PairUpLight: A Multi-agent Reinforcement Learning Approach for Coordinated Multi-intersection Traffic Signal Control

Wenlu Du[*], Jing Li[†], and Guiling "Grace" Wang[†‡], *Fellow, IEEE*
[*]Department of Computer Science, Skidmore College, Saratoga Springs, NY, US
[†]Department of Computer Science, New Jersey Institute of Technology, Newark, NJ, US
wdu@skidmore.edu, {jingli, gwang}@njit.edu

***Abstract*—The management of heavy traffic demands has been significantly improved by employing synchronized traffic signal control at multiple intersections. Multi-agent Reinforcement Learning (MARL) techniques have been widely utilized to achieve this coordination. However, these approaches predominantly depend on manually crafted features from adjacent intersections, which impedes their generalization to new scenarios. Furthermore, while displaying high accuracy for specific traffic flow patterns, these methods often lack the necessary robustness for other patterns. In this study, our objective is to develop an effective signal timing plan by directly learning the minimal required communication between intersections from traffic data. We introduce a novel, comprehensive approach that combines multi-agent reinforcement learning with a learned communication mechanism. Our model incorporates a coordinated actor network and a centralized critic network to address the challenges of non-stationarity. We conducted extensive experiments comparing our model with other commonly used non-RL and benchmark MARL techniques. The evaluation results show that our proposed model, which relies only on local sensory input and a single message from neighboring intersections, excels in managing various traffic flow patterns. Furthermore, our model outperforms competing approaches in terms of robustness, resilience, and overall performance.**

***Index Terms*—Multi-agent Systems, Reinforcement Learning, Traffic Signal Control**

Code – https://github.com/Wenlu057/pairuplight

I. Introduction

The global concern over traffic congestion persists. The 2022 Global Traffic Scorecard[1] reveals that Boston, MA, saw a 72% increase in traffic delays, reaching 128 hours and ranking it as the second highest in the United States for congestion. The TomTom Traffic Index for 2023[2] identified London as the slowest city for drivers, underscoring the extensive impact of congestion on societal well-being and environmental health. Urban intersections are primary sites for daily congestion. Evidence indicates that coordinating traffic signal control at the network level significantly reduces congestion, highlighting the importance of Multi-intersection Traffic Signal Control (TSC) in urban traffic management strategies.

To enhance traffic management, Multi-Agent Reinforcement Learning (MARL) is increasingly used for joint optimization of multiple intersections. Each traffic signal controller acts as a Reinforcement Learning (RL) agent, adaptively adjusting signals based on real-time traffic conditions. MARL outperforms isolated single RL agents due to its ability to coordinate decision making across the network, as shown in Figure 1. It enables agents to learn and adapt not only to local traffic patterns but also to the dynamics of adjacent intersections. This collaborative learning approach results in more efficient overall traffic management, reducing congestion more effectively by considering the interconnected nature of urban traffic networks.

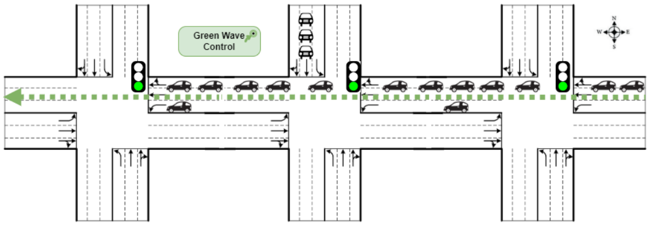


Fig. 1. Coordinated traffic signal control, all the east-to-west direction green.

However, just because of this interconnected nature of intersections, one significant challenge in MARL for TSC is the non-stationarity of the environment. This challenge originates from the scenario where individual agents, namely traffic signal controllers, undergo learning and signal adjustments in real-time. Such modifications can induce a cascading effect on other agents within the system, thereby complicating the learning process. This intricacy presents significant obstacles in achieving stable and optimal learning outcomes, necessitating that agents dynamically adapt not only to the evolving traffic patterns but also to the attributes of other interconnected agents within the network.

Some MARL-based TSC methods [1], [2] tackle the challenge of non-stationarity by incorporating data from intersections seen as having a certain temporal-spatial relationship with the current one (physically adjacent intersections in most cases). Techniques like graph neural networks [3] and attention

[‡] Correponding author.
[1] https://inrix.com/scorecard/
[2] https://www.tomtom.com/traffic-index/ranking/

mechanisms [4], among other advanced neural networks, are employed to distill relevant information from these agents. These embedded representations, combined with the agent's local observations, create a comprehensive state used as input for the vanilla reinforcement learning models (e.g., DQN [5], PPO [6]). However, accurately capturing the intricate dependencies between intersections (the "chain-rule effect") remains an unsolved challenge. There's no consensus or theoretical backing on what constitutes complete and essential input for these models. Furthermore, even if we can identify and correctly weight influential intersections, guaranteeing uniform impact is challenging due to variations in their geometrical configurations. Intuitively, the immediate neighbors have the most significant influence, but this is not always true. Real-world complexities, including differences in local intersection layouts like spacing and lane arrangements, introduce varying cascading effects, highlighting the difficulty in standardizing an approach for all scenarios.

Is there a more effective method for coordinating multiple intersections beyond merely integrating temporal-spatial relationships with others into the RL model's input? In this work, we advance towards this goal by leveraging the concept of how communication develops among intelligent agents [7], akin to how two people playing a game might cooperate through interaction. We propose the PairUpLight model, which introduces a communication protocol within the RL framework, enabling traffic control agents to exchange real-time congestion information. Specifically, at every timestep, incoming messages from one of the most congested upstream neighboring intersections are fed into the RL model. This model then generates outgoing messages for the next timestep's communication. This model minimizes dependence on specific intersection configurations and reduces the necessity for extensive data from neighboring intersections, effectively addressing the challenge of a non-stationary environment. By utilizing Proximal Policy Optimization (PPO) [6] with Generalized Advantage Estimation (GAE) [8] and parameter sharing, PairUpLight demonstrates enhanced performance and stability across both complicated synthetic and real-world traffic scenarios, showcasing its superiority over existing MARL methods in managing traffic congestion and dynamics.

In summary, the contributions of this work are as follows:

- We demonstrate the effectiveness of a message passing mechanism and propose PairUpLight, showcasing the benefits of a learned communication protocol in addressing congestion. Compared with other MARL methods with rich input information, the average travel time is lower in both synthetic and real-world datasets.

- We reveal that while current MARL-based methods achieve high accuracy for certain traffic flow patterns, they fall short in robustness across diverse traffic conditions. This lack of stability hinders their applicability in real-world scenarios.

- Our work is the first to evaluate on a 6×6 synthetic grid and a real-world dataset featuring a heterogeneous environment. We suggest the PPO with GAE framework for large-scale TSC applications.

II. Literature Review

A. Traditional Traffic Signal Control Methods

Traditional Traffic Signal Control (TSC) methods can be broadly categorized into two types: fixed-time control and actuated control [9]. Both types of control methods are primarily driven by predetermined signal timing parameters, offering limited adaptability to real-world traffic fluctuations. In contrast, modern adaptive signal control algorithms employ real-time detection data to dynamically adjust signal timing parameters in response to current traffic conditions. Within this category, many research studies have focused on improving traffic mobility using RL algorithms.

B. Reinforcement Learning in Traffic Signal Control

Some studies have applied Single-Agent Reinforcement Learning (SARL) for TSC at individual intersections. For instance, the study [10] introduced a model called 3DQN and proved it to be an effective SARL model by highlighting its superior performance. Despite their adaptability, these algorithms are only effective for isolated intersections, and their performance significantly deteriorates when applied to interconnected intersections with complex traffic dynamics.

A growing body of literature has been dedicated to exploring the potential of MARL algorithms. Efforts have been made to integrate deep learning (i.e., Deep MARL), attention mechanisms, graph neural networks (GNNs), and other advanced techniques to improve coordination between agents and enhance traffic efficiency. With the growing complexity and interconnectedness of multiple intersections, addressing the non-stationarity issue in the MARL setting becomes crucial.

In our review of recent studies leveraging MARL for TSC at multiple intersections, presented in Table I, we identify three primary limitations through detailed analysis. First, these studies generally maintain the original structure of the RL model, opting instead to augment it with advanced techniques for extracting information from neighboring intersections for input. For instance, Chu et al. [11] incorporate actions from adjacent intersections as input features to an Actor-Critic network. The second limitation pertains to the inadequate description of traffic demand and the lack of evidence supporting high or near-saturated traffic conditions, which are essential for evaluating congestion management capabilities. Without clear information on traffic demand, assessing the effectiveness of these approaches in reducing congestion and recovering from congested states becomes challenging. For example, Wei, Xu, et al., [12] simulate a uniform traffic flow that is unlikely to result in congestion, thereby not accurately reflecting real-world complexities. Additionally, some studies use real-world data without confirming if the scenarios include actual congestion events. Lastly, the majority of these studies rely on simulation tools such as SUMO [13] and CityFlow [14] for performance evaluation, where intersection geometries are simplified, creating a significant disparity with real-world conditions. The scenario where right-turn and through movements share a lane—a common occurrence in

TABLE I
TYPICAL WORKS THAT ADDRESS NON-STATIONARITY OF MARL IN LARGE-SCALE TRAFFIC SIGNAL CONTROL

Ref	Model	Peak Traffic Demand	RL Agent Input Embedding
Wei, Xu, et al., 2019 [15]	CoLight	300 Vehicles/Lane/Hour	Graph Attention Network (GAT)
Chu et al., 2019 [11]	MA2C	1200 Vehicles/Hour	Neighbors' Observations and Fingerprints (Policy Network Parameters)
Wu, Wang, et al., 2021 [16]	DynSTGAT	Not Specified	Graph Attention Network (GAT) Combined With Temporal Convolutional Networks (TCNs) To Capture Neighbors' Influences
Guo et al., 2021 [17]	MaCAR	Not Specified	Message Propagation Graph Neural Network (MPGNN)
Devailly et al., 2022 [18]	IG-RL	1800 Vehicles/Hour	Graph Convolutional Networks (GCNs) By Aggregating Communications From Neighbors
Yang et al., 2023 [19]	HG-M2I	360 Vehicles/Lane/Hour	Hierarchical Graph Neural Networks With Attention-based GRU
Zhu et al., 2023 [20]	ALCORL	3600 Vehicles/Hour	Autoencoder To Generate Communication Messages
Han et al., 2024 [21]	MAAPPO	750 Vehicles/Hour	Attention Mechanism For Selecting Neighbors' State-action Pairs.

reality—raises questions about the applicability of proposed methods in more realistic settings. Thus, while these studies contribute valuable insights into MARL's potential for traffic light control, their real-world applicability and effectiveness in addressing congestion under varied traffic conditions warrant further investigation.

In this paper, we advance the field in several key areas. First, we enhance the vanilla RL model by introducing an innovative RL agent designed to facilitate communication with other RL agents. Second, we construct various traffic flow patterns to ensure the occurrence of congestion under traditional fixed-time control methods. We then assess our proposed method within this context to evaluate its efficacy in alleviating congestion and its ability to facilitate rapid recovery from such conditions. Lastly, we adopt a more realistic intersection scenario that accommodates different movements within a single lane. To bridge the gap with real-world applications, we also consider the effective coverage of loop detectors, cameras, and other sensors. We extend our evaluation to a 6 × 6 grid network, examining the scalability of our proposed method in handling increased network complexity.

III. BACKGROUND

In this section, we briefly explain the background knowledge related to the proposed method.

A. Basic Traffic Engineering Principles

In traffic engineering, understanding and managing queue length, pressure, and the saturated flow rate are pivotal for optimizing traffic flow and reducing congestion at intersections.

Queue length, which indicates the number of vehicles lined up at a traffic signal, serves as a primary indicator of congestion, with long queues often highlighting bottlenecks that impair network efficiency. This metric, measured via technologies such as loop detectors, overhead cameras, and advanced sensors, is crucial for identifying congestion points.

Pressure, on the other hand, assesses imbalances in traffic flow by comparing the volume of incoming and outgoing traffic at an intersection, guiding signal timing optimizations to boost network throughput and decrease travel times.

The saturated flow rate represents the maximum throughput of vehicles an intersection can handle under ideal conditions, informing traffic engineers on optimal signal timings to enhance flow without exceeding capacity limits. Simulation tools further augment these efforts by modeling multi-intersection scenarios, employing lane area detectors to simulate real-world vehicle tracking, thereby providing a comprehensive platform for traffic management analysis and optimization.

B. Reinforcement Learning

An RL agent learns decisions through interactions with the environment that is modeled as a Markov Decision Process (MDP), defined as a tuple (S, A, P, R, γ), where S denotes a state space, A denotes an action space, $P : S \times A \times S \to [0, 1]$ denotes transition probabilities between states, R denotes a reward function, and $\gamma \in [0, 1]$ is a discount factor. Specifically, at each time period t, the agent observes a state $s_t \in S$ and takes an action $a_t \in A$, which is determined by the policy $\pi : S \to A$. Then, the next state s_{t+1} is reached with a transition probability $T(s_{t+1}|s_t, a_t)$, and the agent receives a reward $r_t \in R$. The action-value function Q^π is defined to evaluate how good it is for an agent to pick the action a_t based on policy π in state s_t. It is expressed as the expected cumulative reward: $Q^\pi(s, a) = \mathbb{E}[\sum_{i=t}^{\infty} \gamma^i r_{t+i} | s_t = s, a_t = a]$. The objective of an RL agent is to learn the optimal policy π^* for maximizing $Q^\pi(s, a)$.

1) Actor-Critic Method: In Reinforcement Learning (RL), the Actor-Critic method is a type of policy gradient approach that involves two main components: an actor and a critic. The actor, represented as $\pi(a|s, \theta)$, learns a policy parameterized by θ to select actions a based on a probability distribution, given the current state of the environment at time t. Concurrently, the critic, denoted as $\hat{v}(s, w)$ and implemented via a multi-layer neural network, evaluates the potential of different states by computing an estimated value function. This evaluation helps in assessing the advantage of being in a state, defined as $A(s) = Q(s, a) - V(s) = r + \gamma V(s') - V(s)$, where s' is the subsequent state following s. The policy (θ) and value network (w) parameters are updated through optimization of policy loss, value loss, and entropy loss. These losses are mathematically expressed as:

$$\mathcal{L}(\theta) = -\frac{1}{|B|} \sum_{t \in B} \log \pi_\theta(a_t|s_t) \hat{R}t, \tag{1}$$

$$\mathcal{L}(w) = \sum (R - V(s))^2, \tag{2}$$

$$H = -\sum \pi p(\pi) \log p(\pi), \tag{3}$$

with the entropy loss included to promote exploration by the actor in the policy space.

2) Proximal Policy Optimization: The Actor-Critic method, while effective, faces challenges with high computational complexity due to its reliance on a second-order derivative matrix, making it less practical for large-scale applications. Proximal Policy Optimization (PPO), building on the principles of Trust Region Policy Optimization (TRPO) [22], simplifies this by using a KL divergence constraint to limit the magnitude of policy updates, thereby avoiding the computational burden of second-order methods. PPO further eases implementation and adjustment by incorporating this constraint as a penalty within the objective function, rather than as a strict limit. This approach enables gradual, controlled updates within a defined "trust region", facilitating convergence towards optimal solutions. The core of PPO is a clipped surrogate objective function, detailed as:

$$\mathcal{L}^{CPI}(\theta) = \hat{\mathbb{E}}_t[\min(r_t(\theta)\hat{A}_t, \text{clip}(r_t(\theta), 1-\epsilon, 1+\epsilon)\hat{A}_t)], \tag{4}$$

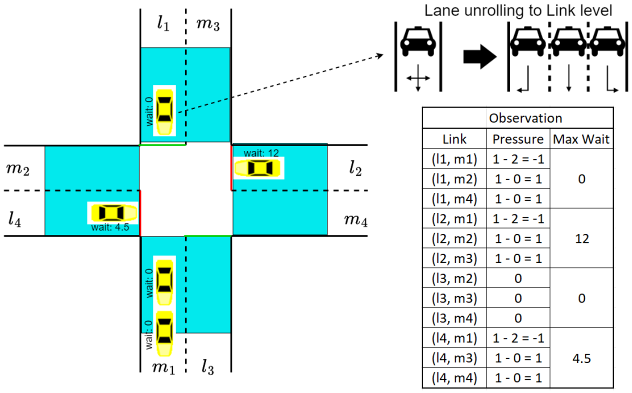
where $r_t(\theta)$ represents the ratio of the new to the old policy probabilities, and $\hat{A}_t$ is the advantage estimate, measuring the benefit of choosing action a_t in state s_t. This formulation balances the exploration of new policies with the stability of incremental learning, optimizing the trade-off between exploration and exploitation.

IV. Reinforcement Learning Model

Each traffic controller at an intersection is managed by an RL agent. Designing effective state, action, and reward is crucial in TSC. In the reminder of this section, we describe the definitions of these three key elements.

A. State

The state S_t at each intersection should provide a comprehensive snapshot of the current traffic situation, enabling the RL agent to make informed decisions to optimize traffic flow. Therefor, it is essential to capture the environmental factors that significantly impact the decision-making process for signal timing. Previous studies often use queue length (the number of approaching vehicles) and accumulated waiting time as the state, proving effective in isolated intersections. However, these metrics may not fully represent traffic conditions under high demand. This limitation partly stems from the finite coverage of sensors or vehicular networks, visualized as a narrow range highlighted in blue in Fig. 2. Relying solely on queue length could lead to inaccuracies; for instance, a sensor with limited range might detect only one vehicle in a congested intersection. To address this, we advocate to include traffic pressure into the state, offering a more accurate reflection of the current traffic scenario.



Observation		
Link	Pressure	Max Wait
(l1, m1)	1 - 2 = -1	0
(l1, m2)	1 - 0 = 1	
(l1, m4)	1 - 0 = 1	
(l2, m1)	1 - 2 = -1	12
(l2, m2)	1 - 0 = 1	
(l2, m3)	1 - 0 = 1	
(l3, m2)	0	0
(l3, m3)	0	
(l3, m4)	0	
(l4, m1)	1 - 2 = -1	4.5
(l4, m3)	1 - 0 = 1	
(l4, m4)	1 - 0 = 1	

Fig. 2. One example of local observation. Link-level pressure and waiting time can be captured via road-side sensors or cameras. If multiple movements share one lane, it is equally distributed to link level.

Specifically, we use the pressure $\texttt{pressure}_t(L, M)$ and accumulated waiting time $\texttt{wait}_t(L, M)$ of head vehicle at the intersection i at time t to represent its current traffic condition:

$$o_{t,i} = \texttt{pressure}_t(L, M), \texttt{wait}_t(L, M) \tag{5}$$

where (L, M) denotes all input and output links at intersection i. Vehicles entering input link in order to make movement join a queue dedicated to that movement. As illustrated in Fig. 2, the $\texttt{pressure}$ of an intersection is defined as the difference between the numbers of vehicles on the input links L and output links M in the last time step. The $\texttt{wait}$ is defined as the cumulative delay of the first vehicle on each link in the last time step.

This study acknowledges that a single lane can support multiple traffic flows, such as combined through/left-turn lanes, as illustrated in Fig. 2. This configuration may lead to "Head of Line" blocking, where a vehicle intending to proceed straight is blocked by a vehicle ahead attempting a left turn. Our model reflects this real-world scenario by equating the vehicle count on a link to that in a shared lane.

B. Action Space

Based on the state, the controller chooses an action (i.e., signaling decision) to take. In our approach, the control action for each local signalized intersection is represented as a phase p, which corresponds to a specific set of permissible traffic movements, as illustrated in Fig 3. At each time step t, each agent selects a phase p as its action a_i^t from the set of all possible phases. Additionally, we establish a fixed execution time Δt for each action, while including a yellow time t_y for the active phase to allow for the safe clearance of vehicles already present in the intersection.

C. Rewards

After executing a signaling decision, the traffic controller receives feedback through a reward, indicating the effectiveness of the action. This reward function serves as a critical guide for the RL agent towards achieving a well-defined goal. In rural settings, where traffic is light and congestion infrequent,

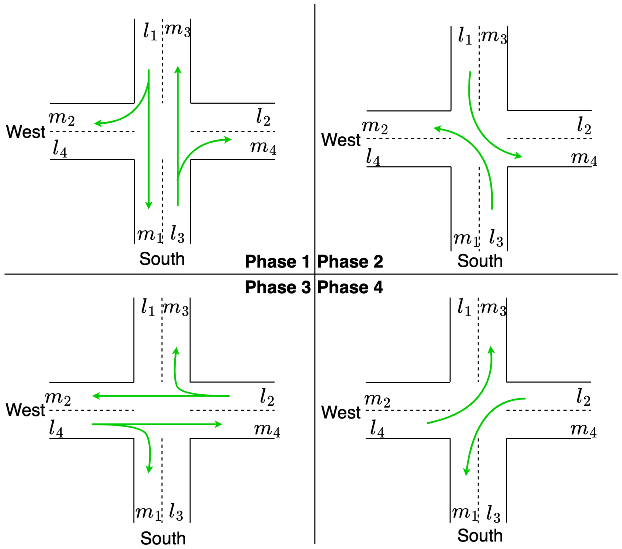


Fig. 3. Sample phase set containing four phases. Phases 1 and 2 correspond to North-South bound movements, while phases 3 and 4 correspond to West-East bound movements. The actual size of the phase set may vary.

the reward should prioritize improving individual travel experiences over congestion mitigation. Typically, a reward function encompasses various performance indicators such as vehicle delay, throughput, and queue length, assigning weights to each based on their relevance to the traffic management system's goals.

Given our focus on reducing congestion, we incorporate metrics like queue length and the maximum waiting time across all lanes at an intersection into the reward function. The reward at time t for intersection i is mathematically expressed as:

$$r_{(t,i)} = -(\sum_{l\in L}(\texttt{halting}_{t+\Delta t}[l]) + \max_{l\in L}(\texttt{wait}_{t+\Delta t}[l])), \quad (6)$$

where $\texttt{halting}_{t+\Delta t}[l]$ denotes the count of halted vehicles in lane l at time $t+\Delta t$, and wait $\texttt{wait}_{t+\Delta t}[l]$ represents the waiting time in lane l at the same timestep. By penalizing the sum of halted vehicles and the maximum waiting time, we encourage the agent to reduce both, enhancing traffic flow efficiency.

V. Proposed Approach

We propose PairUpLight, a system that facilitates communication between two agents, as illustrated in Fig. 4. This system activates a message pipeline where, at each time step, the message m_t^a flows from one agent to another along a congested route, thereby helping to alleviate traffic congestion. In this paper, a denotes the index of agent a. The most straightforward method is to output a message alongside the executable action, allowing this outgoing message to serve as the incoming message for another agent in the subsequent timestep. We will detail the backbone RL model and the design of PairUpLight in the remainder of this section.

A. Backbone RL Model

We selected PPO, an Actor-Critic architecture, with Generalized Advantage Estimation (GAE) as the backbone RL model, due to its proven effectiveness in stabilizing the learning process and reducing variance in policy updates. The integration of the advantage function facilitates efficient learning by prioritizing actions that yield higher rewards. The policy gradient is updated as follows:

$$\nabla_\theta(\mathcal{J}(\theta)) = \mathbb{E}_{s_t\sim\rho^\pi, a_t\sim\pi_\theta}[\nabla_\theta log\pi_\theta(s_t,a_t)A_t^{GAE(\gamma,\lambda)} + \beta\nabla_\theta\mathcal{H}(\pi_\theta(s_t))] \quad (7)$$

where $A_t^{\mathrm{GAE}(\gamma,\lambda)}$ represents the Generalized Advantage Estimator. In many cases, the Actor and Critic share initial neural network layers before diverging into separate "heads" for distinct tasks. However, given the complexity of the multi-intersection environment, we utilize completely separate networks for the Actor and Critic. This approach ensures each network is specialized and optimized for its respective role without compromising on the needs of the other components. Specifically, we incorporate our proposed message-passing mechanism into the Actor network and integrate information from direct and two-hop neighbors as input to the Critic network, resulting in a coordinated Actor network and a centralized Critic network. Furthermore, we embrace the Centralized Training with Decentralized Execution (CTDE) paradigm and employ parameter sharing to enhance sample efficiency.

CTDE. In the CTDE framework, agents are trained together on a central server, leveraging centralized knowledge for coordination. After training, the Actor network is deployed at each intersection for autonomous operation. This method combines centralized training's comprehensive network insight with decentralized execution, allowing intersections to independently manage traffic flow while maintaining effective communication and coordination.

Parameter Sharing. In our study focusing on homogeneous intersections, we utilize parameter sharing to train a unified Actor and Critic network across all intersections, enhancing sample efficiency and convergence speed. Each agent, during decentralized execution, operates independently with its own copy of the Actor network, enabling diverse behaviors based on unique observations of the state and communications. Our experiments predominantly explore these homogeneous settings to leverage shared learning benefits. Additionally, to assess our model's adaptability to varied environments, we tested it on heterogeneous intersections without parameter sharing, aiming to evaluate its generalizability.

B. PairUpLight

Coordinated Actor Network. The coordinated Actor network explicitly learns a communication protocol. We aim to facilitate the exchange of valuable information with the other agent. It enables PairUpLight to jointly find the global optimal solution. The local policy is calculated as follows:

$$\pi_{t+1,a}, m_{t+1,a} = \pi_{\theta^-}(\cdot|s_{t,a}, \hat{m}_{t,a'}), \quad (8)$$

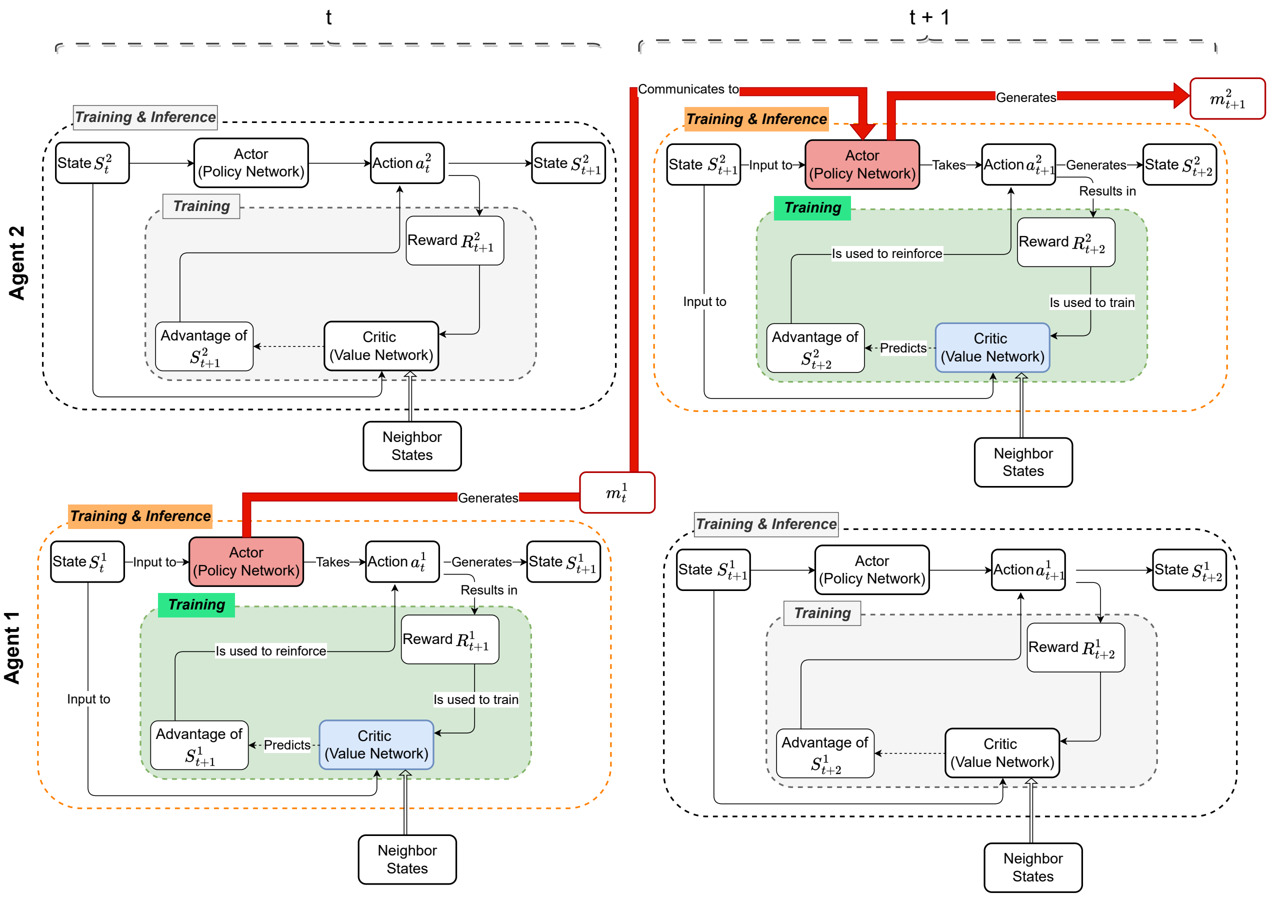


Fig. 4. The proposed communication pipeline, as highlighted in red, involves passing the message m_t^a from one agent to another's Actor network as a continuous value. For simplicity, only one agent is highlighted at each time step, with the other agent being greyed out. We have chosen an Actor-Critic structure with centralized training and decentralized execution (CTDE) as the backbone of our proposed MARL model. The Critic network is not utilized during the inference phase.

where, $s_{t,a}$ represents the local observation (e.g., intersection information from loop detectors, and cameras) that is provided as input to the network at time step t. The output is the action probability distribution, and the action is selected based on an ϵ-greedy strategy. To activate the communication mechanism, an additional real-valued message, $\hat{m}_{t,a'}$, is fed into the network from either the current agent itself or one of its neighboring agents. On the output side, the system also produces a message $m_{t+1,a}$, which is later processed by the regularizer unit and used to update the corresponding message. The network architecture of the coordinated Actor network is depicted in the upper part of Fig. 5.

Determining with whom the current agent should communicate is a key question in our approach. Through empirical study, we have found that among the neighboring nodes, including itself, *the one that experiences congestion first (upstream intersection)* is crucial for the current node. Therefore, in our design each intersection pairs up with the most congested upstream intersection and an communication channel is established between these two. By carefully determining the relevant communication partners based on congestion and latent congestion risk, our coordinated actor network focuses on exchanging critical information that influences the decision-making process. This targeted communication approach allows agents to effectively share important traffic information.

Centralized Critic Network. A separate Critic network is employed, mirroring the Actor network's architecture with key differences in inputs and outputs, as depicted in Fig. 5. Unlike the Actor network, which concentrates on coordination and communication across intersections, the Critic network underscores the centralized learning component of the multi-agent system. The critic network is defined as follows:

$$v_t^{(a)}, h_{t,V}^{(a)} = V(s_t^{(a)}, h_{t-1,v}^{(a)}; w), \tag{9}$$

where, $h_{t-1,v}$ denotes the hidden state from the LSTM layer. Assuming that accessing broader observations, including the global state, facilitates value learning for the critic network, we propose incorporating traffic conditions from one hop and two hop neighboring intersections into its input. This approach aims to provide a more comprehensive view of the traffic network, capturing both the direct impact of immediate neighbors and the broader influence of two hop neighbors. By doing so, the critic network gains a deeper understanding of traffic dynamics and patterns, enhancing its decision-making process. This rich information not only improves value

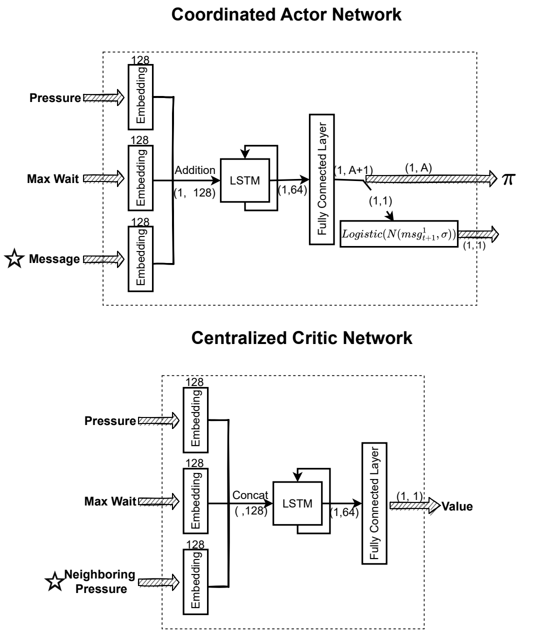


Fig. 5. Architecture of Actor network and Critic network in PairUpLight. The Neighboring Pressure is from one of the one-hop neighbor that experiences congestion first.

estimates by acknowledging wider traffic impacts but also stabilizes and boosts the learning process, leading to better performance in the MARL framework. Notably, incorporating up to two-hop neighbor information presents a challenge for edge intersections in the grid, which have fewer neighbors than those in the center. To address this, we use a padding technique for intersections lacking two-hop neighbors, filling in missing neighbor information. This strategy ensures the critic network accommodates various intersection configurations uniformly across the grid, facilitating efficient and accurate learning.

The PairUpLight framework combines the Actor network's coordination capabilities with the Critic network's centralized learning to efficiently mitigate traffic congestion and optimize traffic flow. The proposed method's pseudo-code is presented in Algorithm 1.

Algorithm 1: PairUpLight

```
Parameter: $\alpha$, learning rate; $\gamma$, discount factor; $T$, planning horizon per episode; $|B|$,batch size; $M$, minibatch size; $K$, epochs; $\epsilon$, epsilon-greedy;
Initialize: $\theta$, the parameters for policy; $w$, the parameters for critic $V$ using Orthogonal initialization;
repeat
    $s_0^a \leftarrow$ initial state, $h_0^a \leftarrow 0$, $m_0^a \leftarrow 0$ for each agent $a$, $t \leftarrow 0$, $B = \emptyset$;
    for each episode $e$ do
        /* explore experience */
        for $i = 1$ to $B$ do
            for all agents $a$ do
                Get message $\hat{m}_{t-1,\pi}^{(a')}$ of previous time-steps from agents $a'$: $p_t^{(a)}, h_{t,\pi}^{(a)}, m_{t,\pi}^{(a)} = \pi(s_t^{(a)}, h_{t-1,\pi}^{(a)}, \hat{m}_{t-1,\pi}^{(a')}; \theta)$;
                $v_t^{(a)}, h_{t,V}^{(a)} = V(s_t^{(a)}, h_{t-1,v}^{(a)}; w)$;
                With probability $\epsilon$ pick random $u_t^{(a)}$, else $u_t^{(a)} = \max p_t^{(a)}$;
                Set message $\hat{m}_t^{(a)} = Logistic(\mathcal{N}(m_t^{(a)}, \sigma))$
            end
            Execute actions $u_t$, observe $r_t, s_{t+1},$;
            /* save all agents' data to the buffer */
            $B \leftarrow B \cup \{(s_t, u_t, r_t, v_t, h_t, \hat{m}_t)\}$;
            $t \leftarrow t + 1$;
        end
        if not Terminated then
            $v_{B+1}^{(a)}, h_{B+1,V}^{(a)} = V(s_{B+1}^{(a)}, h_{B,V}^{(a)}; w)$;
        end
        /* update network parameters */
        Compute advantage estimate $\hat{A}$ via GAE;
        Compute reward-to-go $\hat{R}$;
        update $\theta$ on $L(\theta)$, $w$ on $L(w)$ with $K$ epochs and minibatch size $M$ (Eq. (7));
    end
until Stop condition reached;
```

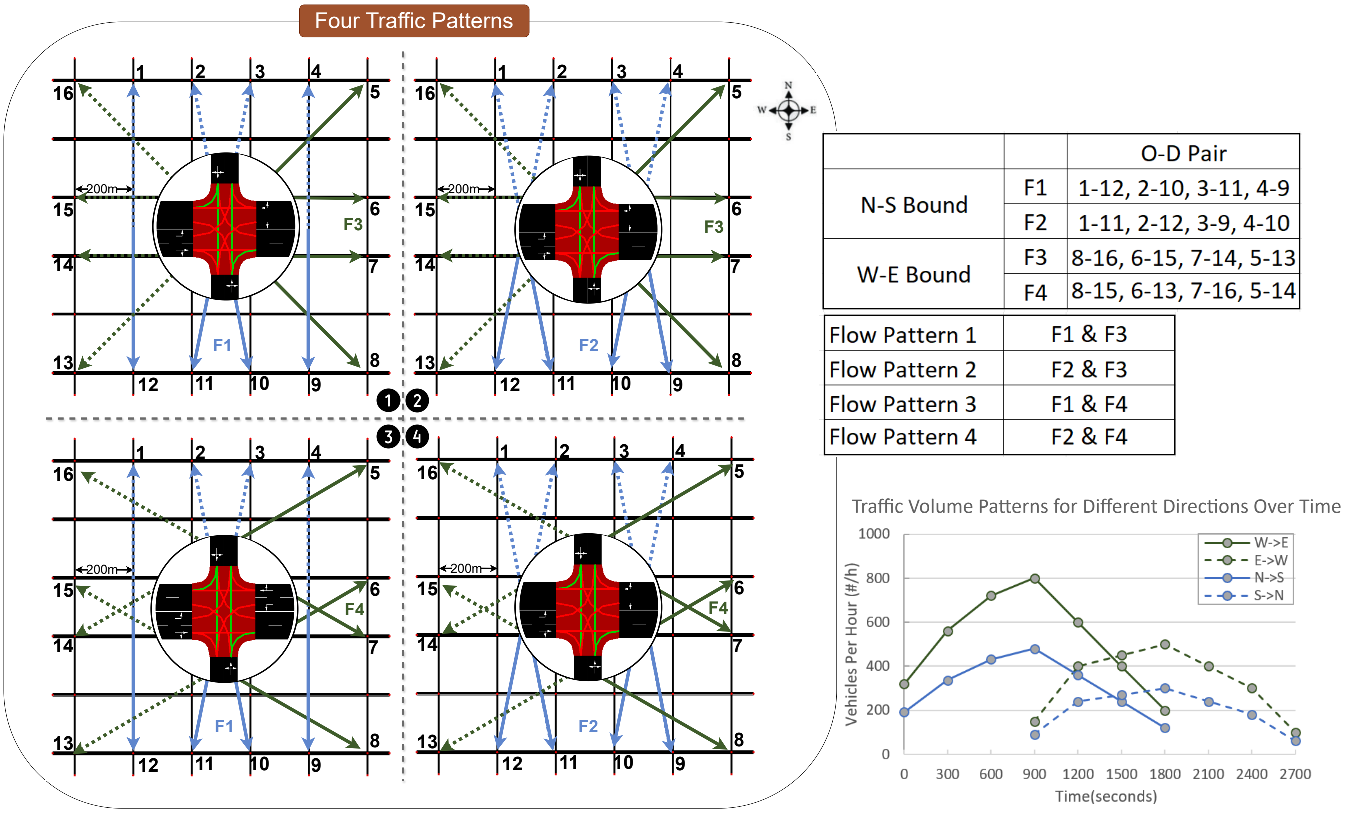
VI. Numerical Experiments

This section's experiments evaluate PairUpLight's performance in managing congestion, its communication effectiveness, and resilience across high-demand traffic scenarios.

A. Datasets and Environment Settings

Experiments utilize the microscopic traffic simulator SUMO [23], which allows for the configuration of traffic environments including road networks, traffic flow patterns and traffic signal timings. SUMO serves as the environment for agent (traffic signal controller) interaction. During simulations, data such as traffic flow, vehicle waiting times, and other metrics are collected to train the RL agents.

Intersection Modeling. We extend the network from Chu et al. (2019) [11] and evaluate PairUpLight on a synthetic 6x6 grid network, marking the largest scale-up to date, as depicted in Fig. 6, which includes intersections with two-lane arterial streets and one-lane avenues. In one-lane avenues, a single lane accommodates left turns, straight movement, and right turns. On two-lane arterial streets, the right lane supports both straight movement and right turns, while the left lane is designated for left turns. The separation between intersections is 200 meters, with loop detectors and cameras monitoring lanes up to 50 meters away. This intersection design aims to more closely replicate real-world traffic scenarios. Additionally, each intersection's signal timing follows a four-phase plan, similar to the one shown in Fig. 3, with each phase



		O-D Pair
N-S Bound	F1	1-12, 2-10, 3-11, 4-9
	F2	1-11, 2-12, 3-9, 4-10
W-E Bound	F3	8-16, 6-15, 7-14, 5-13
	F4	8-15, 6-13, 7-16, 5-14

Flow Pattern 1	F1 & F3
Flow Pattern 2	F2 & F3
Flow Pattern 3	F1 & F4
Flow Pattern 4	F2 & F4

Fig. 6. Four different traffic patterns with time-varying flow rate. The circular area in the center of each scenario represents the configuration of each intersection. The arrows' directions indicate the movement of traffic flowS. The dotted lines represent the flows that begin after the 900^{th} seconds.

lasting five seconds, plus a two-second yellow phase for safety.

Traffic Flow Design. To assess PairUpLight's effectiveness across diverse traffic scenarios, we create four traffic patterns (*Flow Pattern 1-4*) with time-varying traffic flows to simulate congested conditions. We illustrate how traffic is managed in these four scenarios in Fig. 6, with each having two from the four groups (F1-F4) and corresponding origin-destination or OD pairs. Initially, for the first 1800 seconds, eastbound and southbound traffic is loaded. Then, from 900^{th} seconds onward, we introduce westbound and northbound traffic to increase complexity. Take F1 1-12 for instance, one flow begins by loading vehicles from the southbound lane of Node 1 as soon as the simulation starts, with all vehicles exiting at Node 12. The peak flow rate for this direction occurs at the 900^{th} second. Simultaneously, the reverse traffic flow from Node 12 to Node 1 initiates. These two flows overlap from the 900^{th} to the 1800^{th} second, at which point the reverse flow hits its peak rate of 500 vehicles per hour. During the overlap period, a total of 16 O-D pairs (i.e., 16 arrows as indicated in Fig. 6) coexist within the network. To the best of the authors' knowledge, this represents the highest number of O-D pairs reported in recent studies. We employ high traffic volumes to demonstrate that PairUpLight can recover from oversaturated conditions. Additionally, we evaluate PairUpLight's generalizabiliy in light traffic conditions using a uniform flow pattern (*Flow Pattern 5*), with 300 vehicles per hour in the west-east direction and 90 vehicles per hour in the south-north direction.

Our congestion generation strategy, informed by empirical studies, includes: 1) Adding more intersecting O-D pairs to increase traffic intersections, and 2) Staggering vehicle departure times across various O-D pairs to overlap traffic flows and induce congestion.

B. *Comparison Methods*

We have carefully selected representative baselines from the field for providing comprehensive benchmarking. These methods, as shown below, have been widely used as standard benchmarks in similar studies.

- **Fixedtime**: It adopts predetermined signal timing values and does not adapt to changing traffic conditions.
- **SingleAgentRL**: A single agent is trained using the PPO algorithm, and its learned policy is uniformly applied to all intersections. This approach does not involve inter-agent communication or the use of neighboring intersections' information
- **MA2C** [11]: MA2C, a MARL-based TSC approach, integrates policy fingerprints from neighboring intersections and is based on the Actor-Critic RL algorithm.
- **CoLight** [12]: CoLight is a MARL-based TSC method that enhances sampling efficiency through parameter sharing and employs Graph Attention Networks (GAT) to determine the significance of adjacent intersections. It utilizes the Deep Q-Learning algorithm as its backbone RL model.

C. Experiment Result

We evaluate PairUpLight and other models using the previously mentioned five traffic flow patterns, training all models solely on traffic pattern F1 and then evaluating them on the remaining patterns. This approach reflects real-world conditions where traffic patterns fluctuate with time and situation. Unlike previous studies where RL models are trained and tested on identical traffic flows with minimal variance, our method aims to enhance real-world applicability by introducing variability in testing scenarios.

Additionally, to support our viewpoint that benchmark models perform well under light, congestion-free traffic but significantly decline in performance as conflicting flow rates increase, we present further experimental results. Here, all models are trained and evaluated exclusively on the uniform traffic pattern (i.e., Flow Pattern 5).

Performance Metrics. We evaluate performance based on average waiting time and average travel time, in line with standard practice. Average waiting time is calculated as the mean of the maximum waiting times for vehicles across all incoming lanes at every intersection. Meanwhile, average travel time is determined by averaging the travel times of all vehicles entering and exiting the network.

Results During Training. The training performance of PairUpLight is presented in Fig. 7, as indicated by the average waiting time per timestep. It is trained for 1000 episodes, and it starts with a high average waiting time but quickly improves, with the trend showing a sharp decline as the episodes progress. The lowest average waiting time achieved by PairUpLight dips significantly below the performance levels of both the fixed-time control and the single agent RL model, suggesting that PairUpLight outperforms these methods after sufficient training. The wide variance at the beginning that narrows over time indicates that the model becomes more stable and consistent in its performance as it learns. Additionally, we compared the training progress of PairUpLight with that of CoLight and MA2C over the first 200 episodes, as shown in Fig. 8. Despite an initial lag due to the complexity of learning effective communication strategies, PairUpLight eventually outperforms the other two methods. PairUpLight's final convergence at 76 seconds reflects an improvement of 81.46% over CoLight and 83.72% over MA2C.

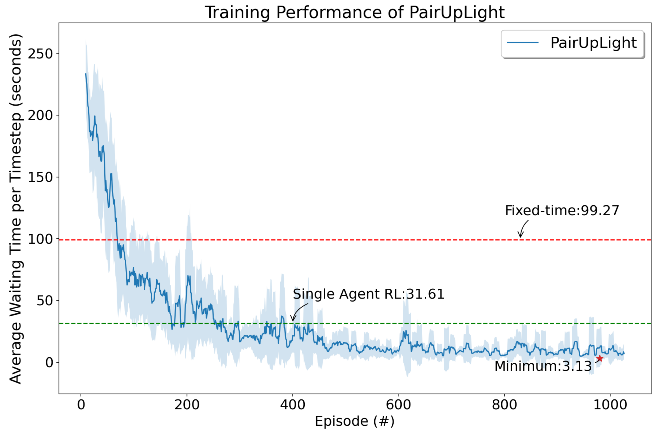


Fig. 7. Training performance of PairUpLight over 1000 episodes, depicted with a solid blue line and shaded area indicating variance. Best performance occurs at episode 980 with a 3.13-second waiting time.

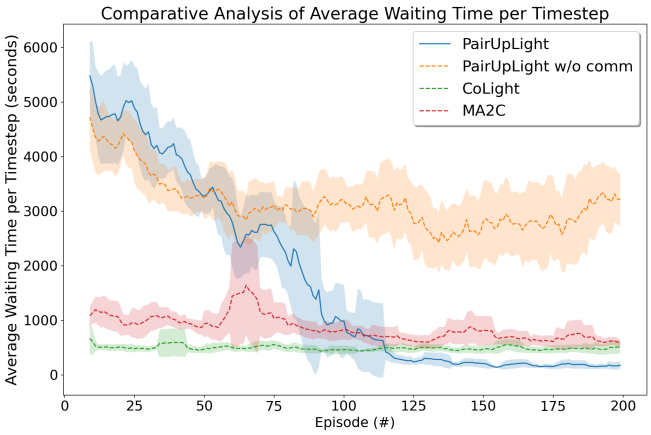


Fig. 8. Training performance over the first 200 episodes for various models.

To illustrate the significance of the communication module, we performed an ablation study by removing the communication module and noted a negative impact on performance. Figure 8 shows this effect, with the orange dotted line indicating performance in the absence of communication.

TABLE II
EVALUATION OF AVERAGE TRAVEL TIME (SECONDS) IN VARIOUS TRAFFIC SCENARIOS

Model	Pattern 1	Pattern 2	Pattern 3	Pattern 4	Pattern 5
Fixedtime	3395.34	6236.73	3446.64	4807.81	262.81
SingleAgent	936.11	3298.14	2740.10	4118.31	99.91
MA2C	15482.22	13327.66	16589.37	15210.02	375.35
CoLight	3072.75	3157.26	2472.13	3151.64	779.16
PairUpLight	388.47	414.29	330.84	445.21	87.50

Evaluation Result. During the evaluation, we take the average travel time in seconds as the performance measure to evaluate the effectiveness of the algorithms and the generalizability to other traffic scenarios. The result is shown in Table II. We have the following oberservations:

- During the evaluation phase, the performance of MA2C and CoLight, despite being promising in training, was unsatisfactory and even worse than the SingleAgent and FixedTime methods in light traffic conditions (Flow Pattern 5). Specifically, CoLight's performance is 7.8 times poorer than SingleAgent and 3 times poorer than FixedTime when tested on Flow Pattern 5. This confirms our perspective that *current MARL approaches face generalizability challenges and require a robust communication design to perform well across various traffic conditions.*
- PairUpLight consistently leads during evaluations, showcasing its ability to manage congestion effectively. Although tailored for congestion mitigation, PairUpLight

also excels under light traffic conditions where communication is unnecessary.

- MA2C's performance declines in the evaluation phase as it struggles with over-saturated traffic conditions, partially due to the absence of parameter sharing among agents. In such conditions, data tends to stay constant, failing to guide improvements. Conversely, PairUpLight's parameter sharing enables agents to collaboratively tackle congestion and its associated challenges.
- PairUpLight outperforms CoLight in evaluations by its success in pinpointing key intersections essential for alleviating congestion. CoLight uses GAT to include neighboring data but lacks a specific method to identify and prioritize critical intersections. Consequently, CoLight may not optimally manage resources at the most congested intersections.

Findings in Light Traffic Scenario. We evaluated a uniform light traffic flow (Traffic Pattern 5) in both training and evaluation phases. The results are presented in Table III.

TABLE III
EVALUATION OF AVERAGE TRAVEL TIME (SECONDS) IN LIGHT TRAFFIC SCENARIO

	Fixedtime	SingleAgent	MA2C	CoLight	PairUpLight
Pattern 5	262.81	99.91	245.64	192.17	86.33

Our experiments on light traffic scenarios reveal that MARL may be unnecessary and potentially adds complexity in such conditions. Specifically, MARL models, including CoLight, underperformed compared to single-agent RL methods in light traffic. This suggests that in low-demand scenarios, single-agent RL or even well-designed fixed-time controls could suffice. These insights highlight the importance of choosing between MARL and single-agent RL based on the traffic scenario's specific demands and conditions.

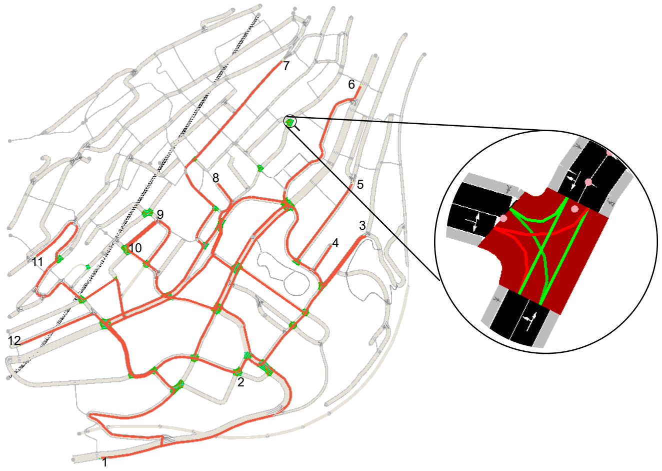


Fig. 9. Traffic scenario in Monaco. The singalized intersections in the network are marked as green. The traffic flows are highlighted in red.

D. Study of Real-world Heterogeneous Environments

We trained PairUpLight on traffic scenarios in Monaco, utilizing a real-world dataset derived from signalized intersections in the region [11]. The Monaco dataset comprises 30 signalized intersections with varying lane configurations and pre-defined signal phase sets, as shown in Fig. 9. Multiple conflicting flows with a peak flow rate of 975 vehs/h were loaded to generate saturated conditions. We presented the performance in Fig. 10. Due to the diverse characteristics of intersections, parameter sharing was not feasible. Therefore, we compared PairUpLight with MA2C and fixed-time control. Single-agent RL and CoLight, while effective in simulated environments with uniform intersections, face challenges in adapting to the heterogeneous nature of real-world settings. Despite the complexity and heterogeneity of the real-world intersections in Monaco, PairUpLight demonstrated its ability to perform effectively and provide efficient traffic management solutions.

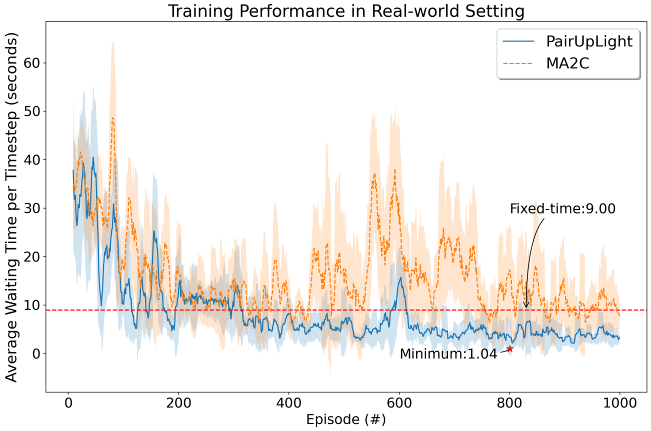


Fig. 10. Training performance under the real-world setting.

E. Communication Overhead Analysis

In PairUpLight, we facilitate coordinated agent behavior towards a global goal with efficient communication, uniquely minimizing the need for data from neighboring intersections to reduce communication delays. Our evaluation, shown in Table IV, compares PairUpLight's communication bandwidth with that of CoLight and MA2C. This comparison underscores PairUpLight's minimal communication requirements during evaluation, presenting an effective, low-overhead solution for traffic management.

TABLE IV
COMMUNICATION OVERHEAD ANALYSIS

Model	Information from Other Intersections	Communication Overhead
MA2C	queue length, policy network outputs from four neighbors	1280bits
CoLight	link-level pressure from four neighbors	1536bits
PairUpLight	message from one of its four neighbors	32bits

We conducted experiments to identify the optimal communication bandwidth for our PairUpLight model, as shown in Fig. 11. Contrary to expectations, we found that increasing the bandwidth did not improve cooperative strategies; instead, it hindered the agents' ability to identify optimal actions. The PairUpLight's Actor network outputs both the action probability distribution and a communication vector, enabling information flowing among agents. Remarkably, a single message proved most effective in our context. By adjusting the communication bandwidth, we optimized agent coordination in PairUpLight, achieving efficient and effective collaboration.

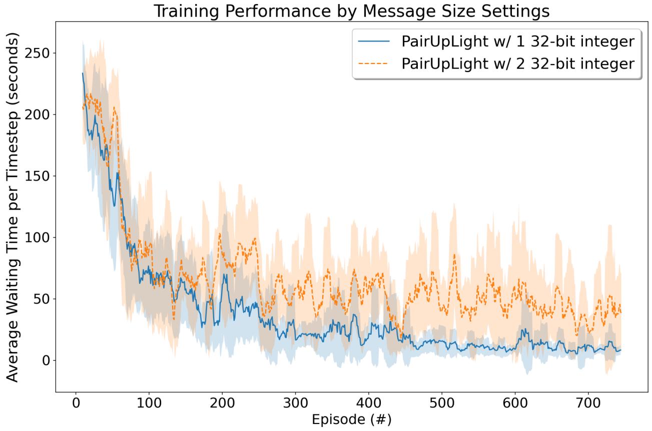


Fig. 11. A comparison of communication length between 1 and 2 32-bit integers during training. Increasing the length of the communication vector does not enhance performance.

VII. Conclusion

In this work, we present PairUpLight, a novel multi-agent reinforcement learning framework that learns effective signal timing plans by minimizing required communication between intersections. By combining a coordinated actor network with a centralized critic, our approach addresses non-stationarity and enables scalable coordination using only local sensory input and a single message from neighbors. Extensive experiments on synthetic and real-world traffic scenarios demonstrate that PairUpLight outperforms traditional MARL methods and baselines like MA2C and CoLight, achieving superior performance, robustness, and resilience. Our results highlight the critical role of efficient communication in traffic signal control and establish PairUpLight as a strong foundation for future intelligent transportation systems.

VIII. Acknowledgment

This work is supported in part by the U.S. Department of Energy, Office of Advanced Scientific Computing Research under Award Number DE-SC0024424 and by the U.S. National Science Foundation under Grant Number CNS-2340171.

References

[1] K.-L. A. Yau, J. Qadir, H. L. Khoo, M. H. Ling, and P. Komisarczuk, "A survey on reinforcement learning models and algorithms for traffic signal control," *ACM Computing Surveys (CSUR)*, vol. 50, no. 3, pp. 1–38, 2017.

[2] H. Zhao, C. Dong, J. Cao, and Q. Chen, "A survey on deep reinforcement learning approaches for traffic signal control," *Engineering Applications of Artificial Intelligence*, vol. 133, p. 108100, 2024.

[3] Z. Wu, S. Pan, F. Chen, G. Long, C. Zhang, and S. Y. Philip, "A comprehensive survey on graph neural networks," *IEEE transactions on neural networks and learning systems*, vol. 32, no. 1, pp. 4–24, 2020.

[4] A. Vaswani, "Attention is all you need," *Advances in Neural Information Processing Systems*, 2017.

[5] V. Mnih, K. Kavukcuoglu, D. Silver, A. A. Rusu *et al.*, "Human-level control through deep reinforcement learning," *Nature*, 2015.

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

[7] J. N. Foerster, Y. M. Assael, N. De Freitas, and S. Whiteson, "Learning to communicate with deep multi-agent reinforcement learning," *Advances in Neural Information Processing Systems*, pp. 2145–2153, 2016.

[8] J. Schulman, P. Moritz, S. Levine, M. Jordan, and P. Abbeel, "High-dimensional continuous control using generalized advantage estimation," *arXiv preprint arXiv:1506.02438*, 2015.

[9] P. Koonce *et al.*, "Traffic signal timing manual," United States. Federal Highway Administration, Tech. Rep., 2008.

[10] X. Liang, X. Du, G. Wang, and Z. Han, "A Deep Reinforcement Learning Network for Traffic Light Cycle Control," *IEEE Transactions on Vehicular Technology*, vol. 68, no. 2, pp. 1243–1253, 2019.

[11] T. Chu, J. Wang, L. Codecà, and Z. Li, "Multi-agent deep reinforcement learning for large-scale traffic signal control," *IEEE Transactions on Intelligent Transportation Systems*, vol. 21, no. 3, pp. 1086–1095, 2019.

[12] H. Wei, N. Xu, H. Zhang, G. Zheng, X. Zang, C. Chen, W. Zhang, Y. Zhu, K. Xu, and Z. Li, "Colight: Learning network-level cooperation for traffic signal control," *International Conference on Information and Knowledge Management, Proceedings*, pp. 1913–1922, 2019.

[13] P. A. Lopez, M. Behrisch, L. Bieker-Walz, J. Erdmann, Y.-P. Flötteröd, R. Hilbrich, L. Lücken, J. Rummel, P. Wagner, and E. Wießner, "Microscopic traffic simulation using sumo," in *2018 21st international conference on intelligent transportation systems (ITSC)*. IEEE, 2018, pp. 2575–2582.

[14] H. Zhang, S. Feng, C. Liu *et al.*, "Cityflow: A multi-agent reinforcement learning environment for large scale city traffic scenario," in *The WebConf*, 2019.

[15] H. Wei, N. Xu, H. Zhang, G. Zheng, X. Zang, C. Chen, W. Zhang, Y. Zhu, K. Xu, and Z. Li, "Colight: Learning network-level cooperation for traffic signal control," in *CIKM*, 2019.

[16] L. Wu, M. Wang, D. Wu, and J. Wu, "Dynstgat: Dynamic spatial-temporal graph attention network for traffic signal control," in *Proceedings of the 30th ACM International Conference on Information & Knowledge Management*, 2021, pp. 2150–2159.

[17] X. Guo, Z. Yu, P. Wang, Z. Jin, J. Huang, D. Cai, X. He, and X. Hua, "Urban traffic light control via active multi-agent communication and supply-demand modeling," *IEEE Transactions on Knowledge and Data Engineering*, 2021.

[18] F.-X. Devailly, D. Larocque, and L. Charlin, "Ig-rl: Inductive graph reinforcement learning for massive-scale traffic signal control," *IEEE Transactions on Intelligent Transportation Systems*, vol. 23, no. 7, pp. 7496–7507, 2021.

[19] S. Yang, "Hierarchical graph multi-agent reinforcement learning for traffic signal control," *Information Sciences*, vol. 634, pp. 55–72, 2023.

[20] R. Zhu, W. Ding, S. Wu, L. Li, P. Lv, and M. Xu, "Auto-learning communication reinforcement learning for multi-intersection traffic light control," *Knowledge-Based Systems*, p. 110696, 2023.

[21] G. Han, X. Liu, H. Wang, C. Dong, and Y. Han, "An attention reinforcement learning–based strategy for large-scale adaptive traffic signal control system," *Journal of Transportation Engineering, Part A: Systems*, vol. 150, no. 3, p. 04024001, 2024.

[22] J. Schulman, S. Levine, P. Abbeel, M. Jordan, and P. Moritz, "Trust region policy optimization," in *International conference on machine learning*. PMLR, 2015, pp. 1889–1897.

[23] D. Krajzewicz, J. Erdmann, M. Behrisch, and L. Bieker, "Recent development and applications of sumo-simulation of urban mobility," *International journal on advances in systems and measurements*, vol. 5, no. 3&4, 2012.