Utilizing Road-side Infrastructure for Location-based Services in Vehicular Ad-hoc Networks

Chengcheng Dai
Department of Computer Science
City University of Hong Kong
Hong Kong
Email: chengcdai2@student.cityu.edu.hk

Chi-Yin Chow
Department of Computer Science
City University of Hong Kong
Hong Kong
Email: chiychow@cityu.edu.hk

Jiadong Zhang
Department of Computer Science
City University of Hong Kong
Hong Kong
Email: jzhang26@student.cityu.edu.hk

***Abstract*—Location-based services (LBS) can be treated as an abstraction of spatial queries, e.g. k-nearest-neighbor (k-NN) queries. In LBS, users retrieve their desired entertainment services and information from their mobile devices based on their geographical position at the request time through mobile network. The IEEE standardization of Dedicated Short Range Communication (DSRC)/Wireless Access in Vehicular Environments (WAVE) is designed for high-speed vehicle-to-vehicle and vehicle-to-infrastructure communication in vehicular ad-hoc networks (VANETs). However, most existing VANET-based LBS are designed for simple applications like information dissemination. In this paper, we develop a VANET-based LBS with the utilization of RSUs as an infrastructure support to process k-NN queries. In our framework, we design road-based semantic broadcasting and scheduling methods for RSUs to pour useful information into the network, and a communication protocol to handle incomplete transmission between vehicles and RSUs. We evaluate our framework with and without RSUs through simulated experiments. Experimental results show that the utilization of RSUs significantly improve the query response time, query answer accuracy, and communication overhead.**

I. Introduction

Nowadays users reply on location-based services (LBS) to obtain information and entertainment based to their current geographic positions detected by their mobile devices. LBS can be defined as an abstraction of spatial queries, e.g., k-nearest-neighbor (k-NN) queries (e.g., "Where are the five nearest Japanese restaurants to my car?"). Due to the growing need for high-speed data transmission in vehicular ad hoc networks (VANETs) (i.e., vehicle-to-vehicle and vehicle-to-infrastructure), the IEEE standardization of Dedicated Short Range Communication (DSRC)/Wireless Access in Vehicular Environments (WAVE) [1], [2] has been realized and it also allocates some channels for mobile infotainment, e.g., LBS. Both LBS and VANETs have been well studied, but the combination of these receives very little attention.

VANETs have many unique characteristics to distinguish themselves from conventional mobile ad hoc networks (MANETs) [1], [3], [4], e.g., constrained and predictable movements in road networks, highly dynamic network topology, carry-and-forward transmission paradigm, and vehicles with ample energy and high computational capacity. These characteristics make existing MANET-based LBS frameworks not suitable for VANETs.

In this paper, we aim to utilize *road-side units* (RSUs) as *infrastructure supports* for processing k-NN queries in VANETs. A user u's k-NN query is defined as $(u.l, u.k, u.r)$, where $u.l$ is u's location at the time of query and u wants to find at most k nearest objects of interests within a network distance $u.r$ from $u.l$. The key challenges of deploying RSUs for VANET-based LBS is two-fold: (1) how RSUs broadcast data with semantics and spatial locality that facilitate spatial query processing in road networks and (2) how vehicles deal with incomplete broadcast from a RSU due to out of the RSU's transmission range or network problems. For the first challenge, we propose a road-based semantic broadcasting for RSUs and use a Hilbert curve to order road-based semantics. For the second challenge, we design a communication protocol for vehicles for handling incomplete broadcast (i.e., a vehicle may lose connection to a RSU before receiving all the information about a road segment).

We compare the performance of VANET-based LBS without and with RSUs through simulated experiments on a real road network. Experimental results show that the utilization of RSUs significantly reduces the query response time, improve the query answer accuracy, and reduce the communication overhead of VANET-based LBS.

The rest of this paper is organized as follows. Section II highlights related work. Section III presents the system model. Section IV delineates the query processing algorithm. Section V describes the utilization of RSUs in VANET-based LBS. Section VI analyzes experiment results. Finally, Section VII concludes this paper.

II. Related Work

Existing MANET-based LBS frameworks can be categorized according to their supported services. (1) resource discovery services [5]–[8]. They simply disseminate queries and resource information in the network. Once there is a match between a query and an available resource, the resource information is sent back. (2) spatio-temporal queries with different focuses. There are interesting works on dealing with inaccurate position information [9], designing a distributed tree-like index structure with semantic caching [10], developing cooperative filtering to process skyline queries [11], and using cooperative monitoring techniques to support continuous LBS [12]–[15]. However, these frameworks assume that mobile nodes are frequently connected, so they cannot be applied to VANETs.

Most research of VANETs focuses on *non-spatial* applications such as routing protocols [4], [16], data aggregation [17], [18], information dissemination [19]–[25] and road-side units scheduling. Although the concepts of traffic-based forwarding [23] and data pouring [24] have been proposed for VANETs, spatio-temporal query processing is not considered. A few information dissemination frameworks have been proposed for VANETs. For example, FleaNet is a virtual market for resource discovery [26]. Vehicles work together to keep messages alive in a target area for a specified period of time and send them to all interested subscribers [27]–[29]. A vehicle can continuously monitor its nearest vehicles within its transmission range for safety purpose [30]. Unfortunately, these frameworks only support simple information dissemination applications [26]–[29] or keep track of neighboring vehicles (i.e., within one hop distance) [30]. Our work goes beyond such simple applications to support efficient k-NN query processing with the utilization of RSUs in VANETs.

III. System Model

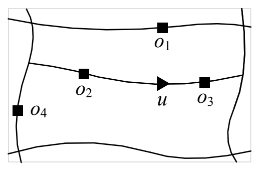
We here present the system model and define our problem.

Road networks. We assume that the road network of a city or county can be downloaded from a server or preloaded in the client device. Road networks are modeled to a weighted graph $G = (V, E)$, where E is the edge set that represents a set of road segments and V is the vertex set of road segments. Each edge is represented by two vertices v_iv_j. Road length is indicated as the weight of an edge. The distance from point p_i to point p_j (i.e., $dist(p_i, p_j)$) in the road network is defined by the shortest network distance from p_i to p_j and calculated through the network expansion technique [31]. Figure 1 depicts a simple weighted graph, where the vertices and edges are represented by circles and lines, respectively, and objects are represented by squares. A user u is represented by a triangle and its trajectory is supposed to be $v_5 \rightarrow v_3 \rightarrow u$. The network distance from u to o_1, i.e., $dist(u, o_1)$, is calculated as the shortest path distance $dist(u, v_4) + dist(v_4, v_2) + dist(v_2, o_1) = 1.5\ km$. A grid index structure is built on the road network to efficiently find an edge on which a given position locates.

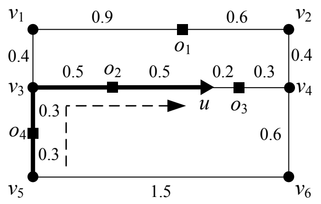
Spatial objects. Spatial objects are able to broadcast $(id, loc, data, TTL)$ to vehicles residing in its transmission range, where id is an unique object identity allocated by a certain registration server, loc is the object's location, $data$ contains both static information (e.g., phone number) and dynamic information (e.g., table availability and waiting time) and TTL is the valid time period for the dynamic information.

Vehicles and queries. Each vehicle is equipped with a positioning device (e.g., GPS) and a wireless communication interface (e.g., IEEE DSRC) to exchange data with other vehicles and receive information from spatial objects. A vehicle u can issue a k-NN query in a form of $Q = (u.l, u.k, u.r)$ to find $u.k$ nearest objects located within a network distance $u.r$ from u's location $u.l$ at the query time.

Road-side units (RSUs). RSUs are deployed in the road network, e.g., intersections of road segments. Each RSU collects nearby objects' information through the Internet and broadcasts the information to vehicles residing in its transmission range. RSUs broadcast information for various applications, so they periodically allocate a certain time period for our LBS application.



(a) Road map



(b) Graph model

Fig. 1: Road network model.

Road-based semantic caching. Each entry in a vehicle's local cache is in the form of $(eid, seg\_list, obj\_cnt, obj\_set)$, indicating that the vehicle user has cached a set obj_set of obj_cnt objects located on edge eid's disjoint segments in seg_list. If a vehicle traveled an entire edge v_iv_j, the seg_list contains only one segment v_iv_j and obj_set stores all the objects on v_iv_j. However, if v_iv_j has no object, $obj\_cnt = 0$ and obj_set is empty. Figure 1b depicts an example, where a user u traveled from v_5 to v_3 and then to her current location $u.l$ on edge v_3v_4. u knows the information of all the objects on edge v_5v_3 (i.e., o_4) and the information of o_2 on the segment $v_3u.l$. Thus, u's local cache stores two entries: (v_3v_5, $\{v_3v_5\}$, 1, $\{o_4\}$) and (v_3v_4, $\{v_3u.l\}$, 1, $\{o_2\}$).

Three key operations are defined to manage a local cache: (1) ***Insertion.*** If a new edge/segment p_ip_j overlaps an existing segment p_np_m, these two segments are combined into one segment p_sp_e where $p_s = \min(p_i, p_n)$ and $p_e = \min(p_j, p_m)$. Otherwise, p_ip_j and its objects are directly inserted to the local cache accordingly. (2) ***Update.*** Vehicles update their local cache when they receive updated information for a cached object. (3) ***Deletion.*** If an object became stale, it is removed from the local cache. For cache replacement, objects that are farthest away from the user have a highest chance to be replaced. When the obj_cnt of an edge/segment becomes 0 due to object deletion, the whole entry needs to be removed. An edge/segment that has no object, i.e. $obj\_cnt = 0$, and is far away from the vehicle's current location is also removed from the local cache during cache replacement. The network distance from a user u to an edge/segment p_sp_e is calculated as $\min(dist(u, p_s), dist(u, p_e))$.

Time-to-live estimation for query messages. We extend an one-way-traffic statistic model [32] to a two-way-traffic model to analyze the carrying and forwarding time for delivering a message in a road network. A message m forwarded over an edge v_iv_j with a length l from v_i to v_j, l may be partitioned into *forwarding distance* l_f, *carry distance* l_c, and *backward-forwarding distance* l_b, where l_f is the distance traveled through wireless transmission by vehicles moving from v_i to v_j, l_b is the distance traveled through wireless transmission by vehicles moving from v_j to v_i, and l_c is the distance traveled by physical movement of vehicles. $l_c = l - l_f - l_b$ and l_f is calculated by the following equation:

$$l_f = s\frac{1/\lambda_i - (R_c/s + 1/\lambda_i)e^{-\lambda_i R_c/s}}{1 - e^{-\lambda_i R_c/s}}, \tag{1}$$

where R_c is the transmission range of a vehicle, s is the movement speed, and λ_i is the vehicle arrival rate (i.e., the number of vehicles per second) at vertex v_i. l_b is calculated by Equation 1 by replacing λ_i with λ_j. The delivery time is the sum of *forwarding*, *carry*, and *backward-forwarding* delays:

$$t_{delivery} = (l_f/R_c) \times t_{1hop} + (l_c/s) + (l_b/R_c) \times t_{1hop}, \quad (2)$$

where the forwarding and backward-forwarding delays are the number of hops required to travel l_f and l_b multiplied by the one-hop transmission time t_{1hop}, respectively, and the carry delay is the time to travel l_c at the speed s.

Since a VANET is frequently partitioned, a *coverage* factor α ($0 \leq \alpha \leq 100\%$) is incorporated into the time-to-live (TTL) estimation of a query message to balance a performance trade-off between the answer accuracy and the query response time. For example, a user may want to search 80% of a required search area in order to obtain a query answer faster. The basic idea of the TTL estimation is to use the network expansion technique to select all possible paths from a querying vehicle's location such that they cover α of all the required search segments, and then TTL is the largest estimated delivery delay of the shortest selected path.

Hybrid routing protocol. After a vehicle u_j receives a query from u_i, u_j needs to send a reply message back to u_i. Considering u_i is moving after sending the query, existing location-aware point-to-point (P2P) routing protocol may be insufficient. To solve this problem, we consider a conservative approach, in which u_i may appear at any location within a network distance $d = (t_c - t_q) \times s_{max}$ from u_i's location at the query time t_q, where t_c is the current time and s_{max} is the maximum possible speed of u_i. u_j first uses a P2P routing protocol (e.g., [33]) to send a message back to u_i's query location, and then all message receivers residing u_i's possible location range broadcast to their neighbors until the message TTL elapses. Such a TTL is calculated as the sum of the P2P delivery time (using Equation 2) and the delivery time to cover u_i's possible location range (using the TTL estimation for query messages).

Problem definition. Given a user u's k-NN query (i.e., $Q = (u.l, u.k, u.r)$), u communicates with other vehicles in a VANET to find at most $u.k$ nearest objects located within a network distance $u.r$ from u's location $u.l$. Our objectives are to utilize road-side units, which push the information of their nearby objects to vehicles residing in their transmission ranges, to reduce the query response time and the communication overhead, and improve the query answer accuracy.

IV. k-NN QUERY PROCESSING

In this section, we present how a vehicle user u finds an answer for her k-NN query $Q = (u.l, u.k, u.r)$. The algorithm has three major steps, namely, *candidate object step*, *verification step*, and *query answer step*. We will present the details of the k-NN query processing algorithm in this section and the utilization of RSUs for VANET-based LBS in Section V.

Step 1 of 3: Candidate object step. This step prunes the user-defined search range $u.r$ by finding $u.k$ candidate objects and takes the network distance from the query location $u.l$ to the k-th farthest candidate as the new search range $u.r'$. The vehicle user issuing a k-NN query $Q = (u_q.l, u_q.k, u_q.r)$ is considered as a *query sender* u_q and the vehicles receiving Q are considered as *query receivers* u_r.

Query sender. u_q issues a candidate object search request $R_{obj} = (u_q.l, u_q.k, u_q.r, u_q.TTL, u_q.List)$ based on the number $u_q.k'$ of objects that located within the network distance $u_q.r$ from $u_q.l$ in its local cache, where $u_q.TTL$ is the TTL for searching edges within $u_q.r$ from $u_q.l$. If $u_q.k' < u_q.k$, $u_q.k$ is set to $u_q.k - u_q.k'$ and $u_q.List$ records the identities of all $u_q.k'$ objects. u_q then sends R_{obj} to its neighbors. If $u_q.k' \geq u_q.k$ or u_q receives $u_q.k - u_q k'$ objects from query receivers, u_q calculates the pruned search range that has $u_q.k$ objects. In the first case, u_q selects the $u_q.k$ nearest objects to $u_q.l$ and the search range $u_q.r$ is reduced to the network distance r' from $u_q.l$ to the $u_q.k$-th nearest object. If $u_q.TTL$ elapses but R_{obj} fails to find $u_q.k - u_q.k'$ objects, u_q reduces $u_q.r$ to the distance to the farthest object in $u_q.List$.

Query receiver. u_r only processes R_{obj} if it has not received R_{obj} from u_q before, $R_{obj}.TTL$ does not expire, and $dist(u_r.l, u_q.l) < u_q.r$. u_r selects the cached objects located within $u_q.r$ from $u_q.l$ but not in $u_q.List$, denoted the number as $u_r.k'$. If $u_r.k' < u_q.k$, u_r updates the $u_q.List$ in R_{obj} and sends it to its neighbors. u_r also sends the object information along with its road-based semantics of $u_r.k'$ objects to u_q. If $u_r.k' > u_q.k$, u_r returns the $u_q.k$ nearest objects to u_q and R_{obj} will not be forwarded.

Step 2 of 3: Verification step. This step searches the network within $u.r'$ from $u.l$ to verify if nearer objects exists till the TTL of the query message expires, or the u obtains all the objects within $u.r'$ based on road-based semantics.

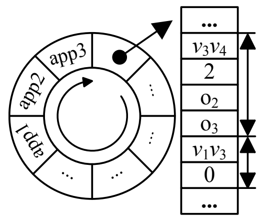
Query sender. u_q broadcasts a verification request $R_{verify} = (u_q.l, u_q.r', u_q.TTL, u_q.List, u_q.Segment)$ to search all the edges within $u_q.r'$ from $u_q.l$, where $u_q.TTL$ is the TTL for searching all the edges within $u_q.r'$, objects located within $u_q.r'$ from $u_q.l$ are added to $u_q.List$, and road-based semantics about the road segments within $u_q.r'$ are added to $u_q.Segment$. When road-based semantics stored in u_q's local cache covers all the edges within $u_q.r'$ or $u_q.TTL$ elapses, u_q proceeds to the next step.

Query receiver. u_r processes R_{verify} if it has not received R_{verify} from u_q before, $R_{verify}.TTL$ does not expire and $dist(u_r.l, u_q.l) < u_q.r'$. u_r updates R_{verify} according to its local cache. For each cached edge/segment $v_i v_j$ that is within $u_q.r'$ from $u_q.l$, if it is not in $u_q.Segment$, objects located on it are added to $u_q.List$ and $v_i v_j$ is added to $u_q.Segment$. If $v_i v_j$ is already in $u_q.Segment$, but $v_i v_j$ is not covered by any segment $v_s v_e$ in $u_q.Segment$, $v_i v_j$ is added to $u_q.Segment$. If there is any object within $u_q.r'$ from $u_q.l$ on $v_i v_j$ but not on $v_s v_e$, $u_q.List$ is updated accordingly. u_r sends the updated R_{verify} to its neighbors and the object information with its road semantics to u_q.

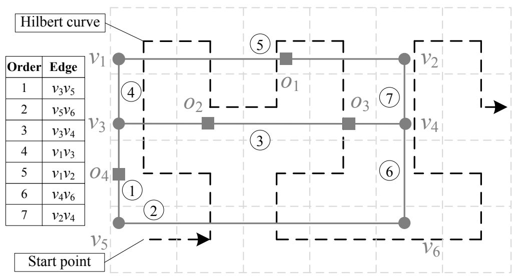
Step 3 of 3: Query answer step. In in step, u_q finds the $u_q.k$ nearest objects within the network distance $u_q.r$ from u_q as a query answer.

V. UTILIZATION OF ROAD-SIDE UNITS

RSUs are deployed to disseminate the information about the objects in their vicinity to vehicles located in their transmission range. Pouring more useful object information with road-based semantics to them can facilitate query processing. The basic idea is to divide the system area into disjoint *duty regions*



(a) Broadcast cycle



Order	Edge
1	v_3v_5
2	v_5v_6
3	v_3v_4
4	v_1v_3
5	v_1v_2
6	v_4v_6
7	v_2v_4

(b) Broadcast scheduling

Case 1: link breaks before transmitting o_2:

edge_id	segment_list	object_set
No update for v_3v_4		

Case 2: link breaks before transmitting o_3:

edge_id	segment_list	object_set
v_3v_4	$\{v_3o_2\}$	$\{(o_2, ...)\}$

Case 3: link breaks after transmitting o_2 & o_3:

edge_id	segment_list	object_set
v_3v_4	$\{v_3v_4\}$	$\{(o_2, ...), (o_3, ...)\}$

(c) Handling incomplete transmission

Fig. 2: The utilization of RSUs for k-NN query processing.

and one RSU is assigned to each duty region. There is no assumption on the partition method as long as the system area is partitioned into disjoint regions. Each RSU is responsible for disseminating the information of the set of edges intersecting its assigned region. We first propose *road-based semantic broadcasting* to facilitate k-NN query processing and use the *Hilbert curve* to order objects and road-based semantics in a broadcast channel. We then describe a communication protocol for vehicles to deal with incomplete transmission for an edge from a RSU.

A. Road-based Semantic Broadcasting and Scheduling

Suppose RSUs are deployed by a service provider and our application is one of many applications using RSUs to disseminate information in VANETs, so they broadcast information for our LBS application every t seconds. Figure 2a illustrates an example, where a RSU broadcasts information for eight applications in its broadcast cycle. The whole system area is divided into equal-sized grid cells. Breath-first search is utilized to find a certain number of grid cells around a RSU as its *duty region*. The RSU is responsible for broadcasting the information of all the edges along with their objects intersecting its duty region.

For each edge v_sv_e intersecting a RSU R's duty region, R first broadcasts its identity v_sv_e (where $v_s < v_e$) and the number of objects n located on it, and then sequentially broadcasts the objects on v_sv_e from v_s to v_e. For the edge v_3v_4 in Figure 1b, the RSU broadcasts the edge id of v_3v_4, the number of objects on v_3v_4 (i.e. 2), the information of o_2, and then the information of o_3 (Figure 2a). If there is no object located on v_sv_e, n is set to 0 and no object will be transmitted (e.g., v_1v_3 in Figure 2a).

To retain the spatial locality of the information about a duty region broadcasted from RSUs, the nearby edges should be scheduled together. We employ the Hilbert space-filling curves [34] to map two-dimensional edges to one-dimensional ordering, so as to order the set of edges. Figure 2b depicts an example to show how the Hilbert curve order the edges in our running example. In this example, the curve represented by a series of black connected dotted lines starts at the left-bottom cell that intersects two edges v_1v_5 and v_5v_6, so v_1v_5 and v_5v_6 are ordered as the first and second edges, respectively, where we break ties by selecting the edge with more objects. The curve next encounters two edges v_3v_4 and v_1v_3, which are ordered as the third and fourth ones, and so on. The table in Figure 2b shows the ordering of all the edges.

B. Handling Incomplete Transmission

A vehicle user u can receive information broadcast from a RSU when it is residing in a RSU's transmission range and updates its local cache accordingly. We here describe a communication protocol for u to handle incomplete transmission of edge v_iv_j, when a communication link between u and the RSU breaks during transmission. The main idea is to cache the segment from v_i to the location of the last completely transmitted object. We use an example to illustrate the communication protocol. Suppose R broadcasts edge v_3v_4 to u (Figure 2a). The order of transmission is the edge id of v_3v_4, $n = 2$, the information of o_2, and the information of o_3 (because o_2 is closer to v_3 than o_3). Figure 2c depicts three possible cases of transmitting v_3v_4. (1) If the communication link breaks before transmitting the information of o_2, there is no update for u's local cache. (2) If the link breaks before transmitting the information of o_3, the segment from v_3 to the location of o_2 with the information of o_2 are added to u's local cache. (3) If the link breaks after transmitting the information of o_2 and o_3, since $n = 2$, u adds the edge v_3v_4 with the information of o_2 and o_3 to its local cache.

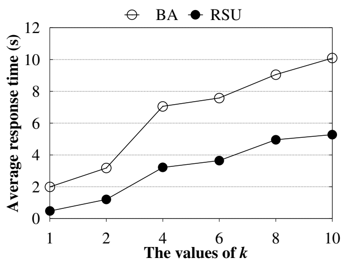
VI. Performance Evaluation

In this section, we first describe our experiment settings, and then analyze experimental results.

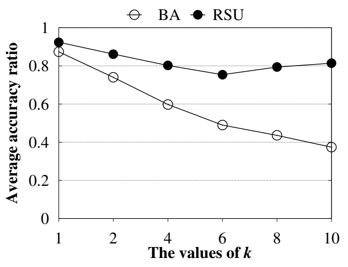
A. Experiment Settings

We study the performance of our k-NN query processing algorithm without and with the support of RSUs, denoted as BA and RSU respectively. Their performance is measured in terms of *average query response time per query*, *average accuracy ratio per query* and *average number of messages per query*. The response time of a query is the duration from the time when the query is issued to the time when its answer is computed. The accuracy ratio is the number of queries that return accurate answers to the total number of queries. For the average number of messages, we consider all the messages generated during the experiment, including query messages and the messages broadcasted from RSUs.

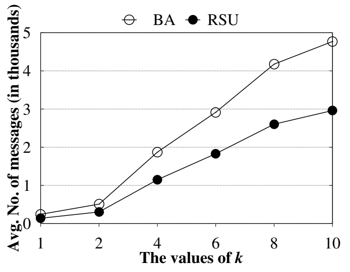
Our experiments were implemented in C++ and run on an Ubuntu 11.10 machine with 3.4GHz Intel Core i7 processor and 16GB RAM. We extracted an area of 4 km × 4 km with 1,772 vertices and 2,181 edges from the road map of Beijing, China from Cloudmade [35]. 10,000 vehicle trajectories are randomly generated by using the A* algorithm [36]. 600 objects are uniformly distributed in the system. All vehicles



(a) Average response time

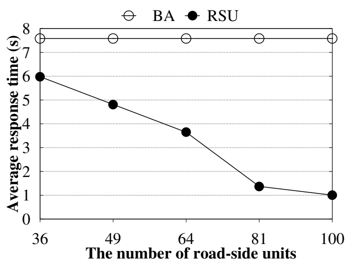


(b) Average accuracy ratio

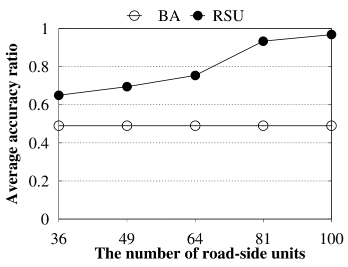


(c) Average number of messages

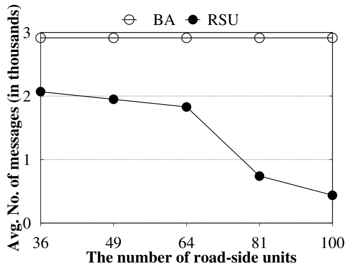
Fig. 3: Effect of the requested number of nearest neighbors (k).



(a) Average response time

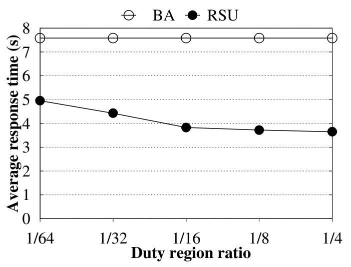


(b) Average accuracy ratio

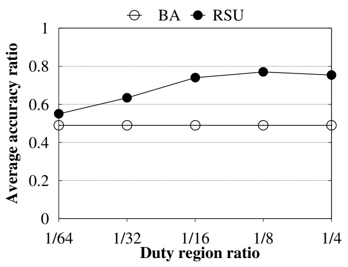


(c) Average number of messages

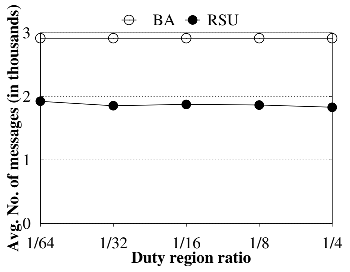
Fig. 4: Effect of the number of RSUs.



(a) Average response time



(b) Average Accuracy ratio



(c) Average number of messages

Fig. 5: Effect of the duty region size of RSUs.

move at a speed of 30 km per hour, their transmission range and bandwidth are 100 meters and 2 Mbps, respectively, and their cache size is 10% of the total number of objects in the system. The time-to-live (TTL) of object information is 60 seconds. The coverage factor α used in the estimation of a query message TLL is set to 80%. The user-specified query range $r = 2$ km. The default RSU broadcast period is 6s. The default number of RSUs is 8×8; and hence, the system area is divided into 64 equal-sized regions and one RSU assigned to each region. We started recording simulation results after 500 queries in order to avoid a transient effect. The simulation results are based on 1,000 queries with the requested number of nearest neighbors $k = 6$.

B. Effect of the Requested Number of Nearest Neighbors (k)

Figure 3 depicts the performance of BA and RSU with respect to various requested numbers of nearest neighbors (k) which is increased from 1 to 10. The results show that RSU effectively improves the query response time and the query answer accuracy, as depicted in Figures 3a and 3b, respectively. The query answer accuracy of BA becomes much lower when the value of k increases. Although RSU broadcasts extra messages to VANETs, it pours more useful information of objects and edges into the network, the querying vehicles of RSU are easier to find nearby objects and a smaller required search range in the *verification step*. As a result, the querying vehicles experience shorter query response time and higher query answer accuracy. The shortened query response time avoids some query message generated by vehicles. This offsets the number of extra messages generated by RSUs (Figure 3c).

C. Effect of the Number of RSUs

Figure 4 depicts the effect of the number of RSUs on the performance of RSU by varying the number of RSUs from 6×6 to 10×10. The number of RSUs does not affect BA, but its performance is also plotted for reference. It is expected that more RSUs reduce the query response time and increase the query answer accuracy, as shown in Figures 4a and 4b, respectively. This is because more useful object and edge information can be poured into the network. When the querying vehicle receives the information of nearby objects from the RSUs, it can derive a smaller required search range r' in the *verification step*. As the number of RSUs increases, the benefit of utilizing RSUs offsets the extra communication overhead of their broadcast messages, and thus, the communication overhead of RSU reduces (Figure 4c).

D. Effect of the RSU Duty Region Size

Figure 5 depicts the effect of various duty region sizes on the performance of RSU by varying the duty region size of every RSU from $1/64$ to $1/4$ of the size of the entire system area. The number of RSUs remains 8×8. The performance of BA is also plotted for reference. It is expected that a larger duty region size let RSUs pour more useful object and edge information into the network that reduces the query response time and increase the query answer accuracy, as shown in Figures 5a and 5b, respectively. It is interesting to see that there is no improvement on the query response time and the query answer accuracy, when the duty region size further increases, as depicted in Figures. 5a and 5b, respectively. The main reason is that querying vehicles are usually interested in nearby objects, i.e., the spatial locality of location-based queries.

VII. Conclusion

In this paper, we proposed a new k-NN query processing framework with the utilization of road-side units (RSUs) as an infrastructure support for VANETs. We designed road-based semantic broadcasting and scheduling methods to enable RSUs to pour useful information to the network, and a communication protocol for a vehicle to deal with incomplete transmission when it suddenly loses connection to a RSU. We evaluated the performance of our framework with and without the deployment of RSUs. Experimental results show that the utilization of RSUs significantly improves the query response time, query answer accuracy, and communication overhead. Our future research directions are to study how to place RSUs in VANETs and how to let vehicles work together with RSUs to further improve the spatial query processing performance in VANETs.

Acknowledgement

The work described in this paper was partially supported by grants from City University of Hong Kong (Project No. 7002867, 7002722, and 7002686).

References

[1] D. Jiang and L. Delgrossi, "IEEE 802.11p: Towards an international standard for wireless access in vehicular environments," in *Proc. of IEEE VTC*, 2008.

[2] IEEE WAVE Working Group, "http://vii.path.berkeley.edu/1609_wave/."

[3] E. M. Belding and C.-K. Toh, "A review of current routing protocols for ad hoc mobile wireless networks," *IEEE Personal Communications*, vol. 6, no. 2, pp. 46–55, 1999.

[4] F. Li and Y. Wang, "Routing in vehicular ad hoc networks: A survey," *IEEE Vehicular Technology Magazine*, vol. 2, no. 2, pp. 12–22, 2007.

[5] H. Cao, O. Wolfson, B. Xu, and H. Yin, "MOBI-DIC: Mobile discovery of local resources in peer-to-peer wireless network," *IEEE Data Eng. Bull.*, vol. 28, no. 3, pp. 11–18, 2005.

[6] O. Wolfson, B. Xu, and R. M. Tanner, "Mobile peer-to-peer data dissemination with resource constraints," in *Proc. of IEEE MDM*, 2007.

[7] O. Wolfson, B. Xu, H. Yin, and H. Cao, "Search-and-discover in mobile P2P network databases," in *Proc. of IEEE ICDCS*, 2006.

[8] X. Zhu, B. Xu, and O. Wolfson, "Spatial queries in disconnected mobile networks," in *Proc. of ACM GIS*, 2008.

[9] D. Dudkowski, P. J. Marrón, and K. Rothermel, "Efficient algorithms for probabilistic spatial queries in mobile ad hoc networks," in *IEEE COMSWARE*, 2006.

[10] Q. Zhu, D. L. Lee, and W.-C. Lee, "Collaborative caching for spatial queries in mobile P2P networks," in *Proc. of IEEE ICDE*, 2011.

[11] Z. Huang, C. S. Jensen, H. Lu, and B. C. Ooi, "Skyline queries against mobile lightweight devices in MANETs," in *Proc. of IEEE ICDE*, 2006.

[12] C.-Y. Chow, M. Mokbel, and H. V. Leong, "On efficient and scalable support of continuous queries in mobile peer-to-peer environments," *IEEE TMC*, vol. 10, no. 10, pp. 1473–1487, 2011.

[13] T. T. Do, K. A. Hua, and C.-S. Lin, "ExtRange: Continuous moving range queries in mobile peer-to-peer networks," in *Proc. of IEEE MDM*, 2009.

[14] P. Galdames, K. Kim, and Y. Cai, "A generic platform for efficient processing of spatial monitoring queries in mobile peer-to-peer networks," in *Proc. of IEEE MDM*, 2010.

[15] K. Kim, Y. Cai, and W. Tavanapong, "Safe-Time: Distributed real-time monitoring of cKNN in mobile peer-to-peer networks," in *Proc. of IEEE MDM*, 2008.

[16] K. C. Lee, U. Lee, and M. Gerla, "Advances in vehicular ad-hoc networks: Developments and challenges," in *Survey of Routing Protocols in Vehicular Ad Hoc Networks*. IGI Global, 2009.

[17] C. Lochert, B. Scheuermann, C. Wewetzer, A. Luebke, and M. Mauve, "Data aggregation and roadside unit placement for a VANET traffic information system," in *Proc. of ACM VANET*, 2008.

[18] B. Yu, J. Gong, and C.-Z. Xue, "Catch-up: A data aggregation scheme for VANETs," in *Proc. of ACM VANET*, 2008.

[19] F. Malandrino, C. Casetti, C.-F. Chiasserini, and M. Fiore, "Content downloading in vehicular networks: What really matters," in *Proc. of IEEE INFOCOM*, 2011.

[20] M. Fiore and J. M. Barceló-Ordinas, "Cooperative download in urban vehicular networks," in *IEEE MASS*, 2009.

[21] M. D. Dikaiakos, A. Florides, T. Nadeem, and L. Iftode, "Location-aware services over vehicular ad-hoc networks using car-to-car communication," *IEEE JSAC*, vol. 25, no. 8, pp. 1590–1602, 2007.

[22] L. Wischhof, A. Ebner, and H. Rohling, "Information dissemination in self-organizing intervehicle networks," *IEEE Trans. on ITS*, vol. 6, no. 1, pp. 90–101, 2005.

[23] J. Zhao and G. Cao, "VADD: Vehicle-assisted data delivery in vehicular ad hoc networks," in *Proc. of IEEE INFOCOM*, 2006.

[24] J. Zhao, Y. Zhang, and G. Cao, "Data pouring and buffering on the road: A new data dissemination paradigm for vehicular ad hoc networks," *IEEE TVT*, vol. 56, no. 6, pp. 3266–3277, 2007.

[25] T. Zhong, B. Xu, and O. Wolfson, "Disseminating real-time traffic information in vehicular ad-hoc networks," in *Prof. of IEEE IV*, 2008.

[26] U. Lee, J. Lee, J.-S. Park, and M. Gerla, "FleaNet: A virtual market place on vehicular networks," *IEEE TVT*, vol. 59, no. 1, pp. 344–355, 2010.

[27] D. Borsetti, M. Fiore, C. Casetti, and C.-F. Chiasserini, "Cooperative support for localized services in vanets," in *ACM MSWiM*, 2009.

[28] I. Leontiadis, P. Costa, and C. Mascolo, "A hybrid approach for content-based publish/subscribe in vehicular networks," *Pervasive and Mobile Computing*, vol. 5, no. 6, pp. 697–713, 2009.

[29] I. Leontiadis and C. Mascolo, "Opportunistic spatio-temporal dissemination system for vehicular networks," in *International Workshop on Mobile Opportunistic Networking*, 2007.

[30] B. Xu, O. Wolfson, and H. J. Cho, "Monitoring neighboring vehicles for safety via V2V communication," in *IEEE ICVES*, 2011.

[31] D. Papadias, J. Zhang, N. Mamoulis, and Y. Tao, "Query processing in spatial network databases," in *Proc. of VLDB*, 2003.

[32] J. Jeong, S. Guo, Y. Gu, T. He, and D. H. Du, "Trajectory-based data forwarding for light-traffic vehicular ad-hoc networks," *IEEE TPDS*, vol. 22, no. 5, pp. 743–757, 2011.

[33] V. Naumov, R. Baumann, and T. Gross, "An evaluation of inter-vehicle ad hoc networks based on realistic vehicular traces," in *ACM MobiHoc*, 2006.

[34] M. F. Mokbel, W. G. Aref, and I. Kamel, "Analysis of multi-dimensional space-filling curves," *Geoinformatica*, vol. 7, no. 3, pp. 179–209, 2003.

[35] CloudMade, http://downloads.cloudmade.com/asia/eastern_asia/china/.

[36] P. Hart, N. Nilsson, and B. Raphael, "A formal basis for the heuristic determination of minimum cost paths," *IEEE Trans. on SSC*, vol. 4, no. 2, pp. 100–107, 1968.