Optimal Cooperative Driving at Signal-Free Intersections With Polynomial-Time Complexity

Huaxin Pei, Yuxiao Zhang, Yi Zhang, *Member, IEEE*, and Shuo Feng, *Member, IEEE*

***Abstract*— Cooperative driving at signal-free intersections, which aims to improve driving safety and efficiency for connected and automated vehicles, has attracted increasing interest in recent years. However, existing cooperative driving strategies either suffer from computational complexity or cannot guarantee global optimality. To fill this research gap, this paper proposes an optimal and computationally efficient cooperative driving strategy with the polynomial-time complexity. By modeling the conflict relations among the vehicles, the solution space of the cooperative driving problem is completely represented by a newly designed small-size state space. Then, based on dynamic programming, the globally optimal solution can be searched inside the state space efficiently. It is proved that the proposed strategy can reduce the time complexity of computation from exponential to a small-degree polynomial. Simulation results further demonstrate that the proposed strategy can obtain the globally optimal solution within a limited computation time under various traffic demand settings.**

***Index Terms*— Connected and automated vehicles, cooperative driving, signal-free intersection, dynamic programming.**

I. Introduction

With the help of vehicle-to-everything (V2X) technologies, the emergence of connected and automated vehicles (CAVs) shows a great potential to revolutionize traffic operations around conflict areas [1]–[7]. For example, recent studies have proposed some promising methods to jointly optimize signal timing and vehicle trajectory at signalized conflict areas, which shows a good performance in improving traffic safety and efficiency [8]–[12]. More significantly, benefiting from the fact that CAVs can share the driving states with adjacent vehicles, we can even realize signal-free traffic control at conflict areas by optimizing the cooperative movements of vehicles [13]–[16]. In such cases, vehicles can keep a close safety gap with each other to pass through the conflict area safely and efficiently. Such techniques can be regarded as cooperative driving and have been widely studied in recent years.

It has been pointed out in [17]–[21] that the key problem of cooperative driving is to determine the proper sequence of vehicles to pass the conflict areas (e.g., on-ramp areas and signal-free intersections), so as to fully utilize the limited road resources. The optimization of passing sequence at on-ramps has been well discussed in recent studies [22]–[25]. However, compared with the merging problem, there exist much more complex conflict relations of vehicles at signal-free intersection, which leads to complicated interactions between vehicles. Thus, cooperative driving at intersections will fail by directly applying the strategies of merging problem, and further investigation is required.

Similar to [25], [26], existing studies of planning the passing sequence of vehicles at signal-free intersections can be classified into two categories, i.e., optimization-based and heuristics-based strategies. Optimization-based strategies can find the globally optimal passing sequence but usually suffer from computational complexity [1], [27]–[29]. Heuristics-based strategies can improve computational efficiency but lack a rigorous theoretical guarantee [30]–[35].

To overcome the limitations of existing cooperative driving strategies for signal-free intersections, we propose an optimal and computationally efficient cooperative driving strategy in this paper. By modeling the conflict relations among the vehicles, the solution space of cooperative driving problem is completely represented by a newly designed small-size state space. Then, based on dynamic programming, the globally optimal solution can be searched inside the state space efficiently. It can reduce the time complexity of computation from exponential to a small-degree polynomial.

This paper significantly extends our previous work for the on-ramp merging problem [22], where a dynamic programming method was proposed to obtain the globally optimal merging solution with polynomial-time complexity. However, this method cannot be directly applied to signal-free intersections, because of the following challenges: First, the number of conflicts between vehicles significantly increases, as the

Manuscript received December 1, 2020; revised July 24, 2021; accepted October 1, 2021. This work was supported in part by the National 135 Key Research and Development Program Projects under Grant 2018YFB1600600. The Associate Editor for this article was M. Wang. *(Corresponding author: Shuo Feng.)*

Huaxin Pei is with the Department of Automation, Tsinghua University, Beijing 100084, China (e-mail: phx17@mails.tsinghua.edu.cn).

Yuxiao Zhang is with the Department of Computer Science and Technology, Tsinghua University, Beijing 100084, China (e-mail: zhang-yx17@tsinghua.org.cn).

Yi Zhang is with the BNRist, Department of Automation, Tsinghua University, Beijing 100084, China, also with Tsinghua-Berkeley Shenzhen Institute (TBSI), Shenzhen 518055, China, and also with Jiangsu Province Collaborative Innovation Center of Modern Urban Traffic Technologies, Nanjing 210096, China (e-mail: zhyi@tsinghua.edu.cn).

Shuo Feng is with the Department of Civil and Environmental Engineering, University of Michigan, Ann Arbor, MI 48109 USA (e-mail: fshuo@umich.edu).

Digital Object Identifier 10.1109/TITS.2021.3118592

interactions between vehicles become more complex. More importantly, both the conflict and conflict-free pairs of vehicles exist in the conflict area of an intersection, which is denoted by the heterogeneous conflict relations in this paper, namely, more than one vehicle may pass the conflict area at the same time without collision. In such a case, state transition could not guarantee the Markov property and thus dynamic programming is invalidated [36], [37].

In this paper, we rebuild the dynamic programming model for signal-free intersections. To keep Markov property, a novel state transition strategy is designed to model the conflict-free pairs of vehicles, where multiple non-conflict vehicles can be assigned with the right of way instead of dealing with only one vehicle during a state transition. It guarantees that each state transition explicitly represents one kind of conflicts between vehicles so that the objective value of the current state is only determined by its predecessor state, i.e., the state transition satisfies Markov property. Moreover, the number of states and state transitions of the constructed state space is restricted as a small-degree polynomial with the number of vehicles. Therefore, dynamic programming is reactivated at signal-free intersections, and we can obtain the globally optimal solution with the polynomial-time complexity of computation. It is a significant improvement compared with most existing studies about cooperative driving at signal-free intersections (e.g., [29], [34], [35]), where the size of solution space increases exponentially with the increasing number of vehicles. It is also worth emphasizing that the proposed strategy is generic in most traffic scenarios where both the conflict and conflict-free pairs of vehicles exist.

Theoretical analysis is proposed to justify the computational time complexity of the proposed optimal strategy. It is proved that the optimal strategy has the polynomial-time complexity of computation, which has a lower bound $O(N^4)$ and an upper bound $O(N^6)$, where N denotes the total number of vehicles within consideration. It is the theoretical foundation for overcoming the limitations of existing strategies. Furthermore, simulation results also demonstrate that the proposed strategy can obtain the globally optimal solution within a limited computation time, comparing with the existing strategies under various traffic demand settings.

The main contributions of this paper include: (a) we construct a novel state space with a much smaller size than that in the existing studies to describe the complete solution space of cooperative driving at signal-free intersections; (b) we realize optimal cooperative driving at signal-free intersections, which overcomes the limitations of the existing strategies regarding global optimality and computational efficiency; (c) we give a rigorous theoretical analysis about the computational complexity of the proposed strategy, which is usually lacking in most existing studies.

The rest of this paper is organized as follows. *Section II* reviews the related works about determining sequence. *Section III* presents the typical signal-free intersection and formulates the cooperative driving problem. *Section IV* proposes the optimal strategy for cooperative driving at intersections. Then, *Section V* gives theoretical analysis about the computational complexity, and simulation results are in *Section VI*. Finally, the conclusion and further works are presented in *Section VII*.

II. Related Work

In this section, we briefly review the related works on determining passing sequence of vehicles at signal-free intersections. As aforementioned, two types of approaches have been proposed to solve the problem of planning passing sequence, i.e., optimization-based and heuristics-based strategies.

Optimization-based strategy aims to enumerate all possible passing sequences of vehicles to find the globally optimal solution. One typical type of the optimization-based strategies is to formulate a large-scale mixed-integer programming (MIP) problem to resolve all conflicts between vehicles, where each conflict introduces a binary variable to mathematically describe the relative passing sequence [1], [27], [28]. However, the computational complexity increases exponentially with the increasing number of conflicts between vehicles, which causes the "curse of dimensionality". Another type of optimization-based strategies is adopting a string to represent the vehicle sequence and then describe the complete solution space of the optimization problem. For example, Li *et al.* [29] formulated a spanning tree and proposed a pruning rule to search for the globally optimal solution. However, due to the complex conflict relations between vehicles, it still suffers from the state space explosion and thus is intractable for real-time applications.

Heuristics-based strategy usually stops at a local optimal solution within a limited computation time, aiming to improve computational efficiency. Autonomous intersection management (AIM) [30], [31] and reservation-based strategy [32], [33] are the typical heuristics-based strategies utilized in the problem of cooperative driving, where heuristic rules are used to instruct the vehicles passing through the conflict area roughly in first-in-first-out (FIFO) manner [22]. Nevertheless, as illustrated in [19], [20], these strategies have a poor performance in improving traffic efficiency. In [38], an auction-based AIM was proposed to determine the passing sequence of vehicles. Simulation results demonstrate that this strategy can outperform the FIFO strategy in most cases, but it theoretically leads to a local optimal solution and thus cannot guarantee traffic efficiency. Grouping vehicles in the same lane into platoons is an effective way to reduce the size of the solution space and then improve computational efficiency [23], [39]–[41]. However, since vehicles in the same group may have different driving intentions (i.e., going straight; turning left/right) at an intersection scenario, how to select the vehicles that can be grouped remains a new challenge. Recently, several promising strategies are proposed to keep a good balance between computational complexity and traffic efficiency. In [34], [35], Monte Carlo tree search (MCTS) method combining with heuristic rules is first introduced to the planning problem of vehicle sequence at a signal-free intersection. Numerical simulation results indicate that these strategies can obtain a near-optimal passing sequence within a limited computation time. However, the performance of these strategies lacks a rigorous theoretical guarantee and further analysis is necessary for practical applications.

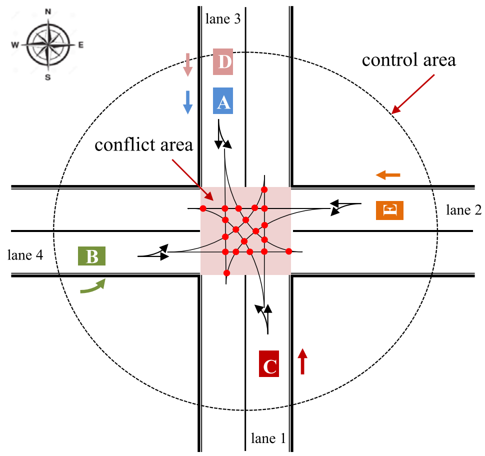

Fig. 1. A typical signal-free intersection scenario with five vehicles.

III. Problem Formulation

A. Scenario and Notations

In this paper, we select a typical intersection scenario to introduce our method, as shown in Fig. 1. The red area is the conflict area, where the vehicles from different directions may collide. The intersection has a control area. The distance from the entry point of the control area to the entry point of the conflict area, denoted as l_c, is assumed the same for all entry directions. The vehicles within the control area can share the driving states with each other and move automatically according to the driving plans determined by the cooperative driving strategies. As traffic collisions and delays mainly occur on the vehicles with left-turn and straight movements, the right-turn vehicles can usually move freely around the intersection. To simplify the descriptions, we only consider the left-turn and straight vehicles in this paper, and the scenario including right-turn vehicles can be tackled in a similar way.

Several reasonable assumptions about the studied scenario are added as follows: a) all vehicles are connected and automated (CAVs); b) lane-change behavior of vehicles within the control area is not allowed; c) vehicles move at a constant velocity v_c after entering the conflict area. Each vehicle is given a unique identity after arriving at the control area, and CAV_i means the i^{th} vehicle that reaches the control area. The vehicle identity sets of lane 1, 2, 3 and 4 are denoted by $\mathbb{N}_1 = \{1, \ldots, \hat{n}_1\}$, $\mathbb{N}_2 = \{1, \ldots, \hat{n}_2\}$, $\mathbb{N}_3 = \{1, \ldots, \hat{n}_3\}$ and $\mathbb{N}_4 = \{1, \ldots, \hat{n}_4\}$ respectively, where $\hat{n}_1$, $\hat{n}_2$, $\hat{n}_3$ and $\hat{n}_4$ denote the total number of vehicles on lane 1, lane 2, lane 3 and lane 4 respectively. The main notations in this paper are shown in TABLE I.

B. Optimization Problem

As illustrated in [20], [29], cooperative driving aims to improve traffic efficiency and safety by planning the sequence of vehicles passing through the conflict area. To reach this goal, we formulate the objective function as

$$\min_{\mathbb{T}^{\text{assign}}} J, \tag{1a}$$

$$J = \max_{i \in \mathbb{N}_1 \cup \mathbb{N}_2 \cup \mathbb{N}_3 \cup \mathbb{N}_4} t_{\text{assign},i}, \tag{1b}$$

$$t_{\text{assign},i} \in \mathbb{T}^{\text{assign}}, \tag{1c}$$

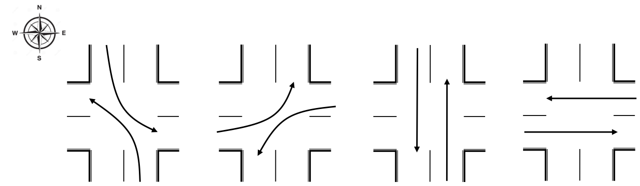

Fig. 2. Four kinds of collision-free pairs of vehicles at an intersection.

where the decision variable $t_{\text{assign},i}$ denotes the arrival time assigned to CAV_i to move into the conflict area, and $\mathbb{T}^{\text{assign}}$ is the set of the arrival time assigned for all vehicles within the control area. Therefore, J denotes the total passing time of all vehicles to pass through the conflict area [22], [42].

The arrival time assigned to each vehicle needs to satisfy vehicle dynamics, i.e.,

$$t_{\text{assign},i} \geq t_{\text{min},i}, \tag{2}$$

where $t_{min,i}$ denotes the minimal arrival time assigned to vehicle i, which can be easily calculated according to vehicle dynamics [22], [34].

As for the vehicles moving on the same lane, the arrival time assigned to each vehicle should satisfy the physical constraints to avoid rear-end collision, i.e.,

$$t_{\text{assign},i} - t_{\text{assign},j} \geq \Delta_{t,1}, \tag{3}$$

where $\Delta_{t,1}$ denotes the minimal safe gap between two consecutive vehicles to avoid rear-end collision. Here, CAV_j is physically ahead of CAV_i, e.g., CAV_A and CAV_D in Fig. 1.

The vehicles from different lanes may have collisions in the conflict area. The method to avoid these collisions is to schedule the vehicles moving into the conflict area sequentially. Thus, the binary variables are introduced to describe the constraints of vehicle sequence, i.e.,

$$t_{\text{assign},i} - t_{\text{assign},j} + \mathbf{M} \cdot b_{i,j} \geq \Delta_{t,2}, \tag{4}$$

$$t_{\text{assign},j} - t_{\text{assign},i} + \mathbf{M} \cdot (1 - b_{i,j}) \geq \Delta_{t,2}, \tag{5}$$

where $\Delta_{t,2}$ denotes the minimal safe gap to avoid collisions between vehicles from different lanes. CAV_i and CAV_j are two vehicles from different lanes with conflicts, e.g., CAV_A and CAV_B in Fig. 1. $\mathbf{M}$ is a positive and sufficiently large number. $b_{i,j}$ is the binary variable and we use $\mathbb{B}$ to denote the set of all binary variables. Here, $b_{i,j} = 1$ implies that vehicle i enters conflict area earlier than vehicle j.

Based on the above descriptions, the planning problem of vehicle sequence can be mathematically formulated as a mixed-integer programming problem (MIP), as shown in (6).

$$\begin{aligned} &\min_{\mathbb{T}^{\text{assign}}, \mathbb{B}} \; J \\ &s.t. \; (2)(3)(4)(5) \end{aligned} \tag{6}$$

C. Problem Challenges

Compared with the on-ramp merging problem that has been widely investigated, there are two major challenges of solving the cooperative driving problem at signal-free intersections:

TABLE I
THE NOMENCLATURE LIST

Variables	Notations
CAV_i	The i^{th} vehicle that reaches the control area.
$\hat{n}_1$, $\hat{n}_2$, $\hat{n}_3$, $\hat{n}_4$	The total number of vehicles on lane 1, lane 2, lane 3 and lane 4, respectively.
$v_{\max}$, $v_{\min}$	The maximal and minimal velocity of vehicles.
$a_{\max}$, $a_{\min}$	The maximal and minimal acceleration of vehicles.
$t_{\text{assign},i}$	The arrival time assigned to CAV_i to move into the control area.
$b_{i,j}$	The binary decision variable used to formulate the collision avoidance constrains between CAV_i and CAV_j.
$\Delta_{t,1}$, $\Delta_{t,2}$	The minimal allowable safe gaps used in constraints (3)-(5).
l_c	The length of the control area.
$\mathbb{N}_1$, $\mathbb{N}_2$, $\mathbb{N}_3$, $\mathbb{N}_4$	The vehicles identity sets of lane 1, lane 2, lane 3 and lane 4, respectively.
$\mathbb{T}^{\text{assign}}$	The set of the arrival time assigned to all vehicles.
$\mathbb{B}$	The binary decision variable set.
$s_r\begin{pmatrix} n_1 & n_2 \\ n_3 & n_4 \end{pmatrix}$	n_1: The accumulated number of vehicles added to the current candidate sequence on lane 1 up to current state. n_2: The accumulated number of vehicles added to the current candidate sequence on lane 2 up to current state. p_3: The accumulated number of vehicles added to the current candidate sequence on lane 3 up to current state. n_4: The accumulated number of vehicles added to the current candidate sequence on lane 4 up to current state. r: The lane number that has been assigned with the right of way at current state.

1) Increased Number of Conflicts: the number of conflicts between vehicles has increased a lot. As shown in Fig. 1, each red dot represents one kind of conflict pairs, and each conflict between two vehicles will introduce a binary variable in the problem (6). Thus, the size of solution space increases exponentially with the increasing number of conflicts [1], [25], [43], [44].

2) Heterogeneous Conflict Relations: besides the conflict pairs of vehicles, there also exist several collision-free pairs of vehicles shown in Fig. 2, where the corresponding vehicles are allowed to move into the conflict area at the same time. The heterogeneous conflict relations between vehicles further increase the number of feasible solutions and lead to state-space explosion for the state-based methods, e.g., [29].

To this end, it is necessary to construct a new cooperative driving strategy to overcome the above challenges and then guarantee computational efficiency and global optimality.

IV. Optimal Cooperative Driving Design

In this section, we will reformulate problem (6) based on dynamic programming to implement optimal cooperative driving with polynomial computational complexity. The key idea is to construct a small-size state space to describe the complete solution space of the large-scale planning problem (6) and then search for the globally optimal solution by dynamic programming. An intersection scenario with five vehicles shown in Fig. 3(a) is taken as an example to introduce our method.

The rest of *Section IV* is organized as follows. The state space describing the complete solution space is constructed in *Section IV-A*, *Section IV-B* and *Section IV-C*. *Section IV-D* introduces the method to search the globally optimal solution inside the constructed state space. In *Section IV-E*, state space construction process and solution searching method are integrated into one overall optimal cooperative driving algorithm to save the computational resources.

A. State Definition

The problem of planning vehicle sequence passing through the conflict area is equivalent to sequentially assign the right of way for each vehicle to move into the conflict area. Thus, the problem (6) can be reformulated as a multi-stage decision process, where the vehicles are sequentially added to the current candidate sequence and only one vehicle is tackled in each stage. Then, as shown in Fig. 3(b), state variable is defined to describe the assignment of the right of way, i.e.,

$$s_r\begin{pmatrix} n_1 & n_2 \\ n_3 & n_4 \end{pmatrix}, \tag{7}$$

where n_1, n_2, n_3 and n_4 denote the accumulated number of vehicles added to current candidate sequence in lane 1, 2, 3 and 4 up to current state, respectively. r denotes the lane identity of the vehicle which obtains the right of way at current state. For instance, $s_3\left(\begin{smallmatrix} 1 & 0 \\ 2 & 0 \end{smallmatrix}\right)$ denotes that one vehicle in lane 1 and two vehicles in lane 3 have been tackled up to current state. In addition, the vehicle with the right of way at current state can be exactly specified. For instance, $s_3\left(\begin{smallmatrix} 1 & 0 \\ 2 & 0 \end{smallmatrix}\right)$ denotes that the second vehicle in lane 3 ($r = 3$, $n_r = 2$) has the right of way at the current state, i.e., CAV_D.

B. Basic State Transition

State transition is used to describe the transition of the right of way between vehicles. We use the lane identity as the decision variable u ($u \in \{1, 2, 3, 4\}$) to determine which lane has obtained the right of way at the current stage so that one vehicle on the corresponding lane is added to the candidate sequence. The state transition function emerges after introducing the state variable and decision variable, i.e.,

$$s_r\left(\begin{smallmatrix} n_1 & n_2 \\ n_3 & n_4 \end{smallmatrix}\right) = g\left(s_{r'}\begin{pmatrix} n'_1 & n'_2 \\ n'_3 & n'_4 \end{pmatrix}, u\right), \tag{8}$$

where $s_{r'}\begin{pmatrix} n'_1 & n'_2 \\ n'_3 & n'_4 \end{pmatrix}$ is the predecessor state of $s_r\left(\begin{smallmatrix} n_1 & n_2 \\ n_3 & n_4 \end{smallmatrix}\right)$. The function $g(\cdot)$ denotes the state transition function, i.e., if $u = i$,

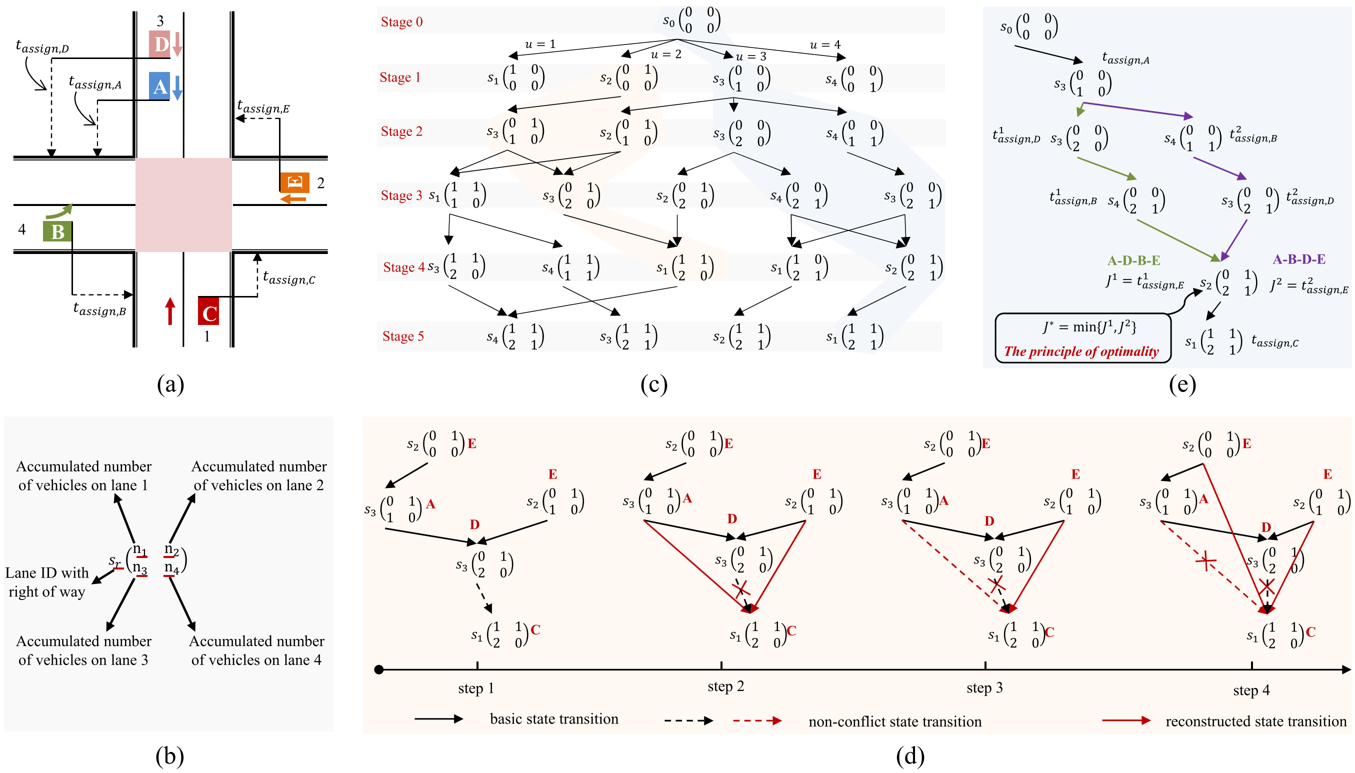

Fig. 3. Optimal cooperative driving strategy applied to a simple signal-free intersection scenario with five vehicles. Fig. 3(a) presents the studied scenario. Fig. 3(b) gives the definition of state variable. Fig. 3(c) presents a part of state space under basic state transition, where many states and state transitions are omitted for simplicity. Fig. 3(d) illustrates how to reconstruct the state transitions. Fig. 3(e) illustrates the decision-making process.

we have $n_i = n_i' + 1$, $r = i$ and other parameters of the state variable stay the same. For example, as shown in Fig. 3(c), the initial state $s_0\left(\begin{smallmatrix}0&0\\0&0\end{smallmatrix}\right)$ connects to different successor states according to different values of the decision variable.

A new vehicle will be assigned an arrival time to the conflict area if it obtains the right of way during a state transition. Thus, the objective value J varies with state transition. As for the conflict pairs of vehicles, the arrival time assigned to the vehicle specified by current state is always larger than that assigned to the vehicle specified by its predecessor state, because the vehicle specified by the predecessor state has the priority over the vehicle specified by current state to enter the conflict area during each state transition. Therefore, the objective value J up to current state is exactly equal to the time assigned to the vehicle specified by current state, i.e.,

$$J(s) = t_{\text{assign},s}, \tag{9}$$

where $J(s)$ denotes the objective value J up to state s, and $t_{\text{assign},s}$ denotes the arrival time assigned to the vehicle specified by state s. Then, considering constraints (2)-(5), the recurrence function of objective value J during state transition can be summarized as

$$\begin{aligned} J(s) &= t_{\text{assign},s} \\ &= \max(t_{\min,s}, t_{\text{assign},s'} + \Delta_T) \\ &= \max(t_{\min,s}, J\left(s'\right) + \Delta_T), \end{aligned} \tag{10}$$

where $t_{\min,s}$ denotes the minimal arrival time constrained by vehicle dynamics of the vehicle specified by state s, and s' is the predecessor state of s. If the vehicles specified by s and s' are in the same lane, $\Delta_T = \Delta_{t,1}$, otherwise, $\Delta_T = \Delta_{t,2}$.

According to Eq. (10), we can find that the objective value $J(s)$ is directly determined by its predecessor state s' and has nothing with other states before the predecessor state. In other words, state transition satisfies Markov property when there exist conflicts between the vehicles specified by the corresponding two states, which is a necessary requirement of using dynamic programming to solve a multi-stage decision problem [36], [37].

However, as for the conflict-free pairs of vehicles presented in *Section III-C*, the objective value J up to current state has nothing with that of its predecessor state, because they are not conflict with each other. As shown in Fig. 3(d), CAV_C specified by $s_1\left(\begin{smallmatrix}1&1\\2&0\end{smallmatrix}\right)$ and CAV_D specified by $s_3\left(\begin{smallmatrix}0&1\\2&0\end{smallmatrix}\right)$ are the so-called conflict-free pairs so that the arrival time assigned to CAV_C is determined by that assigned to the vehicles other than CAV_D. In other words, $J\left(s_1\left(\begin{smallmatrix}1&1\\2&0\end{smallmatrix}\right)\right)$ is not determined by its predecessor state $s_3\left(\begin{smallmatrix}0&1\\2&0\end{smallmatrix}\right)$, but by the states before $s_3\left(\begin{smallmatrix}0&1\\2&0\end{smallmatrix}\right)$. Therefore, the state transition cannot satisfy Markov property and thus invalidates dynamic programming in this problem. To this end, it is necessary to reconstruct state transition function to overcome the challenges from heterogeneous conflict relations.

C. State Transition Reconstruction

In this subsection, we will reconstruct the state transition function to keep Markov property then reactivate dynamic programming for cooperative driving at signal-free intersections.

To keep Markov property, the non-conflict transitions (i.e., the transitions which connect the states that are not in conflict with each other) should be removed from the state space. To guarantee global optimality, we need to build new transitions for the corresponding states to completely describe the assignment of the right of way. Firstly, we need to find the non-conflict transitions in state space and then, for each non-conflict state transition, we need to remove it and connect current state with all predecessor states of its original predecessor state. Repeat the above steps until there is no non-conflict transition in state space. The reconstruction strategy guarantees that each state transition represents one kind of conflict between vehicles and all possible assignments of the right of way are presented in state space.

Here we illustrate our key idea in Fig. 3(d) and will elaborate the implementation algorithm in *Section IV-E*. Firstly, we find that the transition between state $s_3\begin{pmatrix}0&1\\2&0\end{pmatrix}$ and $s_1\begin{pmatrix}1&1\\2&0\end{pmatrix}$ is the so-called non-conflict transition as CAV_D and CAV_C can enter the conflict area at the same time, and thus we need to remove this transition and establish new transitions for $s_1\begin{pmatrix}1&1\\2&0\end{pmatrix}$. Secondly, the non-conflict transition between $s_3\begin{pmatrix}0&1\\2&0\end{pmatrix}$ and $s_1\begin{pmatrix}1&1\\2&0\end{pmatrix}$ is removed, and $s_1\begin{pmatrix}1&1\\2&0\end{pmatrix}$ connects to the predecessor states of $s_3\begin{pmatrix}0&1\\2&0\end{pmatrix}$, i.e., $s_3\begin{pmatrix}0&1\\1&0\end{pmatrix}$ and $s_2\begin{pmatrix}0&1\\1&0\end{pmatrix}$. Thirdly, we find that one of the reproduced transitions is still the non-conflict transition, i.e., the transition between $s_3\begin{pmatrix}0&1\\1&0\end{pmatrix}$ and $s_1\begin{pmatrix}1&1\\2&0\end{pmatrix}$. Finally, the non-conflict transition between $s_3\begin{pmatrix}0&1\\1&0\end{pmatrix}$ and $s_1\begin{pmatrix}1&1\\2&0\end{pmatrix}$ is removed, and $s_1\begin{pmatrix}1&1\\2&0\end{pmatrix}$ connects to the predecessor state of $s_3\begin{pmatrix}0&1\\1&0\end{pmatrix}$, i.e., $s_2\begin{pmatrix}0&1\\0&0\end{pmatrix}$. Therefore, the non-conflict transition between $s_3\begin{pmatrix}0&1\\2&0\end{pmatrix}$ and $s_1\begin{pmatrix}1&1\\2&0\end{pmatrix}$ is modified to guarantee Markov property and global optimality. Compared with the basic state transition in *Section IV-B*, the reconstruction strategy has the following new properties.

Property 1 (The State Transition is Not Just Between the Adjacent Stages): Thus, one state transition does not necessarily represent only one vehicle that obtains the right of way while it may represent a group of vehicles in which there is no conflict of different directions. As shown in Fig. 3(d) step 4, the transition between $s_2\begin{pmatrix}0&1\\0&0\end{pmatrix}$ and $s_1\begin{pmatrix}1&1\\2&0\end{pmatrix}$ represents that CAV_A, CAV_D and CAV_C are assigned with the right of way during current state transition.

Property 2 (Removing One State Transition May Reproduce Multiple Transitions): It increases the number of transitions in the reconstructed state space, and theoretical analysis about this change will be presented in Section V.

By Property 1, the recurrence function of objective value J needs to be modified based on the reconstructed transitions. The objective value J up to current state should consider the assigned arrival time of a group of vehicles specified by the corresponding transition rather than only one vehicle. We assume that state s' is the predecessor state of s and use $\mathbb{G}$ to denote the set of vehicle identity of a group of vehicles that are assigned the right of way during the transition from s' to s. For each vehicle in $\mathbb{G}$, the method to assign arrival time is

$$t_{\text{assign},k} = \max(t_{\min,k}, J(s') + \Delta_T, t_{\text{assign},k'} + \Delta_{t,1}), \quad (11a)$$
$$k, k' \in \mathbb{G}, \quad (11b)$$

where $CAV_{k'}$ is physically ahead of CAV_k in the same lane. If the vehicles specified by s and s' are in the same lane, $\Delta_T = \Delta_{t,1}$, otherwise, $\Delta_T = \Delta_{t,2}$. Then, we can obtain the objective value J up to current state, i.e.,

$$J(s) = \max_{k\in\mathbb{G}} t_{\text{assign},k}. \quad (12)$$

Based on (11a), (11b) and (12), we can find that $J(s)$ is determined by its predecessor state s' and has nothing with other states before state s', i.e., the reconstructed state transition guarantee Markov property. For example, as shown in Fig. 3(d), the state transition between $s_2\begin{pmatrix}0&1\\0&0\end{pmatrix}$ and state $s_1\begin{pmatrix}1&1\\2&0\end{pmatrix}$ represents that CAV_A, CAV_D and CAV_C are assigned with the arrival time at this time, i.e.,

$$t_{\text{assign},A} = \max\left(t_{\min,A}, J\left(s_2\begin{pmatrix}0&1\\0&0\end{pmatrix}\right) + \Delta_{t,2}\right), \quad (13)$$
$$t_{\text{assign},D} = \max(t_{\min,D}, t_{\text{assign},A} + \Delta_{t,1}), \quad (14)$$
$$t_{\text{assign},C} = \max\left(t_{\min,C}, J\left(s_2\begin{pmatrix}0&1\\0&0\end{pmatrix}\right) + \Delta_{t,2}\right), \quad (15)$$
$$J\left(s_1\begin{pmatrix}1&1\\2&0\end{pmatrix}\right) = \max(t_{\text{assign},A}, t_{\text{assign},D}, t_{\text{assign},C}). \quad (16)$$

According to (13)-(16), it can be found that $J\left(s_1\begin{pmatrix}1&1\\2&0\end{pmatrix}\right)$ is determined by $J\left(s_2\begin{pmatrix}0&1\\0&0\end{pmatrix}\right)$ and has nothing with other states before $s_2\begin{pmatrix}0&1\\0&0\end{pmatrix}$.

Based on the aforementioned descriptions, the constructed state space has following properties.

Property 3: There are many phenomena where different vehicle sequences reach the same state in the state space by determining the accumulated assigned number of vehicles in the specific lane, which leads to that the size of the state space is much smaller compared with those methods in which the vehicle sequence is directly used as the state [29], [34].

Property 4: By determining the accumulated assigned number of vehicles in the specific lane, vehicles in the same lane always comply with the physical constraints (3) during each state transition, which leads to that the infeasible sequences that violate the physical constraints are directly eliminated and the state space contains only all feasible solutions.

Property 3 and Property 4 make it possible to describe the complete solution space of the large-scale planning problem (6) utilizing a small-size state space. Theoretical analysis about the size of the state space will be presented in *Section V*.

D. Decision-Making Process

In this subsection, we will introduce the method to search the globally optimal solution in the constructed state space.

As described in *Section IV-C*, state transition satisfy Markov property in the state space. Thus, *the principle of optimality* can be adopted to search the globally optimal solution.

Lemma 1 (The Principle of Optimality): if the optimal path of states passes through a particular state S in the state space, the partial path from the initial state to state S must also be the optimal path from the initial state to state S [36].

According to Eq. (11a)-(12) and Fig. 3(e), the objective value is updated and the corresponding vehicles can get the assigned arrival time during each state transition. If there is only one predecessor state of current state, the objective

Algorithm 1: Optimal Cooperative Driving Algorithm

Input: Driving states of all vehicles within control area.
Output: Optimal arrival time assigned to each vehicle.

for *each* $i \in [1, N]$ **do**
  for *each state* $s_1\binom{n_1\ n_2}{n_3\ n_4}$ *in stage* i **do**
    /* *Determine Candidate Vehicles.* */
    $N_1 = 1,\ N_3 = 0$.
    $j = n_1$.
    while $V_{1,j-1}$ *has the same intention as* V_{1,n_1} **do**
      $j = j - 1,\ N_1 = N_1 + 1$.
    end
    $j = n_3$.
    while $V_{3,j}$ *has the same intention as* V_{1,n_1} **do**
      $j = j - 1,\ N_3 = N_3 + 1$.
    end
    /* *Five Kinds of Predecessor States.* */
    Connect $s_1\binom{n_1\ n_2}{n_3\ n_4}$ with $s_1\binom{n_1-1\ n_2}{n_3\ \ n_4}$.
    for *each* $\overline{n}_3 \in [1, N_3]$ **do**
      Connect $s_1\binom{n_1\ n_2}{n_3\ n_4}$ with $s_1\binom{n_1-N_1\ n_2}{n_3-\overline{n}_3\ n_4}$.
    end
    for *each* $\overline{n}_1 \in [1, N_1]$ **do**
      Connect $s_1\binom{n_1\ n_2}{n_3\ n_4}$ with $s_3\binom{n_1-\overline{n}_1\ n_2}{n_3-N_3\ n_4}$.
    end
    for *each* $\overline{n}_1 \in [1, N_1]$ *and each* $\overline{n}_3 \in [0, N_3]$ **do**
      Connect $s_1\binom{n_1\ n_2}{n_3\ n_4}$ with $s_2\binom{n_1-\overline{n}_1\ n_2}{n_3-\overline{n}_3\ n_4}$.
    end
    for *each* $\overline{n}_1 \in [1, N_1]$ *and each* $\overline{n}_3 \in [0, N_3]$ **do**
      Connect $s_1\binom{n_1\ n_2}{n_3\ n_4}$ with $s_4\binom{n_1-\overline{n}_1\ n_2}{n_3-\overline{n}_3\ n_4}$.
    end
    Update the objective value for each transition.
    Record the optimal predecessor state.
  end
  Other states can be tackled in similar way if $r \neq 1$.
end
Obtain the optimal passing sequence via backtracking.
Assign the optimal arrival time for each vehicle.

value of current state is determined by that of the unique predecessor state. However, if there are multiple predecessor states, we need to find the optimal objective value of current state according to Lemma 1, i.e.,

$$J^* = \min_{i \in \mathbb{S}'} J^i, \tag{17}$$

where $\mathbb{S}'$ denotes the set of all predecessor states of current state. J^i denotes the candidate objective value of current state determined by its i^{th} predecessor state, and J^* is the optimal objective value of current state. According to Eq. (17), we can get the optimal objective value and record the optimal predecessor state of each state.

Based on the above descriptions, it can be easily concluded that the minimal objective value among all terminal states in the last stage is the globally optimal objective value of the problem (6). Then, the optimal sequence can be obtained via a backtracking search from the optimal terminal state according to the recorded optimal predecessor state of each state.

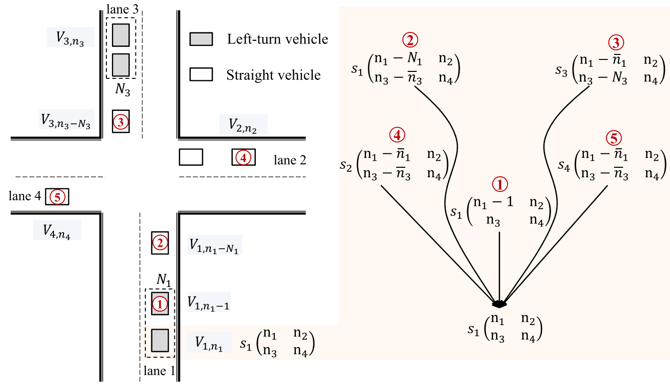

Fig. 4. An illustration of how to find all predecessor states for each state in Algorithm 1. There are five kinds of predecessor states.

E. Algorithm for Implementation

In the above subsections, we introduce our method in three steps for easier understanding, i.e., basic state transition, state transition reconstruction and decision-making process. For implementation, as commonly used in dynamic programming, these three steps can be integrated into one algorithm to save computational resources. Specifically, instead of constructing the state space under basic state transition and then reconstructing the non-conflict transitions, we can directly construct the feasible state transitions by combining the basic state transition function and reconstruction strategy. At the same time, the currently optimal objective value of each state can be updated during each state transition. It avoids repeated computation, and the integrated algorithm is shown in Algorithm 1, where $V_{i,j}$ denotes the j^{th} vehicle in lane i. N denotes the total number of vehicles within the control area. The driving intention refers to whether the vehicle is going straight or turning left.

To obtain the globally optimal solution, for each state $s_r\binom{n_1\ n_2}{n_3\ n_4}$ in the state space, we need to find all possible predecessor states and then obtain the optimal $J\left(s_r\binom{n_1\ n_2}{n_3\ n_4}\right)$ and the optimal predecessor state of $s_r\binom{n_1\ n_2}{n_3\ n_4}$, as described in Algorithm 1. To reach this goal, we firstly need to determine the candidate vehicles that may obtain the right of way during the state transitions connecting to state $s_r\binom{n_1\ n_2}{n_3\ n_4}$ according to the conflict relations between vehicles. Then, all the predecessor states of $s_r\binom{n_1\ n_2}{n_3\ n_4}$ can be divided into five categories according to the lane identity of the vehicle specified by each predecessor state. At the same time, the currently optimal $J\left(s_r\binom{n_1\ n_2}{n_3\ n_4}\right)$ is updated during each state transition, and we can obtain the optimal predecessor state of $s_r\binom{n_1\ n_2}{n_3\ n_4}$. Finally, all the states in the state space can be tackled in the similar way and thus the optimal passing sequence can be exported through backtracking search.

Taking Fig. 4 as the example to illustrate how to find all possible predecessor states of state $s_1\binom{n_1\ n_2}{n_3\ n_4}$. In this case, there are at most N_1 vehicles in lane 1 and N_3 vehicles in lane 3 that may obtain the right of way during the state transitions connecting to $s_1\begin{pmatrix} n_1 & n_2 \\ n_3 & n_4 \end{pmatrix}$, because there is no conflict of different directions among these vehicles. Then, there are five kinds of predecessor states of $s_1\begin{pmatrix} n_1 & n_2 \\ n_3 & n_4 \end{pmatrix}$: ① if the vehicle specified by the predecessor state is in lane 1 and only one vehicle in lane 1 can obtain the right of way, the predecessor state is $s_1\begin{pmatrix} n_1-1 & n_2 \\ n_3 & n_4 \end{pmatrix}$. ② if the vehicle specified by the predecessor state is in lane 1 and both the vehicles in lane 1 and lane 3 can obtain the right of way, the predecessor states can be described as $s_1\begin{pmatrix} n_1-N_1 & n_2 \\ n_3-\overline{n}_3 & n_4 \end{pmatrix}$, where $\overline{n}_3 \in [1, N_3]$. ③ if the vehicle specified by the predecessor state is in lane 3, the predecessor states can be described as $s_3\begin{pmatrix} n_1-\overline{n}_1 & n_2 \\ n_3-N_3 & n_4 \end{pmatrix}$, where $\overline{n}_1 \in [1, N_1]$. ④ if the vehicle specified by the predecessor state is in lane 2, the predecessor states can be described as $s_2\begin{pmatrix} n_1-\overline{n}_1 & n_2 \\ n_3-\overline{n}_3 & n_4 \end{pmatrix}$, where $\overline{n}_1 \in [1, N_1]$ and $\overline{n}_3 \in [0, N_3]$. ⑤ if the vehicle specified by the predecessor state is in lane 4, the predecessor states can be described as $s_4\begin{pmatrix} n_1-\overline{n}_1 & n_2 \\ n_3-\overline{n}_3 & n_4 \end{pmatrix}$, where $\overline{n}_1 \in [1, N_1]$ and $\overline{n}_3 \in [0, N_3]$.

V. Analysis of Computational Complexity

In this section, both the size of the constructed state space and the computational complexity of the proposed strategy are proved through theoretical analysis.

A. Analysis of Number of States

The theorem is proposed for the number of states.

Theorem 1: The total number of states is $O(N^4)$, where N denotes the total number of vehicles within control area.

Proof: See *Appendix I*. ■

By Theorem 1, the number of states is a quartic polynomial with the number of vehicles. Thus, the size of state space grows slowly with the increasing number of vehicles.

B. Analysis of Number of Transitions

By Property 2 in *Section IV-C*, it can be concluded that the number of transitions varies with the conflict relations between vehicles within the conflict area. Thus, there exist the lower and upper bounds on the number of state transitions: if all the vehicles within the control area are in conflict with each other, the number of transitions reaches the lower bound, which is equal to that of the state space constructed under basic state transition; if each vehicle within the control area does not conflict with any vehicle on the opposite lane (e.g., all vehicles within the control area are with straight movements.), the number of transitions reaches to the upper bound, because each state has the largest number of predecessor states in such case. Thus, two theorems are proposed for the number of transitions.

Theorem 2: The lower bound on the number of transitions is $O(N^4)$, where N denotes the total number of vehicles within control area.

Proof: See *Appendix II-A*. ■

Theorem 3: The upper bound on the number of transitions is $O(N^6)$, where N denotes the total number of vehicles within control area.

Proof: See *Appendix II-B*. ■

By Theorem 2-3, both the lower and upper bounds on the number of state transitions are polynomial with the number of vehicles within the control area.

C. Analysis of Computational Time Complexity

The theorem is proposed for computational complexity.

Theorem 4: The proposed strategy has the polynomial-time complexity of computation, which has a lower bound $O(N^4)$ and an upper bound $O(N^6)$, where N denotes the total number of vehicles within control area.

Proof: See *Appendix III*. ■

By Theorem 4, the proposed optimal cooperative driving strategy can reduce the time complexity of computation from exponential to a small-degree polynomial.

Remark 1: In Appendix III, we get that each state transition will produce one computation with constant time. Then, it can be concluded that the time complexity of the proposed strategy is close to the lower bound in most cases, since there are many conflicts between vehicles in most cases and it is hard to satisfy the condition that each vehicle within the control area does not conflict with any vehicle on the opposite lane.

VI. Simulation Results

A. Simulation Settings

We design two kinds of simulations to verify the performance of the proposed optimal strategy. The first simulation aims to validate the global optimality of the proposed strategy. The second simulation aims to further demonstrate the performance of the proposed strategy in a continuous traffic process.

In the simulations, we select the intersection shown in Fig. 1 as the studied scenario, where the length of control area l_c is set as $250m$. For the input lane of each direction, we assume that half of the vehicles will turn left and half will go straight. As suggested in [22], [26], the minimal safe gaps $\Delta_{t,1}$ and $\Delta_{t,2}$ to avoid collisions in conflict areas are set as $1.5s$ and $2s$, respectively. The vehicle dynamics parameters $a_{\max}$, $a_{\min}$, $v_{\max}$, $v_{\min}$, and v_c are set as $3m/s^2$, $-5m/s^2$, $15m/s$, $0m/s$, and $10m/s$, respectively. In addition, all simulations are carried out on the Visual Studio platform in a personal computer with an i7 CPU and a 16 GB RAM.

There are three kinds of performance indices evaluated in the simulations: ① *total passing time:* it refers to the total time when all vehicles within the control area have passed the conflict area. In fact, the total passing time is exactly equal to the objective value of problem (6) and can be calculated by (1b). ② *traffic throughput:* it refers to the total number of vehicles that have passed the conflict area in a specified period of time, which is utilized to evaluate the traffic efficiency in a continuous traffic process. ③ *average computation time:* it refers to the average computation time that is taken in a one-time planning procedure to get the vehicle sequence, which is used to evaluate the computational efficiency of strategies.

There are three strategies selected as the comparison strategies in the simulations: ① *first-in-first-out (FIFO) strategy:* similar to [22], [34], [35], the typical FIFO strategy is utilized in the simulations, where the vehicles move into the conflict

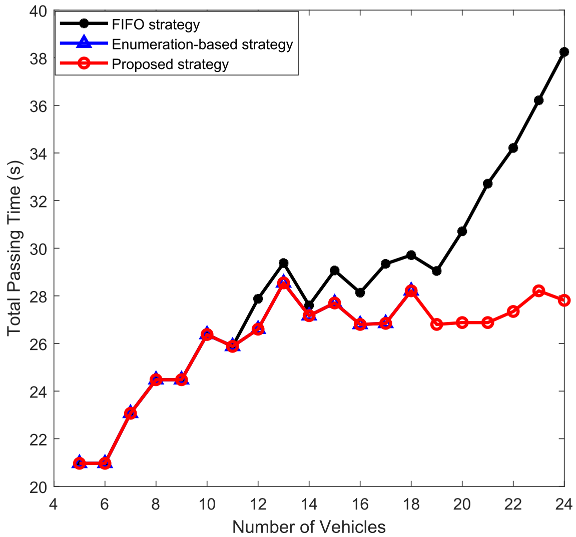

Fig. 5. The total passing time of different strategies with respect to the number of vehicles.

TABLE II

AVERAGE COMPUTATION TIME OF DIFFERENT STRATEGIES

Number of Vehicles	Strategies	Average Computation Time (ms)
10	FIFO strategy	1
	Enumeration-based strategy	18
	Proposed strategy	3
14	FIFO strategy	1
	Enumeration-based strategy	3.0×10^3
	Proposed strategy	6
18	FIFO strategy	1
	Enumeration-based strategy	1.1×10^6
	Proposed strategy	13
24	FIFO strategy	1
	Enumeration-based strategy	/
	Proposed strategy	21

area following the sequence they enter the control area. ② *enumeration-based strategy:* it refers to the strategy that can obtain the globally optimal solution through enumerating all possible solutions, which is utilized to evaluate the global optimality of the proposed strategy. ③ *rule-based strategy:* as presented in [45], the rule-based strategy can dynamically reschedule the sequence of vehicles when a new vehicle enters into the control area, which has a significant advantage over FIFO strategy in improving traffic efficiency.

B. Global Optimality Analysis

In this simulation, we design an intersection scenario with α vehicles ($\alpha \in [5, 24]$) that randomly distributed in the control area to verify the global optimality of the proposed strategy. Three strategies are applied in this scenario, i.e., FIFO strategy, enumeration-based strategy and the proposed strategy. The *total passing time* and *average computation time* are utilized as the performance indices in this simulation. For each α, we simulate 20 times to take the average total passing time and computation time as shown in Fig. 5 and TABLE II. It should be pointed out that the computation time of enumeration-based strategy becomes extremely large when $\alpha > 18$. Thus, enumeration-based strategy is only applied in the scenarios where $\alpha \leq 18$ in this simulation.

According to Fig. 5, we can find that the total passing time of the proposed strategy is exactly equal to that of the enumeration-based strategy, which can obtain the globally optimal solution. Compared with the FIFO strategy, the total passing time of the proposed strategy is significantly decreased especially when the traffic volume is large. Furthermore, according to TABLE II, the average computation time of the proposed strategy is short enough and can realize on-line computation in cooperative driving problem. However, the enumeration-based strategy is computationally intractable for a real-time implementation especially when there are a large number of vehicles within the control area.

Consequently, the proposed strategy can obtain the globally optimal solution with a limited computation time.

TABLE III

COMPARISON RESULTS OF DIFFERENT STRATEGIES

Arrival Rate $veh/(lane \cdot h)$	Strategies	Traffic Throughput	Computation Time (ms)
400	FIFO strategy	253	1
	Rule-based strategy	254	1
	Proposed strategy	255	1
500	FIFO strategy	290	1
	Rule-based strategy	326	2
	Proposed strategy	334	4
600	FIFO strategy	292	1
	Rule-based strategy	351	5
	Proposed strategy	370	7
700	FIFO strategy	303	1
	Rule-based strategy	354	7
	Proposed strategy	378	62
800	FIFO strategy	297	1
	Rule-based strategy	361	10
	Proposed strategy	381	71
900	FIFO strategy	297	1
	Rule-based strategy	363	12
	Proposed strategy	380	76

C. Comparison Results in Continuous Traffic Process

In this simulation, the performance of the proposed strategy is further validated in a continuous traffic process. Similar to [22], [34], we assume that the vehicles arrive in a Poisson Process at each input lane, and the range of the average arrival rate is set as $[400, 900]$ $veh/(lane \cdot h)$. For each parameter setting, we simulate a 10-minute study period (after a 2-minute warm up) to collect the numerical results. In the continuous traffic process, the planning procedure is triggered when a new vehicle moves into the control area to reschedule the sequence of vehicles within the control area. In addition, similar to the relevant studies [26], [34], we select the method presented in [46] to obtain the trajectory for each vehicle based on the assigned arrival time of vehicles. As validated in [46], this trajectory planning method can significantly reduce the emissions according to the pre-assigned arrival time.

Note that it is an extremely time-consuming process to obtain the globally optimal solution using the existing optimal strategies. Thus, the enumeration-based strategy is not adopted in this simulation and we select the FIFO strategy for

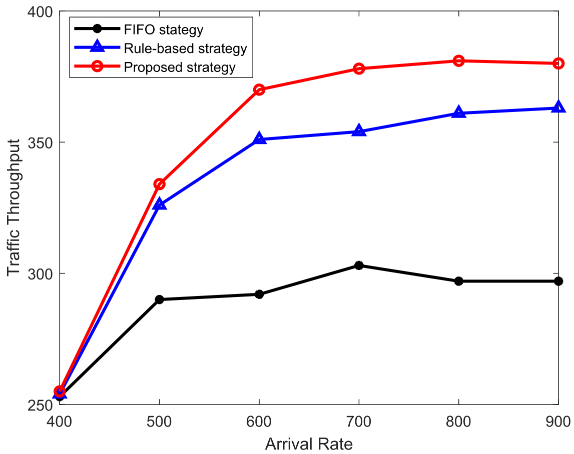

Fig. 6. Throughput of different strategies with respect to the arrival rate.

comparisons. In addition, the rule-based strategy [45] is also utilized as the comparison strategy to verify the advantages of our method. In this simulation, the *traffic throughput* and *average computation time* are selected as the performance indices.

According to TABLE III and Fig. 6, we can find that the proposed strategy outperforms the FIFO strategy and the rule-based strategy in terms of traffic throughput under all traffic demands. Moreover, the average computation time of the proposed strategy is short enough ($< 0.1s$) in all simulations, which can fulfill the real-time computation requirement for real-world implementation.

VII. Conclusion

In this paper, we study the problem of cooperative driving at signal-free intersections, aiming to obtain the globally optimal solution with an efficient computation time. Taking advantage of the conflict relations between vehicles, we show that the large-scale planning problem can be reformulated as a small-size multi-stage decision problem using the idea of dynamic programming, which can reduce the time complexity from exponential to a small-degree polynomial. Multiple simulation results and theoretical analysis jointly demonstrate the promising performance of the proposed strategy in terms of improving traffic efficiency and reducing computational complexity.

In future work, some interesting aspects are also worth further investigation. For instance, cooperative driving at a multi-lane intersection can be considered based on the approach presented in this paper; the scenarios where the human-driven vehicles coexist with CAVs can also be discussed [47], [48].

Appendix I
Proof of Theorem 1

Proof: Recalling that the number of vehicles on lane 1, 2, 3 and 4 is denoted by $\hat{n}_1$, $\hat{n}_2$, $\hat{n}_3$ and $\hat{n}_4$ respectively, and assuming that $\hat{n}_1 > 0$, $\hat{n}_2 > 0$, $\hat{n}_3 > 0$ and $\hat{n}_4 > 0$. To simplify the descriptions, we give a symbolic function, i.e.,

$$\Gamma(x) = \begin{cases} 0 & x = 0 \\ 1 & x \neq 0. \end{cases}$$

Then, the number of states in the solution space can be analyzed by categories. As for $\Gamma(n_1)+\Gamma(n_2)+\Gamma(n_3)+\Gamma(n_4) = 0$, there is only one state satisfying condition, i.e., the initial state in stage 0. As for $\Gamma(n_1)+\Gamma(n_2)+\Gamma(n_3)+\Gamma(n_4) = 1$, there are $(\hat{n}_1+\hat{n}_2+\hat{n}_3+\hat{n}_4)$ states satisfying condition, e.g., $s_1\left(\begin{smallmatrix}1&0\\0&0\end{smallmatrix}\right)$, $s_1\left(\begin{smallmatrix}2&0\\0&0\end{smallmatrix}\right)$, $s_2\left(\begin{smallmatrix}0&1\\0&0\end{smallmatrix}\right)$. As for $\Gamma(n_1)+\Gamma(n_2)+\Gamma(n_3)+\Gamma(n_4) = 2$, there are $(2\hat{n}_1\hat{n}_2 + 2\hat{n}_1\hat{n}_3 + 2\hat{n}_1\hat{n}_4 + 2\hat{n}_2\hat{n}_3 + 2\hat{n}_2\hat{n}_4 + 2\hat{n}_3\hat{n}_4)$ states satisfying condition, e.g., $s_1\left(\begin{smallmatrix}1&1\\0&0\end{smallmatrix}\right)$, $s_1\left(\begin{smallmatrix}1&0\\1&0\end{smallmatrix}\right)$, $s_2\left(\begin{smallmatrix}0&1\\1&0\end{smallmatrix}\right)$. As for $\Gamma(n_1)+\Gamma(n_2)+\Gamma(n_3)+\Gamma(n_4) = 3$, there are $(3\hat{n}_1\hat{n}_2\hat{n}_3 + 3\hat{n}_1\hat{n}_2\hat{n}_4 + 3\hat{n}_1\hat{n}_3\hat{n}_4 + 3\hat{n}_2\hat{n}_3\hat{n}_4)$ states satisfying condition, e.g., $s_1\left(\begin{smallmatrix}1&1\\1&0\end{smallmatrix}\right)$, $s_1\left(\begin{smallmatrix}1&1\\0&1\end{smallmatrix}\right)$, $s_2\left(\begin{smallmatrix}0&1\\1&2\end{smallmatrix}\right)$. As for $\Gamma(n_1)+\Gamma(n_2)+\Gamma(n_3)+\Gamma(n_4) = 4$, there are $(4\hat{n}_1\hat{n}_2\hat{n}_3\hat{n}_4)$ states satisfying condition, e.g., $s_1\left(\begin{smallmatrix}1&1\\1&1\end{smallmatrix}\right)$, $s_1\left(\begin{smallmatrix}1&1\\2&1\end{smallmatrix}\right)$, $s_2\left(\begin{smallmatrix}3&1\\1&2\end{smallmatrix}\right)$. Based on the above descriptions, we can obtain the total number of states N_s in the state space as

$$\begin{aligned} N_s &= 4\hat{n}_1\hat{n}_2\hat{n}_3\hat{n}_4 + 3(\hat{n}_1\hat{n}_2\hat{n}_3 + \hat{n}_1\hat{n}_2\hat{n}_4 + \hat{n}_1\hat{n}_3\hat{n}_4 \\ &\quad + \hat{n}_2\hat{n}_3\hat{n}_4) + 2(\hat{n}_1\hat{n}_2 + \hat{n}_1\hat{n}_3 + \hat{n}_1\hat{n}_4 + \hat{n}_2\hat{n}_3 \\ &\quad + \hat{n}_2\hat{n}_4 + \hat{n}_3\hat{n}_4) + \hat{n}_1 + \hat{n}_2 + \hat{n}_3 + \hat{n}_4 + 1. \end{aligned}$$

Assuming that the total number of vehicles in control area is $N (N \geq 4)$ and $\hat{n}_1 = \hat{n}_2 = \hat{n}_3 = \hat{n}_4 = \frac{N}{4}$, we have

$$N_s = \frac{N^4}{64} + \frac{3N^3}{16} + \frac{3N^2}{4} + N + 1.$$

It is obvious that the number of states is $O(N^4)$. ■

Appendix II
Proof of Theorem 2-3

Recalling that the number of vehicles on lane 1, 2, 3 and 4 is denoted by $\hat{n}_1$, $\hat{n}_2$, $\hat{n}_3$ and $\hat{n}_4$ respectively, and assuming that $\hat{n}_1 > 0$, $\hat{n}_2 > 0$, $\hat{n}_3 > 0$ and $\hat{n}_4 > 0$.

A. Proof of Theorem 2

Proof: The number of transitions reaches the lower bound N_t^l under basic state transition. Usually, each state has four predecessor states except the initial state, i.e.,

$$N_{t,\max}^l = 4(N_s - 1),$$

where $N_{t,\max}^l$ denotes the possible maximum of transitions under basic state transition. However, there exist several infeasible transitions, which need to be subtracted from $N_{t,\max}^l$. As for such states as $s_1\left(\begin{smallmatrix}n_1&n_2\\n_3&n_4\end{smallmatrix}\right)$, there are generally four kinds of predecessor states, i.e., $s_1\left(\begin{smallmatrix}n_1-1&n_2\\n_3&n_4\end{smallmatrix}\right)$, $s_2\left(\begin{smallmatrix}n_1-1&n_2\\n_3&n_4\end{smallmatrix}\right)$, $s_3\left(\begin{smallmatrix}n_1-1&n_2\\n_3&n_4\end{smallmatrix}\right)$, and $s_4\left(\begin{smallmatrix}n_1-1&n_2\\n_3&n_4\end{smallmatrix}\right)$. We now discuss whether these predecessor states exist in state space. ① if $n_1 = 1$, state $s_1\left(\begin{smallmatrix}n_1-1&n_2\\n_3&n_4\end{smallmatrix}\right)$ is infeasible, where $n_2 \in [0, \hat{n}_2]$, $n_3 \in [0, \hat{n}_3]$ and $n_4 \in [0, \hat{n}_4]$. Thus, there are $[(\hat{n}_2 + 1)(\hat{n}_3 + 1)(\hat{n}_4 + 1)]$ states belonging to this category. ② if $n_2 = 0$, $s_2\left(\begin{smallmatrix}n_1-1&n_2\\n_3&n_4\end{smallmatrix}\right)$ is infeasible, where $n_1 \in [1, \hat{n}_1]$, $n_3 \in [0, \hat{n}_3]$ and $n_4 \in [0, \hat{n}_4]$. Thus, there are $[\hat{n}_1(\hat{n}_3 + 1)(\hat{n}_4 + 1)]$ states belonging to this category. ③ if $n_3 = 0$, $s_3\left(\begin{smallmatrix}n_1-1&n_2\\n_3&n_4\end{smallmatrix}\right)$ is infeasible, where $n_1 \in [1, \hat{n}_1]$, $n_2 \in [0, \hat{n}_2]$ and $n_4 \in [0, \hat{n}_4]$. Thus, there are $[\hat{n}_1(\hat{n}_2+1)(\hat{n}_4+1)]$ states belonging to this category. ④ if $n_4 = 0$, $s_4\left(\begin{smallmatrix}n_1-1&n_2\\n_3&n_4\end{smallmatrix}\right)$ is infeasible, where $n_1 \in [1, \hat{n}_1]$, $n_2 \in [0, \hat{n}_2]$ and $n_3 \in [0, \hat{n}_3]$. Thus, there are $[\hat{n}_1(\hat{n}_2 + 1)(\hat{n}_3 + 1)]$ states of this category. In summary, as for such states as $s_1\left(\begin{smallmatrix}n_1&n_2\\n_3&n_4\end{smallmatrix}\right)$,

the number of infeasible transitions $N_{t,inf,1}$ that need to be subtracted from $N^l_{t,\max}$ is

$$\begin{aligned}N_{t,inf,1} = {} & (\hat{n}_2+1)(\hat{n}_3+1)(\hat{n}_4+1)+\hat{n}_1(\hat{n}_3+1)(\hat{n}_4+1)\\ & +\hat{n}_1(\hat{n}_2+1)(\hat{n}_4+1)+\hat{n}_1(\hat{n}_2+1)(\hat{n}_3+1)\end{aligned}$$

Similarly, as for such states as $s_2\begin{pmatrix} n_1 & n_2\\ n_3 & n_4\end{pmatrix}$, $s_3\begin{pmatrix} n_1 & n_2\\ n_3 & n_4\end{pmatrix}$, and $s_4\begin{pmatrix} n_1 & n_2\\ n_3 & n_4\end{pmatrix}$, the number of transitions that need to be subtracted from $N^l_{t,\max}$ is

$$\begin{aligned}N_{t,inf,2} = {} & (\hat{n}_1+1)(\hat{n}_3+1)(\hat{n}_4+1)+\hat{n}_2(\hat{n}_3+1)(\hat{n}_4+1)\\ & +\hat{n}_2(\hat{n}_1+1)(\hat{n}_4+1)+\hat{n}_2(\hat{n}_1+1)(\hat{n}_3+1),\\ N_{t,inf,3} = {} & (\hat{n}_1+1)(\hat{n}_2+1)(\hat{n}_4+1)+\hat{n}_3(\hat{n}_2+1)(\hat{n}_4+1)\\ & +\hat{n}_3(\hat{n}_1+1)(\hat{n}_4+1)+\hat{n}_3(\hat{n}_1+1)(\hat{n}_2+1),\\ N_{t,inf,4} = {} & (\hat{n}_1+1)(\hat{n}_2+1)(\hat{n}_3+1)+\hat{n}_4(\hat{n}_2+1)(\hat{n}_3+1)\\ & +\hat{n}_4(\hat{n}_1+1)(\hat{n}_3+1)+\hat{n}_4(\hat{n}_1+1)(\hat{n}_2+1).\end{aligned}$$

In addition, there are four states that directly connect to the initial state, i.e., $s_1\begin{pmatrix} 1 & 0\\ 0 & 0\end{pmatrix}$, $s_2\begin{pmatrix} 0 & 1\\ 0 & 0\end{pmatrix}$, $s_3\begin{pmatrix} 0 & 0\\ 0 & 1\end{pmatrix}$ and $s_4\begin{pmatrix} 0 & 0\\ 0 & 1\end{pmatrix}$, which produces four special transitions. Therefore, the lower bound is summarized as

$$\begin{aligned}N^l_t &= N^l_{t,\max}-N_{t,inf,1}-N_{t,inf,2}-N_{t,inf,3}-N_{t,inf,4}+4\\ &= 16\hat{n}_1\hat{n}_2\hat{n}_3\hat{n}_4+8(\hat{n}_1\hat{n}_2\hat{n}_3+\hat{n}_1\hat{n}_2\hat{n}_4+\hat{n}_1\hat{n}_3\hat{n}_4+\hat{n}_2\hat{n}_3\hat{n}_4)\\ &\quad +2(\hat{n}_1\hat{n}_2+\hat{n}_1\hat{n}_3+\hat{n}_1\hat{n}_4+\hat{n}_2\hat{n}_3+\hat{n}_2\hat{n}_4+\hat{n}_3\hat{n}_4)\\ &\quad -2(\hat{n}_1+\hat{n}_2+\hat{n}_3+\hat{n}_4)+4.\end{aligned}$$

Assuming that the total number of vehicles in control area is $N(N\geq 4)$ and $\hat{n}_1=\hat{n}_2=\hat{n}_3=\hat{n}_4=\frac{N}{4}$, we have

$$N^l_t = \frac{N^4}{16}+\frac{N^3}{2}+\frac{3N^2}{4}-2N+4.$$

Thus, the lower bound on number of transitions is $O(N^4)$. ■

B. Proof of Theorem 3

Proof: According to Property 2 in *Section IV-C*, we know that removing one state transition may reproduce multiple transitions which do not connect the states in adjacent stages. Thus, the total number of transitions is

$$N_t = N^l_t - N_{t,\text{rem}} + N_{t,\text{rep}}, \tag{18}$$

where $N_{t,\text{rem}}$ denotes the number of removed transitions, and $N_{t,\text{rep}}$ denotes the number of reproduced transitions.

According to Property 1 in *Section IV-C*, the reproduced transitions represent a group of vehicles which do not conflict with each other. Firstly, we aim to find the number of the reproduced transitions that connect to a particular state $s_r\begin{pmatrix} n_1^0+\delta_{n_1} & n_2\\ n_3^0+\delta_{n_3} & n_4\end{pmatrix}$, where we assume that the $n_1^{th}(n_1^{th}\in[n_1^0+1, n_1^0+\delta_{n_1}])$ vehicle on lane 1 has no conflict with the $n_3^{th}(n_3^{th}\in[n_3^0+1, n_3^0+\delta_{n_3}])$ on lane 3. Thus, there are $\delta_{n_1}\delta_{n_3}$ reproduced transitions connecting to state $s_r\begin{pmatrix} n_1^0+\delta_{n_1} & n_2\\ n_3^0+\delta_{n_3} & n_4\end{pmatrix}$ when n_2, n_4 and r are fixed. Secondly, according to *Section V-B*, the total number of transitions reaches the upper bound N^u_t when each vehicle within the control area does not conflict with any vehicle on the opposite lane. In such case, we aim to find the total number of reproduced transitions for state $s_r\begin{pmatrix} n_1^0+\delta_{n_1} & n_2\\ n_3^0+\delta_{n_3} & n_4\end{pmatrix}$, where δ_{n_1} and δ_{n_3} are set as different values. Obviously, we have $\delta_{n_1}\in[1,\hat{n}_1]$ and $\delta_{n_3}\in[1,\hat{n}_3]$. Thus, the total number of the reproduced transitions connecting to this kind of states where n_2, n_4 and r are fixed is

$$\sum_{\delta_{n_1}=1}^{\hat{n}_1}\sum_{\delta_{n_3}=1}^{\hat{n}_3}\delta_{n_1}\delta_{n_3} = \frac{\hat{n}_1(\hat{n}_1+1)\hat{n}_3(\hat{n}_3+1)}{4}.$$

Then, we consider that n_2, n_4 and r can set as different values. As for $n_2\in[1,\hat{n}_2]$ and $n_4\in[1,\hat{n}_4]$, $r\in\{2,4\}$. As for $n_2=0$ and $n_4\in[1,\hat{n}_4]$, $r=4$. As for $n_4=0$ and $n_2\in[1,\hat{n}_2]$, $r=2$. Therefore, for such states as $s_r\begin{pmatrix} n_1^0+\delta_{n_1} & n_2\\ n_3^0+\delta_{n_3} & n_4\end{pmatrix}$, the total number of reproduced transitions $N_{t,\text{rep},1}$ is

$$N_{t,\text{rep},1} = (2\hat{n}_2\hat{n}_4+\hat{n}_2+\hat{n}_4)\frac{\hat{n}_1(\hat{n}_1+1)\hat{n}_3(\hat{n}_3+1)}{4}.$$

Similarly, for such states as $s_r\begin{pmatrix} n_1 & n_2^0+\delta_{n_2}\\ n_3 & n_4^0+\delta_{n_4}\end{pmatrix}$, the total number of reproduced transitions is

$$N_{t,\text{rep},2} = (2\hat{n}_1\hat{n}_3+\hat{n}_1+\hat{n}_3)\frac{\hat{n}_2(\hat{n}_2+1)\hat{n}_4(\hat{n}_4+1)}{4}.$$

In addition, there are $(2\hat{n}_1\hat{n}_3+2\hat{n}_2\hat{n}_4)$ reproduced transitions from the initial state to such states as $s_r\begin{pmatrix} n_1 & 0\\ n_3 & 0\end{pmatrix}$ and $s_r\begin{pmatrix} 0 & n_2\\ 0 & n_4\end{pmatrix}$. Based on above descriptions, we have

$$N_{t,\text{rep}} = N_{t,\text{rep},1}+N_{t,\text{rep},2}+2\hat{n}_1\hat{n}_3+2\hat{n}_2\hat{n}_4. \tag{19}$$

Furthermore, it is easy to obtain $N_{t,\text{rem}}$, i.e.,

$$N_{t,\text{rem}} = 2\hat{n}_1\hat{n}_3(\hat{n}_2+1)(\hat{n}_4+1)+2\hat{n}_2\hat{n}_4(\hat{n}_1+1)(\hat{n}_3+1). \tag{20}$$

Finally, according to (18), (19) and (20), we can obtain the upper bound on the number of transitions, i.e.,

$$\begin{aligned}N^u_t &= N^l_t+(2\hat{n}_2\hat{n}_4+\hat{n}_2+\hat{n}_4)\frac{\hat{n}_1(\hat{n}_1+1)\hat{n}_3(\hat{n}_3+1)}{4}\\ &\quad +(2\hat{n}_1\hat{n}_3+\hat{n}_1+\hat{n}_3)\frac{\hat{n}_2(\hat{n}_2+1)\hat{n}_4(\hat{n}_4+1)}{4}\\ &\quad +2\hat{n}_1\hat{n}_3+2\hat{n}_2\hat{n}_4\\ &\quad -2\hat{n}_1\hat{n}_3(\hat{n}_2+1)(\hat{n}_4+1)-2\hat{n}_2\hat{n}_4(\hat{n}_1+1)(\hat{n}_3+1).\end{aligned}$$

Assuming that the total number of vehicles in control area is $N(N\geq 4)$ and $\hat{n}_1=\hat{n}_2=\hat{n}_3=\hat{n}_4=\frac{N}{4}$, we have

$$N^u_t = \frac{N^6}{4096}+\frac{3N^5}{1024}+\frac{15N^4}{256}+\frac{25N^3}{64}+\frac{3N^2}{4}-2N.$$

Thus, the upper bound on number of transitions is $O(N^6)$. ■

APPENDIX III
PROOF OF THEOREM 4

Proof: In Algorithm 1, for each state transition, we need to calculate and update the objective value of current state. To further reduce the complexity, instead of using Eq. (11a)-Eq. (12), it is easy to obtain the objective value according to the conflict relations between vehicles. Specifically, if multiple vehicles can obtain the right of way during current state transition and these vehicles are divided into two continuous queues in different lanes (e.g., Fig. 4), we just need to calculate the arrival time assigned to the last vehicle in each queue and then pick the maximum as the objective value, i.e.,

$$\begin{aligned} t_{\text{assign},r,l} &= \max\left(t_{\text{assign},r,f} + (N_r - 1)\Delta_{t,1}, t_{\min,r,l}\right), \\ t_{\text{assign},o,l} &= \max\left(t_{\text{assign},o,f} + (N_o - 1)\Delta_{t,1}, t_{\min,o,l}\right), \\ J &= \max\left(t_{\text{assign},r,l}, t_{\text{assign},o,l}\right), \end{aligned}$$

where lane o is the opposite lane of lane r. $t_{\text{assign},r,l}$ and $t_{\text{assign},o,l}$ denote the arrival time assigned to the last vehicle of the queue in lane r and lane o respectively. $t_{\text{assign},r,f}$ and $t_{\text{assign},o,f}$ denote the arrival time assigned to the first vehicle of the queue in lane r and lane o respectively. N_r and N_o denote the size of the queue in lane r and lane$o$ respectively. $t_{\min,r,l}$ and $t_{\min,o,l}$ denote the minimal arrival time of the last vehicle of the queue in lane r and lane o respectively.

Obviously, the time complexity of calculating the objective value is a constant time for each transition, since the computation time does not change with the size of the algorithm input (i.e., the total number of vehicles). Therefore, based on Theorem 2-3, it is obvious that the optimal cooperative driving strategy has the polynomial-time complexity of computation, which has a lower bound $O(N^4)$ and an upper bound $O(N^6)$, where N denotes the number of vehicles. ■

References

[1] P. Li and X. Zhou, "Recasting and optimizing intersection automation as a connected-and-automated-vehicle (CAV) scheduling problem: A sequential branch-and-bound search approach in phase-time-traffic hypernetwork," *Transp. Res. B, Methodol.*, vol. 105, pp. 479–506, Nov. 2017.

[2] Y. Feng, C. Yu, and H. X. Liu, "Spatiotemporal intersection control in a connected and automated vehicle environment," *Transp. Res. C, Emerg. Technol.*, vol. 89, pp. 364–383, Apr. 2019.

[3] C. Yu, W. Sun, H. X. Liu, and X. Yang, "Managing connected and automated vehicles at isolated intersections: From reservation-to optimization-based methods," *Transp. Res. B, Methodol.*, vol. 122, pp. 416–435, Apr. 2019.

[4] A. A. Malikopoulos, S. Hong, B. B. Park, J. Lee, and S. Ryu, "Optimal control for speed harmonization of automated vehicles," *IEEE Trans. Intell. Transp. Syst.*, vol. 20, no. 7, pp. 2405–2417, Jul. 2019.

[5] H. Pei, Y. Zhang, Q. Tao, S. Feng, and L. Li, "Distributed cooperative driving in multi-intersection road networks," *IEEE Trans. Veh. Technol.*, vol. 70, no. 6, pp. 5390–5403, Jun. 2021.

[6] S. Feng, Y. Zhang, S. E. Li, Z. Cao, H. X. Liu, and L. Li, "String stability for vehicular platoon control: Definitions and analysis methods," *Annu. Rev. Control*, vol. 47, pp. 81–97, Mar. 2019.

[7] J. Zhang, Z. Li, L. Li, Y. Li, and H. Dong, "A bi-level cooperative operation approach for AGV based automated valet parking," *Transp. Res. C, Emerg. Technol.*, vol. 128, Jul. 2021, Art. no. 103140.

[8] M. Tajalli, M. Mehrabipour, and A. Hajbabaie, "Network-level coordinated speed optimization and traffic light control for connected and automated vehicles," *IEEE Trans. Intell. Transp. Syst.*, early access, Jun. 5, 2020, doi: 10.1109/TITS.2020.2994468.

[9] M. Tajalli and A. Hajbabaie, "Traffic signal timing and trajectory optimization in a mixed autonomy traffic stream," *IEEE Trans. Intell. Transp. Syst.*, early access, Feb. 18, 2021, doi: 10.1109/TITS.2021.3058193.

[10] R. Niroumand, M. Tajalli, L. Hajibabai, and A. Hajbabaie, "Joint optimization of vehicle-group trajectory and signal timing: Introducing the white phase for mixed-autonomy traffic stream," *Transp. Res. C, Emerg. Technol.*, vol. 116, Jul. 2020, Art. no. 102659.

[11] D. Rey and M. W. Levin, "Blue phase: Optimal network traffic control for legacy and autonomous vehicles," *Transp. Res. B, Methodol.*, vol. 130, pp. 105–129, Dec. 2019.

[12] C. Yu, Y. Feng, H. X. Liu, W. Ma, and X. Yang, "Integrated optimization of traffic signals and vehicle trajectories at isolated urban intersections," *Transp. Res. B, Methodol.*, vol. 112, pp. 89–112, Jun. 2018.

[13] B. Xu *et al.*, "Distributed conflict-free cooperation for multiple connected vehicles at unsignalized intersections," *Transp. Res. C, Emerg. Technol.*, vol. 93, pp. 322–334, Aug. 2018.

[14] J. Lee and B. Park, "Development and evaluation of a cooperative vehicle intersection control algorithm under the connected vehicles environment," *IEEE Trans. Intell. Transp. Syst.*, vol. 13, no. 3, pp. 81–90, Mar. 2012.

[15] A. Mirheli, M. Tajalli, L. Hajibabai, and A. Hajbabaie, "A consensus-based distributed trajectory control in a signal-free intersection," *Transp. Res. C, Emerg. Technol.*, vol. 100, pp. 161–176, Mar. 2019.

[16] A. Mirheli, L. Hajibabai, and A. Hajbabaie, "Development of a signal-head-free intersection control logic in a fully connected and autonomous vehicle environment," *Transp. Res. C, Emerg. Technol.*, vol. 92, pp. 412–425, Jul. 2018.

[17] S. I. Guler, M. Menendez, and L. Meier, "Using connected vehicle technology to improve the efficiency of intersections," *Transp. Res. C, Emerg. Technol.*, vol. 46, pp. 121–131, Sep. 2014.

[18] Q. Guo, L. Li, and X. J. Ban, "Urban traffic signal control with connected and automated vehicles: A survey," *Transp. Res. C, Emerg. Technol.*, vol. 101, pp. 313–334, Apr. 2019.

[19] L. Li, D. Wen, and D. Y. Yao, "A survey of traffic control with vehicular communications," *IEEE Trans. Intell. Transp. Syst.*, vol. 15, no. 1, pp. 425–432, Feb. 2014.

[20] Y. Meng, L. Li, F.-Y. Wang, K. Li, and Z. Li, "Analysis of cooperative driving strategies for nonsignalized intersections," *IEEE Trans. Veh. Technol.*, vol. 67, no. 4, pp. 2900–2911, Apr. 2018.

[21] H. Yu *et al.*, "Automated vehicle-involved traffic flow studies: A survey of assumptions, models, speculations, and perspectives," *Transp. Res. C, Emerg. Technol.*, vol. 127, Jun. 2021, Art. no. 103101.

[22] H. Pei, S. Feng, Y. Zhang, and D. Yao, "A cooperative driving strategy for merging at on-ramps based on dynamic programming," *IEEE Trans. Veh. Technol.*, vol. 68, no. 12, pp. 11646–11656, Dec. 2019.

[23] H. Xu, S. Feng, Y. Zhang, and L. Li, "A grouping-based cooperative driving strategy for CAVs merging problems," *IEEE Trans. Veh. Technol.*, vol. 68, no. 6, pp. 6125–6136, Jun. 2019.

[24] S. Jing, F. Hui, X. Zhao, J. Rios-Torres, and A. J. Khattak, "Cooperative game approach to optimal merging sequence and on-ramp merging control of connected and automated vehicles," *IEEE Trans. Intell. Transp. Syst.*, vol. 20, no. 11, pp. 4234–4244, Nov. 2019.

[25] J. Rios-Torres and A. A. Malikopoulos, "A survey on the coordination of connected and automated vehicles at intersections and merging at highway on-ramps," *IEEE Trans. Intell. Transp. Syst.*, vol. 18, no. 5, pp. 1066–1077, May 2017.

[26] J. Ding, L. Li, H. Peng, and Y. Zhang, "A rule-based cooperative merging strategy for connected and automated vehicles," *IEEE Trans. Intell. Transp. Syst.*, vol. 21, no. 8, pp. 3436–3446, Aug. 2020.

[27] E. R. Müller, R. C. Carlson, and W. K. Junior, "Intersection control for automated vehicles with MILP," *IFAC-Papers Line*, vol. 49, no. 3, pp. 37–42, 2016.

[28] H. Ahn and D. D. Vecchio, "Safety verification and control for collision avoidance at road intersections," *IEEE Trans. Autom. Control*, vol. 63, no. 3, pp. 630–642, Mar. 2018.

[29] L. Li and F. Y. Wang, "Cooperative driving at blind crossings using intervehicle communication," *IEEE Trans. Veh. Technol.*, vol. 55, no. 6, pp. 1712–1724, Nov. 2006.

[30] K. Dresner and P. Stone, "Multiagent traffic management: An improved intersection control mechanism," in *Proc. 4th Int. Joint Conf. Auton. Agents Multiagent Syst.*, 2005, pp. 471–477.

[31] K. Dresner and P. Stone, "A multiagent approach to autonomous intersection management," *J. Artif. Intell. Res.*, vol. 31, pp. 591–656, Mar. 2008.

[32] S. Huang, A. W. Sadek, and Y. Zhao, "Assessing the mobility and environmental benefits of reservation-based intelligent intersections using an integrated simulator," *IEEE Trans. Intell. Transp. Syst.*, vol. 13, no. 3, pp. 1201–1214, Sep. 2012.

[33] M. Choi, A. Rubenecia, and H. H. Choi, "Reservation-based cooperative traffic management at an intersection of multi-lane roads," in *Proc. Int. Conf. Inf. Netw. (ICOIN)*, Jan. 2018, pp. 456–460.

[34] H. Xu, Y. Zhang, L. Li, and W. Li, "Cooperative driving at unsignalized intersections using tree search," *IEEE Trans. Intell. Transp. Syst.*, vol. 21, no. 11, pp. 4563–4571, Nov. 2019.

[35] H. Xu, Y. Zhang, C. G. Cassandras, L. Li, and S. Feng, "A bi-level cooperative driving strategy allowing lane changes," *Transp. Res. C, Emerg. Technol.*, vol. 120, Nov. 2020, Art. no. 102773.

[36] J. F. Baldwin, "Principles of dynamic programming Part 1: Basic analytic and computational methods," *Electron. Power*, vol. 25, no. 4, pp. 230–231, 1979.

[37] E. V. Denardo, *Dynamic Programming: Models and Applications*. New York, NY, USA: Dover, 2003.

[38] D. Carlino, S. D. Boyles, and P. Stone, "Auction-based autonomous intersection management," in *Proc. 16th Int. IEEE Conf. Intell. Transp. Syst. (ITSC)*, Oct. 2013, pp. 529–534.

[39] Q. Jin, G. Wu, K. Boriboonsomsin, and M. Barth, "Platoon-based multi-agent intersection management for connected vehicle," in *Proc. 16th Int. IEEE Conf. Intell. Transp. Syst. (ITSC)*, Oct. 2013, pp. 1462–1467.

[40] M. Bashiri, H. Jafarzadeh, and C. H. Fleming, "PAIM: Platoon-based autonomous intersection management," in *Proc. 21st Int. Conf. Intell. Transp. Syst. (ITSC)*, Nov. 2018, pp. 374–380.

[41] M. Bashiri and C. H. Fleming, "A platoon-based intersection management system for autonomous vehicles," in *Proc. IEEE Intell. Vehicles Symp. (IV)*, Jun. 2017, pp. 667–672.

[42] A. M. I. Mahbub, A. A. Malikopoulos, and L. Zhao, "Decentralized optimal coordination of connected and automated vehicles for multiple traffic scenarios," *Automatica*, vol. 117, Jul. 2020, Art. no. 108958.

[43] L. Chen and C. Englund, "Cooperative intersection management: A survey," *IEEE Trans. Intell. Transp. Syst.*, vol. 17, no. 2, pp. 570–586, Feb. 2016.

[44] E. M. L. Beale, "Branch and bound methods for mathematical programming systems," *Discrete Optim.*, vol. 5, no. 5, pp. 201–219, 1979.

[45] Y. Zhang and C. G. Cassandras, "A decentralized optimal control framework for connected automated vehicles at urban intersections with dynamic resequencing," in *Proc. IEEE Conf. Decis. Control (CDC)*, Dec. 2018, pp. 217–222.

[46] A. A. Malikopoulos, C. G. Cassandras, and Y. Zhang, "A decentralized energy-optimal control framework for connected automated vehicles at signal-free intersections," *Automatica*, vol. 93, pp. 244–256, Jul. 2018.

[47] S. M. A. B. A. Islam, A. Hajbabaie, and H. M. A. Aziz, "A real-time network-level traffic signal control methodology with partial connected vehicle information," *Transp. Res. C, Emerg. Technol.*, vol. 121, Dec. 2020, Art. no. 102830.

[48] S. Feng, Z. Song, Z. Li, Y. Zhang, and L. Li, "Robust platoon control in mixed traffic flow based on tube model predictive control," *IEEE Trans. Intell. Veh.*, early access, Feb. 19, 2021, doi: 10.1109/TIV.2021.3060626.

Huaxin Pei received the B.S. degree from the University of Electronic Science and Technology of China, China, in 2017. He is currently pursuing the Ph.D. degree in the control science and engineering with the Department of Automation, Tsinghua University, China. His current research interests include optimization for intelligent vehicles, cooperative driving for connected and automated vehicles, and intelligent vehicle-infrastructure cooperative systems.

Yuxiao Zhang received the B.S. degree from the Department of Computer Science and Technology, Tsinghua University, China, in 2021. He is currently pursuing the master's degree with the Urban Mobility Institute, Tongji University, China. His current research interests include vehicle-infrastructure cooperative systems and urban traffic management.

Yi Zhang (Member, IEEE) received the B.S. and M.S. degrees from Tsinghua University, China, in 1986 and 1988, respectively, and the Ph.D. degree from the University of Strathclyde, U.K., in 1995. He is currently a Professor in control science and engineering at Tsinghua University, with his current research interests focusing on intelligent transportation systems. His active research areas include intelligent vehicle-infrastructure cooperative systems, analysis of urban transportation systems, urban road network management, traffic data fusion and dissemination, and urban traffic control and management. His research fields also cover the advanced control theory and applications, advanced detection and measurement, and systems engineering.

Shuo Feng (Member, IEEE) received the B.S. and Ph.D. degrees from the Department of Automation, Tsinghua University, China, in 2014 and 2019, respectively. He was also a Visiting Ph.D. Student with the Department of Civil and Environmental Engineering, University of Michigan, Ann Arbor, from 2017 to 2019, where he is currently a Post-Doctoral Researcher. His current research interests include testing, evaluation, and optimization of connected and automated vehicles. He received the Best Ph.D. Dissertation Award from the IEEE Intelligent Transportation Systems Society in 2020.