This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/JIOT.2018.2817459, IEEE Internet of Things Journal

A Distributed Intersection Management Protocol for Safety, Efficiency, and Driver's Comfort

Xiaoyuan Liang, Tan Yan, Joyoung Lee, *Member, IEEE* and Guiling Wang, *Member, IEEE*

***Abstract*—Improving safety and convenience is always the top priority in designing today's intelligent transportation system. In this paper, we study the problem of how to manage vehicle traffic at intersections by jointly considering safety, driver's comfort, and efficiency, in vehicular ad hoc networks. We propose a Distributed Intersection Management Protocol (DIMP), which distributedly coordinates vehicle traffic from different directions by making vehicles exchange critical driving information and adaptively react based on the information. DIMP dynamically guides vehicles to adjust their speed in a way such that both safety and driver's comfort are satisfied. By following DIMP, vehicles can pass the intersections safely and efficiently at a comfortable speed during acceleration/deceleration. We extensively evaluate DIMP, and the evaluation results show that DIMP is both effective and efficient in managing vehicle traffic at intersections.**

I. Introduction

Driving gradually becomes a dominant mode of transportation all over the world. As road topologies and traffic patterns get increasingly complex, safety and energy efficiency become the top concerns. It is reported that nearly half of the 43,000 accidents each year in U.S. highways result from vehicles passing intersections unsafely [1]. Although traffic lights are used at a majority of intersections to manage the passing of vehicles, most systems do not work in an efficient way as they operate with a fixed cycle. It is common to see vehicles stop in front of a red traffic light even when there is no vehicle on the intersecting road. To improve the efficiency of traditional traffic lights, adaptive traffic light control systems are studied in reference [2]–[4], which dynamically adjusts the cycle length of traffic lights based on traffic information estimated from loop detector sensors [5] or cameras. However, the methods cannot be applied widely as additional equipments are needed, and they are not effective enough due to the inaccurate traffic estimation. More importantly, over 48 million intersections in the U.S. do not have traffic lights [6].

With the heavy demand of intelligent transportation, Vehicular Ad hoc NETworks (VANETs) grow rapidly since 1980, where vehicles can be organized and self-coordinated via wireless communication. VANETs are expected to be widely applied to improve road safety and driving connivance [7]. In VANETs, virtual traffic lights are proposed to imitate traditional traffic lights, which generates a self-organization system composed of vehicles [6]. However, existing works assume that communication in VANETs is established under a perfect environment, and the drivers' comfort level, an important factor affecting driving safety, is not considered. In order to make such a self-organization system operate safely and smoothly, vehicles at an intersection have to brake hard enough to achieve safety but not too strong to avoid driver's discomfort. With this objective, in this paper we study the problem on how to carefully coordinate vehicles' speed at intersections to achieve both safety and comfort.

We solve such a problem by considering the following two aspects: 1) traffic management, which schedules the sequential order for vehicles from different road segments to pass an intersection; 2) comfort braking distance, which warns the vehicles that cannot pass the intersection to brake early to avoid safety issues.

We design a Distributed Intersection Management Protocol (DIMP) in VANETs, which makes vehicles exchange messages to decide the order for passing intersections based on current traffic information in an ad hoc fashion. Our general idea is as follows: at an intersection, if all traffic is from only one direction, the vehicles can just drive through the intersection without waiting; if there are vehicles from different directions, they negotiate to agree on an order to pass the intersection so that the average waiting time is minimized. Specifically, in DIMP, vehicles are grouped into clusters based on their locations and driving directions. Each cluster elects a cluster head in charge of exchanging messages with other clusters and managing cluster members. When a vehicle cluster approaches an intersection, the cluster head broadcasts messages to vehicles on other roads to exchange critical driving information. The cluster whose location is the closest to the intersection can pass the intersection first, and others wait. This cluster notifies other clusters after all the vehicles in the cluster pass the intersection, and launches the next round of selection. When there are too many vehicles in the current running cluster, the cluster splits, and the next round of selection is restarted after a portion of the cluster passes the intersection to avoid making other clusters wait too long and to achieve fairness.

We propose a concept of *comfort braking distance* for vehicles, which warns each vehicle cluster that cannot pass the intersection to brake at proper time to avoid danger and

X. Liang and G. Wang are with the Department of Computer Science, New Jersey Institute of Technology, Newark, NJ, 07102 USA e-mail: {xl367, gwang}@njit.edu.

T. Yan is with Ant Financial, Hanzhou, China 310000 email: tanyan.ty@gmail.com.

J. Lee is with the Department of Civil and Environmental Engineering, New Jersey Institute of Technology, Newark, NJ, 07102 USA e-mail: {xl367, gwang}@njit.edu.

Manuscript received September 28, 2017; revised January 26, 2018; accepted March 05, 2018. Date of publication XXX XX, 20XX; date of current version XX XX, 20XX.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/JIOT.2018.2817459, IEEE Internet of Things Journal

discomfort, and to guarantee efficiency at the same time. We design several heuristics to handle the practical issues in on-road scenarios, such as cluster head handover and dynamic cluster formation, to balance the waiting time for different clusters. The performance of DIMP is evaluated extensively with simulator OMNeT++ [8] associated with SUMO [9] and Veins [10]. The results show that DIMP can control the vehicle speed to avoid uncomfortable and unsafe deceleration, and reduce waiting time, and stop times.

The rest of this paper is organized as follows. The related work is presented in Section II. The background and assumptions of the problem are in Section III. Section IV presents the distributed intersection management protocol, DIMP. Section V evaluates the protocol. Finally, the paper is concluded in Section VI.

II. Related Work

The related work is classified into two parts, adaptive traffic control and virtual traffic control.

A. Adaptive Traffic Control

Existing work about adaptive traffic lights usually collects traffic information of each lane via wireless sensor networks or VANETs, and then adjusts the cycle length of traffic lights [2], [11]–[16]. In these systems, a control center is available at the traffic light side to control the traffic light's phase sequence or phases' timing [2], [11]. The schemes to control the traffic light range from fuzzy logic [12], linear programming [13], [14] to deep learning [15], [16]. These systems have two main drawbacks. First, the sensors can only collect the information between them. If vehicle waiting length is longer than the distance between two sensors, the waiting length can not be accurately estimated. Second, these systems can only be applied in intersections with traffic lights and a control sever is required. In our system, VANETs provide a way to detect vehicles in a long distance and no control center is required.

B. Virtual Traffic Control

At intersections without traffic lights, researchers propose to organize vehicles via wireless communication. Ferreira *et al.* [6] first propose the concept of Virtual Traffic Lights (VTLs) to manage traffic at intersections via V2V communication. But the authors make some assumption too idealistic in many situations, such as messages have no latency and the braking distance is not considered, which means, a vehicle can stop immediately with zero braking distance. Some work is done to improve the original VTL in safety, pedestrians' passing or wireless connection [17]–[21]. Neudecker *et al.* evaluate the feasibility of VTL in non-line-of-sight environments [7] and a fail-safe model in VTL is proposed to deal with the scenario when communication can not be established [17]. Viriyasitavat [18] proposes to control the virtual traffic light for both motorized and non-motorized traffic, such as pedestrians. Shi *et al.* [19] add a new type named by 'will' value in the broadcasting messages to evaluate the unbalance among vehicles based on VTL. The negative acceleration in the study is set 5 m/s^2, which is much higher than the comfortable negative acceleration. Tonguz *et al.* [20] propose a system based on VTL to achieve the priority management at intersections for ambulances or fire trucks. Chou *et al.* [21] present all possible phases' combination in a traffic light and provide speed suggestion to drivers based on VTL. These protocols are mainly based on VTL and they do not improve the protocol's efficiency.

Some researchers also propose protocols to manage vehicles at intersections. Azimi *et al.* [22] introduce *stop* and *clear* messages for resource reservation and release at intersections, which is extended into *enter*, *cross* and *clear* messages in [23], but one-lane intersections are considered. Pasin *et al.* [24] propose a first-come-first-serve based intersection management protocol on VANETs. Yang *et al.* [25] implement the in-vehicle traffic light in an all-way stop based on the VANETs to help improve the post-encroachment time when no traffic lights are available. In these protocols, the authors focus on one-lane intersections and changing lanes are not allowed. The most related work is proposed by Bazzi *et al.* [26], which relies on the V2V to organize vehicles at intersections. In their proposed protocol, vehicles are grouped into clusters and clusters' heads negotiate with each other about the passing order. The protocol restrictedly limits only one vehicle to pass the intersection one time, which largely reduces the performance when non-conflicting vehicles pass the intersection. In addition, the protocol does not consider multi-lane situation and a central server is needed when the vehicles cannot obtain the passing order.

Some other protocols propose to optimize the intersection for Connected and Autonomous Vehicles (CAVs). Alejandro *et al.* [27] propose a Cooperative Intersection Control (CIC) method to achieve the virtual platooning based on the V2V communication. Lee *et al.* [28] propose a Cooperative Vehicle Intersection Control (CVIC) system for trajectory planning using a non-linear constraint optimization algorithm. Zhu *et al.* [29] present a Linear Programming formulation for Autonomous Intersection Control (LPAIC) that is able to obtain exact optimal solutions for scheduling vehicles at intersections. To deal with conflicting trajectories, a number of nonlinear constraints are proposed to guarantee the single-occupancy of collision regions. However, in these methods, vehicles are completely autonomous, whose trajectories are exactly predicted in advance. In addition, lane-changing is not allowed in these optimization protocols because the constraints for lane-changing is not able to be linearized [30]. In our protocol, vehicles are allowed to change lanes, pass over vehicles and stop when emergencies happen, which are hard to be predicted and modeled in an optimization system. Drivers are guaranteed to be comfortable and safe by restricting the max negative acceleration and only making non-conflicting vehicles pass the intersection at the same time.

III. Background and Assumptions

A. Background

VANETs are composed of vehicles and Road-Side Units (RSUs). A vehicle is equipped with an On-Board Unit (OBU) and multiple Application Units (AUs). AUs contain applications that take advantage of OBUs' communication capability to communicate with other vehicles or RSUs within the communication range. The communication between vehicles is based on the Dedicated Short-Range Communications (DSRC) technology, which was allocated 75 MHz of licensed spectrum in the 5.9 GHz band by the U.S. Federal Communications Commission. It is a short-range protocol with hundreds of meters as the communication range [31].

B. Assumptions

1) Every vehicle is equipped with an on-board unit and application units in the vehicle can employ it to send and receive messages. A same traffic management system is installed as an application unit in all the vehicles. They are all driven under the same traffic management protocol on the road. As on-board units are standard communication equipment in VANETs [31], it is reasonable to assume that vehicles are equipped with them.

2) Vehicles' locations are available and relatively accurate. Past research shows that localization accuracy can be maintained at less than 30cm errors by integrating GPS, IMU, wheel odometry [32] or less than 50cm by integrating lane and vehicle tracking [33] or other methods, like vision [34], [35]. Thus it is reasonable for us to assume the vehicle's accurate location can be obtained. The location of an intersection is available for every vehicle, which can be achieved by the same map saved in the vehicle.

3) We assume the direction of each vehicle at intersections is known. Usually, the destination of each vehicle is known before drivers start driving, thus the route and the direction at every intersection are also set. Even in traditional traffic light systems, it is also required that turning lights must be turned on before a vehicle passes an intersection if it makes a turn.

IV. Protocol

In this paper, we aim to solve the problem of efficiently managing traffic at intersections in a distributed way without causing discomfort. In order to solve the problem, we design a distributed intersection management protocol in VANETs, which increases the traffic efficiency via reducing vehicles' waiting time and stop times at intersections. There are mainly two challenges: the first one is when to establish communication channels between vehicles when they are approaching an intersection; the second one is how to organize vehicles at intersections in an efficient way under a distributed environment. To deal with the challenges, we first present the concept of comfort braking distance. Based on which, we introduce our distributed intersection management protocol.

A. Comfort Braking Distance

The braking distance is defined as the distance that a vehicle needs from it begins braking to it completely stops. Similarly, the Comfort Braking Distance (CBD) is defined as the minimum distance that does not cause any drivers' discomfort during the deceleration process. CBD indicates the location, starting where vehicles have to get the information about whether it should stop and yield to other vehicles. A similar concept, named safety distance, is proposed in [17]. But it does not consider a driver's comfort level. The discomfort is mainly caused by a too large negative acceleration during braking process [36]. Thus, the negative acceleration should be limited under a threshold.

TABLE I: Braking distance

Speed		Time (s)	d_{CBD} (m)
MPH[1]	MPS[2]		
10	4.47	2.24	4.99
20	8.94	4.47	19.98
30	13.41	6.71	44.96
40	17.88	8.94	79.92

We use d_{CBD} to denote the value of CBD. We use v_0 and v_1 to denote the speed before and after a vehicle brakes, respectively. Let a denote the max value of the comfort negative acceleration, which is a constant, and t denote the duration for the whole braking process. Then we have the following equation to calculate the value of CBD:

$$d_{CBD} = \frac{(v_0 + v_1)}{2} \times t. \tag{1}$$

Considering it is a deceleration process, v_1 and v_0 satisfy the following equation:

$$v_1 = v_0 - (a \times t). \tag{2}$$

Considering a vehicle is stopped after braking, v_1 is equal to 0 m/s. Thus, the value of CBD is obtained by combining Equ. (1) and (2),

$$d_{CBD} = \frac{v_0^2}{2 \times a}. \tag{3}$$

The equation indicates CBD can be calculated based on a vehicle's original speed when it starts braking and the comfort negative acceleration. Wu *et al.* [36] show that the negative acceleration should not exceed a threshold to ensure drivers' comfort. If a driver decelerates in a negative acceleration speed less than this threshold, he/she would not feel discomfortable. The threshold is suggested to be 2 m/s^2 in [36]. Based on the max comfort negative acceleration, the values of CBD under different original vehicle speeds can be calculated, which is shown in Table I. We can see that d_{CBD} increases quadratically with the increasing speed under the fixed negative deceleration.

B. DIMP

This section presents the distributed intersection management protocol in detail. To manage vehicles at intersections, a traffic flow model is built and the traffic is managed based on the model. In such a model, vehicles periodically broadcast their information to create clusters. In each cluster, a cluster head is automatically elected via the broadcasting messages. Cluster heads determine the intersection occupation sequence of every cluster via the negotiation between each other. We

[1] mile per hour
[2] meter per second

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/JIOT.2018.2817459, IEEE Internet of Things Journal

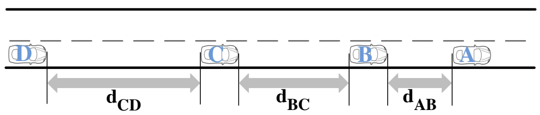


Fig. 1: An example of clustering vehicles

present the protocol in two parts, the first one is about the traffic flow model and the second is about how the sequence is determined by cluster heads.

1) Traffic flow model: In the traffic flow model, a vehicle needs to periodically broadcast a message, named *ClusterInfo*, including the information of its current coordinates and lane. Based on the information, every vehicle's direction towards the intersection and distances between any two are obtained. Thus, vehicles are free to move to any available locations as long as the updated information is broadcast immediately. The clusters are reconstructed correspondingly.

In the clustering process, the current locating lane is first considered. If two vehicles are at two different lanes, they belong to two clusters. The reason to choose one lane instead of one road is that one road may have multiple lanes and vehicles in different lanes have different directions. Thus it is more efficient in resource allocation to cluster vehicles in a lane. When vehicles are in the same lane, their inter-vehicle distance is used to construct clusters. Without loss of generality, assuming vehicles are in the same lane, we use d_{int} to denote the inter-vehicle distance between two vehicles. d_{int} is a variable, which changes with the real-time locations of two vehicles. In order to enable cluster heads to manage max number of vehicles, we set the size of a cluster as large as possible. We use d_{com} to denote the communication range, assuming it is larger than d_{CBD}. Here we just assume d_{com} is larger than d_{CBD}. d_{com} and d_{CBD} are compared in the evaluation section. When $d_{int} > d_{com}$, they are in two different clusters. It means when vehicles cannot receive messages from each other, they are in two different clusters. In other words, only vehicles in the same cluster can receive the broadcasting messages. The communication range can be over 200 meters in real field [37]. The number of vehicles in a cluster may become a large value.

Considering a cluster may be very long, we define chunks in a cluster to make a cluster separable. Each cluster is divided into multiple chunks. Due to the unknown relation between $2 \times d_{CBD}$ and d_{com}, let d_{chunk} satisfy the following equation,

$$d_{chunk} = min\{2 \times d_{CBD}, d_{com}\}. \tag{4}$$

Here, $2 \times d_{CBD}$ is an empirical value, which should be large enough to avoid frequent stopping and not too large to reduce other clusters' waiting time. When $d_{int} < d_{chunk}$, they are in the same chunk; otherwise, they are in different chunks. A cluster may have multiple chunks. Clusters and chunks are designed for different intentions. Clusters are to maximize the number of vehicles available in the same cluster while chunks are to split clusters to avoid other clusters waiting too long, which will be explained later.

An example is shown in Fig. 1. The distances between vehicles satisfy the following formula (assuming $d_{chunk} < d_{com}$; otherwise, a cluster is composed of a chunk),

$$d_{AB} < d_{chunk} < d_{BC} < d_{com} < d_{CD}. \tag{5}$$

In this example, vehicle A, B and C are in the same cluster, while D is in another cluster. A is elected as the first cluster's head and D is the other cluster's head. A and B are in the same chunk while C is in another chunk.

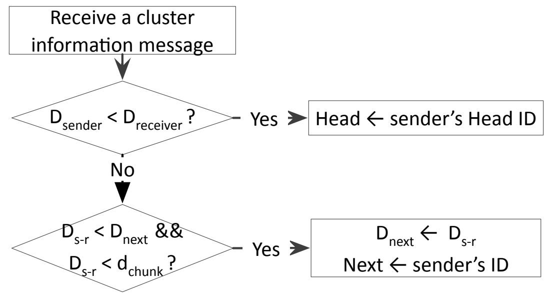


Fig. 2: The process of updating cluster information

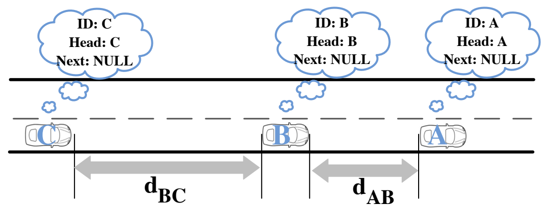
Cluster head selection and updating: While driving, vehicles periodically broadcast a *ClusterInfo* message, which includes its ID, header ID, current location, direction, and timestamp. Initially, every vehicle sets itself as the cluster head and exchanges messages with other vehicles to update the cluster information. After receiving *ClusterInfo* messages, a vehicle records two pieces of information: (1) the cluster head's ID, which can be directly found in the *ClusterInfo* message, and (2) the ID of the vehicle right behind this vehicle, which can be identified based on the *Coordinates* field in the received messages. We name such a vehicle, *Next vehicle*. The vehicle then updates the cluster information as shown in Fig. 2. In this figure, D_{sender} and $D_{receiver}$ are the distance from the sender and receiver to the intersection, respectively. D_{s-r} is the distance between the sender and receiver, which can be calculated from the received message. D_{next} is the distance between the receiver and the vehicle right behind the sender (the *Next vehicle*). Initially, the ID of the vehicle right behind the sender is set as $NULL$, and D_{next} is correspondingly set as infinite. They may be updated after receiving messages from neighbors, the procedure of which is explained later. If the sender is closer to the intersection than the receiver, the receiver updates its cluster head's ID as the sender's head. Otherwise, it means the sender is behind the receiver, and the receiver compares D_{s-r} with the saved D_{next}. If D_{s-r} is less than D_{next} and less than d_{chunk}, the ID of the vehicle right behind the sender is set as the sender's ID. If D_{s-r} is larger than D_{chunk}, it means that they are in different chunks, and the receiver does not record the sender as the *Next vehicle*.

We use Fig. 3 as an example to illustrate the cluster information updating process. We assume the distances between vehicles satisfy the following formula,

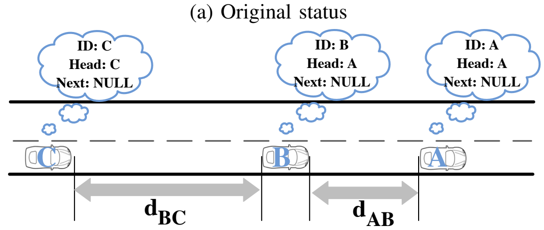
$$d_{AB} < d_{chunk} < d_{BC} < d_{com}. \tag{6}$$

Initially, the cluster head's ID in every vehicle is its own ID as shown in Fig. 3 (a). After vehicle A broadcasts its *ClusterInfo* message, vehicle B updates its *Head* to be A since A is closer to the intersection in Fig. 3 (b). After vehicle B broadcasts its *ClusterInfo* message, vehicle C updates its *Head* to be A

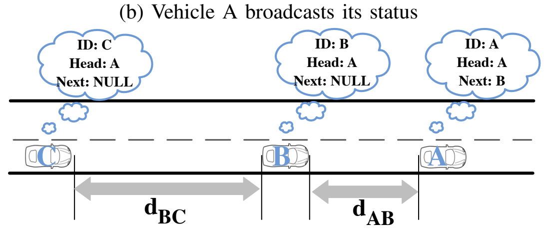
This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/JIOT.2018.2817459, IEEE Internet of Things Journal



(a) Original status



(b) Vehicle A broadcasts its status



(c) Vehicle B, C broadcast its status

Fig. 3: An example of updating cluster information

and vehicle A updates its *Next Vehicle* to be B in Fig. 3 (c). After vehicle C broadcasts its *ClusterInfo*, vehicle B will not updates its *Next vehicle* since $d_{BC} > d_{chunk}$, which is the same as Fig. 3 (c).

2) Intersection traffic management: The main objective of traffic management at intersections is to increase the overall efficiency and throughput, while maintaining the fairness to pass the intersections for vehicles. We aim to minimize their waiting time and number of stops. Generally speaking, to maximize the efficiency, we let vehicles group into clusters. When multiple clusters approach an intersection, the closest cluster obtains the right to pass the intersection first. Clusters which do not have crossing routes with this closest cluster can simultaneously pass the intersection. We say two clusters are conflicting if they have crossing routes. Meanwhile, to achieve the fairness, if a cluster has been waiting for a long enough time, the current passing cluster will yield the road right to that cluster.

Before presenting the detailed protocol, we define a set of vehicle states, and a collection of *Priority Negotiation* messages.

Vehicle states: We use a finite state machine to illustrate the DIMP protocol. In this protocol, a cluster is in one of the following states:

0: At state 0, a cluster has certain distance away from and is getting closer to the intersection. The cluster head will request the right to pass the intersection when it approaches the intersection.

1: a cluster starts braking in front of an intersection. When a cluster cannot pass the intersection directly, it starts braking.

2: a cluster completely stops and is waiting right in front of the intersection. When a cluster completely stops, it records the stop time. This state means the cluster cannot pass the intersection, and is waiting because of that another conflicting cluster is currently passing the intersection. At this state, cluster head periodically broadcasts a message to alert the passing cluster that there is a cluster waiting.

3: a cluster is ready to start moving towards the intersection. If the passing cluster announces that it is ready to yield the road right, clusters in state 2 move into state 3 and get ready to move. However, more than one clusters may be ready to move. Then they need to determine which one can move based on how long they have been waiting. If other clusters have been waiting for a longer time, this cluster will move back to state 2 and keep waiting. If no such a cluster exists, this cluster will wait for a certain time, start passing the intersection and move to state 4.

4: a cluster starts driving and passes an intersection. The cluster will record the time it starts. When the cluster holds the road right for long enough time and other clusters are waiting, this cluster will yield the road right.

Before we present how a cluster transits between the states, we introduce the types of messages exchanged. This set of messages is sent by a cluster head to vehicles in its cluster and other clusters.

Passing request: to request passing an intersection when the cluster approaches the intersection.

IAM waiting: to inform other clusters that the cluster brakes and is waiting for the road right.

Road right handover: to inform clusters waiting at the intersection that the road right is now yielded to others and the waiting clusters need to negotiate who to obtain the road right.

Head handover: to inform cluster members to change the cluster head to its *Next vehicle* after it passes an intersection.

Priority setup: to negotiate the sequence to pass the intersection among clusters, determined by clusters' waiting time at an intersection.

Time notification: to share the time at which the cluster just passes an intersection. The time also indicates when the intersection resource occupied by the current cluster will be released.

State transition: state 0–> state 4: When a cluster head approaches an intersection at state 0 and its distance to the intersection is less than $2 \times d_{CBD}$, it broadcasts a *passing request*. When its distance to the intersection is less than d_{CBD}, no other clusters are closer to the intersection, and no conflicting cluster is currently occupying the intersection, the cluster can move to state 4 and pass the intersection. The latter two conditions are satisfied when no other passing requests are received from clusters closer to the intersection and no head handover messages from the cluster currently passing the intersection are overheard.

state 0–> state 1: If a cluster at state 0 receives *passing request*, *head handover*, or *priority setup* from the heads of other conflicting clusters, the cluster may need to stop and move to state 1. If a *passing request* message is received from a cluster that has a crossing route with it, the cluster head compares its distance to the intersection with that of the sender. If it has a longer distance, it changes its state into 1 and slows down. Otherwise, it broadcasts a *passing request* immediately

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/JIOT.2018.2817459, IEEE Internet of Things Journal

to request passing the intersection. If a *head handover* message is overheard from a cluster having a crossing route with it, which means currently there is a cluster in the process of passing the intersection, the cluster changes its state into ① and slows down. If a *priority setup* message is received from a cluster having a crossing route with it, it means currently several clusters waiting at the intersection are negotiating who should pass the intersection, the cluster head changes its state into ① and slows down.

state ①–> state ②: After a cluster head fully stops in front of an intersection at state ①, it changes the state into ② and records the time of the complete stop. It periodically broadcasts an *IAM waiting* message to inform other clusters of its state.

state ②–> state ③: If a *road right handover* message is received at state ② or the cluster has been at state ② for longer than a threshold period without any messages received from other clusters, the head changes its state into ③ and broadcasts a *priority setup* message including its stop time.

state ①,②–> state ④: If a *head handover* message from a non-conflicting cluster is received at state ① or ②, it means the road right is obtained by a non-conflicting cluster. The current cluster obtains the right to pass the intersection and it changes its state into ④.

state ③–> state ②: Recall that the stop time is included in the *priority setup*. If a cluster head at state ③ receives a *priority setup* message, it compares its stop time with that in the message. The earlier the stop time is, the longer the waiting time is, and thus the cluster has a higher priority to pass the intersection. If its stop time is later than that in the message, it means there is another cluster which has been waiting for a longer period of time. So the cluster changes its state back to ② and keeps waiting.

state ③–> state ④: At state ③, if a cluster head does not receive any *priority setup* messages whose stop time is earlier than its, it means this cluster head has been waiting at the intersection for the longest time and the cluster can move. We specify that, at state ③, after waiting for T_3 without learning any cluster waiting for a longer time, the cluster moves into state ④ and starts passing the intersection. T_3 serves for two purposes: (1) wait for all the other clusters at state ③ which have been waiting for a shorter period time than this cluster to move back to state ②; (2) give the moving clusters enough time to stop in front of the intersection so this cluster can move without any collision. The calculation of T_3 needs how long each moving cluster needs to stop in front of the intersection. After presenting all the state transitions, we will illustrate how T_3 is calculated.

state ④–> state ①: If an *IAM waiting* message is received from a conflicting cluster, the cluster at state ④ checks whether it has occupied the intersection for longer than a threshold period of time. If so, the cluster head broadcasts a *road right handover* message to all the waiting cluster heads, so they can negotiate with each other who can pass the intersection through exchanging *priority setup* messages. Meanwhile, the vehicles inside the cluster which cannot safely stop will keep moving and pass the intersection and vehicles which can will stop. Specifically, the vehicles whose distance to the intersection is shorter than d_{CBD} will keep driving, and other vehicles will stop. The vehicle, which needs to stop and is closest to the intersection among vehicles which need to stop, becomes the new cluster head of the cluster. This is also the vehicle whose distance to the intersection is larger than d_{CBD} but is the closest to d_{CBD}. This new cluster head calculates the time that all the vehicles in front of it can safely pass the intersection, denoted by T_B, and then broadcasts T_B in a *time notification* message. Note that how T_B is calculated will be presented together with how T_3 is calculated. The cluster moves into state ① and brakes.

state ④–> state ⓪: When a vehicle passes an intersection at state ④, the cluster head checks whether it has a *Next vehicle*. The cluster head sends a *head handover* message if it has a *Next vehicle* in the chunk, then the *Next vehicle* becomes the new head at state ④; otherwise, it means the current chunk has passed the intersection. Whether there are still vehicles in the cluster does not affect the process. Even if the cluster has more vehicles, it should also be split by itself to give away the road right by broadcasting a *road right handover* message. Vehicles in the same cluster reset the cluster head's ID as itself upon receiving the *cluster handover* message and forwards the message to update cluster information. They stay moving and change their state into ⓪ to request passing the intersection again.

T_3**'s calculation**: As we mentioned previously, T_3 is the time a cluster waits at state ③ before it starts to drive through the intersection. It serves for two purposes: (1) give all the moving clusters enough time to stop in front of the intersection so the cluster which has been waiting for a long enough time can move without any collision; (2) wait for all the other clusters at state ③ which have been waiting for shorter period time than this cluster to move back to state ②. Objective (2) can be automatically satisfied when objective (1) is satisfied since mechanical movement/stopping takes much longer time than electronic communication and computation. In the following paragraphs, we focus on how to determine a time point that all the moving clusters are guaranteed to stop.

For each moving cluster to stop, we must allow all the vehicles which cannot stop safely to pass the intersection and only stop the remaining vehicles. All the vehicles whose distance to the intersection is larger than d_{CBD} can safely stop and others are allowed to pass. Here we estimate how long it takes for all the vehicles whose distance is less than d_{CBD} to pass the intersection and the road can be clear. For a vehicle A which will pass the intersection, we assume it will not brake nor speed up. To drive at its current speed v_a, it takes d_a/v_a for it to pass the intersection, in which d_a is its distance to the intersection. In a distributed fashion, once sends out *cluster handover* message, the cluster head of a moving cluster, assuming it is A, checks its distance to the intersection. If longer than d_{CBD}, it starts braking and calculates $T_B = d_a/v_a$. Then it broadcasts a *time notification* message including T_B announcing how long its cluster still needs to occupy the intersection. If shorter than d_{CBD}, then it moves on and sends its $NEXT$ vehicle B a $checkup$ message. Vehicle B will repeat the operation. The vehicle which will do the calculation and broadcast the *time notification* is the

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/JIOT.2018.2817459, IEEE Internet of Things Journal

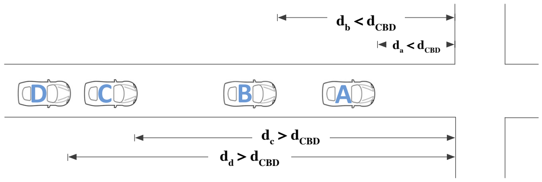


Fig. 4: An example of calculating T_B

first vehicle to stop in the cluster.

Take Fig. 4 as an example to illustrate this procedure. Since the distances between the intersection to vehicle A as well as vehicle B are shorter than d_{CBD}, vehicle A and B cannot brake under the comfort negative acceleration, and thus, they have to pass the intersection. Vehicle C is the closest vehicle to A and B, and its distance to the intersection is longer than d_{CBD}, so C brakes down and broadcasts a *time notification* message including T_B, which is equal to d_b divided by v_b.

When the cluster head with state 3 receives *time notification* messages from all conflicting clusters that are driving, it chooses the latest time among all. Note that, if this cluster head does not receive any *time notification* messages due to message loss or any other reason, it will set up T_3 to be the the maximum time T_{max} that a cluster needs to brake, which is calculated by v_{max}/a. Here v_{max} is the speed limit and a is the comfort negative acceleration. For a cluster waiting at an intersection with state 3, when the waiting time is later than T_3, it begins to pass the intersection. The calculation of T_3 is shown as follows,

$$T_3 = min\{T_{max}, max\{T_B^{received}\}\}, \tag{7}$$

where $T_B^{received}$ is the vector of all the received T_B. T_3 is set to the lesser of T_{max} and the maximum value of $T_B^{received}$.

The detailed protocol is presented in Algorithm 1.

Algorithm 1: Distributed Intersection Management Protocol

Notations:
$stopTime$: the time when the vehicle stops.
$lastTime$: the time when the last 'IAM waiting' is sent.
T_B: the time that the last vehicle in current cluster needs to pass the intersection.
T_3: the extra waiting time from state 3 to 4.

(1) Upon entering $state$ 0
if Distance to an intersection less than $2 \times d_{CBD}$ **then**
  Broadcast a 'passing request' message.
else if Distance to an intersection less than d_{CBD} **then**
  $state \leftarrow$ 4.
end if

(2) Upon entering $state$ 1
if Distance to an intersection is close to d_{CBD} **then**
  Brake down with the max decelerating acceleration.
  $state \leftarrow$ 2.
end if

(3) Upon entering $state$ 2
if Complete stop for the first time **then**
  $stopTime \leftarrow$ current time.
end if
if Timeout since $lastTime$ **then**
  Broadcast an 'IAM waiting' message.
  $lastTime \leftarrow$ current time.
end if
if Timeout since $stopTime$ with no messages **then**
  $state \leftarrow$ 3.
end if

(4) Upon entering $state$ 3
Broadcast a 'priority setup' message.
if Wait for another T_3 **then**
  $state \leftarrow$ 4.
end if

(5) Upon entering $state$ 4
Set speed as the max one.
if There is a *Next vehicle* **then**
  Broadcast a 'head handover' message.
else
  Broadcast a 'road right handover' message.
end if

(6) Upon receiving a 'passing request' message
if It is from a conflicting cluster **then**
  if $state$ is 0 **then**
    if $D_{sender} > D_{receiver}$ **then**
      Broadcast a 'passing request' message.
    else
      $state \leftarrow$ 1.
    end if
  end if
end if

(7) Upon receiving an 'IAM waiting' message
if It is from a conflicting cluster **then**
  if $state$ is 4 **then**
    if The driving cluster holds right right over a fixed time **then**
      Broadcast a 'road right handover' message.
    end if
  end if
end if

(8) Upon receiving a 'head handover' message
if It is from the same cluster **then**
  Update its cluster information.
else if It is from a conflicting cluster **then**
  if $state$ is 0 **then**
    $state \leftarrow$ 1.
  else if $state$ is 3 **then**
    $state \leftarrow$ 2.
  end if
else
  $state \leftarrow$ 4.
end if

(9) Upon receiving a 'road right handover' message
if It is from the same cluster **then**
  Update its cluster information.
else if It is from a conflicting cluster **then**
  if $state$ is 1 **then**
    Stop at the intersection.
    $state \leftarrow$ 3.
  else if $state$ is 2 **then**
    $state \leftarrow$ 3.
  end if
end if

(10) Upon receiving a 'priority setup' message
if It is from a conflicting cluster **then**
  if $state$ is 0 **then**
    $state \leftarrow$ 1.
  else if $state$ is 3 **then**
    if $stopTime$ is later than that in the message **then**
      $state \leftarrow$ 2.
    end if
  else if $state$ is 4 **then**
    Ask the nearest vehicle with a larger distance than d_{CBD} to brake.
    Broadcast a 'time notification' message with T_B.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/JIOT.2018.2817459, IEEE Internet of Things Journal

end if
end if
(11) Upon receiving a 'time notification' message
if $state$ is ④ **then**
Record the latest T_B.
end if

C. Lane Changes and Emergencies

In DIMP, a vehicle's cluster membership is determined by its current lane. In practice, vehicles may change lanes on a multi-lane road or stop for emergencies at any time. Our protocol provides two methods to handle the two scenarios. Lane change is handled by changing the cluster memberships. As designed in DIMP, vehicles are grouped into clusters, each including vehicles in the same lane. A vehicle needs to join another cluster if it switches to another lane. In our protocol, when a vehicle switches, it broadcasts a message to inform that it leaves the current cluster and joins a new cluster. After that, in the current cluster, if this vehicle is the head, it hands over its head status and current state to its *Next* vehicle; otherwise the current cluster just updates the cluster information and does not change its state. In the new cluster which this vehicle switches to, if the vehicle is closer to the intersection than the current head, it becomes a new head and maintain the current cluster's state; otherwise, it just informs the cluster members of its information. This also applies on the scenario when a cluster head separates from a cluster for unknown reasons. When the head plans to leave a cluster, its state will be passed to the new cluster head.

The emergencies are defined as staying in the wrong lane, such as a vehicle is to turn left while it is in the through lane at an intersection. When an emergency happens, the vehicle in the wrong lane waits for the cluster in the correct lane to get the road right and then changes its lane to pass the intersection. For example, if the current moving clusters to turn left are in the through lanes and the cluster head encounters an emergency, it informs the current moving clusters to stop and let other clusters negotiate for a new road right to pass the intersection. If the cluster in the left turn lane gets the road right, the emergency vehicle in the through lane looks for a chance to turn left and to pass the intersection. Under other unpredicted accidents and emergencies, our system, which is the worst case, can backwards to stop-and-go rules, just as in an all-way stop intersection.

V. Analysis and Evaluation

A. Evaluation Methodology

1) Objectives: In this section, we evaluate DIMP with the following two objectives:

- Testing the effectiveness of the protocol by measuring how early a vehicle can receive traffic coordination messages before entering an intersection;
- Evaluating the efficiency of the proposed protocol with the following two metrics: 1) The average time that vehicles need to wait at the intersection; 2) The average number of vehicles that wait at the intersection.

More specifically, for the first objective, we measure a vehicle's distance to an intersection when it receives the first coordination message from the intersecting roads. If a vehicle is able to receive the messages from other clusters before it can safely brake under the max negative acceleration, it means the messages in our protocol can be delivered to the vehicle in time without affecting driving during the process. For the second objective, we compare the performance of DIMP with other two protocols, virtual traffic lights (VTLs) [6], and traditional traffic lights. As mentioned in Section II, many traffic-light-less protocols are based on VTL to improve the performance regarding to safety, pedestrian passing or wireless connection, which do not reduce vehicles' waiting time in passing intersections. Except the VTL-based protocols, some other works based on V2V either mimic the all-way stop or consider one-lane intersections. As we all know, all-way-stop intersections are very inefficient [38] and vehicles with non-conflicting routes can pass the intersection at the same time. More importantly, they do not handle the emergencies and a control center is required in the worst case. Thus, they are not easy to be implemented in a multi-lane intersection without a remote center server. In these protocols for connected autonomous vehicles, vehicles' trajectories are required in advance and their acceleration and deceleration processes need to be completed in a high accuracy, which is hard for human drivers. Also the optimization consumes a lot of computation and changing lanes is not allowed because the constraints for lane-changing is not able to be linearized [30]. Compared to these protocols, VTL is a very flexible protocol, which elects a virtual intersection leader by wireless communication and the leader guides the other vehicles. So it is easy to be extended into multi-lane intersections and to define the exact yellow light time to clear the intersection. Specifically, in VTL, a vehicle needs to be elected as an intersection management leader and stops at an intersection to organize vehicles. The leader hands over its role to another vehicle if it has already been a leader over a certain time. Before that, the leader informs other vehicles to allow them enough time to brake. The VTL does not specify the details in vehicles' movement and grouping vehicles. To have a fair comparison, we enable all non-conflicting vehicles to pass the intersection simultaneously, same as that in DIMP. In traditional traffic lights, the light displays green, yellow, and red periodically. To have a fair comparison, the yellow lights period is the time for vehicles to brake from the speed limit.

2) Simulation scenario: For the first evaluation objective, we vary the speed v when a vehicle should start braking from 10 to 70 MPH. Note that, this speed determines from how

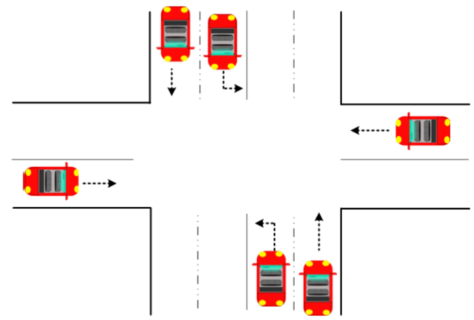


Fig. 5: Simulation scenario

far away from the intersection a vehicle has to start braking, so the driver will not feel uncomfortable, as we illustrate in Section IV-A. In the simulation, vehicles are generated by the simulator and enter the map from two vertical directions, left to right and down to up. Vehicles broadcast messages at a low frequency, one message per second, to reduce unnecessary messages and save bandwidth.

For the second evaluation objective, traffic are generated in two topology scenarios. One is shown in Fig. 5, which contains one intersection. Vehicles enter the map from four directions. There are two types of road segments at the intersection: both the left and the right road segments have two lanes, in and out lanes; both the up and the down road segments have four lanes, including one left turn lane each. The length of every lane is $500m$. The second scenario is called Manhattan grid topology [20]. In this scenario, there are four connected intersections and traffic is from four roads. In both scenarios, similar to previous works, we assume vehicles are driven under the Krauss Following Model [10] to ensure they do not crash on road. Vehicles can change speed from zero to the speed limit. In both scenarios, vehicles randomly enter the map following a Poisson process with parameter λ [39]. The cluster head may fail during driving on the road. If this happens, vehicles in the cluster re-elect another cluster head if they discover the head is disconnected for 3 seconds.

We implement the protocol in OMNeT++ [8] with SUMO [9] and Veins [10]. OMNeT++ is a simulator that provides the network simulation. SUMO provides the mobility of vehicles, and Veins provides the connection between the two parts. The parameters are listed in Table II. It specifies the settings in the wireless communication and traffic flows. In the second objective, the speed limit is set 13.41 m/s, which is equal to 30 MPH. It means a vehicle needs about 7 seconds to brake from the speed limit. Thus, we set the yellow light time 7 seconds. The final results are the averaged results of 10 times of running. No particular dissemination mechanisms are employed in our current simulation, but a series of protocols can be applied in our system [40]. Currently, every packet is sent twice to avoid packet loss. Acknowledgements may be employed in the future work.

B. Simulation Results

1) Protocol effectiveness: The average distance to an intersection when a vehicle receives the first message from the crossing road is shown in Table III. It shows that vehicle speed does not affect this average distance much. A vehicle can receive the stop notification or other messages on negotiating the passing order when it approaches the intersection as far as 358 meters. In addition, associated with the braking distance from Table I, the distance that the first message is received is much longer than the distance that the vehicle needs to brake under the comfort negative acceleration. It means the communication protocol can meet the requirement under the fastest vehicle. In addition, we can also see that d_{CBD} is always smaller than d_{com}, which is consistent with the assumption in the algorithm section.

TABLE II: Parameters in effectiveness measurement

Item	Value
Carrier frequency	5.89 GHz
Transmission power	10 mW
Minimum receive power	-89 dBm
Thermal noise	-110 dBm
Propagation model	Free space
MAC layer	IEEE 1609
Physical layer	IEEE 802.11p
Broadcasting frequency	1 per second
Speed limit	13.41 m/s
Maximum acceleration speed	1.0 m/s^2
Maximum deceleration speed	2.0 m/s^2
Vehicle length	5 m
Vehicle number	100
Time threshold during passing	30.0 s
T_{max}	7.0 s

TABLE III: Distance to the intersection under different speeds

Speed (MPH)	10	30	50	70
Distance (m)	359.5	359.5	359.5	359

2) Results in the random arrival model: Vehicles' waiting time is affected by vehicle density. As mentioned in Section V-A2, vehicle inter-arrival time follows Poisson distribution with parameter λ. With a smaller λ, vehicles are generated much more frequently by the simulator. For example, when λ is 5, every 5 seconds, a vehicle is generated; and when λ is 1, every 1 second, a vehicle is generated.

The average waiting time of vehicles under different λ is shown in Fig. 6. We can see that average waiting time in the three methods tends to increase when λ increases. The average waiting time for vehicles in DIMP is much less than that in VTL and traditional traffic lights, especially when the frequency is low. Specifically, when λ is 10, the average waiting time in VTL and traditional traffic lights are around 35 seconds, and DIMP is about 68.5% less, 11 seconds. The reason for the not-so-good performance in VTL is mainly because an intersection leader needs to be at the intersection in VTL and the protocol does not have a mechanism to detect vehicles from conflicting lanes when generating the first leader, which increases the waiting time. The reason for the slight fluctuation in the figure is that the predefined green light time may affect the vehicles' passing for different generating frequencies, which is out of the scope of this paper.

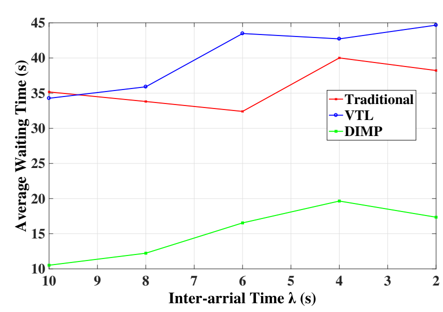


Fig. 6: Average waiting time under the random arrival model

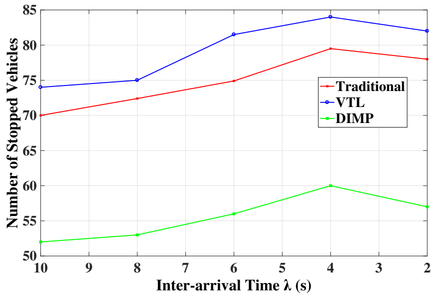


Fig. 7: Number of stopped vehicles under the random arrival model

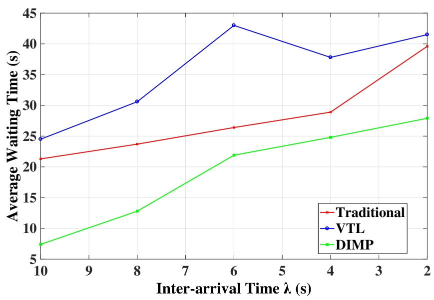


Fig. 8: Average waiting time under the uniform arrival model

The number of stopped vehicles in the three methods under different vehicle inter-arrival time is shown in Fig. 7. We consider a vehicle is a stopped vehicle if it is fully stopped. It shows that the number of stopped vehicles decreases with the increased vehicle generation frequency. DIMP outperforms the other two methods a lot. Specifically, when the vehicles are generated less frequently, the number of stopped vehicles in DIMP drops dramatically. The number in DIMP is less than 55 while that in both VTLs and traditional traffic lights is over 70 when λ is 10. The reason is that when a cluster head that has no *Next vehicles* passes the intersection in DIMP, it broadcasts a message to inform other clusters to pass, so the vehicles in other clusters do not need to stop at the intersection. However, in traditional traffic lights, when the red light is on, vehicles have to stop even there are no vehicles in the other lane. In VTL, the leader has to wait for over a fixed time duration, and thus the vehicles after it have to stop in front of the intersection. The reason for the slight fluctuation in the figure is the same as the reason in the previous figure.

3) Impact of vehicle arrival model: In addition to random arrivals, there are two other models to describe the arrival of vehicles, uniform model and platooned model, which are more suitable for the coordinated signalized systems [39]. Under the uniform arrival model, the experimental results are shown in Fig. 8. From the figure, we can see that our protocol outperforms the other two mechanisms under all generation frequencies. Specifically, when λ is 10 seconds, the waiting time in our protocol is about 7 seconds while that in VTL is about 25 seconds and that in traditional traffic lights is about 22 seconds.

In the platoon arrivals, we keep the interval between vehicles in a platoon as 2 seconds with 10 vehicles in a platoon, and change the interval between platoons from 2 to 10 seconds. The experimental results are shown in Fig. 9. From the figure, we can see that our protocol outperforms the other two mechanisms under all generating frequencies. Especially when the frequency is 10, the waiting time in our protocol is about 5 seconds while that in VTL is over 25 seconds and that in traditional traffic lights is about 20 seconds. The waiting time in our protocol is more than 75% less than the other two.

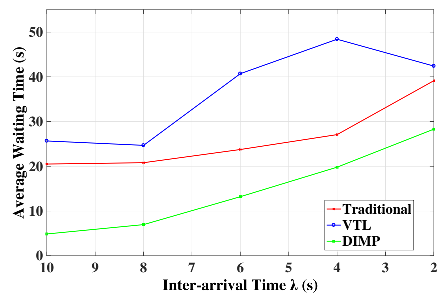


Fig. 9: Average waiting time under the platoon arrival model

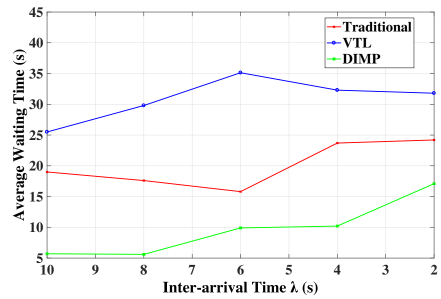


Fig. 10: Average waiting time in the Manhattan grid topology

4) Results in the Manhattan grid topology: We test our protocol on a Manhattan grid topology in traffic roads. Traffic are allowed to go straight only in this test. The average waiting time for vehicles under different vehicle inter-arrival time is shown in Fig. 10. We can see that average waiting time in the three methods tends to increase when vehicle generating frequency increases. The average waiting time for vehicles in DIMP is less than that in VTL and traditional traffic lights, especially when the frequency is low. Specifically, when λ is 10, the average waiting time in VTL and traditional traffic lights are around 20 seconds, and DIMP is about 65% less. The reason for the poor performance in VTL is mainly because an intersection leader needs to be at the intersection in VTL and the protocol does not have a mechanism to detect vehicles from conflicting lanes when generating the first leader, which increases the waiting time. VTL does not detect whether there are vehicles from crossing roads, which makes it even worse than traditional traffic lights in a grid topology.

VI. Conclusion

In this paper, a distributed VANET protocol (DIMP) is proposed to manage vehicles to pass intersections. Our protocol can achieve fairness because the first coming cluster passes first and gives away the road rights to other clusters when it occupies the intersection over the maximum time; it can achieve drivers' safety and comfort because we propose a comfort braking distance and force non-conflicting clusters only to pass the intersection at the same time; it can also achieve the efficiency because vehicles are grouped into clusters and a

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/JIOT.2018.2817459, IEEE Internet of Things Journal

cluster is splittable to balance the waiting time and stopping times. Performance evaluation shows that the protocol can save over 70% time in the average waiting time compared to other schemes. It also outperforms existing mechanisms in waiting time, number of stopped vehicles and traveling time.

REFERENCES

[1] "Vehicular communication systems," http://en.wikipedia.org/wiki/Vehicular_communication_systems, 2017.

[2] N. Maslekar, M. Boussedjra, J. Mouzna, and H. Labiod, "Vanet based adaptive traffic signal control," in *Vehicular Technology Conference (VTC Spring), 2011 IEEE 73rd*. IEEE, 2011.

[3] V. Gradinescu, C. Gorgorin, R. Diaconescu, V. Cristea, and L. Iftode, "Adaptive traffic lights using car-to-car communication," in *Vehicular Technology Conference, 2007. VTC2007-Spring. IEEE 65th*. IEEE, 2007.

[4] B. Zhou, J. Cao, X. Zeng, and H. Wu, "Adaptive traffic light control in wireless sensor network-based intelligent transportation system," in *Vehicular Technology Conference Fall (VTC 2010-Fall), 2010 IEEE 72nd*. IEEE, 2010.

[5] J. Tian, X. Liang, and G. Wang, "Deployment and reallocation in mobile survivability-heterogeneous wireless sensor networks for barrier coverage," *Ad Hoc Networks*, vol. 36, pp. 321–331, 2016.

[6] M. Ferreira, R. Fernandes, H. Conceição, W. Viriyasitavat, and O. K. Tonguz, "Self-organized traffic control," in *Proceedings of the seventh ACM international workshop on VehiculAr InterNETworking*, 2010.

[7] T. Neudecker, N. An, O. K. Tonguz, T. Gaugel, and J. Mittag, "Feasibility of virtual traffic lights in non-line-of-sight environments," in *Proceedings of the ninth ACM international workshop on Vehicular inter-networking, systems, and applications*, 2012.

[8] A. Varga, "Discrete event simulation system," in *Proc. of the European Simulation Multiconference (ESM'2001)*, 2001.

[9] D. Krajzewicz, G. Hertkorn, C. Rössel, and P. Wagner, "Sumo (simulation of urban mobility)," in *Proc. of the 4th middle east symposium on simulation and modelling*, 2002, pp. 183–187.

[10] C. Sommer, R. German, and F. Dressler, "Bidirectionally coupled network and road traffic simulation for improved ivc analysis," *Mobile Computing, IEEE Transactions on*, vol. 10, no. 1, 2011.

[11] L. Qi, M. Zhou, and W. Luan, "A two-level traffic light control strategy for preventing incident-based urban traffic congestion," *IEEE Transactions on Intelligent Transportation Systems*, 2016.

[12] M. A. Salman, S. Ozdemir, and F. V. Celebi, "Fuzzy logic based traffic surveillance system using cooperative v2x protocols with low penetration rate," in *Networks, Computers and Communications (ISNCC), 2017 International Symposium on*. IEEE, 2017, pp. 1–6.

[13] Y. Li, E. Canepa, and C. Claudel, "Optimal traffic control in highway transportation networks using linear programming," in *Control Conference (ECC), 2014 European*. IEEE, 2014.

[14] Y. Dujardin, D. Vanderpooten, and F. Boillot, "A multi-objective interactive system for adaptive traffic control," *European Journal of Operational Research*, vol. 244, 2015.

[15] X. Liang and G. Wang, "A convolutional neural network for transportation mode detection based on smartphone platform," in *2017 IEEE 14th International Conference on Mobile Ad Hoc and Sensor Systems (MASS)*. IEEE, 2017, pp. 338–342.

[16] X. Liang, X. Du, G. Wang, and Z. Han, "Deep reinforcement learning for traffic light control in vehicular networks," *IEEE Transactions on Vehicular Technology*, 2018.

[17] T. Neudecker, N. An, and H. Hartenstein, "Verification and evaluation of fail-safe virtual traffic light applications," in *Vehicular Networking Conference (VNC), 2013 IEEE*. IEEE, 2013, pp. 158–165.

[18] W. Viriyasitavat, "Self-organized intersection control management for motorized and non-motorized traffic," in *Consumer Communications and Networking Conference (CCNC), 2015 12th Annual IEEE*. IEEE, 2015, pp. 437–442.

[19] J. Shi, C. Peng, Q. Zhu, P. Duan, Y. Bao, and M. Xie, "There is a will, there is a way: A new mechanism for traffic control based on vtl and vanet," in *High Assurance Systems Engineering (HASE), 2015 IEEE 16th International Symposium on*. IEEE, 2015, pp. 240–246.

[20] O. K. Tonguz and W. Viriyasitavat, "A self-organizing network approach to priority management at intersections," *IEEE Communications Magazine*, vol. 54, no. 6, pp. 119–127, 2016.

[21] L.-D. Chou, T.-Y. Shen, C.-W. Tseng, Y.-J. Chang, and Y.-W. Kuo, "Green wave-based virtual traffic light management scheme with vanets," *International Journal of Ad Hoc and Ubiquitous Computing*, vol. 24, no. 1-2, pp. 22–32, 2017.

[22] S. R. Azimi, G. Bhatia, R. R. Rajkumar, and P. Mudalige, "Vehicular networks for collision avoidance at intersections," *SAE International Journal of Passenger Cars-Mechanical Systems*, vol. 4, no. 2011-01-0573, pp. 406–416, 2011.

[23] R. Azimi, G. Bhatia, R. Rajkumar, and P. Mudalige, "Intersection management using vehicular networks," SAE Technical Paper, Tech. Rep., 2012.

[24] M. Pasin, B. Scheuermann, and R. F. de Moura, "Vanet-based intersection control with a throughput/fairness tradeoff," in *IFIP Wireless and Mobile Networking Conference (WMNC), 2015 8th*. IEEE, 2015, pp. 208–215.

[25] B. Yang, R. Zheng, and K. Nakano, "Application of in-vehicle traffic lights for improvement of driving safety at unsignalized intersections," in *Intelligent Vehicles Symposium (IV), 2015 IEEE*. IEEE, 2015, pp. 628–633.

[26] A. Bazzi, A. Zanella, and B. M. Masini, "A distributed virtual traffic light algorithm exploiting short range v2v communications," *Ad Hoc Networks*, vol. 49, pp. 42–57, 2016.

[27] A. I. M. Medina, N. van de Wouw, and H. Nijmeijer, "Cooperative intersection control based on virtual platooning," *IEEE Transactions on Intelligent Transportation Systems*, 2017.

[28] 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.

[29] F. Zhu and S. V. Ukkusuri, "A linear programming formulation for autonomous intersection control within a dynamic traffic assignment and connected vehicle environment," *Transportation Research Part C: Emerging Technologies*, vol. 55, pp. 363–378, 2015.

[30] L. Chen and C. Englund, "Cooperative intersection management: a survey," *IEEE Transactions on Intelligent Transportation Systems*, vol. 17, no. 2, pp. 570–586, 2016.

[31] J. B. Kenney, "Dedicated short-range communications (dsrc) standards in the united states," *Proceedings of the IEEE*, vol. 99, no. 7, 2011.

[32] J. Levinson, M. Montemerlo, and S. Thrun, "Map-based precision vehicle localization in urban environments." in *Robotics: Science and Systems*, vol. 4. Citeseer, 2007, p. 1.

[33] S. Sivaraman and M. M. Trivedi, "Integrated lane and vehicle detection, localization, and tracking: A synergistic approach," *IEEE Transactions on Intelligent Transportation Systems*, vol. 14, no. 2, pp. 906–917, 2013.

[34] X. Du and K. K. Tan, "Comprehensive and practical vision system for self-driving vehicle lane-level localization," *IEEE transactions on image processing*, vol. 25, no. 5, pp. 2075–2088, 2016.

[35] X. Gao, J. Tian, X. Liang, and G. Wang, "Arpp: An augmented reality 3d ping-pong game system on android mobile platform," in *Wireless and Optical Communication Conference (WOCC), 2014 23rd*. IEEE, 2014, pp. 1–6.

[36] Z. Wu, Y. Liu, and G. Pan, "A smart car control model for brake comfort based on car following," *Intelligent Transportation Systems, IEEE Transactions on*, vol. 10, no. 1, 2009.

[37] L. Cheng, B. E. Henty, D. D. Stancil, F. Bai, and P. Mudalige, "Mobile vehicle-to-vehicle narrow-band channel measurement and characterization of the 5.9 ghz dedicated short range communication (dsrc) frequency band," *IEEE Journal on Selected Areas in Communications*, vol. 25, no. 8, 2007.

[38] R. Elvik, T. Vaa, A. Hoye, and M. Sorensen, *The handbook of road safety measures*. Emerald Group Publishing, 2009.

[39] R. Roess, E. Prassas, and W. McShane, "Traffic engineering. 3rd," 2004.

[40] S. M. Bilal, C. J. Bernardos, and C. Guerrero, "Position-based routing in vehicular networks: A survey," *Journal of Network and Computer Applications*, vol. 36, no. 2, pp. 685–697, 2013.