Temporal Data Dissemination in Vehicular Cyber–Physical Systems

Kai Liu, *Member, IEEE*, Victor Chung Sing Lee, *Member, IEEE*, Joseph Kee-Yin Ng, *Senior Member, IEEE*, Jun Chen, and Sang Hyuk Son, *Fellow, IEEE*

***Abstract*—Efficient data dissemination is one of the fundamental requirements to enable emerging applications in vehicular cyber–physical systems. In this paper, we present the first study on real-time data services via roadside-to-vehicle communication by considering both the time constraint of data dissemination and the freshness of data items. Passing vehicles can submit their requests to the server, and the server disseminates data items accordingly to serve the vehicles within its coverage. Data items maintained in the database are periodically updated to keep the information up-to-date. We present the system model and analyze challenges on data dissemination by considering both application requirements and communication characteristics. On this basis, we formulate the temporal data dissemination (TDD) problem by introducing the snapshot consistency requirement on serving real-time requests for temporal data items. We prove that TDD is NP-hard by constructing a polynomial-time reduction from the Clique problem. Based on the analysis of the time bound on serving requests, we propose a heuristic scheduling algorithm, which considers the request characteristics of productivity, status, and urgency in scheduling. An extensive performance evaluation demonstrates that the proposed algorithm is able to effectively exploit the broadcast effect, improve the bandwidth efficiency, and enhance the request service chance.**

***Index Terms*—Data dissemination, real-time scheduling, temporal consistency, vehicular cyber–physical system (VCPS).**

I. Introduction

VEHICULAR cyber–physical systems (VCPSs), which embrace the latest advances in communications, computing, electronics, sensing, control, etc., are envisioned as a promising approach to achieving breakthroughs in transportation safety, efficiency, and sustainability [1]. Efficient data dissemination is critical to enable emerging applications in VCPS, such as collision avoidance [2], roadway reservation [3], and autonomous intersection management [4], to name a few. This paper investigates the scenario, where the roadside unit (RSU) is installed along the road and provides real-time data services to passing vehicles. Data items maintained in the database are periodically updated to keep the information up-to-date. Vehicles within the service range of the RSU can submit requests for particular services or information, such as routing advisories, road conditions, parking slots, etc. In response, the RSU disseminates corresponding data items to serve the vehicles. Obviously, to enable such services, it is expected to guarantee the freshness of information and the timeliness of data dissemination.

The vehicular communication system plays an important role in interconnecting the driver, the vehicle, and the cyber information in VCPS. Different parties, including automotive manufacturers, governments, and universities, are actively engaged into the research of vehicular communications. In automotive manufacturers, *MyFord Touch* is an embedded communication system developed by the Ford Motor Company and Microsoft. It enables drivers to interact with vehicles via smartphones. *Toyota Entune* is an integrated multimedia navigation system developed by the Toyota Motor Company. It provides data services, such as stocks and traffic information. *Mbrace2* developed by the Mercedes-Benz Company provides drivers with both safety-critical and value-added information via vehicular communications. The U.S. Department of Transportation is also collaborating with a number of universities on the development of a variety of vehicular communication systems. Examples include the *Connect Vehicle* Research Program, the *Vehicle Infrastructure Integration* project, the MIT*CarTel* project, the Berkeley *PATH* project, etc. Clearly, there is great significance of the research on providing real-time data services via vehicular communications.

Although there has been extensive research on data scheduling in conventional mobile computing environments [5]–[7], none of them addressed unique challenges arising in VCPS for providing real-time data services. In the community of vehicular ad hoc networks (VANETs), great efforts have been put into the roadside-to-vehicle and vehicle-to-vehicle communications [8]–[10]. Nevertheless, these studies mainly focused on the issues resided on medium access control (MAC) or physical (PHY) layers, such as improving wireless communication

Manuscript received June 25, 2013; revised November 11, 2013 and February 14, 2014; accepted March 29, 2014. Date of publication May 13, 2014; date of current version December 1, 2014. This work was supported in part by CPS Global Center of DGIST and GRL Program through NRF funded by MSIP of Korea (2013K1A1A2A02078326), and in part by HKBU Research Centre for Ubiquitous Computing, the Institute of Computational and Theoretical Studies, and the HKBU Strategic Development Fund (HKBU SDF 10-0526-P08). The Associate Editor for this paper was R. J. F. Rossetti. (*Corresponding author: Jun Chen*.)

K. Liu is with the Key Laboratory of Dependable Service Computing in Cyber Physical Society, Chongqing University, Ministry of Education, and also with the College of Computer Science, Chongqing University, Chongqing 400030, China (e-mail: liukai0807@gmail.com).

V. C. S. Lee is with the Department of Computer Science, City University of Hong Kong, Kowloon, Hong Kong (e-mail: csvlee@cityu.edu.hk).

J. K.-Y. Ng is with the Department of Computer Science, Hong Kong Baptist University, Kowloon, Hong Kong (e-mail: jng@comp.hkbu.edu.hk).

J. Chen is with the School of Information Management, Wuhan University, Wuhan 430072, China (e-mail: christina_cj@whu.edu.cn).

S. H. Son is with the Information and Communication Engineering, Daegu Gyeongbuk Institute of Science and Technology, Daegu 711-873, Korea (e-mail: son@dgist.ac.kr).

Color versions of one or more of the figures in this paper are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/TITS.2014.2316006

quality or reliability in VANET. To the best of our knowledge, this is the first work on data dissemination in VCPS, which considers both the time constraint of services and the freshness of data items at the application layer. In particular, the following issues are investigated. First, there is a strict time constraint on serving requests. On the one hand, vehicles cannot retrieve data items after leaving the service region of the RSU. On the other hand, there are a variety of location-dependent data services, such as route queries, which have to be satisfied within a certain time bound (e.g., before reaching the road intersection). Second, a request may ask for multiple dependent data items, and the query cannot be fully processed until all the requested data items are retrieved. For example, in order to compute the optimal route to a destination, the navigation system has to request the road conditions of all possible routes. Since different routing information correspond to different data items, the optimal route can be computed only when all the corresponding data items are retrieved. Third, due to the temporality of data items, the multiple retrieved data items should be consistent in versions. Inconsistent readings of dependent data items may result in fatal failure of services.

The main contributions of this paper are outlined as follows. First, we present the roadside-to-vehicle communication system in VCPS and investigate the newly arising challenges of data dissemination. Second, we formulate the temporal data dissemination problem called TDD by investigating the snapshot consistency requirement on serving real-time requests fortemporal data items, and we prove that TDD is NP-hard by constructing a polynomial-time reduction from the Clique problem. Third, by analyzing time bound of serving real-time requests for temporal data items, we propose a heuristic scheduling algorithm, which aims at enhancing overall system performance by improving the broadcast effect, the bandwidth efficiency and the request service chance. Finally, we build the simulation model for performance evaluation. The comprehensive simulation results validate that the proposed solution is effective in providing real-time data services under different traffic scenarios and application requirements.

The rest of this paper is organized as follows. Section II presents the system model. In Section III, we formulate the TDD problem and prove that it is NP-hard. In Section IV, we analyze the service time bound and propose a heuristic scheduling algorithm. In Section V, we build the simulation model and evaluate the algorithm performance. Section VI reviews the related work. Section VII concludes this paper and discusses future research directions.

II. Data Dissemination Model

The data dissemination model is shown in Fig. 1, where the RSU is installed along the road and provides data services. In general, there are both location-independent and location-dependent services [11]. For the location-independent service, such as multimedia downloads, it may tolerate longer delay to be completed. However, for the location-dependent service such as navigation queries, it may impose both temporal and spatial constraints on successful completion. In this model, we focus on the data dissemination for location-dependent services, where stringent time constraints are required. In addition, it is commonly assumed that RSUs are sparsely deployed along the road [8], [10]. Accordingly, we consider that the data service has to be completed within the coverage of one RSU.

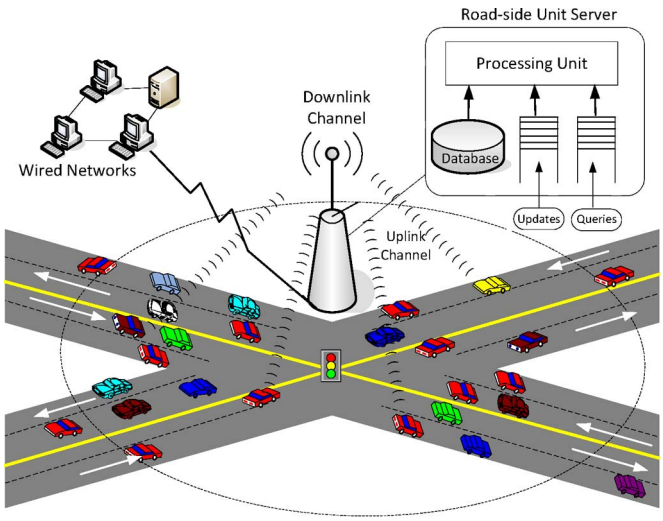


Fig. 1. Data dissemination model in VCPSs.

When vehicles are driving into the service range of the RSU, they may submit requests for particular services, such as road condition queries or routing advisories. Outstanding requests are pended in the service queue. According to a certain scheduling algorithm (e.g., earliest deadline first (EDF) [12]), the RSU retrieves data items from the local database and broadcasts them to passing vehicles. Note that the broadcast nature of wireless communication is commonly exploited for data dissemination in vehicular networks [8], [13], [14]. Therefore, each broadcast data item can be retrieved by all vehicles within the service region. Typically, there are two types of broadcast approaches, i.e., push based and pull based [15]. Safety-critical services are usually provided via push-based broadcast, where the messages are periodically disseminated. On the other hand, many value-added services are provided via the pull-based broadcast, where the messages are only disseminated based on explicit requests. This data dissemination model is dedicated to the latter application scenario.

The RSU is connected to a backbone network, in which it can access the Internet and other specific sensor networks. Due to the highly dynamic nature of traffic information, the data items stored in the local database are periodically updated by the sensors and information providers from the backbone network. Once an update is installed for a data item, a new version is created, and the previous version becomes outdated. Only the latest version of each data item is maintained in the database. Each request is associated with a deadline, which may be either imposed by specific application requirements or bounded by the dwell time of vehicles in the service region [15]. For requests asking for multiple dependent data items, they have to retrieve all the data items before their respective deadlines. In addition, due to the temporality of data items, the multiple dependent data items in a request have to be consistent in versions. Detailed requirements on serving time-critical multi-item requests for temporal data items are formulated as follows.

TABLE I
SUMMARY OF NOTATIONS

Notations	Descriptions	Notes
D	set of data items	$D=\{d_1,d_2,\cdots,d_{\lvert D\rvert}\}$
$V(d_i\lvert t)$	value of d_i at t	
$U(d_i\lvert t)$	update time of d_i at t	$U(d_i\lvert t)\le t$
$E(d_i\lvert t)$	expiration time of d_i at t	$E(d_i\lvert t)>t$
$l(d_i)$	update interval of d_i	$U(d_i\lvert t)+l(d_i)=E(d_i\lvert t)$
$Q(t)$	set of pending requests at t	
Q_m	the mth request	$Q_m\in Q(t)$
q_m^n	the nth data item requested by Q_m	$q_m^n\in D$
$R(Q_m)$	requested data set of Q_m	$R(Q_m)=\{q_m^1,q_m^2,\cdots q_m^{\lvert R(Q_m)\rvert}\}$
$S(Q_m)$	submission time of Q_m	
$L(Q_m)$	deadline of Q_m	
τ	data transmission time	
$T_{Q_m}(q_m^n)$	time when q_m^n is disseminated	
$T_{Q_m}^f$	time when the first data item is disseminated for Q_m	$T_{Q_m}^f=\min(T_{Q_m}(q_m^n))$
$T_{Q_m}^l$	time when the last data item is disseminated for Q_m	$T_{Q_m}^l=\max(T_{Q_m}(q_m^n))$
$E_{Q_m}^e(t)$	earliest expiration time of the requested data items	$E_{Q_m}^e(t)=\min(E(q_m^n\lvert t))$
$R_{Q_m}^u(t)$	unserved set of Q_m	$R_{Q_m}^u(t)\subseteq R(Q_m)$
$\chi_{Q_m}(t)$	tentative time bound	$\chi_{Q_m}(t)=\min(L(Q_m),E_{Q_m}^e(t))$
X_{Q_m}	determined time bound	$\mathrm{X}_{Q_m}=\min(L(Q_m),E_{Q_m}^e(T_{Q_m}^f))$
$Q_s(t)$	set of schedulable requests	$Q_s(t)\subseteq Q(t)$
$P_{d_i}(t)$	effective data productivity of d_i	
$\Lambda_{Q_m}(t)$	effective request productivity	
$\Psi_{Q_m}(t)$	remaining ratio	
$\omega_{Q_m}(t)$	feasible scheduling segment	
$\Omega_{Q_m}(t)$	feasible scheduling period	the length is $\lvert\Omega_{Q_m}(t)\rvert$

III. PROBLEM FORMULATION

The set of data items in the database is denoted by $D=\{d_1, d_2,\ldots,d_{|D|}\}$, where $|D|$ is the total number of data items. Each data item d_i $(1\le i\le |D|)$ is characterized by a 3-tuple: $\langle V(d_i|t),U(d_i|t),E(d_i|t)\rangle$, where $V(d_i|t)$ is the value of d_i at time t. $U(d_i|t)$ and $E(d_i|t)$ represent the update time and the expiration time of d_i in its version at time t. The update interval of d_i is denoted by $l(d_i)$. Accordingly, we have $E(d_i|t)=U(d_i|t)+l(d_i)$. The set of pending requests in the service queue is denoted by $Q(t)$. The request Q_m $(Q_m\in Q(t))$ is characterized by a 3-tuple: $\langle R(Q_m),S(Q_m),L(Q_m)\rangle$. $R(Q_m)$ is the set of requested data items, and it is represented by $R(Q_m)=\{q_m^1,q_m^2,\ldots q_m^{|R(Q_m)|}\}$, where $|R(Q_m)|$ is the number of requested data items. q_m^n $(1\le n\le |R(Q_m)|)$ represents the nth data item requested by Q_m and $q_m^n\in D$. $S(Q_m)$ and $L(Q_m)$ represent the submission time and the deadline of Q_m, respectively. The time taken to broadcast a data item is denoted by τ, which is referred to as the *transmission time*. The primary notations are summarized in Table I.

A. Snapshot Consistency Requirement

Due to the temporality of data items, the versions of the retrieved dependent data items in a request are expected to be correlated in time. Otherwise, the query result could be meaningless. For example, when a vehicle requests the traffic conditions of two routes, it is expected that the time stamps of these two pieces of information (i.e., the update time of the two data items) are close enough for comparison. In order to guarantee certain correlations among the retrieved data items in a request, we define the snapshot consistency requirement as follows. At time t, the database maintains the latest version for each data item, which is regarded as the *snapshot* of the current database. The versions of the multiple retrieved data items in a request are demanded to be in the same snapshot with respect to a particular time instance, whereas the time instance is determined by the time when the first data item for this request is disseminated. In other words, when a request retrieves its first data item, the versions of its remaining requested data items need to be in the same snapshot with regard to the first one. The detailed conditions are stated as follows.

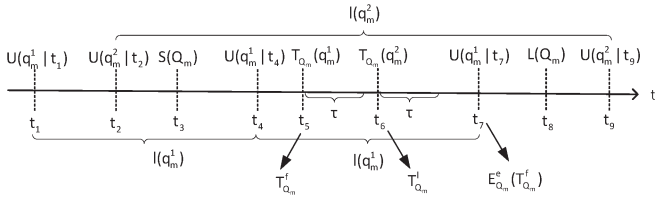


Fig. 2. Relationship of timings on satisfying a request.

- The time to disseminate each q_m^n for Q_m (i.e., $T_{Q_m}(q_m^n)$) has to be later than the submission time of Q_m (i.e., $S(Q_m)$). This is because vehicles only retrieve data items after submitting their requests. Denote $T_{Q_m}^f$ as the time when the first data item is disseminated for Q_m. The first condition is represented by
$$T_{Q_m}^f > S(Q_m) \tag{1}$$
where $T_{Q_m}^f=\min(T_{Q_m}(q_m^n)),\forall q_m^n\in R(Q_m)$.
- Each q_m^n has to be retrieved before the request deadline $L(Q_m)$. The retrieval time of q_m^n is $T_{Q_m}(q_m^n)+\tau$, where τ is the data transmission time. Denote $T_{Q_m}^l$ as the time when the last data item is disseminated for Q_m. The second condition is represented by
$$T_{Q_m}^l+\tau\le L(Q_m) \tag{2}$$
where $T_{Q_m}^l=\max(T_{Q_m}(q_m^n)),\forall q_m^n\in R(Q_m)$.
- The version of each retrieved data item has to be in the same snapshot, which is determined when the first data item is disseminated for Q_m (i.e., $T_{Q_m}^f$). Each q_m^n is associated with an expiration time at $T_{Q_m}^f$, which is $E(q_m^n|T_{Q_m}^f)$. Denote the earliest expiration time of these data items as $E_{Q_m}^e(T_{Q_m}^f)$. The third condition is represented by
$$T_{Q_m}^l+\tau\le E_{Q_m}^e(T_{Q_m}^f) \tag{3}$$
where $E_{Q_m}^e(T_{Q_m}^f)=\min(E(q_m^n|T_{Q_m}^f)),\forall q_m^n\in R(Q_m)$.

Fig. 2 illustrates the relationship of timings for serving a request. Suppose Q_m is submitted at t_3, and the requested data set $R(Q_m)=\{q_m^1,q_m^2\}$. The deadline of Q_m is t_8. q_m^1 is updated at t_1, t_4, and t_7 with the update interval of $l(q_m^1)$, whereas q_m^2 is updated at t_2 and t_9 with the update interval of $l(q_m^2)$. To serve Q_m, according to the first condition, the broadcast time of q_m^1 and q_m^2 should be later than $S(Q_m)$ (i.e., t_3). Since q_m^1 is the first data item broadcast for Q_m, we have $T_{Q_m}^f=t_5$. This satisfies the first condition as $t_5>t_3$. According to the second condition, both q_m^1 and q_m^2 have to be retrieved before $L(Q_m)$ (i.e., t_8). Since q_m^2 is the last data item broadcast for Q_m, we have $T_{Q_m}^l=t_6$. This satisfies the second

condition as $t_6 + \tau \leq t_8$. According to the third condition, the versions of q_m^1 and q_m^2 have to be consistent with regard to the snapshot at time $T_{Q_m}^f$, which is t_5. Since $E(q_m^1|t_5) = t_7$ and $E(q_m^2|t_5) = t_9$, the earliest expiration time $E_{Q_m}^e(T_{Q_m}^f) = t_7$. This satisfies the third condition as $t_6 + \tau \leq t_7$. With the satisfaction of the three conditions, Q_m can be satisfied by such a schedule.

Suppose the duration between t_3 and t_4 is less than the time of transmitting two data items (i.e., $[t_3, t_4] < 2 \cdot \tau$), then Q_m could not be satisfied if either q_m^1 or q_m^2 is disseminated during $[t_3, t_4]$. This is because if $t_3 < T_{Q_m}^f < t_4$, the expiration time of q_m^1 at this snapshot (i.e., $E(q_m^1|T_{Q_m}^f)$) would be t_4. Hence, the earliest expiration time (i.e., $E_{Q_m}^e(T_{Q_m}^f)$) is computed by $\min(t_4, t_9) = t_4$. However, it cannot satisfy $T_{Q_m}^l + \tau \leq t_4$ due to $[t_3, t_4] < 2 \cdot \tau$. Therefore, it violates the third condition and Q_m cannot be served by such a schedule. This example indicates that even for serving a single request, it is not always wise to start the service as early as possible.

B. NP-Hardness

The TDD problem is specified as follows.

Instance: There are M requests $Q_1, Q_2, \ldots, Q_M$, and $R(Q_m)$ is the set of data items requested by Q_m $(1 \leq m \leq M)$. Each data item d_i $(d_i \in D)$ has an update interval of $l(d_i)$. The submission time and the deadline of Q_m are $S(Q_m)$ and $L(Q_m)$, respectively. The data transmission time is τ.

Question: Is there a schedule that can satisfy at least C $(C \in Z^+)$ requests?

We proof TDD is NP-hard by constructing a polynomial-time reduction from a well-known NP-hard problem, namely, the Clique problem [16], to a special instance of TDD.

Proof: Given an instance of Clique $\langle G, k \rangle$ for determining whether there is a clique of size k in graph G, we consider a special instance of TDD as follows to complete the reduction. For any Q_m, let $|R(Q_m)| = 2$, $S(Q_m) = 0$ and $L(Q_m) = k$. For any d_i, let $l(d_i) = k$ and $U(d_i|0) = 0$. Let the data transmission time $\tau = 1$ and the target $C = (k \cdot (k-1))/2$. With the above settings, for graph $G = \langle V, E \rangle$, we construct a one-to-one mapping from each vertex v_i $(v_i \in V)$ to each requested data item d_i $(d_i \in D)$. In addition, for any Q_m with the requested data items d_i and d_j, we construct edge e_m between the two corresponding vertices v_i and v_j, which gives a one-to-one mapping between request Q_m and edge e_m. Consider different requests asking for different sets of data items, namely, $R(Q_i) \neq R(Q_j)$ $(1 \leq i, j \leq M$ and $i \neq j)$. According to the mapping, there is at most one edge between two vertices. An example of this reduction is illustrated in Fig. 3. Suppose $k = 4$, for the Clique problem, the question is whether there is a clique with at least the size of 4. For the TDD problem, since $C = (k \cdot (k-1))/2 = 6$, the question is whether there is a schedule that can satisfy at least six requests. In this example, the answers to both the questions are yes. Specifically, for the Clique problem, there is $V' = \{v_1, v_3, v_4, v_5\} \subset V$, which forms a clique with the size of 4. For the TDD problem, there is a schedule of disseminating d_1, d_3, d_4, and d_5, which can satisfy six requests (i.e., Q_3, Q_4, Q_5, Q_6, Q_7, and Q_8).

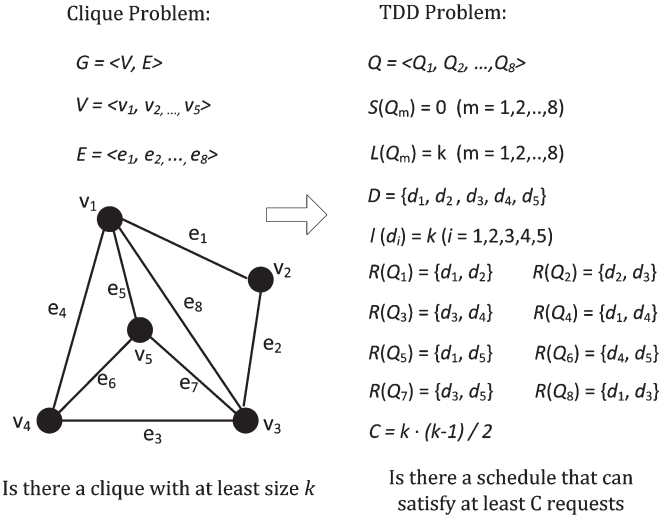


Fig. 3. Example of reduction from the Clique problem to the TDD problem.

In the following, we prove that with the above polynomial-time reduction, graph G has a clique with at least size k if and only if there is a schedule that can satisfy at least C requests, where $C = (k \cdot (k-1))/2$.

Suppose G has a clique $V' \subseteq V$ and $|V'| = k$. According to the mapping, there are k corresponding data items, which are requested by $(k \cdot (k-1))/2$ requests. Each request asks for two out of these k data items. Since $\tau = 1$ and $L(Q_m) = k$ for any Q_m, when k data items are disseminated, none of the requests will miss its deadline. In addition, the expiration time of each d_i is computed by $E(d_i|0) = U(d_i|0) + l(d_i)$, which is equal to k according to the default setting. Hence, the earliest expiration time $E_{Q_m}^e(0) = k$. Therefore, disseminating k data items will not violate the snapshot consistency requirement. To sum up, all the $(k \cdot (k-1))/2$ requests can be satisfied. Hence, given a clique of size k, there is a schedule that can satisfy at least $(k \cdot (k-1))/2$ requests.

Conversely, suppose there is a schedule that can satisfy at least $(k \cdot (k-1))/2$ requests. According to the mapping, there is a subgraph $G' = \langle V', E' \rangle$, which consists of $(k \cdot (k-1))/2$ edges. Since there is at most one edge between two vertices, given k vertices, the number of edges is bounded by $C_k^2 = (k \cdot (k-1))/2$. Therefore, at least k vertices are required, which implies $|V'| \geq k$. In addition, since $L(Q_m) = k$ and $l(d_i) = k$, there are at most k time units (i.e., k data items can be disseminated) to serve these requests. Therefore, G' contains at most k vertices, which implies $|V'| \leq k$. To sum up, we have $|V'| = k$ and G' is a complete subgraph of G. Hence, given a schedule that can satisfy at least $(k \cdot (k-1))/2$ requests, there is a clique of size k.

The above proves the NP-hardness of a special case of the TDD problem. Accordingly, TDD is NP-hard. □

IV. Algorithm Design

A. Time Bound Analysis

Due to the broadcast effect, some requests may be partially served before being scheduled. Specifically, these requests may

have retrieved part of their data items, which are broadcast to serve other scheduled requests. In view of this, we define the *unserved set of a request* as follows.

Definition 1—Unserved set of a request: At time t, the set of unserved data items of request Q_m is represented by $R^u_{Q_m}(t) = \{q_m^1{}', q_m^2{}', \ldots q_m^{|R^u_{Q_m}(t)|'}\}$, where $|R^u_{Q_m}(t)|$ is the number of unserved data items ($0 < |R^u_{Q_m}(t)| \leq |R(Q_m)|$) and $R^u_{Q_m}(t) \subseteq R(Q_m)$.

Given Q_m, if none of its requested data items have been retrieved (i.e., $R^u_{Q_m}(t) = R(Q_m)$), then Q_m is called the *unserved request*. In contrast, if part of the requested data items have been retrieved (i.e., $R^u_{Q_m}(t) \subset R(Q_m)$ and $R^u_{Q_m}(t) \neq \emptyset$), then Q_m is called the *partially served request*. Recall that each request is associated with a deadline. Meanwhile, each data item is associated with an expiration time. Therefore, there is a practical time bound to serve a request when considering both the request deadline and the data expiration time. Indeed, for unserved requests and partially served requests, they have different attributes in terms of the time bound. We analyze this issue by introducing the *tentative time bound* and the *determined time bound* for unserved requests and partially served requests, respectively.

Definition 2—Tentative time bound: At time t, if Q_m is an unserved request, then the tentative time bound for Q_m ($\chi_{Q_m}(t)$) is either its request deadline ($L(Q_m)$), or the earliest expiration time of its requested data items at time t ($E^e_{Q_m}(t)$), whichever is earlier. That is

$$\chi_{Q_m}(t) = \min\left(L(Q_m), E^e_{Q_m}(t)\right). \tag{4}$$

Note that $\chi_{Q_m}(t)$ is not the finalized time bound for Q_m and the value of $\chi_{Q_m}(t)$ may change with time. This is because although the request deadline $L(Q_m)$ has been determined since the submission of Q_m, the value of $E^e_{Q_m}(t)$ may vary with time. Therefore, at different scheduling points, the dynamic value of $E^e_{Q_m}(t)$ may result in different values of $\chi_{Q_m}(t)$. Consider the example shown in Fig. 2 and suppose $t_3 < t < t_4$. We have $E^e_{Q_m}(t) = \min(E(q_m^1|t), E(q_m^2)|t) = t_4$. Accordingly, $\chi_{Q_m}(t) = \min(L(Q_m), t_4)$, which is equal to t_4. In contrast, if $t_4 < t < t_7$, then we have $E^e_{Q_m}(t) = \min(E(q_m^1|t), E(q_m^2|t)) = t_7$, and hence, the value of $\chi_{Q_m}(t)$ changes to $\min(L(Q_m), t_7) = t_7$.

Note that the tentative time bound is only the attribute of unserved requests. As soon as the service starts for a request, due to the snapshot consistency requirement, the versions of all its data items are determined. This gives a *determined time bound* for the partially served request.

Definition 3—Determined time bound: If Q_m is a partially served request, then the determined time bound for Q_m (X_{Q_m}) is either its request deadline $L(Q_m)$, or the earliest expiration time of its requested data items at time $T^f_{Q_m}$ (i.e., $E^e_{Q_m}(T^f_{Q_m})$), whichever is earlier. That is

$$X_{Q_m} = \min\left(L(Q_m), E^e_{Q_m}(T^f_{Q_m})\right) \tag{5}$$

where $T^f_{Q_m}$ is the time when the first data item is disseminated for Q_m.

Since both $L(Q_m)$ and $E^e_{Q_m}(T^f_{Q_m})$ have been fixed at time $T^f_{Q_m}$, X_{Q_m} is a static value. With the above time bound analysis, we can find those requests that have chance to be satisfied at a specific time t, which are called the *schedulable request*. At time t, denote $Q_s(t)$ as the set of schedulable requests. Based on the service status of requests, $Q_s(t)$ can be divided into two subsets (i.e., $Q_{s1}(t)$ and $Q_{s2}(t)$), which are obtained as follows.

a) For an unserved Q_m, it is schedulable if Q_m can retrieve all of its data items before the tentative time bound $\chi_{Q_m}(t)$. The subset is represented by

$$Q_{s1}(t) = \{Q_m | R^u_{Q_m}(t) = R(Q_m) \wedge t + |R(Q_m)| \cdot \tau \leq \chi_{Q_m}(t)\}. \tag{6}$$

b) For a partially served Q_m, it is schedulable if Q_m can retrieve its remaining data items before the determined time bound X_{Q_m}. The subset is represented by

$$Q_{s2}(t) = \left\{Q_m | R^u_{Q_m}(t) \subset R(Q_m) \wedge R^u_{Q_m}(t) \neq \emptyset \wedge t + |R^u_{Q_m}(t)| \cdot \tau \leq X_{Q_m}\right\}. \tag{7}$$

To sum up, the set of schedulable requests is obtained by

$$Q_s(t) = Q_{s1}(t) \cup Q_{s2}(t). \tag{8}$$

B. PSU Algorithm

As the broadcast is an intrinsic nature of wireless communication in VCPS, the scheduling is expected to exploit the broadcast effect for data dissemination and enhance the system scalability. In addition, as a request may correspond to multiple dependent data items for completing the service, it is critical to consider the bandwidth efficiency in terms of satisfying multi-item requests. Finally, due to the time constraint of services and the temporality of data items, it is expected to serve as many vehicles as possible before their respective service time bounds. With the above motivation, we propose a heuristic algorithm called PSU, which considers request characteristics of *productivity*, *status*, and *urgency* in scheduling.

1) Objectives: The primary objectives of PSU include exploiting the broadcast effect, improving the bandwidth efficiency, and enhancing the request service.

- Exploiting the broadcast effect: In roadside-to-vehicle communication, a data item broadcast from the RSU can be retrieved by all the vehicles within its coverage, which is called the *broadcast effect*. Clearly, it is expected to exploit such a benefit to enhance the system scalability. Conventionally, scheduling a data item with the largest number of pending requests would be the best option to maximize the broadcast benefit. However, as analyzed in Section IV-A, not all the requests are schedulable at a scheduling point due to the temporality of data items. Therefore, a more sophisticated approach is expected to effectively exploit the broadcast effect.
- Improving the bandwidth efficiency: Due to the application requirement in VCPS, a request may ask for multiple dependent data items for query processing. In this case, it is a waste of bandwidth if a request retrieved part of its data items but could not be completely served before its time bound. Previous studies [17], [18] have revealed that there

is a tradeoff between maximizing the broadcast effect and improving the bandwidth efficiency for serving multi-item requests. Specifically, it will cause the starvation problem if an algorithm simply schedules data items with a large number of pending requests but ignores the fact that a multi-item request has to retrieve all of its data items before the time bound. To address the starvation problem and avoid the waste of bandwidth, the algorithm should consider the service status of each multi-item request in scheduling.

- Enhancing the request service: Due to the time constraint of services and the temporality of data items, there is a stringent time bound for serving requests. In order to enhance the service chance of requests, conventionally, the algorithms consider either the deadline (e.g., EDF [12]) or the slack time (e.g., slack time inverse number of pending requests (SIN) [6]) in scheduling to capture the attribute of request urgency. Nevertheless, these solutions cannot be directly applied into this new setting, because as analyzed in Definition 2, the time bound cannot be determined for unserved requests (i.e., the tentative time bound). Therefore, it is desirable to design new metrics to reflect the request urgency and enhance the request service.

2) Metrics: For exploiting the broadcast effect, the *effective data productivity* and the *effective request productivity* are defined as follows.

Definition 4—Effective data productivity: At time t, denote $Q_{d_i}(t)$ as the set of requests that satisfies the following two conditions: 1) d_i is in the unserved set of Q_m, namely, $d_i \in R^u_{Q_m}(t)$; and 2) Q_m is schedulable at time t, namely, $Q_m \in Q_s(t)$. The effective data productivity of d_i $(1 \le i \le |D|)$, which is denoted by $P_{d_i}(t)$, is defined as the number of requests in $Q_{d_i}(t)$. It is represented by

$$P_{d_i}(t) = |Q_{d_i}(t)|. \tag{9}$$

It has been demonstrated that it is effective in serving multi-item requests by broadcasting all the unserved data items of a request successively [18], namely, scheduling at the request level. In this regard, the *effective request productivity* is defined as follows.

Definition 5—Effective request productivity: At time t, the effective request productivity of Q_m, which is denoted by $\Lambda_{Q_m}(t)$, is the average of the effective data productivity of its unserved data items, which is computed by

$$\Lambda_{Q_m}(t) = \left(\sum_{d_i \in R^u_{Q_m}(t)} P_{d_i}(t) \right) / \left| R^u_{Q_m}(t) \right|. \tag{10}$$

Selecting requests with higher values of $\Lambda_{Q_m}(t)$ can better exploit the broadcast effect.

For improving the bandwidth efficiency, the *remaining ratio* is defined as follows to capture the service status of requests.

Definition 6—Remaining ratio: At time t, the remaining ratio of Q_m, which is denoted by $\Psi_{Q_m}(t)$, is the number of its unserved data items over the total number of its requested data items, which is computed by

$$\Psi_{Q_m}(t) = \frac{\left| R^u_{Q_m}(t) \right|}{|R(Q_m)|}. \tag{11}$$

A small value of the remaining ratio implies that a large percentage of the requested data items have been retrieved and the request is close to be satisfied. Giving higher priority to a request with a smaller value of the remaining ratio will help to improve the bandwidth efficiency. We justify this claim by the following two cases. Case 1: Consider two requests Q_m and Q_n, which ask for the same number of data items ($|R(Q_m)| = |R(Q_n)|$). Suppose $\Psi_{Q_m}(t) < \Psi_{Q_n}(t)$, which implies that Q_m has fewer unserved data items than Q_n (i.e., $|R^u_{Q_m}(t)| < |R^u_{Q_n}(t)|$). Hence, it requires less bandwidth to complete the service of Q_m. Case 2: Consider two requests Q_m and Q_n, which have the same number of unserved data items ($|R^u_{Q_m}(t)| = |R^u_{Q_n}(t)|$). Suppose $\Psi_{Q_m}(t) < \Psi_{Q_n}(t)$, which implies that Q_m asks for more data items than Q_n. Accordingly, more bandwidth has been consumed to serve Q_m (i.e., $|R(Q_m)| - |R^u_{Q_m}(t)| > |R(Q_n)| - |R^u_{Q_n}(t)|$). Hence, completing the service of Q_m will make the previously consumed bandwidth for Q_m count, which means more cost effective.

For enhancing the request service, we define the metric called *feasible scheduling period* to capture the request urgency. First, we introduce the concept of *feasible and nonfeasible scheduling segments*. At time t, depending on the service status of Q_m, there are two cases for feasible/nonfeasible scheduling segments.

a) Q_m is an unserved request: if Q_m is schedulable (i.e., $t + |R(Q_m)| \cdot \tau \le \chi_{Q_m}(t)$), then $[t, \chi_{Q_m}(t)]$ is a *feasible scheduling segment*. Otherwise, $[t, \chi_{Q_m}(t)]$ is a *nonfeasible scheduling segment*.

b) Q_m is a partially served request: if Q_m is schedulable (i.e., $t + |R^u_{Q_m}(t)| \cdot \tau \le X_{Q_m}$), then $[t, X_{Q_m}]$ is a *feasible scheduling segment*. Otherwise, $[t, X_{Q_m}]$ is a *nonfeasible scheduling segment*.

Note that in Case b), $[t, X_{Q_m}]$ is the only possible feasible/nonfeasible scheduling segment. However, in Case a), it may have multiple feasible/nonfeasible scheduling segments. Recall the example shown in Fig. 2. The tentative time bound $\chi_{Q_m}(t) = t_4$ if $t_3 < t < t_4$. At this time, $[t, \chi_{Q_m}(t)]$ is a nonfeasible scheduling segment because $[t, t_4] < 2 \cdot \tau$. Since $t_4 < L(Q_m)$, the algorithm proceeds to examine other segments by setting t' to t_4. At t', the new tentative time bound $\chi_{Q_m}(t') = t_7$. Clearly, $[t', \chi_{Q_m}(t')]$ is a feasible scheduling segment. Finally, when setting t'' to t_7, the current tentative time bound $\chi_{Q_m}(t'') = t_8$, and $[t'', \chi_{Q_m}(t'')]$ is a nonfeasible scheduling segment (suppose $[t_7, t_8] < 2 \cdot \tau$). As t_8 is equal to the request deadline $L(Q_m)$, which is the latest time for Q_m to be served, it cannot move forward to check other segments. With the above knowledge, we define the *feasible scheduling period* of Q_m as follows.

Definition 7—Feasible scheduling period: At time t, the feasible scheduling period for Q_m, which is denoted by $\Omega_{Q_m}(t)$, is the union of the feasible scheduling segments $(\omega_{Q_m}(t))$ in $[t, L(Q_m)]$. It is represented by

$$\Omega_{Q_m}(t) = \cup \omega_{Q_m}(t) \; in \; [t, L(Q_m)]. \tag{12}$$

For a partially served Q_m, the only possible feasible scheduling segment is $[t, X_{Q_m}]$. Therefore, $\Omega_{Q_m}(t)$ is either $[t, X_{Q_m}]$ or $\emptyset$, depending on whether $[t, X_{Q_m}] \subseteq \omega_{Q_m}(t)$. For an

unserved Q_m, $\Omega_{Q_m}(t)$ is obtained by combining all the feasible scheduling segments of Q_m at t. Note that these feasible scheduling segments may not be consecutive. The operation to compute $\Omega_{Q_m}(t)$ for unserved requests is described as follows.

- Set the feasible scheduling period to the null set ($\Omega_{Q_m}(t) \leftarrow \emptyset$), and set t' to the current time t ($t' \leftarrow t$).
- Check whether $[t', \chi_{Q_m}(t')]$ is a feasible scheduling segment. If it is true, add $[t', \chi_{Q_m}(t')]$ to $\Omega_{Q_m}(t)$ (i.e., $\Omega_{Q_m}(t) \leftarrow \Omega_{Q_m}(t) \cup [t', \chi_{Q_m}(t')]$).
- Check whether $\chi_{Q_m}(t')$ is equal to the request deadline $L(Q_m)$. If it is true, the operation terminates. Otherwise (i.e., $\chi_{Q_m}(t') < L(Q_m)$), set t' to the current tentative time bound (i.e., $t' \leftarrow \chi_{Q_m}(t')$).
- Repeat the update of t' until the tentative time bound is equal to the request deadline (i.e., $\chi_{Q_m}(t') = L(Q_m)$).

The length of feasible scheduling period $|\Omega_{Q_m}(t)|$ reflects the actual duration, in which a request can be served. In particular, for a schedulable request, the shorter $|\Omega_{Q_m}(t)|$ is, the more urgent the request is.

3) Scheduling policy: With the above metric design and analysis, we have the following three observations.

- To exploit the broadcast effect, the algorithm should give a higher priority to the request with a larger value of the effective request productivity.
- To improve the bandwidth efficiency, the algorithm should give a higher priority to the request with a smaller value of the remaining ratio.
- To enhance the request service, the algorithm should give a higher priority to the request with a shorter length of the feasible scheduling period.

Accordingly, the request priority is a compound effect of the effective request productivity ($\Lambda_{Q_m}(t)$), the remaining ratio ($\Psi_{Q_m}(t)$), and the length of feasible scheduling period ($|\Omega_{Q_m}(t)|$), which is defined as follows.

Definition 8—Request priority: At time t, the priority of Q_m is computed by

$$\text{Priority}_{Q_m}(t) = \frac{\Lambda_{Q_m}(t)}{\Psi_{Q_m}(t) \cdot |\Omega_{Q_m}(t)|}. \tag{13}$$

PSU consists of three steps. In Step 1, it constructs the set of schedulable requests $Q_s(t)$ by analyzing the time bound. In Step 2, it computes the priority of requests in $Q_s(t)$ and schedules the one with the highest priority. In Step 3, it broadcasts each of the unserved data item for the scheduled request and updates the status of pending requests in the service queue. The pseudocode of PSU is illustrated in Algorithm 1.

Algorithm 1 PSU

Step 1: Construct the set of schedulable requests $Q_s(t)$
1 : $Q_s(t) \leftarrow \emptyset$;
2 : **for** each unserved request $Q_m \in Q(t)$ **do**
3 : **if** $t + |R(Q_m)| \cdot \tau \leq \chi_{Q_m}(t)$ **then**
4 : $Q_s(t) \leftarrow Q_s(t) \cup \{Q_m\}$;
5 : **end if**
6 : **end for**
7 : **for** each partially served request $Q_m \in Q(t)$ **do**
8 : **if** $t + |R^u_{Q_m}(t)| \cdot \tau \leq X_{Q_m}$ **then**
9 : $Q_s(t) \leftarrow Q_s(t) \cup \{Q_m\}$;
10: **end if**
11: **end for**
Step 2: Schedule the request with the highest priority
12: $maxPriority \leftarrow 0$;
13: **for** each $Q_m \in Q_s(t)$ **do**
14: $Priority_{Q_m}(t) = (\Lambda_{Q_m}(t))/(\Psi_{Q_m}(t) \cdot |\Omega_{Q_m}(t)|)$;
15: **if** $maxPriority < Priority_{Q_m}(t)$ **then**
16: $maxPriority \leftarrow Priority_{Q_m}(t)$;
17: $Q_{\text{selected}} \leftarrow Q_m$;
18: **end if**
19: **end for**
Step 3: Broadcast data items for the selected request and update the service queue
20: broadcast each $d_i \in R^u_{Q_{\text{selected}}}(t)$;
21: **for** each $Q_n \in Q_s(t)$ **do**
22: **if** $d_i \in R^u_{Q_n}$ **then**
23: $R^u_{Q_n}(t) \leftarrow R^u_{Q_n}(t) - \{d_i\}$;
24: **if** d_i is the first retrieved data item of Q_n **then**
25: Compute the determined time bound X_{Q_n};
26: **end if**
27: **if** $R^u_{Q_n}(t) == \emptyset$ **then**
28: $Q(t) \leftarrow Q(t) - \{Q_n\}$;
29: **end if**
30: **end if**
31: **end for**
32: **for** each $Q_n \in Q(t)$ **do**
33: **if** $t > L(Q_n)$ **then**
34: $Q(t) \leftarrow Q(t) - \{Q_n\}$;
35: **end if**
36: **end for**

4) Computation Complexity: It is common to evaluate the computation complexity of the algorithm by measuring the number of data items to be examined in each scheduling point [5], [6]. Suppose there are n requests in the service queue and each request asks for s data items. The computation complexity of PSU is analyzed as follows. First, in order to construct the set of schedulable requests $Q_s(t)$, PSU traverses the service queue and computes the time bound of each request. For this operation, at most s data items will be examined for a request. Accordingly, the total number of data items to be examined is bounded by $s \cdot n$. Second, PSU computes the priority of each request in $Q_s(t)$ and selects the one with the highest priority. As $|Q_s(t)| \leq n$, likewise, there are at most $s \cdot n$ data items need to be examined. Once the request is selected, its unserved data items are scheduled to broadcast successively, where at most s data items are examined, and the maintenance of the service queue examines at most n requests for updating. To sum up, the computation complexity of PSU is $O(s \cdot n)$. Note that existing representative scheduling algorithms have the computation complexity of $O(n)$ [19]. In practice, given a specific application, s is normally a fixed number (or within a fixed range), and it is bounded by the size of the database ($|D|$); whereas n is a variable, which increases with the system

TABLE II
DEFAULT SETTING

Parameter	Default	Description
$\|D\|$	100	number of temporal data items
$\frac{1}{\lambda}$	0.6	mean inter-arrival time of requests
s	3	request size
L_{min}	50	minimum tolerated latency
L_{max}	70	maximum tolerated latency
T_{min}	200	minimum update period
T_{max}	300	maximum update period
θ	0.6	Zipf distribution parameter

scale. Therefore, variable n is the most critical factor to be considered for system scalability. Moreover, unlike previous algorithms [6], [12], which have to make a scheduling decision in every broadcast tick, PSU schedules at the request level (i.e., makes a scheduling decision in every several broadcast ticks, depending on the number of unserved data items in the selected request). Therefore, the scheduling frequency of PSU is much lower. Overall, the scheduling overhead of PSU is reasonable, and it will not be a hurdle of the system scalability.

V. PERFORMANCE EVALUATION

A. Preliminaries

The simulation model is implemented by CSIM19 [20], which captures the data dissemination characteristics as described in Section II. The interarrival time of requests follows the exponential distribution with mean value of $1/\lambda$. Each request may ask for multiple dependent data items and the request size (s) is the number of required data items. Each request is associated with a deadline, which is obtained by $t\_arrival + t\_relative$, where $t\_arrival$ is the submission time of the request, and $t\_relative$ is the relative deadline of the request. For general purposes, the value of $t\_relative$ is uniformly selected from the range $(L_{\min}, L_{\max})$, where $L_{\min}$ and $L_{\max}$ represent the minimum and the maximum tolerated latency, respectively, for serving requests. Database D consists of $|D|$ data items. Each d_i $(d_i \in D)$ has an update interval of $l(d_i)$, which is uniformly generated from the range $(T_{\min}, T_{\max})$, where $T_{\min}$ and $T_{\max}$ represent the minimum and the maximum update period of data items, respectively. The data access pattern follows the commonly used Zipf distribution [21] with a skewness parameter θ. The time unit (i.e., a broadcast tick) refers to the data transmission time. The main parameters and the corresponding descriptions are summarized in Table II. Unless stated otherwise, the simulations are conducted under the default setting.

For performance comparison, we implement two well-known real-time scheduling algorithms. One is EDF [12], and the other is SIN [6]. The statistics including the total number of submitted requests $(N_{\rm sub})$, the number of satisfied requests $(N_{\rm sat})$, and the number of failed requests $(N_{\rm fail})$ are captured for performance analysis, where $N_{\rm sat} + N_{\rm fail} = N_{\rm sub}$. Moreover, we classify the failed requests into two parts. One part is caused by missing deadlines (the number is denoted by $N_{\rm miss}$), and the other part is caused by data expiration (the number is denoted by $N_{\rm exp}$). Accordingly, we have $N_{\rm miss} + N_{\rm exp} = N_{\rm fail}$. With the collected statistics, the following criteria are adopted for performance evaluation.

- Service ratio: The ratio of the number of satisfied requests to the total number of submitted requests, which is computed by $N_{\rm sat}/N_{\rm sub}$. The primary objective of a scheduling algorithm is to maximize the service ratio.
- Bandwidth efficiency ratio: In order to measure the bandwidth efficiency quantitatively, the bandwidth efficiency ratio is computed by $(N_{\rm rev} - N_{\rm bst})/N_{\rm req}$, where $N_{\rm rev}$ is the total number of data items received by satisfied requests. $N_{\rm bst}$ is the number of broadcast data items, and $N_{\rm req}$ is the total number of data items required by all the requests. We give an example to explain this criterion. Assume there are four pending requests $Q_1 = \{a, b\}$, $Q_2 = \{a, b\}$, $Q_3 = \{a, c\}$, and $Q_4 = \{c, d\}$, and all of them have to be served in two broadcast ticks. Case a): if a and b are scheduled to broadcast, then Q_1 and Q_2 would be satisfied, whereas Q_3 and Q_4 would be failed. In this case, $N_{bst} = 2$, $N_{\rm req} = 8$, and $N_{\rm rev} = 4$. Therefore, the bandwidth efficiency ratio is $(4-2)/8 = 1/4$. A positive value means that this schedule has a positive contribution to the bandwidth efficiency, because each broadcast tick contributes to multiple satisfied requests. Case b): if a and c are scheduled to broadcast, then only Q_3 could be satisfied. In this case, the values of $N_{\rm bst}$ and $N_{\rm req}$ remain the same, but the value of $N_{\rm rev}$ is reduced to 2. Therefore, the bandwidth efficiency ratio is $(2-2)/8 = 0$. A zero value means that this schedule has no contribution to the bandwidth efficiency, because each broadcast tick contributes to exactly one satisfied request. Case c): if a and d are scheduled to broadcast, then none of the requests could be satisfied. In this case, the value of $N_{\rm rev}$ becomes 0, and the bandwidth efficiency ratio is $(0-2)/8 = -1/4$. A negative value means that this schedule has a negative contribution to the bandwidth efficiency, because each broadcast tick contributes to less than one satisfied request. To sum up, this criterion reflects the efficiency of allocating bandwidth for satisfying requests.
- Proportion of missing deadline to data expiration: Since $N_{\rm fail} = N_{\rm miss} + N_{\rm exp}$, the percentage of failed requests due to missing deadline $(P_{\rm miss})$ is computed by $P_{\rm miss} = N_{\rm miss}/N_{\rm fail}$, whereas the percentage of failed requests due to data expiration $(P_{\rm exp})$ is computed by $P_{\rm exp} = N_{\rm exp}/N_{\rm fail}$, where $P_{\rm miss} + P_{\rm exp} = 1$. This criterion examines the proportion between $P_{\rm miss}$ and $P_{\rm exp}$, which indicates the influence of these two reasons to the overall performance.
- Data expiration ratio: It is the ratio of the number of failed requests due to data expiration $(N_{\rm exp})$ to the total number of submitted requests $(N_{\rm sub})$, which is computed by $N_{\rm exp}/N_{\rm sub}$. Different from the proportion of the missing deadline to the data expiration, which focuses on examining the weights of the two reasons causing service failure, this criterion concerns the impact of data expiration on the overall scheduling performance. It evaluates the algorithm performance in terms of satisfying the snapshot consistency requirement.

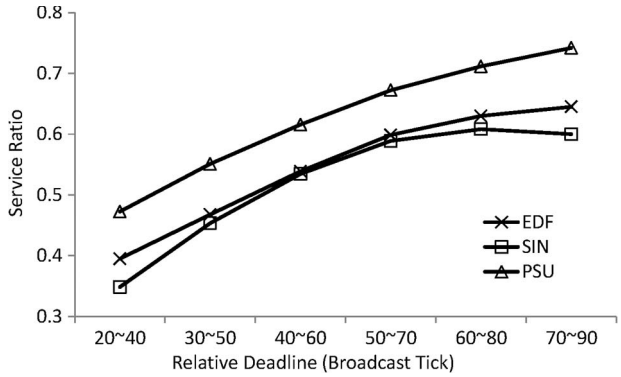


Fig. 4. Service ratio under different relative deadlines.

B. Experimental Results and Analysis

To emphasize the general applicability of the performance analysis, we do not specify the absolute values of the data size and the bandwidth, but rather use the broadcast tick as unit for comparing relative performance of different algorithms.

1) Effect of Relative Deadline: Fig. 4 shows the service ratio of algorithms under different relative deadlines. PSU achieves the highest service ratio across the whole range. In addition, note that, although the relative deadline is getting looser, none of the algorithms can achieve a 100% service ratio. The statistics shown in Fig. 5 explains the reason. When the relative deadline is getting looser, although the number of failed requests due to missing deadline is decreased, more requests are failed because of data expiration. It demonstrates that the snapshot consistency requirement has higher impact to the algorithm performance in a looser relative deadline environment. The data expiration ratio of algorithms under different relative deadlines is shown in Fig. 6. PSU performs the best in terms of satisfying the snapshot consistency requirement, particularly in a looser relative deadline environment. Fig. 7 examines the bandwidth efficiency of algorithms under different relative deadlines, which shows that PSU significantly outperforms other algorithms in terms of improving the bandwidth efficiency.

2) Effect of Data Update: Fig. 8 shows the service ratio of algorithms under different data update periods. The short update period implies a harsh condition on satisfying the snapshot consistency requirement. PSU always outperforms EDF and SIN, particularly in a short update period. This is because PSU prevents from selecting those requests, which may retrieve inconsistent versions of data items by identifying schedulable requests. The proportion of missing deadline to data expiration is shown in Fig. 9. As expected, the data expiration dominates the performance of all the algorithms when the data update period is short. In particular, for EDF and SIN, when the update period ranges between 50 and 150 broadcast ticks, almost 100% of request failures is attributed to data expiration. Fig. 10 shows the data expiration ratio of algorithms under different update periods. It further confirms that PSU has the best performance in terms of satisfying the snapshot consistency requirement, particularly in a highly dynamic vehicular environment, where the data items are updated very frequently. Fig. 11 shows the bandwidth efficiency ratio of algorithms under different data update periods. It demonstrates that no matter whether the

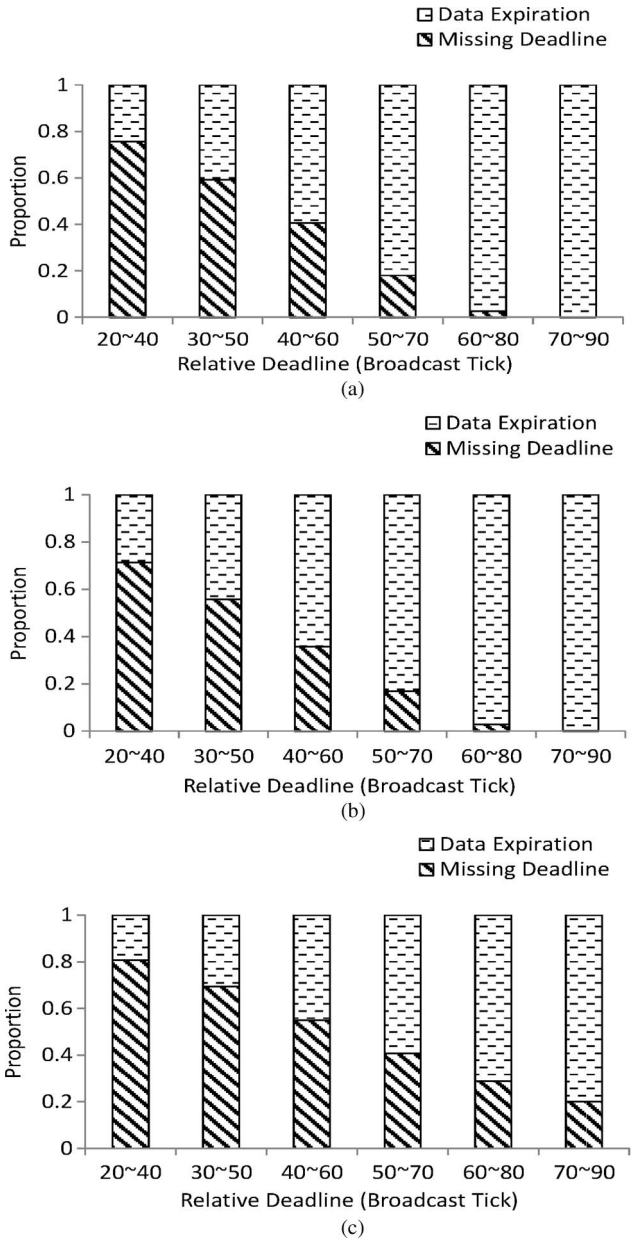


Fig. 5. Proportion of missing deadline to data expiration under different relative deadlines. (a) EDF. (b) SIN. (c) PSU.

missing deadline or the data expiration is the dominate factor, PSU achieves the highest bandwidth efficiency.

3) Effect of Request Size: For a comparison purpose, we maintain a constant system workload across different request sizes by adjusting the request arrival rate. For example, when the request size is 1, the mean request arrival rate (λ) is adjusted to 5 (requests/tick) to maintain the same system workload with the default setting. Fig. 12 shows the service ratio of algorithms under different request sizes. When the request size is equal to 1, SIN has higher service ratio than EDF, and it achieves similar performance with PSU. However, when the request size increases, the performance of SIN drastically drops. These results are consistent with the observations in previous studies [6], [18]. Note that, although the system workload remains, the performance of all the algorithms drops to different extent when the request size increases. This is because the more data

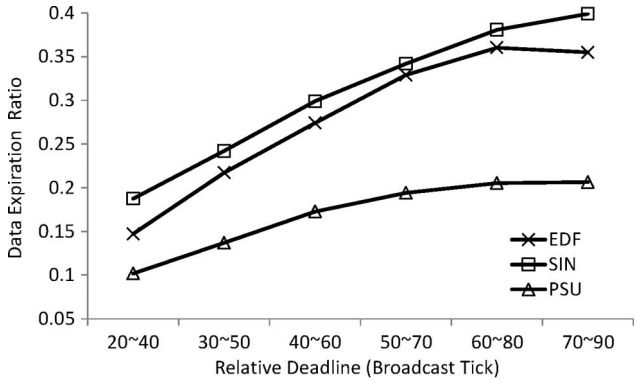


Fig. 6. Data expiration ratio under different relative deadlines.

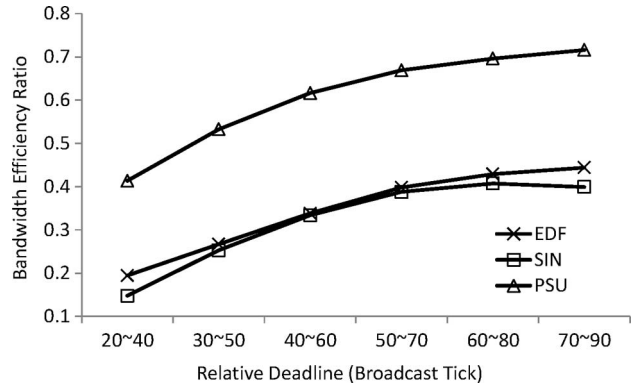


Fig. 7. Bandwidth efficiency ratio under different relative deadlines.

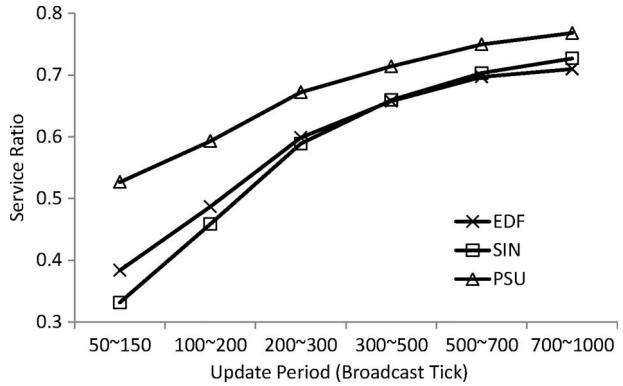


Fig. 8. Service ratio under different data update periods.

items are in a request, the more difficult to satisfy the snapshot consistency requirement. In other words, it is likely that more requests cannot be served due to data expiration. The result shown in Fig. 13 verifies this claim. We note that when the request size increases to 6, for EDF and SIN, around 60% and 70% of requests are failed, respectively, due to data expiration. Nevertheless, PSU maintains a decent data expiration ratio around 40%.

C. Performance Validation

In order to validate the effectiveness of PSU in vehicular environments, we examine the algorithm performance by modeling the following application scenario. The RSU is installed along the highway and provides location-dependent services

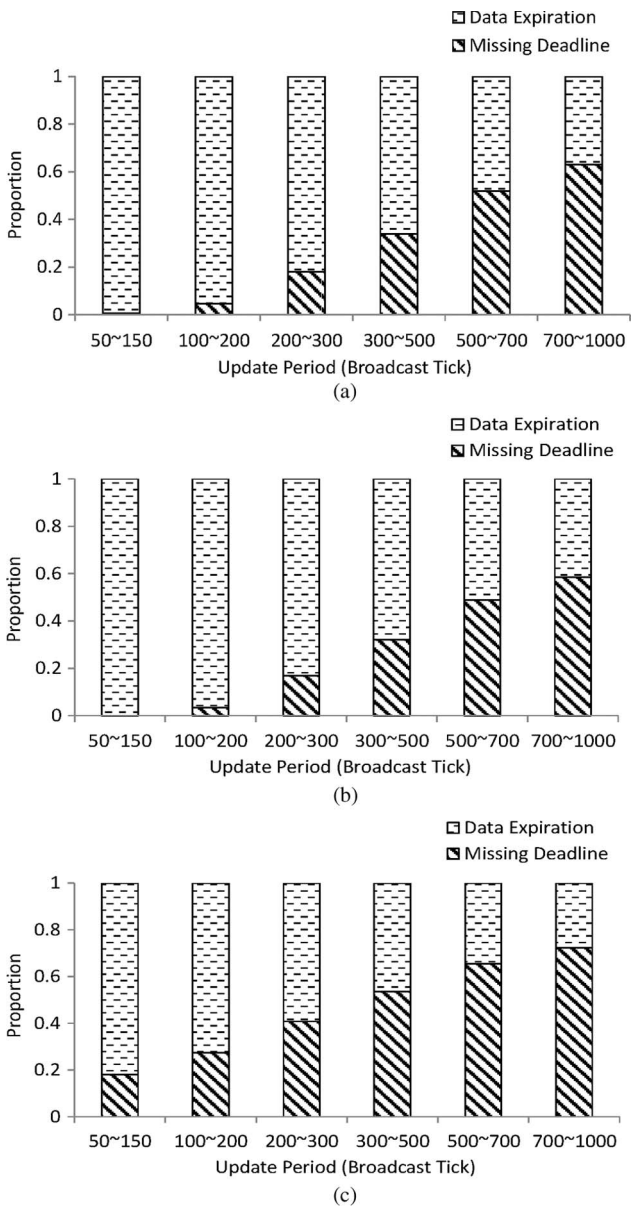


Fig. 9. Proportion of missing deadline to data expiration under different data update periods. (a) EDF. (b) SIN. (c) PSU.

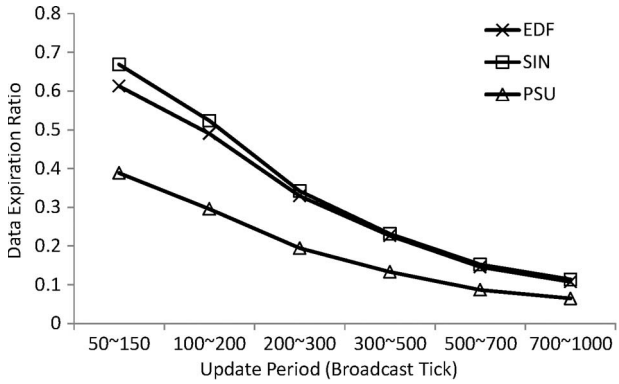


Fig. 10. Data expiration ratio under different data update periods.

such as traffic conditions to passing vehicles. The data service has to be completed before vehicles leave the coverage of the RSU.

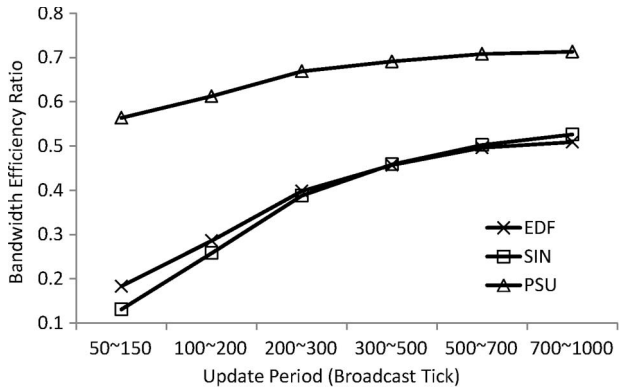


Fig. 11. Bandwidth efficiency ratio under different data update periods.

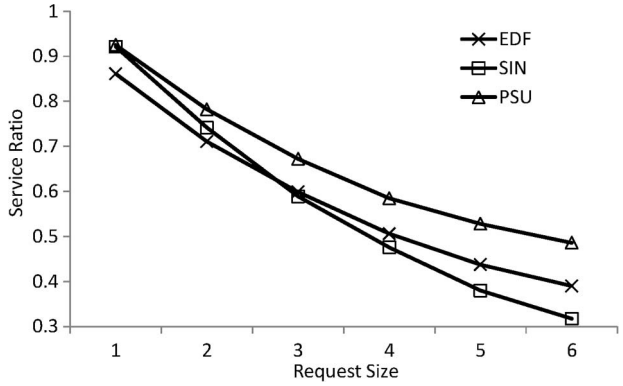


Fig. 12. Service ratio under different request sizes.

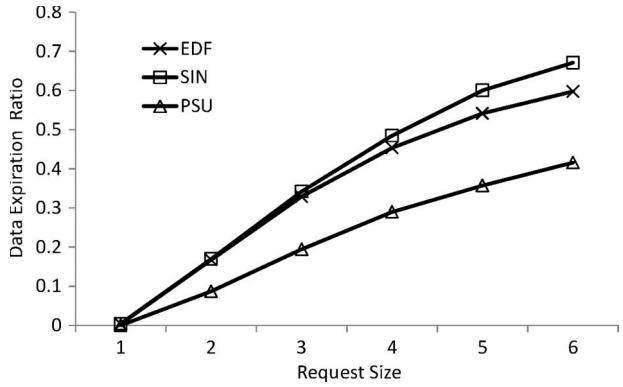


Fig. 13. Data expiration ratio under different request sizes.

Both the traffic and the communication characteristics are considered to simulate a proper vehicular environment. In particular, the traffic characteristic is simulated according to the Greenshield's model [22], which is widely adopted in macroscopic traffic models [23]. Specifically, the relationship between the vehicle speed (v) and the density (k) is represented by $v = v^f - (v^f/k^j) \cdot k$, where v^f is the free-flow speed (i.e., the maximum driving speed), and k^j is the jam density (i.e., the density that causes traffic jam). In this model, we set $v^f =$ 100 km/h and $k^j = 100$ vehicles/km, which are reasonable values in realistic highway environments. We simulate a four-lane highway, and the arrival of vehicles in each lane follows the Poisson process with the mean arrival rates (λ) of 1, 1/2, 1/3, and 1/4 (vehicles/s), respectively. The communication characteristic is simulated based on dedicated short-range communications (DSRC) [24]. In particular, the data transmission rate is set to 6 Mb/s, and the radius of the RSU coverage is set to 300 m. It has been shown that the above settings can provide reliable data dissemination via DSRC [25]. The data size is set to 1 Kb, which is sufficient for normal location-dependent information. The data update period is uniformly distributed from 200 to 300 s. The request size is 3, and the request deadline is bounded by the dwell time of vehicles in the RSU's coverage.

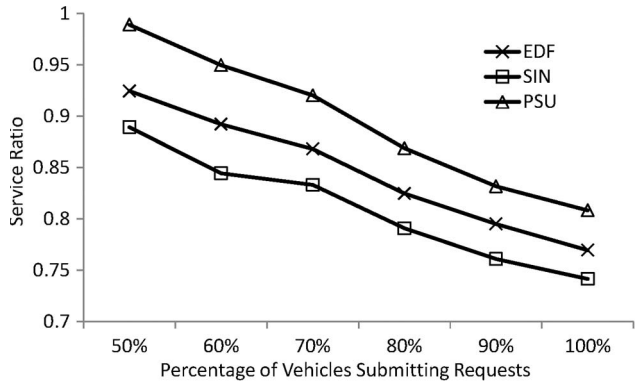


Fig. 14. Service ratio under different percentages of vehicles submitting requests.

Due to the factors such as the market penetration of on-board units mounted on vehicles and the options of drivers, it is not necessary that all the vehicles will submit requests. Fig. 14 shows the algorithm performance with different percentages of vehicles submitting requests. As shown, PSU achieves nearly a 100% service ratio when half of the vehicles submit requests. Moreover, even in a saturated scenario where all the vehicles submit requests, over 80% of requests can be satisfied by PSU. This result demonstrates the scalability of PSU in realistic vehicular environments.

VI. Related Work

In vehicular networks, current studies on data dissemination largely focused on the design of protocols to improve communication quality and reliability. Maeshima *et al.* [26] designed a MAC protocol in supporting emergency message delivery. Whenever an emergency notification occurred, the transmission of general information would be suspended to ensure timely delivery of emergency messages. Jhang and Liao [27] proposed a proxy-based communication protocol for data uploading from vehicles to the RSU. It relieves the uplink channel contention and improves the system throughput by electing proxy vehicles. Nonproxy vehicles, which attempt to communicate with the RSU, must forward their data items to a proxy vehicle. Mak *et al.* [28] proposed a coordinated MAC mode. It improves the performance for both safety and nonsafety applications by adopting a multichannel coordination mechanism, which minimizes the collision between vehicle-to-vehicle and roadside-to-vehicle communications. In order to support high reliability and low delay for data dissemination in vehicular networks, Farnoud and Valaee [29] proposed a topology-transparent broadcast protocol, which enhances the system performance on providing safety-critical services.

These studies focused on vehicular communication issues at MAC and PHY layers, whereas none of them considered the time constraint of services and the freshness of data items at the application level.

In the real-time database community, extensive research has been devoted to maintaining and processing time-variant information, where the values of data items are valid only for a certain time interval [30]. Many approaches have been proposed for managing temporal data items to track the dynamics of the real world [31]–[34]. These studies focused on striking a balance between the quality of service and the quality of data to improve overall system performance. However, none of them are designed to address the scheduling problem of serving real-time requests for temporal data items.

Scheduling algorithms have been extensively examined in the network community. In non-real-time systems, there are a number of classical scheduling algorithms. Most requested first (MRF) [35] broadcasts the data item that has the largest number of pending requests to account for the productivity of broadcast. $R \times W$ (number of pending requests multiply waiting time) [5] calculates the number of pending requests for a data item multiplied by the amount of time that the oldest outstanding request for that data item has been waiting, and it schedules the request with the maximum $R \times W$ value. In real-time systems, EDF [12] is one of the foremost classical scheduling algorithms, which broadcasts the data item with the shortest remaining lifetime to cater for the urgency of requests. SIN [6] is another representative real-time scheduling algorithm, which combines the advantages of both EDF and MRF. It has been demonstrated that SIN outperforms existing algorithms in serving time-critical requests in on-demand broadcast environments. Our recent study has investigated the TDD problem in vehicular networks [36]. This paper significantly extends our preliminary study on both theoretical analysis and simulation results with respect to the investigated problem.

VII. Conclusion and Future Work

Efficient data dissemination is critical to enable innovative applications in VCPS. In this paper, we have introduced the roadside-to-vehicle communication system and investigated the unique characteristics and challenges of data dissemination in such an environment. According to an intensive analysis of the requirement on serving real-time requests for temporal data items, we have presented the snapshot consistency requirement and formulated the TDD problem. We have proven that TDD is NP-hard by constructing a polynomial-time reduction from the Clique problem to TDD. Based on the analysis of the time bound of request services, we have proposed an online scheduling algorithm PSU, which aims at exploiting the broadcast effect, improving the bandwidth efficiency, and enhancing the request service. We have built the simulation model and designed a number of metrics for performance evaluation. A comprehensive simulation study is presented by comparing the performance of PSU with alternative real-time scheduling algorithms, including EDF and SIN. The simulation results validated that PSU is effective in providing real-time data services under different traffic scenarios and application requirements.

As an early stage of exploring the TDD in VCPS, this paper concentrated on a proof of concept of providing real-time data services via roadside-to-vehicle communication. In the future, more realistic approaches (e.g., considering communication influences at MAC and PHY layers such as packet drops and interferences) are expected to be developed for real-world applications. In addition, it is desirable to further enhance the system performance by incorporating intervehicle communication for data sharing among neighboring vehicles.

References

[1] A. Miloslavov and M. Veeraraghavan, "Sensor data fusion algorithms for vehicular cyber-physical systems," *IEEE Trans. Parallel Distrib. Syst.*, vol. 23, no. 9, pp. 1762–1774, Sep. 2012.

[2] S. K. Gehrig and F. J. Stein, "Collision avoidance for vehicle-following systems," *IEEE Trans. Intell. Transp. Syst.*, vol. 8, no. 2, pp. 233–244, Jun. 2007.

[3] K. Liu, E. Chan, V. Lee, K. Kapitanova, and S. H. Son, "Design and evaluation of token-based reservation for a roadway system," *Transp. Res. C, Emerging Technol.*, vol. 26, pp. 184–202, Jan. 2013.

[4] J. Lee and B. Park, "Development and evaluation of a cooperative vehicle intersection control algorithm under the connected vehicles environment," *IEEE Trans. Intell. Transp. Syst.*, vol. 13, no. 1, pp. 81–90, Mar. 2012.

[5] D. Aksoy and M. Franklin, "$R \times W$: A scheduling approach for large-scale on-demand data broadcast," *IEEE/ACM Trans. Netw.*, vol. 7, no. 6, pp. 846–860, Dec. 1999.

[6] J. Xu, X. Tang, and W.-C. Lee, "Time-critical on-demand data broadcast: Algorithms, analysis, and performance evaluation," *IEEE Trans. Parallel Distrib. Syst.*, vol. 17, no. 1, pp. 3–14, Jan. 2006.

[7] K. Liu and V. Lee, "Simulation studies on scheduling requests for multiple data items in on-demand broadcast environments," *Perform. Eval.*, vol. 66, no. 7, pp. 368–379, Jul. 2009.

[8] J. Zhang, Q. Zhang, and W. Jia, "VC-MAC: A cooperative MAC protocol in vehicular networks," *IEEE Trans. Veh. Technol.*, vol. 58, no. 3, pp. 1561–1571, Mar. 2009.

[9] X. Ma, J. Zhang, X. Yin, and K. S. Trivedi, "Design and analysis of a robust broadcast scheme for VANET safety-related services," *IEEE Trans. Veh. Technol.*, vol. 61, no. 1, pp. 46–61, Jan. 2012.

[10] Q. Wang, P. Fan, and K. B. Letaief, "On the joint V2I and V2V scheduling for cooperative VANETS with network coding," *IEEE Trans. Veh. Technol.*, vol. 61, no. 1, pp. 62–73, Jan. 2012.

[11] C.-J. Chang, R.-G. Cheng, H.-T. Shih, and Y.-S. Chen, "Maximum freedom last scheduling algorithm for downlinks of DSRC networks," *IEEE Trans. Intell. Transp. Syst.*, vol. 8, no. 2, pp. 223–232, Jun. 2007.

[12] P. Xuan, S. Sen, O. Gonzalez, J. Fernandez, and K. Ramamritham, "Broadcast on demand: Efficient and timely dissemination of data in mobile environments," in *Proc. 3rd IEEE RTAS*, 1997, pp. 38–48.

[13] Y. Zhang, J. Zhao, and G. Cao, "On scheduling vehicle-roadside data access," in *Proc. 4th ACM Int. Workshop VANET*, 2007, pp. 9–18.

[14] F. J. Ros, P. M. Ruiz, and I. Stojmenovic, "Acknowledgment-based broadcast protocol for reliable and efficient data dissemination in vehicular ad hoc networks," *IEEE Trans. Mobile Comput.*, vol. 11, no. 1, pp. 33–46, Jan. 2012.

[15] K. Liu and V. Lee, "Adaptive data dissemination for time-constrained messages in dynamic vehicular networks," *Transp. Res. C, Emerging Technol.*, vol. 21, no. 1, pp. 214–229, Apr. 2012.

[16] Wikipedia, Clique Problem 2013. [Online]. Available: http://en.wikipedia.org/wiki/Clique_problem

[17] K. Liu and V. Lee, "On-demand broadcast for multiple-item requests in a multiple-channel environment," *Inf. Sci.*, vol. 180, no. 22, pp. 4336–4352, Nov. 2010.

[18] V. Lee and K. Liu, "Scheduling time-critical requests for multiple data objects in on-demand broadcast," *Concurrency Comput., Pract. Exper.*, vol. 22, no. 15, pp. 2124–2143, Oct. 2010.

[19] J.-Y. Ng, V. Chung-Sing, and C. Y. Hui, "Client-side caching strategies and on-demand broadcast algorithms for real-time information dispatch systems," *IEEE Trans. Broadcast.*, vol. 54, no. 1, pp. 24–35, Mar. 2008.

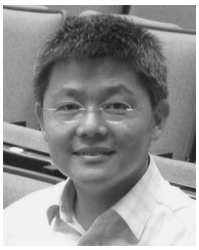
[20] H. Schwetman, "CSIM19: A powerful tool for building system models," in *Proc. 33nd WSC*, 2001, pp. 250–255.

[21] G. K. Zipf, *Human Behavior and the Principle of Least Effort*. Reading, MA, USA: Addison-Wesley, 1949.

[22] C. F. Daganzo, *Fundamentals of Transportation and Traffic Operations*. New York, NY, USA: Pergamon, 1997.
[23] P. Edara and D. Teodorović, "Model of an advance-booking system for highway trips," *Transp. Res. C, Emerging Technol.*, vol. 16, no. 1, pp. 36–53, Feb. 2008.
[24] "Amendment of the commission's rules regarding dedicated short-range communication services in the 5.850-5.925GHz band," Washington, DC, USA, FCC report and Order 06-110, Jul. 20, 2006.
[25] F. Bai and H. Krishnan, "Reliability analysis of DSRC wireless communication for vehicle safety applications," in *Proc. 9th Int. IEEE Conf. ITSC*, 2006, pp. 355–362.
[26] O. Maeshima, S. Cai, T. Honda, and H. Urayama, "A roadside-to-vehicle communication system for vehicle safety using dual frequency channels," in *Proc. 10th Int. IEEE Conf. ITSC*, 2007, pp. 349–354.
[27] M.-F. Jhang and W. Liao, "Cooperative and opportunistic channel access for vehicle to roadside (V2R) communications," *Mobile Netw. Appl.*, vol. 15, no. 1, pp. 13–19, Feb. 2010.
[28] T. K. Mak, K. P. Laberteaux, and R. Sengupta, "A multi-channel VANET providing concurrent safety and commercial services," in *Proc. 2nd ACM Int. Workshop VANET*, 2005, pp. 1–9.
[29] F. Farnoud and S. Valaee, "Reliable broadcast of safety messages in vehicular ad hoc networks," in *Proc. 28th IEEE INFOCOM*, 2009, pp. 226–234.
[30] J. A. Stankovic, S. H. Son, and J. Hansson, "Misconceptions about real-time databases," *Computer*, vol. 32, no. 6, pp. 29–36, Jun. 1999.
[31] M. Amirijoo, J. Hansson, and S. H. Son, "Specification and management of QoS in real-time databases supporting imprecise computations," *IEEE Trans. Comput.*, vol. 55, no. 3, pp. 304–319, Mar. 2006.
[32] W. Kang, K. Kapitanova, and S. H. Son, "RDDS: A real-time data distribution service for cyber-physical systems," *IEEE Trans. Ind. Informat.*, vol. 8, no. 2, pp. 393–405, May 2012.
[33] M. Xiong, S. Han, K.-Y. Lam, and D. Chen, "Deferrable scheduling for maintaining real-time data freshness: Algorithms, analysis, and results," *IEEE Trans. Comput.*, vol. 57, no. 7, pp. 952–964, Jul. 2008.
[34] J. Wang, S. Han, K.-Y. Lam, and A. K. Mok, "Maintaining data temporal consistency in distributed real-time systems," *Real-Time Syst.*, vol. 48, no. 4, pp. 387–429, Jul. 2012.
[35] J. W. Wong, "Broadcast delivery," *Proc. IEEE*, vol. 76, no. 12, pp. 1566–1577, Dec. 1988.
[36] K. Liu, V. Lee, J. Ng, and S. Son, "Scheduling temporal data for real-time requests in roadside-to-vehicle communication," in *Proc. 19th IEEE Int. Conf. RTCSA*, 2013, pp. 297–305.

Kai Liu (S'07–M'12) received the Ph.D. degree in computer science from City University of Hong Kong, Kowloon, Hong Kong, in 2011.

He is currently an Assistant Professor with the College of Computer Science, Chongqing University, Chongqing, China. From December 2010 to May 2011 he was a Visiting Scholar with the Department of Computer Science, University of Virginia, Charlottesville, VA, USA. From 2011 to 2014 he was a Postdoctoral Fellow with Nanyang Technological University, Singapore; City University of Hong Kong; and Hong Kong Baptist University, Kowloon, Hong Kong. His research interests include real-time scheduling, mobile computing, intelligent transportation systems, and vehicular cyber–physical systems.



Victor Chung Sing Lee (M'97) received the Ph.D. degree in computer science from City University of Hong Kong, Kowloon, Hong Kong, in 1997.

He is an Assistant Professor with the Department of Computer Science, City University of Hong Kong. His research interests include data dissemination in vehicular networks, real-time databases, and performance evaluation.

Dr. Lee is a member of the ACM and IEEE Computer Society. From 2006 to 2007 he was the Chairman of the IEEE Hong Kong Section Computer Society Chapter.

Joseph Kee-Yin Ng (M'91–SM'00) received the B.Sc., M.Sc., and Ph.D. degrees from University of Illinois at Urbana-Champaign, Champaign, IL, USA, all in computer science.

In 1993 he joined Hong Kong Baptist University (HKBU), Kowloon, Hong Kong, where he is a Professor with the Department of Computer Science. He is the Program Coordinator of the Computer Science degree program and introduced Health Information Technology and Health Informatics into the undergraduate, as well as the graduate, programs in HKBU. His research interests include real-time and embedded systems, multimedia communications, and ubiquitous/pervasive computing. He holds two patents. He has authored or coauthored over 135 technical papers in journals and conferences. He also served as Steering Chair, Program Chair, and General Chair for numerous International Conferences, as well as Associate Editor and Member of the editorial board of international journals. He is also a Director of the Hong Kong Internet Registration Corporation Limited.

Dr. Ng had also served as the Region 10 Coordinator for the Chapter Activities Board of the IEEE Computer Society and was the Coordinator of the IEEE Computer Society Distinguished Visitors Program (Asia/Pacific). Since 1991 he has been a member of the IEEE Computer Society. He has been an Exco member, General Secretary, Vice Chair, Chair, and now a past Chair and Exco member for the IEEE Hong Kong Section Computer Society Chapter. He has received numerous awards and certificates of appreciation from IEEE, IEEE Region 10, IEEE Computer Society, and from IEEE Hong Kong Section Computer Society Chapter for his Leadership and Services to the ICT Industry. He is also a member of the IEEE Communication Society, ACM, Hong Kong Computer Society, and Founding Member and Exco Member and Treasurer for the Internet Society Hong Kong Chapter.

Jun Chen received the Ph.D degree in computer Science from Wuhan University, Wuhan, China, in 2008.

From 2008 to 2009 she was a Senior Research Associate with the Department of Computer Science, City University of Hong Kong, Kowloon, Hong Kong. In 2012 she was a Visiting Fellow with the Department of Information System, City University of Hong Kong. She is currently an Associate Professor with the School of Information Management, Wuhan University. Her research interests include mobile computing and data management in wireless networks.

Sang Hyuk Son (M'85–SM'98–F'13) received the B.S. degree in electronics engineering from Seoul National University, Seoul, Korea; the M.S. degree from Korea Advanced Institute of Science and Technology (KAIST), Daejeon, Korea; and the Ph.D. degree in computer science from University of Maryland, College Park, MD, USA.

He is a Department Chair of Information and Communication Engineering with Gyeongbuk Institute of Science and Technology, Daegu, Korea. He has been a Professor with the Department of Computer Science, University of Virginia, Charlottesville, VA, USA, and a World Class University Chair Professor with Sogang University, Seoul. He has been a Visiting Professor with KAIST; City University of Hong Kong, Kowloon, Hong Kong; Ecole Centrale de Lille, Villeneuve-d'Ascq, France; Linköping University, Linköping, Sweden; and University of Skövde, Skövde, Sweden. His research interests include cyber–physical systems, real-time and embedded systems, database and data services, and wireless sensor networks. He has authored or coauthored over 290 papers and edited or authored four books in these areas. His research has been funded by National Science Foundation, DARPA, Office of Naval Research, Department of Energy, National Security Agency, and IBM.

From 2007 to 2008, Dr. Son was the Chair of the IEEE Technical Committee on Real-Time Systems. He is an Associate Editor for *Real-Time Systems Journal* and *Journal on Self Computing*, and has served on the editorial board of IEEE TRANSACTIONS ON COMPUTERS and IEEE TRANSACTIONS ON PARALLEL AND DISTRIBUTED SYSTEMS. He is also a member of the steering committee for Real-Time Computing Systems and Applications, Cyber-Physical Systems Week, and Software Technologies for Future Embedded and Ubiquitous Systems. He received the Outstanding Contribution Award form ACM/IEEE Cyber-Physical Systems Week in 2012.