Density-Based Dynamic Revision Path Planning in Urban Area via VANET

Siwei Wu, Demin Li, Guanglin Zhang(✉), Chang Guo, and Leilei Qi

College of Information Science and Technology,
Engineering Research Center of Digitized Textile and Fashion Technology,
Ministry of Education, Donghua University, Shanghai, China
{wusiwei,guochang}@mail.dhu.edu.cn, {deminli,glzhang}@dhu.edu.cn

Abstract. Dynamic path planning can efficiently enhance the validity and reliability of real-time navigation for vehicles to avoid unexpected traffic congestions and to shorten the whole traveling time. In this paper, we present a dynamic path planning method in which the density of road links is set as the main measurement for traffic status. By analyzing the traveling time and waiting time, we propose a mechanism for vehicles to decide whether to revise the current path. Our method aims at offering a path with smallest time cost for vehicles towards the destination. Furthermore, we give the discussion on time complexity and convergence of our method. Finally, simulation is conducted to evaluate the proposed method.

Keywords: Path planning · Dynamic · Density-based · VANET

1 Introduction

Nowadays, path planning for vehicles plays an essential part in Intelligent Traffic System (ITS). The objective is to provide the drivers with a path leading from source to destination with shortest time or distance. The conventional navigation technology is static, which only takes the length of roads or the historical data of traffic mobility into consideration. This kind of navigational service is launched before the travel starts, so it does not make quick response to traffic congestion and select an alternative route for vehicles. So it is crucial to work out an effective dynamic path planning method for urban transportation scenario.

There are many works on dynamic path planning in traffic scenarios. A partial path planning algorithm is proposed in [1]. Several fixed intermediate destinations were selected between the source and the final destination in order to reduce the re-calculation complexity. The segment route was monitored for vehicles' arrival speed and when necessary, a better route was re-calculated. He et al. presented a skyline for candidate path selection method in paper [3]. It mainly focused on the prediction of traffic condition by a density-speed traffic

© ICST Institute for Computer Sciences, Social Informatics and Telecommunications Engineering 2017
X.-L. Huang (Ed.): MLICOM 2016, LNICST 183, pp. 129–138, 2017.
DOI: 10.1007/978-3-319-52730-7_13

flow model. If the estimated speed for a road link was too low, this road link would be discarded for path planning. Other papers such as [2,4] concerned path planning from a global perspective. They studied how to keep balance of the entire road network to avoid congestion.

All the above works are based on a consumption that the traffic congestion lasts for a rather long time and thus the road segments suffering congestion will not be considered for path planning anymore. In addition, the situation that the alternative route to bypass the congestion may suffer another congestion is not taken into consideration. From a realistic point of view, it makes sense to predict the duration of congestion and the time to bypass the congestion. If the former exceeds the latter, then the vehicle will take the alternative roads, and if the former is less than the latter, the vehicle will just wait in the queue for the smoothness of traffic.

In this paper, we use the real-time traffic information obtained by Vehicular Ad hoc Network (VANET) for dynamic navigation path planning service. We propose a recursive algorithm to continuously revise the route for a vehicle on the purpose of offering a dynamic available path which will take the shortest time to traverse. Moreover, we devise a traveling time and waiting time comparison mechanism, which consider both the congestion duration and alternative candidate path time consumption. This mechanism is in accordance with reality and will improve the reliability for path planning.

The remainder of this paper is organized as follows. System model is presented in Sect. 2. Then the analysis of the proposed problem is illustrated in Sect. 3. The simulation is given in Sect. 4. Finally, Sect. 5 concludes this paper.

2 System Model

2.1 VANET in Urban Area Scenario

VANET in urban area scenario is shown in Fig. 1. Each vehicle is assumed to be equipped with Onboard Unit (ONU). A VANET is a self-constructed network without any host server. All vehicles within one-hop region constitute an independent group. If one vehicle is going to launch communication with one from another group, it should call for a specific routing protocol to transfer data packet in different groups.

To further develop the model of path planning, we first devise a symbolized VANET architecture as (I, R). The set of intersections in the urban road network is denoted as a collection I. Let R be the set of all road links. A road link k is denoted in Fig. 1 as $r_k(t) = \{i_k(x, y), j_k(x, y), v_k(t), \mathbf{d}, w_k(t)\}$, where i and j is the start and end point of the road link. $v_k(t)$ is the traffic flow speed of the road link k. $\mathbf{d}$ is the direction, confining to $\mathbf{d} \in \{i \to j, j \to i\}$. The weight is $w_k(t)$, which is related to the variant density of the road.

The symbols and notations used in this paper is shown in Table 1.

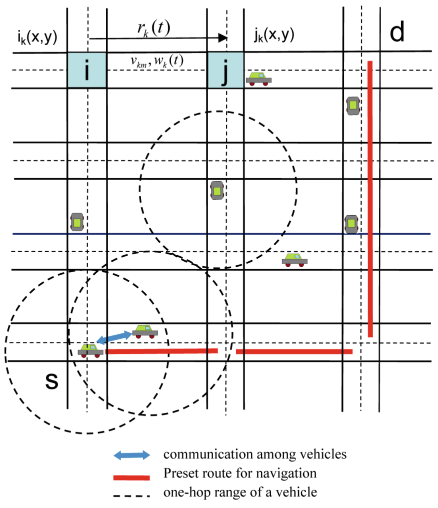

$i_k(x,y)$
$r_k(t)$
$j_k(x,y)$
d
i
$v_{km}, w_k(t)$
j
s
communication among vehicles
Preset route for navigation
one-hop range of a vehicle

Fig. 1. VANET in urban area scenario

Table 1. Symbols and notations

Symbols	Definitions
$r_k(t)$	A road link k
$\|r_k(t)\|$	Length of $r_k(t)$
$w_k(t)$	Weight of road link k
$P(t)$	A path in road network
$den_k(t)$	Road density
$N(t)$	The list each vehicle keeps locally
$\|N(t)\|$	The total number of vehicles
$v_k(t)$	The average spatial velocity of traffic flow on $r_k(t)$
t_{ESTI}	The time candidate path takes to bypass congestion
t_p	The time of congestion to resume smoothless
$p_{cong}(t)$	Set of congested segments

2.2 Dynamic Path Planning Model

In this part, we mainly focus on a dynamic path planning model. Graph Theory is used here to illustrate the model. As is mentioned, I is the set of road network intersections. R is the set of road links. Let $W(t)$ be the set of road link weight, where $W(t) = \{w_k(t)|k \in R\}$. It is related to its real-time traffic density. A navigation path $P(t)$ can be denoted as $P(t) = (r_1(t), r_2(t), \cdots, r_n(t))$ [3], where $r_i \in R$ and $r_n(t) = \{i_n(x,y), j_n(x,y), v_n(t), \mathbf{d}, w_n(t)\}$. Our final objective is to work out a path $P(t)$ along which vehicles can spend the shortest time to the destination.

Referred to the expression in [3], our problem can be formulated as:

Input:

(1) a road network $(I, R, W(t))$
(2) the start point s and the destination point d, where $s, d \in I, s \neq d$

Objective: Find a path $P(t) = (r_1(t), r_2(t), \cdots, r_n(t))$ leading from s to d, which minimize

$$Time = \sum_{i}^{r_i(t) \in P(t)} \frac{|r_i(t)|}{v_i(t)} + \sum_{n} t_n, \tag{1}$$

where $t_n \in Q$, Q is the set of waiting time and $v_i(t)$ is the speed of traffic flow and $|r_i(t)|$ is the length of road link $r_i(t)$.

Subject to: $\sum_{i}^{r_i(t) \in P(t)} |r_i(t)| \leq \Theta$, where Θ is a length constraint.

Our aim is to find a path, which takes the least time to the destination when considering the real time traffic condition. We first find a preset route which consumes minimal time in static situation, which can be obtained by classical algorithms like Dijkstra or A*. In a free traffic situation, a vehicle following the preset route can spend least time to the destination, but if parts of the preset route is congested, our proposed iterative algorithm is used to bypass the congestion.

3 Dynamic Path Planning Solution

3.1 Traffic Information Collection

Since dynamic path planning is based on real-time road condition, it is essential to collect ambient traffic information regularly and efficiently. In vehicular ad-hoc network, this task is accomplished by the vehicle itself. It is assumed that each car maintains a list, in which every entry denotes the parameters of all the other car nodes on the same road link within one-hop region and updates every Δt time. The entry records the vehicle's id, current velocity, road link id it belongs to and time stamp. The standard format is shown in Table 2. As is mentioned above, the list each vehicle keeps can be defined as a set $N(t)$. We define $|N(t)|$ as the total number of vehicles running on the current road link.

Table 2. Beacon message format

Vehicle	Velocity	Road link	Time stamp
ID	$v(t)$	r(i,j)	t

With all these vehicles' information stored, vehicles can get some crucial statistic results such as the mean velocity of vehicles on the road link and its current density. According to the Traffic Flow Theory [5], the space mean speed of traffic flow on road link $r_k(t)$ is denoted as

$$v_k(t) = \frac{|N(t)|}{\sum_{v(t) \in N(t)} \frac{1}{v(t)}}. \tag{2}$$

And density of road link $r_k(t)$ is

$$den_k(t) = \frac{|N(t)|}{|r_k(t)|}. \tag{3}$$

3.2 The Criteria for Revising the Preset Route

When a vehicle launches a request for path planning and navigational service, we assume that with the help of ONU and the static parameters stored, it will first work out an initial path $P_{init} = (r_1, r_2, \cdots, r_n)$ as a preset route leading from the source position to its destination. When the initial route $P_{init} = (r_1, r_2, \cdots, r_n)$ is obtained, it becomes the base to be adjusted and revised with the real-time traffic information. It is assumed that each source node automatically creates a dynamically-updating table, in which there records road links included in P_{init} and the corresponding initial weight and subsequent updated value. The weight of the road links are updated every Δt time. Since road link weight $w(t)$ is a function of the density, it fluctuates all the time. It is impractical to recalculate the route as soon as the weight changes, for in reality a slight fluctuation in road link density may not exert an obvious impact on the real-time traffic condition and thus there is no need to change the current route. So we set a threshold $\alpha (> 0)$ to denote the weight fluctuation. Unless the real-time road link weight fluctuates beyond the threshold, the vehicle keeps moving forward along the preset route. When

$$\frac{w_n(t + \Delta t) - w_n(t)}{w_n(t)} = \frac{w_n(t + \Delta t)}{w_n(t)} - 1 > \alpha, \tag{4}$$

where

$$w_n(t) = \frac{A}{den_k(t)}, \tag{5}$$

A is a non-negative constant, it is necessary to revise the preset route and select a candidate path. By substituting inequality (4) with equation (5), we get

$$\frac{den_k(t)}{den_k(t + \Delta t)} - 1 > \alpha. \tag{6}$$

Since weight $w_n(t)$ is a function of the inverse of density $den_k(t)$, when density increases, its weight should be set lower due to the high possibility of congestion. $w_n(t + \Delta t)$ lower than $w_n(t)$ shows the increased density of the road. When $\frac{w_n(t+\Delta t)}{w_n(t)} - 1 < \alpha$, it means the density of road links fluctuates within a rational range and thus there is no need to launch a path planning service.

3.3 Candidate Path Selection

Once a set of consecutive road links are congested, i.e. the density increases and the corresponding weight declines beyond the threshold α, the candidate path selection algorithm is launched. A general example of the proposed algorithm is explained in Fig. 2(a). When there is a congested segment AB on the preset route $s \rightarrow d$, we just work out a candidate path to bypass the congestion.

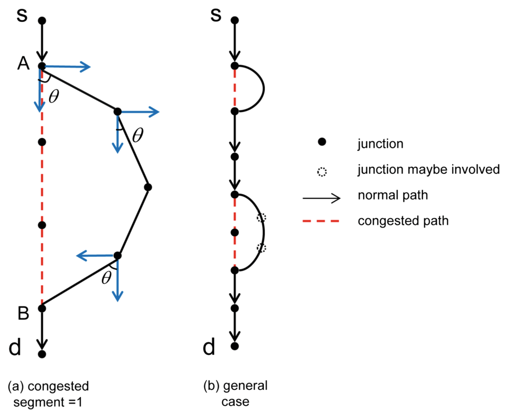


Fig. 2. Example for proposed algorithm

The current route is denoted as $P(t) = \{r_1(t), r_2(t), \cdots, r_n(t)\}$ and the congested road segments is $p_{cong}(t) = \{r_k(t), r_{k+1}(t), \cdots, r_m(t)\}$, which is a subset of $P(t)$. We select the subset $p_{cong}(t)$ as the next miniature of path planning, i.e. $r_k(t)$ is selected as a sub-source and $r_m(t)$ is selected as the sub-destination. A preset route leading from $r_k(t)$ to $r_m(t)$ is first worked out by static methods. The vehicle goes along the preset route and bypass congestion. It is an iterative process and manifests the dynamic feature.

Let $P_{alter}(t) = (r'_k(t), r'_{k+1}(t), \cdots, r'_m(t))$ be the candidate path to bypass the congested roads $p_{cong}(t)$. Then we analyze the time to take the candidate path t_{ESTI} and the time for congestion to consume free t_p.

$$t_{ESTI} = \frac{|r'_k(t)|}{v_{k'}(t)} + \frac{|r'_{k+1}(t)|}{v_{k+1'}(t)} + \cdots + \frac{|r'_m(t)|}{v_{m'}(t)} \tag{7}$$

$$t_p = \frac{|r_k(t)|}{v_k(t)} + \frac{|r_{k+1}(t)|}{v_{k+1}(t)} + \cdots + \frac{|r_m(t)|}{v_m(t)} \tag{8}$$

If $t_{ESTI} < t_p$, it demonstrates that taking the candidate path will help cut the total traveling time towards destination and so the vehicle should take it to bypass the congestion. Otherwise, if $t_{ESTI} > t_p$, the vehicle should wait in the queue for the congestion to resume free. The algorithm is described in pseudocode Algorithms 1 and 2.

Algorithm 1. Dynamic Path Planning

```
Initialize preset route
$P_{init} = \emptyset$
**for** $r_n(t_0) \in R$ **do**
    $T_{init}(t_0) = \frac{|r_n(t_0)|}{v_n(t_0)}$
    $P_{init} \leftarrow P_{init} \bigcup \{argmin \sum_i \frac{|r_i(t_0)|}{v_i(t_0)}\}$
    apply Dijkstra to work out $P_{init}$
**end for**
**procedure** DynamicRevising
    $p_{cong}(t) = \emptyset$
    **for** $i = 0, i <= n, i++$ **do**

        **if** $\frac{w_n(t+\Delta t)}{w_n(t)} - 1 > \alpha$ and $w_n(t + \Delta t) < w_n(t)$ **then**
            $p_{cong}(t) \leftarrow p_{cong}(t) \bigcup r_i(t)$
        **else**continue
        **end if**
    **end for**
    launch $FindAlternativePath()$
    compare $t_{ESTI}$ and $t_p$
    **if** **then**$t_{ESTI} > t_p$
        stick to the current path and wait for congestion relieved
    **end if**
**end procedure**
```

In Algorithm 1, a preset route is first worked out by static method. Then the vehicle collects the statistics of road links in the preset route to detect congestion. Once the road weight drops beyond the threshold, Algorithm 2 is launched to find an alternative path. By analyzing the time, whether to take the candidate path or wait in the queue is decided.

Algorithm 2. Find Alternative Path

```
$source \leftarrow r_k(t), destination \leftarrow r_m(t)$
$P_{init} \leftarrow P_{init} \bigcup \{arg \frac{Q_a(t).length}{|r_a|}\}$
DynamicPathPlanning()
**return** $P_{alter}(t)$
```

Algorithm 2 is a subfunction of Algorithm 1. Its function is to find an Alternative path. It also calls Algorithm 2 to form an iterative method.

3.4 Algorithm Analysis

The analysis of time complexity of the algorithm is given as the following theorem.

Theorem 1. The time complexity is in proportion to the number of congested road segments which need revising.

Proof. First we assume that the preset route is free and sound enough to go through without any other effort to revise. In this case the complexity is $O(n)$, where n is the number of road links contained in preset route. Then if there is only one congested segment containing k consecutive road links, the complexity is $O(n)+O(k)$. And further if the there is m congested segments, the complexity is $O(n)+mO(k)$. The deduction shows that the complexity of this algorithm is related to the number of iteration, which is actually the congested road segments to bypass. So the whole time complexity of the algorithm can be denoted as

$$KO(n), \tag{9}$$

where K is the number of iteration and can be regarded as a constant.

Theorem 2. The revision process of a route is convergent as long as the intersection angle between the search direction and the direction pointing from starter to destination or sub-destination is less than 90°.

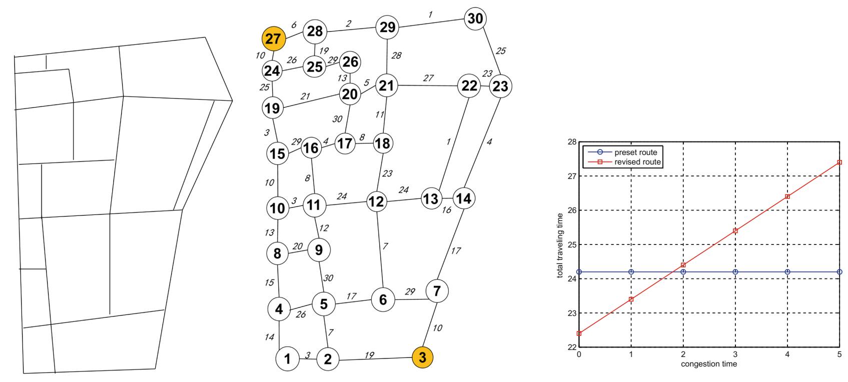
Proof. Figure 2(b) shows the search direction should be always towards the destination. We define the intersection angle between searching direction and the direction pointing from starter to destination as θ. θ should be less than 90° because based on common sense, drivers never retrace their steps. For further proof, we introduce the theory of gradient descent here to prove the above theorem. Based on the gradient descent, we have the iterative formula $a_{k+1} = a_k + \rho_k s^{-(k)}$, where $s^{-(k)}$ represents the negative gradient direction and ρ_k is the step. In our realistic urban traffic situation, the negative gradient direction $s^{-(k)}$ can be specified as the direction pointing from the starter to the destination or the sub-destination and the step ρ_k end coordinate of one road link for the candidate path search. In addition, a_k stands for the terminal coordinate of road link k and a_{k+1} stands for the successor's terminal coordinate. So the objective function is

$$|a_{k+1} - x_d|, \tag{10}$$

where x_d is the sub-destination's coordinate. During the exploring process, we minimize the objective function, that is, the candidate path is found towards the destination and finally the whole process will be convergent for the destination is locally unique.

4 Numerical Simulation

In this section, we evaluate the performance of the path planning algorithm proposed in Sect. 3. First we construct a virtual urban area road network shown in Fig. 3(a). Fig. 3(b) is the corresponding graph, in which the road length is randomly created by Matlab ranging from 0 to 30.



(a) Virtual Road Network (b) Abstract Graph (c) Total Traveling Time

Fig. 3. Performance evaluation

We consider the total traveling time from source 3 to destination 27 with the increasing congestion time (recovery time). From the urban area road graph shown in Fig. 3(b), the preset route from node 3 to node 27 is $3 \to 2 \to 1 \to 4 \to 8 \to 10 \to 15 \to 19 \to 24 \to 27$, with the total length of 112. And further we suppose that the velocity of vehicle is 5, so the time to go along the preset route is 22.4. Now the road link $3 \to 2 \to 1 \to 4 \to 8 \to 10$ (total length 64) is congested. So we alternatively select $3 \to 7 \to 6 \to 12 \to 11 \to 10$ (total length 73) to replace the congested road segments. Results on the total traveling time for vehicles with increasing congestion time is shown in Fig. 3(c).

From Fig. 3(c) we can see that when congestion time is short, sticking to the preset route and wait for congestion recovery is a better choice. When congestion time is long, taking an alternative route to bypass the congestion costs less total traveling time.

5 Conclusion

In this paper, we propose a dynamic path planning mechanism in urban area traffic scenario. Our method is based on the principal of consuming the least time to travel to the destination rather than the shortest distance in the time-sensitive traffic situation. If congestion emerges, the source node would be notified ahead of time and launch alternative path selection algorithm to bypass the congestion. In addition, we devise a time-comparison mechanism to decide whether to take the alternative route or wait for the congestion recovery. Further, we discuss the time complexity and convergence of our proposed algorithm and simulation results are given, which shows that the revision-based dynamic path planning fits the traffic reality better. Future work will concentrate on the study of routing protocols in VANET for this dynamic path planning method.

Acknowledgments. This work is supported by the NSF of China under Grant No. 61301118; the Innovation Program of Shanghai Municipal Education Commission under Grant No. 14YZ130; the International S&T Cooperation Program of Shanghai Science and Technology Commission under Grant No. 15220710600; and the Fundamental Research Funds for the Central Universities under Grant No. 16D210403.

References

1. Xu, J., Guo, L., Ding, Z., Sun, X., Liu, C.: Traffic aware route planning in dynamic road networks. In: Lee, S., Peng, Z., Zhou, X., Moon, Y.-S., Unland, R., Yoo, J. (eds.) DASFAA 2012. LNCS, vol. 7238, pp. 576–591. Springer, Heidelberg (2012). doi:10.1007/978-3-642-29038-1_41
2. Wang, M., Shan, H., Lu, R., et al.: Real-time path planning based on hybrid-VANET-enhanced transportation system. IEEE Trans. Veh. Technol. **64**(5), 1664–1678 (2015)
3. He, Z., Cao, J., Li, T.: MICE: a real-time traffic estimation based vehicular path planning solution using VANETs. In: International Conference on Connected Vehicles and Expo, pp. 172–178. IEEE (2012)
4. Rajabi-Bahaabadi, M., Shariat-Mohaymany, A., Babaei, M., et al.: Multi-objective path finding in stochastic time-dependent road networks using non-dominated sorting genetic algorithm. Expert Syst. Appl. **42**(12), 5056–5064 (2015)
5. Lieu, H.: Traffic-flow theory. Public Roads **62**(1–2), 5–7 (1999)