FogFly: A Traffic Light Optimization Solution based on Fog Computing

Quang Tran Minh
VNU-HCM, Ho Chi Minh City
University of Technology
Ho Chi Minh, Vietnam
quangtran@hcmut.edu.vn

Binh Thai Nguyen
VNU-HCM, Ho Chi Minh City
University of Technology
Ho Chi Minh, Vietnam
1410289@hcmut.edu.vn

Chanh Minh Tran
VNU-HCM, Ho Chi Minh City
University of Technology
Ho Chi Minh, Vietnam
1410333@hcmut.edu.vn

Triet Minh Tran
VNU-HCM, Ho Chi Minh City
University of Technology
Ho Chi Minh, Vietnam
1414169@hcmut.edu.vn

Tuan An Le
VNU-HCM, Ho Chi Minh City
University of Technology
Ho Chi Minh, Vietnam
1414381@hcmut.edu.vn

Rajesh Krishna BALAN
Singapore Management
University
Singapore
rajesh@smu.edu.sg

UbiComp/ISWC'18 Adjunct, October 8–12, 2018, Singapore, Singapore
ACM ISBN 978-1-4503-5966-5/18/10...$15.00
https://doi.org/10.1145/3267305.3274169

Abstract
This paper provides a fog-based approach to solving the traffic light optimization problem which utilizes the Adaptive Traffic Signal Control (ATSC) model. ATSC systems demand the ability to strictly reflect real-time traffic state. The proposed fog computing framework, namely FogFly, aligns with this requirement by its natures in location-awareness, low latency and affordability to the changes in traffic conditions. As traffic data is updated timely and processed at fog nodes deployed close to data sources (i.e., vehicles at intersections) traffic light cycles can be optimized efficiently while virtualized resources available at network edges are efficiently utilized. Evaluation results show that services running in FogFly produce better performance comparing to those in cloud computing approaches.

Author Keywords
Fog Computing; Edge Computing; Cloud Computing; Intelligent Transportation System; Adaptive Traffic Signal Control; Traffic Light Optimization.

ACM Classification Keywords
Computer systems organization [Architectures]: Distributed architectures

Introduction

Transportation plays an important role in any economy all over the globe. Intelligent Transportation Systems (ITS) attracts a numerous researches in various fields such as vehicular technology, transportation, civil engineering, statistical study, computational science, communications [9], and so on. Even though, traffic congestion still challenges large cities, causing not only economic loss but also pollution (e.g., air, noise pollution), violence and other social issues [11]. Hopelessly stuck ambulances; frustrated/stressful people on messy streets to school/office, to name just a few. Such uncomfortable and even disastrous traffic environments decline the quality of life.

One of many suggested solutions to traffic jam problem is applying the Adaptive Traffic Signal Control (ATSC) that changes the signal cycles of traffic lights to dynamically adapt with current traffic states. Such systems require a computation model to collect vehicle data to identify traffic state (e.g., light or heavy traffic flows) and control the traffic lights with the ability to seamlessly scale and provide services in a strictly low latency.

Traditionally, ITS applications in general or ATSC services in particular are provided by cloud-based systems where all data gathering as well as computing tasks are performed at centralized data centers (DCs) on the cloud. Although this scheme is robust for applications requiring large computational resources and historical data, it is inefficient for delay-sensitive ITS services such as the ATSC ones as it requires significant delays to communicate data and decisions from vehicles or road-side IoT devices to the DCs, and vices versa.

Fog computing [2] has been introduced to accelerate this approach by helping to distribute tasks to the edges of the network (i.e., fog nodes), which are close to data sources. Consequently, it is crucial to propose an ATSC system based on Fog Computing to solve traffic jam problem. This work devises a fog-computing based approach, namely the FogFly to which fog nodes are distributed at traffic light control units and regions to efficiently collect and process traffic related data from vehicles for effective ATSC services. The FogFly is distinguished with existing work by its main contributions as follows:

- We thoroughly analyze parameters that need to be real-time updated for efficient traffic lights controls in ATSC services
- We proposes an appropriate fog-computing model which is effective for ATSC systems.
- We develop a prototype and conduct various evaluations to prove the effectiveness of the proposed approach in terms of latency, energy consumption and cost of operations.

Related Work

Adaptive Traffic Signal Control (ATSC) [1] is a popular method of traffic control optimization that has widely been applied around the world. By collecting traffic data which represent current traffic state, traffic light signal cycles of each intersection can be calculated and updated continuously. This way, the system can predict traffic congestion or detect unusual incidents in order to control traffic flows by changing traffic light cycles.

There have been many approaches to calculation of traffic light signal cycles which require specific information of roads, traffic flows and turning rate. According to [14], every intersection has to find its saturation flow and turning rate in order to calculate traffic light cycles. By applying Webster

formulas, the study in [7] has modeled traffic optimization in an area by optimizing traffic light cycles in every of its intersection.

In every research above, proposed system needs to collect a huge amount of traffic data. Centralized computing method such as Cloud Computing has been used extensively to support and deliver transportation application in general where traffic data is gathered and processed at data centers on the cloud. Although this scheme is robust for applications requiring large computational resources and historical data, it is inefficient for delay-sensitive ATSC services due to the latency sensitivity and the high volume nature of traffic data, processing data and carrying out reaction in a real-time manner is a big challenge for Cloud-based infrastructure [4]. Such consideration is also stated in [3], that is using the Cloud Computing model in the ITS case may not be efficient.

Fortunately, solution for such shortcomings of Cloud Computing in ITS has been proposed in [2] - the Fog Computing model. Fog Computing utilizes computing, storage and network resources within and at the edges of the network. The mechanism of this model is moving processing closer to network edges and data sources. This way, computing tasks can be distributed closer to users and things, which greatly reduce the burden of the Cloud while still achieve the latency-sensitivity requirements. Studies in [3] and [4] also support the idea of apply Fog Computing to solving traffic problems, thanks to its mobility, location awareness and its ability to lower service's latency.

Studies in [12] and [15] present a visionary concept on fog vehicle computing where vehicles are utilized to augment the computing and storage power of fogs, providing on-demand ITS services. They also discuss remained challenges such as service provision, privacy, security, and task scheduling on the fogs. Study in [6] proposes a fog-based privacy-preserved pseudonym scheme where the pseudonym management is shifted to the fogs to utilize contextual information such as location, number of vehicles at hotspots for making decisions.

The FogFly proposed in this work is different from existing work discussed above as it focuses on traffic management services, namely traffic light optimization leveraging the potential of fog computing. FogFly constitutes the use of traffic signals to control traffic flows at intersections. The service utilizes the familiar red, amber and green traffic signals to provide time-sharing for traffic at intersections. The traffic conditions of entries (road segments) at each intersection is update timely by fog nodes deployed at corresponding intersection, hence significantly mitigate latency and deployment cost.

Traffic Light Optimization Problem

This section describes the traffic light optimization (TLO) problem and analyzes parameters that need to be real-time updated for efficient ATSC services.

Traffic light cycle calculation

Traffic light optimization which is adaptive with current traffic states at intersections could help to optimize the road utilization and avoid traffic congestion. This part presents a model that optimizes traffic light cycle for an intersection. This work considers TLO problems for typical intersections in urban traffic networks as depicted in Fig. 1 and described as follows.

As shown in Fig. 1, vehicles enter the intersection from four directions, namely from the west (FW - as illustrated in the figure), from the east (FE), from the south (FS) and from the north (FN). Each entry consists of two lanes denoted as *inner-lane* (a) and the *outer-lane* (b). For instance, vehi-

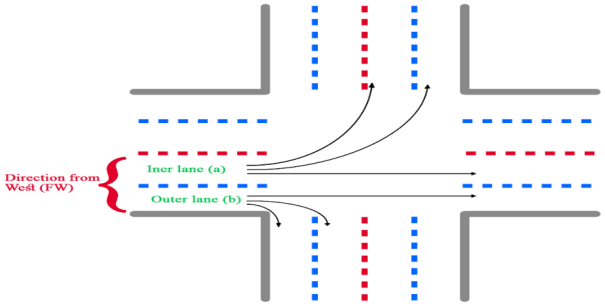

Figure 1: An intersection with four directions (entries), two lanes for each

cles traveling from the west (FW) may either belong to the lane FW_a if is running on the inner-lane or FW_b otherwise.

The TLO function will identifies the duration of traffic light cycles in accordance the current state of traffic in different directions. In order to clarify this model, we starts with following definitions:

Definition 1 - Phase: A phase consists of directions in the same line. For example, there are two phases, namely phase $E-W$ consists of direction FE and FW, and phase $N-S$ consists of direction FN and FS.

Definition 2 - Light cycle: A light cycle consists of *red* time, *green* time and *lost* time.

The *lost* time is the sum of clear time (the time during which no vehicles may enter the intersection) and the responding time of vehicles (acceleration time, reaction of drivers/riders, etc.). It is worth to notice that the lost time must be smaller than green and red time, and can be ignorable.

Each cycle has two moving phases (i.e., the green duration in each phase) during which only vehicles coming from certain directions are allowed to pass the intersection. Figure 2 shows the relation between red time and green time for each phase during a traffic light cycle. In order to optimize the duration of each light (green or red) of each phase, the traffic volume at each phase needs to be estimated in real-time.

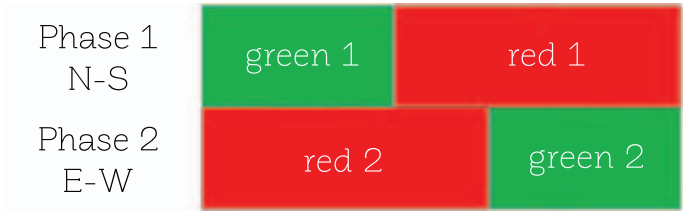

Figure 2: Phases and duration of traffic lights on each phase in a traffic light cycle

Definition 3 - Traffic volume: Traffic volume (v) on the lane m denoted as v_m is the number of vehicles passing the intersection on lane m in one hour [13].

Since there are two phases in a cycle (i.e., E-W and N-S phases), each of which consists of two possible movements (e.g., the E-W phase has two opposite movements, namely from East to West and vice versa), there are a total of eight v values corresponding to eight lanes.

In this sense, if volume V of phase i denoted as V_{ci} is defined as the largest v value of that phase, one intersection will have two V values for two phases in a traffic light cycle.

Moreover, in order to identify the traffic flow at different phases, an additional terms, namely the **headway** [13], needs to be defined as follows.

Definition 4 - Headway: is defined when the traffic light signal turns from green to red as below:

- the first headway is the duration between the moment the traffic light signal turns to green and the time when the first vehicle crosses the stop line
- a following headway is the duration between the moment a vehicle passing the stop line and the time when the next one does.
- headway is measured in seconds.

Definition 5 - Saturation headway: is the stable headway, denoted as h [13], calculated after having a number of vehicles passed the stop line.

For example, after the fourth or fifth vehicle has passed the stop line, headway will become stable and can be considered a constant. This constant is called **saturation headway** h [13].

Definition 6 - Saturation flow rate: the number of vehicles passing the stop line in an hour is called saturation flow rate denoted as s and is calculated in equation (1)

$$s = \frac{3600}{h} \tag{1}$$

The saturation flow rate of lane m denoted as s_m and of phase i denoted as S_{ci} are measured as vehicles/hour/lane and defined similarly to v and Vci [13] described previously. According to the Kimber formula [8], s_m is calculated in equation (2)

$$s_m = \frac{2084 - 42dg * G + 100(W_m - 3.25) - 140dn}{1 + 1.5\frac{f_m}{r}} \tag{2}$$

where dg is 1 (lane to uphill) or 0 (lane to downhill), G is the percentage gradient of entry road, W_m is lane m's width (in meter), dn is 1 (outer-lane) or 0 (inner-lane), r is the radius of road (in meter), f_m is the turn frequency in lane m.

The highway capacity manual (HCM) [10] has given a model for determining the optimized cycle length in equation 3.

$$C = \frac{X_C * N * L}{X_C - \sum_{i=1}^{N} \frac{v_{ci}}{s_{ci}}} \tag{3}$$

where, X_C is the *quality factor*, i.e., the *ratio of flow rate and capacity*, N is the number of phases in a cycle (in this work N=2), L is the total lost time (s) which can be defined beforehand.

From the above equations (i.e., equations (2) and (3)), to calculate a new traffic light cycle for each intersection, we need to gather 8 pairs of values v and f corresponding to 8 lanes.

The Proposed FogFly

This section presents our proposed framework on fog-computing, the FogFly, to collect real-time data about the traffic condition, namely the 8 pairs of values v and f for each intersection in real-time to optimize traffic light cycles, providing appropriate n ATSC services.

Overall architecture

The overall architecture of the FogFly system is depicted in figure 3 and is described as follows.

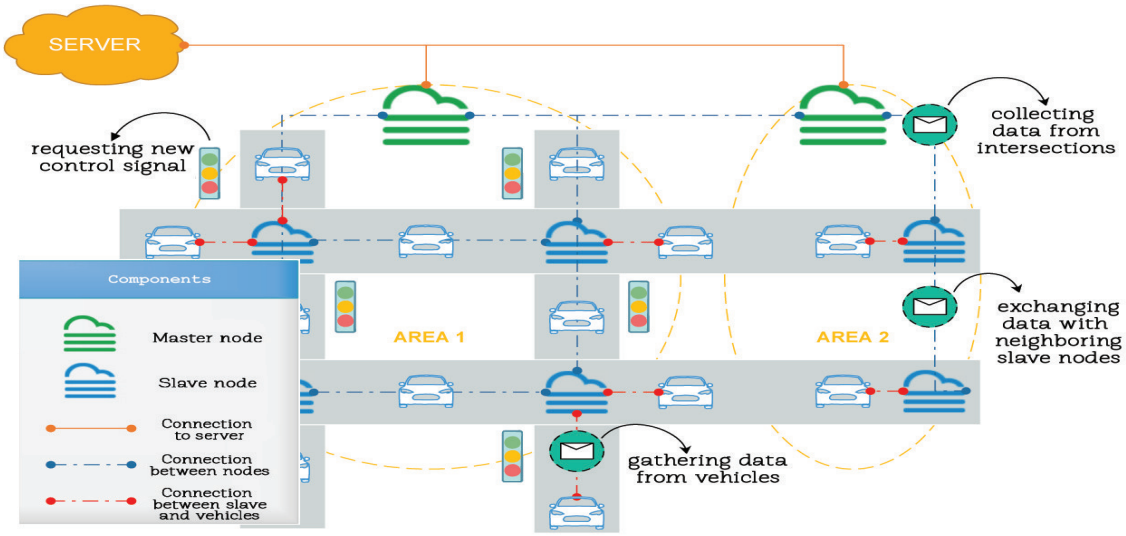

Figure 3: Overall architecture of the FogFly

FogFly consists of two types of fog nodes: *slave* and *master* nodes. Slave node is installed at each intersection to collect data from vehicles and perform necessary calculations to produce optimized traffic light cycles. Short-range communications such as WiFi, RFID, etc., is employed for the communication between the slave node and vehicles to assure that only vehicles travelling through the intersection are involved. Master node, on the other hand, is deployed in a region (e.g., each ward has a master node) to manage a group of slave nodes and to continuously conduct statistical analysis of traffic conditions under its supervision. For example, the master node can analyze traffic condition of the whole region and provide different policies on optimizing traffic light cycles to different slave nodes at different intersections.

Data collection model
As stated, the system needs to identify 8 pairs of $\{v$ and $f\}$ for each intersection in real-time to optimize traffic light cycle for the ATSC services.

By collecting data from vehicles, slave node can only know the total number of current vehicles coming from a specific direction (e.g., the total number of vehicles coming into the intersection from the West, or the East and so forth, regardless of on which lanes they come from). To identify the volume *v* in each lane, FogFly provides a mean for adjacent slave nodes share the necessary information to which the considering slave node can estimate the number of vehicles on each lane. This process is briefly described as follows.

A slave node periodically requests from its neighboring

slave nodes the number of vehicles passing through them. Based on this information, the considering slave node can compute the distribution of vehicles entering the intersection from different directions. For example, the percentage of vehicles that go straight or turn either left or right after entering the intersection from the West can be calculated. The slave node uses these distributions and the total number of vehicles entering the neighbour nodes at different directions to calculate the necessary pairs of $\{v$ and $f\}$.

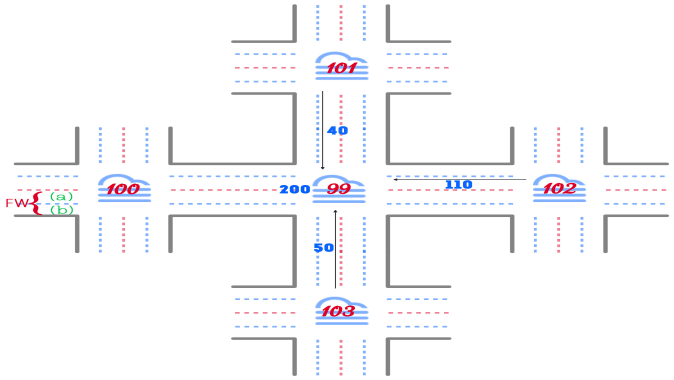

Figure 4: Adjacent slave nodes sends information to node 99

Figure 4 shows an example in which the slave node 99 uses information requested from adjacent slave nodes, namely nodes 100, 101, 102, and 103, to update the distributions necessary for calculating the $\{v$ and $f\}$ pairs. Node 99 inquires node 103 about the number of vehicles traveling from the intersection managed by node 100 to the one managed by node 99 and continue to the one under node 103, which is 50 vehicles. Obtaining this information, node 99 can infer that there are 50 vehicles entering the considering intersection on the lane FW_b during sometime in the recent past. In the same manner, 99 will gather the other seven past volumes. These values are then used to calculate the distribution of directions taken by vehicles after entering the intersection from each direction which is described as follows.

Assuming that the number of vehicles going straight from a particular direction in each lane is the same, the distribution of direction taken by a vehicle entering the intersection from the West is given as follows (also refer to Fig. 4):

- the ratio of vehicles going straight and turning left = $(110/2 + 50)/(110 + 40 + 50) = 0.525$
- the ratio of vehicles going straight and turning right = $(110/2 + 40)/(110 + 40 + 50) = 0.475$

These values are the distribution of vehicles coming from the West on lanes FW_a and FW_b during sometime in the recent past. The current number of vehicles coming from the West stored in the slave node 99's database is 200. Therefore, it can estimate the current volumes of lanes FW_a and FW_b as follows:

- Estimated current volume of lane $FW_a = 0.525 * 200 = 105$ (vehicles)
- Estimated current volume of lane $FW_b = 0.475 * 200 = 95$ (vehicles)

In the same manner, the slave node 99 will calculate the volume values of 6 remaining lanes.

As for the turning frequency f in each lane, the slave node 99 can easily compute it using the earlier information gathered from the adjacent nodes. Taking the turning frequency

in lane FW_a denoted as f_{FW_a} for example, this is the ratio between the number of vehicles turning left after entering the intersection on lane FW_a over the total number of vehicles on this lane. Using the same assumption mentioned before (i.e., the number of vehicles going straight from a particular direction in each lane is the same), the turning frequency in lane FW_a is given as follow: $f_{FW_a} = 40/(110/2 + 40) = 0.421$. The other turning frequencies on other lanes can be computed following this pattern.

Evaluation

This section evaluates the effectiveness of the proposed FogFly framework in terms of mitigating latency, energy consumption and operational cost when deploying ATSC services.

Evaluation environment and setting

To evaluate the effectiveness of the FogFly compared to Cloud-based approach for ATSC services, we use iFogSim [5] to describe network topology, specify system instances and links between them, and simulate tasks that run throughout the system. We have simulated FogFly that mimics real-world applications on traffic optimization. We have evaluated the latency, energy consumption and cost of operations in FogFly compared to the cloud-based counterpart.

In this simulation, we assume that there are A intersections, each of which has 1 fog node collecting information from B cars (e.g. 4 fog nodes x 30 cars). The simulation is conducted in two modes, namely the Fog mode and the Cloud mode as illustrated in Fig. 5 and is described as follows:

- **Fog mode**: In this mode, traffic related data is gathered and processed at fog nodes placed at intersections, utilizing the proposed FogFly framework. These fog nodes also control their traffic lights.
- **Cloud mode**: In this mode, data gathering, processing and traffic light controls are placed at the cloud.

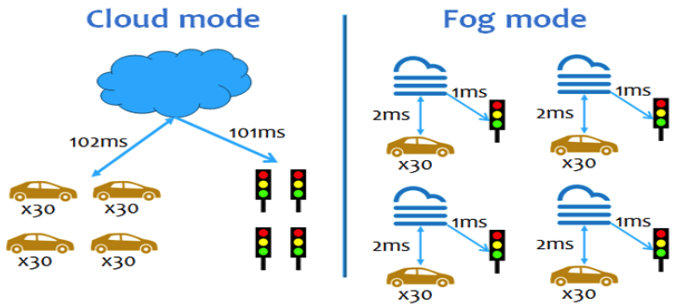

Figure 5: Simulation topology

Specification of devices used at each layer in the above topology are shown in Table 1.

Table 1: Devices' specification

Device	Cloud	Fog Node	Vehicle
CPU (MIPS)	684,000	2,200	1,000
RAM (MB)	128,000	1,000	500
Bandwidth (Mbps)	150	50	10
Power (W)	120 - 50	5.1 - 1.9	1.5 - 0.5
Cost per MIPS ($/MIPS/s)	0.0003	0.0003	0.0003

Clients/vehicles periodically send GPS data received from their own GPS modules to Fog nodes or Cloud corresponding to different deployment modes. Fog nodes or Cloud periodically updates the new traffic light cycles to the traffic lights. Description for each task is described in Table 2.

Table 2: Task specification

Task	**CPU Length** (MIP)	**Network Length** (Bytes)
GPS	2	0
SEND_DATA	3	500
UPDATE_NODE_ID	2	500
TRAFFIC_LIGHTS_CTRL	2	500

Evaluation results

The effectiveness of the FogFly compared to the Cloud-based approach, in terms of latency mitigation is shown in Fig. 6. The proposed FogFly obtains significantly lower latency (around 4ms) compared to its cloud-based counterpart (around 200ms).

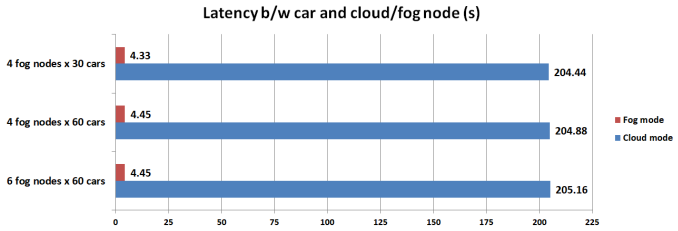

Figure 6: Effectiveness of the FogFly compared to the cloud-based approach in terms of latency

Figure 7 shows that the centralized data processing at the cloud is more power-consuming than using a set of multiple lightweight nodes in FogFly. The energy consumes in the Fog mode is the sum of all fog nodes' energy consumption as the cloud has not been used in this mode. The results also reveal that the more fog nodes being deployed, the more energy the system consumes.

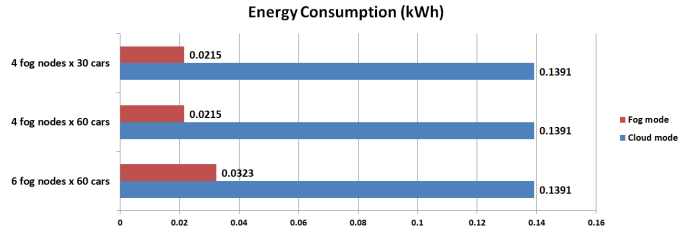

Figure 7: Effectiveness of the FogFly compared to the cloud-based approach in terms of energy consumption

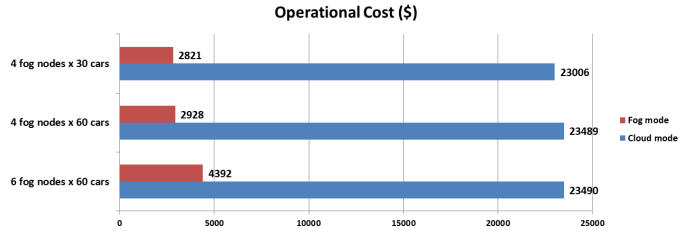

Figure 8: Effectiveness of the FogFly compared to the cloud-based approach in terms of cost saving

Conclusion

This paper proposed a novel approach to traffic optimization using fog computing. Applying fog architecture to the ATSC solution, FogFly enables traffic system to automatically and efficiently adapt to the current traffic conditions. Simulation results conducted by iFogSim show that the proposed fog computing model can provide remarkably better performances than the traditional cloud computing approach in terms of latency, energy consumption and operation cost saving.

Acknowledgment

This research is funded by Vietnam National Foundation for Science and Technology Development (NAFOSTED) under grant number 102.01-2016.28.

REFERENCES

1. Federal Highway Administration. 2017. Adaptive Signal Control Technology. (2017). https://www.fhwa.dot.gov/innovation/everydaycounts/edc-1/asct.cfm.
2. F. Bonomi, R. Milito, J. Zhu, and S. Addepalli. 2012. Fog computing and its role in the internet of things. Proceedings of the First Edition of the MCC Workshop on Mobile Cloud Computing, 14–15.
3. T. S. J. Darwish and K. Abu Bakar. 2018. Fog Based Intelligent Transportation Big Data Analytics. In *The Internet of Vehicles Environment: Motivations, Architecture, Challenges, and Critical Issues*, Vol. 6. IEEE Access, 15679–15701.
4. N. Giang, V. C.M. Leung, and R. Lea. 2016. On Developing Smart Transportation Applications in Fog Computing Paradigm. In *the 6th ACM Symposium on Development and Analysis of Intelligent Vehicular Networks and Applications*. 91–97.
5. H. Gupta, A.V. Dastjerdi, S. K. Ghosh, and R. Buyya. 2016. iFogSim: A Toolkit for Modeling and Simulation of Resource Management Techniques in Internet of Things, Edge and Fog Computing Environments. (2016).
6. J. Kang, R. Yu, X. Huang, and Y. Zhang. 2017. Privacy-Preserved Pseudonym Scheme for Fog Computing Supported Internet of Vehicles. *IEEE Transactions on Intelligent Transportation Systems* (2017), 1–11.
7. K. B. Kesur. 2012. Optimization of mixed cycle length traffic signals. Journal of advanced transportation, 432–437.
8. R. M. Kimber, M. McDonald, and N. Hounsell. 1958. The prediction of saturation flow for road junction controlled by traffic signal. Transport and Road Research Laboratory Report 67.
9. M. Ndoye, V.F. Totten, J.V. Krogmeier, and D.M. Bullock. 2011. Sensing and signal processing for vehicle reidentification and travel time estimation. *IEEE Transactions on Intelligent Transportation Systems* 12, 1 (2011), 119–131.
10. National Academy of Sciences. 2000. *HCM 2000: Highway Capacity Manual*. Transportation Research Board. 10.45 pages.
11. T. M. Quang and K. Eiji. 2011. Traffic State Estimation with Mobile Phones Based on The "3R" Philosophy. *IEICE Transactions on Communications* E94-B, 12 (2011), 3447–3458.
12. M. Sookhak, F. R. Yu, Y. He, H. Talebian, N. Sohrabi Safa, N. Zhao, M. K. Khan, and N. Kumar. 2017. Fog Vehicular Computing: Augmentation of Fog Computing Using Vehicular Cloud Computing. *IEEE Vehicular Technology Magazine* 12, 3 (2017), 55–64.
13. Tom V. Mathew. 2014. *Transportation Systems Engineering*. 34.1 – 34.13 pages.
14. F. V. Webster. 1958. *Traffic signal settings*. 1–45 pages.
15. Y. Xiao and Chao Zhu. 2017. Vehicular fog computing: Vision and challenges. In *2017 IEEE International Conference on Pervasive Computing and Communications Workshops (PerCom Workshops)*. 6–9.