Real-Time Large-Scale Traffic Signal Control: A Decentralized Multiagent Coordination Approach

Wanyuan Wang, Tianchi Qiao, Weiwei Wu, and Yichuan Jiang

Abstract—Traffic signal control (TSC) is one of the most beneficial ways for reducing urban road congestion. It has been a challenging problem because of 1) the complexity in traffic dynamics, 2) the essential of real time responsive policy and 3) the network-wide coordination. Reinforcement learning can find policies for mapping dynamic traffic states to signal actions, however, is inadequate for abnormal traffic. Once observing the real-time traffic, online planning recomputes the signal policy in a best-response manner. Due to the computation complexity and the lack of network-wide coordination, existing online planning methods have limited scalability and efficiency. Against this background, we propose an explicit multiagent coordination (MAC) method to satisfy adaptive, real-time and network-wide TSC. In view of MAC, we model the TSC as a distributed constraint optimization problem (DCOP). Each intersection is modeled as an individual agent, and the coordination is modeled by a utility function between direct neighbor intersections. By monitoring real traffic information, each agent exchanges utility messages with his neighbors. The network-wide coordination can be achieved by iterative message-passing. Finally, we test our MAC method in different traffic flow and network structure. Experimental results are encouraging: compared to state-of-the-art RL and online planning methods, our Ex-MAC method performs reasonably well in terms of adapting to real-time traffic dynamics and minimizing vehicle traveling time.

Index Terms—Multiagent Coordination, Traffic Light Control, Distributed Constraint Optimization Problem

1 Introduction

Nowadays, traffic signal control (TSC) is still dominated by the use of fixed timing plans such as SCATS [1] and SCOOT [2], which are carefully tuned by domain experts according to regular traffic conditions. However, with the ever-increasing number of vehicles traveling on saturated urban road networks, these pre-calculated TSC rules are quickly become outdated and traffic congestion has become the key inconvenience to drivers. According to the INRIX Global Traffic Scorecard, in 2019, an average driver in the U.S. lost 99 hours a year due to congestion, costing them nearly $88 billion[1]. It is commonly recognized that modern intelligent TSC should be real-time responsive to traffic dynamics and scalable to network-wide optimization [3], [4].

To dynamically adjust TSC policies according to real-time traffic, reinforcement learning (RL) has attracted lots of attention [5]. By interacting with traffic environment, RL can learn to map the real-time observations to policies [6], [7], [8]. The independent RL, however, ignores the temporal and spatial influences among intersections, where local connected intersection policies might have cross-purposes [9]. Multiagent RL (MARL) attempts intersection coordination, where the local policy is regulated by the global joint state-action credits [10], [11], [12], [13], [14]. To achieve scalable network-wide optimization, hierarchical RL can be applied by decomposing the network into sub-networks, each sub-network controls its signals in the low-level, and the centralized upper-level constrains and evaluates the sub-network policy [15], [16], [17]. These RL approaches solve the problem of real-time in principle, but are trained offline for average traffic flows, have difficulty applying in an online manner if traffic flows change irregularly with emergency events and disruptions.

In control community, model-predictive control has focused on online planning, which proceeds according to a traffic prediction model and attempts to find a good signal sequence in a prediction horizon for the current observation [18]. Unfortunately, MPC needs a complex mathematical program to optimize the multiple intersections' signal sequences, which is inherently susceptible to scalability issues. A recent development in online planning that overcomes this network-wide issue is schedule-driven coordination approach [19], [3]. The key idea behind this approach is to formulate the intersection TSC problem as a single machine scheduling problems, where jobs are represented by a sequence of clusters consisting of spatially adjacent vehicles. Scalability is ensured by the fact that intersections only search for their own schedule space. Coordination is achieved by exchanging the traffic flow information with their local direct neighbors [4]. This non-local coordination of a projection of traffic flows from direct neighbors, however, is implicit, do not coordinate the network-wide intersections in a synchronous manner.

To satisfy adaptive, real-time and network-wide coordination, in this paper, we propose an explicit multiagent coordination (Ex-MAC) approach [20]. Each intersection is modeled as an autonomous and cooperative agent and the coordination among agents to regulate traffic is represented as a constraint function. To well characterize the traffic state, we propose load balance to model the coordination function, which is claimed as a main contribution of this paper. In order to enforce a truly decentralized solution, each intersection has only knowledge of and can directly communicate with direct neighbors, on whose state the coordination function depends. The network-wide coordination function is optimized by message-passing, which is widely used in distributed constraint optimization problems (DCOP) [21], [22].

W. Wang, W. Wu and Y. Jiang are with the School of Computer Science and Engineering, Southeast University, Nanjing, China (e-mail: wywang@seu.edu.cn; weiweiwu@seu.edu.cn, yjiang@seu.edu.cn).
Z. Li is with the School of Transportation, Southeast University, Nanjing, China (e-mail:boan@ntu.edu.sg).
Manuscript received Oct. 20, 2021.

1. https://inrix.com/press-releases/2019-traffic-scorecard-us/

The main technique contribution is to propose a efficient edeg-to sink message-passing order, which can speed the convergence. We theoretically show that the stable property can be guaranteed by our Dec-MAC approach. Finally, we conduct comprehensive experiments using both synthetic data and real data. We demonstrate the power of Dec-MAC method over traditional Max-pressure method as Dec-MAC optimizes the coordination of intersections. Our method also consistently outperforms state-of-the art RL methods, which shows that explicit coordination is necessary.

The remainder of this paper is organized as follows. In Section II, we provide a brief review of related researches on peer grading systems. In Section III, we model the OptSC problem. We analyse the problem in Section IV and propose an efficient approximation algorithm in Section V. In Section VI, we consider the uncertain OptSC variant where peers have uncertain reliability and cost. In Section VII, we conduct a set of experiments to evaluate our proposed algorithm's performance on evaluation accuracy. Finally, we conclude our paper and discuss future work in Section VIII.

2 Related Work

Network-level TSC should take the uncertain traffic flow, real-time signal control, and adjacent intersections coordination into consideration. In the following, we review and discuss how related researches concern on these three aspects. Owing to the practical benefits and the challenging nature of the underlying control problem, there have been multiple threads of research dedicated to online traffic signal control. TSC has been addressed by researchers in many different fields, including AI decision making, control theory and transportation research.

2.1 Multiagent Reinforcement Learning

Because of the highly dynamic traffic flows, reinforcement learning (RL), which can directly learn how to take actions in complex and uncertain environment, has recently been investigated for TSC [5], [36]. In RL, each intersection can be modeled by an autonomous agent that interacts with the traffic environment and learns from rewards to achieve optimal mapping between the traffic state and the corresponding signal control action. Thus, careful reward design is essential to minimize network travel time [37]. Wei et al. [23] and Pol and Oliehoek [11] combine multiple factors such as queue length, delay, waiting time, traffic flow and vehicle speed as the immediate reward. Inspired by the max-pressure theory proposed in transportation literature [38], Wei et al. [24] further propose a "pressure" reward that has the network flow stability guarantee. For the roundabout intersections with heavy traffic volumes, Rizzo et al.[39] shape the reward by the lost green time due to traffic congestion.

At the network level, independent RL methods can directly transfer the single agent's plan to other agents [40], [8]. However, some intersections are tightly coupled, without coordination, congestion starts from the upstream intersections will spread to downstream intersections [41]. Multiagent RL, by modeling the agent that can communicate and coordinate with neighbor agents, has proposed to optimize network-wide performance. For example, by extending the observable surroundings of agents with neighbors' state information, Liu et al. [25] propose a distributed RL method. The temporal and spatial influences of neighboring agents to the target agent can be learned by the graph attention networks [13]. Yu et al. [14] further propose an active communication mechanism where the historical action and state information is actively communicating to neighbors. In terms of policy coordination, a modular Q-learning framework is proposed to partition the state space to partial state spaces that only consist of two neighboring agents and each agent selects the action optimizing the Q function of each pair of agents [9]. Kuyer et al. [10] propose a factored Q-function to learn the coordination policy between a pair of connected agents, and coordinated plan can be transferred to other pairs of agents [11]. Using the centralized training and decentralized execution framework, Chu et al. [12] propose an advantage actor critic (A2C) mechanism to improve the coordination. Exploiting the advantage of RL on local neighboring intersections, hierarchical RL can be applied by the decomposing the network into sub-networks, where each sub-network controls its signals in a decentralized manner, and the global centralized Q function is used to evaluate and regulate the sub-network policy [15], [16]. The up-to-bottom feudal RL is to produce sub-goals, for which each sub-network learns to reach [26].

Unfortunately, RL based approaches have multiple issues that have prevented deployment on real-world TSC: (1) elaborate simulators and millions of simulations of traffic flows are required to learn good policies for a single intersection, and the number of simulations required grows exponentially in number of agents; (2) since the traffic environment will be nonstationary with multiple agents learning concurrently, deep MARL approaches for TSC are not stable and may trap into sub-optimal solution; (3) learning-based TSC methods can be trained efficiently for regular traffic flows, but difficult to apply in an online manner if traffic flows are changing irregularly (e.g.,unexpected incidents such as accidents and lane closure happen).

2.2 Model-Predictive Control

The framework of model-predictive control (MPC) approach contains 1) a prediction model describing the traffic dynamics in a finite-time horizon, and 2) an online long-term traffic optimization [18]. For example, one or multiple intersections traffic arrivals can be estimated [42], which can be used to build a linear program (LP) [28] or Markov decision process (MDP) model [27]. Efficient dynamic programming is adopted to generate the real-time strategies. Building the traffic flow update model, the quadratic-program [29] and the mixed-integer linear program (MILP) [30] can be used to optimize signal control. To alleviate the computation problem, Zhou et al. [32] propose a two-level hierarchical framework, where the upper level coordinates the traffic flow among subnetworks, and the lower-level coordinates the signal control strategy with the traffic flow constraint. For the arterial road with several intersections, bandwidth criterion can be achieved by a centralized MPC controller for signal strategies coordination [43]. However, these centralized MPC solutions to traffic regulation result in high computational requirements, and create a single point of failure [44]. For example, Heung et al. [45] propose a centralized coordination mechanisms for dynamically adjusting offsets, which are inherently susceptible to scalability issues with tens of intersections. For network-level scenarios, a distributed MPC approach for online signal control is necessary. For example, Oliveira et al. [31] decompose the centralized MILP formulations into separable optimizations, each can be computed by an independent agent in parallel [32]. For more details and background on reinforcement learning see

MPC is a real-time and traffic-responsive control approach, but needs to decomposes the network-level intersections into

Approaches		Response Time (seconds)	Coordination		Scalability # of Intersections
			Local	Global	
Reinforcement Learning	Independent RL [23], [24], [8]	$\leq$3	✖	✖	$\sim$ 2500
	Coordinated RL [10], [9], [11], [25], [13], [14]	$\leq$3	✔	✖	$\sim$ 200
	Hierarchical RL [15], [16], [26]	$\leq$3	✔	✔	$\leq$36
Model Predictive Control (MPC)	Centralized MPC [27], [28], [29], [30]	$\geq$3	✔	✖	$\geq$15
	Decentralized MPC [31], [32]	$\geq$80	✔	✔	$\leq$55
Online Planning [19], [3], [33], [34], [4], [35]		$\leq$3	✔	✖	$\leq$25
Our MAC Approach		$\leq$3	✔	✔	$\sim$ 400

TABLE 1: Summary of the related literature.

structured sub-networks. Since the hierarchical structure is serial and the decomposition is approximated, MPC is limited in the scalability to network-level applications. A detailed survey of existing formal models, complexity results and planning algorithms is available in (ZJU). In contrast to MPC, this paper directly models network-wide TSC as a distributed problem, and propose a scalable network-wide approaches.

2.3 Decentralized Local Coordination Control

At the network level view, SURTRAC (Scalable Urban Traffic Control) is recently proposed as a leading approach for online traffic signal control, which is a real-time, distributed, schedule-driven approach [3]. By modeling a cluster of vehicles as indivisible jobs and intersections as machines, a schedule-driven intersection control framework is proposed where jobs are served within the minimum delay [19], [34]. The coordination is achieved by each intersection that receives and responds outflows from its upstream neighbors. Considering the impact of the vehicle queue length on traveling delay, a weight is computed and assigned to each road. For example, Hu and Smith [33] use the link softpressure as the weight and Wu et al. [35] use the occupancy ratio as the weight. Given the weight of each road, the signal timing plan can be optimized to optimize the weighted traffic efficiency. Hu and Smith [4] further extend the basic single direction coordination algorithm to a bi-directional coordination algorithm where the downstream estimated congestion information is also communicated to upstream intersections. To tackle with the uncertainty associated with the vehicle turn movements at intersections,a sample-based constrained optimization is proposed to minimize expected delay over all vehicles [46].

Despite its scalability and effectiveness, key to SURTRAC approaches is an ability to accurately predict outgoing traffic. While these approximations and asynchronise ensure real-time and coordination tractability, control quality with respect to estimated traffic degrades as the phase duration increases. By contrast, we relax the asynchroy signal control assumption and model each intersection governs the meta signal control stategy and coordinates the signal control in a synchronous manner, which reduces the negative effect of traffic uncertainty.

3 Problem Description

This section presents the TSC problem including the concepts of traffic network, movement phases, network state update, and the object of TSC.

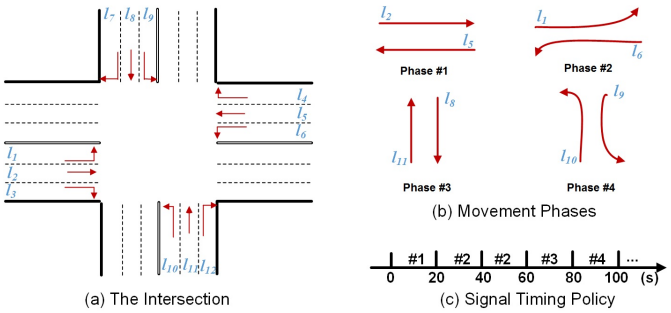
Traffic network and intersections. Let $G = \langle \mathcal{N}, \mathcal{L}_{all} \rangle$ denote the traffic network, where $\mathcal{N} = \{1, 2, \cdots, n\}$ indicates a set of n signalized intersections, and $\mathcal{L}_{all} = \mathcal{L} \cup \mathcal{L}_{entry} \cup \mathcal{L}_{exist}$ indicates a set of directed links (i.e., road) in the network. There are three types of links: an internal link $l \in \mathcal{L}$ goes from its start intersection $i \in \mathcal{N}$ to its end intersection $j \in \mathcal{N}$; an entry link $l \in \mathcal{L}_{entry}$ has no start intersection; and an exit link $l \in \mathcal{L}_{exist}$ has no end intersection. Link l is input to link h if l has an end intersection i and h has a start intersection i. An intersection j is a neighbor of the intersection i, if there exists a link l whose start (end) and end (start) intersections are i and j respectively. Let $Neg(i)$ denote the neighbors of the intersection i. Let $I(i)$ and $O(i)$ denote the set of links entering and leaving the intersection i, respectively. The $l_{ij} = I(j) \cup O(i)$ indicates the link leaving j and entering i. Let In_l and Out_l denote the the set of neighbor links which the link l enters to and leaves from, respectively.

Notation	Description
$\mathcal{N} = \{1, \ldots, i, \ldots, n\}$	the set of intersections
$\mathcal{L}_{all} = \mathcal{L} \cup \mathcal{L}_{entry} \cup \mathcal{L}_{exist}$	the set of links/roads
$I(j)$	the peers who grade assignment j
$J(i)$	the assignments graded by peer i
$l = \lvert I(j) \rvert = \lvert J(i) \rvert$	the load (number of assignments) of peers
$e_{ij} \in \{0, 1\}$	peer i's effort level on assignment j
$c_{ij} \in [0, 1]$	peer i's cost of putting full effort on j
$p_i^0 = 0.5$	the reliability of peer i with zero effort
p_i^1	the reliability of peer i with full effort
$r_{ij} \in [0, 1]$	the reward paid to i for diligent grading j
$x_j \in [0, 1]$	the probability of j to be checked
$q_j \in \{-1, 1\}$	the true quality of j
$\tilde{q}_j \in \{-1, 1\}$	the estimated quality of j
$p_{ij} \in \{p_i^0, p_i^1\}$	the reliability of i on grading j
$w_{ij} \in [0, 1]$	the weight of i's grade on j
$\mathbf{p}_j = (p_{ij})_{i \in I(j)}$	peers reliability profile on j
$\mathbb{P}_e(j, \mathbf{p}_j)$	j's error rate with $\mathbf{p}_j$
$\mathbb{P}_e^u(j, \mathbf{p}_j)$	peers upper bound error rate with $\mathbf{p}_j$
θ_{ij}	the critical checking probability of j on i
η_j	the critical checking probability of j on $I(j)$

TABLE 2: Notation Overview

Movement phase. A *traffic movement* (l, h) is defined as the traffic traveling across an intersection i from entering link $l \in I(i)$ to leaving link $h \in O(i)$. The set of movement phases at an intersection i is $\{(l, h) \in I(i) \times O(i)\}$. Conflict-free phases at intersection i can be simultaneously activated. The signal control at the intersection i can be represented by $\mathcal{X}_i = \{x_i(l, h) \in \{0, 1\}, l \in I(i), h \in O(i)\}$, where $x_i(l, h) = 1$ indicates the green light is activated and the movement is allowed, and $x_i(l, h) = 0$ indicates the red light is activated and the movement prohibited. In a typical intersection as shown in Fig. 1, there are twelve traffic movements, which can be controlled by four *movement phases*: *WE-Straight* (going straight from West and East), *SN-Straight* (going straight from South and North), *WE-Left* (turning left from West and East), *SN-Left* (turning left from South and North)[2].

2. It is worth noting that 1) the "turn right" traffic movement signal can be always activated and 2) there might be different number of phases in the real world intersections and four phases model is not a must



(a) The Intersection (b) Movement Phases (c) Signal Timing Policy

Fig. 1: Intersection, movement phases and signal control policy.

Network State Update. Time is discretized into periods, $t = 1, 2, \cdots, T$, each includes a fixed duration of τ seconds (e.g., 20 seconds). Let $q(l,h)$ denote queue length of movement (l,h), i.e., the number of vehicles leaving the link l and entering the link h. Let $f(l,h)$ denote the saturation flow of traffic movement (l,h), i.e., if movement (l,h) is activated at the beginning of the period t, there will be at most $f(l,h)$ vehicles traveling from link l to link h at the end of t. An inactivated phase serves zero flow of vehicles. At the beginning of the period t, let $Q(t) = \{q(l,h)(t)\}$ denote the queue state of the traffic network, and $x(l,h)(t)$ denote whether the phase (l,h) is active (i.e., $x(l,h)(t) = 1$) or not (i.e., $x(l,h)(t) = 0$). The queue state for internal link $l \in \mathcal{L}$ updates according to [38]:

$$q(l,h)(t+1) = q(l,h)(t) - \overbrace{f(l,h)(t)x(l,h)(t) \wedge q(l,h)(t)}^{outgoing\ \ vehicles} + \overbrace{\sum_{e \in In_l} [f(e,l)(t)x(e,l)(t) \wedge q(e,l)(t)] r(l,h)(t+1)}^{incoming\ \ vehicles}. \quad (1)$$

where the function $x \wedge y = \min\{x, y\}$. The second term on the right in Eq.(1) indicates that the queue length $x(l,h)(t)$ decreases by up to $f(l,h)(t)$ vehicles during t if $x(l,h)(t) = 1$; the third term indicates that up to $f(e,l)(t)$ vehicles will move from queue (e,l) during t if $x(e,l)(t) = 1$ and they will join queue (l,h) with a probability of $r(l,h)(t+1)$ at link next period $t+1$. This turning proportion parameter $r(l,h)(t+1)$ indicates the proportion of vehicles leaving l and entering h, which are assumed available and can be estimated from the route navigation systems [47].

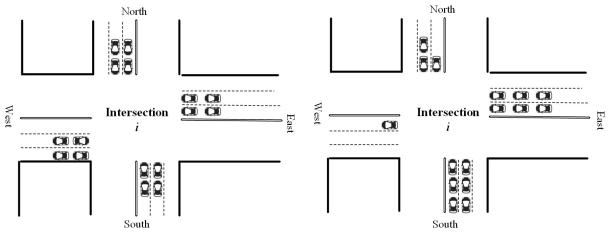
The queue update equation for an entry link $l \in \mathcal{L}_{entry}$ is a bit different since it has no input links but exogenous arrivals:

$$q(l,h)(t+1) = q(l,h)(t) - \overbrace{f(l,h)(t)x(l,h)(t) \wedge q(l,h)(t)}^{outgoing\ \ vehicles} + \overbrace{d(l)r(l,h)(t+1)}^{incoming\ \ vehicles}. \quad (2)$$

where $d(l)$ denote the exogenous flow of vehicles in entry link $l \in \mathcal{L}_{entry}$, which is also assumed available by estimation, and $r(l,h)(t+1)$ is the turning proposition.

The Objective. At period t, given the queue state, let $Q_i(t) = \{q(l,h) | l \in I(i), h \in O(i)\}$ denote the queue state at the intersection i. We utilize the *balance* concept $B_i(t)$ to characterize the traffic congestion, which is defined as the sum of squares of all movements' queue length at intersection i, i.e.,

$$B_i(t) = \sum_{(l,h): l \in I(i), h \in O(i)} [q(l,h)]^2. \quad (3)$$



(a) The Balance Scenario. (b) The Imbalance Scenario

Fig. 2: The advantage of balance function.

the lower value $B_i(t)$ indicates that the intersection i is more balance, and there will be the less average waiting time for vehicles.

Example 1. *In Fig.2, at the intersection i, assume that the saturation flow of each traffic movement is 2, i.e., $f(l,h) = 2, \forall l \in I(i), h \in O(i)$. In the balance scenario (i.e., Fig.2(a)), the optimal control policy is to active the four phases in turn (i.e., the rotation policy), by which the average waiting time of all vehicles is $\frac{4\times(1+2+3)}{16} = 1.5$. However, given the same number of vehicles of queuing at intersection i, in the imbalance scenario (i.e., Fig.2(a)), after one round rotation there are still vehicles waiting at the movements of* WE-Straight *and* WE-Left*, which will incur a larger average waiting time.*

Finally, at period t, given the network state $Q(t) = \{Q_i(t)\}_{i \in \mathcal{N}}$, exogenous demands $d_l(t)$, saturation flow $f(l,h)(t)$ and turning proportion $r(l,h)$, the optimal TSC policy $\mathcal{X}^*(t) = \{x(l,h)^*(t)\}$ should be selected with the objective of minimizing the network-level balance, i.e.,

$$\mathcal{X}^*(t) = \arg\min_{\mathcal{X}(t)} \sum_{i \in \mathcal{N}} B_i(t+1). \quad (4)$$

where the queue state $Q(t+1)$ is updated by the policy $\mathcal{X}(t)$ on current network state $\{Q(t), f(l,h)(t), d(l)(t), r(l,h)\}$.

4 Multiagent Coordination Model for TSC

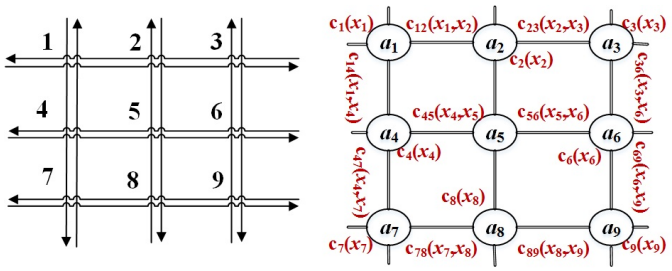
This section models the TSC problem using multiagent coordination (MAC), and real-time large-scale algorithm will build on top of this particular framework.

Coordination Graph (CG). Given the real TSC problem, a CG can provide an useful and appropriate MAC model [48]. A CG is a tuple $(\mathcal{A}, \mathcal{X}, \mathcal{D}, \mathcal{E}, \mathcal{C})$ such that:

- $\mathcal{A} = \{a_1, a_2, \cdots, a_n\}$ is the set of agents, each agent is equipped with computation capacity and able to compute and send messages.
- $\mathcal{X} = \{x_1, x_2, \cdots, x_n\}$ is the set of variables, each takes the value from the finite discrete domain $D_i \in \mathcal{D}$. Each variable x_i is controlled by an agent a_i, and each agent has an individual cost function $c_i(x_i)$.
- $e_{ij} \in E$ is the edge connects agents a_i and a_j if their cost depends on each other. The interaction between connected agents a_i and a_j is constrained by the local cost function $c_{ij}(x_i, x_j)$. Here, the cost function $c_{ij} \in C$ is defined as a mapping from the assignments of the involved binary variables (x_i, x_j) to a positive real, $c_{ij} : D_i \times D_j \to \mathbb{R}_{\geq 0}$.

For CGs in the multiagent settings, the global cost for a joint action $\bar{x}$ is factored as the sum of individual costs and local component costs, i.e.,

$$C(\bar{x}) = \sum_{a_i \in \mathcal{A}} c_i(x_i) + \sum_{e_{ij} \in \mathcal{E}} c_{ij}(x_i, x_j). \quad (5)$$



(a) Traffic Network (b) Coordination Graph

Fig. 3: An example of modeling TSC by CG: each intersection i is modeled as an autonomous agent a_i, the internal link congestion between intersections i and j is modeled by the constraint cost c_{ij}, and the entry link congestion is modeled by the individual agent cost c_i.

The objective of a CG is to find the best joint action, $\arg\min_{\bar{x}} C(\bar{x})$, that minimizes the global cost.

Modeling TSC by CG. The key step toward modeling the TSC problem as a CG is mapping intersection, movement phase, and objective to the three tuples in the CG model. By capturing the structure of TSC, a straightforward CG is modeling each agent a_i in CG an intersection i in TSC[3]. Throughout this paper, we use agent and intersection interchangeable. Each agent a_i holds a decision variable x_i determining the movement phase. The domain of the variable x_i can be denoted by $\mathcal{D}_i = \{1, 2, 3, 4\}$, which corresponds to four movement phases, i.e., 1=*WE-Straight*, 2=*WE-Left*, 3=*SN-Straight* and 4=*SN-Left* is activated.

Mapping Link Congestion to the Constraint Cost. Given a particular variable assignment, one challenge is to construct constraint cost such that when the resulting CG is solved, we obtain a solution that is congruent to the original TSC problem. Considering different kinds of links (e.g., entry link and internal link[4]), we model two kinds of constraint cost,

- Internal link congestion model: given two neighbor agents a_i and a_j, we define their constraint cost c_{ij} as the congestion of links (i.e., l_{ij} and l_{ji}) between them, i.e.,

$$c_{ij}(x_i, x_j) = \sum_{h \in O(i)} [q(l_{ij}, h)]^2 + \sum_{h \in O(j)} [q(l_{ji}, h)]^2. \quad (6)$$

- Entry link congestion model: For the entry link $l \in \mathcal{L}_{entry}$, we map the link congestion to the entering intersection, i.e., for each boundary agent $a_i \in \mathcal{N}_B$ that connects entry links, the individual agent cost c_i can be defined as:

$$c_i(x_i) = \sum_{l \in I(i), h \in O(i)} [q(l, h)]^2. \quad (7)$$

For any other agent $a_i \notin \mathcal{N}_B$, the individual cost c_i is set to be zero, i.e., $c_i(\cdot) = 0, \forall a_i \notin \mathcal{N}_B, x_i \in D_i$.

Lemma 1. *The object function of the CG is in consistent with the object function of the TSC.*

3. In practice, the intersection signal controller is equipped with computation unit that can control the traffic light signal.

4. In this paper, we assume that vehicles arrive their destination at the exit link, thus the congestion on the exit link can be omitted

Proof. Given any variable assignment $X = (x_1, x_2, \cdots, x_n)$, the object function of the CG is

$$\begin{aligned}
&\sum_{a_i \in \mathcal{B}} c_i(x_i) + \sum_{e_{ij} \in E} c_{ij}(x_i, x_j) \\
&= \sum_{l \in \mathcal{L}_{entry}, h \in Out(l)} [q(l,h)]^2 + \sum_{l \in \mathcal{L}, h \in Out(l)} [q(l,h)]^2 \\
&= \sum_{l \in \mathcal{L}_{all}, h \in Out_l} [q(l,h)]^2 \\
&= \sum_{i \in \mathcal{N}} \sum_{(l,h): l \in I(i), h \in O(i)} [q(l,h)]^2 = \sum_{i \in \mathcal{N}} B_i.
\end{aligned}$$

□

5 The Real-Time and Decentralized Algorithm

In this section, on top of the MAC model, we propose a message-based decentralized algorithm to control the signalized light. The proposed algorithm mainly consists of two phases: 1) global joint action coordination (i.e., Section 5.1) and individual action improvement 5.2.

5.1 Global Joint Action Coordination

In this section, we propose a message passing algorithm to coordinate the global solution. In order to avoid the message passing pathology caused by cycles in a CG, we first select an order on all agents, i.e., transforming a cyclic CG to a directed acyclic graph (DAG) for message passing.

5.1.1 Transforming the CG to the DAG

Existing work [21] has constructed the DAG by ordering agents according to the indices of agents, that is an agent a_i is ordered before agent a_j if $i < j$. However, this kind of DAG has an arbitrary diameter, in which the message passing complexity can be further reduced. In this section, we transform a given CG to a DAG with the minimum diameter.

The main idea of DAG constructing is that we first determine a sink agent, and then determine the message passing order of agents according to their distance to the sink agent. A detailed DAG construction is shown in Algorithm 1, which consists of two phases.

- Determine the sink agent a_s. From steps 2-5, we compute the diameter of the DAG if the agent a_i is determined as the sink agent, $dia(a_i) = \max_{a_j \in A} d(a_i, a_j)$, where $d(a_i, a_j)$ denote the shortest distance between agents a_i and a_j. The agent a_s that has the shortest diameter $d(a_s)$ is denoted as the sink agent.
- Determine the message passing order. From steps 6-10, once the sink agent a_s is determined, the message passing order can be selected from "external" agents (i.e., these agents that are far away from the sink agent a_s) to internal agents (i.e., these agents that are close by the sink agent a_s) (Steps 7-8). For these agents a_i and a_j that have the same distance to the sink agent a_s, the message passing order between a_i and a_j is arbitrarily selected.

It should be noted that Algorithm 1 generates minimum diameter DAG with a single sink agent.

Algorithm 1: Directed Acyclic Graph (**DAG**(o))

Input : The coordination G.
Output: The message passing order o.
Initialize $dia = +\infty$, $o = \emptyset$;
for $a_i \in \mathcal{A}$ **do**
  $dia(a_i) = \max_{a_j \in A} d(a_i, a_j)$;
  if $dia(a_i) < dia$ **then**
    $dia = dia(a_i)$, $a_s = a_i$; // sink agent determination
for $a_i, a_j \in \mathcal{A} : e_{ij} \in E$ **do**
  if $d(a_s, a_i) \leq d(a_s, a_j)$ **then**
    $o_{ij} = a_j \rightarrow a_i$;
  else
    $o_{ij} = a_i \rightarrow a_j$;

Algorithm 2: Global Joint Action Coordination (**Glo-Coor**($o, dia(o)$))

Input : The message passing order o.
Output: The joint action $\mathcal{X} = \{x_1, x_2, \cdots, x_n\}$.
while *time budget is remained* **do**
  for $a_i \in \mathcal{A}$ **do**
    perform MESSAGE-PASSING(a_i,$\langle o, dia(o)\rangle$);
  $o \leftarrow$ reverse(o);
  for $a_i \in \mathcal{A}$ **do**
    perform MESSAGE-PASSING(a_i,$\langle o, dia(o)\rangle$);
  Copmpute coordianted action x_i by Eq.(9);
Function MESSAGE-PASSING(a_i,$\langle o, dia(o)\rangle$)**:**
  $Neg_{foll}(i) \leftarrow \{a_j \in Neg(i) : a_j$ is before a_i in order $o\}$;
  $Neg_{prev}(i) \leftarrow Neg(i) \setminus Neg_{foll}(i)$;
  for $dia(o)$ iterations **do**
    Collect messages from $Neg_{prev}(i)$;
    for $a_j \in Neg_{foll}(i)$ **do**
      send message Q_{ij} by Eq.(8) to a_j;

5.1.2 Message Passing-based Coordination

To coordinate joint actions of agents, we would like to employ the Max-sum_ADVP (Max-Sum through value propagation on an alternating DAG) algorithm [49], [21], [22], an algorithm based on the exchange of messages among agents. Despite the fact that our formulation is compatible with any complete or incomplete message passing algorithm, we chose to use Max-Sum_ADVP as it is one of the fastest and most efficient algorithms in many multiagent domains. It should be noted that the TSC is actually a minimization problem, however, we will continue to refer to it as Max-sum since this name is widely accepted.

Given a DAG, let $Neg_{foll}(i) \subseteq Neg(i)$ denote the subset of a_i's neighbor agents that are ordered after a_i, and $Neg_{prev}(i) \subseteq Neg(i)$ denote the subset of a_i's neighbor agents that are ordered before a_i. During the message passing process, each agent a_i iteratively collect messages from $Neg_{prev}(i)$ and sends messages to $Neg_{foll}(i)$ in the DAG. At each iteration, a message Q_{ij} sent from agent a_i to the follow agent $a_j \in Neg_{foll}(i)$ is a scalar-valued function of the action space of receiving agent a_j,

$$Q_{ij}(x_j) = \min_{x_i}\{c_i(x_i) + c_{ij}(x_i, x_j) + \sum_{a_k \in Neg_{prev}(i)} Q_{ki}(x_i)\}. \tag{8}$$

This message for each possible value $x_j \in D_j$ can be explained as follows. When agent a_j selects the action x_j, agent a_i attempts to coordinate with a_j by selecting his action x_i to minimize the sum of costs incurred by its own c_i, the constraint c_{ij} and it received from all previous agents $Neg_{prev}(i)$.

Agents exchange messages until convergence such that the received messages do not update. Finally, each agent decides its optimal action. When an agent makes decisions, it accumulates all costs it receives, and selects an action to minimize the sum costs. The action selection procedure for each agent a_i can be formalized by

$$x_i^* = \arg\max_{x_i}\{c_i(x_i) + \sum_{j \in Neg_{prev}(i)} Q_{j \rightarrow i}(x_i)\}. \tag{9}$$

In Algorithm 2, the function MESSAGE-PASSING($a_i, \langle o, dia(o)\rangle$) presents a sketch of the message passing mechanism. At each iteration, each agent a_i first aggregates the message received from "previous" agents $Neg_{prev}(i)$ and sent messages to "follow" agents $Neg_{foll}(i)$ according to Eq.(8). In step 11, the message passing process terminates after $dia(o)$ iterations, where $dia(o)$ denote the diameter of the message passing order o.

Lemma 2. *For the DAG o, the message passing can converge within $dia(o)$ iterations, after $dia(o)$ iterations of message passing, the content of the messages each agent a_i receives does not change as long as messages are sent according to o, i.e., the message passing process converges after $dia(o)$ iterations.*

Since the content of messages produced by agents is MESSAGE-PASSING dependent only on the content of messages they received last iteration from previous agents. Starting from the 1st iteration, the boundary agents (that have no previous agents) will send the identical agents. Starting from the 2nd iteration, the agents that are followed by the boundary agents will send the identical agents. By induction, after the $dia(o)$ iterations, the sink agent will receive the identical messages. The detail proof can refer to [21].

Lemma 3. *The message passing procedure converges the fastest on the DAG generated by Algorithm 1.*

Derived from the Lemma 2 that MESSAGE-PASSING algorithm converges within $dia(o)$ iterations, we can conclude that the DAG generated by Algorithm 2 will converge with the minimum communication complexity.

Finally, we formulate the global joint action coordination algorithm in Algorithm 2. In steps 2-3, each agent a_i performs MESSAGE-PASSING process on the forward message passing order o. After $dia(o)$ iterations in current order o, the order is reversed (step 4) and messages are sent for the next $dia(o)$ iterations only in the opposite direction (steps 5-6). After message passing converge in both directions, value propagation procedure starts, where the action x_i^* of each agent a_i is selected by 9.

5.1.3 Stability Analysis

Stability is an important metric to evaluate the TSC policy in transportation domain, which is defined as follows.

Definition 1. *(Network stability). The network state process $Q(t) = \{q(l,m)(t)\}$ is stable in the mean (and $\mathcal{X}$ is a stabilizing TSC policy) if for some $K < \infty$*

$$\frac{1}{T}\sum_{t=1}^{T}\sum_{l,h}\mathbb{E}[q(l,h)(t)] < K, \forall T. \tag{10}$$

where $\mathbb{E}$ denotes the expectation.

Varaiya [38] has proposed the *max pressure*(MP) signal control policy to guarantee the network stability without the knowledge of the average demand $\{d_l\}$.

Definition 2. *MP signal control policy [38]. Given the network state $Q(t)$ at current period t, each agent/intersection selects the action with the maximum pressure: $\vec{X}^*(t) = \arg\max_{\vec{X}(t)\in \mathrm{X}(t)} \sum_{(l,m):x(l,m)=1} f(l,m)[q(l,m) - \sum_{p\in Out_m} r(m,p)q(m,p)]$, where $q(l,m)$ is the upstream queue length $\sum_{p\in Out_m} r(m,p)q(m,p)$ is the (average) downstream queue length, the pressure $q(l,m) - \sum_{p\in Out_m} r(m,p)q(m,p)$ of a movement phase is simply the difference between upstream and downstream queue lengths.*

MP signal control policy [38] has proven to be stable, and the main idea for MP policy stability is to guarantee

$$\mathbb{E}|Q(t+1)|^2 - \mathbb{E}|Q(t)|^2 \le K - \epsilon E|Q(t)|. \tag{11}$$

where $|Q(t)| = \sum_{(l,m)} |q(l.m)|$, $|Q(t)|^2 = \sum_{(l,m)} |q(l,m)|^2$, and $K < \infty$.

Theorem 1. *The signal control policy $\mathcal{X}^{MP}$ selected by MP and the policy $\mathcal{X}^{Glo-Coor}$ selected by our global joint action coordination algorithm (i.e., Algorithm 2) are both stablizing the network.*

Proof. Given the network state $Q(t)$ at the current period t, let $\mathcal{X}^{Opt}$ denote the optimal action that minimizes the sum of squares of all the queue lengths at the next period, i.e., $\mathcal{X}^{Opt} = \arg\min_{\mathcal{X}} \mathbb{E}|Q(t+1)|^2$. Let $\mathbb{E}|Q^{Opt}(t+1)|^2$, $\mathbb{E}|Q^{Glo-Coor}(t+1)|^2$ and $\mathbb{E}|Q^{MP}(t+1)|^2$ denote the sum of squares of all the queue lengths returned by optimum, our algorithm and MP. One the one hand, we have that

$$\mathbb{E}|Q^{Opt}(t+1)|^2 \le \mathbb{E}|Q^{MP}(t+1)|^2. \tag{12}$$

On the other hand, the Max-sum_ADVP mechanism has proved to converge to the optimum in an acyclic graph. Fortunately, in generalized cyclic graphs, empirical results show that Max-sum_ADVP is adequately approximate the optimization, i.e., $\mathbb{E}|Q^{Glo-Coor}(t+1)|^2 \approx \mathbb{E}|Q^{Opt}(t+1)|^2$. Moreover, in the experiments, we also validate that $\mathbb{E}|Q^{Glo-Coor}(t+1)|^2 \le \mathbb{E}|Q^{MP}(t+1)|^2$. Combining these conclusions, we derive that the proposed Glo-Coor algorithm can also guarantee the stability Ineq.(11). □

5.2 Individual Action Improvement

The main object of the global joint coordination is to minimize the whole network queue length. Since these vehicles queuing at the internal link might not reduce network queue balance, the proposed global joint coordination might struggle to driven vehicles to exit links for network queue length minimization. However, such "clean" policy will increase the overall vehicles' waiting time, thereby reducing network throughput. We take an example to illustrate this drawback.

Algorithm 3: Best Response-based Individual Action Improvement(**BR-IA**)

Input : The coordination graph G.
Output: The joint action $\mathcal{X} = \{x_1, x_2, \cdots, x_n\}$.
1 Initial the joint action as the value **X** returned by Glo-Coor;
2 **while** *time budget is remained* **do**
3 **for** $a_i \in \mathcal{A}$ **do**
4 collect new action message from neighbors $Neg(i)$ as $\mathcal{X}_{Neg(i)}$;
5 Select the best response action $x_i^* = \arg\min_{x_i\in D_i, X_{Neg(i)}} B_i$ to minimize the intersection queue balance;

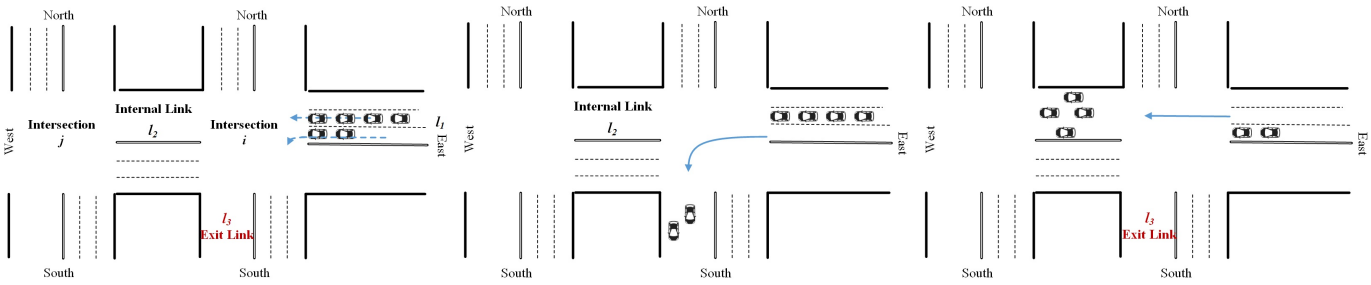
Example 1. *Fig.4(a) shows a toy traffic network including two intersections i and j, an entry link l_1, an internal link l_2 and an exit link l_3. Assume that there are four vehicles queuing at the movement phase (l_1, l_2) and two vehicles queuing at the movement phase (l_1, l_3). From the global coordination perspective (i.e., Algorithm 2), the intersection i should activate the WE-Left movement phase such that these vehicles on (l_1, l_3) will exit the network, and the vehicles on (l_1, l_2) are still waiting for next periods, where the network queue balance is $4^2 = 16$ (i.e., Fig.4(b)). Otherwise, activate the WE-Straight movement phase will cause both the vehicles on (l_1, l_2) to wait at link l_2 and the vehicles on (l_1, l_3) to wait for next period (i.e., Fig.4(c)), where the network queue balance is the $4^2 + 2^2 = 20$. However, it is straightforward that activate the WE-Straight movement phase will reduce sum of vehicles' waiting time since more vehicles get their destination closer.*

To avoid this "clean" policy cased by global coordination algorithm, we propose a best response-based local action improvement mechanism from the perspective of each individual agent a_i. The main idea is that after collecting the neighbors' actions (selected by the Glo-Coor algorithm), each a_i selects a best response action with the aim of minimizing his own intersection queue balance B_i (defined by Eq.(3)). Algorithm 3 illustrates this best response-based individual action improvement in detail. In step 4, each agents a_i collect the action information from neighbors $Neg(i)$ (The neighbor action information can be achieved either in a synchronization or asynchronous manner [50]). Given the neighbors' actions, each agent a_i selects the action x_i^* that can minimize its balance B_i in a best-response manner.

6 Experimental Evaluation

6.1 Experiment Setup

We evaluate the proposed TSC method on a Cityflow-like simulation platform [51] with large scale and heterogeneous intersections in real-world traffic networks and synthetic traffic networks. **??** shows the screenshot of our platform under several scenarios. We estimate queue update based on the original simulation way of Cityflow for the input of our proposed method. The car's traveling is set as a usual speed of 10m/s. In a traffic dataset, each vehicle is described as (o, t, d), where o is origin location, t is time, and d is destination location. Locations o and d are both locations on the road network. Traffic data is taken as input for simulator. In a multi-intersection network setting, we use the real road network to define the network in simulator. For a single intersection, unless



(a) A toy traffic network. (b) Global Joint Action Coordination. (c) Greedy Local Individual Action

Fig. 4: The necessary of greedy local individual action.

Dataset	# of Intersections	Arrival Rate (vehicles/s)	Duration (s)
syn_3× 3	9	0.84	3600
syn_4 × 4	16	1.76	3600
syn_20× 20	400	0.80	3600
syn_25× 25	625	0.77	3600

TABLE 3: Data statistics of synthetic traffic dataset.

Dataset	# of Intersections	Arrival Rate (vehicles/s)	Duration (s)
Jinan	3×4	0.84	3600
Hangzhou	4×4	1.76	3600
NewYork	1×16	0.80	3600
Manhattan	16×3	0.77	3600

TABLE 4: Data statistics of real-world traffic dataset.

otherwise specified, the road network is set to be a four-way intersection.

Synthetic Traffic Network. We use four synthetic data in our experiments, Detailed statistics are listed in Table 3. Large datasets with more than 400 intersections are used to evaluate effectiveness and effeciency of different methods. The vehicles come in different directions uniformly.

Real-World Traffic Network. We collect four representative traffic flow data from four cities to evaluate the performance of our model [13]. Detailed descriptions on these datasets are as follows, with data statistics listed in Table 4.

- Hangzhou: Flow data of 16 intersections in Gudang Sub-district generated from roadside surveillance cameras.
- Jinan: Flow data collected by roadside cameras from 12 intersections in Dongfeng Sub-district, Jinan, China.
- NewYork: Flow data of 16 intersections from 8-th Avenue in New york City with uni-directional traffic on both the arterial and the side streets.
- Manhattan: Flow data generated from open-source taxi trip data of 196 intersections in the road network of Manhattan, New york City.

Comparison Methods. We compare our EMC methods with the following two categories of methods: transportation methods and RL methods. Note that all methods are carefully tuned and their best results are reported. For a fair comparsion, all the RL methods are learned without pre-trained process and the action interval is set as 10 seconds. All RL methods are trained using 500 episodes. We report final results as the average of the last 5 evaluations for all methods.

- **FixedTime**: Fixed-time with random offset. Each phase has a fixed time of 15 seconds followed by a pre-determined plan. In this paper, there are four phases, i.e., WE-Straight, WE-Left, SN-Straight, SN-Left.
- **Maxpressure** [38]: Max pressure TSC method is the state-of-the-art network-level traffic signal control method, which greedily chooses the phase with the maximum pressure, as introduced in Section 4.3.
- **PressLight** [24]: PressLight TSC method is the state-of-the-art RL method. This method models each intersection as an independent agent and learns the policy of choosing next phase by vanilla DQN. The state is set as the queue length on lanes and reward is set as the pressure (as introduced in Section 4.3).
- **LIT**: LIT method is an individual deep reignforcement learning method which uses phase and vehicles numbers as the state, and use sum of queue length as the reward. Compared with PressLight, it does not consider the traffic condition from downstream.
- **MA2C**: Multi-agent A2C is a mutil-agent RL method. This method uses information of neighborhood policies to improve the observation of each local agent.
- **FMA2C**: FMA2C mathod is a start-of-the-art multi-agent RL method in solving TFC problem. it extends MA2C with feudal hierarchy. We corporate with its author and adopt identical parameters.

Performance Metric. We use the average travel time in seconds to evaluate the performance, which is the most frequently used measure in the transportation field and can be calculated as the average travel time of all vehicles spent in the system.

All computations are performed on a 64-bit workstation with 64 GB RAM and a 16-core 3.5 GHz processor. All records are averaged over 40 instances, and each record is statistically significant at 95% confidence level unless otherwise specified. The algorithm is parallelizable and hence this running time could be reduced substantially using multiple cores

Experiment Result. Table 5 shows the average traveling time performance of the proposed DCOP to the baseline methods in both real datasets and synthetic datasets. We have the following findings. DCOP method achieves evident performance improvement in both synthetic and real datasets compared with other transporation and RL methods. The potential reason is that the DCOP method enables direct modeling of the flow relationship between neighboring intersections and aims to optimize global traffic pressure. In comparsion, max-pressure only optimizes local pressure of individual intersection, thus cannot adapt to complex traffic situation. From the result we can see for max-pressure

	syn_3 × 3	syn_4 × 4	syn_15 × 15	syn_20 × 20	jinan_3 × 4	hangzhou_4 × 4	manhattan_16 × 3	ny_1 × 16
FixedTime	<0.1	<0.1	<0.1	<0.1	<0.1	<0.1	<0.1	<0.1
MaxPressure	<0.1	<0.1	<0.1	<0.1	<0.1	<0.1	<0.1	<0.1
LIT	16×3	0.77	3600					
PressLight	<0.1	<0.1	<0.1	<0.1	<0.1	<0.1	<0.1	<0.1
MA2C	<0.1	<0.1	<0.1	<0.1	<0.1	<0.1	<0.1	0.1002
FMA2C	<0.1	<0.1	<0.1	<0.1	<0.1	<0.1	<0.1	0.101
DCOP	0.118	0.169	1.276	2.013	0.155	0.165	0.268	0.174

TABLE 5: Performance comparison between all the methods w.r.t average travel time.

	syn_3 × 3	syn_4 × 4	syn_15 × 15	syn_20 × 20	jinan_3 × 4	hangzhou_4 × 4	manhattan_16 × 3	ny_1 × 16
FixedTime	205.16±0.00	232.24±0.00	207.11±0.00	331.88±0.00	331.23±0.00	377.48±0.00	292.32±0.00	199.49±0.00
MaxPressure	155.99(±0.00)	196.63(±0.00)	184.53(±0.00)	269.22(±0.00)	342.48(±0.00)	416.71(±0.00)	**180.38(±0.00)**	**102.82(±0.00)**
LIT	160.15(±0.12)	184.61(±0.10)	203.59(±0.16)	302(±0.21)	316.40(±0.01)	362.17(±0.13)	295.77(±0.06)	103.11(±0.01)
PressLight	148.89(±1.67)	184.82(±3.54)	206.57(±3.98)	345.53(±0.76)	301.32±8.91	365.01(±11.41)	182.36(±0.02)	106.23(±0.75)
MA2C	170.81(±2.47)	205.02(±0.49)	244.8(±0.69)	330.81(±2.73)	306.41(±0.34)	364.06(±1.826)	244.08(±1.26)	115.43(±0.50)
FMA2C	163.32(±0.90)	199.63(±0.80)	241.17(±0.45)	315.62(±3.01)	304.59(±0.85)	361.14(±1.25)	241.20(±2.49)	120.475(±0.95)
DCOP	**144.07(±0.19)**	**180.32(±0.20)**	**178.59(±0.07)**	**251.59(±0.04)**	**290.90(±0.07)**	**355.04(±0.76)**	182.48(±0.04)	104.31(±1.42)

TABLE 6: Performance comparison between all the methods w.r.t average decision time.

method there's performance gap between sythetic datasets and real datasets. The max-pressure method behaves evidently worse in the real scenenaries like jinan and hangzhou. Similar is the LIT method, which behaves fairly well in sythetic datasets but does not fit real data with irregular flow data like manhanttan. Presslight shows good result in these datasets except the large-size sythetic data. Such divergence conforms to the deficency in individual rl methods, that it is incapable of capturing communication between neighboring intersections in large-scale or complex scenaries. The performance advantage of FMA2C over MA2C corresponds to the FMA2C method's optimization effect in serving global coordination. These methods' parameters are hard to train thus not easily adapt to the experiment tasks. And for large size data, the trainging network's size has to be shortened, which influence their performace.

Table 4 shows the average decision time of each method. Even in the largest-size data, our method's decision time is significantly less than 5 seconds(a reasonable maximum time limit for traffic signal control). Considering our method uses any-time mechanism, which can visit more action combinations given more time budget and can return a best solution under the existing time, it enables scaling to large-size TSC problems with hundreds of regions.

Case study:

In this case, we make observations on the vehicle flow movement in the real data of city Jinan in the time range from 1500s to 1800s. We use the time-space diagram to illustrate the trajectory of each vehicle and compare the performance of our method with max-pressure method. In the time-space diagram, the x-axis is the time and the y-axis is the distance of the westernmost main road from south to north. The three green-yellow-red color bands represent the three intersections' phase schedule. The flow arrival rate is set as 1.68/s, which is a peak time for the traffic system. It can be found that under the max-pressure method's control, many cars stop and accumulate before the red light. As the traffic is too heavy, the max-pressure method has to keep the road stopped passing long time to make way for other high-demand roads. In comparsion, most orange and blue lines are straight in the DCOP traffic graph, which indicates few cars are stopped by the red light. The DCOP method can detect the actual traffic change on each road and optimize the next stage's global pressure in the roadnet, which enables its finer granularity of control and thus can make flexible scheduling to arrange short-time green light phase for incoming cars to pass through.

7 Conclusion and Future Work

This paper studies Unfortunately, MBDP's communication cost (i.e., message size) requirements still grow exponentially with the size of the observation set. Even for very small problems this can be prohibitive. POMDP is a method of reducing messaging where each agent only observes his own local information. have issues related to large message sizes

Acknowledgment

This research is supported by National Key Research and Development Project of China (2019YFB1405000), the National Natural Science Foundation of China (61932007, 61806053 and 61807008), the Natural Science Foundation of Jiangsu Province of China (BK20171363, BK20180356, and BK20180369).

References

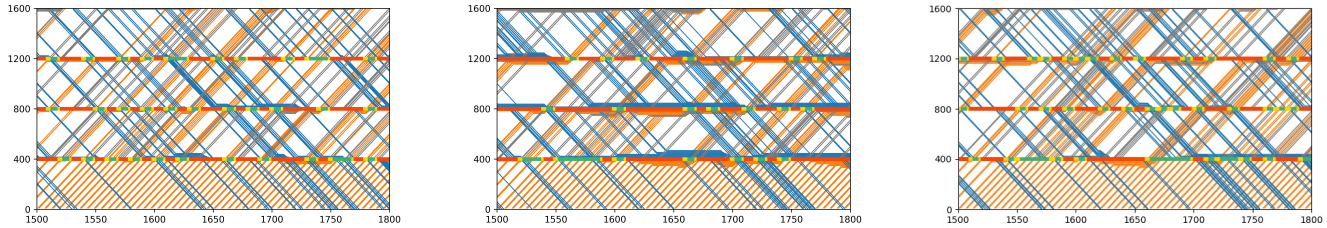
[1] P. Lowrie, "Scats: Sydney co-ordinated adaptive traffic system: a traffic responsive method of controlling urban traffic," 1990.

[2] D. Robertson and R. Bretherton, "Optimizing networks of traffic signals in real time-the scoot method," *IEEE Transactions on Vehicular Technology*, vol. 40, no. 1, pp. 11–15, 1991.

[3] S. F. Smith, G. J. Barlow, X. Xie, and Z. B. Rubinstein, "Smart urban signal networks: Initial application of the SURTRAC adaptive traffic signal control system," in *Proceedings of the Twenty-Third International Conference on Automated Planning and Scheduling(ICAPS'13), Rome, Italy, June 10-14, 2013*, 2013.

[4] H. Hu and S. F. Smith, "Using bi-directional information exchange to improve decentralized schedule-driven traffic control," in *Proceedings of the Twenty-Ninth International Conference on Automated Planning and Scheduling, ICAPS 2018, Berkeley, CA, USA, July 11-15, 2019*, 2019, pp. 200–208.

[5] M. A. Wiering, "Multi-agent reinforcement leraning for traffic light control," in *Proceedings of the Seventeenth International Conference on Machine Learning (ICML 2000), Stanford University, Stanford, CA, USA, June 29 - July 2, 2000*, 2000, pp. 1151–1158.



(a) DCOP Method (b) Max-pressure Algorithm (c) PressLight Algorithm

Fig. 5: Time-Space Diagram of Algorithms

[6] H. Wei, C. Chen, G. Zheng, K. Wu, V. V. Gayah, K. Xu, and Z. Li, "Presslight: Learning max pressure control to coordinate traffic signals in arterial network," in *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining, KDD 2019, Anchorage, AK, USA, August 4-8, 2019*, 2019, pp. 1290–1298.

[7] H. Wei, G. Zheng, V. V. Gayah, and Z. Li, "Recent advances in reinforcement learning for traffic signal control: A survey of models and evaluation," *SIGKDD Explor.*, vol. 22, no. 2, pp. 12–18, 2020.

[8] C. Chen, H. Wei, N. Xu, G. Zheng, M. Yang, Y. Xiong, K. Xu, and Z. Li, "Toward A thousand lights: Decentralized deep reinforcement learning for large-scale traffic signal control," in *The Thirty-Fourth AAAI Conference on Artificial Intelligence, AAAI 2020, The Thirty-Second Innovative Applications of Artificial Intelligence Conference, IAAI 2020, The Tenth AAAI Symposium on Educational Advances in Artificial Intelligence, EAAI 2020, New York, NY, USA, February 7-12, 2020*, 2020, pp. 3414–3421.

[9] 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, 2013.

[10] L. Kuyer, S. Whiteson, B. Bakker, and N. A. Vlassis, "Multiagent reinforcement learning for urban traffic control using coordination graphs," in *Machine Learning and Knowledge Discovery in Databases, European Conference, ECML/PKDD 2008, Antwerp, Belgium, September 15-19, 2008, Proceedings, Part I*, 2008, pp. 656–671.

[11] E. van der Pol and F. Oliehoek, "Coordinated deep reinforcement learners for traffic light control," in *NIPS-16*, 2016.

[12] T. Chu, J. Wang, L. Codeca, and Z. Li, "Multi-agent deep reinforcement learning for large-scale traffic signal control," *IEEE Transactions on Intelligent Transportation Systems*, vol. 21, no. 3, pp. 1086–1095, 2020.

[13] H. Wei, N. Xu, H. Zhang, G. Zheng, X. Zang, C. Chen, W. Zhang, Y. Zhu, K. Xu, and Z. Li, "Colight: Learning network-level cooperation for traffic signal control," in *Proceedings of the 28th ACM International Conference on Information and Knowledge Management, CIKM 2019, Beijing, China, November 3-7, 2019*, 2019, pp. 1913–1922.

[14] Z. Yu, S. Liang, L. Wei, Z. Jin, J. Huang, D. Cai, X. He, and X. Hua, "Macar: Urban traffic light control via active multi-agent communication and action rectification," in *Proceedings of the Twenty-Ninth International Joint Conference on Artificial Intelligence, IJCAI 2020*, 2020, pp. 2491–2497.

[15] T. Tan, F. Bao, Y. Deng, A. Jin, Q. Dai, and J. Wang, "Cooperative deep reinforcement learning for large-scale traffic grid signal control," *IEEE Transactions on Cybernetics*, vol. 50, no. 6, pp. 2687–2700, 2020.

[16] Z. Zhang, J. Yang, and H. Zha, "Integrating independent and centralized multi-agent reinforcement learning for traffic signal network optimization," in *Proceedings of the 19th International Conference on Autonomous Agents and Multiagent Systems, AAMAS'20, Auckland, New Zealand, May 9-13, 2020*, 2020, pp. 2083–2085.

[17] B. Xu, Y. Wang, Z. Wang, H. Jia, and Z. Lu, "Hierarchically and cooperatively learning traffic signal control," in *Thirty-Fifth AAAI Conference on Artificial Intelligence, AAAI 2021, Virtual Event, February 2-9*, 2021, pp. 669–677.

[18] B.-L. Ye, W. Wu, K. Ruan, L. Li, T. Chen, H. Gao, and Y. Chen, "A survey of model predictive control methods for traffic signal control," *IEEE/CAA Journal of Automatica Sinica*, vol. 6, no. 3, pp. 623–640, 2019.

[19] X. Xie, S. F. Smith, and G. J. Barlow, "Schedule-driven coordination for real-time traffic network control," in *Proceedings of the Twenty-Second International Conference on Automated Planning and Scheduling, ICAPS 2012, Atibaia, São Paulo, Brazil, June 25-19, 2012*, 2012.

[20] R. Junges and A. L. C. Bazzan, "Evaluating the performance of DCOP algorithms in a real world, dynamic problem," in *7th International Joint Conference on Autonomous Agents and Multiagent Systems (AAMAS 2008), Estoril, Portugal, May 12-16, 2008, Volume 2*, 2008, pp. 599–606.

[21] R. Zivan and H. Peled, "Max/min-sum distributed constraint optimization through value propagation on an alternating DAG," in *International Conference on Autonomous Agents and Multiagent Systems, AAMAS 2012, Valencia, Spain, June 4-8, 2012 (3 Volumes)*, 2012, pp. 265–272.

[22] Z. Chen, Y. Deng, and T. Wu, "An iterative refined max-sum_ad algorithm via single-side value propagation and local search," in *Proceedings of the 16th Conference on Autonomous Agents and MultiAgent Systems, AAMAS 2017, São Paulo, Brazil, May 8-12, 2017*, K. Larson, M. Winikoff, S. Das, and E. H. Durfee, Eds., 2017, pp. 195–202.

[23] H. Wei, G. Zheng, H. Yao, and Z. Li, "Intellilight: A reinforcement learning approach for intelligent traffic light control," in *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining, KDD 2018, London, UK, August 19-23, 2018*, 2018, pp. 2496–2505.

[24] H. Wei, C. Chen, G. Zheng, K. Wu, V. Gayah, K. Xu, and Z. Li, "Presslight: Learning max pressure control to coordinate traffic signals in arterial network," in *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, ser. KDD'19, 2019, pp. 1290–1298.

[25] Y. Liu, L. Liu, and W. Chen, "Intelligent traffic light control using distributed multi-agent Q learning," in *20th IEEE International Conference on Intelligent Transportation Systems, ITSC 2017, Yokohama, Japan, October 16-19, 2017*, 2017, pp. 1–8.

[26] J. Ma and F. Wu, "Feudal multi-agent deep reinforcement learning for traffic signal control," in *Proceedings of the 19th International Conference on Autonomous Agents and MultiAgent Systems*, ser. AAMAS '20. International Foundation for Autonomous Agents and Multiagent Systems, 2020, pp. 816–824.

[27] C. Cai, C. K. Wong, and B. G. Heydecker, "Adaptive traffic signal control using approximate dynamic programming," *Transportation Research Part C: Emerging Technologies*, vol. 17, no. 5, pp. 456–474, 2009.

[28] S. Chen and D. J. Sun, "An improved adaptive signal control method for isolated signalized intersection based on dynamic programming," *IEEE Intelligent Transportation Systems Magazine*, vol. 8, no. 4, pp. 4–14, 2016.

[29] K. Aboudolas, M. Papageorgiou, A. Kouvelas, and E. Kosmatopoulos, "A rolling-horizon quadratic-programming approach to the signal control problem in large-scale congested urban road networks," *Transportation Research Part C: Emerging Technologies*, vol. 18, no. 5, pp. 680–694, 2010, applications of Advanced Technologies in Transportation: Selected papers from the 10th AATT Conference.

[30] S. Lin, B. D. Schutter, Y. Xi, and H. Hellendoorn, "Fast model predictive control for urban road networks via MILP," *IEEE Trans. Intell. Transp. Syst.*, vol. 12, no. 3, pp. 846–856, 2011.

[31] L. B. de Oliveira and E. Camponogara, "Multi-agent model predictive control of signaling split in urban traffic networks," *Transportation Research Part C: Emerging Technologies*, vol. 18, no. 1, pp. 120–139, 2010.

[32] Z. Zhou, B. D. Schutter, S. Lin, and Y. Xi, "Two-level hierarchical model-based predictive control for large-scale urban traffic networks," *IEEE Transactions on Control Systems Technology*, vol. 25, no. 2, pp. 496–508, 2017.

[33] H. Hu and S. F. Smith, "Softpressure: A schedule-driven backpressure algorithm for coping with network congestion," in *Proceedings of the Twenty-Sixth International Joint Conference on Artificial Intelligence,*

IJCAI 2017, Melbourne, Australia, August 19-25, 2017, 2017, pp. 4324–4330.

[34] R. Goldstein and S. F. Smith, "Expressive real-time intersection scheduling," in *Proceedings of the Thirty-Second AAAI Conference on Artificial Intelligence, (AAAI-18), the 30th innovative Applications of Artificial Intelligence (IAAI-18), and the 8th AAAI Symposium on Educational Advances in Artificial Intelligence (EAAI-18), New Orleans, Louisiana, USA, February 2-7, 2018*, 2018, pp. 6177–6185.

[35] N. Wu, D. Li, and Y. Xi, "Distributed weighted balanced control of traffic signals for urban traffic congestion," *IEEE Transactions on Intelligent Transportation Systems*, 2021, in press.

[36] B. Scholkopf, J. Platt, and T. Hofmann, *Natural Actor-Critic for Road Traffic Optimisation*, 2007, pp. 1169–1176.

[37] J. Ault, J. P. Hanna, and G. Sharon, "Learning an interpretable traffic signal control policy," in *Proceedings of the 19th International Conference on Autonomous Agents and MultiAgent Systems*, ser. AAMAS'20, 2020, pp. 89–96.

[38] P. Varaiya, "Max pressure control of a network of signalized intersections," *Transportation Research Part C: Emerging Technologies*, vol. 36, pp. 177–195, 2013.

[39] S. G. Rizzo, G. Vantini, and S. Chawla, "Time critic policy gradient methods for traffic signal control in complex and congested scenarios," in *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, ser. KDD '19. New York, NY, USA: Association for Computing Machinery, 2019, pp. 1654–1664.

[40] B. Bakker, S. Whiteson, L. Kester, and F. C. A. Groen, *Traffic Light Control by Multiagent Reinforcement Learning Systems*, 2010, pp. 475–510.

[41] A. L. C. Bazzan, "Opportunities for multiagent systems and multiagent reinforcement learning in traffic control," *Autonomous Agents and Multi-Agent Systems*, vol. 18, pp. 342–375, 2009.

[42] Y. Lv, Y. Duan, W. Kang, Z. Li, and F. Wang, "Traffic flow prediction with big data: A deep learning approach," *IEEE Trans. Intell. Transp. Syst.*, vol. 16, no. 2, pp. 865–873, 2015.

[43] B.-L. Ye, W. Wu, and W. Mao, "A two-way arterial signal coordination method with queueing process considered," *IEEE Transactions on Intelligent Transportation Systems*, vol. 16, no. 6, pp. 3440–3452, 2015.

[44] F. Ashtiani, S. A. Fayazi, and A. Vahidi, "Multi-intersection traffic management for autonomous vehicles via distributed mixed integer linear programming," in *2018 Annual American Control Conference (ACC)*, 2018, pp. 6341–6346.

[45] T. H. Heung, T. Ho, and Y. Fung, "Coordinated road-junction traffic control by dynamic programming," *IEEE Trans. Intell. Transp. Syst.*, vol. 6, no. 3, pp. 341–350, 2005.

[46] S. Dhamija, A. Gon, P. Varakantham, and W. Yeoh, "Online traffic signal control through sample-based constrained optimization," in *Proceedings of the Thirtieth International Conference on Automated Planning and Scheduling, Nancy, France, October 26-30, 2020*, J. C. Beck, O. Buffet, J. Hoffmann, E. Karpas, and S. Sohrabi, Eds., 2020, pp. 366–374.

[47] P. Mirchandani and L. Head, "A real-time traffic signal control system: architecture, algorithms, and analysis," *Transportation Research Part C: Emerging Technologies*, vol. 9, no. 6, pp. 415–432, 2001.

[48] H. Vu, S. Aknine, and S. D. Ramchurn, "A decentralised approach to intersection traffic management," in *Proceedings of the Twenty-Seventh International Joint Conference on Artificial Intelligence, IJCAI 2018, July 13-19, 2018, Stockholm, Sweden*, J. Lang, Ed., 2018, pp. 527–533.

[49] N. Vlassis, R. Elhorst, and J. Kok, "Anytime algorithms for multiagent decision making using coordination graphs," in *2004 IEEE International Conference on Systems, Man and Cybernetics (IEEE Cat. No.04CH37583)*, 2004, pp. 953–957.

[50] W. Zhang, G. Wang, Z. Xing, and L. Wittenburg, "Distributed stochastic search and distributed breakout: properties, comparison and applications to constraint optimization problems in sensor networks," *Artif. Intell.*, vol. 161, no. 1-2, pp. 55–87, 2005.

[51] H. Zhang, S. Feng, C. Liu, Y. Ding, Y. Zhu, Z. Zhou, W. Zhang, Y. Yu, H. Jin, and Z. Li, "Cityflow: A multi-agent reinforcement learning environment for large scale city traffic scenario," in *The World Wide Web Conference*, ser. WWW '19, 2019, pp. 3620–3624.

Wanyuan Wang is an associate professor with the School of Computer Science and Engineering, Southeast University, Nanjing, China. He received his PhD. degree in computer science from Southeast University, China, 2016. He has published several articles in refereed journals and conference proceedings, such as the IEEE Transactions, AAAI, and AAMAS. He won the best student paper award from ICTAI14. His main research interests include artificial intelligence, multiagent systems, and game theory.

Bo An is a President's Council Chair Associate Professor in Computer Science and Engineering, Nanyang Technological University, Singapore. He received the Ph.D degree in Computer Science from the University of Massachusetts, Amherst. His current research interests include artificial intelligence, multiagent systems, computational game theory, reinforcement learning, and optimization. His research results have been successfully applied to many domains including infrastructure security and e-commerce. He has published over 100 referred papers at AAMAS, IJCAI, AAAI, ICAPS, KDD, UAI, EC, WWW, ICLR, NeurIPS, JAAMAS, AIJ and ACM/IEEE Transactions. Dr. An was the recipient of the 2010 IFAAMAS Victor Lesser Distinguished Dissertation Award, an Operational Excellence Award from the Commander, First Coast Guard District of the United States, the 2012 INFORMS Daniel H. Wagner Prize for Excellence in Operations Research Practice, and 2018 Nanyang Research Award (Young Investigator). His publications won the Best Innovative Application Paper Award at AAMAS'12 and the Innovative Application Award at IAAI'16. He was invited to give Early Career Spotlight talk at IJCAI'17. He led the team HogRider which won the 2017 Microsoft Collaborative AI Challenge. He was named to IEEE Intelligent Systems' 'AI's 10 to Watch' list for 2018. He is PC Co-Chair of AAMAS'20. He is a member of the editorial board of JAIR and the Associate Editor of JAAMAS, IEEE Intelligent Systems, and ACM TIST. He was elected to the board of directors of IFAAMAS and senior member of AAAI.

Yichuan Jiang is a full professor with the School of Computer Science and Engineering, Southeast University, Nanjing, China. He received his PhD degree in computer science from Fudan University, Shanghai, China, in 2005. His main research interests include multiagent systems, social computing, and social networks. He has published more than 90 scientific articles in refereed journals and conference proceedings, such as the IEEE Transactions on Parallel and Distributed Systems, IEEE Journal on Selected Areas in Communications, IEEE Transactions on Systems, Man, and Cybernetics-Part A: Systems and Humans, IEEE Transactions on Systems, Man, and Cybernetics-Part C: Applications and Reviews, IEEE Transactions on Systems, Man, and Cybernetics: Systems, IEEE Transactions on Cybernetics, the ACM Transactions on Autonomous and Adaptive Systems, the Journal of Autonomous Agents and Multi-Agent Systems, the Journal of Parallel and Distributed Computing, IJCAI, AAAI and AAMAS. He won the best paper award from PRIMA06 and best student paper awards twice from ICTAI13 and ICTAI14. He is a senior member of IEEE.