Multi-Agent Deep Reinforcement Learning for Large-Scale Traffic Signal Control

Tianshu Chu, Jie Wang, Lara Codecà, and Zhaojian Li, *Member, IEEE*

***Abstract*— Reinforcement learning (RL) is a promising data-driven approach for adaptive traffic signal control (ATSC) in complex urban traffic networks, and deep neural networks further enhance its learning power. However, the centralized RL is infeasible for large-scale ATSC due to the extremely high dimension of the joint action space. The multi-agent RL (MARL) overcomes the scalability issue by distributing the global control to each local RL agent, but it introduces new challenges: now, the environment becomes partially observable from the viewpoint of each local agent due to limited communication among agents. Most existing studies in MARL focus on designing efficient communication and coordination among traditional Q-learning agents. This paper presents, for the first time, a fully scalable and decentralized MARL algorithm for the state-of-the-art deep RL agent, advantage actor critic (A2C), within the context of ATSC. In particular, two methods are proposed to stabilize the learning procedure, by improving the observability and reducing the learning difficulty of each local agent. The proposed multi-agent A2C is compared against independent A2C and independent Q-learning algorithms, in both a large synthetic traffic grid and a large real-world traffic network of Monaco city, under simulated peak-hour traffic dynamics. The results demonstrate its optimality, robustness, and sample efficiency over the other state-of-the-art decentralized MARL algorithms.**

***Index Terms*— Adaptive traffic signal control, reinforcement learning, multi-agent reinforcement learning, deep reinforcement learning, actor-critic.**

I. Introduction

As a consequence of population growth and urbanization, the transportation demand is steadily rising in the metropolises worldwide. The extensive routine traffic volumes bring pressures to existing urban traffic infrastructure, resulting in everyday traffic congestions. Adaptive traffic signal control (ATSC) aims for reducing potential congestions in saturated road networks, by adjusting the signal timing according to real-time traffic dynamics. Early-stage ATSC methods solve optimization problems to find efficient coordination and control policies. Successful products, such as SCOOT [1] and SCATS [2], have been installed in hundreds of cities across the world. OPAC [3] and PRODYN [4] are similar methods, but their relatively complex computation makes the implementation less popular. Since the 90s, various interdisciplinary techniques have been applied to ATSC, such as fuzzy logic [5], genetic algorithm [6], and immune network algorithm [7].

Reinforcement learning (RL), formulated under the framework of Markov decision process (MDP), is a promising alternative to learn ATSC based on real-world traffic measurements [8]. Unlike traditional model-driven approaches, RL does not rely on heuristic assumptions and equations. Rather it directly fits a parametric model to learn the optimal control, based on its experience interacting with the complex traffic systems. Traditional RL fits simple models such as piece-wise constant table and linear regression (LR) [9], leading to limited scalability or optimality in practice. Recently, deep neural networks (DNNs) have been successfully applied to enhance the learning capacity of RL on complex tasks [10].

To utilize the power of deep RL, appropriate RL methods need to be adapted. There are three major methods: value-based, policy-based, and actor-critic methods. In value-based methods, such as *Q-learning*, the long-term state-action value function is parameterized and updated using step-wise experience [11]. Q-learning is off-policy,[1] so it enjoys efficient updating with bootstrapped sampling of *experience replay*. However its update is based on one-step *temporal difference*, so the good convergence relies on a stationary MDP transition, which is less likely in ATSC. As the contrast, in policy-based methods, such as REINFORCE, the policy is directly parameterized and updated with sampled episode return [12]. REINFORCE is on-policy so the transition can be nonstationary within each episode. The actor-critic methods further reduce the bias and variance of policy-based methods by using another model to parameterize the value function [13]. A recent work has demonstrated that actor-critic outperforms Q-learning in ATSC with centralized LR agents [14]. This paper focuses on the state-of-the-art *advantage actor-critic* (A2C) where DNNs are used for both policy and value approximations [15].

Even though DNNs have improved the scalability of RL, training a centralized RL agent is still infeasible for large-scale ATSC. First, we need to collect all traffic measurements in the network and feed them to the agent as the global state. This centralized state processing itself will cause high latency

Manuscript received June 18, 2018; revised October 21, 2018 and December 26, 2018; accepted February 16, 2019. Date of publication March 15, 2019; date of current version February 28, 2020. The work of L. Codecà was supported in part by the French National Research Agency (ANR) Project ANR-11-LABX-0031-01, and in part by the EURECOM partners: BMW Group; IABG; Monaco Telecom; Orange; SAP; ST Microelectronics; and Symantec. The Associate Editor for this paper was I. Papamichail. *(Corresponding author: Tianshu Chu.)*

T. Chu and J. Wang are with the Department of Civil and Environmental Engineering, Stanford University, CA 94305 USA (e-mail: cts198859@hotmail.com; jiewang@stanford.edu).

L. Codecà is with the Communication Systems Department, EURECOM, 06904 Sophia-Antipolis, France (e-mail: codeca@eurecom.fr).

Z. Li is with the Department of Mechanical Engineering, Michigan State University, East Lansing, MI 48824 USA (e-mail: lizhaoj1@egr.msu.edu).

Digital Object Identifier 10.1109/TITS.2019.2901791

[1] In off-policy learning, the behavior policy for sampling the experience is different from the target policy to be optimized

and failure rate in practice, and the topological information of the traffic network will be lost. Further, the joint action space of the agent grows exponentially in the number of signalized intersections. Therefore, it is efficient and natural to formulate ATSC as a cooperative multi-agent RL (MARL) problem, where each intersection is controlled by a local RL agent, upon local observation and limited communication. MARL has a long history and has mostly focused on Q-learning, by distributing the global Q-function to local agents. One approach is to design a coordination rule based on the tradeoff between optimality and scalability [16], [17]. A simpler and more common alternative is *independent* Q-learning (IQL) [18], in which each local agent learns its own policy independently, by modeling other agents as parts of the environment dynamics. IQL is completely scalable, but it has issue on convergence, since now the environment becomes more partially observable and nonstationary, as other agents update their policies. This issue was addressed recently for enabling experience replay in deep MARL [19].

To the best of our knowledge, this is the first paper that formulates independent A2C (IA2C) for ATSC, by extending the idea of IQL on A2C. In order to develop a stable and robust IA2C system, two methods are further proposed to address the partially observable and nonstationary nature of IA2C, under limited communication. First, we include the observations and *fingerprints* of neighboring agents in the state, so that each local agent has more information regarding the regional traffic distribution and cooperative strategy. Second, we introduce a *spatial* discount factor to scale down the observation and reward signals of neighboring agents, so that each local agent focuses more on improving traffic nearby. From the convergence aspect, the first approach increases the fitting power while the second approach reduces the fitting difficulty. We call the stabilized IA2C the multi-agent A2C (MA2C). MA2C is evaluated in both a synthetic large traffic grid and a real-world large traffic network, with delicately designed traffic dynamics for ensuring a certain difficulty level of MDP. Numerical experiments confirm that, MA2C outperforms IA2C and state-of-the-art IQL algorithms in robustness and optimality. The code of this study is open sourced.[2]

II. Background

A. Reinforcement Learning

RL learns to maximize the long-term return of a MDP. In a fully observable MDP, the agent observes the true state of the environment $s_t \in \mathcal{S}$ at each time t, performs an action $u_t \in \mathcal{U}$ according to a policy $\pi(u|s)$. Then transition dynamics happens $s_{t+1} \sim p(\cdot|s_t, u_t)$, and an immediate step reward $r_t = r(s_t, u_t, s_{t+1})$ is received. In an infinite-horizon MDP, sampled total return under policy π is $R_t^\pi = \sum_{\tau=t}^{\infty} \gamma^{\tau-t} r_\tau$, where $\gamma \in [0, 1)$ is a discount factor. The expected total return is represented as its *Q-function* $Q^\pi(s, u) = \mathbb{E}[R_t^\pi | s_t = s, u_t = u]$. The optimal Q-function $Q^* = \max_\pi Q^\pi$ yields an optimal greedy policy $\pi^*(u|s) : u \in \text{argmax}_{u'} Q^*(s, u')$, and Q^* is obtained by solving the Bellman equation $\mathcal{T}Q^* = Q^*$ [20], with a *dynamic programming* (DP) operator $\mathcal{T}$:

$$\mathcal{T}Q(s, u) = r(s, u) + \gamma \sum_{s' \in \mathcal{S}} p(s'|s, u) \max_{u' \in \mathcal{U}} Q(s', u'), \quad (1)$$

where $r(s, u) = \mathbb{E}'_s r(s, u, s')$ is the expected step reward. In practice, r and p are unknown to the agent so the above *planning* problem is not well defined. Instead, RL performs data-driven DP based on the sampled experience (s_t, u_t, s'_t, r_t).

1) Q-Learning: Q-learning is the fundamental RL method that fits the Q-function with a parametric model Q_θ, such as Q-value table [11], LR [9], or DNN [10]. Given experience (s_t, u_t, s'_t, r_t), Eq. (1) is used to estimate $\hat{\mathcal{T}}Q_{\theta^-}(s_t, u_t) = r_t + \gamma \max_{u' \in \mathcal{U}} Q_{\theta^-}(s'_t, u')$ using a frozen recent model θ^-, then temporal difference $(\hat{\mathcal{T}}Q_{\theta^-} - Q_\theta)(s_t, u_t)$ is used to updated θ. A behavior policy, such as $\epsilon-$greedy, is used in Q-learning to explore rich experience to reduce the regression variance. Experience replay is applied in deep Q-learning to reduce the variance furthermore by sampling less correlated experience in each minibatch.

2) Policy Gradient: Policy gradient (REINFORCE) directly fits the policy with a parameterized model π_θ [21]. Each update of θ increases the likelihood for selecting the action that has the high "optimality", measured as the sampled total return. Thus the loss is

$$\mathcal{L}(\theta) = -\frac{1}{|B|} \sum_{t \in B} \log \pi_\theta(u_t|s_t) \hat{R}_t, \quad (2)$$

where each minibatch $B = \{(s_t, u_t, s'_t, r_t)\}$ contains the experience trajectory, *i.e.*, $s'_t = s_{t+1}$, and each return is estimated as $\hat{R}_t = \sum_{\tau=t}^{t_B-1} \gamma^{\tau-t} r_\tau$, here t_B is the last step in minibatch. Policy gradient does not need a behavior policy since $\pi_\theta(u|s)$ naturally performs exploration and exploitation. Further, it is robust to nonstationary transitions within each trajectory since it directly uses return instead of estimating it recursively by $\hat{\mathcal{T}}$. However it suffers from high variance as $\hat{R}_t$ is much more noisy than fitted return $Q^{\pi_{\theta^-}}$.

3) Advantage Actor-Critic: A2C improves the policy gradient by introducing a *value* regressor V_w to estimate $\mathbb{E}[R_t^\pi | s_t = s]$ [13]. First, it reduces the bias of sampled return by adding the value of the last state $R_t = \hat{R}_t + \gamma^{t_B-t} V_{w^-}(s_{t_B})$; Second, it reduces the variance of sampled return by using $A_t := R_t - V_{w^-}(s_t)$, which is interpreted as the sampled advantage $A^\pi(s, u) := Q^\pi(s, u) - V^\pi(s)$. Then Eq. (2) becomes

$$\mathcal{L}(\theta) = -\frac{1}{|B|} \sum_{t \in B} \log \pi_\theta(u_t|s_t) A_t. \quad (3)$$

The loss function for value updating is:

$$\mathcal{L}(w) = \frac{1}{2|B|} \sum_{t \in B} (R_t - V_w(s_t))^2. \quad (4)$$

B. Multi-Agent Reinforcement Learning

In ATSC, multiple signalized intersection agents participate a cooperative game to optimize the global network traffic objectives. Consider a multi-agent network $G(\mathcal{V}, \mathcal{E})$, where each agent $i \in \mathcal{V}$ performs a discrete action $u_i \in \mathcal{U}_i$, communicates to a neighbor via edge $ij \in \mathcal{E}$, and shares the global

[2]See https://github.com/cts198859/deeprl_signal_control

reward $r(s,u)$. Then the joint action space is $\mathcal{U} = \times_{i\in\mathcal{V}}\mathcal{U}_i$, which makes centralized RL infeasible. MARL, mostly formulated in the context of Q-learning, distributes the global action to each local agent by assuming the global Q-function is decomposable $Q(s,u) = \sum_{i\in\mathcal{V}} Q_i(s,u)$. To simplify the notation, we assume each local agent can observe the global state, and we will relax this assumption for ATSC in Section IV and V.

Coordinated Q-learning is one MARL approach that performs iterative message passing or control syncing among neighboring agents to achieve desired tradeoff between optimality and scalability. In other words, $Q_i(s,u) \approx Q_i(s,u_i) + \sum_{j\in\mathcal{V}_i} M_j(s,u_j,u_{\mathcal{V}_j})$, where $\mathcal{V}_i$ is the neighborhood of agent i, and M_j is the message from neighbor j, regarding the impact of $u_i \in u_{\mathcal{V}_j}$ on its local traffic condition. [22] applied variable elimination after passing Q-function as the message, while [17] proposed a max-plus message passing. This approach (1) requires additional computation to obtain the coordinated control during execution, and (2) requires heuristics and assumptions to decompose Q-function and formulate message-passing, which can be potentially learned by IQL described shortly.

IQL is the most straightforward and popular approach, in which each local Q-function only depends on the local action, *i.e.*, $Q_i(s,u) \approx Q_i(s,u_i)$ [18]. IQL is completely scalable, but without message passing, it suffers from partial observability and non-stationary MDP, because it implicitly formulates all other agents' behavior as part of the environment dynamics while their policies are continuously updated during training. . To address this issue, each local agent needs the information of other agents' policies. Reference [23] included the policy network parameter of each other agent for fitting local Q-function, *i.e.*, $Q_i(s,u) \approx Q_{\theta_i}(s,u_i,\theta_{-i})$, while [19] included low-dimensional fingerprints, *i.e.*, $Q_i(s,u) \approx Q_{\theta_i}(s,u_i,x_{-i})$. To handle the nonstationary transition in experience replay, *importance sampling* is applied to estimate the temporal difference of other agents' policies between the sampled time t and the updating time τ, as $\pi_{\tau,-i}(u_{-i}|s)/\pi_{t,-i}(u_{-i}|s)$, and to adjust the batch loss. MARL with A2C has not been formally addressed in literature, and will be covered in Section IV.

III. Related Work

The implementation of RL has been extensively studied in ATSC. Tabular Q-learning was the first RL algorithm applied, at an isolated intersection [24]. Later, LR Q-learning was adapted for scalable fitting over continuous states. References [25] and [26] designed heuristic state features, while [27] integrated macroscopic fundamental diagram to obtain more informative features. However, LR was too simple to capture the Q-function under complex traffic dynamics. Thus kernel method was applied to extract nonlinear features from low-dimensional states [28]. Kernel method was also applied in LR actor-critic recently, under realistically simulated traffic environments [14]. Alternatively, natural actor-critic was applied to improve the fitting accuracy of LR in ATSC [29]. Deep RL was implemented recently, but most of them had impractical assumptions or oversimplified traffic environments. References [30] and [31] verified the superior fitting power of deep Q-learning and deep deterministic policy gradient, respectively, under simplified traffic environment. Reference [32] applied deep Q-learning in a more realistic traffic network, but with infeasible microscopic discrete traffic state. Reference [33] explored A2C with different types of state information. Due to the scalability issue, most centralized RL studies conducted experiments in either isolated intersections or small traffic networks.

Despite rich history of RL, only a few studies have addressed MARL in ATSC, and most of them have focused on Q-learning. Reference [34] applied model-based tabular IQL to each intersection while [35] extended LR IQL to dynamically clustered regions to improve observability. Reference [36] studied LR IQL and its on-policy version independent SARSA learning, with observability improved by neighborhood information sharing. Reference [37] applied deep IQL, with the partial observability addressed with *transfer planning*, but the states were infeasible and the simulated traffic environments were oversimplified in that study. Alternatively, coordinated Q-learning was implemented with various message-passing methods. Reference [38] designed a heuristic neighborhood communication for tabular Q-learning agents, where each message contained the estimated neighbor policies. Reference [39] proposed a junction-tree based hierarchical message-passing rule to coordinate tabular Q-learning agents. Reference [40] applied the max-sum communication for LR Q-learning agents, where each message indicated the impact of a neighbor agent on each local Q-value.

To summarize, traditional RL has been widely applied in ATSC, and some works have proposed realistic state measurements and decent traffic dynamics based on insightful domain-specific knowledge. However, benchmark traffic environments are still missing for fair comparisons across proposed algorithms. On the other hand, there is still no comprehensive and realistic studies proposed for implementing deep RL in practical ATSC. MARL has been addressed in a few works, mostly under the context of Q-learning.

IV. Multi-Agent Advantage Actor-Critic

MARL is mostly addressed in the context of Q-learning. In this section, we first formulate IA2C by extending the observations of IQL to the actor-critic method. Furthermore, we propose two approaches to stabilizing IA2C as MA2C. The first approach is inspired from the works for stabilizing IQL [19], [23], where the recent policies of other agents are informed to each local agent. The second approach proposes a novel spatial discount factor to scale down the signals from agents far away. In other words, each local value function is smoother and more correlated to local states, and each agent focuses more on improving local traffic condition. As a result, the convergence becomes more stable even under limited communication and partial observation. This section proposes a general MA2C framework under limited neighborhood communication, while its implementation details for ATSC will be described in Section V.

A. Independent A2C

In a multi-agent network $G(\mathcal{V}, \mathcal{E})$, i and j are neighbors if there is an edge between them. The neighborhood of i is denoted as $\mathcal{N}_i$ and the local region is $\mathcal{V}_i = \mathcal{N}_i \cup i$. Also, the distance between any two agents $d(i, j)$ is measured as the minimum number of edges connecting them. For example, $d(i, i) = 0$ and $d(i, j) = 1$ for any $j \in \mathcal{N}_i$. In IA2C, each agent learns its own policy π_{θ_i} and the corresponding value function V_{w_i}.

We start with a strong assumption where the global reward and state are shared among agents. Then centralized A2C updating can be easily extended to IA2C, by estimating local return as

$$R_{t,i} = \hat{R}_t + \gamma^{t_B - t} V_{w_i^-}(s_{t_B} | \pi_{\theta_{-i}^-}). \tag{5}$$

The value gradient $\nabla \mathcal{L}(w_i)$ is consistent since $\hat{R}_t$ is sampled from the same stationary policy π_{θ^-}. To obtain policy gradient $\nabla \mathcal{L}(\theta_i)$, $V_{w_i} : \mathcal{S} \times \mathcal{U}_i \to \mathbb{R}$ is served as the estimation of marginal impact of π_{θ_i} on the future return. However, if each agent follows Eq. (5), each value gradient will be identical towards the global value function V^{π} instead of the local one $V^{\pi_i} = \mathbb{E}_{\pi_{-i}} V^{\pi}$. As far as θ_{-i} is fixed, π_{θ_i} will still converge to the best according policy under this updating, and optimal policy $\pi_{\theta_i^*}$ can be achieved if $\theta_{-i} = \theta_{-i}^*$. However, when θ_{-i} is actively updated, the policy gradient may be inconsistent across minibatches, since the advantage is conditioned on changing $\pi_{\theta_{-i}}$, even it is stationary per trajectory.

Global information sharing is infeasible in real-time ATSC due to the communication latency, so we assume the communication is limited to each local region. In other words, local policy and value regressors take $s_{t,\mathcal{V}_i} := \{s_{t,j}\}_{j \in \mathcal{V}_i}$ instead of s_t as the input state. Global reward is still allowed since it is only used in offline training. Eq. (5) is valid here, by replacing the value estimation of the last state with $V_{w_i^-}(s_{t_B, \mathcal{V}_i} | \pi_{\theta_{-i}^-})$. Then the value loss Eq. (4) becomes

$$\mathcal{L}(w_i) = \frac{1}{2|B|} \sum_{t \in B} \left(R_{t,i} - V_{w_i}(s_{t,\mathcal{V}_i}) \right)^2. \tag{6}$$

Clearly, V_{w_i} suffers from partial observability, as $s_{t,\mathcal{V}_i}$ is a subset of s_t while $\mathbb{E} R_{t,i}$ depends on s_t. Similarly, the policy loss Eq. (3) becomes

$$\mathcal{L}(\theta_i) = -\frac{1}{|B|} \sum_{t \in B} \log \pi_{\theta_i}(u_{t,i} | s_{t,\mathcal{V}_i}) A_{t,i}, \tag{7}$$

where $A_{t,i} = R_{t,i} - V_{w_i^-}(s_{t,\mathcal{V}_i})$. The nonstationary updating issue remains, since $R_{t,i}$ is conditioned on the current policy $\pi_{\theta_{-i}^-}$, while both θ_i^- and w_i^- are updated under the previous policy $\pi_{\theta'_{-i}}$. This inconsistency effect can be mitigated if each local policy updating is smooth, *i.e.*, the KL divergence $D_{KL}(\pi_{\theta_{-i}^-} || \pi_{\theta_{-i}})$ is small enough, but it will slow down the convergence. Partial observability also exists as $\pi_{\theta_i}(\cdot | s_{t,\mathcal{V}_i})$ cannot fully capture the impact of $\hat{R}_t$.

B. Multi-Agent A2C

In order to stabilize IA2C convergence and enhance its fitting power, we propose two approaches. First, we include information of neighborhood policies to improve the observability of each local agent. Second, we introduce a spatial discount factor to weaken the state and reward signals from other agents. In IQL, additional information is included to represent the other agents' behavior policies. Reference [23] directly includes the Q-value network parameters θ_{-i}^-, while [19] includes low-dimensional fingerprints, such as ϵ of the ϵ-greedy exploration and the number of updates so far. Fortunately, the behavior policy is explicitly parameterized in A2C, so a natural approach is including probability simplex of policy $\pi_{\theta_{-i}^-}$. Under limited communication, we include sampled latest policies of neighbors $\pi_{t-1,\mathcal{N}_i} = [\pi_{t-1,j}]_{j \in \mathcal{N}_i}$ in the DNN inputs, besides the current state $s_{t,\mathcal{V}_i}$. The sampled local policy is calculated as

$$\pi_{t,i} = \pi_{\theta_i^-}(\cdot | s_{t,\mathcal{V}_i}, \pi_{t-1,\mathcal{N}_i}). \tag{8}$$

Rather than long-term neighborhood behavior, the real-time recent neighborhood policy is informed to each local agent. This is based on two ATSC facts: 1) the traffic state is changing slowly in short windows, so the current step policy is very similar to last step policy. 2) the traffic state dynamics is Markovian, given the current state and policy.

Even if the local agent knows the local region state and the neighborhood policy, it is still difficult to fit the global return by local value regressor. To relax the global cooperation, we assume the global reward is decomposable as $r_t = \sum_{i \in \mathcal{V}} r_{t,i}$, which is mostly valid in ATSC. Then we introduce a spatial discount factor α to adjust the global reward for agent i as

$$\tilde{r}_{t,i} = \sum_{d=0}^{D_i} \left(\sum_{j \in \mathcal{V} | d(i,j) = d} \alpha^d r_{t,j} \right), \tag{9}$$

where D_i is the maximum distance from agent i. Note α is similar to the temporal discount factor γ in RL, rather it scales down the signals in spatial order instead of temporal order. Compared to sharing the same global reward across agents, the discounted global reward is more flexible for the trade-off between greedy control ($\alpha = 0$) and cooperative control ($\alpha = 1$), and is more relevant for estimating the "advantage" of local policy π_{θ_i}. Similarly, we use α to discount the neighborhood states as

$$\tilde{s}_{t,\mathcal{V}_i} = [s_{t,i}] \cup \alpha [s_{t,j}]_{j \in \mathcal{N}_i}. \tag{10}$$

Given the discounted global reward, we have $\hat{R}_{t,i} = \sum_{\tau=t}^{t_B - 1} \gamma^{\tau - t} \tilde{r}_{\tau,i}$, and the local return Eq. (5) becomes

$$\tilde{R}_{t,i} = \hat{R}_{t,i} + \gamma^{t_B - t} V_{w_i^-}(\tilde{s}_{t_B, \mathcal{V}_i}, \pi_{t_B - 1, \mathcal{N}_i} | \pi_{\theta_{-i}^-}). \tag{11}$$

The value loss Eq. (6) becomes

$$\mathcal{L}(w_i) = \frac{1}{2|B|} \sum_{t \in B} \left(\tilde{R}_{t,i} - V_{w_i}(\tilde{s}_{t,\mathcal{V}_i}, \pi_{t-1,\mathcal{N}_i}) \right)^2. \tag{12}$$

This updating is more stable since (1) additional fingerprints $\pi_{t-1,\mathcal{N}_i}$ are input to V_{w_i} for fitting $\pi_{\theta_{-i}^-}$ impact, and (2) spatially discounted return $\tilde{R}_{t,i}$ is more correlated to local

Algorithm 1: Synchronous Multi-Agent A2C

Parameter: α, β, γ, T, $|B|$, η_w, η_θ.
Result: $\{w_i\}_{i\in\mathcal{V}}, \{\theta_i\}_{i\in\mathcal{V}}$.
1 **initialize** s_0, π_{-1}, $t \leftarrow 0$, $k \leftarrow 0$, $B = \emptyset$;
2 **repeat**
/* explore experience */
3 **for** $i \in \mathcal{V}$ **do**
4 **sample** $u_{t,i}$ from $\pi_{t,i}$ (Eq. (8));
5 **receive** $\tilde{r}_{t,i}$ (Eq. (9)) and $\tilde{s}_{t,i}$;
6 **end**
7 $B \leftarrow B \cup \{(t, \tilde{s}_{t,i}, \pi_{t,i}, u_{t,i}, \tilde{r}_{t,i}, \tilde{s}_{t+1,i})\}_{i\in\mathcal{V}}$;
8 $t \leftarrow t+1$, $k \leftarrow k+1$;
/* restart episode */
9 **if** $t = T$ **then**
10 **initialize** s_0, π_{-1}, $t \leftarrow 0$;
11 **end**
/* update A2C */
12 **if** $k = |B|$ **then**
13 **for** $i \in \mathcal{V}$ **do**
14 **estimate** $\hat{R}_{\tau,i}$, $\forall \tau \in B$;
15 **estimate** $\tilde{R}_{\tau,i}$, $\forall \tau \in B$;
16 **update** w_i with $\eta_w \nabla \mathcal{L}(w_i)$ (Eq. (12));
17 **update** θ_i with $\eta_\theta \nabla \mathcal{L}(\theta_i)$ (Eq. (13));
18 **end**
19 $B \leftarrow \emptyset$, $k \leftarrow 0$;
20 **end**
21 **until** *Stop condition reached*;

region observations $(\tilde{s}_{t,\mathcal{V}_i}, \pi_{t-1,\mathcal{N}_i})$. The policy loss Eq. (13) becomes

$$\mathcal{L}(\theta_i) = -\frac{1}{|B|}\sum_{t\in B}\Bigg(\log \pi_{\theta_i}(u_{t,i}|\tilde{s}_{t,\mathcal{V}_i}, \pi_{t-1,\mathcal{N}_i})\tilde{A}_{t,i} - \beta \sum_{u_i\in\mathcal{U}_i} \pi_{\theta_i}\log \pi_{\theta_i}(u_i|\tilde{s}_{t,\mathcal{V}_i}, \pi_{t-1,\mathcal{N}_i})\Bigg) \quad (13)$$

where $\tilde{A}_{t,i} = \tilde{R}_{t,i} - V_{w_i^-}(\tilde{s}_{t,\mathcal{V}_i}, \pi_{t-1,\mathcal{N}_i})$, and the additional regularization term is the entropy loss of policy π_{θ_i} for encouraging the early-stage exploration. This new advantage emphasizes more on the marginal impact of π_{θ_i} on traffic state within local region.

Algorithm 1 illustrates the MA2C algorithm under the synchronous updating. T is the planning horizon per episode, $|B|$ is the minibatch size, η_w and η_θ are the learning rates for critic and actor networks. First, each local agent collects the experience by following the current policy, until enough samples are collected for minibatch updating (lines 3 to 8). If the episode is terminated in the middle of minibatch collection, we simply restart a new episode (lines 9 to 11). However the termination affects the return estimation. If T is reached in the middle, the trajectory return (line 14) should be adjusted as

$$\hat{R}_{t,i} = \begin{cases} \sum_{\tau=t}^{T-1} \gamma^{\tau-t}\tilde{r}_{\tau,i} & \text{before reset,} \\ \sum_{\tau=t}^{t_B-1} \gamma^{\tau-t}\tilde{r}_{\tau,i} & \text{after reset.} \end{cases} \quad (14)$$

If T is reached at the end, $\tilde{R}_{t,i} = \hat{R}_{t,i}$ (line 15), without the value estimation on future return in Eq. (11). Next, the mini-batch gradients are applied to update each actor and critic networks (lines 16, 17), with constant or adaptive learning rates. Usually the first order gradient optimizers are used, such as stochastic gradient descent, RMSprop, and Adam. Finally, the training process is terminated if the maximum step is reached or a certain stop condition is triggered.

V. MA2C for Traffic Signal Control

This section describes the implementation details of MA2C for ATSC, under the microscopic traffic simulator SUMO [41]. Specifically, we address the action, state, reward definitions, the A2C network structures and normalizations, the A2C training tips, and the evaluation metrics.

A. MDP Settings

Consider a simulated traffic environment over a period of T_s seconds, we define Δt as the interaction period between RL agents and the traffic environment, so that the environment is simulated for Δt seconds after each MDP step. If Δt is too long, RL agents will not be adaptive enough. If Δt is too short, the RL decision will not be delivered on time due to computational cost and communication latency. Further, there will be safety concerns if RL control is switched too frequently. To further guarantee the safety, a yellow time $t_y < \Delta t$ is enforced after each traffic light switch. A recent work suggested $\Delta t = 10$s, and $t_y = 5$s [14]. In this paper, we set $\Delta t = 5$s, $t_y = 2$s to ensure more adaptiveness, resulting in a planning horizon of $T = T_s/\Delta t$ steps.

1) Action Definition: There are several standard action definitions, such as phase switch [38], phase duration [14], and phase itself [26]. We follow the last definition and simply define each local action as a possible phase, or red-green combinations of traffic lights at that intersection. This enables more flexible and direct ATSC by RL agents. Specifically, we predefine $\mathcal{U}_i$ as a set of all feasible phases for each intersection, and RL agent selects one of them to be implemented for a duration of Δt at each step.

2) State Definition: After combining the ideas of [26] and [14], we define the local state as

$$s_{t,i} = \{\texttt{wait}_t[l], \texttt{wave}_t[l]\}_{ji\in\mathcal{E}, l\in L_{ji}}, \quad (15)$$

where l is each incoming lane of intersection i. `wait` [s] measures the cumulative delay of the first vehicle, while `wave` [veh] measures the total number of approaching vehicles along each incoming lane, within 50m to the intersection. Both `wait` and `wave` can be obtained by near-intersection induction-loop detectors (ILD), which ensures real-time ATSC. To simplify the implementation, we use `laneAreaDetector` in SUMO to collect the state information.

3) Reward Definition: Various objectives are selected for ATSC, but an appropriate reward of MARL should be spatially decomposable and frequently measurable. Here we refine the definition in [26] as

$$r_{t,i} = -\sum_{ji\in\mathcal{E}, l\in L_{ji}} (\texttt{queue}_{t+\Delta t}[l] + a\cdot \texttt{wait}_{t+\Delta t}[l]), \quad (16)$$

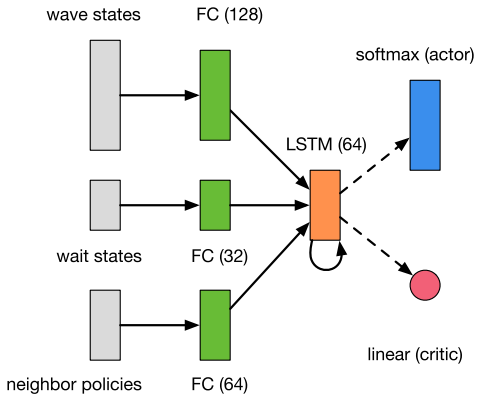

Fig. 1. Proposed DNN structures of MA2C in ATSC. Hidden layer size is indicated inside parenthesis.

where a [veh/s] is a tradeoff coefficient, and `queue` [veh] is the measured queue length along each incoming lane. Notably the reward is post-decision so both `queue` and `wait` are measured at time $t + \Delta t$. We prefer this definition over others like cumulative delay [38] and wave [14], since this reward is directly correlated to state and action, and emphasizes both traffic congestion and trip delay.

B. DNN Settings

1) Network Architecture: In practice, the traffic flows are complex spatial-temporal data, so the MDP may become non-stationary if the agent only knows the current state. A straightforward approach is to input all historical states to A2C, but it increases the state dimension significantly and may reduce the attention of A2C on recent traffic condition. Fortunately, *long-short term memory* (LSTM) is a promising DNN layer that maintains hidden states to memorize short history [42]. Thus we apply LSTM as the last hidden layer to extract representations from different state types. Also, we train actor and critical DNNs separately, instead of sharing lower layers between them. Fig. 1 illustrates the DNN structure, where `wave`, `wait`, and neighbor policies are first processed by separate fully connected (FC) layers. Then all hidden units are combined and input to a LSTM layer. The output layer is softmax for actor and linear for critic. For DNN training, we use the state-of-the-art orthogonal initializer [43] and RMSprop as the gradient optimizer.

2) Normalization: Normalization is important for training DNN. For each of `wave` and `wait` states, we run a greedy policy to collect the statistics for a certain traffic environment, and use it to obtain an appropriate normalization. To prevent gradient explosion, all normalized states are clipped to [0, 2], and each gradient is capped at 40. Similarly, we normalize the reward and clip it to [-2, 2] to stabilize the minibatch updating.

VI. Numerical Experiments

MARL based ATSC is evaluated in two SUMO-simulated traffic environments: a 5×5 synthetic traffic grid, and a real-world 30-intersection traffic network extracted from Monaco city [44], under time-variant traffic flows. This section aims to design challenging and realistic traffic environments for interesting and fair comparisons across controllers.

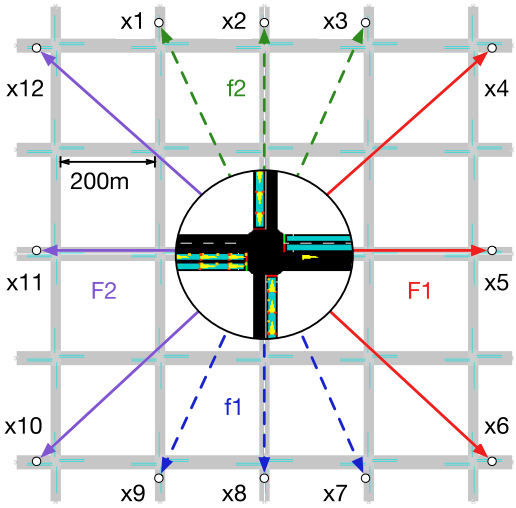

Fig. 2. A traffic grid of 25 intersections, with an example intersection shown inside circle. Time variant major and minor traffic flow groups are shown as solid and dotted arrows.

A. General Setups

To demonstrate the efficiency and robustness of MA2C, we compare it to several state-of-the-art benchmark controllers. IA2C is the same as MA2C except the proposed stabilizing methods, so it follows the updating rules Eq. (6) and Eq. (7). IQL-LR is LR based IQL, which can be regarded as the decentralized version of [26]. IQL-DNN is the same as IQL-LR but uses DNN for fitting the Q-function. To ensure fair comparison, IQL-DNN has the same network structure, except the LSTM layer is replaced by a FC layer. This is because Q-learning is off-policy and it becomes meaningless to use LSTM on randomly sampled history. Finally, Greedy is a decentralized greedy policy that selects the phase associated with the maximum total green `wave` over all incoming lanes. All controllers have the same action space, state space, and interaction frequency.

We train all MARL algorithms over 1M steps, which is around 1400 episodes given episode horizon $T = 720$ steps. We then evaluate all controllers over 10 episodes. Different random seeds are used for generating different training and evaluation episodes, but the same seed is shared for the same episode. For MDP, we set $\gamma = 0.99$ and $\alpha = 0.75$; For IA2C and MA2C, we set $\eta_\theta = 5e{-}4$, $\eta_w = 2.5e{-}4$, $|B| = 120$, and $\beta = 0.01$ in Algorithm 1; For IQL, we set the learning rate $\eta_\theta = 1e{-}4$, the minibatch size $|B| = 20$, and the replay buffer size 1000. Note the replay buffer size has to be small due to the partial observability of IQL. Also, $\epsilon-$greedy is used as the behavior policy, with ϵ linearly decaying from 1.0 to 0.01 during the first half of training.

B. Synthetic Traffic Grid

1) Experiment Settings: The 5×5 traffic grid, as illustrated in Fig. 2, is formed by two-lane arterial streets with speed limit 20m/s and one-lane avenues with speed limit 11m/s. The action space of each intersection contains five possible phases: E-W straight phase, E-W left-turn phase, and three straight and left-turn phases for E, W, and N-S. Clearly, centralized

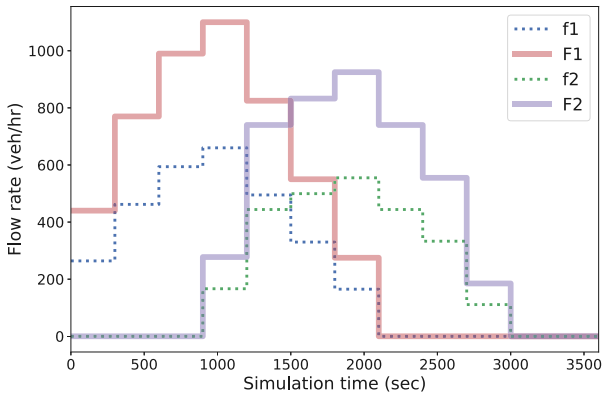

Fig. 3. Traffic flows vs simulation time within the traffic grid.

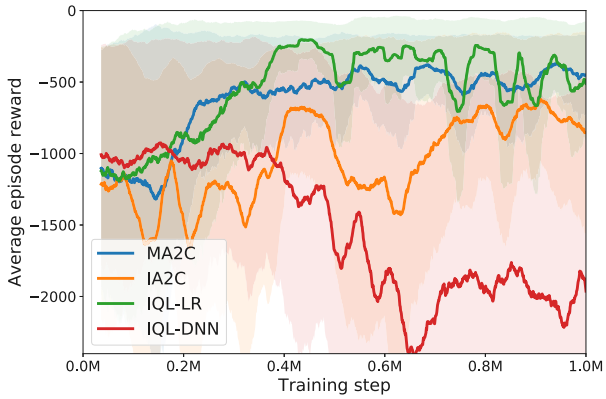

Fig. 4. MARL training curves for synthetic traffic grid.

RL agent is infeasible since the joint action space has the size of 5^{25}. To make the MDP problem challenging, four time-variant traffic flow groups are simulated. At beginning, three major flows F_1 are generated with origin-destination (O-D) pairs x_{10}-x_4, x_{11}-x_5, and x_{12}-x_6, meanwhile three minor flows f_1 are generated with O-D pairs x_1-x_7, x_2-x_8, and x_3-x_9. After 15 minutes, the volumes of F_1 and f_1 start to decrease, while their opposite flows (with swapped O-D pairs) F_2 and f_2 start to be generated, as shown in Fig. 3. Here the flows define the high-level traffic demand only, and the route of each vehicle is randomly generated during run-time. Regarding MDP settings, the reward coefficient a is 0.2veh/s, and the normalization factors of `wave`, `wait`, and reward are 5veh, 100s, and 2000veh, respectively.

2) Training Results: Fig. 4 plots the training curve of each MARL algorithm, where the solid line shows the average reward per training episode

$$\bar{R} = \frac{1}{T}\sum_{t=0}^{T-1}\left(\sum_{i\in\mathcal{V}} r_{t,i}\right), \tag{17}$$

and the shade shows its standard deviation. Typically, training curve increases and then converges, as RL learns from cumulated experience and finally achieves a local optimum. In Fig. 4, IQL-DNN is failed to learn, while IQL-LR achieves a performance as good as MA2C does. This may because DNN overfits the Q-function using the partially observed states, misleading the exploitation when ϵ decreases. On the other hand,

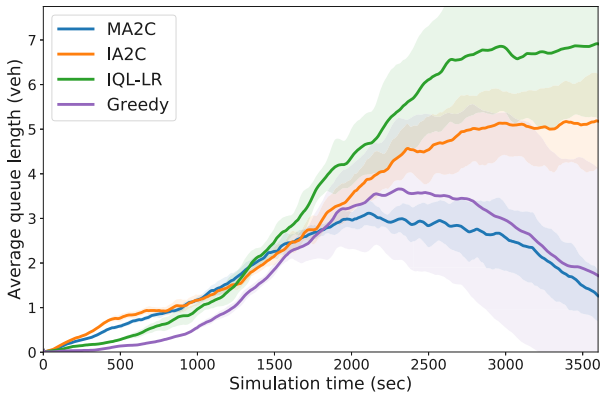

Fig. 5. Average queue length in synthetic traffic grid.

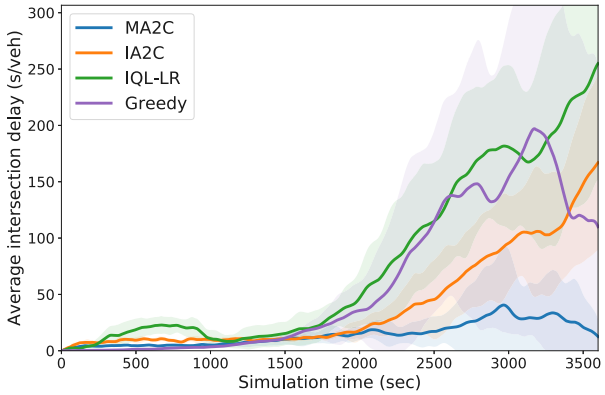

Fig. 6. Average intersection delay in synthetic traffic grid.

MA2C shows the best and the most robust learning ability, as its training curve steadily increases and then becomes stable with narrow shade.

3) Evaluation Results: IQL-DNN is not included in evaluation as its policy is meaningless. The average $\bar{R}$ over evaluation are -414, -845, -972, and -1409, for MA2C, IA2C, Greedy, and IQL-LR. Clearly MA2C outperforms other controllers for the given objective. Also, IQL-LR is failed to beat IA2C in this evaluation over more episodes, which may due to the high variance in the learned policy. Fig. 5 plots the average queue length of network at each simulation step, where the line shows the average and the shade shows the standard deviation across evaluation episodes. Both IQL-LR and IA2C fail to lean a sustainable policy to recover the congested network near the end. As contrast, a simple greedy policy does well for queue length optimization by maximizing the flow at each step, but its performance variation is high (see wide shade). MA2C leans a more stable and sustainable policy that achieves lower congestion level and faster recovery, by paying a higher queue cost when the network is less loaded at early stage.

Fig. 6 plots the average intersection delay of network over simulation time. As expected, both IQL-LR and IA2C have monotonically increasing delays as they are failed to recover from the congestion. However, IA2C is able to maintain a more slowly increasing delay when the traffic grid is less saturated, so its overall performance is still better than the greedy policy,

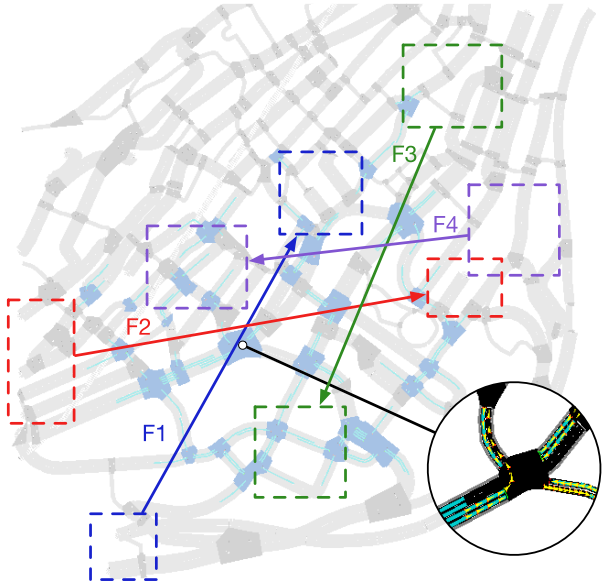

Fig. 7. Monaco traffic network, with signalized intersections colored in blue. Four traffic flow groups are illustrated by arrows, with origin and destination inside rectangular areas.

which does not explicitly optimize the waiting time. MA2C is able to maintain the delay at low level even at the peak.

C. Monaco Traffic Network

1) Experiment Settings: Fig. 7 illustrates the studied area of Monaco city for this experiment, with a variety of road and intersection types. There are 30 signalized intersections in total: 11 are two-phase, 4 are three-phase, 10 are four-phase, 1 is five-phase, and the reset 4 are six-phase. Further, in order to test the robustness and optimality of algorithms in challenging ATSC scenarios, intensive, stochastic, and time-variant traffic flows are designed to simulate the peak-hour traffic. Specifically, four traffic flow groups are generated as a multiple of "unit" flows of 325veh/hr, with randomly sampled O-D pairs inside given areas (see Fig. 7). Among them, F_1 and F_2 are simulated during the first 40min, as $[1, 2, 4, 4, 4, 4, 2, 1]$ unit flows with 5min intervals, while F_3 and F_4 are generated during a shifted time window from 15min to 55min.

As the MDP becomes challenging in this experiment, we remove `wait` terms in both reward and state, making the value function easier to be fitted. As the price, MARL algorithms will not learn to explicitly optimize the delay. The normalization factor of `wave` is 5veh, and that of reward is 20veh per intersection involved in reward calculation.

2) Training Results: Fig. 8 plots the training curves of MARL algorithms. IQL-DNN still does not learn anything, while IQL-LR does not converge, despite good performance in the middle of training. Again, both IA2C and MA2C converge to reasonable policies, and MA2C shows a faster and more stable convergence.

3) Evaluation Results: Table I summarizes the key performance metrics for comparing ATSC in evaluation. The measurements are first spatially aggregated at each time over evaluation episodes, then the temporal average and max are calculated. Except for trip delay, the values are calculated

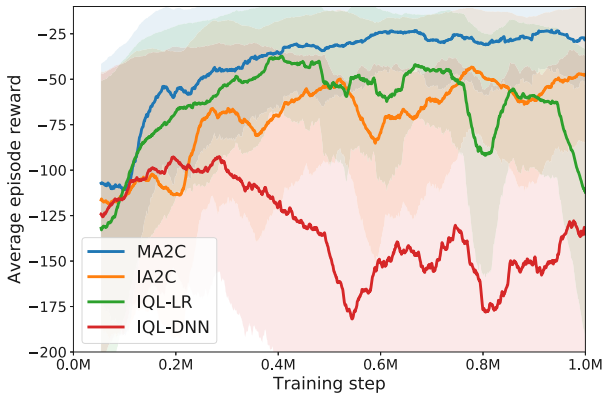

Fig. 8. MARL training curves for Monaco traffic network.

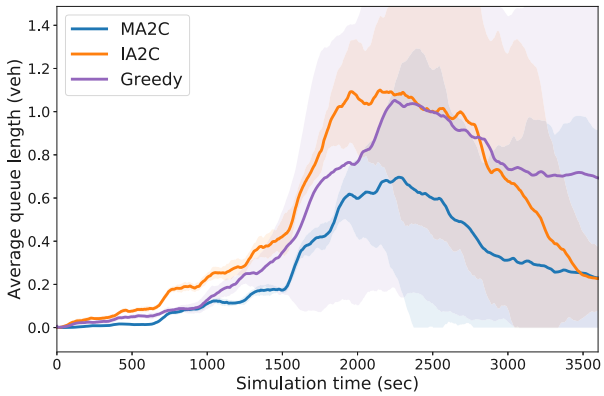

Fig. 9. Average queue length in Monaco traffic network.

over all evaluated trips. MA2C outperforms other controllers in almost all metrics. Unfortunately, other MARL algorithms are failed to beat the greedy policy in this experiment. Therefore extensive effort and caution is needed before the field deployment of any data-driven ATSC algorithm, regarding its robustness, optimality, efficiency, and safety.

IQL algorithms are not included in following analysis as their training performance is as bad as that of random exploration. Fig. 9 and Fig. 10 plot the average queue length and average intersection delay over simulation time, under different ATSC policies. As expected, both IA2C and Greedy are able to reduce the queue lengths after peak values, and IA2C achieves a better recovery rate. However, both of them are failed to maintain sustainable intersection delays. This may be because of the "central area" congestion after upstream intersections greedily maximizing their local flows. On the other hand, MA2C is able to achieve lower and more sustainable intersection delays, by distributing the traffic more homogeneously among intersections via coordination with shared neighborhood fingerprints.

Fig. 11 scatters the output (trip completion) flow and network vehicle accumulation for different ATSC algorithms. Macroscopic fundamental diagram (MFD) is present for each controller: when the network density is low, output increases as accumulation grows; when the network becomes more saturated, further accumulation will decrease the output, leading to a potential congestion. As we can see, compared to other

TABLE I
ATSC PERFORMANCE IN MONACO TRAFFIC NETWORK BEST VALUES ARE IN BOLD

Metrics	Temporal Averages					Temporal Peaks				
	Greedy	MA2C	IA2C	IQL-LR	IQL-DNN	Greedy	MA2C	IA2C	IQL-LR	IQL-DNN
reward	-41.8	**-31.4**	-54.6	-109.8	-151.8	-86.4	**-78.7**	-117.9	-202.1	-256.2
avg. queue length [veh]	0.51	**0.29**	0.52	1.19	1.57	1.08	**0.75**	1.16	2.21	2.69
avg. intersection delay [s/veh]	65.5	**23.5**	60.7	100.3	127.0	272	**104**	316	202	238
avg. vehicle speed [m/s]	6.06	**6.81**	5.36	4.04	2.07	**14.96**	14.26	14.26	14.26	13.98
trip completion flow [veh/s]	0.54	**0.67**	0.62	0.44	0.28	2.10	**2.40**	2.10	1.60	1.20
trip delay [s]	144	**114**	165	207	360	2077	**1701**	2418	2755	3283

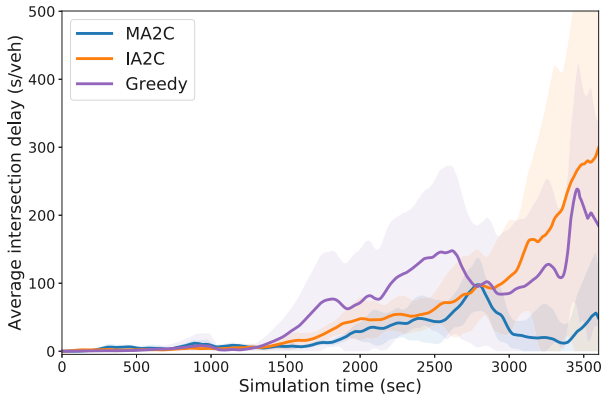

Fig. 10. Average intersection delay in Monaco traffic network.

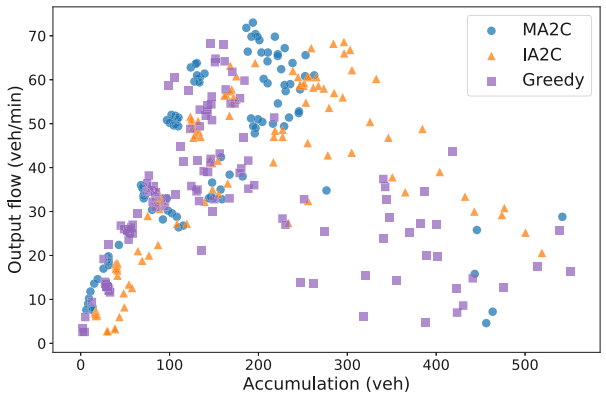

Fig. 11. Output flow vs vehicle accumulation scatter in Monaco traffic network. Each point is aggregated over 5min.

controllers, MA2C is able to maintain most points around the "sweet-spot", maximizing the utilization of network capacity.

VII. CONCLUSIONS

This paper proposed a novel A2C based MARL algorithm for scalable and robust ATSC. Specifically, 1) fingerprints of neighbors were adapted to improve observability; and 2) a spatial discount factor was introduced to reduce the learning difficulty. Experiments in a synthetic traffic grid and a Monaco traffic network demonstrated the robustness, optimality, and scalability of the proposed MA2C algorithm, which outperformed other state-of-the-art MARL algorithms.

Non-trivial future works are still remaining for the real-world deployment of proposed MARL algorithm. These include 1) improving the reality of traffic simulator to provide reliable training data regarding real-world traffic demand and dynamics; 2) improving the algorithm robustness on noisy and delayed state measurements from road sensors; 3) building a pipeline that can train and deploy deep MARL models to each intersection controller for a given traffic scenario; 4) improving the end-to-end pipeline latency, with a focus on the inference time and memory consumption of model query at each intersection, as well as the communication delay among neighboring intersections for state and fingerprint sharing.

REFERENCES

[1] P. B. Hunt, D. I. Robertson, R. D. Bretherton, and M. C. Royle, "The SCOOT on-line traffic signal optimisation technique," *Traffic Eng., Control*, vol. 23, no. 4, pp. 190–192, Apr. 1982.
[2] J. Y. K. Luk, "Two traffic-responsive area traffic control methods: SCAT and SCOOT," *Traffic Eng., Control*, vol. 25, no. 1, p. 14, 1984.
[3] N. H. Gartner, "Demand-responsive decentralized urban traffic control, Part I: Single intersection policies," U.S. Dept. Transp., Washington, DC, USA, Tech. Rep. DOT/RSPA/DPB-50/81/24, 1982.
[4] J. J. Henry, J. L. Farges, and J. Tuffal, "The PRODYN real time traffic algorithm," in *Proc. the 4th IFAC/IFIP/IFORS Conf.*, Jan. 1984, pp. 305–310.
[5] B. P. Gokulan and D. Srinivasan, "Distributed geometric fuzzy multiagent urban traffic signal control," *IEEE Trans. Intell. Transp. Syst.*, vol. 11, no. 3, pp. 714–727, Sep. 2010.
[6] H. Ceylan and M. G. Bell, "Traffic signal timing optimisation based on genetic algorithm approach, including drivers' routing," *Transp. Res. B, Methodol.*, vol. 38, no. 4, pp. 329–342, May 2004.
[7] S. Darmoul, S. Elkosantini, A. Louati, and L. B. Said, "Multi-agent immune networks to control interrupted flow at signalized intersections," *Transp. Res. C, Emerg. Technol.*, vol. 82, pp. 290–313, Sep. 2017.
[8] R. S. Sutton and A. G. Barto, "Reinforcement learning: An introduction," *IEEE Trans. Neural Netw.*, vol. 9, no. 5, p. 1054, Sep. 1998.
[9] C. Szepesvári, "Algorithms for reinforcement learning," *Synthesis lectures Artif. Intell. Mach. Learn.*, vol. 4, no. 1, pp. 1–98, Jul. 2010.
[10] V. Mnih *et al.*, "Human–level control through deep reinforcement learning," *Nature*, vol. 518, no. 7540, pp. 529–533, Feb. 2015.
[11] C. J. C. H. Watkins and P. Dayan, "Q-learning," *Mach. Learn.*, vol. 8, nos. 3–4, pp. 279–292, 1992.
[12] R. J. Williams, "Simple statistical gradient-following algorithms for connectionist reinforcement learning," *Reinforcement Learn.*, vol. 173, pp. 5–32, May 1992.
[13] V. R. Konda and J. N. Tsitsiklis, "Actor-critic algorithms," in *Proc. 12th Int. Conf. Neural Inf. Process. Syst.*, Nov. 1999, vol. 13, pp. 1008–1014.
[14] M. Aslani, M. S. Mesgari, and M. Wiering, "Adaptive traffic signal control with actor-critic methods in a real-world traffic network with different traffic disruption events," *Transp. Res. C, Emerg. Technol.*, vol. 85, pp. 732–752, Dec. 2017.
[15] V. Mnih *et al.*, "Asynchronous methods for deep reinforcement learning," in *Int. Conf. Mach. Learn.*, Jun. 2016, pp. 1928–1937.
[16] C. Guestrin, M. Lagoudakis, and R. Parr, "Coordinated reinforcement learning," in *Proc. ICML*, Jul. 2002, pp. 227–234.
[17] J. R. Kok and N. Vlassis, "Collaborative multiagent reinforcement learning by payoff propagation," *J. Mach. Learn. Res.*, vol. 7, pp. 1789–1828, Sep. 2006.
[18] M. Tan, "Multi-agent reinforcement learning: Independent vs. cooperative agents," in *Proc. 10th Int. Conf. Mach. Learn.*, Jun. 1993, pp. 330–337.
[19] J. Foerster *et al.* (Feb. 2017). "Stabilising experience replay for deep multi-agent reinforcement learning." [Online]. Available: https://arxiv.org/abs/1702.08887

[20] R. Bellman, "A Markovian decision process," *J. Math. Mech.*, vol. 6, no. 5, pp. 679–684, 1957.
[21] R. S. Sutton, D. McAllester, S. Singh, and Y. Mansour, "Policy gradient methods for reinforcement learning with function approximation," in *Proc. Adv. Neural Inf. Process. Syst. (NIPS)*, Dec. 1999, pp. 1057–1063.
[22] C. Guestrin, D. Koller, and R. Parr, "Multiagent planning with factored MDPs," in *Proc. 14th Int. Conf. Neural Inform. Process. Syst., Natural Synthetic*, vol. 1, Jan. 2001, pp. 1523–1530.
[23] G. Tesauro, "Extending Q-learning to general adaptive multi-agent systems," in *Proc. Adv. Neural Inf. Process. Syst.*, 2004, pp. 871–878.
[24] M. A. Wiering, J. Van Veenen, J. Vreeken, and A. Koopman, "Intelligent traffic light control," Ph.D. dissertation, Dept. Inst. Inf. Comput. Sci., Utrecht Univ., Utrecht, The Netherlands, 2004.
[25] C. Cai, C. K. Wong, and B. G. Heydecker, "Adaptive traffic signal control using approximate dynamic programming," *Transp. Res. C, Emerg. Technol.*, vol. 17, no. 5, pp. 456–474, Oct. 2009.
[26] P. La and S. Bhatnagar, "Reinforcement learning with function approximation for traffic signal control," *IEEE Trans. Intell. Transp. Syst.*, vol. 12, no. 2, pp. 412–421, Jun. 2011.
[27] T. Chu and J. Wang, "Traffic signal control with macroscopic fundamental diagrams," in *Proc. Amer. Control Conf. (ACC)*, Jul. 2015, pp. 4380–4385.
[28] T. Chu, J. Wang, and J. Cao, "Kernel-based reinforcement learning for traffic signal control with adaptive feature selection," in *Proc. IEEE Conf. Decision Control*, Dec. 2014, pp. 1277–1282.
[29] S. Richter, D. Aberdeen, and J. Yu, "Natural actor-critic for road traffic optimisation," in *Proc. Adv. Neural Inf. Process. Syst.*, 2007, pp. 1169–1176.
[30] T. Chu, S. Qu, and J. Wang, "Large-scale multi-agent reinforcement learning using image-based state representation," in *Proc. IEEE 55th Conf. Decision Control*, Dec. 2016, pp. 7592–7597.
[31] N. Casas. (Mar. 2017). "Deep deterministic policy gradient for urban traffic light control." [Online]. Available: https://arxiv.org/abs/1703.09035
[32] W. Genders and S. Razavi. (Nov. 2016). "Using a deep reinforcement learning agent for traffic signal control." [Online]. Available: https://arxiv.org/abs/1611.01142
[33] W. Genders and S. Razavia, "Evaluating reinforcement learning state representations for adaptive traffic signal control," *Procedia Comput. Sci.*, vol. 130, pp. 26–33, Dec. 2018.
[34] M. A. Wiering, "Multi-agent reinforcement learning for traffic light control," in *Proc. 17th Int. Conf. Mach. Learn. (ICML)*, Jun. 2000, pp. 1151–1158.
[35] T. Chu, S. Qu, and J. Wang, "Large-scale traffic grid signal control with regionalreinforcement learning," in *Proc. Amer. Control Conf.*, Jul. 2016, pp. 815–820.
[36] H. M. A. Aziz, F. Zhu, and S. V. Ukkusuri, "Learning-based traffic signal control algorithms with neighborhood information sharing: An application for sustainable mobility," *J. Intell. Transp. Syst.*, vol. 22, no. 1, pp. 40–52, 2018.
[37] E. E. Van der Pol and F. A. Oliehoek, "Coordinated deep reinforcement learners for traffic light control," in *Proc. NIPS*, Aug. 2016, vol. 16, pp. 1–9.
[38] S. El-Tantawy, B. Abdulhai, and H. Abdelgawad, "Multiagent reinforcement learning for integrated network of adaptive traffic signal controllers (MARLIN-ATSC): Methodology and large-scale application on downtown Toronto," *IEEE Trans. Intell. Transp. Syst.*, vol. 14, no. 3, pp. 1140–1150, Sep. 2013.
[39] F. Zhu, H. M. A. Aziz, X. Qian, and S. V. Ukkusuri, "A junction-tree based learning algorithm to optimize network wide traffic control: A coordinated multi-agent framework," *Transp. Res. C, Emerg. Technol.*, vol. 58, pp. 487–501, Sep. 2015.
[40] T. Chu and J. Wang, "Traffic signal control by distributed Reinforcement Learning with min-sum communication," in *Proc. Amer. Control Conf.*, May 2017, pp. 5095–5100.
[41] D. Krajzewicz, J. Erdmann, M. Behrisch, and L. Bieker, "Recent development and applications of SUMO-simulation of urban mobility," *Int. J. Advances Syst. Measurements*, vol. 5, nos. 3–4, pp. 128–138, Dec. 2012.
[42] S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural Computation*, vol. 9, no. 8, pp. 1735–1780, 1997.
[43] A. M. Saxe, J. L. McClelland, and S. Ganguli. (Dec. 2013). "Exact solutions to the nonlinear dynamics of learning in deep linear neural networks." [Online]. Available: https://arxiv.org/abs/1312.6120
[44] L. Codeca and J. Härri, "Monaco SUMO traffic (MoST) scenario: A 3D mobility scenario for cooperative ITS," in *Proc. SUMO User Conf., Simulating Auton. Intermodal Transp. Syst.*, Berlin, Germany, May 2018, pp. 14–16.
[45] Z. Wang *et al.* (2016). "Sample efficient actor-critic with experience replay." [Online]. Available: https://arxiv.org/abs/1611.01224

Tianshu Chu received the B.S. degree in physics from Weseda University, Tokyo, Japan, in 2010, and the M.S. and Ph.D. degrees from the Department of Civil and Environmental Engineering, Stanford University, in 2012 and 2016, respectively. He is currently a Data Scientist with Uhana Inc., and also an Adjunct Professor with the Stanford Center for Sustainable Development and Global Competitiveness. His research interests include reinforcement learning, deep learning, multi-agent learning, and their applications to traffic signal control, wireless network control, autonomous driving, and other engineering control systems.

Jie Wang received the B.S. degree from Shanghai Jiao Tong University, the M.S. degrees from Stanford University and the University of Miami, and the Ph.D. degree in civil and environmental engineering from Stanford University, in 2003, where he is currently an Adjunct Professor with the Department of Civil and Environmental Engineering, and also the Executive Director of the Stanford Center for Sustainable Development and Global Competitiveness. His research interests include information and knowledge management for sustainable development and innovation, enterprise IT infrastructure management, smart manufacturing, smart infrastructures and smart city, and environmental informatics.

Lara Codecà received the master's degree in computer sciences from the University of Bologna, Italy, in 2011, and the Ph.D. degree from the University of Luxembourg, in 2016. In 2011, she was a Visiting Fellow with Prof. Dr. Mario Gerla's Vehicular Lab, University of California at Los Angeles, Los Angeles, CA, USA. She is currently a Post-Doctoral Fellow at the CATS Group, EURECOM, France. Her research interests include (cooperative) intelligent transportation systems, vehicular traffic modelling, and big-data analysis. She is active in the SUMO community and collaborates with the SUMO developers at DLR (German Aerospace Centre).

Zhaojian Li received the bachelor's degree from the Department of Civil Aviation, Nanjing University of Aeronautics and Astronautics, China, and the M.S. and Ph.D. degrees in aerospace engineering (flight dynamics and control) from the University of Michigan, Ann Arbor, MI, USA, in 2013 and 2015, respectively. From 2016 to 2017, he has worked as an Algorithm Engineer at General Motors. He is currently an Assistant Professor with the Department of Mechanical Engineering, Michigan State University. His research interests include learning-based control, nonlinear and complex systems, and robotics and automated vehicles. He was a recipient of the National Scholarship from China.