Routing Algorithm and Traffic Light Control Based on Vehicular Delay-Tolerant Networks

Haiping Huang[1, 2, 3], Dan Sha[1, 2], Yongcan Zhang[2, 4], and Pengfei Li[1, 2]
[1] College of Computer, Nanjing University of Posts and Telecommunications, Nanjing 210003, China
[2] Jiangsu High Technology Research Key Laboratory for Wireless Sensor Networks, Nanjing 210003, China
[3] College of Computer Sci. and Tec., Nanjing University of Aeronautics and Astronautics, Nanjing 210016, China
[4] Ericsson (China) Communications Co. LTD, Shanghai 201822, China
Email: hhp@njupt.edu.cn; hello_shadan@163.com; zyclyy@sina.cn; skynets@yeah.net

Abstract —There are problems of low delivery radio and high transmission delay in Vehicular Delay-Tolerant Networks (a.b. VDTN. To address these problems, this paper proposes a Vehicular Delay-tolerant Network routing algorithm based on Contention (a.b. VDNC). VDTN algorithm consists of three strategies: the intersection selection strategy based on the Manhattan distance and traffic information, the first competitive strategy based on the direction of movement and the second competitive strategy based on the position information. Moreover, traffic light also plays an important role to ease traffic congestion and reduce transmission delay. So an Adaptive Traffic-light Control algorithm based on Green-Computing (a.b. ATCG) is proposed. ATCG algorithm involves the following two parts: the calculation of the optimal sequence and that of the recommended speed. Simulation results show that compared to other traditional DTN (Delay-Tolerant Networks) protocols, VDNC algorithm has a higher delivery rate and lower transmission delay. And meanwhile, ATCG algorithm enables vehicles pass through the intersection with fewer stoppages and shorter waiting time. At the same time, the carbon dioxide emission is also able to achieve a minimum for the purpose of environmental protection. Consequently, the collaboration between VDNC and ATCG can further improve the traffic efficiency in VDTN.

Index Terms—Delay tolerant network, intelligent traffic light control, green computing, routing algorithm, vehicular networks, wireless sensor network

I. INTRODUCTION

Due to the disadvantages of current vehicular traffic such as the curing time of traffic light, the behindhand traffic control method, vehicle signal disconnection between vehicles and vehicles as well as vehicles and base-stations and, serious urban vehicle congestion, researchers pay more attention to Intelligent Transportation System (ITS) which is based on Vehicular Delay Tolerant Networks (VDTN). Therein to, the severe packet loss and link disruption have become challenges in Vehicular Networks. To address these challenges, researchers design a series of delay tolerant routing algorithms. However, many classic routing algorithms in Delay Tolerant Networks (DTN), such as Epidemic, Prophet, etc., have the defects of low delivery rate and high transmission delay, so it is difficult to meet the demands of urban transport. Therefore, this paper is committed to propose a novel routing protocol applied to VDTN. In addition, it also needs to design a set of intelligent traffic light control strategies based on the novel routing protocol to relieve the traffic pressure and decrease transmission delay. With the aid of wireless sensor nodes, the traditional curing time mode of traffic light control achieves the strict control, but lacks of efficiency. Therefore, the adaptive intelligent traffic light control system has become another focus for recent years, and it also utilizes sensor nodes to realize more humanized transport service. Meanwhile, automobile exhaust and other greenhouse gases have been involved in every aspect of people's life, which indicates that Green Computing has become a global strategic theme. Fortunately, the adaptive traffic light control system also can restrict automobile exhaust emissions effectively.

The main contributions of this paper include:

(1) We present a model of intelligent transportation system, including the vehicular delay tolerant network model, the traffic light control model, the carbon dioxide emissions model and the smooth travelling model.

(2) We propose a Vehicular Delay-tolerant Network routing algorithm based on Contention (a.b. VDNC), including the intersection selection strategy based on the Manhattan distance and traffic information, the first competitive strategy based on the direction of movement and the second competitive strategy based on the position information. Meanwhile, we compare VDNC with traditional algorithms in DTN, and simulation results show that our proposal has higher delivery radio and more desirable transmission delay.

(3) We design an Adaptive Traffic-light Control algorithm based on Green-Computing (a.b. ATCG), including the branch and bound algorithm and the

---

Manuscript received August 23, 2015; revised March 1, 2016.

This work was supported by the National Natural Science Foundation of P. R. China (No. 61170065, 61202355, 61373138), Natural Science Key Fund for Colleges and Universities in Jiangsu Province (No.12KJA520002), the Key Research and Development Program of Jiangsu Province (Social Development Program, No.BE2015702), Postdoctoral Foundation (No. 2015M570468), the Sixth Talent Peaks Project of Jiangsu Province (No. DZXX-017), Science and Technology Innovation Fund for Postgraduate Education of Jiangsu Province (No. KYLX15_0853).

Corresponding author email: hhp@njupt.edu.cn

doi:10.12720/jcm.11.3.305-316

optimal sequence algorithm. ATCG can calculate the recommended speed in three phase modes to avoid switching the speed frequently and simultaneously reduce CO_2 emissions. Finally, simulation results show that ATCG algorithm is superior to Fuzzy algorithm in the average waiting time, the average travel time and the CO_2 emissions.

The rest of this paper is organized as follows. Section II gives a brief review on the related work. Section III presents our system model, including the vehicular delay tolerant network model, the traffic light control model, the carbon dioxide emissions model and the smooth travelling model. In Section IV, we describe the proposed routing protocol VDNC in details. Section V carries out the communication scheme between vehicles and vehicles, as well as vehicles and base-stations, and calculates the recommended speed. The simulation parameters and results are illustrated in Section VI. Finally, Section VII provides our conclusions and the future work.

II. RELATED WORK

Currently, vehicle delay tolerant network routing technology and intelligent traffic light control technology have been considerably studied.

A. *Routing Algorithms*

Firstly, we analyze routing algorithms in VDTN. Epidemic routing [1] is the earliest one among all replication routing algorithms. In Epidemic, every node sends all messages it receives to all neighboring nodes within its communication range. Therefore, it achieves very high data delivery ratio, but degrades the network performance and wastes energy resources. ProPhet routing [2] takes advantage of historical information to reduce network overhead, but it is still insufficient in the handling of congestion. In First Contact (FC) routing [3], a message is forwarded along an edge chosen randomly among all the current contacts. If all edges are currently unavailable, the message waits for an edge to become available and is assigned to the first available contact. Spray-and-Wait/Focus (SW) routing [4]-[5] constrains the number of copies to reduce the energy consumption and improve the network performance. Zhu *et al.* [6] predict the interval when vehicles meet with each other on the basis of Markov Chain and determine the next hop according to the interval when the neighbor vehicle encounters the target vehicle. Furthermore, Zhu *et al.* [7] adopt the same delay estimation method, as the same with [6], further tap the social dimension of mobility, and construct the ego-contact diagram to transfer messages when the current vehicle is lack of delay estimation from the neighbor vehicle to the target vehicle. Holger *et al.* [8] propose a Contention-Based Forwarding method which can select the most appropriate node as a forwarder and suppresses other node’s forwarding. But the method can only be applied to one-dimensional scene. Tournoux *et al.* [9] put forward RECOR routing, which is a centralized resource-constrained DTN routing protocol and can transmit the request by a number of "Store-Carry-Forward" paths to conform to the restrictions of node storage and transmission links. Xu *et al.* [10] propose a data forwarding model based on vehicle's path and encounter probability, but historic information can only predict, not determine the future status. Taking the delay, cache and congestion into account, Zeng *et al.* [11] utilize Nash Q-learning method to optimize energy efficiency, and meanwhile design DRSS routing which is suitable for green VDTN. Wei *et al.* [12] propose novel semantic models for DTN traffic prediction, design a traffic prediction scheme based routing algorithm and explain how the routing performance is affected by the availability of traffic prediction. Tayal *et al.* [13] use different communication ranges in proposed protocol for public and private vehicles so that packet is delivered to farther distance. Nasir *et al.* [14] propose a position-based Delay Tolerant Network (DTN) routing protocol in VANET for the highways. The solution is designed for partitioned environment which suffers from frequent network disconnections, and it presents the most suitable next hop selection mechanism through a filtration process. The filtration process utilizes node position, current direction, speed and the predicted direction that are acquired by Direction Indicator Light (DIL). To the best of our knowledge, little has been done to understand the performance of DTNs under realistic settings involving the interplay of diverse factors such as bundle fragmentation, scheduling, and buffer spacing. So Mahendran *et al.* [15] take Queueing Petri Nets (QPNs) as a modeling framework to study the performance of DTN routing and develop QPN models for DTNs. The complete QPN model considers a number of realistic factors that impact performance such as finite buffer space, finite link bandwidth, bundles with different priorities and intra-scheduling delays arising due to different levels of the memory hierarchy at the nodes.

B. *Traffic Control Systems*

Secondly, we analyze current intelligent traffic control systems. Pappis [16] describes an artificial intelligence system which is based on fuzzy logic control for a single intersection. Balaji *et al.* [17] propose a distributed multi-agent traffic light control system to reduce travel time and delay, which uses the data communication between an agent and another to optimize green time. Zhou *et al.* [18] suggest an adaptive algorithm to determine traffic light cycle sequence and duration based on real-time traffic data such as vehicle detection data. Abdulhai *et al.* [19], [20] present a traffic light control algorithm based on Q-learning method which is applied to 50 intersections, where each intersection can be seen as a proxy, and then the whole network forms a multi-proxies system. For special application scenarios, Leu *et al.* [21] propose an intelligent traffic light control system for an ambulance to find a neighboring hospital quickly. El-Tantawy *et al.* [22] describe two different adaptive traffic light control modes:

independent mode and cooperative mode. In independent mode, each control module works independently in an intersection, and in collaborative mode, each controller collaborates with the adjacent one. Chiou *et al*. [23] propose a distributed genetic fuzzy logic controller, where they treat the length of vehicular queue and the volume of traffic as a state variable respectively, and the green duration as a control variable to obtain a minimum vehicle delay. Mahler *et al*. [24] introduce the prediction model of signal phase and use historical data and real-time phase data to execute a planning algorithm in order to calculate the optimal speed, thereby enhancing energy efficiency. Araghi *et al*. [25] study three traffic light control methods, including Q-learning, neural network and fuzzy logic system, each of which is significantly superior to conventional fixed signal control methods.

In particular, taking Green Computing into consideration, Alsabaan *et al*. [26] list the current communication methods used in ITS, as well as the quantitative model of CO_2 emission.

III. SYSTEM MODEL

In this section, some assumptions about the system model are considered. And they are presented separately in the vehicular delay tolerant network model, the traffic light control model, the carbon dioxide emissions model and the smooth travelling model.

A. Vehicular Delay Tolerant Network Model

The Manhattan Model is adopted in VDTN, and the Manhattan Distance denotes the distance between one vehicle and another. We can suppose that every vehicle is equipped with GPS positioning and navigation system (or sensor positioning system) to get its own location and the information of the actual road with an electronic map. At the same time, every vehicle can be treated as a node in the wireless sensing network. So we use an undirected graph $G = G(V, E)$ to represent the Vehicular Delay Tolerant Network Model. In the graph, the vertex set V is the set of intersections, and the edge set E is the set of streets between one intersection and another. So vehicular node moves from one vertex to another. For $e(i, j) \in E$, we define $md(i, j)$ and $tf(i, j)$, where $md(i, j)$ denotes the Manhattan Distance from one vehicular node i to another node j, and $tf(i, j)$ represents traffic conditions between node i and node j. For example, the number of vehicles that pass by within a unit time can be used to represent $tf(i, j)$.

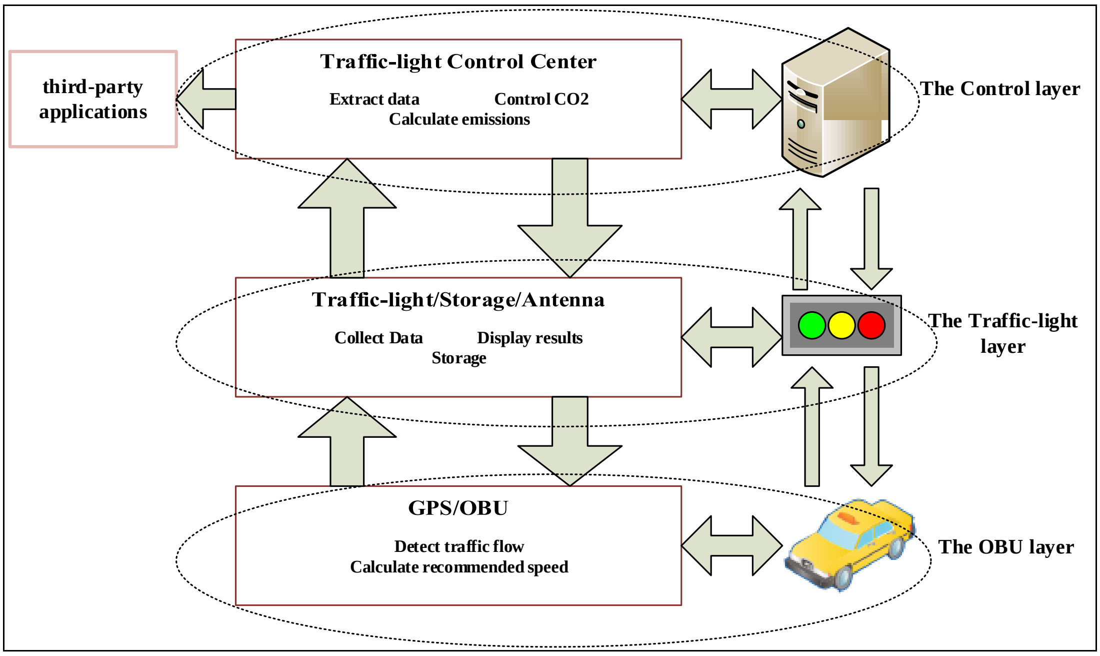


Fig. 1. Three-layers open traffic light control model

B. Traffic Light Control Model

We design a three-layers Open Traffic Light Control Model, as shown in Fig. 1.

The first layer is to collect traffic information, transmit messages, receive traffic light signals and calculate the recommended speed. The vehicle information will be available through GPS or sensor positioning device. OBU (On Board Unit) device interacts with traffic light in the second layer and calculates the recommended speed for the driver as a reference.

The second layer is to receive and store traffic control information and transmit results to OBU.

The third layer is responsible for data processing, divided into three parts: data extraction, traffic light control and an open interface (or the third-party applications).

C. Carbon Dioxide Emissions Model

Taking the realities of urban roads into account, we use the method in [27] to estimate carbon dioxide emissions of vehicles. The Carbon Dioxide Emissions Model can be described as the following two equations:

$$E = K_c(0.3T + 0.028L + 0.056A_{ee}) \tag{1}$$

$$Aee = \sum_{k=1}^{K} \sigma_k (v_k^2 - v_{k-1}^2) \tag{2}$$

where the meanings of symbols can be listed as follows:

E: The amount of carbon dioxide emissions (g)

K_c: The coefficient of relationship between gasoline consumption and carbon dioxide emissions

L: Travelling distance (m)

T: Travelling time for the length of L (s)

A_{ee}: Accelerating energy equivalent (m^2/s^2)

v_k: Velocity at time k (m/s)

σ_k: σ_k=1 if $v_k > v_{k-1}$ or σ_k =0

D. *Smooth Travelling Model*

In order to reduce carbon dioxide emissions, we use the following two ways to enable vehicles to drive more smoothly:

(1) We take control of traffic light cycle to reduce the average waiting time in intersection, and reduce the amount of carbon dioxide emissions during idling. In order to evacuate vehicles in the waiting queue, the order of vehicles must be assigned. And then it becomes a sequential decision problem. This paper adopts the branch and bound method to deal with this problem.

(2) A recommended speed is supplied to drivers before they enters intersection so as to reduce carbon dioxide emissions during travelling.

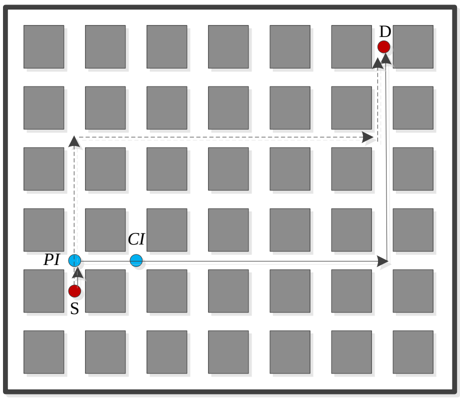


Fig. 2. Manhattan model

IV. VEHICULAR DELAY-TOLERANT NETWORK ROUTING ALGORITHM BASED ON CONTENTION

A. *Intersection Selection Strategy Based on the Manhattan Distance and Traffic Information*

The header of message contains two symbols, where one represents the next intersection (marked with *CI*), and the other represents the current intersection that the message just leaves (marked with *PI*), as shown in Fig. 2. Furthermore, the gray squares represent buildings, white portions indicate streets, S denotes the source node and D represents the destination node in Fig. 2. When the vehicular node goes into the area of the intersection during the process of forwarding messages, intersection selection is needed.

We exploit Dijkstra algorithm to compute the Manhattan distance *md*(*i*, *j*) between the forwarder and the destination, and then the shorter path is chosen. If two or more paths have the same Manhattan distance, their respective property *tf*(*i*, *j*) would be compared. As shown in Fig. 2, the Manhattan distance of the solid path and that of the dotted path are the same. So it requires three cases discussed.

1) If $\rho \geq tf\,(i, j)_{dotted} \geq tf\,(i, j)_{solid}$, (where $\rho \geq 0$, ρ can be set according to the specific scene), then the preferred path for message forwarding is the solid one.

2) If $tf\,(i, j)_{solid} \geq \rho \geq tf\,(i, j)_{dotted}$, then the preferred path for message forwarding is the solid one.

3) If $tf\,(i, j)_{solid} \geq tf\,(i, j)_{dotted} \geq \rho$, then the preferred path for message forwarding is the solid one.

B. *First Competitive Strategy Based on the Direction of Movement*

According to the location and moving direction of the forwarder, the first competitive strategy is classified into the following two cases.

(1) Forwarder is located in the street. After a message is broadcasted to the neighbor, the neighbor detects whether the moving direction of forwarder is the same with its own. If matched, the message is cached and treated as a candidate for the second competition. If not, the message is discarded, not to take participate in the second competition, and the first competition is over.

(2) Forwarder is located in the intersection. After a message is broadcasted to the neighbor, the neighbor detects whether its moving direction is near or far from the current intersection *CI*. If near, the message is cached and treated as a candidate for the second competition. If far, the message is discarded, not to take participate in the second competition, and the first competition is over.

C. *Second Competitive Strategy Based on the Information of Position*

The forwarder broadcasts messages to its neighbors, and the header of the message consists of the location information of the forwarder, the current intersection (*CI*) and the Previous Intersection (*PI*). Candidates that win in the first competition will take participate in the second competition based on the location information. We set a timer *t* for each message in terms of the location information in the header of this message. Closer to *CI*, the smaller *t* is. The vehicle node whose timer *t* expires wins the second competition to become the next hop and then it broadcasts messages, while suppressing other nodes involved in the second competition to forward the message. Those nodes involved in the second competition whose timers do not expire receive the repeated message, which indicates some other node wins the competition. Consequently the failure nodes discard incoming message, delete the message from the cache and cancel the timer.

Based on the location information of the forwarder, the second competitive strategy is divided into two cases.

1) Forwarder is located in the street

The forwarder *p* is located in the street, and its communication range covers *PI*, as shown in Fig. 3 (a). The solid line represents a circle whose center is *p*, and whose radius is *p*'s normal communication range *r*, and nodes within the solid circle can receive the message from *p*. According to the first competitive strategy based on the direction of movement, only nodes whose moving direction is the same with *p*'s can cache the message, and the remaining nodes will discard the message. The dotted line represents a circle whose center is *CI*, and whose radius is *md* (*CI*, *p*), and nodes' Manhattan distances within the dotted circle are smaller than *md* (*CI*, *p*). Thus, nodes in the public area (i.e. shaded area) between the two circles are closer to *CI*, and will become competitive nodes.

Nodes in the public area take participate in the first competition based on the direction of movement, if win, their respective timer *t* would be calculated as shown in (3).

$$t = T(1 - md(s,p)/r) \tag{3}$$

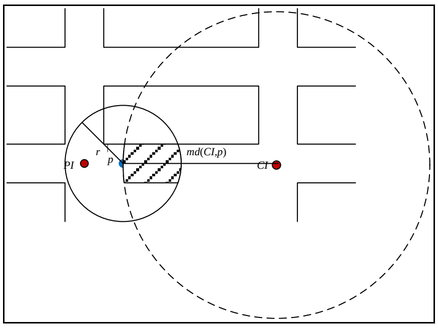
where *T* is the maximum delay, which can be set according to [5], *r* is the normal communication range of *p*, *p* is the forwarder, *s* is the node that wins in the first competition and is located in the public areas of two circles and *md*(*s*,*p*) is the Manhattan distance between *s* and *p*.

The forwarder *p* is located in the street, and its communication range doesn't cover *PI* or *CI*, as shown in Fig. 3 (b). The meaning of each symbol is the same with that in Fig. 3 (a). In this case, forwarding strategy is also the same with that in Fig. 3 (a).

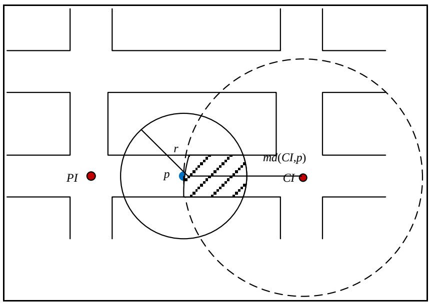
The forwarder *p* is located in the street, and its communication range covers *CI*, as shown in Fig. 3 (c). The meaning of each symbol is the same with that in Fig. 3 (a). The public area (i.e., shaded area) in Fig. 3 (c) is divided into two parts by *CI*, and the way to calculate the timer *t* is different on both sides of *CI*. For the left half, *t* is calculated according to (3), and then for the right half, *t* is calculated as shown in (4).

$$t = T(1 + md(s,CI)/r) \tag{4}$$

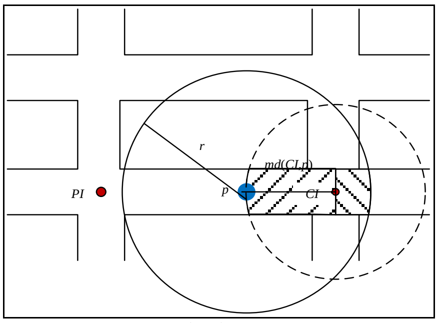
where *CI* is the current intersection and *md*(*s, CI*) is the Manhattan distance between *s* and *CI*.



(a) *p*'s communication range covers *PI*



(b) *p*'s communication range doesn't cover *PI* or *CI*



(c) *p*'s communication range covers *CI*

Fig. 3. *p* is located in the street

2) Forwarder is located in the intersection

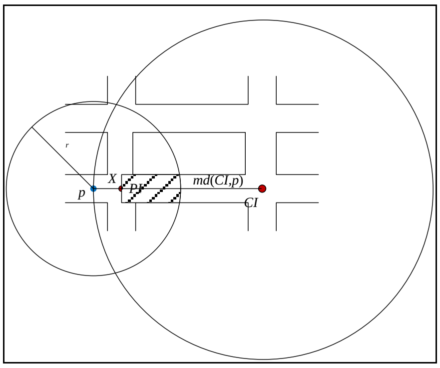
The forwarder *p* is located in the region of intersection (when the forwarder is near to the intersection, due to the obstruction of buildings its communication range can't achieve *r*, and then the area where any node's distance is less than *X* from the intersection *CI* and can communicate with each other is defined as the region of intersection, the same below) and the forwarding direction is unchanged, as shown in Fig. 4 (a), and the meaning of each symbol is the same with that in Fig. 3 (a). Therefore, nodes in shaded areas become competitive nodes. In this case, we define (5) to calculate the timer *t*.

$$t = T(1 - (md(p,CI) - md(s,CI))/r) \tag{5}$$

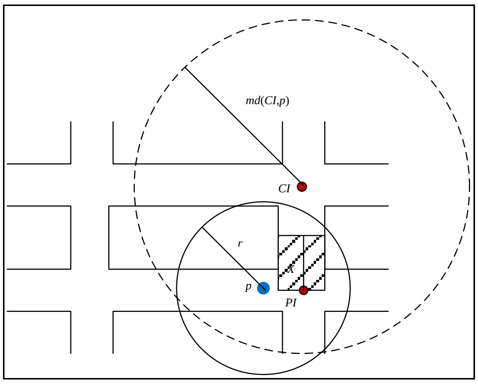
The forwarder *p* is located in the intersection while the forwarding direction is changed, as shown in Fig. 4 (b), and the meaning of each symbol is the same with that in Fig 3 (a). So nodes in shaded areas become competitive nodes. In this case, we define (6) to calculate the timer *t*.

$$t = T(1 - (md(p,CI) - md(s,CI))/(md(p,PI) + X)) \tag{6}$$

where nodes whose distance is *X* from the intersection *PI* can communicate with each other.



(a) forwarding direction is unchanged



(b) forwarding direction is changed

Fig. 4. *p* is located in the intersection

Thus, the second competitive strategy based on location information is shown as in Table I.

TABLE I: SECOND COMPETITIVE STRATEGY BASED ON LOCATION INFORMATION

Strategy 1
1. Search the local cache to see whether the current node has cached a copy of the message. If does, turn to 2. If not, turn to 3;
2. Analyze whether the distance between the current node and the current intersection is less than the distance between the forwarder and the current intersection. If less, turn to 11. If not, clear the cache and turn to 11;
3. Analyze whether the distance between the current node and the current intersection is less than the distance between the forwarder and the current intersection. If less, turn to 4. If not, turn to11;
4. Analyze whether the forwarder is located in the intersection. If located, turn to 5. If not, turn to 8;
5. Analyze whether the current node and the forwarder is located in the same street. If true, turn to 6. If not, turn to 7;
6. Cache the message to the local cache, calculate the timer *t* according to (5) and turn to 12;
7. Cache the message to the local cache, calculate the timer *t* according to (6) and turn to 12;
8. Analyze whether the distance between the forwarder and the current node is less than the distance between the forwarder and the current intersection. If less, turn to 9. If not, turn to10;
9. Cache the message to the local cache, calculate the timer *t* according to (3) and turn to 12;
10. Cache the message to the local cache, calculate the timer *t* according to (4) and turn to 12;
11. Discard the received message, turn to 12;
12. Return.

V. ADAPTIVE TRAFFIC-LIGHT CONTROL ALGORITHM BASED ON GREEN-COMPUTING

In this section, we provide the detailed description about our Adaptive Traffic-light Control algorithm based on Green-Computing (a.b. ATCG).

A. Calculation Policy to the Optimal Sequence

For the sake of clarity, we introduce the following notations:

r: The serial number of roads, r=1, 2

n_r: The number of vehicles on the road r

V_r^q: The *q-th* vehicle on the road r

P_r^q: The time for the vehicle V_r^q to pass through the intersection

$C_r^{r'}$: The time wasted to transfer the green light from the road r to the road r'

q_r: The number of vehicles having passed the road r

W_r^q: The waiting time for the *q-th* vehicle on the road r

A_r^q: The arriving time for the *q-th* vehicle on the road r

The dredging time can be defined as the sum of two kinds of time:

(1) The time for all detected vehicles to pass through the intersection;

(2) The time wasted to transfer the green light between the two roads.

Searching for a shorter dredging time is called to find the lower bound LB_{root}. Therefore, LB_{root} can be expressed as (7)

$$LB_{root} = \sum_{r=1}^{2}\sum_{q=1}^{n_r} P_r^q + C_2^1 \qquad (7)$$

As traffic light has to change at least once from one road to another adjacent road, the process can be simulated as a binary search algorithm, where in one branch the next vehicle located in the active road can pass through the intersection, while in another branch the second road is activated and open the green light to allow vehicles in this road to pass through the intersection. This method is called to search for each node's lower bound LB_{node}. LB_{node} can be expressed as (8):

$$LB_{node} = T_{(q_1,q_2,r)}^{(V_r^q,\cdots)} + \sum_{q=q_1+1}^{n_1} P_1^q + \sum_{q=q_2+1}^{n_2} P_2^q + \lambda \qquad (8)$$

where $T_{(q_1,q_2,r)}^{(V_r^q,\cdots)}$ represents the passing time for the last vehicle behind q_1 vehicles in the road r_1 and q_2 vehicles in the road r_2, and λ is defined as:

$$\lambda = \begin{cases} C_r^{r'},\ if\ n_{r'} - q_{r'} > 0 \\ 0,\ if\ n_{r'} - q_{r'} = 0 \end{cases} \qquad (9)$$

So $T_{(q_1,q_2,r)}^{(V_r^q,\cdots)}$ is calculated as (10):

$$T_{(q_1,q_2,r)}^{(V_r^q,\cdots)} = T_{(q_1',q_2',r')}^{(V_{r'}^{q'},\cdots)} + C_r^{r'} + P_r^q + W_r^q \qquad (10)$$

In (9), if $r'=r$, then $C_r^{r'} = 0$.

The waiting time for the *q-th* vehicle on the road r is calculated as (11):

$$W_r^q = \max\begin{cases} 0, \quad A_r^q - T_{(q_1',q_2',r')}^{(V_{r'}^{q'},\cdots)},\ if\ r' = r \\ 0,\ A_r^q - T_{(q_1',q_2',r')}^{(V_{r'}^{q'},\cdots)} - C_r^{r'},\ if\ r' \neq r \end{cases} \qquad (11)$$

Assumed that there are two roads in one intersection: r_1 and r_2, the first detected vehicle is located in the road r_1, and the time of the green light has been 0 second. Meanwhile, the first vehicle V_1^1 in the road r_1 has reached the stop line and is ready to pass through the intersection. So A_1^1 = 0 second, which means the start time is 0 second. In order to get the optimal sequence, the first vehicle to pass through the intersection should be V_1^1. We use the depth-first search algorithm, as specified in Algorithm 1 and Algorithm 2.

1) Lower bound search algorithm

If the green light is on one road, but still there is a vehicle on another road, so the traffic light has to change at least once to ensure that all vehicles on the two roads can pass through the intersection. After each branch, it compares the LB values of two branches, selects and records the smaller and then continues to the next branch. After several iterations, we can find a recording sequence, and that is optimal sequence. Thus, the LB value of the current node is the best.

However, there may be some nodes cannot complete the search. It would compare the current optimal LB value with that of these nodes. If one of the LB values of these nodes is smaller than the current optimal LB value, the node needs to take participate in the above branch search. In this case, we may probably find a new sequence which is better than the current one, and then its value becomes the new current optimal value. The lower bound search algorithm is shown in Table II.

TABLE II: LOWER BOUND SEARCH ALGORITHM

```
Algorithm 1
time ← LB(root)
Branching(root)      // the root node joins in the tree
for $i$ ← 2 to ($n_1$ + $n_2$ − 1) do      // $n_1$+$n_2$ is the total number of waiting vehicles
  Branching(min(LB(left node), LB(right node)))
  $i$ ← $i$ + 1
  time ← (min(LB(left node), LB(right node))
end for
for $j$ ← 1 to active node do      //the current node
  Branching(list[$j$])      //the $j$-th node joins in the tree
  if LB(node) ≥ time then
     prune this branch      //if LB(node) ≥ time, prune this branch
  else if(reach a leaf node)
     time ← (min(LB(left node),LB(right node))
  if (time ≤ min_green)      //the min time of the green light and the max time of the red light
     time ← min_green
  else if (time ≥ max_red)
      time ← max _red
   end if
  end if
 return (time)
end for
```

TABLE III: OPTIMAL SEQUENCE FINDING ALGORITHM

```
Algorithm 2
Branching(node)
LB(node)
use (7) and (8)      // use (7) and (8)
CHANGESEQUENCE($i$)      //change the $i$-th data of the sequence
change the node sequence at $i$
LB(new node)
if $i$ < $n_1$+$n_2$− 1 then
   list[node] ← max(LB(left node),LB(right node))
   add active nodes to list      // the node joins in the tree
   activenode ← activenode + 1
   else if (max(LB(left node),LB(right node)) ≤time
   list[node] ← max(LB(left node),LB(right node))
   activenode ← activenode + 1
end if
```

2) Optimal sequence finding algorithm

After finding the optimal LB values using the lower bound search algorithm, we need to find the optimal path from the root to the target node.

According to (7) and (8), once the *i-th* data of the sequence is changed, the new vehicular node is added to the tree. After several iterations, we compare the value of left child and that of the right child of the target node. Subsequently the bigger one is stored in the list, and the target node is incremented downwards. Algorithm 3 is shown in Table III.

B. Calculation Policy to the Recommended Speed

Traffic light sends the message to the vehicle. After the vehicle receives the message, it can obtain the following information to calculate the recommended speed:

(1) the distance d between the vehicle and traffic light;

(2) the current traffic light cycle C, the duration of the three phase in the current cycle, respectively denoted as T_g (green light), T_y (yellow light) and T_r (red light), and $C = T_g + T_y + T_r$;

(3) the remaining time of the current three phases, respectively denoted as t_g (green light), t_y (yellow light) and t_r (red light).

Because traffic light will change the duration of the three phases according to the real-time traffic information, so the duration of the three phases is different from each other. Based on the above information, OBU can calculate the recommended speed.

The calculation would use the following symbols:

D: The transmission delay from traffic light to the vehicle

T_a: The time to accelerate from the current speed to the max speed

v: The new speed after OBU recommends

v_{cur}: The current speed

v_{max}: The max speed

v_{min}: The min speed

$t_{cur} = d/v_{cur}$: The time to pass through d with the current speed

$t_{min} = d/v_{max}$: The time to pass through d with the max speed

1) The phase of red light

(1) If the vehicle with the current speed reaches the stop line when the traffic light just turns green, the system would inform the driver to maintain the current speed v_{cur}.

$$\text{If } t_r < t_{cur} < t_r + T_g \text{ , then } v = v_{cur} \tag{12}$$

(2) If the vehicle with the current speed cannot reaches the stop line within the time of $t_r + T_g$, but can pass through the intersection with the maximum speed v_{max}, then the system will inform the driver to accelerate to the maximum speed v_{max} under the premise of traffic permission.

$$\text{If } t_r < t_{min} < t_r + T_g < t_{cur} \text{, then } v = v_{max} \tag{13}$$

(3) If the vehicle with the maximum speed cannot reaches the stop line before the green light turns yellow,

the vehicle will need to wait at least one or more traffic light cycle. In this case, the vehicle must slow down.

$$\text{If } t_{min} > t_r + T_g + C \text{, then } v = \min\left(\max\left((d/t_1), v_{min}\right), v_{max}\right) \tag{14}$$

where $t_1 = t_{min} + T_a - T_g - D$.

2) The phase of green light

(1) If the vehicle with the current speed can pass through the intersection within t_g, then the system would inform the driver to maintain the current speed.

$$\text{If } t_{cur} \le t_g \text{, then } v = v_{cur} \tag{15}$$

(2) If the vehicle with the current speed cannot pass through the intersection within the time of t_g , but can achieve with the maximum speed v_{max}, then the system will inform the driver to accelerate to the maximum speed v_{max} in order not to wait for the next cycle.

$$\text{If } t_{min} \le t_g < t_{cur} \text{, then } v = v_{max} \tag{16}$$

(3) If the vehicle with the maximum speed cannot pass through the intersection within the time of t_g, it must wait for the next traffic light cycle or even for several cycles. In this case, the vehicle must slow down to wait for at least one red phase plus a yellow one.

$$\text{If } t_{min} > t_g \text{, then } v = \min\left(\max\left((d/t_2), v_{min}\right), v_{max}\right) \tag{17}$$

where $t_2 = t_{min} + T_r + T_y + T_a - C - D$.

3) The phase of yellow light

Compared with the situation of red light, the vehicle needs to wait one extra time of the yellow light. The speed calculation is substantially the same as that of the red light phase:

$$v = v_{cur} \text{, if } t_y + T_r < t_{cur} \le t_y + T_r + T_g \tag{18}$$

$$v = v_{max} \text{, if } t_y + T_r + T_g < t_{min} \le t_y + C \tag{19}$$

$$v = \min\left(\max\left((d/t_3), v_{min}\right), v_{max}\right) \text{, if } t_y + C \le t_{min} \tag{20}$$

where $t_3 = t_{min} + T_a - T_g - D$.

Among them, (12), (15) and (18) suggest the driver to maintain the current speed. In order to avoid parking, (13), (16) and (19) suggest the driver to accelerate to the maximum speed in the current traffic light cycle. For the vehicle cannot pass through the intersection within the current cycle, (14), (17) and (20) suggest the driver to reduce or maintain the speed to wait for the next cycle.

C. Control Algorithm Description

Based on the Fig. 1, the entire process of control is described as follows:

(1) OBU submits the current speed and the distance away from the intersection to traffic light when the vehicle would arrive at the intersection.

(2) Traffic light receives the vehicle information by the antenna and feedback the current phase and the remaining time.

3) Traffic light calculates the optimal sequence by received traffic information and change the time of the next cycle.

4) The vehicle receives traffic light's feedback, calculates the recommended speed, and decides whether to pass through the intersection. According to the recommended speed, the driver keeps the current speed or accelerates to pass through the intersection safely.

VI. SIMULATION EVALUATION

A. Routing Algorithm

In this section, we conduct simulation experiments to evaluate the performance of VDNC algorithm compared with First-Contact(FC) and Spray-and-Wait (SW) protocols by the ONE[28]. The ONE is an Opportunistic Network Environment simulator which provides a powerful tool for generating mobility traces, running DTN messaging transmission with different routing protocols, and visualizing both interactive simulations process in real-time and results after completion

1) Simulation parameters

The key parameters of simulation scenario are summarized in Table IV.

TABLE IV: SIMULATION PARAMETERS

parameter	value
network size	5000m*4000m
simulation time	12h
mobility model	Shortest Path Map Based Movement
interval of nodes' speed	(0, 20) m/s
interval of message generation	(30, 40)s
message size	10 KB
initial number of copies (only for SW)	8

2) Simulation results

In this paper, we consider the following metrics to evaluate the performance of VDNC: the delivery radio and the transmission delay. In order to ensure the accuracy of experiments, the data results are the average of multiple simulation experiments.

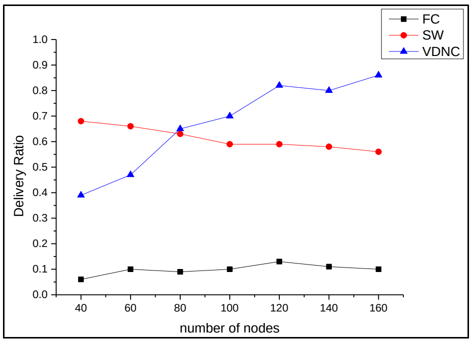


Fig. 5. Delivery radio versus number of nodes

a) Influence of the number of nodes

Fig. 5 and Fig. 6 show the influence of the number of nodes on the delivery radio and the transmission delay. When the number of nodes increases, the delivery radio of VDNC algorithm is significantly higher than the others and the transmission delay is lower. However, when nodes are few, its delivery radio is low, and the transmission delay is relatively high. This is because when the number of nodes is sparse the forwarder cannot find a suitable next hop node, but only to carry the message itself. However, when the number of nodes is very large, the traffic congestion is caused, and the transmission delay increases.

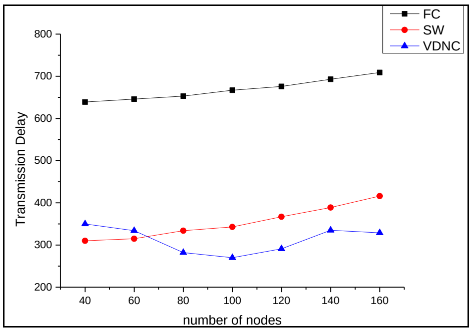


Fig. 6. Transmission delay versus number of nodes

b) *Influence of the range of transmission*

Fig. 7 and Fig. 8 show the influence of the range of transmission on the delivery radio and the transmission delay. When the range of transmission increases, the delivery radio of all three algorithms increases and the transmission delay reduces. We can find the performance of VDNC algorithm is superior to the other two. This is because when the forwarder selects the next hop in VDNC algorithm, the location of the node is considered, and only the node closer to the destination possibly competes to become the next hop. And when the range of transmission increases, it is possible to select the more suitable node to become the next node so that the delivery radio increases and the transmission delay reduces.

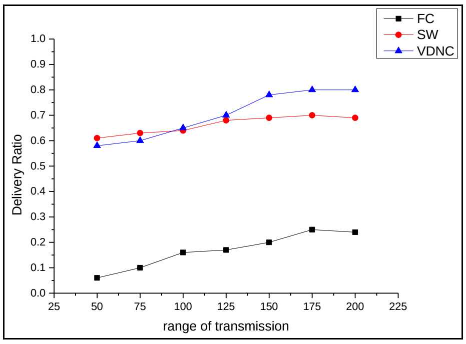


Fig. 7. Delivery radio versus range of transmission

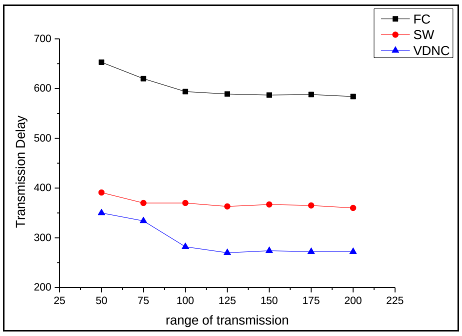


Fig. 8. Transmission delay versus range of transmission

c) *Influence of TTL*

Fig. 9 and Fig. 10 show the influence of TTL on the delivery radio and the transmission delay. When TTL increases, the delivery radio of all three algorithms increases and the transmission delay increases slightly. We can find the performance of VDNC algorithm is better than the other two. This is because when TTL increases, the survival time of one message becomes longer, and the forwarder has greater probability to encounter the destination. So the delivery radio will increase, but the transmission delay will also increase accordingly.

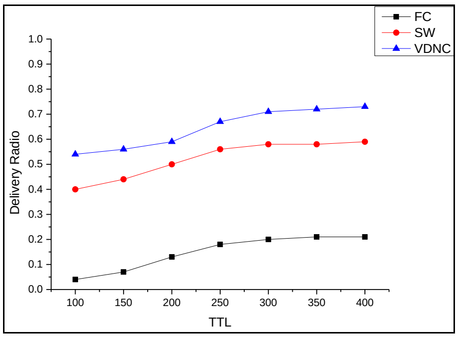


Fig. 9. Delivery radio versus TTL

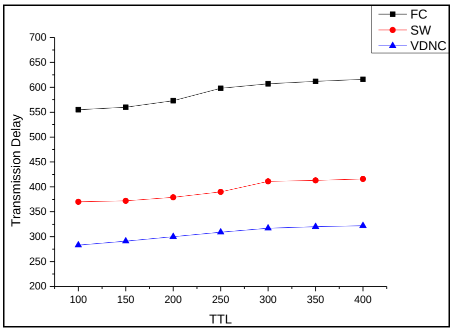


Fig. 10. Transmission delay versus TTL

Traffic-Light Control Algorithm

1) *Simulation parameters*

For simplicity, the following assumptions are made:

(1) If the vehicle has to stop, we assume that the vehicle stops only once time within a traffic light cycle.

(2) The vehicle can select one of the two ways to arrive at the destination: a) from r_1; b) from r_2.

(3) The time to change the speed from v_{cur} to v is 3 seconds ($T_a = 3$).

(4) Arrival of the vehicle obeys Poisson Distribution.

(5) Set the maximum time of the red light max_red is 90 seconds, the minimum time of the green light min_green is 15 seconds, and according to the real-time traffic flow, the traffic light cycle ranges from 30 to 180 seconds. Meanwhile, v_{max} is 60 km / h, and v_{min} is 10 km / h.

We compare our algorithm with the adaptive fuzzy traffic light control algorithm. For comparison convenience, we assume that the maximum number of detected vehicles is 30. We set the following three variables as inputs:

(1) Arrival: the number of vehicles to arrive at the intersection during the green phase, with the possible states being {Zero, Small, Medium, Many}.

(2) Queue: the number of vehicles to arrive at the intersection during the red phase, with the possible states being {Zero, Small, Medium, Many}.

(3) Volume: the total number of vehicles to arrive at the intersection in all directions, with the possible state being {Small, Medium, Large}.

We set the variable "Extension" as output (extension time of the green phase), with the possible state being {Zero, Short, Medium, Long}.

Control rules are divided into three groups, including small, medium and large, where the number of rules in small group is 7, that in the medium group is 9, and that in the large group is 7. We can cite some rules but not all:

When Volume = Small, if Arrival = Small and Queue = Zero, then Extension = Short.

When Volume = Large, if Arrival = Many and Queue = Many, then Extension = Long.

When Volume = Medium, if Arrival = Medium and Queue = Medium, then Extension = Medium.

The remaining rules are arranged with a permutation and combination of the above conditions, but not involved unnecessary repeats.

All these experiments are implemented with the platform of Windows 7 and MATLAB R2009a. We run 1,000 times of traffic light cycle for each traffic data and assume that the traffic flow of two roads is equal.

2) Simulation results

As it can be seen from Fig. 11 and Fig. 12, compared with Fuzzy algorithm, the average travelling time and the average waiting time is shorter in ATCG algorithm especially from about 7:00 am to 20:00 pm. It is because Fuzzy algorithm depends on the extension of green time to achieve lower waiting time. ATCG algorithm is not only to minimize the waiting time, but also to give the driver the recommended speed to help the driver not to stop or to decrease stoppages when passing through the intersection. In addition, the adjacent intersections will share information of traffic conditions for each other. The average CO_2 emissions can be calculated from Fig. 13. During peak hours, compared with Fuzzy algorithm, the average CO_2 emissions can be reduced by 60% on r_1 and 50% on r_2 when using ATCG algorithm.

In Fuzzy algorithm, the vehicle cannot get any information about the current traffic light phase. Therefore, when the vehicle approaches the intersection, it must wait for the change of traffic light. In contrast, in ATCG algorithm, the control center can obtain real-time traffic information from the OBU by sensor nodes installed on the vehicle, and then change the traffic light cycle based on real-time traffic conditions. Furthermore, according to the received information before the vehicle arrive traffic light, OBU device can provide a recommended speed to avoid unnecessary parking for the vehicle. Thus, the vehicle can pass through the intersection with fewer stoppages and shorter waiting time, so that CO_2 emissions can be achieved least. Thus, compared to Fuzzy algorithm, ATCG algorithm has more advantages on efficiency and environmental protection.

(1) Average Waiting Time from the source to the destination, shown in Fig. 11.

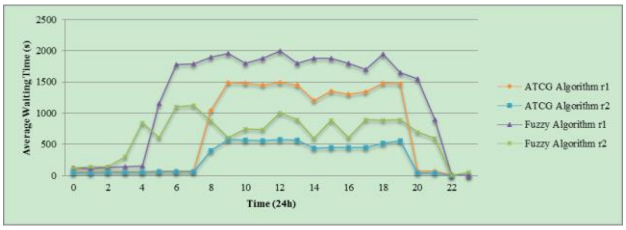


Fig. 11. Average waiting time

(2) Average Travelling Time from the source to the destination, shown in Fig. 12.

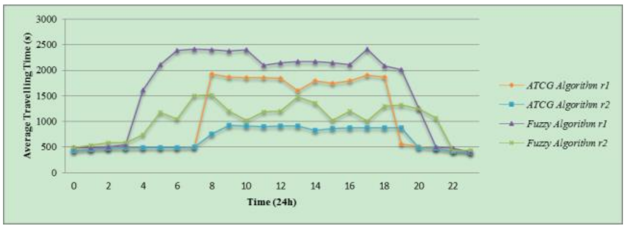


Fig. 12. Average travelling time

(3) Average CO_2 Emissions from the source to the destination, shown in Fig. 13.

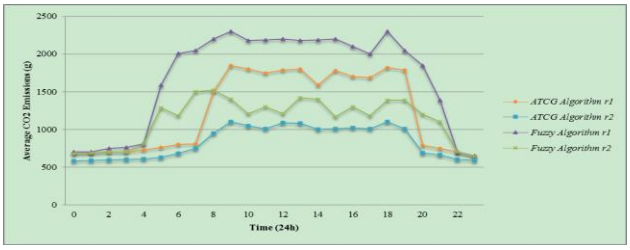


Fig. 13. Average CO_2 emissions

VII. CONCLUSIONS

This paper presents a Vehicular Delay-tolerant Network routing algorithm based on Contention (VDNC). VDNC algorithm consists of three strategies: the intersection selection strategy based on the Manhattan distance and traffic information, the first competitive strategy based on the direction of movement and the second competitive strategy based on the information of position. Simulation experiments measure the delivery radio and the transmission delay. Compared to SW algorithm and FC algorithm, simulation results show that VDNC algorithm enables the node to deliver messages with a higher transmission success rate and a shorter delay.

This paper also proposes an Adaptive Traffic-light Control algorithm based on Green-Computing (ATCG), including the lower bound search algorithm, the calculation policy for the optimal sequence and the recommended speed. And the core idea is to reduce CO_2 emissions by the guidelines of acceleration and deceleration for the driver. The average waiting time, the average travelling time and the average CO_2 emissions are considered in the simulations. Compared to Fuzzy algorithm, experiment results show that ATCG algorithm makes the vehicle pass through the intersection with fewer stoppages and shorter waiting time, and consequently CO_2 emissions can be achieved the least. However, this paper only studies a single intelligent traffic light intersection, without the consideration of coordinated control algorithm in multi-intersections, which is the focus of future work.

ACKNOWLEDGEMENT

The authors would like to thank the anonymous reviewers of this paper for his/her objective comments and helpful suggestions while at the same time helping us to improve the English spelling and grammar throughout the manuscript.

REFERENCES

[1] A. Vahdat and D. Becker, “Epidemic routing for partially connected ad hoc networks,” Master thesis, 2000.

[2] A. Lindgren, A. Doria, and O. Schelén, “Probabilistic routing in intermittently connected networks,” *ACM SIGMOBILE Mobile Computing and Communications Review*, vol. 3, pp. 19-20, Jul. 2003.

[3] S. Jain, K. Fall, and R. Patra, “Routing in a delay tolerant network,” in *Proc. ACM SIGCOMM*, 2004, pp. 145−158.

[4] T. Spyropoulos, K. Psounis, and C. S. Raghavendra, “Spray and wait: An efficient routing scheme for intermittently connected mobile networks,” in *Proc. ACM SIGCOMM Workshop on Delay-Tolerant Networking*, 2005, pp. 252-259.

[5] T. Spyropoulos, K. Psounis, and C. S. Raghavendra, “Spray and focus: Efficient mobility-assisted routing for heterogeneous and correlated mobility,” in *Proc. Fifth Annual IEEE International Conference on Pervasive Computing and Communications Workshops*, White Plains, New York, USA, 2007, pp. 79-85.

[6] H. Zhu, S. Chang, and M. Li, “Exploiting temporal dependency for opportunistic forwarding in urban vehicular networks,” in *Proc. Infocom, IEEE*, 2011, pp. 2192-2200.

[7] H. Zhu, S. Chang, and Y. Zhu, “Scaling the mobility for fast opportunistic forwarding in vehicular networks,” in *Proc. INFOCOM, 2013 Proceedings* 2013, pp. 2832-2840.

[8] F. Holger, K. Michael, and K. Michael, “CBF: Contention-based forwarding for mobile ad-hoc networks,” *Elsevier Science*, vol. 4, pp. 351-369, Jan. 2003.

[9] P. Tournoux, V. Conan, and J. Crowcroft, “Wardrop equilibrium formulation of resource-constrained dtn routing in public safety networks,” in *Proc. IEEE 8th International Conference on Mobile Adhoc and Sensor Systems*, 2011, pp. 968-974.

[10] F. Xu, S. Guo, and J. Jeong, “Utilizing shared vehicle trajectories for data forwarding in vehicular networks,” in *Proc. 30th IEEE International Conference on Computer Communications*, Kyoto, Japan, 2011, pp. 441-445.

[11] Y. Zeng, K. Xiang, and D. Li, “Directional routing and scheduling for green vehicular delay tolerant networks,” *Wireless Networks*, vol. 2, pp. 161-173, Sep. 2013.

[12] Y. T. Wei and J. W. Wang, “A delay/disruption tolerant routing algoritnm based on traffic prediction,” in *Proc. 27th Chinese Control and Decision Conference*, May. 2015, pp. 3253-3258.

[13] S. Tayal and M. R. Tripathy, “Efficient packet forwarding approach using public vehicles,” in *Proc. Fifth International Conference on Advanced Computing & Communication Technologies*, Feb. 2015, pp. 475-480.

[14] M. K. Nasir, S. A. Ali Shah, M. A. Qureshi, and M. Oche, “Adapting geographical DTN routing for enhanced connectivity in partitioned VANETs on highways,” in *Proc. Region 10 Symposium*, Apr 2014, pp. 105-110.

[15] V. Mahendran, R. Gunasekaran, and R. M. C. Siva, “Performance modeling of delay-tolerant network routing via queueing petri nets,” *IEEE Trans on Mobile Computing*, vol. 13, no. 8, pp. 1816-1828, Aug. 2014.

[16] C. P. Pappis and E. H. Mamdani, “A fuzzy logic controller for a traffic junction,” *IEEE Transactions on Systems, Man and Cybernetics*, vol. 7, no. 10, pp. 707-717, Jul. 1977.

[17] P. G. Balaji, X. German, and D. Srinivasan, “Urban traffic signal control using reinforcement learning agents,” *IET Intelligent Transport Systems*, vol. 4, no. 3, pp. 177-188, Apr. 2010.

[18] B. Zhou, J. Cao, and X. Zeng, “Adaptive traffic light control in wireless sensor network-based intelligent transportation system,” in *Proc. 72nd Vehicular Technology Conference Fall*, 2010, pp. 1-5.

[19] M. Abdoos, N. Mozayani, and A. L. C. Bazzan, “Traffic light control in non-stationary environments based on multi agent Q-learning,” in *Proc. 14th International IEEE Conference on Intelligent Transportation Systems*, 2011, pp. 1580-1585.

[20] M. Abdoos, N. Mozayani, and A. L. C. Bazzan, “Holonic multi-agent system for traffic signals control,” *Engineering Applications of Artificial Intelligence*, vol. 5, pp. 1575-1587, Dec. 2013.

[21] F. Y. Leu, M. H. Chen, and Y. L. Huang, “Controlling traffic lights for ambulances,” in *Proc. Seventh International Conference on Broadband*, 2012, pp. 462-467.

[22] S. El-Tantawy and B. Abdulhai, “Multi-agent reinforcement learning for integrated network of adaptive traffic signal controllers (MARLIN-ATSC),” in *Proc. 15th International IEEE Conference on Intelligent Transportation Systems*, 2012, pp. 319-326.

[23] Y. C. Chiou and Y. F. Huang, “Stepwise genetic fuzzy logic signal control under mixed traffic conditions,” *Journal of Advanced Transportation*, vol. 47, no. 1, pp. 43-60, Jan. 2013.

[24] G. Mahler and A Vahidi, “Reducing idling at red lights based on probabilistic prediction of traffic signal timings,” in *Proc. American Control Conference*, 2012, pp. 6557-6562.

[25] S. Araghi, A. Khosravi, and D. Creighton, “A review on computational intelligence methods for controlling traffic signal timing,” *Expert Systems with Applications*, vol. 42, no. 3, pp. 1538-1550, Mar. 2015.

[26] M. Alsabaan, K. Naik, and T. Khalifa, “Applying vehicular networks for reduced vehicle fuel consumption and CO_2 emissions,” in *Book Chapter in Intelligent Transportation Systems*, Mar. 2012, pp. 7-16.

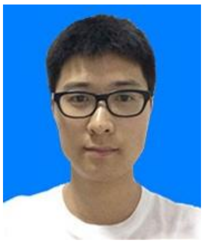
[27] T. Oda, T. Otokita, and S. Niikura, “Online traffic signal control for reducing vehicle carbon dioxide emissions,” *IEEE Transactions on Industry Applications*, vol. 126, no. 11, pp. 1522-1530, Nov. 2006.

[28] A. Keränen, J. Ott, and T. Kärkkäinen, “The ONE simulator for DTN protocol evaluation,” in *Proc. 2nd International Conference on Simulation Tools and Techniques. ICST (Institute for Computer Sciences, Social-Informatics and Telecommunications Engineering)*, 2009, p. 55.

Hai-Ping Huang was born in Fujian Province, China, in 1981. He received the B.S. degree and the M.S. degree from Nanjing University of Posts and Telecommunications of China (NUPT), Nanjing, respectively in 2002 and 2005, both in Computer Science and Technology. He received the Ph.D. degree from Soochow University of China, Suzhou, in 2009, in Computer Application Technology. His research interests include wireless sensor networks, coverage control, and information security.

Dan Sha was born in Jiangsu Province, China, in 1990. She received her B.S degrees from Nanjing University of Posts and Telecommunications in 2013. She is now a postgraduate in Software Engineering at Nanjing University of Posts and Telecommunications, China. Her current research interest is routing algorithms in Delay Tolerant Networks.

Yong-Can Zhang was born in Jiangsu Province, China, in 1990. He received the B.S. degree from Changzhou Institute of Technology of China, Changzhou in 2012, in Computer Science and Technology. He is currently pursuing the M.S. degree with the College of Computer, Nanjing University of Posts and Telecommunications of China. His research interests include wireless sensor networks and coverage control.



Pengfei Li was born in Jiangsu Province, China, in 1989. He received his B.S degrees from Nanjing University of Posts and Telecommunications in 2012. He is now a postgraduate in Software Engineering at Nanjing University of Posts and Telecommunications, China. His current research interest is routing algorithms in Delay Tolerant Networks.