Intersection Traffic Management with Lane Advising and Optimized Scheduling

Michael I.-C. Wang*, Jianyu Pang‖, Charles H.-P. Wen†, Yang Xu‡, H. Jonathan Chao§
*†National Chiao Tung University, ‡Fudan University, *‖§New York University
Email: {*icw238, ‖qz761, §chao}@nyu.edu, ‡xuy@fudan.edu.cn, †opwen@g2.nctu.edu.tw

***Abstract*—Every year, millions of vehicle crashes occur in the world. One of the measures that have been taken to reduce car accidents is to use automated driving and manage vehicles at intersections using sophisticated scheduling algorithms through Vehicle-to-Everything (V2X) communications without relying on traffic signals. In this paper, we propose a novel intersection management system, called Roadrunner, which consists of a vehicle scheduler and a lane adviser. The former determines a time for every vehicle entering the intersection without causing any collision with other vehicles. The latter advises every vehicle which lane to stay on to pass the intersection with a possible shortest delay. Vehicles when approaching the intersection are grouped into batches and are optimized for minimizing their travel delay through the intersection. Our simulation studies show that Roadrunner outperforms existing scheduling schemes by at least 73.74% in throughput and delay under uniform traffic. Under different traffic patterns and different acceptance rates of lane advising, the lane adviser helps improve throughput by up to 50.61%.**

***Index Terms*—intersection management, lane advise, vehicle scheduling**

I. Introduction

EVERY year, millions of motor vehicle crashes occur on US road. In 2015, more than six million vehicle crashes occurred in the US. Among them, 5,326,000 vehicles were involved in intersection-related crashes [1] and 45% of the fatal traffic crashes occurred in urban areas [2]. While various factors may lead to the car accidents, driver carelessness is considered to be the main cause. Hence, automated driving has been proposed and studied for decades to reduce the casualty.

Due to space limitation in urban areas [3], traffic management and operation of intelligent transportation system is particularly important in metropolitans and has recently motivated much research in this area [4]. Some researchers focus on traffic signal control system with either on-line optimization or off-line strategies for signal timing control [3]. Others focus on developing an automated driving environment to reduce accidents, while enhancing the intersection management efficiency for a higher vehicular throughput. With emerging connected vehicle technologies facilitated by the 5G wireless and Dedicated Short Range Communications (DSRC), traffic signals may be totally removed and replaced with automated vehicle driving where vehicles can communicate with roadside infrastructures and collaborate with other vehicles to react accordingly. More specifically, Vehicle-to-Everything (V2X) communication enables the connected vehicles to share Vehicle-to-Infrastructure (V2I) and Vehicle-to-Vehicle (V2V) information among them [5], where connected vehicles among them are able to communicate with a very low latency.

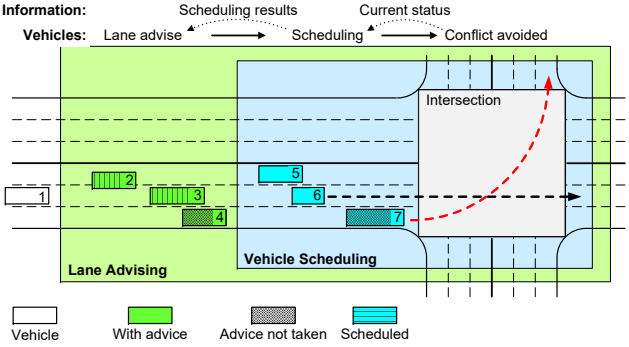


Fig. 1. The overview of Roadrunner. Roadrunner is consist of two main stages: lane advising and vehicle scheduling.

Several reservation-based intersection management systems [6]–[8] have been proposed for collision avoidance of automated vehicles. While being computationally efficient, they can only achieve sub-optimal throughput performance [9]. Hence, several optimization-based intersection management systems [10]–[12] have been proposed to achieve better throughput performance. ICACC, proposed by Zohdy and Rakha [12], leverages mixed-integer linear programming (MILP) to minimize traveling delay at the intersection and the fuel consumption. However, ICACC does not consider various vehicle sizes, nor provides the flexibility of letting vehicles on any lane to turn on any direction. More severely, a so-called ***"resource locking"*** was discovered in their MILP formulation. In ICACC, vehicles are served in batches. Before a batch of vehicles can enter the intersection, the vehicles in the previous batch have to completely pass through the intersection, which may cause unnecessary gaps between the vehicles in the two batches. In other words, the previous batch may unnecessarily "lock up" the entire intersection until all vehicles in the previous batch have passed through the intersection. We have hence improved the performance by eliminating the "lock-up" situation (to be explained later in the paper).

The intersection throughput can not only be enhanced by effectively scheduling vehicles, but also by lane control (i.e, advising vehicles to change to a particular lane). Researchers [13]–[15] have shown that lane assignment or lane change can improve road performance. However, these studies mainly focus on motorways rather than the intersections. Moreover, these studies estimate the traffic load with prediction models,

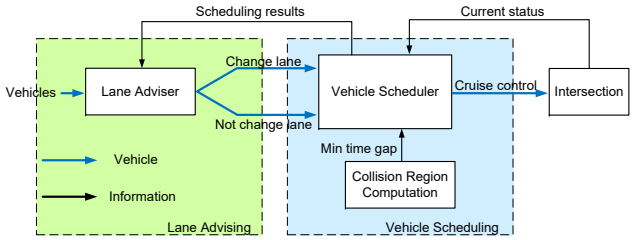


Fig. 2. System structure of Roadrunner. Roadrunner is designed with a pipeline structure, where vehicles move from stage to stage, so as information is feedback from stage to stage.

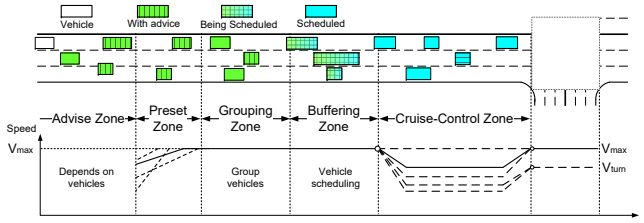


Fig. 3. Zones in Roadrunner. Each zone has its own function for vehicle scheduling.

while, in fact, it can be precisely predicted by vehicle scheduling.

Here, we propose a novel intersection management system, called ***Roadrunner***. It has a pipeline structure with two stages of control: (1) lane advising and (2) vehicle scheduling. In the first stage, vehicles are advised to stay on (or change to) a particular lane to reduce the delay for passing the intersection. While "lane advising" gives a better throughput/delay performance, vehicles may not take the advice due to physical limitations. In other words, lane changing is not mandatory, but voluntary. In the second stage, Roadrunner schedules the time for each vehicle entering the intersection by controlling their speed (i.e., cruise control).

Fig. 1 shows the two stages of the control of Roadrunner. When a vehicle (Vehicle 1) approaches the intersection, it first enters the first stage: lane advising, where Roadrunner gives lane advice to each vehicle (Vehicle 2, 3, 4) using FCFS (first come first serve) policy so that vehicles can take lane advice as early as possible. As a result, vehicles may change to a particular lane for a better performance. For example, Vehicle 2 changes to the inner lane, while Vehicle 3 stays at the middle lane. Vehicle 4 is advised to move to the middle lane, but it is blocked by Vehicle 3. Once vehicles enter the second stage: vehicle scheduling, they are forbidden to change their lanes. In this stage, Roadrunner schedules the vehicles (Vehicle 5, 6, 7) with MILP, where the collision avoidance is formulated as constraints, and the objective is to minimize the average traveling delay. For vehicles that cannot change their lanes in time (e.g., Vehicle 7), Roadrunner can also minimize their traveling delays in the second stage.

In our experiments, we show that Roadrunner can achieve 73.74% and 72.73% higher throughput than the ones with the FCFS reservation-based algorithm and the optimization-based algorithm (ICACC). Compared to traditional traffic signals, Roadrunner increases the throughput by 130.84%. Also, we show that the lane advising in Roadrunner improves up to 50.61% throughput compared to Roadrunner without lane advising. In the experiments, several traffic patterns and traffic loads are taken into account to show that Roadrunner can perform well in different conditions.

There are three main contribution of this paper.

- We improve the optimization-based vehicle scheduling by resolving the resource locking issue in ICACC.
- We propose a novel architecture of intersection management, Roadrunner, integrating lane advising and vehicle scheduling, and show that it outperforms other proposed schemes.
- We construct a simulation environment to evaluate the performance of Roadrunner under different traffic conditions to prove its robustness and effectiveness.

The paper is organized as follows. In Sec. II, we introduce the system structure of Roadrunner and describe the detailed design of vehicle scheduler and lane adviser in Sec. III and Section. IV, respectively. In Sec. V, we evaluate the performance of Roadrunner with different experiments. Sec. VI presents the related works and finally Sec. VII concludes the paper.

II. Structure of The Proposed Roadrunner

In this section, we will introduce the system structure of Roadrunner as shown in Fig. 2 and illustrate how it can adapt to limited computation resource. Roadrunner consists of two components: lane adviser and vehicle scheduler. When vehicles approach the intersection (following the blue arrows), they first receive advice from the lane adviser in the first stage. Then, Roadrunner schedules these vehicles at the second stage. While the lane adviser and the vehicle scheduler are busy on their execution, vehicles continue moving toward the intersection. To achieve high efficiency, different zones are used to regulate the interaction between the vehicles and Roadrunner. These zones are arranged in a **pipeline structure**, and are explained in Subsec. II-A. Also, in Subsec. II-B, we will first describe the pipeline structure and then show how the length of the zones can be modified to adapt to the limited computation resource and the limited length of road.

A. Zones in Roadrunner

The roads close to each intersection are divided into five zones: (1) Advise Zone (A-Zone), (2) Preset Zone (P-Zone), (3) Grouping Zone (G-Zone), (4) Buffering Zone (B-Zone), and (5) Curse-Control Zone (CC-Zone), as shown in Fig. 3, with the corresponding speeds in each zone. It is worthy to mention that the control of the speed according to the pattern as shown in the figure can reduce fuel consumption [12]. These zones are explained in detail below:

1) Advise Zone (A-Zone): When a vehicle arrives at the A-Zone, it will first receive a lane advice signal from the lane adviser with a suggestion to change its lanes or stay on the same lanes. The advice is not mandatory, as vehicles can decide whether they want to change their lane depending on the surrounding traffic condition.

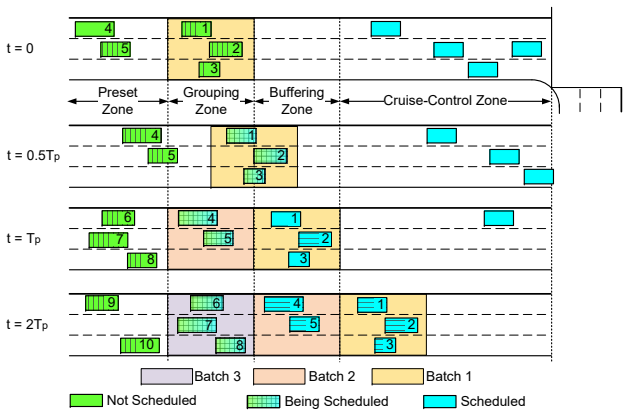


Fig. 4. The demonstration of the pipeline structure in Roadrunner.

2) Preset Zone (P-Zone): As soon as vehicles enter this zone, they are not allowed to change their lane, but accelerate their speed to V_{max} before exiting the zone. By regulating vehicles' speeds, their relative positions are fixed until Roadrunner instructs them to change their speeds. This zone allows the vehicles to move smoothly toward the intersection while Roadrunner is optimizing the scheduling for vehicles.

3) Grouping Zone (G-Zone): In this zone, vehicles **from all directions** are grouped into batches periodically, and scheduled by the vehicle scheduler for the time to enter the intersection. Meanwhile, vehicles in G-Zone continue moving toward the intersection with the speed V_{max}.

4) Buffering Zone (B-Zone): This zone provides a buffering time for the vehicle scheduler to finish the computation for each vehicle in the batch as vehicles move from G-Zone to B-Zone at the highest speed. Before vehicles exit B-Zone, the computation of each vehicle's entering time to the intersection must be completed.

5) Cruise-Control Zone (CC-Zone): In this zone, each vehicle is instructed to control its speed according to its determined arrival time at the intersection with their cruise control system. More specifically, vehicles first will decelerate their speed from V_{max} to a certain rate, and then accelerate to V_{max} or V_{turn} before entering the intersection, where V_{turn}, the turning speed in the intersection, is normally lower than V_{max} to prevent lateral force rollovers.

B. Pipeline in Roadrunner

The zones in Roadrunner provide different functions in a pipeline structure. To ensure that no vehicles are missed during the transition between zones, the time spent of each vehicle at each zone is fixed with a period T_p. In other words, vehicles in G-Zone are groups every T_p (analogy to taking a snapshot of the vehicles every T_p). Upon grouping a batch of vehicles, scheduling can immediately begin and must complete in T_p, which is also performed every T_p for each group.

Fig. 4 shows an example of the pipeline structure. At $t = 0$, vehicles in G-Zone are grouped into Batch 1 (or taken a snapshot), immediately followed by the scheduling. At $t = 0.5T_p$, while the scheduling is continuing, vehicles in Batch 1 keep moving to B-Zone. At $t = T_p$, the scheduling is completed and each vehicle of Batch 1 will immediately receive the instruction of cruising speeds in the CC-Zone. Meanwhile, vehicles in G-Zone are grouped into Batch 2, and another scheduling cycle starts. At $t = 2T_p$, Batch 1 enters CC-Zone, and the scheduling for Batch 2 is completed. At the same time, vehicles in G-Zone are grouped into Batch 3.

T_p is a system design parameter. The larger it is, the more vehicles are scheduled with MILP, achieving a better system performance. However, this also stresses the computation of the scheduler when optimizing delay performance using MILP. Since the computation time of the scheduler is bounded to T_p, a sweet spot of T_p needs to be determined to trade off system performance and computation complexity. By observing the computation time (T_c) with corresponding T_p's, the sweet spot can be determined with the largest feasible T_p, where corresponding T_c is bounded by T_p.

Fig. 5 shows a sample case to determine the sweet spot of T_p by simulating a 4-way 3-lane intersection in Manhattan. The vehicle arrival rate is 3.97 vehicles per second, and V_{max} is set as 11.176(m/s) (25mph) according to [16]. Roadrunner is executed as a single-threaded process on an Intel® Core™ i9-7900X CPU. In the figure, $T_p = 4.56seconds$ is the sweet spot, where system performance is maximized with limited computation resource.

T_p can also be used to determine the length of G-Zone (l_{GZ}) and B-Zone (l_{BZ}) with the speed V_{max}:

$$l_{GZ} = l_{BZ} = T_p * V_{max} \tag{1}$$

, since vehicles travel at V_{max} in both zones. Hence, for the example in Fig. 5, l_{GZ} should be set as 51(m) ($4.56*11.176 \geq 51$). In practice, considering the communication delay between Roadrunner and vehicles, B-Zone can be lengthen to make Roadrunner more robust.

In an urban area, the length of the zones might be restricted by the limited space around the intersection, meaning that l_{GZ} is shorter than the sweet spot. Even so, Roadrunner can still function normally. When the batch size is reduced to one vehicle, Roadrunner becomes a FCFS reservation-based vehicle scheduling. Therefore, Roadrunner can adapt to any intersection with arbitrary spaces and any available computation resources.

III. Vehicle-Scheduling Stage in Roadrunner

In this section, two main components in the second stage (vehicle scheduling): vehicle scheduler and collision region computation (CRC) are introduced as shown in Fig. 2. MILP is applied to minimize the average traveling delay, while collision prevention is formulated as constraints. To reduce computation complexity, MILP is formulated with time-relative variables. CRC is an offline process that transforms potential collision regions from the space domain to the time domain. The MILP formulation is explained in subsec. III-B, the CRC in subsec. III-C, and the "resource locking issue" in subsec. III-A.

A. Resource Locking Issue in Vehicle Scheduler

As mentioned before, ***resource locking*** unnecessarily locks up the intersection in a way that a batch of vehicles needs

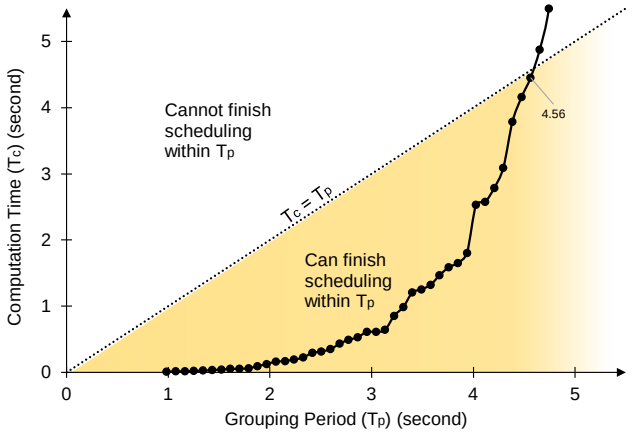
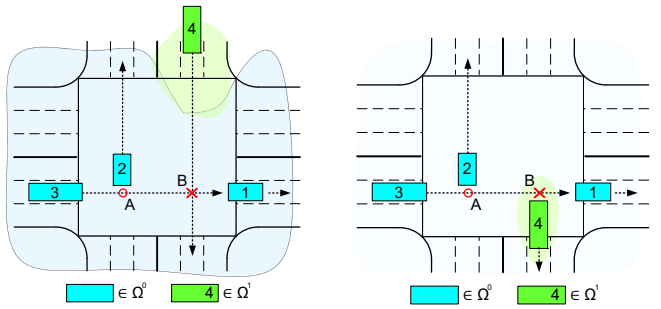


Fig. 5. The relation between the grouping period and the computation time. The period should be set as 4.56 second, since it is the largest feasible value.



(a) ICACC with locking issue (b) Roadrunner without locking issue

Fig. 6. Demonstration of the "locking issue". Vehicle 4 has to wait for Vehicle 3 in ICACC, while Vehicle 4 can pass through the intersection earlier in Roadrunner.

to wait until all vehicles in the previous batch pass the intersection. To the best of our knowledge, this is the first time that such problem is identified.

To explain the issue, a set of vehicles that have not yet been scheduled in G-Zone is denoted as Ω^1, and the set of vehicles that have been scheduled is denoted as Ω^0. For example, in Fig. 4, at time slot $t = T_p$, Batch 2 is denoted as Ω^1 and Batch 1 is denoted as Ω^0. At time slot $t = 2T_p$, Batch 3 is denoted as Ω^1, and the union of Batch 2 and Batch 1 is denoted as Ω^0.

At each time slot, vehicles in Ω^1 (not scheduled) need to avoid collision with vehicles in Ω^0 (scheduled). In ICACC, Ω^1 is forbidden to pass the intersection earlier than Ω^0, if there is a potential collision point on their trajectories. In other words, Ω^0 "locks up" the intersection. This increases the unnecessary waiting time of Ω^1 and reduces the overall throughput. In this paper, we found that vehicles in Ω^1 could, in fact, enter the intersection, before vehicles in Ω^0 arrive at the potential collision points.

Fig. 6 demonstrates the scheduling with and without the locking issue. For the potential collision point A, blue vehicles in Ω^0 are assumed to be scheduled in the order of: Vehicle 1, Vehicle 2, Vehicle 3. Specifically, Vehicle 3 can arrive at point A no earlier than Vehicle 2. In ICACC (Fig. 6a), Vehicle 4 in Ω^1 can only arrive at potential collision point B after Vehicle 3. However, since Vehicle 3 has to wait for Vehicle 2, Vehicle 4 can first have passed point B. Hence, in Roadrunner (Fig. 6b), Vehicle 4 can pass point B before Vehicle 3, and the locking issue is resolved. The details of the solution will be explained later.

B. MILP in Vehicle Scheduler

MILP is applied to formulate the intersection management problem and to optimize the scheduling. In subsec. II-A, we show that vehicles in CC-Zone can decelerate for a predetermined arrival time at the intersection, incurring a traveling delay (d) due to the deceleration. The traveling time in the CC-zone at V_{max} is denoted as (OT). Therefore, $ST = OT + d$. In our MILP formulation, we consider the length of vehicles (LV), and the headway (h) between the vehicles. The distance between a vehicle and the intersection is denoted as X, and the lane, where the vehicle i is, is denoted as L_i. The objective of the MILP is to minimize the total traveling delay of the vehicles without any collision. The MILP formulation for the scheduling problem is shown as following:

$$Min : \sum_{i}^{\Omega^1} d_i \tag{2}$$

subject to:

$$\begin{aligned}(OT_i + d_i) - (OT_j + d_j) \geq (LV_j + h)/V_{max};\\ \forall i, j \in \{\Omega^1 \cup \Omega^0\},\ X_i > X_j,\ L_i = L_j\end{aligned} \tag{3}$$

$$\begin{aligned}|(OT_m + d_m) - (OT_n + d_n)| \geq CRC(m,n));\\ m \neq n,\ \forall m, n \in \Omega^1\end{aligned} \tag{4}$$

$$\begin{aligned}|(OT_k + d_k) - (OT_l + d_l)| \geq CRC(k,l));\\ \forall k \in \Omega^1,\ l \in \Omega^0,\ d_l \in \mathbb{R}\end{aligned} \tag{5}$$

$$d_i \geq 0;\ \forall i \in \Omega^1 \tag{6}$$

Eq. (2) minimizes the traveling delay for vehicles in Ω^1, while the traveling delay (d) of Ω^0 is already determined in the previous optimization. Since vehicles in Ω^0 have already entered CC-Zone and start to change their speeds, their cruising speeds in CC-zone will not be modified by the scheduler. Therefore, in each optimization process, only vehicles in Ω^1 are optimized.

Constraint (3) is a first-come-first-serve (FCFS) policy for vehicles on the **same lane** to prevent rear-end collisions. Specifically, on the same lane, the vehicle closest to the intersection must enter the intersection before the others can. Since vehicles in CC-Zone have the same decelerating pattern (as shown in Fig. 3), this constraint with the scheduled time ST also ensures that vehicles will not overtake others on the same lane.

Constraint (4) ensures that no collision happens in the intersection between any two vehicles in Ω^1. In the left side of the inequation, CRC is a function that returns the minimal time gap between any two vehicles when they arrive at the

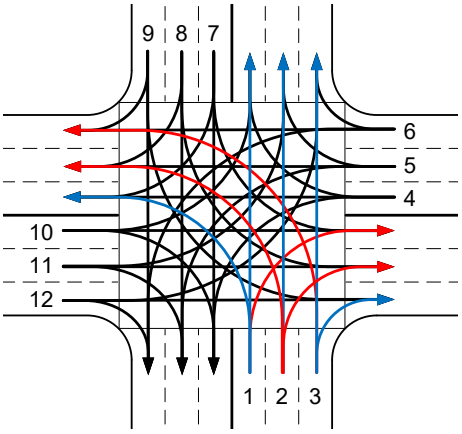


Fig. 7. The intersection model in Roadrunner. For vehicles coming from bottom, blue routes are the shortest trajectories for them to go to any directions, while red routes are other longer trajectories.

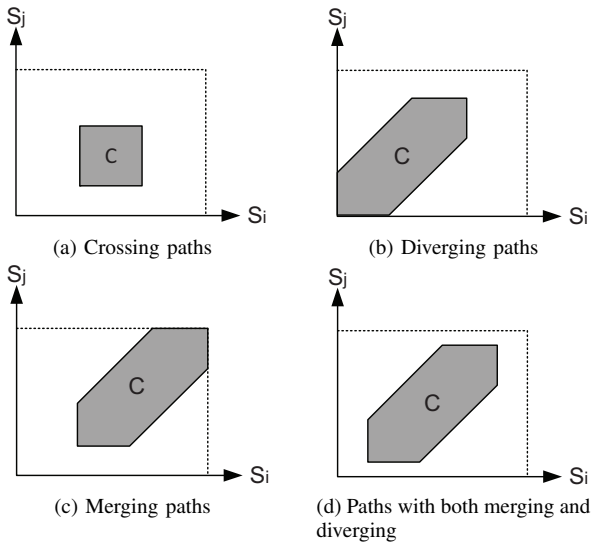


Fig. 8. Four types of collision for two vehicles: i and j.

intersection. CRC calculates the gap taking into account of the sizes, turning directions of the two vehicles and the lanes they are on. By setting the time gap between the two vehicles, they can pass the intersection without collisions.

Constraint (5) ensures that no collision happens between the vehicles of Ω^1 and Ω^0. The constraint is similar to constraint (4), while d_l is a constant (d_l is given by the previous optimization). Also, this constraint avoids the locking issue in ICACC by reserving the potential collision points only at the time when vehicles arrive (instead of locking the point all the time until the last vehicle in Ω^0 passes). Constraint (6) ensures that all the traveling delays are non-negative values.

With the MILP formulation, the vehicle scheduler can optimize the intersection usage. Not only collisions are avoided, but also the average traveling delay is minimized. In the next subsection, we will introduce how the function CRC in Constraint (4) works.

C. Collision Region Computation (CRC)

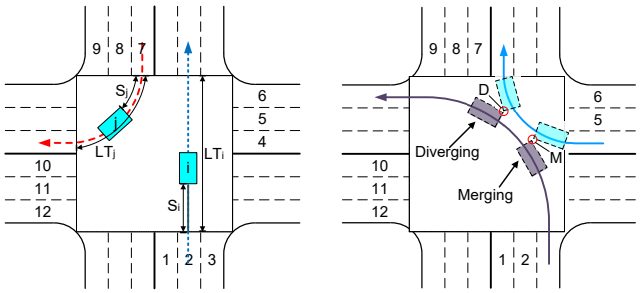
CRC can handle any size of intersection and any turn of the vehicle at the intersection. Fig. 7 shows a 4-way 3-lane intersection model and all the possible trajectories, some of which are colored. Blue routes are the main trajectories that are commonly designed for this size of intersection, while red routes are alternatives when the main trajectories are congested. The figure also shows all the possible collision points between two trajectories. The CRC process is done offline.

CRC is formulated with another set of MILP. According to Altch et al [10], conflicts between any two vehicles can be categorized into four types. According to the model in Fig. 7, they are: (a) crossing paths, (b) diverging paths, (c) merging paths, and (d) paths with both merging and diverging. For example, in Fig. 7, when two vehicles on lane 1 and on lane 10, respectively, are going straight, the possible conflict is defined as type (a). If a vehicle on lane 1 turns left and another one on lane 8 turns left, the possible conflict is defined as type (d).

Fig. 8 illustrates the collision regions of the four types. The two axes are the traveling distances(s) of two vehicles, each of which is defined as the distance that the corresponding vehicle has traveled **after** entering the intersection. The origin of the coordinate is the position of the two vehicles when they just enter the intersection. These graphs show whether collisions will occur when two vehicles have traveled for some distance in the intersection. The shadowed area are the region where two vehicles collide at the corresponding traveling distance. The traveling distance of vehicle i and vehicle j is denoted as s_i and s_j as shown in Fig. 9a. The collision region are modeled with traveling distances. Next, we will explain how to convert the models from distances to minimal time gaps.

Another set of MILP is formulated to find the minimal time gap between any two vehicles for collision avoidance. In the formulation, the four types of collision region (shadowed area in Fig. 8) are formulated into constraints. In fact, these types can be generalized into a single model as shown in Fig. 10. l_T is defined as the length of a trajectory in the intersection, and LV is the length a vehicle. To illustrate, Fig. 9b shows a case of type (d) collision path. The critical point when two vehicles start to merge is called merging point, and the critical point when two vehicles diverge is called diverging point. In other words, merging point is the minimal s_i and s_j where two vehicles collide, while diverging point is the maximal s_i and s_j where the two vehicles collide. The corresponding (s_i, s_j) at the merging point is denoted as (M_i, M_j), where M_i is the distance vehicle i has traveled when it arrives at the merging point. The corresponding (s_i, s_j) at the diverging point is denoted as (D_i, D_j), where D_i is the distance vehicle i has traveled when it arrives at the diverging point. These points are found with a simulation in a virtual environment that enumerates every position of two given trajectories and checks whether the two positions are too close.

In the MILP formulation of CRC, V is denoted as the speed of a vehicle in the intersection, which can be either V_{max} or V_{turn} depending on whether the vehicle is turning or not. $\Delta\tau$ is the time gap for two vehicles and finding the minimal $\Delta\tau$ is the objective of CRC. In fact, since the safe region is a concave area (unshaded area in Fig. 10), we instead formulate the MILP



(a) Notations in CRC (b) Example of type (d)

Fig. 9. The notations for CRC and an example of the collision of type (d).

problem with the collision region (shadowed area), which is convex. In Fig. 10, the red line represents the objective function, which is $\Delta\tau$. The graph shows that finding the feasible minimal gap in the safe region is equivalent to finding the boundary within the collision region. Hence, the MILP for CRC can be formulated as:

$$\begin{aligned} Min: \Delta\tau = \frac{s_i}{V_i} - \frac{s_j}{V_j} \\ where\ V_i,\ V_j \in \{V_{max},\ V_{turn}\} \end{aligned} \tag{7}$$

subject to:

$$\begin{aligned} 0 < s_i < LT_i \\ 0 < s_j < LT_j \end{aligned} \tag{8}$$

$$\begin{aligned} M_i < s_i < D_i \\ M_j < s_j < D_j \end{aligned} \tag{9}$$

$$(s_j - M_j) < (s_i - M_i) + LV_j \tag{10}$$

$$(s_j - M_j) + LV_i > (s_i - M_i) \tag{11}$$

$\Delta\tau$, s_i and s_j are variables, and other notations are given constants.

Eq. 7 describes the objective function which finds the minimal time gap. Specifically, this function optimizes the time-domain variable ($\Delta\tau$) transformed from the space-domain variables (s_i and s_j) which are used for describing collision regions. Given vehicle's speeds, the traveling distance with s_i and s_j can be mapped to the time gap $\Delta\tau$.

Constraint 8 limits the collision region by the length of trajectories. This constraint is influential when the conflict type is diverging paths (Fig. 8b) or merging paths (Fig. 8c). Constraint 9 limits the collision region by diverging points and merging points. If there is no diverging point, D_i and D_j are set to be infinite; similarly, if there is no merging point, M_i and M_j are set to be negative infinite. Constraints 10 and 11 represent the upper and lower slanted edge of the collision region. In crossing path (Fig. 8a), these two constraints can always be satisfied.

In fact, CRC does not need to solve the MILP for every pair of trajectories, because the intersection model is symmetric. Some similar collision conditions can share the same solution of the MILP by rotating the model. For example, in Fig. 7, trajectories starting at lane 1 shares the same collision conditions with trajectories starting at lanes 4, 7 and 10. Therefore, CRC only needs to record the minimal time gaps $\Delta\tau$ of every collision conditions with trajectories starting at lane 1, lane 2 and lane 3; for other collision conditions, CRC can find the corresponding minimal time gap by rotating the model.

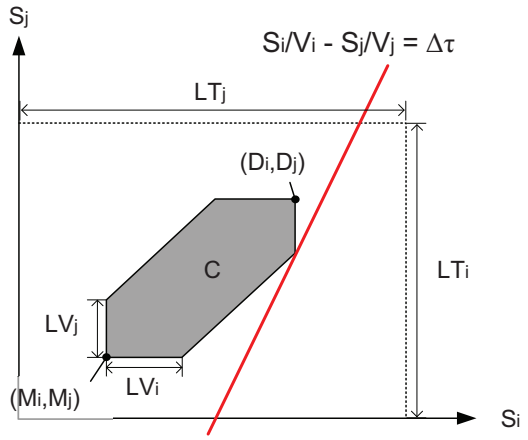


Fig. 10. The general form of the four collision types.

IV. Lane-Advising Stage in Roadrunner

Besides vehicle-scheduling, the throughput can be further improved by assigning a specific lane for every vehicle to stay on until passing the intersection. The assignment is done by a lane adviser. Since Roadrunner targets level-3 (conditional driving automation) [17] vehicles, the lane changing is not mandatory in light of unpredictable traffic conditions.

Lane advising is based on the information provided by the scheduler (shown in Fig. 2) as soon as vehicle scheduling cycle is completed. Here, we discuss the design of the lane adviser in details.

A. Latest Occupied Time (LOT)

The intersection is first partitioned into grids. A notion of Latest Occupied Time (LOT) is introduced here to facilitate lane advising. There are two kinds of LOTs, one for each grid and another for each trajectory. As the name says, LOT is the latest time that a grid or a trajectory will be occupied by vehicles. Once a batch of vehicles have been scheduled, the time that each of them will enter the intersection is determined. The LOT of all the grids on the trajectory of a vehicle is updated to the entering time of the vehicle to the intersection.

For example, in Fig. 11, vehicle A and B in Ω^1 are scheduled and their entering times are $t1$ and $t2$, respectively. The grids on vehicle A's trajectory are then updated with $t1$ and those on vehicle B's trajectory with $t2$. For the grids overlapped with the two trajectories, their LOTs are updated with $max(t1, t2)$. Once the LOT of all grids have been updated right after the scheduling cycle, the LOT of a trajectory is determined by the maximum LOT of the grips on the trajectory.

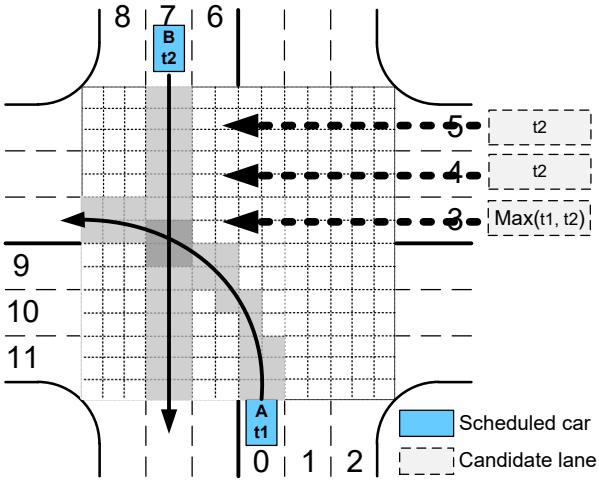


Fig. 11. Scheduled status determined by Roadrunner.

During lane advising, when a vehicle is assigned to a particular lane, the LOT of the grids on the vehicle's trajectory is updated as the following:

$$LOT \leftarrow LOT + \frac{LT_i}{V_i} \tag{12}$$

, where $V_i \in \{V_{max}, V_{turn}\}$ and LT_i is the length of the trajectory of Vehicle i in the intersection. The above update can only be approximated as the vehicle has not yet been scheduled.

Our proposed lane advising uses two different strategies: minimal-LOT advising (MLA in Sec. IV-B) and shortest-trajectory advising (STA in Sec. IV-C) depending on traffic loads. The details are described in subsec. IV-D.

B. Minimal-LOT Advising (MLA)

When a vehicle is advised for a lane, the adviser will choice the one where the vehicle's trajectory has the minimal LOT. The strategy is called Minimal-LOT Advising (MLA). As shown in Fig. 11, for a vehicle coming from right of the intersection and heading straight, there are three candidate lanes, Lane 3, 4, 5. Lane 3's LOT is $Max(t1, t2)$ and Lane 4's and Lane 5's LOT are both $t2$. Hence, either Lane 4 or Lane 5 is selected for the vehicle to stay on. Upon the advice given to the vehicle, the LOT of the lane is updated with Eq. 12.

Minimal-LOT advising is a greedy approach, regardless of its influence to other vehicles. When the traffic load is low, MLA can effectively reduce the average delay of vehicles. However, when the traffic load is high, vehicles are more likely to affect each other, and another strategy (shortest-trajectory advising) is more effective.

C. Shortest-Trajectory Advising (STA)

Shortest-Trajectory Advising (STA) suggests vehicles to stay on the lane where they can spend less time in the intersection, i.e., staying on the shortest trajectories. Specifically, left turning vehicles are advised to stay at the innermost lane, whereas right turning vehicles are advised to stay at the outermost lane as shown with the blue trajectory in Fig. 7. By taking the shortest trajectories, vehicles will block less vehicles on other lanes. For example, in Fig. 7, if a vehicle stays on lane 3 and turn left, it affects vehicles on lane 1, 2 and 4-12; however, if the vehicle stays on lane 1 and turn left, it only affects vehicles on lane 4 and 7-12. Because of this reason, shortest-trajectories advising is effective when traffic load is high.

D. Lane Advising in Roadrunner

As discussed above, MLA is effective when traffic load is low, while STA is effective when traffic load is high. In Roadrunner, the two advising strategies are used to adapt to traffic load conditions, which can be determined by observing the current utilization of the intersection. If Roadrunner starts to delay vehicles on all lanes (i.e., the demand is higher than the capacity), then traffic load is defined as high. In other words, a newly arriving vehicle is delayed no matter to which lane it is assigned. It cannot travel with V_{max}, because the intersection is occupied at the time it arrives. Therefore, during high traffic load, a lane candidate i can be written as:

$$LOT_i \leq \frac{(l_{PZ} + l_{GZ} + l_{BZ} + l_{CCZ})}{V_{max}} \tag{13}$$

,where l's are the lengths of the zones. If traffic load is high for **all** lane candidates, Roadrunner will switch from MLA to STA. It is worthwhile to mention that since LOTs depend on the trajectories of the vehicles, Roadrunner determines the traffic load considering the turning of vehicles.

V. Experimental Results

The experiments are conducted with a simulator implemented with Python, which is a single-threaded process running on a server with an Intel® Core™ i9-7900X CPU and 132 GB memory. An intersection is simulated with three lanes on each direction (Fig. 7). Preset Zone, Grouping Zone and Buffering Zone are set as 51 meters (as shown in Sec. II-B), and Cruise-Control Zone is set as 100 meters. We first compare different vehicle scheduling algorithms proposed in previous works, and then evaluate the performance of Roadrunner with different lane advising strategies to show that throughput can be improved and delay can be reduced. Experiments are also conducted under different traffic patterns.

A. Comparison of Different Vehicle Scheduling Algorithms

In this experiment, the traffic pattern is 30% right-30% left-40% straight, and arriving vehicles are equally distributed on each lane. We compare several vehicle scheduling algorithms with Roadrunner including traditional traffic signal (FixSignal), the latest optimization algorithm ICACC [12], and the FCFS strategy [6]. In this experiment, because vehicles can stay at any lane regardless its turning, for FixSignal, we set the green light independently on each lane to avoid any collision. The green light is set as 4.56 seconds for each lane.

Throughput of the intersection is first evaluated and shown in Fig. 12. Roadrunner with lane adviser provides highest

throughput among all vehicle schedulers. Compared to ICACC and FCFS, the throughput is improved by 73.74% and 72.73%, respectively. Compared to the traditional traffic signal, the throughput is improved by 130.84%. Compared to Roadrunner without lane adviser, the throughput is improved by 50.61%. Even without lane adviser, Roadrunner still provides a higher throughput than the others. For instance, Roadrunner without lane adviser improves 15.49% throughput over ICACC, resulted from the resolution of the locking issues (Sec. III-A) in ICACC.

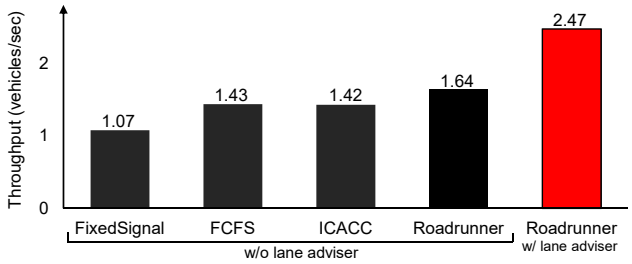


Fig. 12. Comparison of the throughput under different vehicle scheduling algorithms.

The average delay are compared in Fig. 13. Roadrunner with lane adviser provides the lowest average delay at both high and low traffic loads.

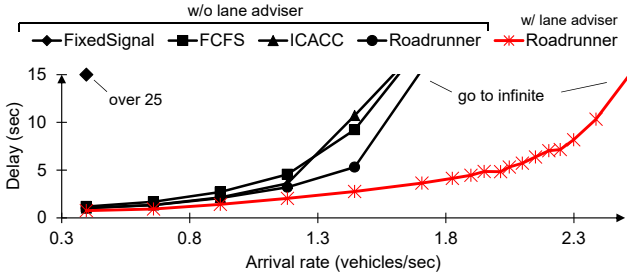


Fig. 13. Comparison of the average delay under different vehicle scheduling algorithms.

B. Comparison Between Lane Advising Strategies

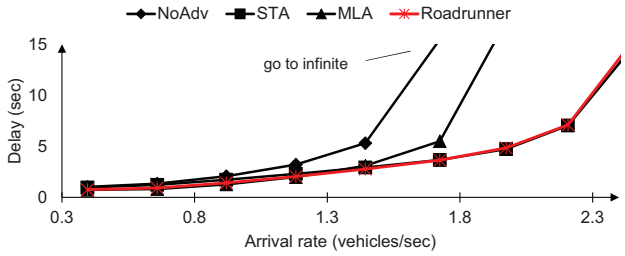
As shown above, the performance of Roadrunner with and without lane advising improves the performance in throughput in delay. We have also evaluated lane adviser in Roadrunner by comparing it to minimal-LOT advising (MLA) and shortest-trajectory advising (STA), respectively, under different traffic loads. In the experiments, we test two types of traffic pattern: (1) an uniform case, where traffic is equally distributed to all possible directions, and (2) an nonuniform case, where most vehicles go to the same destination due to special events such as concerts or the Super Bowl game.

TABLE I
THROUGHPUT WITH LANE ADVISER

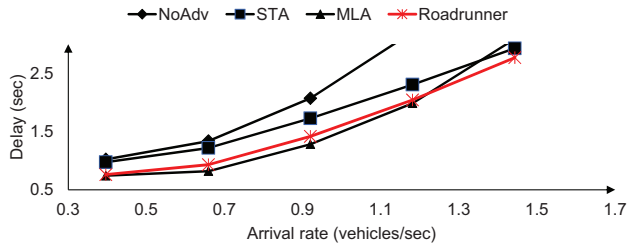
Traffic mix	NoAdv	STA	MLA	Roadrunner
Uniform	1.64	2.46	1.70	**2.47**
Nonuniform	1.08	0.88	1.23	**1.21**

1) Uniform Case: For the uniform case, the traffic pattern is 30% right-30% left-40% straight, and vehicles from every direction are equally distributed. Throughput is compared in Table. I. In the uniform case, Roadrunner without a lane adviser (NoAdv) has the lowest throughput compared with others with a lane adviser. Roadrunner has the highest throughput, because Roadrunner switches to STA when the traffic load is high. It achieves 50.61% higher throughput compared to NoAdv and 45.29% higher throughput compared to STA.

The average delays of vehicles are shown in Fig. 14a. NoAdv has the highest vehicular delay. In the figure, Roadrunner and STA has the lowest vehicular delay of vehicles when traffic load is high. In the low traffic load (Fig. 14b), MLA has the lowest vehicular delay and Roadrunner has a lower delay than STA. This shows that Roadrunner can have low vehicular delay by switching to MLA when traffic load is low.



(a) Both high and low traffic loads



(b) Low traffic load

Fig. 14. Comparison of the average delay with different lane advising strategies in uniform case.

2) Nonuniform Case: For the nonuniform case, the traffic pattern is 90% left-0% right-10% straight, and vehicles are from only one direction. Throughput is compared in Table. I. MLA performs better than STA, because STA suggests all the left-turning vehicles staying on the innermost lane, resulting in a severe backlog on the that lane. On the other hand, Roadrunner performs as good as MLA, because it detects that traffic load is low on some lanes and chooses MLA for lane advising. Roadrunner achieves 12.04% higher throughput compared to NoAdv and 37.50% higher throughput compared to STA. The average vehicular delay are shown in Fig. 15. Roadrunner and MLA have the lowest vehicular delay as shown in the figure. This shows that even with nonuniform traffic distributions, Roadrunner still has a high throughput and a low delay.

C. Different Lane-Advice Acceptance Rates

Here, we consider a realistic condition that vehicles might not be able to change their lanes before they enter the G-

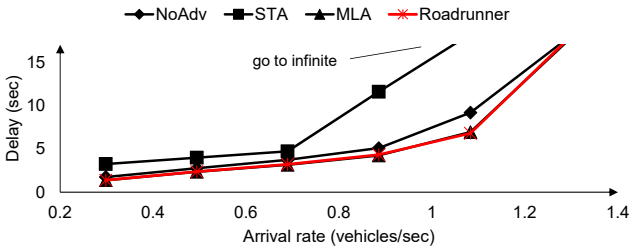


Fig. 15. Comparison of the average delay with different lane advising strategies in nonuniform case.

Zone. Vehicles are allowed to ignore lane advice due to safety concerns or space limitation of the road. In this experiment, we evaluate the performance of Roadrunner under different acceptance rates of lane advice as shown in Fig. 16. The traffic pattern is 30% right-30% left-40% straight, and vehicles from any direction are equally distributed to all other possible directions. The 100% lane-advice acceptance rate means that all vehicles take the lane advice and successfully change to the suggested lane. The 0% lane-advice acceptance rate means that no vehicles take the lane advice. When the lane-advice acceptance rate decreases, the throughput of the intersection decreases linearly.

The lowest throughput locates at 0% lane-advice acceptance rate, which is 1.64 vehicles per second. Even under this situation, Roadrunner's throughput is still higher than other vehicle schedulers' as shown in Fig. 12.

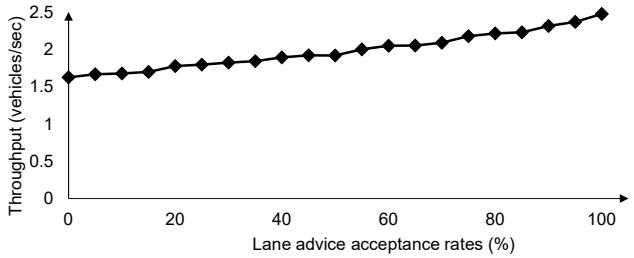


Fig. 16. Throughput of the intersection under different lane advice acceptance rates.

VI. Related Works

Intersection management has been long regarded as a key function in cooperative intelligent transportation systems (C-ITS). Several systems [18]–[20] have been proposed to improve the vehicular throughput and minimize delay by controlling traffic signals. These systems relied on the traffic signals and human drivers, which became the bottleneck for roadway efficiency due to human's slow reacting time. Hence, some researchers [6]–[8], [10], [12], [21] proposed some no-traffic-signal management systems for automated vehicles to eliminate human factor. Dresner and Stone [6] developed an intersection management based on a FCFS reservation algorithm for autonomous vehicles. Choi et al. [7], [8] also proposed a reservation-based scheme for intersection management. Their experiments indicate that the reservation-based system outperform the traffic lights mechanism. However, the FCFS reservation policy, while being computationally efficient, can be further improved with optimization [9].

For this reason, other researchers have proposed several optimization-based intersection management systems [10]–[12]. Altch et al. [10] formulated the management problem as a mixed-integer linear program (MILP) and minimized the sojourn time (which is equivalent to maximize the throughput). Vu et al. [11] argued that centralized optimal solutions resulted in high computation requirements. Hence, they proposed a distributed optimization approach using a max-sum algorithm. However, as we have showed in the paper, the complexity of the optimization can be controlled and the computation can be finished in a reasonable time. Zohdy and Rakha [12] proposed ICACC with simplified formulation which computed only the time that vehicles entered the intersection for collision free. However, ICACC did not consider various vehicle sizes nor the flexibility of turning to any direction on each lane. Also, there is a "resource locking issue" in their MILP formulation.

For lane assignment, researchers [13]–[15] have proposed several algorithms to improve road performance. Schakel and Arem [13] proposed an advisory system based on a prediction model that gives advice on vehicles' lane, speed and headway. Xing et al. [14] proposed an application that could balance lane usage to achieve better lane utilization for motorway traffic. Yao et al. [15] used genetic algorithm to optimize a collective optimum, a group optimum, and a user optimum to improve efficiency. These studies mainly focus on roads instead of intersections, and their methods are mainly based on a predicting model. In this paper, we have designed a completely new lane advising scheme taking advantages of the information from a previous scheduling cycle to perform more effective lane advising than the predication-based schemes.

VII. Conclusion

In this paper, we propose Roadrunner, a novel architecture of intersection management. It consists of a vehicle scheduler and a lane adviser, arranged in a pipeline structure. For vehicle scheduling, two sets of MILP are formulated to minimize vehicle's delay for passing the intersection and collision region between the vehicles. Our proposed algorithms also eliminate so-called "resource locking" issue to further improve the throughput. For lane advising, it takes advantages of knowing the latest occupied time (LOT) of every trajectory in the intersection (such information is provided by the vehicle scheduler), and is hence able to provide intelligent advice to each vehicle on which lane it should stay to possibly have the shortest travel time. It uses two advising strategies, Minimal-LOT Advising and Shortest-Trajectory Advising (STA), to adapt to traffic load conditions. Roadrunner outperforms the existing algorithm ICACC by 73.74% in throughput and delay under uniform traffic. Under different traffic patterns, the lane adviser helps improve throughput by up to 50.61%. We also showed that even under low lane-advice acceptance rates, Roadrunner's throughput is still higher than other vehicle scheduling schemes.

ACKNOWLEDGMENT

We would like to thank Professor Joseph Chow for giving us valuable inputs during the course of our research, and Qing Zhang for his assistance in programming some of the algorithms of the Rouadrunner.

REFERENCES

[1] NHTSA, *2015 Traffic Safety Facts*. National Highway Traffic Safety Administration, U.S. Department of Transportation, 2017.

[2] ——, *2015 Rural/Urban Comparison of Traffic Fatalities Traffic Safety Fact*. National Highway Traffic Safety Administration, U.S. Department of Transportation, 2017.

[3] M. Dotoli, M. P. Fanti, and C. Meloni, "A signal timing plan formulation for urban traffic control," *Control engineering practice*, vol. 14, no. 11, pp. 1297–1311, 2006.

[4] A. Di Febbraro and N. Sacco, "On modelling urban transportation networks via hybrid petri nets," *Control engineering practice*, vol. 12, no. 10, pp. 1225–1239, 2004.

[5] J. A. Guerrero-Ibanez, S. Zeadally, and J. Contreras-Castillo, "Integration challenges of intelligent transportation systems with connected vehicle, cloud computing, and internet of things technologies," *IEEE Wireless Communications*, vol. 22, no. 6, pp. 122–128, 2015.

[6] K. Dresner and P. Stone, "A multiagent approach to autonomous intersection management," *Journal of artificial intelligence research*, vol. 31, pp. 591–656, 2008.

[7] M. Choi, A. Rubenecia, and H. H. Choi, "Reservation-based cooperative traffic management at an intersection of multi-lane roads," in *Information Networking (ICOIN), 2018 International Conference on*. IEEE, 2018, pp. 456–460.

[8] ——, "Reservation-based cooperative intersection crossing scheme for autonomous driving in the intersection," in *2018 Tenth International Conference on Ubiquitous and Future Networks (ICUFN)*. IEEE, 2018, pp. 654–659.

[9] F. Altché and A. De La Fortelle, "Analysis of optimal solutions to robot coordination problems to improve autonomous intersection management policies," in *Intelligent Vehicles Symposium (IV), 2016 IEEE*. IEEE, 2016, pp. 86–91.

[10] F. Altché, X. Qian, and A. de La Fortelle, "Time-optimal coordination of mobile robots along specified paths," *arXiv preprint arXiv:1603.04610*, 2016.

[11] H. Vu, S. Aknine, and S. D. Ramchurn, "A decentralised approach to intersection traffic management." in *IJCAI*, 2018, pp. 527–533.

[12] I. H. Zohdy and H. A. Rakha, "Intersection management via vehicle connectivity: The intersection cooperative adaptive cruise control system concept," *Journal of Intelligent Transportation Systems*, vol. 20, no. 1, pp. 17–32, 2016.

[13] R. W. Hall and D. Lotspeich, "Optimized lane assignment on an automated highway," *Transportation Research Part C: Emerging Technologies*, vol. 4, no. 4, pp. 211–229, 1996.

[14] J. Xing, E. Muramatsu, and T. Harayama, "Balance lane use with vms to mitigate motorway traffic congestion," *International journal of intelligent transportation systems research*, vol. 12, no. 1, pp. 26–35, 2014.

[15] S. Yao, V. L. Knoop, and B. van Arem, "Optimizing traffic flow efficiency by controlling lane changes: Collective, group, and user optima," *Transportation Research Record: Journal of the Transportation Research Board*, no. 2622, pp. 96–104, 2017.

[16] N. Y. C. DOT, *Manhattan Speed Limits Maps*. New York City Department of Transportation. [Online]. Available: http://www.nyc.gov/html/dot/downloads/pdf/current-pre-vision-zero-speed-limit-maps.pdf

[17] S. International, "Taxonomy and definitions for terms related to driving automation systems for on-road motor vehicles," 2018.

[18] D. A. Roozemond, "Using intelligent agents for pro-active, real-time urban intersection control," *European Journal of Operational Research*, vol. 131, no. 2, pp. 293–301, 2001.

[19] M. C. Choy, D. Srinivasan, and R. L. Cheu, "Cooperative, hybrid agent architecture for real-time traffic signal control," *IEEE Transactions on Systems, Man, and Cybernetics-Part A: systems and humans*, vol. 33, no. 5, pp. 597–607, 2003.

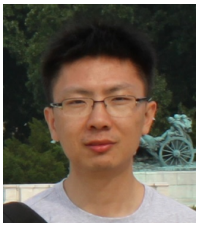
[20] X. X. Weng, S. S. Yao, and X. F. Zhu, "Architecture of multi-agent system for traffic signal control," in *Control, Automation, Robotics and Vision Conference, 2004. ICARCV 2004 8th*, vol. 3. IEEE, 2004, pp. 2199–2204.

[21] J. Lee and B. Park, "Development and evaluation of a cooperative vehicle intersection control algorithm under the connected vehicles environment," *IEEE Transactions on Intelligent Transportation Systems*, vol. 13, no. 1, pp. 81–90, 2012.

Michael I.-C. Wang received the M.S. degree in electrical and computer engineering from National Chiao Tung University, Taiwan, 2018. He is currently pursuing a joint Ph.D. degree in electrical and computer engineering in National Chiao Tung University and New York University Tandon School of Engineering. His research has been focused on software defined networking (especially on architecture design and resource allocation for datacenters). Recently, he is working on integrating the techniques of networking and intelligent transportation system (ITS) to improve ITS under upcoming 5G environment.

Jianyu Pang received a B.S. degree in South China University of Technology, Guangzhou, China in 2017. Currently, he is working toward the Masters degree at the ECE department of Tandon school of Engineering, Brooklyn, New York. His main research interests include optimization control, reinforcement learning, non-signal traffic control, and intelligent transportation systems.

Charles H.-P. Wen (M'07) received the Ph.D. degree in very-large-scale integration verification and test from the University of California, Santa Barbara, Santa Barbara, CA, USA, 2007. He is an Associate Professor with National Chiao Tung University, Hsinchu, Taiwan, and is a specialist in computer engineering. Over the past few years, his research has been focused on applying data mining and machine learning techniques to SoC designs (including radiation hardening, functional verification and timing analysis) and cloud networking (especially on performance analysis and architecture design of largescale datacenters). Prof. Wen was a recipient of the best paper awards from 2012 ASP-DAC, 2014 SASIMI, 2016 ICOIN, 2017 ICOIN and the Distinguished Young Scholar Award of Taiwan IC Design Society.



Yang Xu (S'05-M'07) received the B.E. degree from Beijing University of Posts and Telecommunications in 2001, the M.Sc. degree in computer science and technology from Tsinghua University in 2003, and the Ph.D. degree in computer science and technology from Tsinghua University, China, in 2007. From 2007 to 2019, he was with the Department of Electrical and Computer Engineering, New York University Tandon School of Engineering, where he was a Research Associate Professor since 2013. Since March 2019, he has been a Full Professor with the School of Computer Science, Fudan University, China. He has published about 70 journal and conference papers and holds more than 10 U.S. and international granted patents on various aspects of networking and computing. His research interests include software-defined networks, data center networks, network function virtualization, and network security. He served as a TPC member for many international conferences, as an Editor for the Journal of Network and Computer Applications (Elsevier), and as a Guest Editor for the IEEE Journal on Selected Areas in CommunicationsSpecial Series on Network Softwarization & Enablers and Wiley Security and Communication Networks JournalSpecial Issue on Network Security and Management in SDN.

H. Jonathan Chao (M'83-F'01) received the B.S. and M.S. degrees in electrical engineering from National Chiao Tung University, Taiwan, in 1977 and 1980, respectively, and the Ph.D. degree in electrical engineering from The Ohio State University in 1985. He has been a Professor with the Department of Electrical and Computer Engineering (ECE), NYU, since 1992. He was the Head of the ECE Department from 2004 to 2014. He is currently the Director of the High-Speed Networking Lab. He holds 61 patents and has published more than 260 journal and conference papers. He has co-authored three networking books, Broadband Packet Switching TechnologiesA Practical Guide to ATM Switches and IP Routers (New York: Wiley, 2001), Quality of Service Control in High-Speed Networks (New York: Wiley, 2001), and High-Performance Switches and Routers (New York: Wiley, 2007). He has been doing research in software defined networking, network function virtualization, datacenter networks, high-speed packet processing/switching/routing, network security, quality of service control, network on chip, and machine learning for networking. From 1985 to 1992, he was a Member of Technical Staff at Bellcore, where he was involved in transport and switching system architecture designs and ASIC implementations, such as the worlds first SONET-like Framer chip, ATM Layer chip, Sequencer chip (the first chip handling packet scheduling), and ATM switch chip. From 2000 to 2001, he was a Co-Founder and a CTO of Coree Networks, NJ, USA, where he led a team to implement a multi-terabit multi-protocol label switching (MPLS) switch/router with carrier-class reliability. He is a Fellow of the National Academy of Inventors for having demonstrated a highly prolific spirit of innovation in creating or facilitating outstanding inventions that have made a tangible impact on quality of life, economic development, and the welfare of society. He is a Fellow of the IEEE for his contributions to the architecture and application of VLSI circuits in high-speed packet networks. He was a recipient of the Bellcore Excellence Award in 1987. He was a co-recipient of the 2001 Best Paper Award from the IEEE Transaction On Circuit And System For Video Technology.