Back-Pressure Based Traffic Scheduling Algorithm for Urban Vehicular Networks with Self-Driving Vehicles

YISHAN LIN †1,a) YING LIU †2,b) JUNTAO GAO†2,c) MINORU ITO†2,d)

Abstract: Back-pressure algorithm, which works as water flows through pipe networks according to pressure gradients, has been increasingly attractive to reduce traffic congestion for urban vehicular networks. Recent work has shown the performance superiority of back-pressure based traffic scheduling algorithms, such as throughput optimality, distributed implementation, low computational complexity, etc. However, these back-pressure based traffic scheduling algorithms either assume each road can hold infinite vehicles (infinite road capacity) or need to have prior knowledge of vehicle turning ratios, all of which are not realistic for applications. In this paper, we propose a back-pressure based traffic scheduling algorithm that can efficiently reduce traffic congestion for realistic urban vehicular networks with finite road capacity and without prior knowledge of vehicle turning ratios.

Keywords: Back-pressure routing, traffic scheduling control algorithm, urban vehicular networks

1. Introduction

Traffic congestion has become a significant problem due to increasing vehicles every year. Human-driven vehicles' actions only depend on the traffic lights with fixed-cycle control, while it cannot ensure road utilization. Recently, self-driving vehicles have been developed that more information (e.g., traffic conditions and navigations) can be collected and shared between vehicles to enable intelligent driving. Based on this environment, an efficient traffic scheduling control can be achieved to solve this congestion problem.

Several related studies for traffic scheduling has been conducted, in which the backpressure routing [1] has been adopted to control traffic signals at road intersection [2-5] for reducing traffic congestion. Back-pressure routing is an algorithm for directing traffic control that works as water flows through pipe networks based on pressure gradients to optimize network throughput. Accordingly, the pressure of roads can be denoted as the number of vehicles, and the scenario can be considered as the traffic that flows from a high-pressure upstream area to a low-pressure downstream area. Namely, the vehicular traffic flows to the roads with more remaining capacity in the network. This algorithm can not only support the arrival traffic but also be implemented in distributed manner with low computational complexity. However, those back-pressure based traffic scheduling algorithms for urban vehicular networks assume that each road can hold infinite vehicles (infinite road capacity) [2-4] or need to have prior knowledge of vehicle turning ratios (the ratio of vehicles that will turn right, turn left and go straight after entering a road segment) [5], all of which are not realistic for applications.

In this paper, we propose a back-pressure based traffic scheduling algorithm (BPTSA) to face such problems in realistic urban vehicular networks with self-driving vehicles. BPTSA can efficiently reduce traffic congestion with finite road capacity and without prior knowledge of vehicle turning ratios.

2. System Model

2.1 Road Network Model

A road network G consists of N roads and M junctions which are respectively denoted as a road set $R = \{R_1, R_2, \dots, R_N\}$ and a junction set $J = \{J_1, J_2, \dots, J_M\}$. For road R_i, the road length, speed limit and capacity are denoted as d_i, v_i and C_i, respectively. Vehicles enter the network from one origin roads R_i and depart the network from another destination road R_j, and they may pass multiple roads between origin R_i and destination R_j. Further, each road R_i is divided into several lanes, denoted as L_{ij}, representing the lane in which the vehicles waiting in road R_i will move to an adjacent road R_j, as shown in Fig. 1. Each lane is modeled as a queue. System time is slotted as $t \in \{0, 1, 2 \dots\}$, where each slot indicates a certain period of time. $Q_{ij}(t)$ denotes the number of vehicles queued at lane L_{ij}. Therefore, $Q_i(t) = \sum_j Q_{ij}(t)$ is the number of vehicles waiting at road R_i.

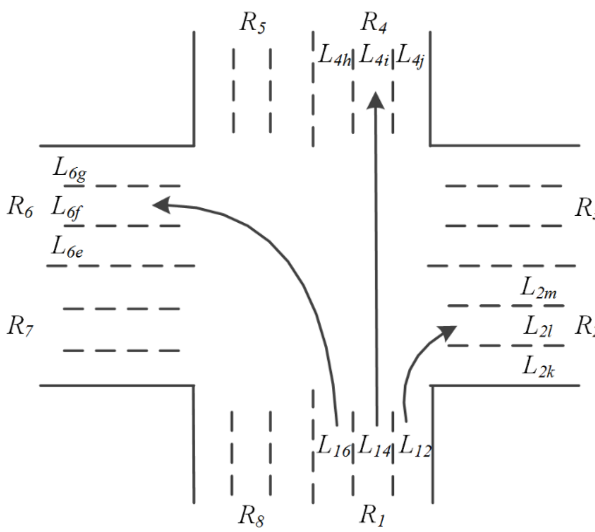

Fig. 1 Possible traffic movement from road R_1 at junction J_i.

†1 Institute of Computer Science and Engineering, National Chiao Tung University, Taiwan
†2 Graduate School of Information Science, Nara Institute of Science and Technology, Japan

a) lin.yi_shan.lu2@is.naist.jp
b) liu.ying.lm3@is.naist.jp
c) jtgao@is.naist.jp
d) ito@is.naist.jp

2.2 Vehicle Arrival and Routing Process

For the arrival and routing process of vehicles, we define $A_{ij}(t)$ as the number of vehicles that just enter the network and initially located at L_{ij} in time slot t. Since some vehicles may enter L_{ij} from other roads, we define $f_{ki\rightarrow ij}(t)$ to represent the number of vehicles moving from L_{ki} to L_{ij} in time slot t, so $\sum_k f_{ki\rightarrow ij}(t)$ represents the total number of vehicles moving to L_{ij} in time slot t. When a vehicle enters L_{ij} in time slot t (either just arriving or entering from other roads), the value of $Q_{ij}(t)$ is increased by 1. All queues are divided into three groups: ingress queues $Q^{(1)}$ (storing exogenous arriving vehicles), sink queues $Q^{(2)}$ (from which vehicles depart) and common queues $Q^{(3)}$ (storing endogenous arriving vehicles). The queue dynamics (the variation of $Q_{ij}(t)$ in the lane L_{ij}'s queue) can be descried as follows:

$$Q^{(1)}{}_{ij}(t+1) = Q^{(1)}{}_{ij}(t) - \sum\nolimits_k f_{ij\rightarrow jk}(t) + A_{ij}(t) \quad (1)$$

$$Q^{(2)}{}_{i}(t) = 0,\ \forall t \in \{0,\ 1,\ 2,\ \ldots\} \quad (2)$$

$$Q^{(3)}{}_{ij}(t+1) = Q^{(3)}{}_{ij}(t) - \sum\nolimits_k f_{ij\rightarrow jk}(t) + \sum\nolimits_k f_{ki\rightarrow ij}(t) \quad (3),$$

where $Q^{(2)}{}_i(t)$ equals to 0 because vehicles have already left the network from R_i in the next time slot.

2.3 Phase of Traffic Signal

At each junction, if a vehicle can move from road R_i to road R_j, then such a movement is called a traffic movement from R_i to R_j. Some traffic movements can occur simultaneously and are considered as a "traffic phase". Upstream road set U_i and downstream road set D_i of junction J_i are also defined; road R_i belongs to U_i if and only if a vehicle can travel through R_i then junction J_i and enter next road R_j. Here, R_j is said to be in D_i. For example, R_1 belongs to U_i; R_2, R_4 and R_6 are in D_i. Let $P_i = \{ p_i^1, p_i^2, \ldots, p_i^{max} \}$ be the set of all possible phases at a junction J_i. $p_i(t)$ denotes the phase activated during the time slot t for junction J_i. $\mu_{jk}(p_i(t))$ represents the maximum number of vehicles leaving from R_j to R_k if phase $p_i(t)$ is activated. The actual number of vehicles leaving from L_{jk} to L_{kr} during slot t is

$$f_{jk\rightarrow kr}(t) = \min\left\{C_k - Q_{kr}(t),\ Q_{jk}(t),\ \mu_{jk}\left(p_i(t)\right)\right\} \quad (4),$$

where C_k is the capacity of lane L_{kr}. If $Q_{kr}(t) = C_k$, L_{kr} is full, indicating that no more vehicles can enter L_{kr}.

3. Methods

Consider that each vehicle enters the network with a fixed destination and runs according to the fixed shortest-path route. Our proposed BPTSA proceeds the following two stages.

- Stage 1: each vehicle calculates the shortest path from origin to its destination using Dijkstra's algorithm [6]. Here, the cost of road R_i is defined as

$$T_i = \frac{d_i}{v_i} \quad (5),$$

that is, time cost of travelling through road R_i.

- Stage 2: for each junction J_i, the following procedure is executed.
  - Calculate the traffic pressure $F_j(t)$ of road $R_j \in U_i \cup D_i$ in time slot t based on the number of vehicles and road capacity. $F_j(t)$ is defined as

$$F_j(t) = \frac{Q_j(t)}{C_j} \quad (6)$$

  - Calculate the pressure difference $W_{jk}(t)$ between R_j and R_k in time slot t, which is defined as

$$W_{jk}(t) = \max\{F_j(t) - F_k(t),\ 0\},\ \forall R_j \in U_i \text{ and } R_k \in D_i \quad (7)$$

  - Release traffic pressure defined as follows.

$$p_i(t) = \arg\max_{p_i^r \in P_i} \sum_{j,k} W_{jk}(t) \cdot \mu_{jk}(p_i^r) \quad (8)$$

Traffic phase $p_i(t)$ for junction J_i is activated that maximizes the pressure release.

4. Results

Since we have designed the model for this algorithm, we will evaluate BPTSA by simulations. The result of comparison with previous studies will be presented.

5. Conclusion

In this paper, BPTSA, a traffic scheduling algorithm based on back-pressure routing algorithm is proposed. BPTSA can work with the constraint of finite road capacity without prior knowledge of vehicle turning ratios by means of the combination of shortest-path route and back-pressure routing algorithm. In future work, we will implement BPTSA by simulation and show the evaluation result. Besides, the comparison with previous traffic scheduling algorithms will be also presented.

Acknowledgments

This work was supported by JSPS KAKENHI Grant Numbers JP15K15981, JP16K01288.

Reference

[1] Backpressure routing. Available online: https://en.wikipedia.org/wiki/Backpressure_routing (accessed on Jan. 2017).

[2] Varaiya, P.: A universal feedback control policy for arbitrary networks of signalized intersections (2009).

[3] Wongpiromsarn, T., Uthaicharoenpong T., Wang, Y., Frazzoli, E. and Wnag, D.: Distributed traffic signal control for maximum network throughput, *in Proc. IEEE 15th Int. Conf. Intell. Transport. Syst.*, pp. 588-595 (2012).

[4] Zaidi, A. A., Kulcsár, B. and Wymeersch, H.: Back-pressure traffic signal control with fixed and adaptive routing for urban vehicular networks (2015).

[5] Gregoire, J., et al.: Capacity-aware backpressure traffic signal control, *IEEE Trans. Contr. Netw. Syst.*, Vol. 2, No. 2, pp. 164-173 (2015).

[6] Dijkstra's algorithm. Available online: https://en.wikipedia.org/wiki/Dijkstra's_algorithm (accessed on Jan. 2017).