailing (1). The heat maps illustrate the uneven spatial distribution of resources

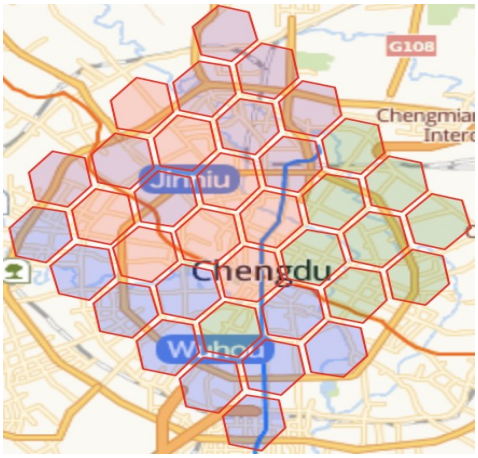


Figure 10: Visualization of the role grouping result of SePS method in the scenario of Ride-Hailing (1), where each color indicates one group of agents.

and the mismatched distribution between resources and demands, indicating the importance of efficient resource allocation strategies.

To gain a more intuitive understanding of the ride-hailing scenario, We compare two common rule-based methods in the ride-hailing scenario: (1) Greedy, which prioritizes high-priced orders, and (2) Nearest, which prioritizes the nearest orders. The results are shown in Table 7.

A.3 Details in Training Process

We draw the learning curves of DyPS and all baselines on scenario Ride-Hailing (2) in Figure 9. It illustrates that both the training speed and final effect of our method are much better than all baselines.

In addition, we can also observe that the training speed of NoPS is slower than the other baselines that consider parameter sharing, which verifies the effectiveness and efficiency of parameter sharing.

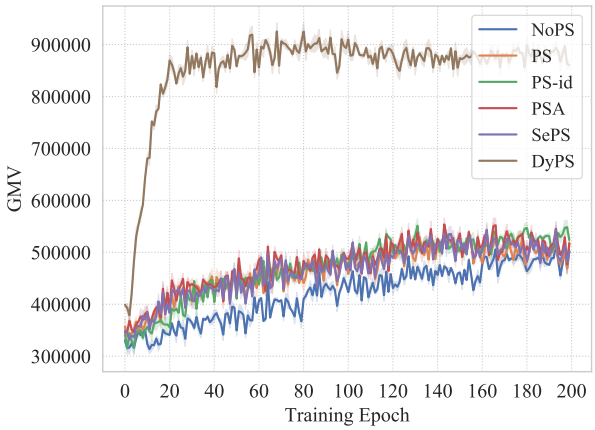


Figure 9: Learning curves showing the mean GMVs during training for a selection of the environments on scenario Ride-Hailing (2). The shaded area represents standard deviation across 5 seeds.

A.4 Extended Visualizations

It can be observed from Figure 10 that in the scenario of Ride-Hailing (1), SePS has learned to group agents with spatial adjacency into a role. However, since SePS fixes the role grouping before training, it fails to adjust the role grouping according to the learning of the resource allocation agent's strategy.