Multi-Agent Reinforcement Learning for Traffic Signal Control through Universal Communication Method

Qize Jiang\(^1\).2,.3, Minhao Qin\(^1\).2,.3, Shengmin Sh\(^1\).2,.3, Weiwei Sun\(^1\).2,.3 and Baihua Zheng\(^4\)

\(^1\)School of Computer Science, Fudan University
\(^2\)Shanghai Key Laboratory of Data Science, Fudan University
\(^3\)Shanghai Institute of Intelligent Electronics & Systems
\(^4\)School of Computing and Information Systems, Singapore Management University

\{qzjiang21,mhqin21\}@m.fudan.edu.cn, \{cmshi20,wwsun\}@fudan.edu.cn, bhzheng@smu.edu.sg

Abstract

How to coordinate the communication among intersections effectively in real complex traffic scenarios with multi-intersection is challenging. Existing approaches only enable the communication in a heuristic manner without considering the content/importance of information to be shared. In this paper, we propose a universal communication form UniComm between intersections. UniComm embeds massive observations collected at one agent into crucial predictions of their impact on its neighbors, which improves the communication efficiency and is universal across existing methods. We also propose a concise network UniLight to make full use of communications enabled by UniComm. Experimental results on real datasets demonstrate that UniComm universally improves the performance of existing state-of-the-art methods, and UniLight significantly outperforms existing methods on a wide range of traffic situations. Source codes are available at https://github.com/zyr17/UniLight.

1 Introduction

Traffic congestion is a major problem in modern cities. While the use of traffic signal mitigates the congestion to a certain extent, most traffic signals are controlled by timers. Timer-based systems are simple, but their performance might deteriorate at intersections with inconsistent traffic volume. Thus, an adaptive traffic signal control method, especially controlling multiple intersections simultaneously, is required.

Existing conventional methods, such as SOTL [Cools et al., 2013], use observations of intersections to form better strategies, but they do not consider long-term effects of different signals and lack a proper coordination among intersections. Recently, reinforcement learning (RL) methods have showed promising performance in controlling traffic signals. Some of them achieve good performance in controlling traffic signals at a single intersection [Zheng et al., 2019a; Oroojlooy et al., 2020]; others focus on collaboration of multi-intersections [Wei et al., 2019b; Chen et al., 2020].

Multi-intersection traffic signal control is a typical multi-agent reinforcement learning problem. The main challenges include stability, nonstationarity, and curse of dimensionality. Independent Q-learning splits the state-action value function into independent tasks performed by individual agents to solve the curse of dimensionality. However, given a dynamic environment that is common as agents may change their policies simultaneously, the learning process could be unstable.

To enable the sharing of information among agents, a proper communication mechanism is required. This is critical as it determines the content/amount of the information each agent can observe and learn from its neighboring agents, which directly impacts the amount of uncertainty that can be reduced. Common approaches include enabling the neighboring agents (i) to exchange their information with each other and use the partial observations directly during the learning [El-Tantawy et al., 2013], or (ii) to share hidden states as the information [Wei et al., 2019b; Yu et al., 2020]. While enabling communication is important to stabilize the training process, existing methods have not yet examined the impact of the content/amount of the shared information. For example, when each agent shares more information with other agents, the network needs to manage a larger number of parameters and hence converges in a slower speed, which actually reduces the stability. As reported in [Zheng et al., 2019b], additional information does not always lead to better results. Consequently, it is very important to select the right information for sharing.

Motivated by the deficiency in the current communication mechanism adopted by existing methods, we propose a universal communication form UniComm. To facilitate the understanding of UniComm, let’s consider two neighboring intersections \(I_i\) and \(I_j\) connected by a unidirectional road \(R_{i,j}\) from \(I_i\) to \(I_j\). If vehicles in \(I_i\) impact \(I_j\), they have to first pass through \(I_i\) and then follow \(R_{i,j}\) to reach \(I_j\). This observation inspires us to propose UniComm, which picks relevant observations from an agent \(I_i\) who manages \(I_i\), predicts their impacts on road \(R_{i,j}\), and only shares the prediction with its neighboring agent \(I_j\) that manages \(I_j\). We conduct a theoretical analysis to confirm that UniComm does pass the most important information to each intersection.

While UniComm addresses the inefficiency of the current communication mechanism, its strength might not be fully achieved by existing methods, whose network structures are designed independent of UniComm. We therefore design UniLight, a concise network structure based on the observa-
tions made by an intersection and the information shared by UniComm. It predicts the Q-value function of every action based on the importance of different traffic movements.

In brief, we make three main contributions in this paper. Firstly, we propose a universal communication form UniComm in multi-intersection traffic signal control problem, with its effectiveness supported by a thorough theoretical analysis. Secondly, we propose a traffic movement importance based network UniLight to make full use of observations and UniComm. Thirdly, we conduct experiments to demonstrate that UniComm is universal for all existing methods, and UniLight can achieve superior performance on not only simple but also complicated traffic situations.

2 Related Works

Based on the number of intersections considered, traffic signal control problem (TSC) can be clustered into i) single intersection traffic signal control (S-TSC) and ii) multi-intersection traffic signal control (M-TSC).

S-TSC. S-TSC is a sub-problem of M-TSC, as decentralized multi-agent TSC is widely used. Conventional methods like SOTL choose the next phase by current vehicle volumes with limited flexibility. As TSC could be modelled as a Markov Decision Process (MDP), many recent methods adopt reinforcement learning (RL) [van Hasselt et al., 2016; Mnih et al., 2016; Haarnoja et al., 2018; Ault et al., 2020]. In RL, agents interact with the environment and take rewards from the environment, and different algorithms are proposed to learn a policy that maximizes the expected cumulative reward received from the environment. Many algorithms [Zheng et al., 2019a; Zang et al., 2020; Oroojlooy et al., 2020] perform well for S-TSC. Their performance at M-TSC is not stable, as they suffer from a poor generalizability and their models are hard to train.

M-TSC. Conventional methods for M-TSC mainly coordinate different traffic signals by changing their offsets, which only works for a few pre-defined directions and has low efficiency. When adapting RL based methods from single intersection to multi-intersections, we can treat every intersection as an independent agent. However, due to the unstable and dynamic nature of the environment, the learning process is hard to converge [Bishop, 2006; Nowé et al., 2012]. Many methods have been proposed to speedup the convergence, including parameter sharing and approaches that design different rewards to contain neighboring information [Chen et al., 2020]. Agents can also communicate with their neighboring agents via either direct or indirect communication. The former is simple but results in a very large observation space [El-Tantawy et al., 2013; Arel et al., 2010]. The latter relies on the learned hidden states and many different methods [Nishi et al., 2018; Wei et al., 2019b; Chen et al., 2020] have been proposed to facilitate a better generation of hidden states and a more cooperative communication among agents. While many methods show good performance in experiments, their communication is mainly based on hidden states extracted from neighboring intersections. They neither examine the content/importance of the information, nor consider what is the key information that has to be passed from an agent to a neighboring agent. This makes the learning process of hidden states more difficult, and models may fail to learn a reasonable result when the environment is complicated. Our objective is to develop a communication mechanism that has a solid theoretical foundation and meanwhile is able to achieve a good performance.

3 Problem Definition

We consider M-TSC as a Decentralized Partially Observable Markov Decision Process (Dec-POMDP) [Oliehoek et al., 2016], which can be described as a tuple \( G = (S, A, P, r, Z, O, N, \gamma) \). Let \( s \in S \) indicate the current true state of the environment. Each agent \( i \in \mathcal{N} := 1, \cdots, N \) chooses an action \( a_i \in \mathcal{A} \), with \( \mathcal{A} := \{a_1, \cdots, a_N\} \) referring to the joint action vector formed. The joint action then transits the current state \( s \) to another state \( s' \), according to the state transition function \( P(s' | s, a) : \mathcal{S} \times \mathcal{A}^N \times \mathcal{S} \rightarrow [0, 1] \).

The environment gets the joint reward by reward function \( r(s, a) : \mathcal{S} \times \mathcal{A}^N \rightarrow \mathbb{R} \). Each agent \( i \) can only get partial observation \( z \in Z \) according to the observation function \( O(s, i) : \mathcal{S} \times i \rightarrow Z \). The objective of all agents is to maximize the cumulative joint reward \( \sum_{t=0}^{\infty} \gamma^t r(s_t, a_t) \), where \( \gamma \in [0, 1] \) is the discount factor.

Following CoLight [Wei et al., 2019b] and MPLight [Chen et al., 2020], we define M-TSC in Problem 1. We plot the schematic of two adjacent 4-arm intersections in Figure 1 to facilitate the understanding of following definitions.

Definition 1. An intersection \( I_1 \in \mathcal{T} \) refers to the start or the end of a road. If an intersection has more than two approaching roads, it is a real intersection \( I_1^{RN} \in \mathcal{T} \) as it has a traffic signal. We assume that no intersection has exactly two approaching roads, as both approaching roads have only one outgoing direction and the intersection could be removed by connecting two roads into one. If the intersection has exactly one approaching road, it is a virtual intersection \( I_1^V \in \mathcal{T}^V \), which usually refers to the border intersections of the environment, such as \( I_2 \) to \( I_1 \) in Figure 1. The neighboring intersections \( \mathcal{I} \) of \( I_1 \) is defined as \( \mathcal{I} = \{I_1\} \cup \{I_j|\forall I_j \in \mathcal{R}\} \), where roads \( R_{i,j} \) and \( \mathcal{R} \) are defined in Definition 2.

Definition 2. A Road \( R_{i,j} \in \mathcal{R} \) is a unidirectional edge from intersection \( i \) to another intersection \( j \). \( \mathcal{R} \) is the set of all valid roads. We assume each road has multiple lanes, and each lane belongs to exactly one traffic movement, which is defined in Definition 3.

Definition 3. A traffic movement \( T_{x,y} \) is defined as the traffic movement travelling across \( I_1 \) from entering lanes on road \( R_{i,j} \) to exiting lanes on road \( R_{i,y} \). For a 4-arm intersection, there are 12 traffic movements. We define the set of traffic movements passing \( I_1 \) as \( \mathcal{T} = \{T_{x,y}|x, y \in \mathcal{I} \} \), represented by orange dashed lines in Figure 1 are two example traffic movements.

Definition 4. A vehicle route is defined as a sequence of roads \( V \) with a start time \( e \in \mathbb{R} \) that refers to the time when the vehicle enters the environment. Road sequence \( V = (R^e, R^{e+1}, \cdots, R^n) \), with a traffic movement \( T \) from \( R^e \) to \( R^{e+1} \) for every \( e \). We assume that all vehicle
routes start from/end in virtual intersections. $\mathcal{V}$ is the set of all valid vehicle routes. $V_0$ in Figure 1 is one example.

**Definition 5.** A traffic signal phase $P_i$ is defined as a set of permissible traffic movements at $I_i$. The bottom of Figure 1 shows eight phases. $A_i$ denotes the complete set of phases at $I_i$, i.e., the action space for the agent of $I_i$.

**Problem 1.** In multi-intersection traffic signal control (M-TSC), the environment consists of intersections $\mathcal{I}$, roads $\mathcal{R}$, and vehicle routes $\mathcal{V}$. Each real intersection $I_i \in \mathcal{I}$ is controlled by an agent $A_i$. Agents perform actions between time interval $\Delta t$ based on their policies $\pi_i$. At time step $t$, $A_i$ views part of the environment $z_i$ as its observation, and tries to take an optimal action $a_i \in A_i$ (i.e., a phase to set next) that can maximize the cumulative joint reward $r$.

As we define the M-TSC problem as a Dec-POMDP problem, we have the following RL environment settings.

**True state $s$ and partial observation $z$.** At time step $t \in \mathbb{N}$, agent $A_i$ has the partial observation $z_i^t \subset s^t$, which contains the average number of vehicles $n_{x,i,y}$ following traffic movement $T_{x,i,y} \subset \mathcal{T}_i$ and the current phase $P_i^t$.

**Action $a$.** After receiving the partial observation $z_i^t$, agent $A_i$ chooses an action $a_i^t$ from its candidate action set $A_i$ corresponding to a phase in next $\Delta t$ time. If the activated phase $P_i^{t+1}$ is different from current phase $P_i^t$, a short-all-red phase will be added to avoid collision.

**Joint reward $r$.** In M-TSC, we want to minimize the average travel time, which is hard to optimize directly, so some alternative metrics are chosen as immediate rewards. In this paper, we set the joint reward $r^t = \sum_{I_i \in \mathcal{I}} r_i^t$, where $r_i^t = -r_{x,i,y}^{t+1}$, s.t. $x, y \in \mathcal{I}^N$ and $T_{x,i,y} \in \mathcal{T}_i$ indicates the reward received by $I_i$, with $n_{x,i,y}$ the average vehicle number on the approaching lanes of traffic movement $T_{x,i,y}$.

### 4 Methods

In this section, we first propose UniComm, a new communication method, to improve the communication efficiency among agents; we then construct UniLight, a new controlling algorithm, to control signals with the help of UniComm. We use Multi-Agent Deep Q Learning with double Q learning and dueling network as the basic reinforcement learning structure. Figure 2 illustrates the newly proposed model, and the pseudo codes of UniComm and UniLight are presented in our technique report [Jiang et al., 2022].

#### 4.1 UniComm

The ultimate goal to enable communication between agents is to mitigate the nonstationarity caused by decentralized multi-agent learning. Sharing of more observations could help improve the stationarity, but it meanwhile suffers from the curse of dimensionality.

In existing methods, neighboring agents exchange hidden states or observations via communication. However, as mentioned above, there is no theoretical analysis on how much information is sufficient and which information is important. While with deep learning and back propagation method, we expect the network to be able to recognize information importance via well-designed structures such as attention. Unfortunately, as the training process of reinforcement learning is not as stable as that of supervised learning, the less useful or useless information might affect the convergence speed significantly. Consequently, how to enable agents to communicate effectively with their neighboring agents becomes critical. Consider intersection $I_0$ in Figure 1. Traffic movements such as $T_{4,0.2}$ and $T_{3,0.4}$ will not pass $R_{0,1}$, and hence their influence to $I_1$ is much smaller than $T_{2,0.1}$. Accordingly, we expect the information related to $T_{4,0.2}$ and $T_{3,0.4}$ to be less relevant (or even irrelevant) to $I_1$, as compared with information related to $T_{2,0.1}$. In other words, when $I_0$ communicates with $I_1$ and other neighboring intersections, ideally $I_0$ is able to pass different information to different neighbors.

We propose to share only important information with neighboring agents. To be more specific, we focus on agent $A_1$ on intersection $I_1$ which learns maximizing the cumulative reward of $r_1$ via reinforcement learning and evaluate the importance of certain information based on its impact on $r_1$.

We make two assumptions in this work. First, spillback that refers to the situation where a lane is fully occupied by vehicles and hence other vehicles cannot drive in never happens. This simplifies our study as spillback rarely happens and will disappear shortly even when it happens. Second, within action time interval $\Delta t$, no vehicle can pass though an entire road. This could be easily fulfilled, because $\Delta t$ in M-TSC is usually very short (e.g. 10s in our settings).

Under these assumptions, we decompose reward $r_1^t$ as follows. Recall that the reward $r_1^t$ is defined as the average of $n_{x,1,y}$ with $T_{x,1,y} \in \mathcal{T}_1$ being a traffic movement. As the number of traffic movements $|\mathcal{T}_1|$ w.r.t. intersection $I_1$ is a constant, for convenience, we analyze the sum of $n_{x,1,y}$. i.e., $|\mathcal{T}_1| r_1$, and $r_1^t$ can be derived by dividing the sum by $|\mathcal{T}_1|$.

\[
|\mathcal{T}_1| r_1^t = \sum_{x,y \in \mathcal{I}_1^N} n_{x,1,y}^{t+1} x, y \in \mathcal{I}_1^N
\]

\[
= \sum_{x,y \in \mathcal{I}_1^N} \left( (n_{x,1,y}^t - m_{x,1,y}^t) + t_{x,1,y}^t \right)
\]

\[
= f(z_i^t) + g(s_i^t)
\]
In Eq. (1), we decompose $n_{x,y}^{t+1}$ into three parts, i.e., current vehicle number $n_{x,y}^t$, approaching vehicle number $l_{x,y}^t$, and leaving vehicle number $m_{x,y}^t$. Here, $n_{x,y}^t$, $l_{x,y}^t$, and $m_{x,y}^t$ can be observed directly. All leaving vehicles in $m_{x,y}^t$ are on $T_{x,y}$ now and can drive into next road without the consideration of spillback (based on first assumption). $P_{x,y}^t$ is derived by $\pi_1$, which uses partial observation $z_1^t$ as input. Accordingly, $m_{x,y}^t$ is only subject to $n_{x,y}^t$ and $P_{x,y}^t$ that both are captured by the partial observation $z_1^t$. Therefore, approaching vehicle number $l_{x,y}^t$ is the only variable affected by observations $s \notin z_1^t$. We define $f(z_1^t) = \sum (n_{x,y}^t - m_{x,y}^t)$ and $g(s^t) = \sum l_{x,y}^t$, as shown in Eq. (2).

To help and guide an agent to perform an action, agent $A$ will receive communications from other agents. Let $c_0$ denote the communication information received by $A_0$ in time step $t$. In the following, we analyze the impact of $c_0$ on $A_0$.

For observation of next phase $z_{x,y}^{t+1} = \{n_{x,y}^{t+1}, P_{x,y}^{t+1}\}$, we have i) $n_{x,y}^{t+1} = n_{x,y}^t + l_{x,y}^t - m_{x,y}^t$, and ii) $P_{x,y}^{t+1} = \pi_1(z_1^t, c_1)$. As $n_{x,y}^t$ and $m_{x,y}^t$ are part of $z_1^t$, there is a function $Z_1$ such that $z_{x,y}^{t+1} = Z_1(z_1^t, c_1, l_{x,y}^t)$. Consequently, to calculate $z_{x,y}^{t+1}$, the knowledge of $l_{x,y}^t$ is necessary. Without loss of generality, we assume $y_{x,y}^t \subseteq c_1$, so $z_{x,y}^{t+1} = Z_1(z_1^t, c_1)$. In addition, $z_{x,y}^{t+1}$ can be represented as $Z_2(z_1^t, c_1, l_{x,y}^t, \ldots, l_{x,y}^{t+1-1})$. Specifically, we define $Z_2(\cdots) = z_{x,y}^{t+1}$, regardless of the input.

We define the cumulative reward of $A_0$ as $R_1$, which can be calculated as follows:

$$ R_1^{t} = \sum_{j=0}^{\infty} \gamma^j I_{1}^{t+j} = \sum_{j=0}^{\infty} \gamma^j (f (z_{1}^{t+j}) + g (s^{t+j})) $$

$$ = \sum_{j=0}^{\infty} \gamma^j (f (Z_1 (z_{1}^{t}, c_1^t, \ldots, c_1^{t-j-1})) + g (s^{t+j})) $$

From above equation, we set $c_1$ as the future values of $I$:

$$ c_1 = \{ g (s^t), g (s^{t+1}), \ldots \} = \{ \sum l_{x,y}^t, \sum l_{x,y}^{t+1}, \ldots \} $$

All other variables to calculate $R_1^t$ can be derived from $c_1$:

$$ c_1^{t+j} = c_1 \setminus \{ g (s^t), \ldots, g (s^{t+j-1}) \} \quad j \in \mathbb{N}^+ $$

$$ g (s^{t+k}) \in c_1 \quad k \in \mathbb{N} $$

Hence, it is possible to derive the cumulative rewards $R_1^t$ based on future approaching vehicle numbers of traffic movements $l_{x,y}^{t+1}$s from $c_1$, even if other observations remain unknown. As the conclusion is universal to all existing methods on the same problem definition regardless of the network structure, we name it as UniComm. Now we convert the problem of finding the cumulative reward $R_1^t$ to how to calculate approaching vehicle numbers $l_{x,y}^{t+1}$ with current full observation $s^t$. As we are not aware of exact future values, we use existing observations in $s^t$ to predict $l_{x,y}^{t+1}$.

We first calculate approaching vehicle numbers on next interval $\Delta t$. Take intersections $I_0, I_1$, and road $R_{0,1}$ for example, for traffic movement $T_{0,1}$ which passes intersections $I_0, I_1, I_2$, and $I_3$, approaching vehicle number $l_{x,y}^{t+1}$ depends on the traffic phase $P_{0,1}^{t+1}$ to be chosen by $A_0$. Let’s revisit the second assumption. All approaching vehicles should be on $T_{0,1} = \{ T_{y,0.1} \cap T_{y,0.1} \cap I_0, y \in T_{y} \}$, which belongs to $z_0^t$. As a result, $T_{0,1}$ and the phase $P_{0}^{t+1}$ affect $l_{0,1}$, even though there might be other factors.

We convert observations in $T_0$ into hidden layers $h_0$ for traffic movements by a fully-connected layer and ReLU activation function. As $P_{0}^{t+1}$ can be decomposed into the permission for every $T \in T_0$, we use self-attention mechanism [Vaswani et al., 2017] between $R_{0,1}$ with Sigmoid activation function to predict the permissions $g_0$, traffic movements in next phase, which directly multiplies to $h_0$. This is because for traffic movement $T_{x,y} \in T_0$, if $T_{x,y}$ is permitted, $h_{x,y}$ will affect corresponding approaching vehicle numbers $l_{x,y}^t$; otherwise, it will become an all-zero vector and has no impact on $l_{x,y}^t$. Note that $R_{x,0}$ and $R_{0,y}$ might have different numbers of lanes, so we scale up the weight by the lane numbers to eliminate the effect of lane numbers. Finally, we add all corresponding weighted hidden states together and use a fully-connected layer to predict $l_{0,y}^t$.

To learn the phase prediction $P_{0}^{t+1}$, a natural method is using the action $a_0$ finally taken by the current network. However, as the network is always changing, even the result phase action $a_0$ corresponding to a given $s$ is not fixed, which makes phase prediction hard to converge. To have a more stable action for prediction, we use the action $a_0^s$ stored in the replay buffer of DQN as the target, which makes the phase prediction more stable and accurate. When the stored action $a_0^s$ is selected as the target, we decompose corresponding phase $P_{0}^{t}$ into the permissions $g_0^p$ of traffic movements, and calculate the loss between recorded real permissions $g_0^p$ and predicted permissions $g_0^p$ as $L_p = \text{BinaryCrossEntropy}(g_0^p, g_0^p)$.
For learning approaching vehicle numbers $l_{0,y,z}$ prediction, we get vehicle numbers of every traffic movement from replay buffer of DQN, and learn $l_{0,y,z}$ through recorded results $l_{0,y,z}$. As actions saved in replay buffer may be different from the current actions, when calculating volume prediction loss, different from generating UniComm, we use $g_0^t$ instead of $g_0$, i.e., $L_v = \text{MeanSquaredError}(g_0^t \cdot h_0, l_{0,y,z})$.

Based on how $c_k^t$ is derived, we also need to predict approaching vehicle number for next several intervals, which is rather challenging. Firstly, with off-policy learning, we can’t really apply the selected action to the environment. Instead, we only learn from samples. Considering the phase prediction, while we can supervise the first phase taken by $A_0$, we, without interacting with the environment, don’t know the real next state $s' \sim P(s,a)$, as the recorded $a_0^t$ may be different from $a_0$. The same problem applies to the prediction of $l_{0,y,z}$ too. Secondly, after multiple $\Delta t$ time intervals, the second assumption might become invalid, and it is hard to predict $l_{0,y,z}$ correctly with only $z_0$. As the result, we argue that as it is less useful to pass unsupervised and incorrect predictions, we only communicate with one time step prediction.

4.2 UniLight

Although UniComm is universal, its strength might not be fully achieved by existing methods, because they do not consider the importance of exchanged information. To make better use of predicted approaching vehicle numbers and other observations, we propose UniLight to predict the Q-values.

Take prediction of intersection $I_1$ for example. As we predict $l_{x,1,y}$ based on traffic movements, UniLight splits average number of vehicles $n_{x,1,y}$, traffic movement permissions $g_1$ and predictions $l_{x,1,y}$ into traffic movements $T_{x,1,y}$, and uses a fully-connected layer with ReLU to generate the hidden state $h_1$. Next, considering one traffic phase $P$ that permits traffic movements $T_P$ among traffic movements $T_{1}$ in $I_1$, we split the hidden states $h_1$ into two groups $G_1 = \{ h_{1,x} | T_{1} \in T_{P} \}$ and $G_2 = \{ h_{1,x} | h_{1,x} \in G_1, T_{x} \notin T_{P} \}$, based on whether the corresponding movement is permitted by $P$ or not. As traffic movements in the same group will share the same permission in phase $P$, we consider that they can be treated equally and hence use the average of their hidden states to represent the group. Obviously, traffic movements in $G_1$ are more important than those in $G_2$, so we multiply the hidden state of $G_1$ with a greater weight to capture the importance. Finally, we concatenate two hidden states of groups and a number representing whether current phase is $P$ through a fully-connected layer to predict the final Q-value.

5 Experiments

We conduct our experiments on the microscopic traffic simulator CityFlow [Zhang et al., 2019], set the action time interval $\Delta t$ to 10s and the all-red phase to 5s. We train our newly proposed methods and their competitors, including newly proposed ones and the existing ones selected for comparison, with 240,000 frames on a cloud platform with 2 virtual Intel 8269CY CPU core and 4GB memory. We run all the experiments 4 times independently to pick the best one, which is then tested 10 times to get the average performance.

5.1 Experimental Setup

Datasets. We use the real-world datasets from four different cities, including Hangzhou (HZ), Jinan (JN), and Shanghai (SH) from China, and New York (NY) from USA. The HZ, JN, and NY datasets are publicly available [Wei et al., 2019b], and widely used in many related studies. Though these datasets are constructed based on real traffics, they have short simulation time, same road length, a small number of lanes, and vehicle trajectories are simulated. To simulate an environment that is much closer to reality, we construct SH dataset based on real taxi trajectories in Shanghai. The detail of SH dataset (including $SH_1$ and $SH_2$) can be found in our technique report [Jiang et al., 2022].

Competitors. To evaluate the effectiveness of newly proposed UniComm and UniLight, we implement five conventional TSC methods and six representative reinforcement learning M-TSC methods as competitors, which are listed below. (1) SOTL [Cools et al., 2013], a S-TSC method based on current vehicle numbers on every traffic movement; (2) MaxPressure [Varaiya, 2013], a M-TSC method that balances the vehicle numbers between two neighboring intersections; (3) MaxBand [Little et al., 1981] that maximizes the green wave time for both directions of a road; (4) TFP-TCC [Jiang et al., 2021] that predicts the traffic flow and uses traffic congestion control methods based on future traffic condition; (5) MARLIN [El-Tantawy et al., 2013] that uses Q-learning to learn joint actions for agents; (6) MA-A2C [Mnih et al., 2016], a general RL method with actor-critic structure and multiple agents; (7) MA-PPQ [Schulman et al., 2017], a popular policy based method; (8) PressLight [Wei et al., 2019a], a RL based method motivated by MaxPressure; (9) CoLight [Wei et al., 2019b] that uses GAT for communication; (10) MPLight [Chen et al., 2020], a state-of-the-art M-TSC method that combines FRAP [Zheng et al., 2019a] and PressLight; and (11) AttendLight [Oroojlooy et al., 2020] that uses attention and LSTM to select best actions.

Performance Metrics. Following existing studies [Oroojlooy et al., 2020; Chen et al., 2020], we adopt the average travel time of all the vehicles as the performance metric to evaluate the performance of different control methods. We also evaluate other commonly-used metrics such as average delay and throughput, as well as their standard deviation in our technique report [Jiang et al., 2022].

5.2 Evaluation Results

Overall Performance. Table 1 reports the overall evaluating results. We focus on the results without UniComm, i.e., all competitors communicate in their original way, and UniLight runs without any communication. The numbers in bold indicate the best performance. We observe that RL based methods achieve better results than traditional methods in public datasets. However, in a complicated environment like SH$_1$, agents may fail to learn a valid policy, and accordingly RL based methods might perform much worse than the traditional methods. UniLight performs the best in almost all datasets, and it demonstrates significant advantages in complicated environments. It improves the average performance by 8.2% in three public datasets and 35.6% in more compli-
Table 1: The overall performance (average travel time) of UniLight and its competitors with and without UniComm.

| Datasets | JN       | HZ       | NY       | SH1      | SH2      |
|----------|----------|----------|----------|----------|----------|
| SOTL     | 420.70   | 451.62   | 1137.29  | 2562.73  | 2084.93  |
| MaxPressure | 434.75   | 408.47   | 243.59   | 4843.07  | 788.55   |
| MaxBand  | 378.45   | 458.00   | 1038.67  | 5675.98  | 4501.39  |
| TFP-TCC  | 362.70   | 400.24   | 1216.77  | 2403.66  | 1380.05  |
| MARLIN   | 409.27   | 388.00   | 1274.23  | 6623.83  | 5373.53  |

| DRL Algorithms | MA-A2C | MA-PPO | PressLight | CoLight | MPLight | AttendLight |
|----------------|--------|--------|------------|---------|---------|-------------|
| Hidden State   | 355.29 | 552.64 | 460.18     | 394.61  | 384.04  | 361.94      |
| UniComm        | 359.73 | 558.41 | 463.92     | 398.47  | 387.94  | 364.31      |
| No Com.        | 365.85 | 564.24 | 470.86     | 402.79  | 390.83  | 367.66      |

| UniLight       | 325.47 | 323.03 | 180.72     | 159.88  | 208.06  |

Table 2: Compare UniComm with hidden state in average travel time.

| Datasets | JN       | HZ       | NY       | SH1      | SH2      |
|----------|----------|----------|----------|----------|----------|
| No Com.  | 335.85   | 324.24   | 186.85   | 232.69   | 209.89   |
| Hidden State | 330.99  | 323.88   | 180.99   | 1874.11  | 224.55   |
| UniComm  | 325.47   | 323.03   | 180.72   | 159.88   | **208.06** |

Figure 3: Phase prediction loss of different phase prediction target in JN and HZ.

The Impact of UniComm. As UniComm is universal for existing methods, we apply UniComm to the six representative RL based methods and re-evaluate their performance, with results listed in the bottom portion of Table 1. We observe that UniLight again achieves the best performance consistently. In addition, almost all RL based methods (including UniLight) are able to achieve a better performance with UniComm. This is because UniLight predicts the approaching vehicle number on \( R_{i,j} \) mainly by the hidden states of traffic movements covering \( R_{i,j} \). Consequently, \( A_{i} \) is able to produce predictions for neighboring \( A_{i} \) based on more important/relevant information. As a result, neighbors will receive customized results, which allow them to utilize the information in a more effective manner. In addition, agents only share predictions with their neighbors so the communication information and the observation dimension remain small. This allows existing RL based methods to outperform their original versions whose communications are mainly based on hidden states. Some original methods perform worse with UniComm in certain experiments, e.g., PressLight in SH1. This is because these methods have unstable performance on some datasets with very large variance, which make the results unstable. Despite of a small number of outliers, UniLight makes consistent boost on all existing methods.

Phase Prediction Target Evaluation. As mentioned previ-ously, instead of directly using current phase action to calculate phase prediction loss \( L_{p} \), we use actions stored in replay buffer. To evaluate its effectiveness, we plot the curve of \( L_{p} \) in Figure 3. Due to space limitation, we only show the curve of datasets JN and HZ. Current and Replay refer to the use of the action taken by the current network and that stored in the replay buffer respectively when calculating \( L_{p} \). The curve represents the volume prediction loss, i.e., the prediction accuracy during the training process. We can observe that when using stored actions as the target, the loss becomes smaller, i.e., it has learned better phase predictions. The convergence speed of phase prediction loss with stored actions is slower at the beginning of the training. This is because to perform more exploration, most actions are random at the beginning, which is hard to predict.

UniComm vs. Hidden States. To further evaluate the effectiveness of UniComm from a different angle, we introduce another version of UniLight that uses a 32-dimension hidden state for communication, the same as [Wei et al., 2019b]. In total, there are three different versions of UniLight evaluated, i.e., No Com., Hidden State, and UniComm. As the names suggest, No Com. refers to the version without sharing any information as there is no communication between agents; Hidden State refers to the version sharing 32-dimension hidden state; and UniComm refers to the version that implements UniComm. The average travel time of all the vehicles under these three variants of UniLight is listed in Table 2. Note, Hidden State shares the most amount of information and is able to improve the performance, as compared with version of No Com.. Unfortunately, the amount of information shared is not proportional to the performance improvement it can achieve. For example, UniLight with UniComm performs better than the version with Hidden State, although it shares less amount of information. This further verifies our argument that the content and the importance of the shared information is much more important than the amount.

6 Conclusion

In this paper, we propose a novel communication form UniComm for decentralized multi-agent learning based M-TSC problem. It enables each agent to share the prediction of approaching vehicles with its neighbors via communication. We also design UniLight to predict Q-value based on UniComm. Experimental results demonstrate that UniComm is universal for existing M-TSC methods, and UniLight outperforms existing methods in both simple and complex environments.

Acknowledgements

This research is supported in part by the National Natural Science Foundation of China under grant 62172107.
