2023 62nd IEEE Conference on Decision and Control (CDC)
December 13-15, 2023. Marina Bay Sands, Singapore

Multi-Agent Deep Reinforcement Learning for Large-scale Platoon Coordination with Partial Information at Hubs

Dixiao Wei, Peng Yi and Jinlong Lei

Abstract— **This paper considers the hub-based platoon coordination problem in a large-scale transportation network, to promote cooperation among trucks and optimize the overall efficiency of the transportation network. We design a distributed communication model for transportation networks and transform the problem into a Dec-POMDP (Decentralized-Partial Observable Markov Decision Process). We then propose an A-QMIX deep reinforcement learning algorithm to solve the problem, which adopts centralized training and distributed execution and hence provides a reliable model for trucks to make quick decisions using only partial information. Finally, we carry out experiments with 100 trucks in the transportation network of the Yangtze River Delta region in China to demonstrate the effectiveness of the proposed algorithm.**

I. Introduction

Vehicle platooning is a technology in intelligent transportation networks that allow vehicles to travel closely together in a column, while maintaining a fixed relative distance between each other [1]. It can achieve efficient operation of the transportation system and increase the capacity of the transportation system [2], especially in heavy transportation industries such as truck transport. In addition, vehicle platooning can reduce fuel consumption and carbon emissions [3], [4].

In the past few decades, truck platooning technology has been continuously and widely investigated through coordinated control to achieve the stable formation of multiple trucks, see e.g., [5], [6]. With the developments of platoon control technology [7] and vehicle-to-vehicle communication technology [8], it is expected that a large-scale truck platoon transportation network will be formed in future. Therefore, platoon coordination in large-scale transportation networks is an active and challenging research topic [9], which allows trucks to obtain the benefits of platooning as much as possible in the transportation network. As far as we know, the investigated platoon coordination schemes include the vehicle speed planning [10], the path selection [11], and the regulation of waiting and departure times at hubs [12].

This work will focus on regulating the waiting time of trucks at hubs in a large-scale transportation network. There exist some works proposing centralized schemes to resolve the problem. For example, [13], [14] are dedicated to promoting truck cooperation at hubs in limited-scale transportation networks to save fuel and improve traffic efficiency. On the other hand, several other works consider the problem of non-cooperative platoons of trucks at hubs in a large-scale transportation network. The authors of [15] considered platoon coordination for a group of trucks with the same starting point but different ending points at the departure hub. Then [12] expanded it to a group of trucks with different starting and ending points with a centralized method. However, centralized algorithms will lead to a heavy computational burden in large-scale transportation networks and affect the timeliness of the algorithm.

There exist some other works proposing distributed algorithms to schedule the waiting time of trucks at hubs in large-scale transportation networks. The authors of [16] proposed a distributed MPC framework to regulate the waiting time for trucks at hubs, and used an event-triggered mechanism to calculate the waiting time for each truck to maximize its benefits in future. They further presented a distributed approximate dynamic programming scheme to solve the problem of waiting time at hubs that complies with Hours-of-Service regulations [17]. The aforementioned works mainly focus on distributed algorithms under non-cooperative relationships that might lead to a worse system-level performance than cooperative relationships in a large-scale transportation network.

Therefore, this paper focuses on proposing a distributed scheme merely using partial information to resolve the cooperative truck platoon coordination problem in large-scale transportation networks. Each truck is treated as an intelligent agent, hence this problem is transformed into a large-scale multi-agent cooperative task, which however is difficult to rapidly achieve a distributed resolution. It is worth noting from [18] that deep reinforcement learning can effectively solve large-scale cooperative multi-agent problems with high-dimensional state space while merely using partial information. Therefore, based on multi-agent deep reinforcement learning, this work proposes a distributed cooperative framework that adopts a centralized training and distributed independent execution method to address the truck platoon coordination problem with partial information in large-scale transportation networks. Centralized training concentrates the experience data of all agents into a centralized training model to improve the learning efficiency of agents. Then, in the distributed execution, each agent can independently use the trained model to make distributed decisions based only on partial observation information. The main contributions are

The authors are with the Department of Control Science and Engineering, Tongji University, Shanghai, 201804, China; and the Shanghai Research Institute for Intelligent Autonomous Systems, Tongji University, Shanghai, 201210, China. E-mails: {weidx, yipeng, leijinlong}@tongji.edu.cn

The paper was sponsored by the National Key Research and Development Program of China under No 2022YFA1004701, the National Natural Science Foundation of China under Grant No. 62003239, No. 62003240, No. 72271187 and No. 62373283, and partially by Shanghai Municipal Science and Technology Major Project No. 2021SHZDZX0100.

summarized as follows.

- We design a distributed communication model for the large-scale transportation network, where a single truck can only receive partial information about nearby trucks for deciding its waiting time at hubs.
- We propose an A-QMIX deep reinforcement learning algorithm for the muti-agent cooperative task in the large-scale transportation network. A-QMIX combines the truck attention block to promote truck cooperation, which can be distributedly executed through centralized training, enabling each truck to independently solve the scheduling problem with partial information.
- We set up simulation experiments with 100 trucks in the Yangtze River Delta transportation network in China. In comparison with the centralized schemes, our method can achieve similar performance to the centralized algorithm with merely using partial information.

The rest of the paper is organized as follows. Section II describes the modeling process of a distributed framework and Dec-POMDP. Section III introduces detailed information on A-QMIX. Section IV shows the experimental results in the transportation network. Section V concludes this paper and outlines future research directions.

II. Problem Formulation

This section introduces a large-scale distributed transportation network architecture and the process of transforming the platoon coordination at hubs problem into Dec-POMDP. We first describe the dynamic model of trucks in the transportation network as follows.

A. Dynamical Model in Transportation Network

We consider the platoon coordination problem based on the transportation network $G = (V, E)$. Here, V represents the set of hubs where trucks can wait and form platoons with other trucks, while E represents the set of edges that connect the hubs in V. The trucks in the transportation network are represented by $\mathcal{N} = \{1, 2, \ldots, N\}$.

For each truck $i \in \mathcal{N}$ in the transportation network, there is a starting point g_i, an endpoint f_i, an originally scheduled departure time d_i, a fixed set of hubs $\mathcal{V}_i = \{i_1, i_2, \ldots, i_{N_i}\} \subseteq V$, and a fixed path $\mathcal{E}_i = \{(i_1, i_2), (i_2, i_3), \ldots, (i_{N_i-1}, i_{N_i})\} \subseteq E$ from the starting point $g_i = i_1$ to the endpoint $f_i = i_{N_i}$. For $k \in \{1, 2, \ldots, N_i\}$, we denote i_k as the k-th hub on the journey of truck i. Meanwhile, for $k \in \{1, 2, \ldots, N_i - 1\}$, $(i_k, i_k + 1)$ represents the directed edge of truck i from hub i_k to the next hub i_{k+1}. The starting point $g_i = i_1$ is indexed by the 1-th hub while the ending point $f_i = i_{N_i}$ is indexed by the N_i-th hub.

Each truck can wait at a hub for a certain amount of time before departing to form a platoon with other trucks. As shown in Fig. 1, for every truck $i \in \mathcal{N}$ at hub i_{k+1}, its departure time is defined as

$$t_i^d(i_{k+1}) = t_i^a(i_{k+1}) + \tau_i^w(i_{k+1}), \tag{1}$$

where $t_i^a(i_{k+1})$ denotes the arrival time of truck i at hub i_{k+1} and $\tau_i^w(i_{k+1}) \in \mathbb{Z}_+$ is the waiting time of truck i at hub i_{k+1}. In particular, $t_i^d(i_1) = d_i + \tau_i^w(i_1)$. The arrival time of truck i at hub i_{k+1} is given by

$$t_i^a(i_{k+1}) = t_i^d(i_k) + \tau_i(i_k, i_{k+1}), \tag{2}$$

where $\tau_i(i_k, i_{k+1}) \in \mathbb{N}$ represents the travel time of truck i on edge (i_k, i_{k+1}).

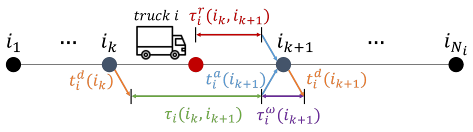

Fig. 1. Truck i in the transportation network

Similarly to [16] considering the travel time as a known constant, we also assume that each truck $j \in \mathcal{N}$ takes the same known travel time $\tau_j(i_k, i_{k+1})$ on edge (i_k, i_{k+1}).

Platoon Definition: For two trucks $i, j \in \mathcal{N}$, $i \neq j$, if there exist $k \in \{1, 2, \ldots, N_i\}$ and $k' \in \{1, 2, \ldots, N_j\}$ such that $(i_k, i_{k+1}) = (j_{k'}, j_{k'+1})$ and $t_i^d(i_k) = t_j^d(j_{k'})$. Then truck i and truck j form a platoon on edge (i_k, i_{k+1}).

Each truck decides its waiting time at the hub to form a platoon. And the reward of the truck depends on whether it can accurately calculate the waiting time at the hub to form a platoon. But in a large-scale transportation network, due to some uncertain factors, it is hard to obtain accurate global information to calculate the waiting time of long-distance hubs. Therefore, we will propose a distributed architecture in the following subsection to provide partial information for truck decision-making.

B. Distributed Communication in Transportation Network

We first denote the state of the transportation network by $S = \{s_1, s_2, \ldots, s_N\}$. For every truck $i \in \mathcal{N}$, we define s_i as a two-tuple (s_i^1, s_i^2). When truck i is at hub i_k, $k \in \{1, 2, \ldots, N_i\}$, we denote by $s_i^1 = i_k$ and let s_i^2 be the time that the truck i has already waited at hub i_k. When truck i is traveling on edge (i_k, i_{k+1}), we denote by $s_i^1 = (i_k, i_{k+1})$ and let s_2^i represent the remaining travel time of truck i to hub i_{k+1}. Thus, s_i defined is as follows.

$$(s_i^1, s_i^2) = \begin{cases} (i_k, \tau_i^{aw}(i_k)), & \text{truck } i \text{ at hub } i_k, \\ ((i_k, i_{k+1}), \tau_i^r(i_k, i_{k+1})), & \\ & \text{truck } i \text{ traveling on edge } (i_k, i_{k+1}). \end{cases} \tag{3}$$

Here, $\tau_i^{aw}(i_k) \in \mathbb{N}$ denotes the time that truck i has already waited at hub i_k, while $\tau_i^r(i_k, i_{k+1}) \in \mathbb{N}$ represents the remaining travel time of truck i from hub i_k to i_{k+1}.

In the transportation network, each truck $i \in \mathcal{N}$ can communicate with some hub $h_i(s_i)$ defined below.

$$h_i(s_i) = \begin{cases} i_k, & s_i^1 = i_k, \\ i_{k+1}, & s_i^1 = (i_k, i_{k+1}). \end{cases} \tag{4}$$

For example, as shown in Fig. 2, the orange truck i located at hub i_k will communicate with hub i_k, while the blue truck

j driving on edge $(j_{k'}, j_{k'+1})$ will communicate with hub $j_{k'+1}$. The communicated information of truck i through hub $h_i(s_i)$ is denoted by $M_i(s_i) = (M_i^1(s_i), M_i^2(s_i))$, where $M_i^1(s_i)$ denotes the next hub of the hub $h_i(s_i)$ that truck i is going to, and $M_i^2(s_i) = s_i^2$. Thus, $M_i(s_i)$ is defined by

$$M_i(s_i) = (M_i^1(s_i), M_i^2(s_i)) \begin{cases} (i_{k+1}, s_i^2), & s_i^1 = i_k, \\ (i_{k+2}, s_i^2), & s_i^1 = (i_k, i_{k+1}). \end{cases} \tag{5}$$

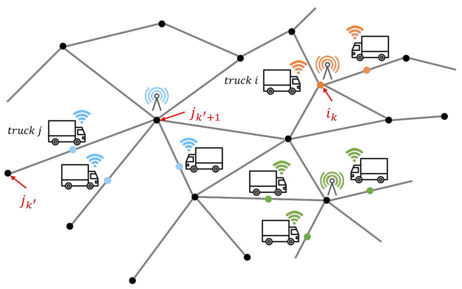

Fig. 2. Distributed communication in transportation network

The set of trucks that has the same communication hub $h_i(s_i)$ as truck i is defined as

$$\mathcal{C}_i(s_i) = \{j : h_j(s_j) = h_i(s_i), j \in \mathcal{N}, j \neq i\}. \tag{6}$$

Meanwhile, the partner set of truck i, representing the set of trucks that may form a platoon with truck i, is denoted as

$$\mathcal{P}_i(s_i) = \{j : j \in \mathcal{C}_i(s_i), M_i^1(s_i) = M_j^1(s_j)\}. \tag{7}$$

When truck i is at hub $h_i(s_i)$, i.e., $s_i = i_k$ for some $i_k \in \mathcal{V}_i$, it has an observation function to obtain information about nearby trucks, which can be described as

$$O_i(s_i) = \{s_j : j \in \mathcal{P}_i(s_i)\}. \tag{8}$$

Specifically, when truck i is driving towards its destination i_{N_i}, it no longer communicates with hub i_{N_i}.

Under this distributed framework, we transform the problem of scheduling truck wait time at hubs into a multi-agent cooperative problem. It will be reformulated as a Dec-POMDP as shown in the next subsection, which can be solved using deep reinforcement learning.

C. Dec-POMDP Formulation

The Dec-POMDP problem is composed of a tuple $G = (N, S, A, T, R, Z, O, \gamma)$ [19]. Here, $N = |\mathcal{N}|$ represents the number of agents, and $\gamma \in [0, 1)$ represents the discount factor, which defines the relative importance of future rewards to current rewards. In the following, we will show G in details.

1) ***State Space:*** State space $S = \{s_1, s_2, \ldots, s_N\}$, where each $s_i, i \in \mathcal{N}$ is defined as (3).

2) ***Action Space:*** The joint action space of all trucks is defined as $A = \{a_1, a_2, \ldots, a_N\}$. The action space a_i of truck i is represented as

$$a_i = \begin{cases} 0, & \text{truck } i \text{ waits at a hub,} \\ 1, & \text{truck } i \text{ departs from a hub,} \\ 2, & \text{truck } i \text{ travels on an edge.} \end{cases} \tag{9}$$

Specifically, if truck i is traveling on an edge, it can only take the action $a_i = 2$, i.e., if $s_i = (i_k, i_{k+1})$, then $a_i = 2$.

3) ***State Transition Function:*** $T(S, A) : S \times A \to S'$ represents the function that S transfers to S' after performing the joint action A. We define Δt to be the time step for the state space to change from S to S'.

Since each truck's decision is independent of the others, changing its actions will not affect the state of other trucks. So, the transfer function T_i of truck i depends only on the state s_i and the action a_i. When $s_i^1 = (i_k, i_{k+1})$, namely, truck i is traveling on edge (i_k, i_{k+1}), $a_i = 2$, (i) if the remaining travel time $s_i^2 > \Delta t$, then at the next stage, the truck i is still traveling on edge (i_k, i_{k+1}), while the remaining travel time is decreased to $s_i^2 - \Delta t$; (ii) if the remaining travel time $s_i^2 \leq \Delta t$, since Δt is small enough, it can be considered that the truck has arrived at the hub i_{k+1} at the next stage and the waiting time is 0. When $s_i^1 = i_k$, namely, truck i is at hub i_k, (i) if $a_i = 0$, then the truck will wait at the hub for Δt, hence at the next stage, the truck i is still located at hub i_k while the waiting time is increased to $s_i^2 + \Delta t$; (ii) if $a_i = 1$, then the truck i departs from hub i_k, hence at the next stage, the truck i traveling on the edge (i_k, i_{k+1}) and the remaining travel time is $\tau_i(i_k, i_{k+1}) - \Delta t$. In summary, the transfer function T_i of truck i after performing action a_i can be denoted as

$$T_i(s_i, a_i) = \begin{cases} (s_i^1, s_i^2 - \Delta t), & s_i^1 = (i_k, i_{k+1}) \wedge s_i^2 > \Delta t, \\ (i_{k+1}, 0), & s_i^1 = (i_k, i_{k+1}) \wedge s_i^2 \leq \Delta t, \\ (s_i^1, s_i^2 + \Delta t), & s_i^1 = i_k \wedge a_i = 0, \\ ((i_k, i_{k+1}), \tau_i(i_k, i_{k+1}) - \Delta t), & \\ & s_i^1 = i_k \wedge a_i = 1. \end{cases} \tag{10}$$

In particular, if truck i arrives at its ending point i_{N_i} (i.e., $s_i^1 = i_{N_i}$), its state $s_i = (i_{N_i}, 0)$ will not change.

4) ***Observation Space and Observation Function:*** Observation space is denoted by $Z = \{z_1, z_2, \ldots, z_N\}$. According to (8), when truck i is at hub i_k, $z_i = O_i(s_i)$. And when truck i is traveling on an edge, $z_i = \varnothing$.

5) ***Reward:*** Our goal is to form as many platoons as possible while reducing waiting time. Extensive experiments have shown that the fuel savings for followers in a platoon are roughly the same, and leaders almost do not save fuel [20][21]. Therefore, we distribute the fuel benefits obtained by the platoon equally among each truck in the platoon.

We let c_u represent the fuel profit per time step Δt while truck i is driving in a platoon. If truck i is at hub i_k (i.e., $s_i^1 = i_k$) and will depart from i_k, we let $\mathcal{N}_i^p(\cdot)$ represent the set of trucks that will form a platoon with truck i (formally defined in (11)). Its reward is $\frac{c_u(|\mathcal{N}_i^p(i_k)|-1)}{|\mathcal{N}_i^p(i_k)|}$. When truck i is traveling on the edge (i_k, i_{k+1}) (i.e., $s_i^1 = (i_k, i_{k+1})$), we let $\mathcal{N}_i^p((i_k, i_{k+1}))$ represent the set of trucks that formed a platoon with truck i (formally defined in (12)). Its reward is computed as $\frac{c_u(|\mathcal{N}_i^p(i_k, i_{k+1})|-1)}{|\mathcal{N}_i^p(i_k, i_{k+1})|}$.

$$\mathcal{N}_i^p(i_k) = \{j : j \in \mathcal{N}, s_j^1 = i_k \wedge t_i^d(i_k) = t_j^d(s_j^1)\}, \tag{11}$$

$$\mathcal{N}_i^p((i_k, i_{k+1})) = \{j : j \in \mathcal{N}, s_j^1 = (i_k, i_{k+1}) \wedge s_i^2 = s_j^2\}. \tag{12}$$

Let $C_i(i_k)$ denote the reward of truck i for waiting in hub i_k, which will be defined in (14). Thus, for every time step Δt, the reward function of truck i is defined as follows.

$$R_i(s_i, a_i) = \begin{cases} \frac{c_u(|\mathcal{N}_i^p(i_k, i_{k+1})|-1)}{|\mathcal{N}_i^p(i_k, i_{k+1})|}, & s_i^1 = (i_k, i_{k+1}), \\ -C_i(i_k), & s_i^1 = i_k \wedge a_i = 0, \\ \frac{c_u(|\mathcal{N}_i^p(i_k)|-1)}{|\mathcal{N}_i^p(i_k)|}, & s_i^1 = i_k \wedge a_i = 1. \end{cases} \tag{13}$$

Let $w_{total}(i_k) = \sum_{x=0}^{x=k} \tau_i^{aw}(i_x)$ represent the sum of the waiting time for truck i to pass through the hub. Taking into consideration the potential loss of benefits caused by delays resulting from trucks waiting at hubs, we give a maximum waiting time w_{bound} for trucks and set the loss function for truck i as follows.

$$C_i(i_k) = \begin{cases} c_w, & w_{total}(i_k) \le w_{bound} \\ c_{bound}, & w_{total}(i_k) > w_{bound} \end{cases} \tag{14}$$

Here, c_w is the penalty factor for waiting, and c_{bound} is a large penalty factor for exceeding the waiting time limit.

The total reward of the whole transportation network is:

$$R = \sum_{i=1}^{N} R_i(s_i, a_i). \tag{15}$$

III. A-QMIX Algorithm for Platoon Coordination

This section elaborates on the operational process of applying multi-agent reinforcement learning algorithms to the formulated problem.

A. Transformation for State and Observation

Neural networks usually require fixed-length inputs. S-ince the dimensions of $S = \{s_1, s_2, \ldots, s_N\}$ and $Z = \{z_1, z_2, \ldots, z_N\}$ are determined by the number of trucks, they do not apply to the neural network. Considering the variability of the transportation network scale, we transform the state and the observation into a fixed size.

For each edge $e^m \in E$, we discretize the travel time $\tau(e^m)$ into α segments as follows.

$$\bar{\tau}(e^m) = (\tau(e^m[1]), \tau(e^m[2]), \ldots, \tau(e^m[\alpha])), \tag{16}$$

where

$$\tau(e^m[j]) = \left(\frac{\alpha - j}{\alpha}\tau(e^m), \frac{\alpha + 1 - j}{\alpha}\tau(e^m)\right], \quad j \in \{1, 2, \ldots, \alpha\}.$$

We let the state of edge e^m be the number of trucks driving on each segment of the edge, i.e.,

$$\mathcal{S}^m = \{|\mathcal{M}(e^m[1])|, |\mathcal{M}(e^m[2])|, \ldots, |\mathcal{M}(e^m[\alpha])|\}. \tag{17}$$

Here, $\mathcal{M}(e^m[j]), j \in \{1, 2, \ldots, \alpha\}$ denotes the set of trucks driving on the edge e^m, for which the remaining travel time belongs to the time interval $\tau(e^m[j])$, i.e., $\mathcal{M}(e^m[j]) = \{i : i \in \mathcal{N}, s_i^1 = e^m \wedge s_i^2 \in \tau(e^m[j])\}$. Specifically, when truck i is at the starting hub of edge e^m, we let $i \in \mathcal{M}(e^m[1])$. Hence the state of all edges is denoted as $\mathcal{S} = (\mathcal{S}^1, \mathcal{S}^2, \ldots, \mathcal{S}^{|E|})$, which has a fixed size of $\alpha|E|$.

Likewise, for each truck i's observation at hub $h_i(s_i)$, we discretize the remaining travel time to hub $h_i(s_i)$ into β segments. We define $\lambda = \frac{w_{bound}}{\beta}$ as the time of each segment. Let $\mathcal{O}^i(q)$ denote the truck $o, o \in \mathcal{P}_i$ of which the remaining travel time belongs to the time interval $(\lambda(q-1), \lambda q]$, i.e.,

$$\mathcal{O}^i(q) = \{o : o \in \mathcal{P}_i, s_o^2 \in (\lambda(q-1), \lambda q]\}, \quad q \in \{1, 2, \ldots, \beta\}. \tag{18}$$

The new observation of truck i is denoted as

$$\mathcal{Z}^i = \{|\mathcal{O}^i(0)|, |\mathcal{O}^i(1)|, \ldots, |\mathcal{O}^i(\beta)|\}, \tag{19}$$

where $\mathcal{O}^i(0)$ represents the set of the partners $o, o \in \mathcal{P}_i$ at hub $h_i(s_i)$. When truck i is traveling on an edge, $\forall q \in \{1, 2, \ldots, \beta\}, |\mathcal{O}^i(q)| = 0$. Hence, the new observation $\mathcal{Z} = \{\mathcal{Z}^1, \mathcal{Z}^2, \ldots, \mathcal{Z}^{|\mathcal{N}|}\}$ has a fixed size of $(\beta + 1)|\mathcal{N}|$.

B. A-QMIX for Large-scale Platoon Coordination

QMIX is a well-known deep reinforcement learning algorithm to resolve multi-agent cooperative problems [22]. For a variable-scale transportation network, we will propose A-QMIX combined with the attention block [23] to solve the platoon problem.

A-QMIX consists of N individual agent networks, $\{\bar{Q}^i\}_{i=1}^N$ with parameters $\{\omega^i\}_{i=1}^N$, and an attention mixing network f with parameters ω_f. Similarly to QMIX, A-QMIX also employs the target network method for training, which has target agent networks $\{\bar{Q}'^i\}_{i=1}^N$ with parameters $\{\omega'^i\}_{i=1}^N$, and a target attention mixing network f' with parameters $\omega_{f'}$.

1) ***Agent Network:*** $\{\bar{Q}^i\}_{i=1}^N$ adopt DRQN (Deep Recurrent Q-Network) [24], which captures time dependencies between states through recurrent neural networks, enabling trucks to consider previous decision information when making decisions at hubs and handle continuous decision problems. We define $t \in \{1, 2, \ldots, T_e\}$ with time step Δt, where T_e represents a time constant long enough for the last truck to reach its ending point. At each t, $\bar{Q}^i$ takes $\mathcal{Z}_t^i$ and previous action a_{t-1}^i as input to calculate Q_t^i.

$$Q_t^i = \max_{a_i} \bar{Q}^i(\mathcal{Z}_t^i, a_{t-1}^i, a_i; \omega^i), \tag{20}$$

where a_{t-1}^i is the action taken by truck i at time $t-1$, a_i represents the action space of truck i defined in (9). Especially, $a_0^i = 0$. Q_t^i is the prediction of future rewards for truck i after acting.

Likewise, the calculation formula of $\bar{Q}'^i$ is denoted as

$$Q'^i_{t+1} = \max_{a_i} \bar{Q}'^i(\mathcal{Z}_{t+1}^i, a_t^i, a_i; \omega'^i). \tag{21}$$

2) ***Attention Mixing Network:*** Attention mixing network f takes $\mathcal{S}_t = \{\mathcal{S}_t^1, \mathcal{S}_t^2, \ldots, \mathcal{S}_t^{|E|}\}$ and $\mathbf{Q}_t = \{Q_t^1, Q_t^2, \ldots, Q_t^N\}$ as input and Q_{total_t} as output, which can be denoted as

$$Q_{total_t} = f(\mathcal{S}_t, \mathbf{Q}_t) = f_2(\mathbf{Q}_t, f_1(\mathcal{S}_t)). \tag{22}$$

Here, Q_{total_t} aims to measure the quality of the joint action space of multiple trucks at time t. In addition, f' has the same structure as f in (23).

$$Q'_{total_{t+1}} = f'(\mathcal{S}_{t+1}, \mathbf{Q}_{t+1}) = f'_2(\mathbf{Q}_{t+1}, f'_1(\mathcal{S}_{t+1})). \quad (23)$$

Moreover, attention mixing network f can be divided into f_1 and f_2. As shown in (24) and (25), f_1 aims to encode the state $\mathcal{S}_t$ at time t as weights.

$$f_1(\mathcal{S}_t) = (W_1, B_1, W_2, B_2). \quad (24)$$

$$(W_1, B_1, W_2, B_2) = (f_{11}(\mathcal{S}_t), f_{12}(\mathcal{S}_t), f_{13}(\mathcal{S}_t), f_{14}(\mathcal{S}_t)). \quad (25)$$

Here, $f_{11}(\cdot)$,$f_{12}(\cdot)$,$f_{13}(\cdot)$ represent the three independent linear layers. $f_{14}(\cdot)$ consists of a linear layer and a ReLU activation function.

Although $S \to \mathcal{S}$ makes the size of the state fixed, the information between trucks is lost. It is hard to find which trucks may form a platoon based on $\mathcal{S}$. From this, we combine the attention block [23] to let the network learn the weight of each segment explicitly and find the correlation between trucks. Specifically, the truck attention block is $W'_1 = f_{scale}(f_a(W_1), W_1) = W^a_1 \cdot W_1$, where $f_a(\cdot)$ is a linear layer equal in size to W_1. Fig. 3 illustrates the process of the truck attention block.

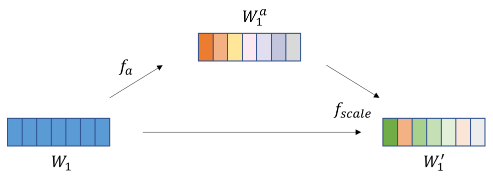

Fig. 3. Truck attention block

Equation (26) illustrates that f_2 multiplies the weights by $\mathbf{Q}_t$ calculated by the agent networks.

$$f_2(\mathbf{Q}_t, W_1, B_1, W_2, B_2) = |W_2| \cdot \mathrm{ELU}(|W'_1| \cdot \mathbf{Q}_t + B_1) + B_2, \quad (26)$$

where ELU$(\cdot)$ is the ELU activation function. $|\cdot|$ is the absolute activation function, which is to ensure W'_1 and W_2 are non-negative. By adding absolute value constraints to the weights, the contribution of each truck's action to the global joint action is reasonable (i.e., each truck tends to cooperate with the other).

C. Training Process

The entire training process can be divided into experience data generation and network training, which alternate in execution. We summarize the pseudo-code of A-QMIX in Algorithm 1.

*1) **Experience Data Generation**:* During experience data generation, each truck $i \in \mathcal{N}$ continuously makes decisions based on its current $\bar{Q}^i$ in the environment until all trucks reach the endpoint (complete an episode). The generated data is then stored in the experience replay buffer D. The process of experience data generation can be found in (Algorithm 1, lines 3-8).

Algorithm 1 Training Process of A-QMIX Algorithm

```
Input: $N$ trucks in the transport network, Replay buffer $\mathcal{D}$,
    episode length $T_e$, joint action space $\mathbf{a}$, batch size $B$,
    discount factor $\gamma$, target network update frequency $C$,
    greedy policy $\epsilon$.
Output: Agent-networks $\{\bar{Q}^i\}_{i=1}^N$ and attention mixing net-
    work $f$ characterized by $\{w^i\}_{i=1}^N$ and $w_f$ respectively.
 1: Initialize Agent-networks $\{\bar{Q}^i\}_{i=1}^N$ with random weights
    $\{w^i\}_{i=1}^N$, target Agent-networks $\{\bar{Q}'^i\}_{i=1}^N$ with weights
    $\{w'^i\}_{i=1}^N \leftarrow \{w^i\}_{i=1}^N$, attention mixing network $f$ with
    random weights $w_f$, target attention mixing network $f'$
    with weights $w_{f'} \leftarrow w_f$.
 2: for each training step do
 3:     for $t = 1$ to $T_e$ do
 4:         Obtain joint observation $\mathbf{z}_t$, last joint actions
    $\mathbf{a}_{t-1}$and state $s_t$.
 5:         Sample joint actions $\mathbf{a}_t$ from the Agent-networks
    with $\epsilon$-greedy policy.
 6:         Execute $\mathbf{a}_t$ and observe rewards $r_t$, next joint
    observation $\mathbf{z}_{t+1}$ and next state $s_{t+1}$.
 7:     end for
 8:     Store transition $(\mathbf{z}_{1:T_e+1}, \mathbf{a}_{0:T_e}, s_{1:T_e+1}, r_{1:T_e})$ in $\mathcal{D}$.
 9:     if $|\mathcal{D}| \geq B$ then
10:         Choose min-batch of transitions with B samples
    in $\mathcal{D}$.
11:         for $t = 1$ to $T_e$ do
12:             for $b = 1$ to $B$ do
13:                 Calculate each truck's local Q-values $\mathbf{Q}_t^b$
    and local target Q-values $\mathbf{Q}'^b_{t+1}$ by (20) and (21).
14:                 Calculate total Q-value $Q^b_{total_t}$ and target
    Q-value $Q'^b_{total_{t+1}}$ with $\mathbf{Q}^b_t$ and $\mathbf{Q}'^b_{t+1}$ by (22) and (23).
15:             end for
16:         end for
17:         Calculate A-QMIX loss by (27).
18:         Perform a gradient descent step on
    $\mathcal{L}(\{w^i\}_{i=1}^N, w_f)$ with respect to the parameters
    $\{w^i\}_{i=1}^N$ and $w_f$.
19:     end if
20:     Every $C$ train steps reset $\{w'^i\}_{i=1}^N \leftarrow \{w^i\}_{i=1}^N$,
    $w_{f'} \leftarrow w_f$.
21: end for
```

*2) **Network Training**:* After data generation, B episode data is randomly selected from the buffer for training. According to (20) and (21), for $t \in \{1, 2, \ldots, T_e\}$ and $b \in \{1, 2, \ldots, B\}$, the agent networks and the target networks calculate $\mathbf{Q}^b_t$ and $\mathbf{Q}'^b_{t+1}$ from the data. Then f and f' output the total Q-value Q_{total_t} and $Q'_{total_{t+1}}$ according to (22) and (23). The process can be found in (Algorithm 1, lines 9-14).

Finally, A-QMIX is trained by the following loss:

$$\mathcal{L}(\{\omega^i\}_{i=1}^N, \omega_f) = \sum_{b=1}^{B} (r_{1:T_e} + \gamma Q'^b_{total_{2:T_e+1}} - Q^b_{total_{1:T_e}})^2. \quad (27)$$

Here, $r_{1:T}$ represents the reward from time 1 to T_e, and γ represents the discount factor.

Distributed execute: For each truck i, the trained model $\bar{Q}^i$ is used to make decisions throughout the traveling journey. Algorithm 2 illustrates the process.

Algorithm 2 Execute Process of Single Truck

```
1: Initialize truck $i$'s agent-network $\bar{Q}^i$ with the trained weights $w^i$, state $s_i$, action space $a_i$, depeparture time $d_i$, a fixed set of hubs $\mathcal{V}_i = \{i_1, i_2, \dots, i_{N_i}\}$, $k \leftarrow 1$.
2: for $t = d_i$ to $T_e$ do
3:     Obtain observation $\mathcal{Z}_t^i$, last actions $a_{t-1}^i$and state $s_i$.
4:     $a_t^i = \underset{a_i}{argmax}\, \bar{Q}^i(\mathcal{Z}_t^i, a_{t-1}^i, a_i; \omega^i)$.
5:     if $s_i = i_k \wedge a_t^i = 1$ then
6:         $k \leftarrow k + 1$.
7:     end if
8:     $s_i \leftarrow T_i(s_i, a_t^i)$,
9: end for
```

IV. Experiment

This section will introduce experiments on the transportation network in the Yangtze River Delta region of China to demonstrate the effectiveness of the proposed method. As shown in Fig. 4, the transportation network covers Shanghai, Zhejiang, Jiangsu, and Anhui provinces, including 41 hubs with each hub corresponding to a city, and 202 directed edges. The transportation network data comes from Amap[1].

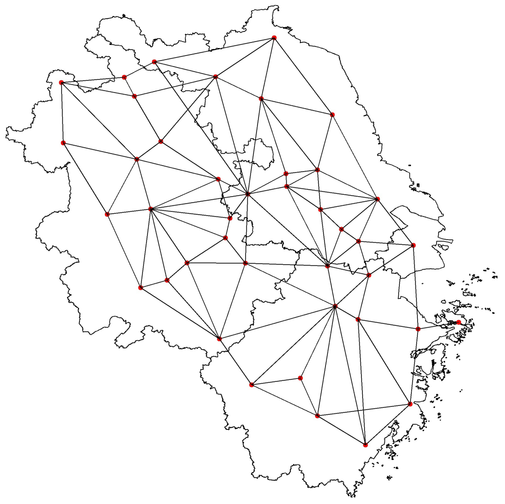
Fig. 4. Transportation Network Map of Yangtze River Delta Region

Experiment settings. We set $N = 100$ and assume that the starting and ending points of each truck are random and have the shortest fixed driving route. The departure time of 100 trucks is randomly selected between 5:00 am and 11:00 am. During the entire journey, we assume that the truck moves at a uniform speed of 80 km/h, and time step $\Delta t = 1$ minute. The longest waiting time is $w_{bound} = 20$ minutes. The platoon benefits variable and waiting time penalty variables are set as $c_u = 1.7$, $c_w = 4.4$, and $c_{bound} = 500$. We assume that the driving time is not affected by the distance between the starting and ending points. We set up a training set and a test set for training, and they both choose 100 randomly initialized values. The random initial values include the starting point, ending point, driving route, and departure time of 100 trucks.

[1]The city location and road information come from www.amap.com/

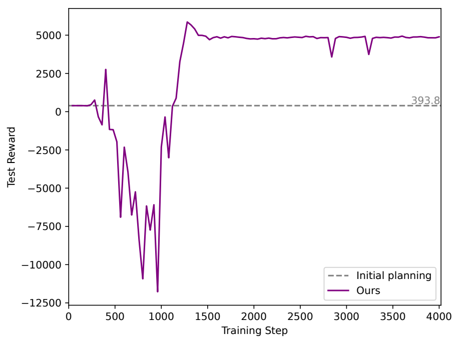

Fig. 5. Training process

In Algorithm 1, we set the number of training steps as 4000, the batch size $B = 32$, the experience replay buffer size $|\mathcal{D}| = 100$, the episode length $T = 1500$, $\gamma = 0.99$, and let the target network update every 50 epochs. We set the initial value of the greedy policy ϵ to be 1 and the minimum value as 0.05, and let it decrease by 0.00019 each time step. Each edge of the transportation network is divided into $\alpha = 10$ segments and observation parameters are set as $\lambda = 4$ minutes and $\beta = 5$.

A. Training Result

The training process of the model is shown in Fig. 5, where the horizontal axis represents the number of training steps, and the vertical axis represents the average reward obtained by testing the current model in the training set. The purple solid line represents the training performance of the model at different stages, where a test is conducted every 40 epochs. The grey dashed line represents the average reward obtained by the trucks that make no decision at hubs.

B. Comparison Experiment

We now compare our method with the Nash method [12], which considers each truck as a non-cooperative agent and uses potential games to calculate the Nash equilibrium solution for all trucks in the transportation network. This method essentially uses the potential game to solve the optimal solution at the decision-making moment of the transportation network. It is worth noting that our method merely used partial information for decision in comparison with global information utilized by the Nash method.

We implement the two methods under the same initial values, where the Nash method has a calculation interval of 5 minutes and a node number of $H = 1$ in [12]. The empirical rewards are displayed in Fig. 6. In addition, we compare the two methods on the test set for 100 initial values that has not been seen before and demonstrate the empirical results in Fig. 7. This indicates that our well-trained model is not sensitive to the initial values and can find decision results similar to the Nash method in the same scenario.

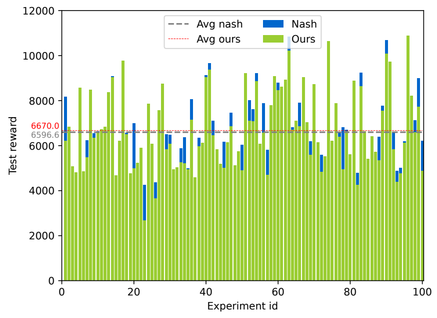

Fig. 6. Training set comparison results

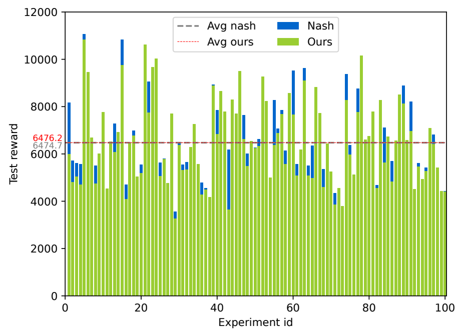

Fig. 7. Testing set comparison results

V. Conclusion

In this paper, we propose a distributed decision-making framework based on A-QMIX deep reinforcement learning that uses partial information to address the coordination problem in the large-scale transportation network. We then validate the effectiveness of the algorithm by simulating a transportation network with 100 trucks in the Chinese transportation map. It is of interest to expand the action space of the agents to combine multiple heterogeneous strategies, enabling simultaneous decisions on driving speed, driving path, and waiting time to further optimize the overall benefits of the transportation network.

References

[1] C. Bergenhem, S. Shladover, E. Coelingh, C. Englund, and S. Tsugawa, "Overview of platooning systems," in *Proceedings of the 19th ITS World Congress, Oct 22-26, Vienna, Austria (2012)*, 2012.

[2] T. Robinson, E. Chan, and E. Coelingh, "Operating platoons on public motorways: An introduction to the sartre platooning programme," in *17th world congress on intelligent transport systems*, vol. 1, 2010, p. 12.

[3] S. Tsugawa, "Results and issues of an automated truck platoon within the energy its project," in *2014 IEEE Intelligent Vehicles Symposium Proceedings*. IEEE, 2014, pp. 642–647.

[4] S. Tsugawa, S. Jeschke, and S. E. Shladover, "A review of truck platooning projects for energy savings," *IEEE Transactions on Intelligent Vehicles*, vol. 1, no. 1, pp. 68–77, 2016.

[5] B. Besselink and K. H. Johansson, "String stability and a delay-based spacing policy for vehicle platoons subject to disturbances," *IEEE Transactions on Automatic Control*, vol. 62, no. 9, pp. 4376–4391, 2017.

[6] G. Rödönyi, "An adaptive spacing policy guaranteeing string stability in multi-brand ad hoc platoons," *IEEE Transactions on Intelligent Transportation Systems*, vol. 19, no. 6, pp. 1902–1912, 2017.

[7] T. Chu and U. Kalabić, "Model-based deep reinforcement learning for cacc in mixed-autonomy vehicle platoon," in *2019 IEEE 58th Conference on Decision and Control (CDC)*. IEEE, 2019, pp. 4079–4084.

[8] Y. Wu, X. Fang, C. Luo, and G. Min, "Intelligent content precaching scheme for platoon-based edge vehicular networks," *IEEE Internet of Things Journal*, vol. 9, no. 20, pp. 20 503–20 518, 2022.

[9] L. Aarts and G. Feddes, "European truck platooning challenge," in *Proceedings of the HVTT14: International Symposium on Heavy Vehicle Transport Technology, Rotorua, New Zealand*, 2016, pp. 15–18.

[10] K.-Y. Liang, J. Mårtensson, and K. H. Johansson, "Heavy-duty vehicle platoon formation for fuel efficiency," *IEEE Transactions on Intelligent Transportation Systems*, vol. 17, no. 4, pp. 1051–1061, 2015.

[11] M. Abdolmaleki, M. Shahabi, Y. Yin, and N. Masoud, "Itinerary planning for cooperative truck platooning," *Transportation Research Part B: Methodological*, vol. 153, pp. 91–110, 2021.

[12] A. Johansson, E. Nekouei, K. H. Johansson, and J. Mårtensson, "Strategic hub-based platoon coordination under uncertain travel times," *IEEE Transactions on Intelligent Transportation Systems*, vol. 23, no. 7, pp. 8277–8287, 2021.

[13] R. Larsen, J. Rich, and T. K. Rasmussen, "Hub-based truck platooning: Potentials and profitability," *Transportation Research Part E: Logistics and Transportation Review*, vol. 127, pp. 249–264, 2019.

[14] W. Zhang, E. Jenelius, and X. Ma, "Freight transport platoon coordination and departure time scheduling under travel time uncertainty," *Transportation Research Part E: Logistics and Transportation Review*, vol. 98, pp. 1–23, 2017.

[15] A. Johansson, E. Nekouei, K. H. Johansson, and J. Mårtensson, "Multi-fleet platoon matching: A game-theoretic approach," in *2018 21st international conference on intelligent transportation systems (itsc)*. IEEE, 2018, pp. 2980–2985.

[16] T. Bai, A. Johansson, K. H. Johansson, and J. Mårtensson, "Event-triggered distributed model predictive control for platoon coordination at hubs in a transport system," in *2021 60th IEEE Conference on Decision and Control (CDC)*. IEEE, 2021, pp. 1198–1204.

[17] Bai, Ting and Johansson, Alexander and Johansson, Karl Henrik and Mårtensson, Jonas, "Approximate dynamic programming for platoon coordination under hours-of-service regulations," in *2022 IEEE 61st Conference on Decision and Control (CDC)*. IEEE, 2022, pp. 7663–7669.

[18] T. Li, K. Zhu, N. C. Luong, D. Niyato, Q. Wu, Y. Zhang, and B. Chen, "Applications of multi-agent reinforcement learning in future internet: A comprehensive survey," *IEEE Communications Surveys & Tutorials*, 2022.

[19] F. A. Oliehoek and C. Amato, *A concise introduction to decentralized POMDPs*. Springer, 2016.

[20] A. Davila, E. Del Pozo, E. Aramburu, and A. Freixas, "Environmental benefits of vehicle platooning," SAE Technical Paper, Tech. Rep., 2013.

[21] R. Bishop, D. Bevly, L. Humphreys, S. Boyd, and D. Murray, "Evaluation and testing of driver-assistive truck platooning: phase 2 final results," *Transportation research record*, vol. 2615, no. 1, pp. 11–18, 2017.

[22] T. Rashid, M. Samvelyan, C. S. De Witt, G. Farquhar, J. Foerster, and S. Whiteson, "Monotonic value function factorisation for deep multi-agent reinforcement learning," *The Journal of Machine Learning Research*, vol. 21, no. 1, pp. 7234–7284, 2020.

[23] J. Hu, L. Shen, and G. Sun, "Squeeze-and-excitation networks," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2018, pp. 7132–7141.

[24] M. Hausknecht and P. Stone, "Deep recurrent q-learning for partially observable mdps," *arXiv preprint arXiv:1507.06527*, 2015.