# Real-Time Network-Wide Traffic Signal Control: An Explicit Multiagent Coordination Approach

Wanyuan Wang, Tianchi Qiao, Bo An, and Yichuan Jiang

***Abstract*—Intersection 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 a 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**

## I. 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number of vehicles traveling on saturated urban road networks, these pre-calculated TSC rules are quickly become outdated and traffic congestion has become the key inconvenience to drivers.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 . 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

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/

(DCOP) [20], [21]. 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.

## II. 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.

### A. 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], [22]. 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 [23]. Wei et al. [24] 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 [25], Wei et al. [26] further propose a "pressure" reward that has the network flow stability guarantee. For the roundabout intersections with heavy traffic volumes, Rizzo et al.[27] 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 [39], [8]. However, some intersections are tightly coupled, without coordination, congestion starts from the upstream intersections will spread to downstream intersections [40]. 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. [28] 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 [29].

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).

### B. 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 [41], which can be used to build a linear program (LP) [31] or Markov decision process (MDP) model [30]. Efficient dynamic programming is adopted to generate the real-time strategies. Building the traffic flow update model, the quadratic-program [32] and the mixed-integer linear program (MILP) [33] can be used to optimize signal control. To alleviate the computation problem, Zhou et al. [35] 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

| Approaches | | Response Time (seconds) | Coordination | | Network-Wide Scalability # of Intersections |
|---|---|---|---|---|---|
| | | | Local | Global | |
| Reinforcement Learning | Independent RL [24], [26], [8] | $\leq$3 | ✖ | ✖ | $\sim$ 2500 |
| | Coordinated RL [10], [9], [11], [28], [13], [14] | $\leq$3 | ✔ | ✔ | $\sim$ 200 |
| | Hierarchical RL [15], [16], [29] | $\leq$3 | ✔ | ✔ | $\leq$36 |
| Model Predictive Control (MPC) | Centralized MPC [30], [31], [32], [33] | $\geq$3 | ✔ | ✖ | $\geq$15 |
| | Decentralized MPC [34], [35] | $\geq$80 | ✔ | ✔ | $\leq$55 |
| Online Planning [19], [3], [36], [37], [4], [38] | | $\leq$3 | ✔ | ✖ | $\leq$25 |
| Our MAC Approach | | $\leq$3 | ✔ | ✔ | $\sim$ 400 |

TABLE I: Summary of the related literature.

a centralized MPC controller for signal strategies coordination [42]. However, these centralized MPC solutions to traffic regulation result in high computational requirements, and create a single point of failure [43]. For example, Heung et al. [44] 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. [34] decompose the centralized MILP formulations into separable optimizations, each can be computed by an independent agent in parallel [35]. 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 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.

### C. 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], [37]. 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 [36] use the link softpressure as the weight and Wu et al. [38] 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 [45].

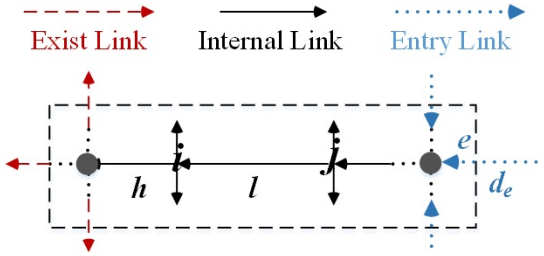


Fig. 1: Schematic of traffic network. Links are denoted by arrows, and intersections by dark circles. Intersections $i$ and $j$ are neighbors.

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.

## III. Model

This section first presents the TSC problem, subsequently, present the DCOP formulation to model TSC.

### A. TSC Problem Description

**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. Let $In_l$ and $Out_l$ denote the set of links input to and output from the link $l$, respectively. Fig.(1) depicts the traffic network.

As in previous work [19], we assume turning proportions at each intersection are available. In practice, quite accurate parameters can be estimated by using measured data. **Movement phase.** A *movement phase* $(l, h)$ 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 TSC 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 Figure 2, there are twelve incoming lanes and outgoing lanes, respectively. These twelve traffic movements 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]. Time is discretized into periods, $t = 1, 2, \cdots, T$, each includes a fixed duration of $\tau$ seconds (e.g., 20 seconds). Let $f(l,h)$ denote the saturation flow of movement phase $(l,h)$, i.e., if phase $(l,h)$ 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.

**Network State Update.** Let $q_{max}(l)$ denote the capacity of link $l$, i.e., the maximum permissible vehicle number in $l$. Let $q(l,h)$ denote the number of vehicles leaving the link $l$ and entering the link $h$, and $q(l) = \sum_{h \in Out_l} q(l,h)$ denote the number of vehicles (i.e., queue length) in $l$. 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 [25]:

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

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 the next period $t+1$. This parameter $r(l,h)(t+1)$ indicates the proportion of vehicles leaving $l$ and entering $h$, which can be achieved from the route navigation systems. We assume turning proportions $r(l,h)(t+1)$ at each intersection are available [46].

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:

$$q(l,h)(t+1) = q(l,h)(t) - f(l,h)(t)x(l,h)(t) \wedge q(l,h)(t) + d(l)r(l,h)(t+1). \quad (2)$$

where $d(l)$ denote the exogenous flow of vehicles in entry link $l \in \mathcal{L}_{entry}$.

[2]It is worth noting that 1) the right 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

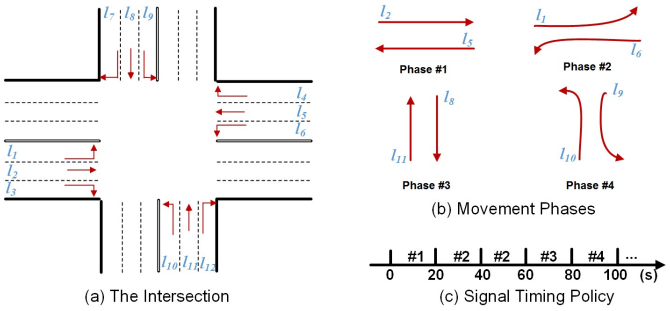


Fig. 2: Intersection, movement phases and signal control policy in ITSC.

**The Objective.** Let $\frac{q(l,h)}{q_{max}(l)}$ denote the density of phase $(l,h)$. At period $t$, given the queue state, $Q_i(t) = \{q(l,h) | l \in I(i), h \in O(i)\}$ at the intersection $i$, the *balance* at $i$ is defined as the sum of squares of all the queue density, i.e.,

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

Intuitively, given the same number of vehicles waiting at the intersection $i$, the less the value $B_i(t)$, the more balance the queue length $q(l,h)$, thereby the average travel time of vehicles can be reduced. Formally, 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 $\{Q(t), f(l,h)(t), d(l)(t), r(l,h)\}$.

### *B. Explicit Multiagent Coordination Formulation for TSC*

In this section, we will provide an explicit multiagent coordination (EMC) framework to formulate the TSC problem.

Given the real TSC problem, a Distributed Constraint Optimization Problem (DCOP) provides the potential as an useful and appropriate EMC approach [47]. A DCOP is defined as a tuple $\langle \mathcal{A}, \mathcal{X}, \mathcal{D}, \mathcal{C} \rangle$ such that: $\mathcal{A} = \{a_1, a_2, \cdots, a_n\}$ is the set of agents, $\mathcal{X} = \{x_1, x_2, \cdots, x_n\}$ is the set of variables, each is controlled by an agent, and takes the value from the finite discrete domain $D_i \in \mathcal{D}$, and $\mathcal{C} = \{c_1, c_2, \cdots, c_m\}$ is a set of constraints, where the constraint cost $c_i$ is defined as a mapping from the assignments of the involved $k$ variables $(x_{i_1}, \cdots, x_{i_k})$ to a positive real, $c_i : D_{i_1} \times D_{i_2} \times \cdots D_{i_k} \to \mathbb{R}_{\geq 0}$. A solution to a DCOP is an assignment to all variables that minimizes the total cost, which is the sum of all constraint costs: $\mathcal{X}^* = \arg\min_{\mathcal{X} \in \mathcal{D}} \sum_{c_i \in \mathcal{C}} c_i(x_{i_1}, \cdots, x_{i_k})$.

The key step toward modeling the TSC problem as a DCOP is mapping intersection, movement phase, and objective to the four tuples in the DCOP model. By capturing the structure of TSC, a straightforward DCOP model can be formulated where each agent $a_i$ in DCOP represents an intersection $i$ in TSC. Throughout this paper, we use agent and intersection

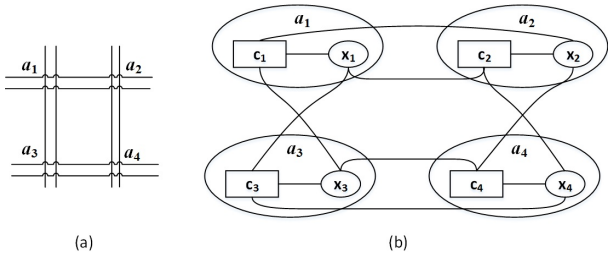


Fig. 3: An illustration of factor graph.

interchangeable. Each agent $a_i$ holds a decision variable $x_i$ determining the movement phase. The domain of the variable $x_i$ is denoted by $D_i = \{1, 2, 3, 4\}$, which corresponds to four movement phases, i.e., 1=*WE-Straight*, 2=*WE-Left*, 3=*SN-Straight* and 4=*SN-Left* is activated.

**The Intersection Balance as the Constraint Cost.** Given a particular variable assignment, one challenge is to construct constraint cost such that when the resulting DCOP is solved, we obtain a solution that is congruent to the original TSC problem. Each constraint cost $c_i$ is defined as the balance of each agent $a_i$, which is associated with the variable assignments of $a_i$ as well as his neighbor agents $Neg(i)$

$$c_i(x_i, \mathcal{X}(Neg(i)))=B_i=\sum_{(l,h):l\in I(i),h\in O(i)}[\frac{q(l,h)}{q_{max}(l)}]^2. \quad (5)$$

where $x_i$ is the value assignment of agent $a_i$ and $\mathcal{X}(Neg(i))$ is the value assignment of neighbor agents $Neg(i)$.

The object of the DCOP is to minimize $\sum_{c_i\in\mathcal{C}} c_i(x_i, \mathcal{X}(Neg(i)))$, in consistent with the objective of minimizing the network-level balance in TSC.

## IV. The Real-Time Algorithm for City-Scale DCOP

This section presents an efficient and real-time algorithm for city-scale DCOPs. We first review the state-of-the-art Max-sum_ADVP algorithm for DCOPs[20], [21], and improve it by optimizing the message passing order.

### A. Algorithm Preliminaries

To solve the DCOPs modeled above, we would like to employ and improve the Max-sum_ADVP (Max-sum through value propagation on an alternating directed acyclic graph) algorithm, an incomplete DCOP algorithm based on the exchange of messages between agents. Despite the fact that our formulation is compatible with any complete or incomplete DCOP algorithm, we chose to use Max-sum_ADVP as it is one of the fastest and most efficient algorithms in many multiagent domains. In the following, we first describe standard Max-sum_ADVP algorithm. It should be noted that the TSC is actually a Min-sum problem, however, we will continue to refer to it as Max-sum since this name is widely accepted.

*1) Standard Max-Sum_ADVP Algorithm:* In this section, we review the standard Max-sum_ADVP algorithm for DCOPs, which mainly consists of two processes, i.e., transforming the DCOPs to factor graphs and determining the message-passing order in the transformed factor graphs.

**Algorithm 1:** Message-Passing (node $i$,$\langle o, dia(o)\rangle$)

```
while time budget is remained do
    Neg_pre_i ← {ĵ ∈ Neg(i) :
        ĵ is before i in order o };
    Neg_follow_i ← Neg(i) \ Neg_pre_i;
    for dia(o) iterations do
        collect messages from Neg_pre_i;
        for j ∈ Neg_follow_i do
            if j is a variable node then
                produce message Q_{i→j} using messages
                  from Neg(i) \ {j};
            if j is a function node then
                produce message R_{i→j} using constraint
                  and messages from Neg(i) \ {j};
            send Q_{i→j} or R_{i→j} to j;
    current-order o ← reverse(o).
```

**Algorithm 2:** Max-Sum_ADVP (node $i$,$\langle o, dia(o)\rangle$)

**Input :** The factor graph.
**Output:** The action $x_i$ of each agent $a_i$.

```
current_order ← o;
backward_order ← reverse(o);
perform Message-Passing(i,⟨current_order, dia(o)⟩);
current-order o ← reverse(o);
perform Max-sum_AD(i,⟨backward_order, dia(o)⟩);
while time budget is remained do
    x_i^* ← current optimal decision;
    perform Message-Passing with value assignment
      x_i^* on alternate order;
```

**Transforming DCOPs to Factor Graphs.** Max-sum_ADVP operates on a factor graph: a bipartite, undirected graph, that contains a variable node for each variable $x_i$, a function node for each cost function $c_j$, and an edge connecting a variable node $x_i$ with a function node $c_j$ if and only if $x_i$ is involved in $c_j$. Each agent $a_i$ in takes the role of the variable node which represents its own variable $x_i$ and cost function $c_i$. Variable nodes and function nodes are considered "agents" in Max-sum, i.e., they can send messages, read messages and perform computation. Fig.3 shows the factor graph-based intersection. In Fig.3(a), there are four intersections $\{a_1, a_2, a_3, a_4\}$. In Fig.3(b) each intersection $a_i$ consists of the variable node $x_i$ (i.e., labeled as oval) and cost function node $c_i$ (labeled as rectangle). The connections between an variable node $x_i$ and a function node $c_j$ indicates that $c_j$ is a function of $x_i$.

The operation for variable and function nodes is similar apart from the content of messages to be sent. A message sent from a variable node $i$ to a function node $j$ at iteration $k$, includes for each value $x_i \in D_i$ the sum of costs for this value it received from all function neighbors apart from $j$ at iteration $k-1$. Formally, the message from variable $i$ to function $j$

includes for each value $x_i \in D_i$

$$Q_{i\to j}(x_i) = \sum_{j'\in Neg(i), j'\neq j} R_{j'\to i}(x_i) - \alpha. \quad (6)$$

where $R_{j'\to i}(x_i)$ is the cost for value $x_i$ included in the messages received from the function node $j'$ at iteration $k-1$. $\alpha$ is a constant that is reduced from all costs included in the message (i.e., for each value $x_i \in D_i$) in order to prevent the costs carried by messages from growing arbitrarily. A reasonable choice of $\alpha$ is the average on all costs included in the message, i.e., $\alpha = \sum_{x_i\in D_i}\sum_{j'\in Neg(i), j'\neq j} R_{j'\to i}(x_i)/|D_i|$. As long as the amount reduced from all costs is identical, the algorithm is not affected by this reduction since only the differences between the costs for the different values matter.

A message sent from function node $j$ to variable node $i$ at iteration $k$ includes for each possible value $x_i \in D_i$ the minimal cost of any combination of assignments to the variables involved in $j$ apart from $i$ and the assignment of value $x_i$ to the variable node $i$. Formally, the message from function $j$ to variable $i$ includes for each value $x_i \in D_i$:

$$R_{j\to i}(x_i) = \min_{\mathcal{X}(Neg(j)\setminus i)} \big(c(\mathcal{X}(Neg(j))) + \sum_{i'\in Neg(j), i'\neq i} Q_{i'\to j}(x_{i'})\big). \quad (7)$$

where $c(\mathcal{X}(Neg(j)))$ is the local function that represents the cost involving variable nodes $Neg(j)$.

When a variable node makes decisions, it accumulates all costs it receives, and selects a value to minimize the sum costs. The value selection procedure for each variable node $i$ can be formalized by

$$x_i^* = \arg\max_{x_i} \sum_{j\in Neg(i)} R_{j\to i}(x_i). \quad (8)$$

**Determining the Message-Passing Order on Factor Graphs.** In order to avoid the pathology caused by cycles, we need to select an order on all nodes in the factor graph. Existing Max-sum_ADVP variants [20], [21] determine node order according to the indices of agents. A node whose role is performed by agent $a_i$ is ordered before a node whose role is performed by agent $a_j$ if $i < j$. Given the order of agents, for variable and function nodes held by the same agent, a variable-node is ordered before the function-nodes. Let $o$ denote the message-passing order and $dia(o)$ denote the diameter (i.e., the longest distance between two nodes in the factor graph) of the order. Give the order-tuple $\langle o, dia(o)\rangle$, let $Neg_{pre_i}$ and $Neg_{follow_i}$ denote these nodes ordered "before" and "after" the node $i$.

**Max-Sum_ADVP Algorithm.** Algorithm 1 first presents a sketch of the message-passing process. After $dia(o)$ iterations in current order $o$ (Steps 5-12), the order is reversed and messages are sent for the next $dia(o)$ iterations only in the opposite direction (Step 13). In each iteration, variable (resp. function) node aggretages the message received from "before" function (resp. variable) nodes and sent messages to "after" function (resp. variable) nodes acoording to Eq.(6) (resp. Eq.(7)).

**Algorithm 3:** Minimum Diameter Order (**MDO**)

**Input :** The traffic network $G$.
**Output:** The sink agent $a_s$ and the message-passing order $o$.

1 Initialize $dia = +\infty$, $o = \emptyset$;
2 **for** $a_i \in A$ **do**
3  $dia(a_i) = \max_{a_j\in A} d(a_i, a_j)$;
4  **if** $dia(a_i) < dia$ **then**
5  $dia = dia(a_i)$, $a_s = a_i$; // sink agent determination
6 **for** $a_i, a_j \in A : l_{ij} \in \mathcal{L}_{all}$ **do**
7  **if** $d(a_s, a_i) \leq d(a_s, a_j)$ **then**
8  $o_{ij} = a_j \to a_i$;
9  **else**
10  $o_{ij} = a_i \to a_j$;

**Lemma 1.** *[20]. Given the order $o$ of the factor graph, after $dia(o)$ iterations of message-passing, the content of the messages each node $i$ receives does not change as long as messages are sent according to $o$, i.e., the message-passing process converges after $dia(o)$ iterations.*

Finally, we formulate the Max-Sum_ADVP algorithm in Algorithm 2. In steps 3 and 5, the node $i$ performs the message-passing processes on the forward direction order and backward direction order, respectively. After message-passing converge in both directions, value propagation procedure starts, where the value assignment $x_i^*$ of each variable node $i$ is computed by 8.

### B. Minimum Diameter Order Determination

To reduce message size communicated during massage passing process and improve efficiency, in this section, we propose a more efficient message-passing order such that the constructed DAG has the minimum diameter.

Algorithm 3 shows the message-passing order that has the minimum diameter, which consists of two phases.

- Sink agent determination. 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 produces the shortest diameter $d(a_s)$ will become the sink agent.
- The message-passing order determination. From steps 6-10, once the sink agent $a_s$ is determined, the order can be determined as from edge agents (i.e., these agents that have longer distance) to central agents (i.e., these agents that are close to 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, the message-passing order between is arbitrary, i.e., ties break arbitrary.

The proposed real-time city-scale is based on stard Max-sum_ADVP (i.e., Algorithm 2), in which the message-order is generated by the minimum diameter algorithm MDO (i.e., Algorithm 3).

Derived from the Lemma 1 that the message-passing preducre of Max-sum_ADVP algorithm (i.e., steps 3 and 5) converges within $dia(o)$ iterations. Thus, the

**Lemma 2.** *For any acyclic factor graph, compared to any other message-passing order, the message-passing procedure converges the fastest on the MDO order generated by Algorithm 3.*

The proof is stragithfoward that MDO order has the minimium diameter.

### C. Stability Analysis of TSC

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

**Definition 1.** *(Queue length stability [25]). The queue length stability $Q(t) = \{q(l,m)(t)\}$ is stable in the mean (and $\mathcal{X}$ is stable TSC policy) if for some $K < \infty$*

$$\frac{1}{T}\sum_{t=1}^{T}\sum_{l,h}\mathbb{E}[q(l,h)(t)] < K, \forall T. \tag{9}$$

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

The max-pressure generated from the set of propositions, defined in the following.

**Definition 2.** *Max-pressure control policy [25]. At each period $t$, each agent (intersection) selects the action with maximum pressure at current queue state $Q(t)$: $\tilde{X}^*(t) = \arg\max_{\tilde{X}(t)\in \mathrm{X}(t)} \sum_{(l,m):x(l,m)=1} q(l) - q(m)$, where $q(l) - q(m)$ is the pressure of each movement $(l,m)$.*

Max-pressure policy [25] has proven to be stable. The main step for Max-pressure policy stability is to prove

$$\mathbb{E}|Q(t+1)|^2 - \mathbb{E}|Q(t)|^2 \le K - \epsilon E|Q(t)|. \tag{10}$$

where $E|Q(t)|^2 = \sum_{(l,m)}|q(l,m)|^2$, and and $K < \infty$. For our optimal coordination policy $X^*(t)$ at each period $t$ that minimize the balance at each intersection, we have

$$\mathbb{E}|Q^*(t+1)|^2 \le \mathbb{E}|Q_{\tilde{X}}(t+1)|^2. \tag{11}$$

**Theorem 1.** *The TSC policy selected by our MAC policy $\mathcal{X}^*$ is stablizing the system.*

*Proof.* Derived from the optimal balance of our MAC policy from 11, given the queue state $Q(t) = Q^{MAC}(t) = Q^{MP}(t)$ at the decision point $t$, we have

$$\mathbb{E}|Q^{MAC}(t+1)|^2 + \mathbb{E}|Q^{MAC}(t+1)| \tag{12}$$
$$\le \mathbb{E}|Q^{MP}(t+1)|^2 + \mathbb{E}|Q^{MP}(t+1)|. \tag{13}$$

where $Q^{MAC}(t+1)$ and $Q^{MP}(t+1)$ are the queue state at $t+1$ of the proposed MAC and existing MP approaches respectively. Moreover, given the $Q(t)$, the difference between $Q^{MAC}(t+1)$ and $Q^{MP}(t+1)$ can be bounded. That is

$$|\mathbb{E}|Q^{MAC}(t+1)| - \mathbb{E}|Q^{MP}(t+1)|| \tag{14}$$
$$\le \sum_{l,h}\big[f(l,h) + \sum_{e\in l} f(e,l)\bar{R}(e,l)\big]. \tag{15}$$

| Dataset | # of Intersections | Arrival Rate (cars/s) | Duration (s) |
|---|---|---|---|
| Manhattan_1 | 16×3 | 0.77 | 3600 |
| Manhattan_2 | 28×7 | 2.82 | 3600 |

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

This can be derived that the queue $Q^{MAC}(t+1)$ depends on the saturation flow $f$. Denoted by the constant $\sum_{l,h}\big[f(l,h) + \sum_{e\in l} f(e,l)\bar{R}(e,l)\big]$ as $K_1$, we have

$$\begin{aligned}
&\mathbb{E}|Q^{MAC}(t+1)|^2 - \mathbb{E}|Q^{MAC}(t)|^2 \\
&= \mathbb{E}|Q^{MAC}(t+1)|^2 + \mathbb{E}|Q^{MAC}(t+1)| \\
&\qquad - \mathbb{E}|Q^{MAC}(t)|^2 - \mathbb{E}|Q^{MAC}(t+1)| && (16)\\
&\le \mathbb{E}|Q^{MP}(t+1)|^2 - \mathbb{E}|Q(t)|^2 \\
&\qquad + \mathbb{E}|Q^{MP}(t+1)| - \mathbb{E}|Q^{MAC}(t+1)| && (17)\\
&\le \mathbb{E}|Q^{MP}(t+1)|^2 - \mathbb{E}|Q(t)|^2 + K_1 && (18)\\
&= (K + K_1) - \epsilon E|Q(t)|. && (19)
\end{aligned}$$

Replace the new constant $K = K + K_1$, and summing over $\tau = 1, \cdots, t$ gives

$$\mathbb{E}|Q^{MPC}(t+1)|^2 - \mathbb{E}|Q^{MAC}(1)|^2 \le Kt - \epsilon\sum_{\tau=1}^{t}\mathbb{E}|Q(t)| \tag{20}$$

which can derive that

$$\epsilon\frac{1}{T}\sum_{\tau=1}^{t}\mathbb{E}|Q(t)| \le K + \frac{1}{t}\mathbb{E}|Q(1)|^2. \tag{21}$$

which immediately implies the stability condition. □

## V. Experimental Evaluation

### A. Experiment Setup

We evaluate the proposed TSC method on the Cityflow simulation platform [48] with larger scale and heterogeneous intersections in real-world traffic networks and a synthetic traffic network. 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 otherwise specified, the road network is set to be a four-way intersection.

**Real-World Traffic Network.** We collect two representative traffic flow datasets from to evaluate the performance of our model [13]. the Manhattan city. Detailed statistics of these datasets are listed in Table II. The vehicle arrival rate that pass through road intersections from the governments.

**Synthetic Traffic Network.**

**Comparison Methods.** We compare our EMC methods with the following two categories of methods: transportation methods and RL methods. Note that all methods are state-of-the-art, carefully tuned and their best results are reported.

- FixedTime: Fixed-time with random offset. Each phase has a fixed time of 15 seconds. In this paper, there are

four phases, i.e., WE-Straight, WE-Left, SN-Straight, SN-Left.

- Maxpressure [25]: Max pressure TSC method is the state-of-the-art network-level traffic signal control method, enjoys the state-of-the-art network -level performance, greedily chooses the phase with the maximum pressure, as introduced in Section 4.3.
- PressLight [26]: 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).

**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

**Discussion:** In real dataset, peers have high average reliability accuracy ($\sim$0.82). Even with the limited budget, these high reliable peers can be elicited to be diligent, leading to a high base accuracy. When budget becomes moderate, such incremental budget can only improve limited accuracy due to the submodular property and the high base accuracy. Finally, when the budget becomes so large that it exceeds the critical budget, all peers will be diligent. The remaining budget will be allocated to the assignment that has the largest error rate, thereby improving the increment rate again. Although the fitted Gaussian distribution of peers reliability might not have the same average and variance with the statistical results (e.g., for this real dataset, the fitted Gaussian distribution might be $\mathcal{N}(0.86, 0.14)$), it is practical of modeling peers' reliability to follow a Gaussian distribution.

## VI. 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.

[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. 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.

[21] 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.

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

[23] 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.

[24] 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.

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

[26] 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.

[27] 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.

[28] 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.

[29] 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.

[30] 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.

[31] 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.

[32] 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.

[33] 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.

[34] 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.

[35] 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.

[36] 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.

[37] 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.

[38] 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.

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

[40] 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.

[41] 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.

[42] 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.

[43] 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.

[44] 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.

[45] 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.

[46] 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.

[47] 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.

[48] 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 assistant 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.