A Cooperative Driving Strategy for Merging at On-Ramps Based on Dynamic Programming

Huaxin Pei, Shuo Feng, Yi Zhang, *Member, IEEE*, and Danya Yao, *Member, IEEE*

***Abstract*—Cooperative driving emerges as a promising way to improve efficiency and safety for Connected and Automated Vehicles (CAVs). Its key idea is to design a strategy to schedule the movements of neighboring vehicles. The typical cooperative driving strategies can be categorized into two categories. The first category is optimal strategy, which aims to find the globally optimal passing order of vehicles, but the computational cost of this strategy grows significantly with the increasing number of vehicles. The second category is sub-optimal strategy, which uses heuristic rules or other methods to export an acceptable local optimal solution within a limited computation time. However, there usually lacks a rigorous theoretical guarantee of the performances, and further validation is always required for practical applications. To overcome all these limitations, a computationally efficient strategy is proposed to obtain the globally optimal passing order based on dynamic programming (DP). Specifically, the problem of merging at on-ramps is resolved by a DP method, which uses the domain knowledge to reduce the complexity by well defining the state space, state transition, and criterion function. With the DP method, it is proved that the globally optimal passing order can be obtained with the quadratic polynomial computational complexity of $O(N^2)$, where N denotes the number of vehicles. Simulation results demonstrate the performances of the proposed strategy regarding optimality and efficiency.**

***Index Terms*—Connected and automated vehicles (CAVs), cooperative driving, on-ramp merging problem, dynamic programming.**

I. Introduction

COOPERATIVE driving emerges as a promising method to improve traffic efficiency and safety [1]–[7], which is usually utilized in typical traffic scenarios, e.g., merging at on-ramps. With the help of vehicle to vehicle (V2V) and vehicle to infrastructure (V2I) technology, the key idea of cooperative driving is to design a strategy to schedule the movements of neighboring vehicles [8]–[14].

Manuscript received June 1, 2019; revised August 28, 2019; accepted October 7, 2019. Date of publication October 14, 2019; date of current version December 17, 2019. This work was supported in part by National 135 Key R&D Program Projects under Grant 2018YFB1600600, and in part by the National Natural Science Foundation of China under Grant 61673233. The review of this article was coordinated by Dr. H. Zhang. *(Corresponding authors: Shuo Feng; Yi Zhang.)*

H. Pei, S. Feng, and D. Yao are with the Department of Automation, Tsinghua University, Beijing 100084, China (e-mail: phx17@mails.tsinghua.edu.cn; s-feng14@mails.tsinghua.edu.cn; yaody@tsinghua.edu.cn).

Y. Zhang is with the Department of Automation, BNRist, Tsinghua University, Beijing 100084, and 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).

Digital Object Identifier 10.1109/TVT.2019.2947192

As pointed out in [4] and [8], the performance of a cooperative driving strategy is mainly determined by the passing order of vehicles which pass through the conflict zone. To optimize an appropriate passing order, numerous cooperative driving strategies have been proposed. According to their optimization methods, most strategies can be classified into two categories:

1) Optimal strategy: This kind of strategies aims to find the globally optimal passing orders of vehicles. The most straightforward strategy is to enumerate all possible passing order to get the globally optimal solution, which suffers from the high computational complexity. Li *et al.* [15], Müller *et al.* [16] and Ahn *et al.* [17] formulated and solved the problem as a mixed-integer linear programming (MILP) problem. Li *et al.* [8] described the solution space of passing orders by a spanning tree, and a pruning rule was proposed to search the globally optimal passing order. Compared with the enumeration strategy, these strategies can improve computational efficiency by utilizing optimization methods or pruning rules. However, the computational complexity still increases exponentially with the number of passing vehicles, which makes these strategies intractable for real-time applications, especially for high traffic volume scenarios.

2) Sub-optimal strategy: To mitigate the limitation of the optimal strategies, a sub-optimal strategy usually stops at a local optimal passing order within a limited computation time. Ad hoc negotiation based strategies use heuristic rules to get a feasible passing order, e.g., autonomous intersection management (AIM) [18], [19] and reservation strategy [20], [21], which instruct the vehicles to pass through the conflict zone roughly in first-in-first-out (FIFO) manner. However, as pointed out in [20], [22], [23], ad hoc negotiation based strategy cannot effectively alleviate traffic congestion in many situations. To keep a tradeoff between traffic efficiency and computation flexibility, several cooperative driving strategies that focus on local optimal passing order have been recently proposed. Xu *et al.* [22] proposed a grouping-based strategy, which instructed the vehicles to pass through the conflict zone in the form of groups. By Monte Carlo tree search (MCTS) algorithm, heuristic rules were developed to search a local optimal passing order [23]. In [24], the conflicts between different vehicles were calculated offline, which significantly reduced the computational burden. Based on this fundamental framework, the optimization of passing orders was formulated as a non-linear mathematical program, which was solved by a meta-heuristic Tabu search method. The major advantage of these sub-optimal strategies lies at the computational efficiency. To validate the effectiveness of these strategies, numerical

experiments are usually utilized to show that these strategies can obtain a good enough passing order. Nevertheless, there lacks a rigorous theoretical guarantee of the performances and further validation is always required for practical applications.

To overcome the above limitations, this paper proposes a globally optimal yet computationally efficient cooperative driving strategy for merging at on-ramps. Specifically, a dynamic programming (DP) based strategy is designed to obtain the globally optimal passing order with a quadratic polynomial complexity of the number of vehicles, i.e., $O(N^2)$, where N denotes the number of vehicles. Compared with sub-optimal strategies, the proposed strategy guarantees the optimality of the passing order by the principle of optimality of the DP method [25], [26]. Compared with existing optimal strategies, the proposed strategy significantly reduces the computational complexity by utilizing the domain knowledge, e.g., the first-in-first-out principle in the same lane. As pointed out in [25], [26], a DP method can utilize domain knowledge of a specific problem to reduce the complexity by well defining the state space, state transition, and criterion function. For the merging at on-ramps problem, the DP based strategy is designed as follows:

First, the state of the DP model is constructed as a triplet, i.e., $s_i(m_i, n_i, r_i)$, where s_i denotes the state at the i^{th} stage, m_i denotes the total number of vehicles at link 1 which have been assigned right of way, n_i denotes the total number of vehicles at link 2 which have been assigned right of way, and r_i denotes the link ID of the vehicle with the right of way at this stage. By this state definition, *Markovian* property holds, and different passing orders can reach the same state, which can extremely reduce the number of states as a quadratic polynomial of the number of vehicles. It is a significant improvement compared with most existing studies, e.g., [8], [23], where the passing orders are usually directly used as the state, and the number of states increases exponentially with the number of vehicles.

Second, the state transition is well designed considering the physical constraints of vehicles in the same lane (i.e., domain knowledge). Specifically, under the assumption that lane-change behavior is not allowed in the Control Zone, vehicles of the same lane follow the first-in-first-out (FIFO) principle. It is the reason why the defined state $s_i(m_i, n_i, r_i)$ does not explicitly represent the vehicle which is assigned the right of way at the current stage. By determining the total assigned number of vehicles at the specific link, the vehicle with the right of way can be exactly specified. It significantly reduces the size of solution space, compared with methods, which use the passing order as the state and then prune the states violating the FIFO principle in the same lane.

Third, the objective function regarding traffic efficiency is established as the criterion function of the DP model to instruct decision making. The recurrence relations of criterion function are built based on *the principle of optimality*. For the states with multiple predecessor states, the criterion function is utilized to decide the optimal predecessor states. It guarantees the optimality of the passing order obtained by the DP method.

Theoretical analysis and simulations are proposed to justify and validate the proposed DP based strategy. It is proved that the proposed strategy has quadratic polynomial time complexity of the number of vehicles. It is the theoretical foundation to overcoming the limitations of existing optimal strategies and sub-optimal strategies. Moreover, the simulation results also demonstrate that the proposed strategy can reach the globally optimal passing order with much less computational time, compared with existing optimal strategies.

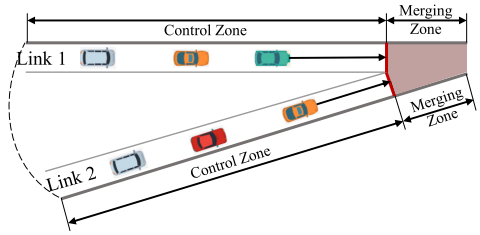


Fig. 1. Typical merging scenario.

The rest of this paper is organized as follows. Section II formulates the merging problem as a general optimization problem. Section III proposes a DP based strategy to reformulate and solve the merging problem. Section IV proves that the DP based strategy is quadratic polynomial time complexity of the number of vehicles. Then, we provide simulation results in Section V. Finally, conclusion and further works are presented in Section VI.

II. Problem Formulation

A. Scenario and Notations

A typical highway on-ramp with a single lane in each link is considered in this paper, as shown in Fig. 1. The light red area is the Merging Zone where the vehicles on different links may collide. Each link has a Control Zone where the vehicles in the Control Zone are controllable. The length of the Control Zone is defined as the distance from the entry of the Control Zone to the entry of the Merging Zone. Moreover, some reasonable assumptions are as follows:

Assumption 1: All vehicles are CAVs and controlled by the system.

Assumption 2: Only a single lane in each link is considered, and lane-change behavior is not allowed.

As presented in [6], [9], [22], [23], [27], most studies usually assume that there is no lane-change behavior in the Control Zone to simplify the problem. It is reasonable because vehicles are usually not allowed to change lanes when approaching the Merging Zone for safety consideration. Therefore, this paper also assumes that the lane-change behavior is not allowed in the Control Zone. Although the scenario is quite simple, it is the foundation for cooperative driving in more complex scenarios.

Each vehicle is given a unique identity after arriving the Control Zone, and vehicle i means the i^{th} vehicle that reaches the Control Zone. The vehicle identity sets of link 1 and link 2 are denoted by $\mathbb{M} = \{1, \ldots, m\}$ and $\mathbb{N} = \{1, \ldots, n\}$, respectively. Moreover, the main notations in this paper are shown in details in Table I.

B. Optimization Problem

Cooperative driving strategy for merging problem aims to improve traffic efficiency by optimizing the passing order of the

TABLE I
MAIN NOTATIONS IN THIS PAPER

Variables	Notations
Δ_{t_1}	The minimal allowable safe gap for avoiding the rear-end collision at the Merging Zone.
Δ_{t_2}	The maximal allowable safe gap for avoiding the converging collision at the Merging Zone.
$v_{\max}$, $v_{\min}$	The maximal and minimal velocity of vehicles.
$a_{\max}$, $a_{\min}$	The maximal and minimal acceleration of vehicles.
m	The number of vehicles on link 1 in the Control Zone.
n	The number of vehicles on link 2 in the Control Zone.
t_i^{assign}	The access time to the Merging Zone assigned to vehicle i.
k_{ij}	The binary decision variable used to formulate the passing order constrains between vehicle i and vehicle j.
$\mathbb{M}$	The vehicles identity set of link 1.
$\mathbb{N}$	The vehicles identity set of link 2.
$\mathbb{T}^{assign}$	The set of the access time assigned to all vehicles.
$\mathbb{K}$	The binary decision variable set.

vehicles. To reach this goal, we formulate the objective function as

$$J = \max t_i^{assign}, t_i^{assign} \in \mathbb{T}^{assign}. \tag{1}$$

where t_i^{assign} denotes the access time to the Merging Zone assigned to vehicle i. $\mathbb{T}^{assign}$ is the set of the access time of all vehicles in the Control Zone. Obviously, J denotes the total access time of the vehicles.

With the consideration of vehicle dynamics, the access time t_i^{assign} has a lower bound $t_i^{\min}$

$$t_i^{assign} \geq t_i^{\min}. \tag{2}$$

$$t_i^{\min} = \begin{cases} \dfrac{\sqrt{v_0^2 + 2a_{\max} \cdot x_{0,i}} - v_{0,i}}{a_{\max}} + t_0, & \text{if } \dfrac{v_{\max}^2 - v_{0,i}^2}{2a_{\max}} > x_0. \\ \dfrac{v_{\max} - v_{0,i}}{a_{\max}} + \dfrac{x_{0,i} - \frac{v_{\max}^2 - v_{0,i}^2}{2a_{\max}}}{v_{\max}} + t_0, & \text{otherwise.} \end{cases} \tag{3}$$

Meanwhile, the access time has an upper bound $t_i^{\max}$.

$$t_i^{assign} \leq t_i^{\max}. \tag{4}$$

$$t_i^{\max} = \begin{cases} \dfrac{\sqrt{v_0^2 + 2a_{\min} \cdot x_{0,i}} - v_{0,i}}{a_{\min}} + t_0, & \text{if } \dfrac{v_{\min}^2 - v_{0,i}^2}{2a_{\min}} > x_0. \\ \dfrac{v_{\min} - v_{0,i}}{a_{\min}} + \dfrac{x_{0,i} - \frac{v_{\min}^2 - v_{0,i}^2}{2a_{\min}}}{v_{\min}} + t_0, & \text{otherwise.} \end{cases} \tag{5}$$

where $x_{0,i}$ is the distance from the current location of vehicle i to the entry of the Merging Zone. $v_{0,i}$ is the velocity of vehicle i at the current time.

Suppose that vehicle i and vehicle $(i+1)$ are two consecutive vehicles at the same link and vehicle i is physically ahead of vehicle $(i+1)$. To avoid the rear-end collision at the Merging Zone, we impose the rear-end safety constraint

$$t_{i+1}^{assign} - t_i^{assign} \geq \Delta_{t_1}. \tag{6}$$

where Δ_{t_1} is the minimal allowable safe gap for avoiding the rear-end collision.

Suppose that vehicle i and vehicle j are two vehicles from different links. To avoid the converging collision at the Merging Zone, we impose the converging safety constraints

$$t_i^{assign} - t_j^{assign} \geq \Delta_{t_2} \quad \text{or} \quad t_j^{assign} - t_i^{assign} \geq \Delta_{t_2}. \tag{7}$$

where Δ_{t_2} is the minimal allowable safe gap for avoiding the converging collision. Moreover, the vehicles from different lanes need larger safe gap than the vehicles in the same lane, i.e., Δ_{t_2} is larger than Δ_{t_1}.

To formulate the whole optimization problem, some binary variables ($\mathbb{K} = \{k_{11}, \ldots, k_{ij}, \ldots, k_{mn}\}$, $(i,j) \in \mathbb{M} \times \mathbb{N}$, $\mathbb{K} \in \{0,1\}^{m \times n}$) are introducing to impose the passing order constrains

$$t_i^{assign} - t_j^{assign} + M \cdot k_{ij} \geq \Delta_{t_2}. \tag{8}$$

$$t_j^{assign} - t_i^{assign} + M \cdot (1 - k_{ij}) \geq \Delta_{t_2}. \tag{9}$$

where M is a positive and sufficiently large number. Obviously, $k_{ij} = 1$ means vehicle i can enter the Merging Zone earlier than vehicle j.

As the above descriptions, the whole optimization problem of merging problem is formulated as

$$\min_{\mathbb{T}^{assign}, \mathbb{K}} \max t_i^{assign}$$
$$\text{subject to } (2)(4)(6)(8)(9). \tag{10}$$

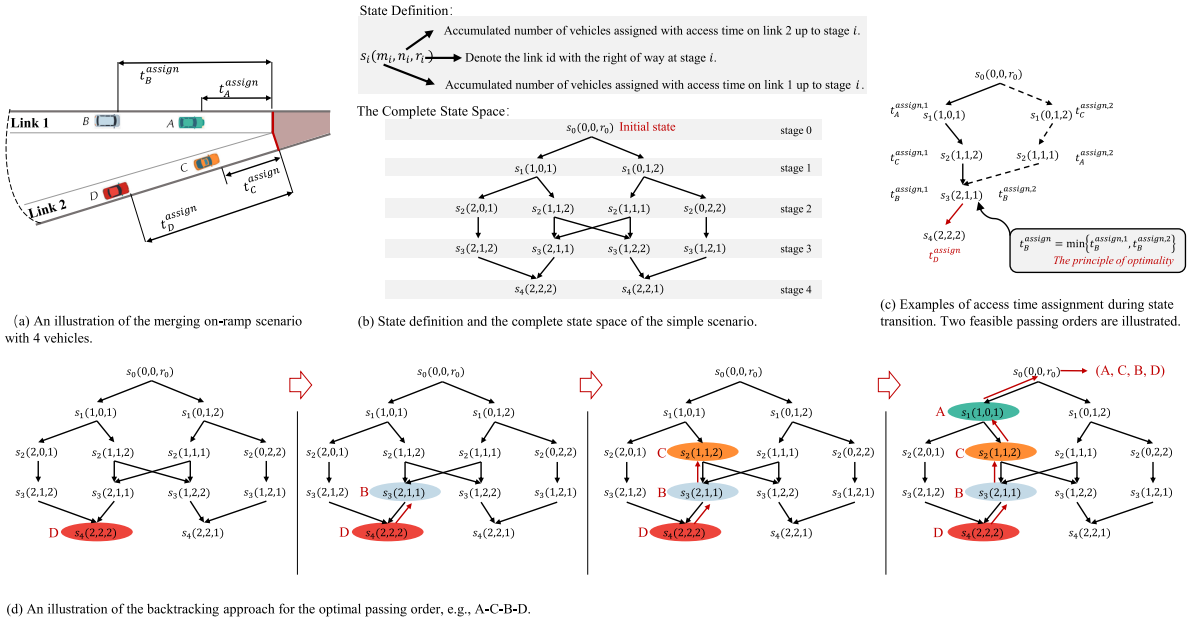
Problem (10) is a mixed-integer linear programming (MILP) problem. The branch-and-bound method can directly solve a small scale MILP problem to obtain the globally optimal solution [28]. Note here that the size of the solution space of problem (10) (i.e., the total number of the possible passing orders) is 2^{mn}. Hence the number of solutions grows exponentially with the increasing number of vehicles. To reach the globally optimal solutions based on current methods is an extremely time-consuming process.

III. DYNAMIC PROGRAMMING BASED STRATEGY

In this section, we reformulate the merging problem and propose a DP based strategy to find the globally optimal passing order with polynomial computational complexity. The rest of Section III will elaborate on the DP method step by step. For easier understanding, we take a simple example with four vehicles in the Control Zone shown in Fig. 2(a) to present the process of searching for the globally optimal solution.

A. Decision Variable

The problem of searching a passing order is equivalent to a process of assigning *the right of way* of the Merging Zone to the vehicles sequentially. Thus, problem (10) can be treated as a sequential decision problem whose *decision variable* r_i is the right of way. Obviously, the decision variable has at most two



(a) An illustration of the merging on-ramp scenario with 4 vehicles.

(b) State definition and the complete state space of the simple scenario.

(c) Examples of access time assignment during state transition. Two feasible passing orders are illustrated.

(d) An illustration of the backtracking approach for the optimal passing order, e.g., A-C-B-D.

Fig. 2. DP based cooperative driving strategy applied to a simple merging scenario with four vehicles.

possible values in a merging scenario, i.e., $r_i \in \boldsymbol{RW} = \{1, 2\}$. If $r = 1$, one vehicle at link 1 can obtain the right of way to enter the Merging Zone. Similarly, if $r = 2$, one vehicle at link 2 can get the right of way.

B. Stage Partition

Since only one vehicle can obtain the right of way at a time, the process of assigning the right of way can be particularized to a multi-stage decision process. At each stage, one vehicle in the Control Zone can get the right of way. Meanwhile, it can be assigned an access time to enter the Merging Zone.

Stage partition is used to portion the original problem as a class of similar problems to accommodate the framework of DP. In our model, $(m + n)$ stages are needed to accomplish the assignment of the right of way for all vehicles, where $(m + n)$ is the total number of vehicles in the Control Zone. Considering the initial stage, the stages can be numbered stage 0 through stage $(m + n)$. For example, since there are four vehicles in the Control Zone in Fig. 2(a), the number of stages of this scenario is five, as shown in Fig. 2(b).

To efficiently make decisions in a multi-stage decision process, we adopt the DP method which regards the original problem as a class of similar problems to find the best decisions one after another [25], [26]. In the rest of Section III, we will focus on the formation of state space, state transition and criterion function to construct the DP model.

C. State Space

The model is said to have $(m + n + 1)$ stages. Hence, the *state space* $\boldsymbol{S}$ of the DP model can be partitioned into $(m + n + 1)$ sets $\boldsymbol{S_0}$, $\boldsymbol{S_1}$, $\boldsymbol{S_2}$, …, $\boldsymbol{S_{m+n}}$.

Let us take the triplet $s_i(m_i, n_i, r_i)$ as a state, $s_i(m_i, n_i, r_i) \in \boldsymbol{S_i}$. As shown in Fig. 2(b), m_i denotes the accumulated number of vehicles at link 1 that have been assigned right of way (access time) up to stage i. Similarly, n_i denotes the accumulated number of vehicles at link 2 that have been assigned right of way (access time) up to stage i. r_i denotes the link ID of the vehicle with the right of way at stage i, $r_i \in \boldsymbol{RW}$.

The initial state is given as $s_0(0, 0, r_0)$, and the terminal state set is $\boldsymbol{S_{m+n}} = \{s_{m+n}(m, n, 1), s_{m+n}(m, n, 2)\}$. As shown in Fig. 2(b), $\boldsymbol{S_4} = \{s_4(2, 2, 1), s_4(2, 2, 2)\}$.

D. State Transition

State transition is used to describe the transition of the right of way between vehicles. The state transition equation emerges after introducing the state variable and the decision variable

$$s_i\left(m_i, n_i, r_i\right) = g\left(s_{i-1}\left(m_{i-1}, n_{i-1}, r_{i-1}\right), r_i\right). \tag{11}$$

where r_i is the decision at stage i. $g(\cdot)$ is considered as the state transition function that works as (as shown in Fig. 2(b))

i) if $r_i = 1$ and $m_{i-1} < m$, then

$$m_i = m_{i-1} + 1, n_i = n_{i-1}.$$

ii) if $r_i = 2$ and $n_{i-1} < n$, then

$$m_i = m_{i-1}, n_i = n_{i-1} + 1.$$

According to the above, the state and state transition possess following properties.

Property 1: The state of the model is indeed a summary of all past decision behavior [25]. In other words, the results of the decisions from stage 1 to stage i are embodied in the parameters (i.e., m_i, n_i, and r_i) of state $s_i(m_i, n_i, r_i)$.

Property 2: State transition only occurs between two adjacent stages.

Property 3: Different passing orders can reach the same state in the state space. As shown in Fig. 2(b), both $s_2(1, 1, 2)$ and $s_2(1, 1, 1)$ in stage 2 reach the same state, i.e., $s_3(2, 1, 1)$.

Property 4: Instead of using the state to explicitly represent the vehicle with the right of way, by determining the accumulated assigned number of vehicles at the specific link, the vehicle with the right of way can be exactly specified. Thus, the passing orders that violate the FIFO principle in the same lane are directly ignored during the state transition. In other words, the infeasible solutions are directly eliminated during the construction of solution space.

By Property 1, the state holds *Markovian* property which is the feasible conditions of DP model. By Property 2, it can extremely decrease the number of transitions of DP model. By Property 3, it can extremely decrease the number of states of DP model. By Property 4, it significantly reduces the size of solution space on the premise of ensuring optimality, and the state space just presents all feasible passing orders.

E. Criterion Function

For the states with multiple predecessor states, *criterion function* is utilized to instruct decision making to find the optimal predecessor state. In this part, we will present the formulation of the criterion function in detail.

1) Vehicle Identity Information Implied in State Variable: Note that the state variable implies the identity information of the vehicle that obtains an access time at the current stage. As shown in Fig. 2(b), $s_3(2,1,1)$ implies that the second vehicle $(m_3 = 2)$ on link 1 $(r_3 = 1)$ (i.e., the vehicle B) is assigned to pass the Merging Zone at stage 3.

Then, suppose that vehicle j and vehicle k are the vehicles that obtain an access time at stage $(i-1)$ (state $s_{i-1}(m_{i-1}, n_{i-1}, r_{i-1})$) and stage i (state $s_i(m_i, n_i, r_i)$), respectively. To minimize the maximal access time of vehicles, according to the safety constraints (2), (4), (6), (8), (9) and the state transition Equation (11), the recurrence relations of the access time of vehicle j and vehicle k can be written as

$$t_k^{assign} = \begin{cases} \min\left\{\max\left\{t_k^{\min}, t_j^{assign} + \Delta_{t_1}\right\}, t_k^{\max}\right\}, \\ \qquad\qquad \text{if } r_i = r_{i-1}. \\ \min\left\{\max\left\{t_k^{\min}, t_j^{assign} + \Delta_{t_2}\right\}, t_k^{\max}\right\}, \\ \qquad\qquad \text{if } r_i \neq r_{i-1}. \end{cases} \tag{12}$$

Generally, $t_k^{\max}$ is a very large number. Therefore, t_k^{assign} and t_j^{assign} satisfy formula (13) during the process of state transition.

$$t_k^{assign} > t_j^{assign}. \tag{13}$$

For example, $t_A^{assign} < t_C^{assign} < t_B^{assign} < t_D^{assign}$, as shown in Fig. 2(d).

2) Criterion Function Formulation: We use $MAAT_i$ to denote the maximal assigned access time from state $s_0(0,0,r_0)$ to $s_i(m_i, n_i, r_i)$. According to the objective function of problem (10), we define the criterion function of state $s_i(m_i, n_i, r_i)$ as

$$f_i\left(s_i\left(m_i, n_i, r_i\right)\right) = \min_{p_{i-1}} MAAT_i. \tag{14}$$

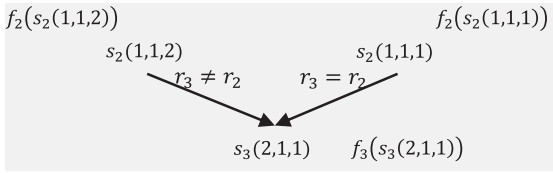


Fig. 3. Examples of making decisions during state transition.

where p_{i-1} records the path from predecessor state in $\boldsymbol{S}_{i-1}$ to state $s_i(m_i, n_i, r_i)$.

Actually, by formula (13), the value of $MAAT_i$ is equal to the access time assigned to the vehicle at stage i. Then, by formula (12), the recurrence relation between $MAAT_{i-1}$ and $MAAT_i$ is

$$MAAT_i = \begin{cases} \min\left\{\max\left\{t_k^{\min}, MAAT_{i-1} + \Delta_{t_1}\right\}, t_k^{\max}\right\}, \\ \qquad\qquad \text{if } r_i = r_{i-1}. \\ \min\left\{\max\left\{t_k^{\min}, MAAT_{i-1} + \Delta_{t_2}\right\}, t_k^{\max}\right\}, \\ \qquad\qquad \text{if } r_i \neq r_{i-1}. \end{cases} \tag{15}$$

Now, we introduce Lemma 1 to derive the recurrence relation of criterion function.

Lemma 1 (The principle of optimality [25]): if an optimal sequence of states passes through a particular state $s_{i-1}(m_{i-1}, n_{i-1}, r_{i-1})$, then the portion of the sequence from $s_0(0,0,r_0)$ to $s_{i-1}(m_{i-1}, n_{i-1}, r_{i-1})$ must be the optimal sequence from $s_0(0,0,r_0)$ to $s_{i-1}(m_{i-1}, n_{i-1}, r_{i-1})$.

Considering Lemma 1, we adopt the minimal value of $MAAT_{i-1}$ (i.e., $f_{i-1}(s_{i-1})$) to give the recurrence relation of criterion function as

$$\begin{aligned} f_i\left(s_i\left(m_i, n_i, r_i\right)\right) &= \min_{p_{i-1}} MAAT_i \\ &= \min_{p_{i-1}} \begin{cases} \min\left\{\max\left\{t_k^{\min}, f_{i-1}\left(s_{i-1}\right) + \Delta_{t_1}\right\}, t_k^{\max}\right\}, \\ \qquad\qquad \text{if } r_i = r_{i-1}. \\ \min\left\{\max\left\{t_k^{\min}, f_{i-1}\left(s_{i-1}\right) + \Delta_{t_2}\right\}, t_k^{\max}\right\}, \\ \qquad\qquad \text{if } r_i \neq r_{i-1}. \end{cases} \end{aligned} \tag{16}$$

where s_{i-1} denotes the predecessor states of $s_i(m_i, n_i, r_i)$ in $\boldsymbol{S}_{i-1}$. Meanwhile, s_{i-1} and $s_i(m_i, n_i, r_i)$ satisfy the state transition Equation (11).

By formula (16), we can find that $f_i(s_i(m_i, n_i, r_i))$ is determined by $f_{i-1}(s_{i-1})$ and r_i. Thus, we can say that the recurrence relation of criterion function follows Lemma 1. Obviously, there are at most two predecessor states of $s_i(m_i, n_i, r_i)$ in stage $(i-1)$, i.e., there are at most two possible values of $MAAT_i$ of state $s_i(m_i, n_i, r_i)$. Therefore, to solve problem (16), it is computationally efficient by comparing all values of $MAAT_i$, as shown in Fig. 2(c).

Algorithm 1: A DP Based Cooperative Driving Strategy.

Input: Locations and velocities of all vehicles

Output: The globally optimal passing order and the access time of all vehicles

1: **State space construction:** Build the complete state space from the initial state based on the state transition function, as shown in Fig. 2(b).

2: **Access time assignment:** Calculate the minimal $MAAT_i$ for all states during state transition: If there is more than one predecessor state of the current state, the criterion function is applied to decide the optimal predecessor state according to *the principle of optimality*, as shown in Fig. 2(c).

3: **Backtracking search:** Backtrack to export the optimal passing order and assign the access time to all vehicles, as shown in Fig. 2(d).

We use p^*_{i-1} to record the optimal path from the predecessor states in $\boldsymbol{S}_{i-1}$ to state $s_i(m_i, n_i, r_i)$. Let us take state $s_3(2,1,1)$ in Fig. 3 as an example to explain the process of making decisions. There are two paths from the predecessor states in $\mathbf{S}_2$ to state $s_3(2,1,1)$. The one is $s_2(1,1,2) \rightarrow s_3(2,1,1)$, the other is $s_2(1,1,1) \rightarrow s_3(2,1,1)$. Each path leads to a corresponding value of $MAAT_3$. Thus, we can obtain p^*_2 by comparing the values of $MAAT_3$.

F. Backtracking Approach

The minimal criterion function value of the terminal states in $\boldsymbol{S_{m+n}}$ is the optimal value of problem (10) based on the description above. Then, we can get the optimal path from state $s_{m+n}(m,n,r_i)$ to state $s_0(0,0,r_0)$ based on the sequence $(p^*_{m+n-1}, p^*_{m+n-2}, \ldots, p^*_0)$ via backtracking approach, as shown in Fig. 2(d). Finally, the optimal passing order and the access time of vehicles are exported naturally.

As an example shown in Fig. 2(d), suppose that the criterion function value of $s_4(2,2,2)$ is the minimal one in stage 4. Then, the optimal path from $s_4(2,2,2)$ to $s_0(0,0,r_0)$ based on $(p^*_3, p^*_2, p^*_1, p^*_0)$ will be acquired via backtracking. Finally, the corresponding optimal passing order (A-C-B-D) and the access time of vehicles will be exported. The DP based cooperative driving strategy is summarized as **Algorithm 1**.

IV. Analysis of Computational Time Complexity

The principle of optimality is adopted in decision-making process, which ensures that the passing order obtained by the proposed strategy is the globally optimal one. In this section, theoretical analysis of the computational time complexity will be presented. Recall that the number of vehicles on link 1 and link 2 are denoted by m and n, respectively.

A. Analysis of the Size of the DP Model

Two theorems of the size of the DP model are proposed.

Theorem 1: The number of the states of the DP model is $(2\,\mathrm{mn} + m + n + 1)$.

Theorem 2: The number of the transitions of the DP model is $(4\,\mathrm{mn})$.

Proof: See Appendix A. ■

By Theorem 1 and Theorem 2, the number of states and the number of transitions of the DP model are quadratic polynomial with the number of the vehicles. Thus, the size of the DP model grows slowly as the number of vehicles increasing.

B. Analysis of Computational Time Complexity

The theorem of the computational time complexity of the DP based strategy is proposed.

Theorem 3: The computational time complexity of the DP based cooperative driving strategy is $O(\mathrm{mn})$.

Proof: See Appendix B. ■

By Theorem 3, DP based cooperative driving strategy can obtain the globally optimal passing order with quadratic polynomial time complexity of the number of vehicles, i.e., the computational time complexity is $O(N^2)$, where N denotes the number of vehicles.

V. Simulations

In this section, we will further validate the performance of the DP based strategy via exploring some comparison simulations. The proposed strategy is compared with an existing optimal strategy and a sub-optimal strategy as classified in Section I. As for the existing optimal strategy, problem (10) is directly solved by the CVX software with Mosek solver in the simulations. Thus, the access time for all vehicles can be exported directly. As for the sub-optimal strategy, we select the ad hoc negotiation based strategy as the benchmark, which instructs the vehicles to pass through the conflict zone roughly in FIFO order. Thus, the recurrence relations of the access time for the ad hoc negotiation based strategy presents as follows:

$$t_i^{assign} = \begin{cases} t_i^{\min}, & \text{if } i = 1. \\ \min\left\{\max\left\{t_i^{\min}, t_{i-1}^{assign} + \Delta\right\}, t_i^{\max}\right\}, & \text{if } i > 1. \end{cases} \tag{17}$$

where if vehicle i and vehicle $(i-1)$ are driving on the same lane, $\Delta = \Delta_{t_1}$. Otherwise, $\Delta = \Delta_{t_2}$.

There are three evaluation indicators, namely *the total passing time*, *the traffic throughput* and *the computation time*. The total passing time is the access time to the Merging Zone of the last vehicle in the Control Zone, i.e., the objective function presented in formula (1). The traffic throughput is the total number of vehicles that have entered the Merging Zone in a specified period of time. In addition, the computation time denotes the time used to obtain the solution in one-time optimization process.

The first simulation aims to illustrate that the total passing time obtained by the proposed strategy is exactly equal to that of the existing optimal strategy (the total passing time obtained by the existing optimal strategy is the globally optimal one), and the computation time of the proposed strategy is much less than the existing optimal strategy. The second simulation explores the comparison experiments of different cooperative driving

TABLE II
PARAMETERS SETTING IN THE SIMULATIONS

Parameters	Simulation setting
The length of Control Zone L	250 m
Δ_{t_1}	1.5 s
Δ_{t_2}	2 s
$v_{\max}$, $v_{\min}$	$15, 0$ (m/s)
$a_{\max}$, $a_{\min}$	$3, -5$ (m/s^2)

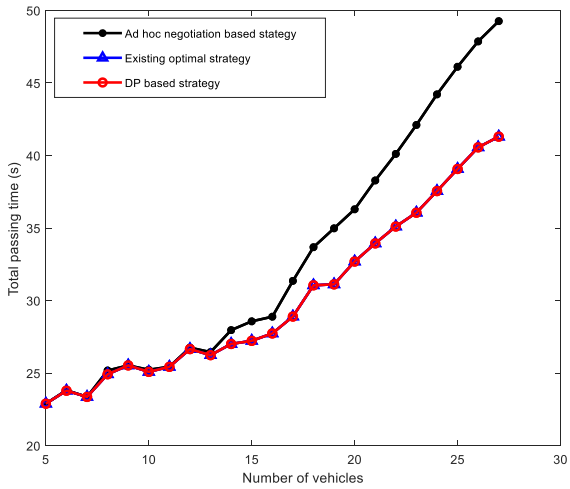


Fig. 4. Total passing time with respect to the number of vehicles.

strategies respect to the arrival flow rate in the continuous traffic process.

The parameters utilized in the simulations are set as in Table II, and the initial velocity of the vehicle is the uniform distribution of $[v_{\min}, v_{\max}]$. All simulations are carried out on MATLAB platform in a personal computer with an i7 CPU and an 8 GB RAM.

A. The Optimality of the DP Based Strategy

To evaluate the optimality and the computational efficiency of the DP based strategy, we design a merging scenario with c vehicles ($c \in [5, 27]$) that randomly distributed in the Control Zone. Then, three different cooperative driving strategies are applied to this merging problem to get the corresponding passing orders. For each c, we repeat the simulation 20 times to take the average total passing time and the average computation time.

As shown in Fig. 4 and Fig. 5, the simulation results indicate that the total passing time of the DP based strategy is exactly equal to that of the existing optimal strategy, and the total passing time of the DP based strategy is significantly decreased compared to the ad hoc negotiation based strategy when the number of vehicles is larger than 15. Moreover, the computation time of the DP based strategy is close to that of the ad hoc negotiation based strategy, while the average computation time of the existing optimal strategy grows almost exponentially with the increasing number of vehicles.

Consequently, we can conclude that the DP based strategy can find the optimal solution with short enough computation time for merging problem.

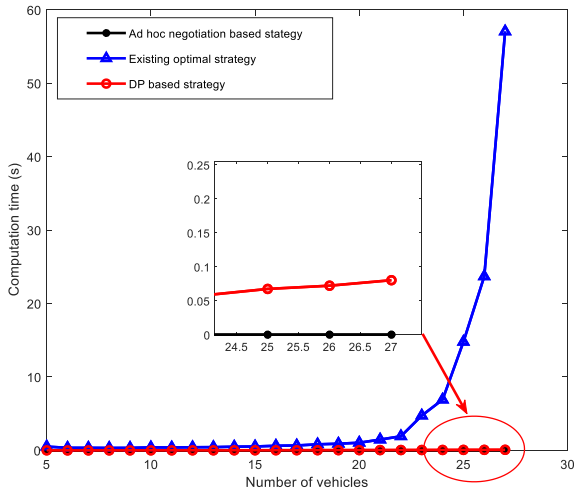


Fig. 5. Computational time with respect to the number of vehicles.

B. To Evaluate the Performance of the DP Based Strategy in Continuous Traffic Process

In this simulation, the cooperative driving strategies are applied to continuous traffic scenarios. The passing order is replanned when a new vehicle enters the Control Zone. We assume that the vehicles arrive in a Poisson Process at each link [4], [22], [29], and the arrival rate λ of vehicles varies from $0.1veh/(lane \cdot s)$ to $0.33veh/(lane \cdot s)$. For each arrival rate, we simulate a 10-minutes traffic process (supposed that two links possess the same arrival rate).

As pointed out in [4], traffic efficiency is mainly decided by the passing order of vehicles, and different motion planning methods, which control the vehicle to reach the Merging Zone at the assigned access time, have a similar influence on traffic efficiency. Thus, for simplicity, we adopt a simple motion planning method to simulate the process of vehicle movement, referring to Appendix C for details [22]. In addition, there are also several other motion planning methods can be applied in the simulations, e.g., the method introduced in [27].

As shown in Table III and Fig. 6, the difference in the traffic throughput of the DP based strategy and the ad hoc negotiation based strategy increases with the increasing arrival rate. For instance, when the arrival rate is $0.33veh/(lane \cdot s)$, the traffic throughput of the DP based strategy is about 20% higher than the ad hoc negotiation based strategy. Also, the traffic throughput of the ad hoc negotiation based strategy reaches a saturation state as the arrival rate increasing. Moreover, the computation time of the DP based strategy is below 100 ms in all scenarios of the simulations, as shown in Table III. In other words, the DP based strategy exactly meets the real-time requirement when applied to the real traffic scenario. Thus, it can be concluded that the coordination performance of the DP based strategy is extremely better than the ad hoc negotiation based strategy in the continuous traffic process.

Considering the comparison results of the DP based strategy and the existing optimal strategy, we can find that the traffic throughput of the DP based strategy is nearly equal to the existing optimal strategy. However, the computation time of the existing optimal strategy grows sharply as the average arrival

TABLE III
COMPARISON RESULTS OF DIFFERENT COOPERATIVE DRIVING STRATEGIES

Arrival rate $(veh/(lane \cdot s))$	Cooperative driving strategies	Traffic throughput	Average computation time (ms)	Average number of vehicles [1]
0.1	Ad hoc negotiation based strategy	119	0.3	4.4
	Existing optimal strategy	119	269.1	5.1
	Dynamic programming based strategy	119	1.5	4.5
0.2	Ad hoc negotiation based strategy	244	0.2	8.6
	Existing optimal strategy	241	362.8	9.3
	Dynamic programming based strategy	243	4.8	8.5
0.25	Ad hoc negotiation based strategy	301	0.2	11.7
	Existing optimal strategy	300	446.7	11.1
	Dynamic programming based strategy	301	8.2	10.7
0.28	Ad hoc negotiation based strategy	332	0.2	20.1
	Existing optimal strategy	353	965.5	14.6
	Dynamic programming based strategy	354	15.5	14.5
0.3	Ad hoc negotiation based strategy	329	0.3	23.7
	Existing optimal strategy	365	3077.5	15.7
	Dynamic programming based strategy	365	19.5	15.4
0.33	Ad hoc negotiation based strategy	328	0.3	24.1
	Existing optimal strategy	403	25502.7	21.1
	Dynamic programming based strategy	402	42.5	21.8

[1] Average number of vehicles: the average number of vehicles in the Control Zone.

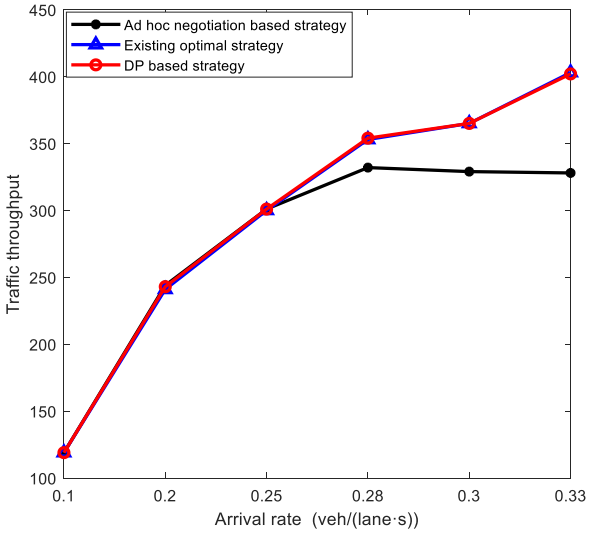


Fig. 6. Traffic throughput of different strategies with respect to the arrival rate in a 10-minutes traffic process.

rate increasing. Therefore, in terms of traffic efficiency, the performance of the DP based strategy is almost the same as the existing optimal strategy in the continuous traffic scenario. In terms of computational complexity, the DP based strategy can complete the optimization of passing order within a short enough time.

Note here that there are a few differences in the traffic throughput of the DP based strategy and the existing optimal strategy. Actually, only the vehicles in the Control Zone will be considered into the optimization of passing order at each time period, and the optimal total passing time may correspond to several different passing orders. Thus, for the same specific merging scenario, both the DP based strategy and the existing optimal strategy can obtain the optimal total passing time and an optimal passing order, as illustrated in Section V-A. However, for the continuous traffic scenario, the optimization of passing order is a rolling optimization process. We can guarantee that the passing order derived by the cooperative driving strategies is an optimal one at current optimization process, but the influence of the current passing order on the next time interval has not been evaluated. Therefore, we can say that the passing order derived by the DP based strategy or the existing optimal strategy is a *sub-optimal* passing order for a continuous traffic scenario. Also, it is hard to say which strategy getting a better passing order in a rolling optimization process for the continuous traffic process.

VI. CONCLUSION

In this paper, we propose a cooperative driving strategy for merging at on-ramps based on DP. The key idea is to consider the domain knowledge of the merging problem into the DP method to reduce the complexity by well defining the state space, state transition and criterion function. The passing order obtained by the proposed strategy is proved the globally optimal solution. It is also proved that the DP based strategy has quadratic polynomial time complexity of the number of vehicles, i.e., $O(N^2)$. It demonstrates that the proposed strategy is globally optimal yet computationally efficient, which can overcome the limitations of existing optimal strategies and sub-optimal strategies. Furthermore, the simulation results further validate that the DP based strategy can find the globally passing order efficiently.

Moreover, some interesting and critical research topics will be further studied in the near future. First, to relax the assumption of pure connected and automated vehicles (CAVs), the proposed strategy will be extended to a mixed traffic scenario by

predicting the movements of human-driven vehicles [30], [31]. Second, to relax the assumption of prohibiting lane changing, the scenario with lane-change cases will be considered into the cooperative driving problem. Third, the real-world scenario is an infinite-horizon problem. How to improve the performance of the proposed strategy for the infinite-horizon scenario also deserves to be further studied.

APPENDIX A
PROOF OF THEOREM 1–2

Without losing generality, we suppose that $m > n$, where m and n denote the number of vehicles on link 1 and link 2, respectively. Recall that the stage number is denoted by i, $i \in \{i|0 \le i \le m+n, i \in \boldsymbol{Z}\}$, and a state in stage i is denoted by $s_i(m_i, n_i, r_i)$, $s_i(m_i, n_i, r_i) \in \boldsymbol{S_i}$. Also, we have $i = m_i + n_i$, $m_i \le m$ and $n_i \le n$ by the definition of the state space of the DP model.

Proof. (Theorem 1): The number of states varies in different stages, so we will analyze the number of states by categories, which are classified by stage number.

1) *Category 1:* $i = 0$
There is only 1 state (the initial state $s_0(0, 0, r_0)$) in stage 0, i.e.,
$$N_1 = 1. \tag{18}$$

2) *Category 2:* $0 < i \le n$
In this category, the state $s_i(m_i, n_i, r_i)$ satisfies the following formulas:
$$0 \le m_i \le \min\{i, m\} = i. \tag{19a}$$
$$0 \le n_i \le \min\{i, n\} = i. \tag{19b}$$
$$m_i + n_i = i. \tag{19c}$$

i) *Case 1:* $m_i = 0, n_i = i, r_i = 2$
There is only 1 state (i.e., state $s_i(0, i, 2)$) satisfying Case 1 in stage i.
ii) *Case 2:* $m_i = i, n_i = 0, r_i = 1$
There is only 1 state (i.e., state $s_i(i, 0, 1)$) satisfying Case 2 in stage i.
iii) *Case 3:* $0 < m_i < i, 0 < n_i < i$ and $r_i = 1$ or 2
According to 19, we can obtain that there are $2 \times (i-1)$ states satisfying Case 3 in stage i.

Therefore, according to Case 1, Case 2 and Case 3, the number of states in stage i is $2 \times (i-1) + 2$. Then, the number of states satisfying Category 2 in the DP model is
$$N_2 = \sum_{i=1}^{n} [2 \times (i-1) + 2]. \tag{20}$$

3) *Category 3:* $n < i \le m$
In this category, the state $s_i(m_i, n_i, r_i)$ satisfies the following formulas:
$$0 \le m_i \le \min\{i, m\} = i. \tag{21a}$$
$$0 \le n_i \le \min\{i, n\} = n. \tag{21b}$$
$$m_i + n_i = i. \tag{21c}$$

Then, it can be obtained that m_i and n_i satisfy the following formulas:
$$i - n \le m_i \le i. \tag{22a}$$
$$0 \le n_i \le n. \tag{22b}$$

i) *Case 1:* $m_i = i, n_i = 0, r_i = 1$
There is only 1 state (i.e., state $s_i(i, 0, 1)$) satisfying Case 1 in stage i.
ii) *Case 2:* $i - n \le m_i < i, 0 < n_i \le n$ and $r_i = 1$ or 2
According to (21)–(22), we can obtain that there are $2n$ states satisfying Case 2 in stage i.

Therefore, according to Case 1 and Case 2, the number of states in stage i is $2n + 1$. Then, the number of states satisfying Category 3 in the DP model is
$$N_3 = \sum_{i=n+1}^{m} (2n + 1). \tag{23}$$

4) *Category 4:* $i > m$
In this category, the state $s_i(m_i, n_i, r_i)$ satisfies the following formulas:
$$0 \le m_i \le \min\{i, m\} = m. \tag{24a}$$
$$0 \le n_i \le \min\{i, n\} = n. \tag{24b}$$
$$m_i + n_i = i. \tag{24c}$$

Then, it can be obtained that m_i and n_i satisfy the following formulas:
$$i - n \le m_i \le m. \tag{25a}$$
$$0 \le n_i \le \min\{i, n\} = n. \tag{25b}$$
$$i - m \le n_i \le n. \tag{25c}$$

According to (24)–(25), we can obtain that there are $2 \times (m + n - i + 1)$ in stage i.

Therefore, the number of states satisfying Category 4 in the DP model is
$$N_4 = \sum_{i=m+1}^{m+n} 2 \times (m + n - i + 1). \tag{26}$$

Therefore, based on (18), (20), (23) and (26), we can conclude that the total number of states in the DP model is
$$N_{\text{state}} = N_1 + N_2 + N_3 + N_4 = 2mn + m + n + 1. \tag{27}$$

Obviously, (27) is a complete symmetry formula. Therefore, formula (27) still holds when $m \le n$. ■

Proof. (Theorem 2): According to the state transition Equation (11), each state in stage i has two transitions to expand to stage $(i+1)$ except the state $s_i(m, i-m, r_i)$ or $s_i(i-n, n, r_i)$, which has only one transitions. We can also adopt the classified method to obtain the number of transitions of the DP model.

1) *Category 1:* $i = 0$
According to formula (18), the state $s_i(m, i-m, r_i)$ or $s_i(i-n, n, r_i)$ does not exist in stage 0. Thus, the number of transitions in stage 0 is
$$N_1' = 2N_1 = 2. \tag{28}$$

2) *Category 2:* $0 < i < n$
According to (20), the state $s_i(m, i-m, r_i)$ or $s_i(i-n, n, r_i)$ does not exist in stage i.
Thus, the number of transitions satisfying Category 2 in the DP model is

$$N_2' = 2N_2 = \sum_{i=1}^{n-1}[4 \times (i-1) + 4]. \tag{29}$$

3) *Category 3:* $i = n$
According to (20), there are $2 \times (i-1) + 2$ states in stage n including the state $s_n(0, n, 2)$.
Thus, the number of transitions satisfying Category 3 in the DP model is

$$N_3' = 2 \times [2 \times (n-1) + 1] + 1 = 4 \times (n-1) + 3. \tag{30}$$

4) *Category 4:* $n < i < m$
According to (23), there are $2n+1$ states in stage i including the states $s_i(i-n, n, 1)$ and $s_i(i-n, n, 2)$.
Thus, the number of transitions satisfying Category 4 in the DP model is

$$N_4' = \sum_{i=n+1}^{m-1} [2 \times (2n-1) + 2 \times 1] = \sum_{i=n+1}^{m-1} 4n. \tag{31}$$

5) *Category 5:* $i = m$
According to (23), there are $2n+1$ states in stage m including the states $s_m(m, 0, 1)$, $s_m(m-n, n, 1)$ and $s_m(m-n, n, 2)$.
Thus, the number of transitions satisfying Category 5 in the DP model is

$$N_5' = 2 \times (2n + 1 - 3) + 3 = 4n - 1. \tag{32}$$

6) *Category 6:* $m < i < m+n$
According to (26), there are $2 \times (m+n-i+1)$ states in stage i including the states $s_i(m, i-m, 1)$, $s_i(m, i-m, 2)$, $s_i(i-n, n, 1)$ and $s_i(i-n, n, 2)$.
Thus, the number of transitions satisfying Category 6 in the DP model is

$$N_6' = \sum_{i=m+1}^{m+n-1} \{2 \times [2 \times (m+n-i+1) - 4] + 4\}. \tag{33}$$

7) *Category 7:* $i = m+n$
Obviously, there is no transition expanding to next stage in stage $m+n$, i.e.,

$$N_7' = 0. \tag{34}$$

Therefore, based on (28)–(34), we can conclude that the total number of transitions of the DP model is

$$N_{\text{transition}} = N_1' + N_2' + N_3' + N_4' + N_5' + N_6' + N_7' = 4\,mn. \tag{35}$$

Obviously, (35) is a complete symmetry formula. Therefore, formula (35) still holds when $m \le n$. ■

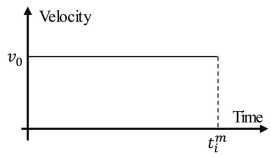
APPENDIX B
PROOF OF THEOREM 3

Proof. (Theorem 3): The computations of the DP based cooperative driving strategy proposed in this paper mainly include two terms: the one is the process of state transition, the other is the evaluation of the criterion function value for each state. The computational time of each computation is a constant time, which does not change with the size of the input of the algorithm (i.e., the total number of the vehicles). Therefore, the total number of computations is about $(N_{\text{transition}} + N_{\text{state}})$, and the computational time complexity of the DP based cooperative driving strategy is $O(mn)$. ■

APPENDIX C
A SIMPLE MOTION PLANNING METHOD

A simple motion planning method is utilized in the simulations. Several modes of vehicle movement are shown as following [22]:

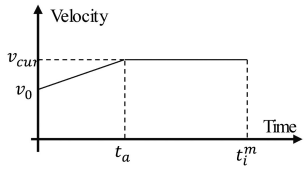
Mode 1: Keeping a constant velocity.



$$a_i = 0. \tag{36a}$$

$$x_{0,i} = v_{0,i} \cdot t_i^m. \tag{36b}$$

Mode 2: Accelerating to a cruising velocity, then driving at the cruising speed.



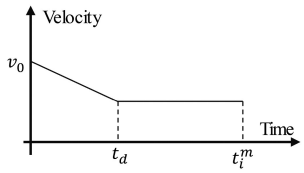
$$a_i = a_{\max}. \tag{37a}$$

$$t_a = t_i^m - \Delta T. \tag{37b}$$

$$v_{cur} = v_{0,i} + a_i \cdot t_a. \tag{37c}$$

where t_a denotes the accelerating time.

Mode 3: Decelerating to a cruising velocity, then driving at the cruising speed.



$$a_i = a_{\min}. \tag{38a}$$

$$t_a = t_i^m + \Delta T. \tag{38b}$$

$$v_{cur} = v_{0,i} + a_i \cdot t_d. \tag{38c}$$

where t_d denotes the decelerating time.
where $\Delta T = \frac{\sqrt{(a_i \cdot t_i^m)^2 + 2a_i(v_0 \cdot t_i^m - x_{0,i})}}{a_i}$.

REFERENCES

[1] J. A. Misener and S. E. Shladover, "Path investigations in vehicle-roadside cooperation and safety: A foundation for safety and vehicle-infrastructure integration research," in *Proc. IEEE Intell. Transp. Syst. Conf.*, 2006, pp. 9–16.

[2] R. Ke, Z. Zeng, Z. Pu, and Y. Wang, "New framework for automatic identification and quantification of freeway bottlenecks based on wavelet analysis," *J. Transp. Eng., Part A: Syst.*, vol. 144, no. 9, 2018, Art. no. 04 018 044.

[3] S. Tsugawa, "Inter-vehicle communications and their applications to intelligent vehicles: An overview," in *Proc. Intell. Vehicle Symp.*, vol. 2, 2002, pp. 564–569.

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

[5] S. Feng, H. Sun, Y. Zhang, J. Zheng, H. X. Liu, and L. Li, "Tube-based discrete controller design for vehicle platoons subject to disturbances and saturation constraints," *IEEE Trans. Control Syst. Technol.*, to be published, doi: 10.1109/TCST.2019.2896539.

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

[7] R. F. Atallah, C. M. Assi, and J. Y. Yu, "A reinforcement learning technique for optimizing downlink scheduling in an energy-limited vehicular network," *IEEE Trans. Veh. Technol.*, vol. 66, no. 6, pp. 4592–4601, Jun. 2017.

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

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

[10] Y. Bi, L. X. Cai, X. Shen, and Z. Hai, "Efficient and reliable broadcast in intervehicle communication networks: A cross-layer approach," *IEEE Trans. Veh. Technol.*, vol. 59, no. 5, pp. 2404–2417, Jun. 2010.

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

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

[13] 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. Part B*, vol. 112, pp. 89–112, 2018.

[14] Y. Feng, C. Yu, and H. X. Liu, "Spatiotemporal intersection control in a connected and automated vehicle environment," *Transp. Res. Part C Emerg. Technologies*, vol. 89, pp. 364–383, 2018.

[15] 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. Part B Methodological*, vol. 105, pp. 479–506, 2017.

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

[17] 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.

[18] K. M. Dresner and P. Stone, "Multiagent traffic management: A reservation-based intersection control mechanism," in *Proc. 3rd Int. Joint Conf. Auton. Agents Multiagent Syst.*, 2004, pp. 530–537.

[19] K. M. Dresner and P. Stone, "A multiagent approach to autonomous intersection management," *J. Artif. Intell. Res.*, vol. 31, no. 3, pp. 591–656, 2008.

[20] 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.

[21] 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.*, 2018, pp. 456–460.

[22] 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.

[23] H. Xu, Y. Zhang, L. Li, and W. Li, "Cooperative driving at unsignalized intersection using tree search," *IEEE Trans. Intell. Transp. Syst.*, submitted for publication.

[24] Z. Li *et al.*, "Temporal-spatial dimension extension-based intersection control formulation for connected and autonomous vehicle systems," *Transp. Res. Part C: Emerg. Technol.*, vol. 104, pp. 234–248, 2019.

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

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

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

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

[29] C. Liu, C. W. Lin, S. Shiraishi, and M. Tomizuka, "Distributed conflict resolution for connected autonomous vehicles," *IEEE Trans. Intell. Vehicles*, vol. 3, no. 1, pp. 18–29, Mar. 2018.

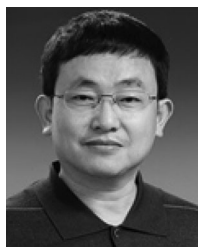
[30] S. Gong and L. Du, "Cooperative platoon control for a mixed traffic flow including human drive vehicles and connected and autonomous vehicles," *Transp. Res. Part B: Methodological*, vol. 116, pp. 25–61, 2018.

[31] F. Li and Y. Wang, "Cooperative adaptive cruise control for string stable mixed traffic: Benchmark and human-centered design," *IEEE Trans. Intell. Transp. Syst.*, vol. 18, no. 12, pp. 3473–3485, Dec. 2017.

Huaxin Pei received the B.S. degree from the University of Electronic Science and Technology of China, Chengdu, China, in 2017. He is currently working toward the Ph.D. degree with the Department of Automation, Tsinghua University, Beijing, China. His current research interests include cooperative driving and intelligent vehicles.

Shuo Feng received the B.S. degree from the Department of Automation, Tsinghua University, Beijing, China, in 2014. He is currently working toward the Ph.D. degree with the Department of Automation, Tsinghua University, China. He is also a Visiting Ph.D. Student with the Department of Civil and Environmental Engineering, University of Michigan, Ann Arbor, MI, USA. His current research interests include optimal control, connected and automated vehicle evaluation, and transportation data analysis.

Yi Zhang (S′01–M′04) received the B.S. and M.S. degrees, from Tsinghua University, Beijing, in China, in 1986 and 1988, respectively, and the Ph.D. degree from the University of Strathclyde, Glasgow, in the UK, in 1995. He is a Professor in Control Science and Engineering with Tsinghua University. His current research interests focuses 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, systems engineering, etc.



Danya Yao (M′04) received the B.S., M.S., and Ph.D. degrees from Tsinghua University, Beijing, China, in 1988, 1990, and 1994, respectively. He is a Full Professor with the Department of Automation, Tsinghua University. His research interests include intelligent detection technology, system engineering, mixed traffic flow theory, and intelligent transportation systems.