MQry: Elastic Validity Region for Querying Mobile Point-of-Interests in Infrastructure-Less Networks

Byungkwan Jung, Sunho Lim, Jinseok Chae, Cong Pu, and Manki Min

***Abstract*—A region-based querying has been widely used to update the freshness of query result and reduce the query traffic in diverse wireless networks. Most prior querying approaches implicitly assume infrastructure-based networks but we need to relax this assumption by considering limited coverage or even unavailability of the network. In this paper, we propose an elastic validity region and its corresponding query operation, called MQry, in an infrastructure-less network with mobile point-of-interests (POIs). Due to the mobility of nodes and POIs, it is challenging how to efficiently build and update a validity region. Each query replying node that contributes in building an initial validity region judiciously switches its monitor operation on the queried POI. The query issuing node adjusts the current validity region to reduce the number of redundant queries without hurting the freshness of queried result. The query operations of both query issuing and replying nodes are depicted using finite state machines for clarity. We conduct extensive simulation experiments using the OMNeT++ for performance evaluation and comparison with four existing schemes. The simulation results show that the MQry can adaptively change the size of validity region and be a viable querying approach in infrastructure-less networks.**

***Index Terms*—Infrastructure-less networks, mobile point-of-interests, query processing, validity region.**

I. Introduction

As WIRELESS and mobile devices such as smartphones have become increasingly popular and resource-rich, users (denoted later as nodes) can freely query any point-of-interest (POI) and enjoy location-based service (LBS) deployed in diverse infrastructure-based wireless networks. A region-based querying strategy and its variants [1]–[5] have been widely used to efficiently update a query result received from an LBS server (denoted later in short as a server). A query issuing node sends a query to the server to receive and collect the information of a queried object, such as a static or mobile POI. The server replies a query result piggybacked with a set of locations (e.g., (x, y) coordinates) that can form a virtual area, called as a *validity region*. The query result remains the same as long as the query issuing node is located within the validity region. The query issuing node does not need to periodically check the validity of query result with the server, and it requeries to update its query result only when it is out of its validity region. Thus, the query traffic can be significantly reduced, and the scarce wireless bandwidth can also be saved.

In this paper, we identify two major issues in order to realize a ubiquitous query operation in the LBS. First, due to the limited coverage or unavailability of the 3G/4G networks, nodes may not be connected with the infrastructure but be remained uncovered for a certain period in a blackspot, urban, or remotely isolated area. The current infrastructure could even be collapsed and become unavailable when an unexpected disaster occurs, such as a recent earthquake or hurricane and its follow-up flooding. For example, it is critical for a rescue team to transceive emergency queries and check their validity of query results through single or multihop relays without the help of infrastructure [6]–[9]. Most of the validity region-based approaches [2]–[5] may not directly be applied to an infrastructure-less network, because an initial validity region built with the replies of the first query is not updated until a new query is generated. No or intermittent connections to the server could mislead nodes to generate redundant queries or answer a query with an obsolete query result. Second, in addition to the node mobility, the location of *mobile* POI is time-varying and its corresponding validity region can shortly be obsolete. This is different from a static POI, where the location of POI is fixed or is not changed in a short period of time and thus, its corresponding validity region is often considered as valid until the query issuing node moves out of the boundary of validity region. In [10] and [11], a set of validity region update strategies is proposed to approximate the size of validity region based on the overhearing, mobility of POIs, and requery probability. For example, an initial size of validity region is intentionally built smaller than that of the originally planned validity region. Since the mobility of mobile POIs often affects the validity of the original query result, an initially built validity region can quickly be obsolete. Thus, the query issuing node may not be aware of whether it is already located outside of the validity region and missed to regenerate and send a new query. Under the smaller validity region, the query issuing node proactively generates a new query even when it is located within the validity region, incurring redundant queries. In fact, the initially built validity region is not adaptively updated during the query operation.

In light of these issues, we focus on a mobile querying operation by investigating how to efficiently query mobile POIs and

Manuscript received December 24, 2018; revised May 24, 2019; accepted June 29, 2019. This work was supported by International Cooperative Research Grant from Incheon National University, Incheon, South Korea, in 2016. *(Corresponding authors: Sunho Lim; Jinseok Chae.)*

B. Jung and S. Lim are with the T^{2}WISTOR: TTU Wireless Mobile Networking Laboratory, Department of Computer Science, Texas Tech University, Lubbock, TX 79409 USA (e-mail: byung.jung@ttu.edu; sunho.lim@ttu.edu).

J. Chae is with the Department of Computer Science and Engineering, Incheon National University, Incheon 22012, South Korea (e-mail: jschae@inu.ac.kr).

C. Pu is with the Division of Computer Science, Marshall University, Huntington, WV 25755 USA (e-mail: puc@marshall.edu).

M. Min is with the Computer Science Program, Louisiana Tech University, Ruston, LA 71272 USA (e-mail: mankimin@latech.edu).

Digital Object Identifier 10.1109/JSYST.2019.2929559

This article has been accepted for inclusion in a future issue of this journal. Content is final as presented, with the exception of pagination.

update the corresponding validity region in an infrastructure-less wireless network, where a mobile ad hoc network (MANET) is deployed to see the impact of limited connectivity and frequent disconnections due to the mobility. Our threefold contributions are briefly summarized as follows.

1) We first observe the impact of mobile POIs on an initially built validity region and compare the query performance with three different combinations of static and mobile POIs in a MANET. We also investigate the query overhearing and measure the effect of query traffic in terms of the number of overheard query replies.
2) We then propose a mobile querying operation with a validity region update technique, called MQry, to efficiently query mobile POIs and reduce the redundant query traffic. In the MQry, each query replying node judiciously switches its monitor operation on the queried POI and the query issuing node efficiently adjusts the current validity region without hurting the freshness of queried result. A finite state machine (FSM), called FSMQry, is also presented to see the proposed querying operation clearly.
3) We finally modify the safe exit algorithm [4] originally designed for an infrastructure-based network to work in a MANET, called *Greedy*, and use it as a performance lower bound. We also compare the query performance with *Adapt:Ovhr* [10], where a query issuing node adaptively extends its initially built validity region.

In this paper, we develop a customized discrete-event driven simulator using the OMNeT++ [12] and conduct the performance evaluation and comparison in terms of the time spent in validity region and the average number of queries as well as replies per query. The simulation results with a set of network snapshots show that the MQry can efficiently extend the time spent in validity region, reduce the number of queries, and adaptively change the size of validity region in the presence of mobile POIs. Note that previous works [10], [11] are focused on a set of basic *static* validity regions and its variants to sense the query performance. However, the primary contribution of this paper is to develop an elastic validity region and its corresponding algorithm and communication protocol that can adaptively update its shape using shrinkage and expansion operations.

The rest of this paper is organized as follows. The prior querying schemes are reviewed and analyzed in Section II. We discuss the validity region based querying with mobile POIs and its expected query traffic and propose an elastic validity region based querying scheme in Section III. The proposed approach is evaluated with extensive experiments in Section IV. The conclusion and future work are presented in Section VI.

II. Related Work

In this section, we review and analyze prior querying schemes in terms of static or adaptive validity regions in diverse wireless networks.

A. Static Validity Region

Once a validity region is initially set up with the first query replies, it is not updated until a query issuing node moves out of the region. A centralized server deployed in diverse infrastructure-based networks often receives the query, provides the information of queried POIs, the mobility of query issuing node, and rebuilds the validity region [2]–[5]. When a query issuing node generates a query, a centralized server builds a safety region consisting of a set of exit points in terms of the road segments in a vehicular ad hoc network (VANET) [4]. Similar to the validity region, the query result will be valid as far as the query issuing node is roaming within the safety region. The query issuing node is aware of the exit points and regenerates a new query to update the query result when it passes one of the exit points. Then, the server resets a new safety region and sends new exit points to the query issuing node. In [5], a range safe region is proposed to reduce continuous monitoring of queried POIs, where each POI is treated as a node and covered by the same detection range of the node. The server overlaps the multiple detection ranges of queried POIs and sends their intersected area as a safe region to the query issuing node. A grid-based rectangle safe region and its corresponding region computing techniques are proposed to implement scalable processing of spatial alarms, alerting a future spatial location of POI [3]. A safe region is optimized in a greedy manner considering a tradeoff between the server workload and the node energy consumption. A continuous spatial querying scheme is proposed in a mobile peer-to-peer network [13]. Each query replying node located within the query range receives a query periodically broadcasted by a query issuing node and sets a safe period based on the Euclidean distance between the two query replying and issuing nodes. When the safe period is expired, the query replying node sends a query result of POI and waits for an acknowledgment from the query issuing node for updating the safe period. Here, the size of the detection range is the same as the one-hop communication range of node.

A variant of range-skyline query approach is proposed in a mobile environment, considering both spatial and nonspatial criteria to improve the quality-of-service of query result [14]. A query issuing node sets its query detection range, floods a query request message, and receives query results from mobile query replying nodes continuously during a certain period. The query issuing node predicts a safe time, implying how long a mobile query replying node will stay in the range, and decides whether to regenerate a new query for updating the query result. In [15], a top-k query with the help of data replication approach is proposed in MANETs. Initially, a query issuing node floods a query request message and then disseminates the information of top-k query result and the average number of neighbors back to the network. Then, each node calculates and allocates the replicas. The query issuing node gradually expands its search range in terms of the number of hops to the neighbor nodes based on the number of neighbors until the top-k is acquired. The query issuing node also increases its search range to maximize the probability of finding the top-k query result or minimize the number of nodes accessed.

B. Adaptive Validity Region

An initially built validity region can be updated even before a query issuing node moves out of the region to improve the query performance. Several validity region-based query processing schemes are proposed in a MANET, where the region is formed by x-y and/or convex-hull-based techniques [10]. A query issuing node can actively overhear any on-flying query and extend its initially built region to improve the query performance. In [16], multiple safe regions are generated and located along the route between a source and destination locations for a route planning query in a road network. A centralized server receives a set of POIs, searches a trip route that visits all the POIs to the destination, and builds a set of safe regions that includes each queried POI among rival POIs located around the route.

In summary, due to the mobility of nodes and POIs and lack of centralized coordination, prior validity region based querying approaches with infrastructure support may not directly be applied to MANETs. A limited adaptation of validity region solely operated by a query issuing node also may not be enough to update the validity region efficiently.

III. Proposed Elastic Validity Region Aware Querying

In this section, we first introduce a validity region-based query operation and its potential issue in the presence of mobile POIs. We experiment three different query strategies with static and mobile POIs to compare the query performance. Then, we investigate query overhearing and propose an elastic validity region and its corresponding query operations for mobile POIs to improve query performance and reduce query traffic in MANETs. Note that static POI and its corresponding validity region-based query operation are investigated in [10] and [11].

A. *Validity Region Based Querying*

When a node retrieves location information of a static object (e.g., POI), it floods a query to other nodes to cooperatively collect the location information of the POI in MANETs. Each node is assumed to equip with an on-board global positioning system (GPS) and is aware of the current location. Since the flooding of a query may incur the broadcast storm problem [17], a controlled flooding in terms of the number of hops to be propagated in the network is often deployed to reduce the query traffic. In Fig. 1(a), when a node (e.g., n_b or n_c) receives the query and detects the queried POI located in the detection range, it approximates the location of the POI by transmitting a query reply containing its current location to the query issuing node (e.g., n_a) as a query result. Note that it is hard (if it is not impossible) for the query replying node to identify the exact location of the POI without additional equipment. The detection range may vary depending on the equipment, but we assume that the detection range is the same as the communication range of the query replying node for the sake of simplicity in this paper. When the query issuing node begins to receive replies, it builds a validity region based on the locations of query replying nodes (e.g., n_b to n_h) as shown in Fig. 1(b). A validity region is defined as a virtual area where a query result remains the same as long as a query issuing node is located within the area. Here, we deploy a convex-hull-based approach called Conv [10], to efficiently create a validity region. The Conv deploys the Graham-Scan-based method [18] to construct the validity region that is the smallest convex polygon containing a given set of locations of the query replying nodes in a two-dimensional area. The query issuing node does not generate a query again for the same POI as long as it is located within the validity region. In order to facilitate to identify whether the query issuing node is located within the validity region, the network is virtually divided into a set of grids, where each square region is called *cell* of the same size. Then, the query issuing node stores the validity region in the form of a set of cell *ids*. As soon as the query issuing node moves out of the validity region, it generates and floods a new query for the same POI again to receive the updated query result and build a new validity region accordingly.

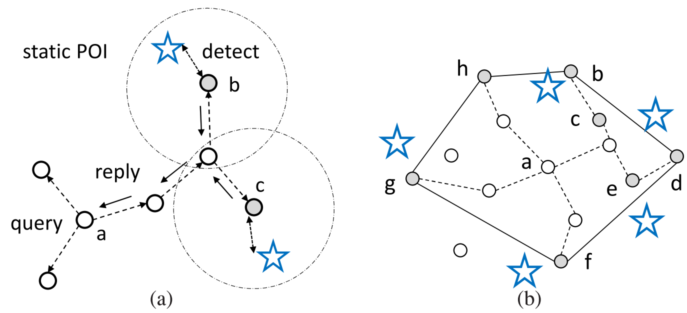

Fig. 1. Query issuing node (n_a) queries a static POI and builds a validity region. Here, each node is marked as an empty circle. A query replying node and a queried POI are also marked as a shaded circle and a star, respectively. The directions of propagated query and its reply are represented by dashed and solid arrows, respectively. The detection range is also represented by a dash-dotted line. In addition, a validity region is the closed area connected by query replying nodes, n_b, n_d, n_f, n_g, and n_h, using a solid line. In Fig. 1(b), one node located outside of the validity region is not a query participating node because it may not receive a query or does not detect the queried POI from the query issuing node.

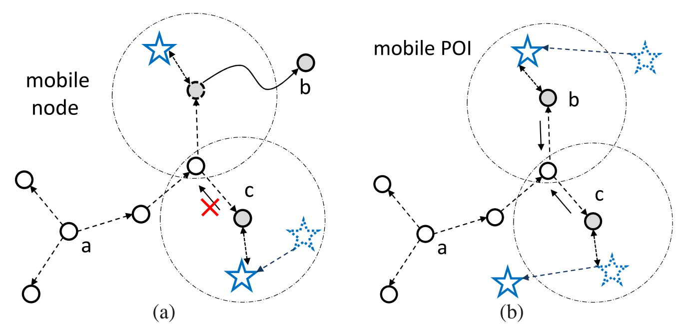

Fig. 2. Mobility of query replying nodes (e.g., n_b and n_c) and POIs and its corresponding reply to the query issuing node (e.g., n_a) for updating the validity region.

B. *Validity Region for Mobile POI*

In static POI, the mobility of query replying node does not affect the validity region because a query issuing node builds the validity region with initially received replies, containing the original locations of the query replying nodes when they detect a queried POI. The query issuing node is not aware of whether the query replying nodes have moved away from the queried POI. The query issuing node updates the validity region only when it moves out of the current validity region. In Fig. 2(a), a query replying node (e.g., n_b) has detected a queried POI and replied a query result, and then later it moves out of the detection range. In mobile POI, however, the mobilities of both POI and query replying node affect the validity region. If the query issuing node is not aware of the whereabouts of the POI, it cannot make sure whether the current validity region is still valid. The validity region becomes invalid if the received query results are changed. For example, in Fig. 2(a), if a queried POI moves within the same detection range of query replying node (e.g., n_c), this does not change the query result that the query replying node sent before. Thus, the query replying node does not send an additional reply. In Fig. 2(b), if a query replying node (e.g., n_b) detects the same type[1] of queried POI moving into the detection range, it sends another reply to the query issuing node for updating the current validity region. If the POI moves out of the detection range, the query replying node (e.g., n_b) that originally detected the POI sends another reply. Whenever the query issuing node receives a new reply or any additional reply, it updates the current validity region even if it does not move out of the validity region. In order to efficiently update the validity region, it is essential to keep track of the queried POI, but this could incur a significant communication overhead.

In this paper, we measure the communication overhead in terms of the number of transmitted packets to update the validity region with static and mobile POIs. We experiment with a 200×500 m^2 rectangle network, where 100 nodes are uniformly distributed and move according to the random waypoint mobility, 1.0 m/sec. The total number of POIs is 200, and 10% of POIs are mobile in the network. An adaptive approach is deployed to conduct the query operation called Adapt:Ovhr [10]. The Adapt:Ovhr is designed to reduce the number of queries by opportunistically utilizing the overheard on-flying replies and adaptively update the current validity region without regenerating a new query. A query issuing node quickly builds an initial validity region as soon as it receives the first reply and keeps extending the validity region upon receiving the following replies, combining the Rect [10] and Conv. For performance comparison, we first consider a case where all queried POIs are static, denoted as Case I, which is used as a performance lower bound. We also consider a case, where all queried POIs are mobile, denoted as Case II. In both Case I and Case II, when a node receives a query and detects a queried POI, it sends a reply to the query issuing node only once. In addition, we further extend the Adapt:Ovhr by actively monitoring mobile POIs, called Case III. In the Case III, each query replying node continuously monitors the queried POI and sends an additional reply to the query issuing node, whenever the queried POI moves out of the detection range or the same type of queried POI moves into the detection range.

We show the results in terms of the average number of queries, the average time spent in a validity region, total time spent in the validity region in percentage over total simulation time, and the average number of replies per query in Fig. 3. In Fig. 3(a) and (b), the Case I shows the highest number of queries but the lowest time spent in the validity region. Since each query replying node replies only once, query issuing nodes have a less chance to overhear on-flying replies and update the validity region. The Case III shows the best performance due to the active monitoring of the queried POI and thus, query issuing nodes can frequently overhear on-flying replies and update the validity region accordingly. This leads to the lowest number of queries but the highest time spent in the validity region. The Case II shows the performance between that of the Case I and Case III. In Fig. 3(c), all Cases I–III show that querying issuing nodes spend about 98% of total simulation time in the validity region, indicating that each query issuing node generates few queries. In Fig. 3(d), the Case III shows the highest number of replies per query because each query replying node actively replies whenever the queried POI is moved out of the detection range or newly detected.

[1]In this paper, we assume that the POI type can be diverse including but not limited to any geographical spot, wildfire, and moving vehicle or drone depending on the target applications, such as collaborative exploration, search and rescue, disaster recovery, or surveillance [11].

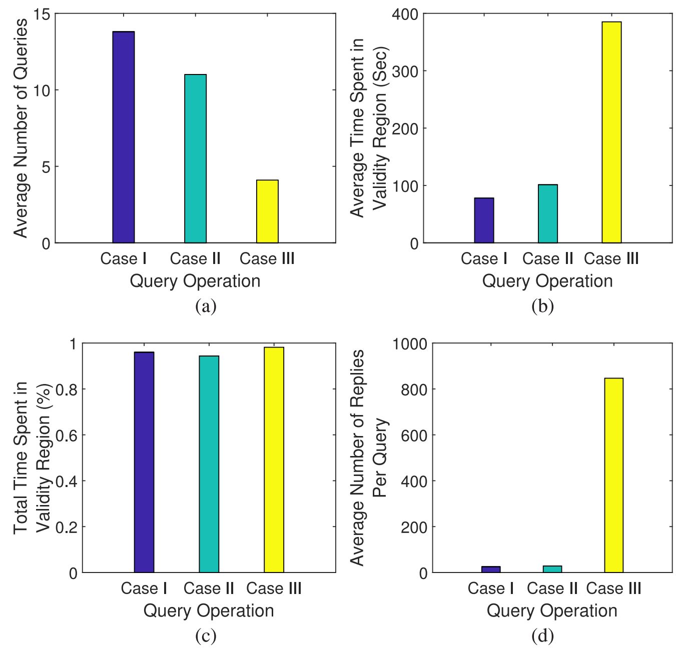

Fig. 3. Performance comparison of three different query operations with static and mobile POIs: Case I, Case II, and Case III.

In summary, if the query issuing node receives more replies from query replying nodes by active monitoring the queried POI, it can frequently update the current validity region and reduce the number of queries. However, the query traffic in terms of the number of replies per query can increase significantly.

C. Query Overhearing

Due to the nature of wireless communication, a *Reply* packet originally designed to unicast back to a query issuing node is in fact propagated omnidirectionally, such as a broadcast packet. Any adjacent node located along the path between the query replying and issuing nodes can overhear on-flying packets. Overhearing can improve the query performance by eavesdropping other communications (e.g., *Query* or *Reply*) to collect query information. To see the effect of query performance, we measure an average number of overheard replies against the number of POI types by deploying the Adapt:Ovhr, where the overheard on-flying replies are utilized to update the current validity region.

This article has been accepted for inclusion in a future issue of this journal. Content is final as presented, with the exception of pagination.

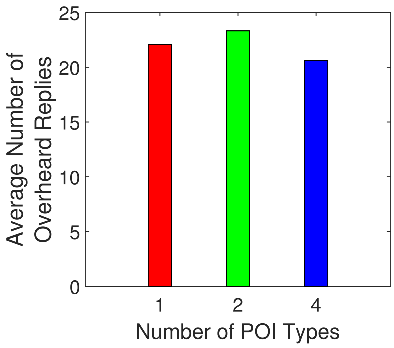

Fig. 4. Average number of overheard replies against a number of POI types. The total number of POIs is set to 200 in the network, where 10% of POIs are mobile.

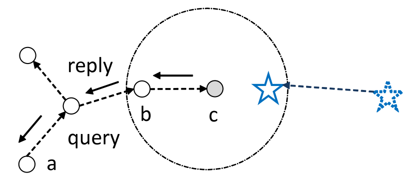

Fig. 5. Snapshot of network topology, where n_a generates a query and n_c cannot detect a queried POI initially but can detect it later and send a *Reply* because the queried POI moves into the detection range of n_c.

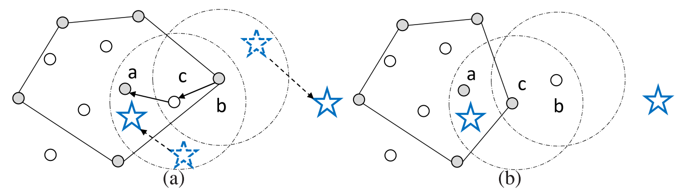

Fig. 6. Due to the mobile POIs, an initially built validity region shrinks. Here, a query replying node, n_b, sends a *Reply'* to a query issuing node n_a via n_c.

In Fig. 4, as the number of POI types decreases from four to two, the average number of overheard replies increases. Each query issuing node has more chances to overhear the same type of POI replies destined to other query issuing nodes. When there is only a single POI type in the network, however, the average number of overheard replies decreases. This is because each query issuing node can frequently update its current validity region with any overheard on-flying reply. Thus, each query issuing node can stay longer within the validity region, leading to generate the lesser number of new queries.

In summary, overhearing[2] on-flying replies and updating the validity region can extend the time spent of query issuing node in the validity region and reduce the number of new queries that significantly increases the query traffic.

D. Elastic Validity Region

Unlike prior approaches [1], [3]–[5], [10], where either a query issuing node or a centralized server operates in updating the validity region, we propose an elastic validity region based querying approach for mobile POIs, called MQry, to efficiently update the validity region by the collaboration of both query replying and issuing nodes in MANETs. The basic idea is that each query replying node judiciously decides whether to begin or stop responding to the queried POI to shrink or expand the validity region. The query issuing node also overhears any on-flying *Reply* to expand the validity region as well. Detailed querying operations are followed.

First, when a query issuing node generates a query for a POI, it broadcasts a *Query* to other nodes in the network. The *Query* is propagated to the nodes located up to the maximum three hops ($h_{\max}$) away from the query issuing node. When a node receives the *Query* and detects the queried POI, it sends a *Reply* containing its current location as a query result to the query issuing node. When the query issuing node begins to receive the *Reply*, it initiates to build a validity region using the convex-hull and then updates the validity region based on the additionally arriving *Reply*.

Second, when a node receives a *Query*, it sets its monitor state to M_{on}, monitors the queried POI, and sends a *Reply* if the queried POI is detected. If the node cannot find the queried POI within its detection range, it keeps its monitor state to M_{on} and continues to monitor the queried POI. Here, depending on whether to detect the queried POI, we classify a node into either a query replying node that replies a *Reply* back to the query issuing node, or a query receiving node that receives a *Query* but cannot detect the queried POI. If a query replying node receives a *Reply* from the node located farther in terms of the number of hops from the querying issuing node, it forwards the *Reply* and resets its monitor state to M_{off}. This is because the validity region is built based on the convex-hull, where the nodes located farther most likely contribute in building the validity region. If a query receiving node receives a *Reply* from its adjacent nodes, it keeps its monitor state to M_{on}.

Third, if a query replying node does not detect the queried POI anymore, it sends a *Reply'* to the query issuing node, resets its monitor state to M_{off}, and does not involve in any querying operation for the queried POI. If a query replying node newly detects the queried POI, it also sends a *Reply'* and keeps its monitor state to M_{on}. If a query receiving node newly detects the queried POI, it sends a *Reply* to the query issuing node and sets its monitor state to M_{on}. When the query issuing node receives either *Reply'* or *Reply*, it updates the initially built validity region.

Fourth, suppose a query generating node (e.g., n_a) broadcasts a *Query* and query receiving nodes (e.g., n_b and n_c) initially cannot detect a queried POI in the network, where the queried POI is moving toward n_c as shown in Fig. 5. When n_c detects the queried POI, it replies a *Reply* to n_a and set its monitor state to M_{on} because it is a potential contributor in building the validity region. Note that when an intermediate node (e.g., n_b) forwards the *Reply* received from its farther node, n_c, it sets its monitor state to M_{off}. However, if n_b receives a *Reply'* from n_c implying that n_c does not detect the queried POI anymore and affects the initially built validity region, it sets its monitor state to M_{on} to seek the contribution in updating the validity region by detecting the queried POI.

[2]In this paper, we use the terms *receiving* and *overhearing* interchangeably unless otherwise specified.

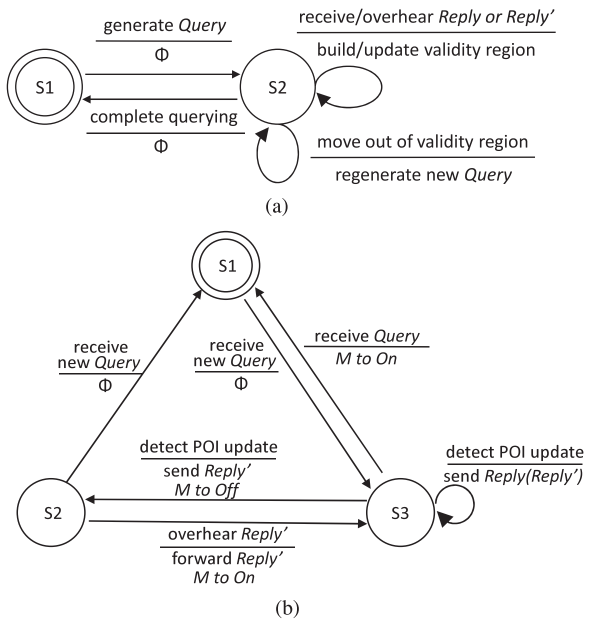

Fig. 7. FSMs of node behavior in the MQry, where ∅ and double circles denote no operation and initial state, respectively. (a) Query issuing node. (b) Query replying node.

Notations:

• n_i, n_j (or n_k): A query issuing node and a query replying node, respectively.
• h, h_{max}: A number of hops from a query issuing node (e.g., n_i), and the maximum number of hops for a *Query* that can be propagated.
• $Query[nid, POI_{type}, h]$: A query packet containing the id of query issuing node (nid), a queried POI type (POI_{type}), and the number of hops (h).
• $Reply[nid, POI_{type}, l_{nid}(x, y)]$: A reply packet containing the id of query replying node (nid), a queried POI type (POI_{type}), and the current location of query replying node ($l_{nid}(x, y)$).
• M_{nid}: A monitoring status of query replying node (e.g., n_j or n_k).

◇ When n_j receives a $Query[i, POI_{type}, h]$ broadcasted from n_i,
  if $h_{n_j} > h_{max}$
    Drop the $Query[i, POI_{type}, h]$ and exit;
  else
    if POI_{type} is detected
      Send a $Reply[j, POI_{type}, l_j(x, y)]$ to n_i;
    else
      Cache the $Query[i, POI_{type}, h]$;
    Rebroadcast the $Query[i, POI_{type}, h$++];
  Set M_j to On;
◇ When n_k receives a $Reply[j, POI_{type}, l_j(x, y)]$ forwarded from n_j,
  if $h_{n_k} < h_{n_j}$
    Set M_k to *Off*;
    Forward the $Reply[j, POI_{type}, l_j(x, y)]$ to n_i
◇ When n_i receives a $Reply[j, POI_{type}, l_j(x, y)]$ (or $Reply[k, POI_{type}, l_k(x, y)]$) from n_j (or n_k),
  Build a validity region using the Adapt:Ovhr [10];

Fig. 8. Pseudo code of basic query processing with emphasis on the changes of monitoring state.

Notations:

• $S_{loc,i}$: A set of locations (e.g., $l_j(x, y)$ or $l_k(x, y)$) cached in n_i.
• $Reply'[nid, M_{nid}]$: A reply packet containing the id of query replying node (nid) and its monitoring status (M_{nid}).

◇ When n_j no longer detects the queried POI,
  if M_j is On
    Set M_j to *Off*;
    Send a $Reply'[j, M_j]$ to n_i;
◇ When n_j detects the queried POI moved into or out of its detection range,
  if M_j is On
    Send a $Reply'[j, M_j]$ to n_i;
◇ When n_k overhears the $Reply'[j, M_j]$ sent from n_j,
  if $h_{n_j} > h_{n_k}$ $\wedge$ M_k is *Off* $\wedge$ M_j is *Off*
    Set M_k to On;
  Forward the $Reply'[j, M_j]$ to n_i;
◇ When n_i receives the $Reply'[j, M_j]$ sent from n_j,
  if $l_j(x, y) \notin S_{loc,i}$;
    Add $l_j(x, y)$ to the $S_{loc,i}$;
  Select contributing locations and update the validity region accordingly;

Fig. 9. Pseudo code of updating the validity region.

In Fig. 6, we show a case when a validity region shrinks. Query replying nodes (e.g., n_b and n_c) initially detect the queried POIs located within their detection range and a query issuing node (n_a) builds a validity region accordingly as shown in Fig. 6(a). Here, n_b contributes in building the validity region because it is located farther than n_c. Then, one of the queried POIs moves out of the detection range of n_b. Since n_b does not detect the queried POI anymore, it sends a *Reply'* to n_a via n_c. The other queried POI also moves but it is still located within the detection range of n_c. When n_a receives the *Reply'*, it updates the initially built validity region by adding the location of n_c as shown in Fig. 6(b).

Finally, FSMs of query issuing and replying nodes for the MQry, called FSMQry, are presented to clearly see the monitor operation and its corresponding state changes in Fig. 7. The FSMQry begins with state s_1. The state transition is initiated by an event (e) and then its corresponding action (a) is followed. Both event and action in state i (s_i) are represented as $\frac{e_i}{a_i}$. Note that the states s_2 and s_3 correspond to M_{on} and M_{off}, respectively, in Fig. 7(b). A set of major querying operations is also shown in Figs. 8 and 9.

IV. Performance Evaluation

We evaluate the proposed scheme and conduct the performance comparison study by changing a set of key simulation parameters in MANETs.

A. Simulation Testbed

We developed a customized discrete-event driven simulator based on the OMNeT++ [12]. A 500×200 m^2 rectangle network is deployed, where 100 nodes are uniformly distributed. The network area is virtually divided into a set of grids, where each cell size (c_{size}) is set to 10×10 m^2. Each node is equipped with an on-board GPS receiver and aware of its current location and its corresponding cell. Here, the performance impact of varying cell sizes has been studied in [11]. The POIs are also uniformly distributed in the network, where the number of mobile POIs varies from 20% to 100% of the total number of POIs and a single POI type is used. The random waypoint mobility model [19] is used to simulate both node and POI mobility patterns. The speed of both node and POI is set to 1 and 5 m/sec, respectively. The radio transmission range (r) is set to 50 m and the two-ray ground propagation channel is assumed with a data rate of 2 Mb/s. We deploy a simple CSMA/CA-based medium access

TABLE I
SIMULATION PARAMETERS

Parameter	Value
Node velocity	1.0 m/sec
Mobile POI velocity	5.0 m/sec
Number of nodes	100
Number of POIs	50
Communication range	50 m
Mobility type	Random Waypoint
Network size	500×200 m^2
Virtual grid size	10×10 m^2
Data transmission rate	2 Mbps

for the link layer. A controlled flooding in terms of the number of hops ($h_{\max}$) is deployed to limit the propagation of query generated from a query issuing node to reduce query traffic. Since a query issuing node regenerates a new query whenever it moves out of its validity region, it does not follow a periodic query arrival pattern. A queried data size is set to 16 kb. A set of major simulation parameters are summarized in Table I.

B. Simulation Results

We measure the performance in terms of the time spent in validity region and the average number of queries as well as replies per query. The simulation is repeated ten times by changing random seed numbers that primarily affect network topology and node and POI mobilities. Each simulation lasts up to 1000 s. The results are collected using 90% confidence interval and the predicted values lie within 10% of the mean. For performance comparison, we compare our approach with the following four schemes [4], [11]. In this paper, we modify the safe exit algorithm [4], originally designed for an infrastructure-based network, to work in a MANET, called *Greedy*. The Greedy can be used as a performance lower bound.

1) *Greedy:* We assume that there is a centralized server, which is aware of the mobilities of nodes and POIs in a real-time manner. When a query issuing node generates a query, it sends the query to the server directly. The server replies a query result with a set of locations of query replying nodes, ideally located the maximum number of hops (i.e., three hops) away from the query issuing node. Whenever the query issuing node moves, it measures the current distances to the locations of all query replying nodes to check whether each distance is less than and equal to the distance of maximum three hops.
2) *Adapt:Ovhr:* A validity region is built with the combination of basic rectangle and convex-hull-based validity regions. The validity region can also be extended by opportunistic overhearing of a query reply forwarded to another query issuing node.
3) *Reduced:* A query issuing node intentionally reduces the size of validity region to proactively avoid any miss in regenerating a new query by further limiting the propagation of query, resulting in a lesser number of query replies and a small sized validity region.
4) *Prob:* A set of requery probabilities (p_{qry}) is assigned in each cell located within a validity region. Whenever a query issuing node moves and visits a new cell, it generates a random number (e.g., rand[0, 1]), compares the number with p_{qry} assigned, and decides whether to reset the validity region and regenerate a new query.

1) Snapshots of Validity Region: We first snapshot a set of validity regions to observe any change depending on the number of mobile POIs as shown in Fig. 10. In this paper, we experiment with the low and high mobilities of POI by deploying 20% and 80% of mobile POIs out of a total number of POIs in the network, respectively. In Fig. 10(a)–(c), an initially built validity region [see Fig. 10(a)] is not changed much as shown in Fig. 10(b) and (c). A validity region is initially built based on the locations of six query replying nodes that detect the queried POI. With the low mobility of POIs, the query replying nodes do not frequently detect the changes of queried POI, such as a new POI is detected or the queried POI moves out of the detection range. Thus, the number of query replying nodes that contribute in updating the validity region is not significantly changed. In Fig. 10(d)–(f), due to the high mobility of mobile POIs, however, the size of validity region changes significantly. Note that the locations contributing to the validity region fluctuate because the query replying nodes frequently detect the changes of the queried POI.

2) Average Time Spent in Validity Region: We measure the time spent in validity region and compare the MQry with four existing schemes by changing the number of mobile POIs in the network. In Fig. 11(a), both Adapt:Ovhr and Prob show the longest time spent in the low mobility of POIs. This is because the Adapt:Ovhr adaptively extend the validity region. In the Prob, a query issuing node has a higher requery probability as it moves toward the boundary of validity region. Thus, the query issuing node can stay longer within the validity region than that of the other schemes. The MQry shows a similar performance with both Greedy and Reduced schemes because query replying nodes do not frequently detect any change of the queried POI and thus, the validity region is not updated. In the high mobility of POIs, however, the MQry shows the longest time spent in validity region for the rest of schemes. All schemes except the MQry show the reduced time spent in validity region. In particular, the Greedy shows the shortest time spent in validity region because it is sensitive to the distance changes between a query issuing node and query replying nodes. In the MQry, the query issuing node keeps updating the validity region based on the *Reply* sent from its query replying nodes whenever they detect any change of the queried POI, leading to a longer time spent in the updated validity region.

3) Average Number of Queries: We count the number of queries in Fig. 11(b). The Greedy shows the worst performance in both low and high mobilities of POIs. Since the Greedy is sensitive to the mobilities of nodes and POIs, a query issuing node frequently generates a new query whenever its current distances to all locations of query replying nodes are more than $h_{\max}$. The Reduced shows higher number of queries because it has a smaller validity region than that of the other three schemes. The MQry shows the best performance in the high mobility of POIs because it adaptively updates the current validity region according to the newly or additionally detected queried POIs, leading to generate fewer new queries. The MQry also shows

This article has been accepted for inclusion in a future issue of this journal. Content is final as presented, with the exception of pagination.

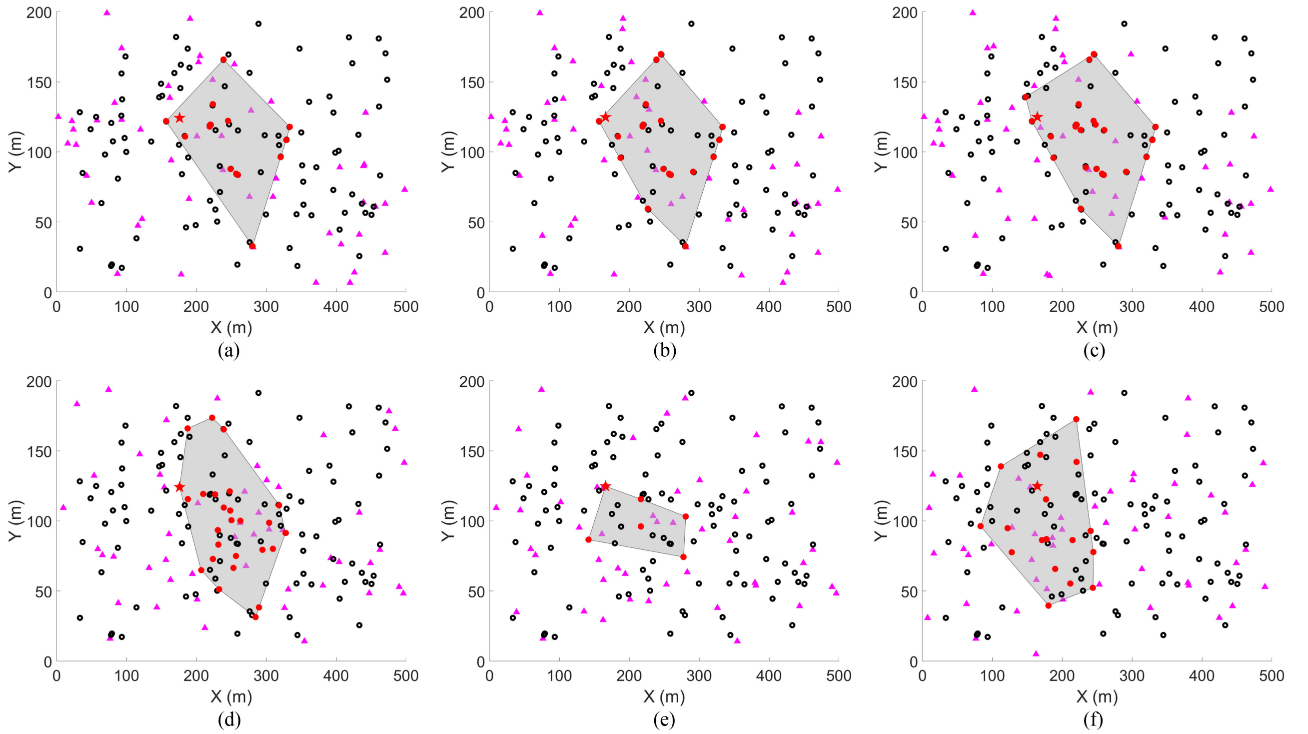

Fig. 10. Set of snapshots is shown to observe the changes of validity region depending on the number of mobile POIs in the network, where 20% and 80% of total number of POIs are mobile POIs in (a)–(c) and (d)–(f), respectively. Here, locations of original query issuing node and its query replying nodes are marked by a red star and a red circle, respectively. Both nodes and POIs are also marked by a white circle and a pink triangle, respectively. (a) $t = 100$ s, mobile POIs = 20%. (b) $t = 110$ s, mobile POIs = 20%. (c) $t = 120$ s, mobile POIs = 20%. (d) $t = 100$ s, mobile POIs = 80%. (e) $t = 110$ s, mobile POIs = 80%. (f) $t = 120$ s, mobile POIs = 80%.

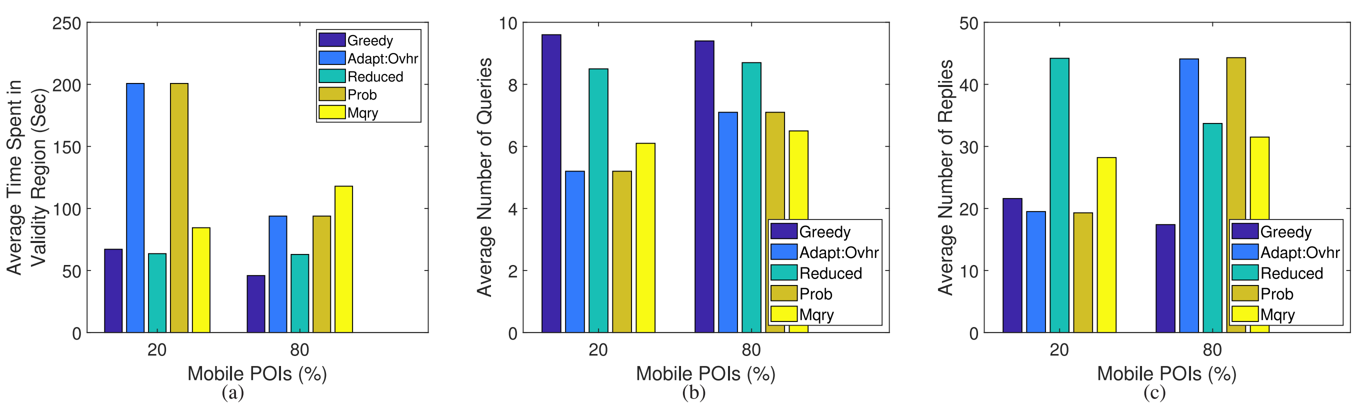

Fig. 11. Proposed approach is compared with four existing schemes, Greedy, Adapt:Ovhr, Reduced, and Prob, by changing the number of mobile POIs in the network.

a less sensitivity to the mobilities of POIs compared to that of both Adapt:Ovhr and Prob schemes.

4) Average Number of Replies Per Query: We count the number of replies per query in Fig. 11(c). In this paper, a query issuing node initiates to build a validity region using the convex-hull after receiving at least three replies from its corresponding query replying nodes. The Reduced shows the worst performance because the query issuing node receives the replies from the smaller number of query replying nodes, located less number of hops away than $h_{\max}$, and builds a smaller size of validity region. Since the query issuing node frequently moves out of the validity region, however, it repeatedly builds a new validity region based on the additional replies from its query replying nodes. Thus, the Reduced shows the highest number of replies per query. The Greedy shows the competitive performance with both Adapt:Ovhr and Prob in the low mobility of POIs but has the lowest number of replies per query in the high mobility of POIs. In this paper, we implicitly assume that the query issuing node is able to measure the distances to its corresponding query replying nodes without additional message exchanges in the Greedy. Both Adapt:Ovhr and Prob schemes show the fluctuated performance depending on the mobilities of POIs. In particular, they show the

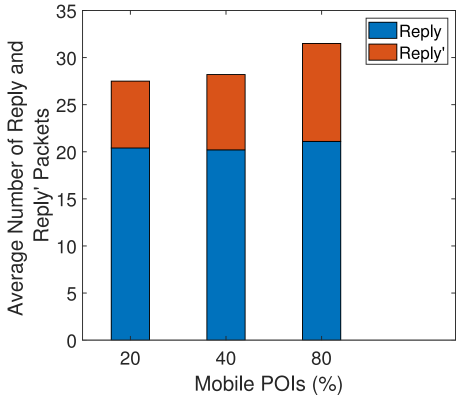

Fig. 12. Average number of *Reply* and *Reply'* packets against the number of mobile POIs in the network.

worst performance in the high mobility of POIs. This is because the validity region built based on the high mobility of POIs can lead the query issuing node to move out and regenerate a new query frequently. The MQry shows the competitive performance in both mobilities of a little higher number of replies per query than that of Greedy, Adapt:Ovhr, and Prob. Note that the number of replies per query increases a little in the high mobility of POIs compared to that of low mobility of POIs.

5) Average Number of Reply and Reply' Packets: Finally, we count the number of *Reply* and *Reply'* packets by varying the number of mobile POIs in Fig. 12. A query issuing node builds a validity region based on the *Reply* packet replied from its query replying nodes. According to each *Reply*, the query issuing node maintains a list of locations that contribute in building the initial validity region. As shown in Fig. 12, the number of *Reply* packets is not changed because the number of query replying nodes is fixed and the collection of *Reply* packets is a one-time operation. The mobility of POIs does not impact on the number of *Reply* packets. Whenever the query replying node detects an additional queried POI moved into or the existing queried POI moved out of its detection range, it generates and replies a *Reply'* containing its updated location to the query issuing node for a possible update of validity region. Once the query issuing node receives the *Reply'*, it decides whether the updated location can contribute in updating the validity region. As the number of mobile POIs increases, the query replying node can frequently detect the change of queried POI and send the *Reply'*. Thus, the number of *Reply* packets increases as shown in Fig. 12.

V. Discussion

In this section, we identify two real-world systems that can be further extended. We discuss a pre-evacuation system and a privacy conservation system deployed in different infrastructure-based networks in order to see the full potential of the proposed scheme. We also discuss the impact of mobility model on the region-based query operation followed by additional research issues for future work.

A. Region-Sensitive Evacuation Query and Traffic Reduction

Ubiquitous natural disasters cause tremendous damage of property and loss of human life. For example, due to the huge impact of hurricanes, a mandatory evacuation conducted before hurricanes land is essential. Since hurricanes can be predicted well in advance in the sense of path and category, casualties caused by them could be reduced through efficient evacuation planning and dissemination. During the evacuation period, vehicles frequently send queries for update sensitive evacuation data items to an evacuation server or adjacent vehicles through vehicle-to-infrastructure or vehicle-to-vehicle communication, respectively, in a VANET. Vehicles can also broadcast their recently received evacuation data items from the server to its adjacent vehicles for fast dissemination through vehicle-to-vehicle communication. However, blind queries and broadcasts followed by a series of unconditional forwarding operations are inefficient because they can cause redundant retransmissions and packet contentions and collisions. Since wireless infrastructure networks are often bandwidth and resource limited, they can quickly be saturated by bulk evacuation traffic and this can reduce network throughput and increase end-to-end latency.

In light of this, the proposed region-based querying scheme can reduce the evacuation traffic by reducing the number of redundant evacuation queries without degrading the freshness of evacuation query result. The basic idea is that the evacuation server assigns a set of locations forming a validity region, where an evacuation query result is valid as far as a query issuing vehicle is located within the region. When the query issuing vehicle moves and becomes closer to the boundary of validity region, it has a higher probability of reissuing a new evacuation query to fresh its evacuation query result and rebuild the validity region. Thus, the query issuing vehicle does not need to repetitively send the same evacuation query to the evacuation server. Note that the query issuing vehicle does not blindly wait until it moves out of the validity region, but it can proactively update its validity region by reissuing an evacuation query.

B. Region-Based Location Privacy Conservation

When a user requests a location-based POI, it sends a spatial query piggybacked with its current location to an untrusted LBS server. This indicates that the user implicitly has agreed to share its location with the server. In fact, more than 65% users reveal or share their current location with a weather app [20]. Since the LBS server can collect and store location queries in its local storage, it can track and analyze users' location data and infer a certain degree of private information, such as users' home locations, mobility patterns, lifestyle, or health conditions. An adversary can compromise the server to misuse sensitive location data. The LBS server can even release location data to the third parties for mobile marketing without the consensus of users.

When a user sends a query to the LBS server, it is focused on how to efficiently generate a set of dummy locations or a cloaking area to hide the exact location of a user from the LBS server. Upon receiving the queried result from the LBS server, the user answers the query and caches the queried result in its local storage. Due to the mobility, however, the user may not know whether the cached query result is still valid to answer the following query. Thus, the user may send more queries to the LBS server and this can increase the probability of revealing its current location or mobility pattern. In order to reduce the number of queries, prior region-based querying approaches [11], [21]–[23] could implicitly be deployed but it is not explicitly integrated with the operation of dummy location or cloaking area generation. The proposed elastic validity region can reduce the number of spatial queries and the contributing nodes of validity region can be dummy users in order not to provide the server with the exact location of a user.

C. Mobility Model Versus Region Based Querying

In this paper, we deploy a simple and popular mobility model, the random waypoint mobility, to clearly see the impact of elastic validity region on the query performance. If the mobility model changes, such as Levy-walk, clustered mobility model [24], or platoon group mobility model, the MQry would be affected and its query performance could also be changed. For example, when a query issuing node moves out of the current validity region, it regenerates a new query to update the validity region. The frequency of query regeneration for updating the validity region may vary depending on the mobility model and thus, the operation or policy of updating the validity region can be changed. A mobility-aware approach or a fine-grained mobility management can be applied to the query operation, in which each node needs to maintain or predict the mobility of other nodes, but a nonnegligible communication overhead can be incurred in MANETs.

In addition, multiple aspects can be incorporated to extend the proposed query operation. Channel error or packet loss and its corresponding countermeasure mechanism can be embedded for a region-based reliable query operation. To further reduce the query traffic, intermediate nodes located between a query issuing node and its query replying node(s) can judiciously decide whether to forward incoming packets immediately or after a certain delay not to repeatedly forward redundant packets for the same POI.

VI. Concluding Remarks

In this paper, we investigated the elastic validity region and its corresponding query processing scheme to efficiently query mobile POIs in MANETs. A query replying node that contributes in building an initial validity region judiciously switches its monitor operation to observe any change of the queried POI and interacts with a query issuing node. The query issuing node adjusts the current validity region to reduce the number of queries without hurting the freshness of queried result based on the updated query replies. We clearly described and depicted the query operations of both query issuing and replying node sides using FSMs. We conducted extensive simulation-based experiments and performance comparison with four competitive schemes. The simulation results show that the proposed approach can adaptively shrink or extend the size of validity region and can be a viable approach in MANETs with mobile POIs.

References

[1] B. Zheng and D. L. Lee, "Semantic caching in location-dependent query processing," in *Proc. Int. Symp. Spatial Temporal Databases*, 2001, pp. 97–113.

[2] F. Liu, K. Hua, and F. Xie, "On reducing communication cost for distributed moving query monitoring systems," in *Proc. Mobile Data Manage.*, 2008, pp. 156–164.

[3] B. Bamba, L. Liu, A. Iyengar, and S. Y. Philip, "Distributed processing of spatial alarms: A safe region-based approach," in *Proc. 29th IEEE Int. Conf. Distrib. Comput. Syst.*, 2009, pp. 207–214.

[4] H. Cho, S. Kwon, and T. Chung, "A safe exit algorithm for continuous nearest neighbor monitoring in road networks," *Mobile Inf. Syst.*, vol. 9, pp. 37–53, 2013.

[5] A.-K. Haidar, D. Taniar, J. Betts, and S. Alamri, "On finding safe regions for moving range queries," *Math. Comput. Model.*, vol. 58, no. 5, pp. 1449–1458, 2013.

[6] Y. Sasaki, T. Hara, and S. Nishio, "Two-phase top-k query processing in mobile ad hoc networks," in *Proc. Int. Conf. Netw.-Based Inf. Syst.*, 2011, pp. 42–49.

[7] D. Amagata, Y. Sasaki, T. Hara, and S. Nishio, "A robust routing method for top-k queries in mobile ad hoc networks," in *Proc. Mobile Data Manage.*, 2013, pp. 251–256.

[8] D. Amagata, T. Hara, Y. Sasaki, and S. Nishio, "Efficient cluster-based top-k query routing with data replication in MANETs," *Soft Comput.*, vol. 21, pp. 1–18, 2015.

[9] Y. Komai, Y. Sasaki, T. Hara, and S. Nishio, "*k*NN query processing methods in mobile ad hoc networks," *IEEE Trans. Mobile Comput.*, vol. 13, no. 5, pp. 1090–1103, May 2014.

[10] B. Jung, S. Lim, J. Chae, and C. Pu, "Validity region sensitive query processing strategies in mobile ad hoc networks," in *Proc. IEEE Mil. Commun. Conf.*, 2016, pp. 1022–1027.

[11] B. Jung, S. Lim, J. Chae, and C. Pu, "VRSense: Validity region sensitive query processing strategies for static and mobile point-of-interests in MANETs," *Comput. Commun.*, vol. 116, no. 2018, pp. 132–146, 2018.

[12] *OMNeT++*. [Online]. Available: http://www.omnetpp.org/

[13] T. Do, K. Hua, and C. Lin, "ExtRange: Continuous moving range queries in mobile peer-to-peer networks," in *Proc. Mobile Data Manage.*, 2009, pp. 317–322.

[14] C. Lai, Z. F. Akbar, C. Liu, V. Ta, and L. Wang, "Distributed continuous range-skyline query monitoring over the Internet of mobile Things," *IEEE Internet Things J.*, to be published, doi: 10.1109/JIOT.2019.2909393.

[15] Y. Sasaki, T. Hara, and Y. Ishikawa, "Top-k query processing with replication strategy in mobile ad hoc networks," in *Proc. Mobile Data Manage.*, 2018, pp. 217–216.

[16] Y. Ohsawa, H. Htoo, and T. Win, "Safe-region generation methods for continuous trip route planning queries," *Informatica*, vol. 28, no. 1, pp. 131–154, 2017.

[17] S. Ni, Y. Tseng, Y. Chen, and J. Sheu, "The broadcast storm problem in a mobile ad hoc network," in *Proc. ACM MobiCom*, 1999, pp. 151–162.

[18] R. L. Graham, "An efficient algorithm for determining the convex hull of a finite planar set," *Inf. Process. Lett.*, vol. 1, no. 4, pp. 132–133, 1972.

[19] D. B. Johnson and D. A. Maltz, "Dynamic source routing in ad hoc wireless networks," in *Mobile Computing*. Norwell, MA, USA: Kluwer, 1996, pp. 153–181.

[20] *Location Based Services: Expected Trends and Technological Advancements*, Jun. 2017, Accessed: Dec. 2018. [Online]. Available: http://geoawesomeness.com/expected-trends-technological-advancements-location-based-services/

[21] J. Zhang, M. Zhu, D. Papadias, Y. Tao, and D. L. Lee, "Location-based spatial queries," in *Proc. ACM SIGMOD*, 2003, pp. 443–454.

[22] H. Hu, J. Xu, and D. L. Lee, "A generic framework for monitoring continuous spatial queries over moving objects," in *Proc. ACM SIGMOD*, 2005, pp. 479–490.

[23] D. Yung, M. L. Yiu, and E. Lo, "A safe-exit approach for efficient network-based moving range queries," *Data Knowl. Eng.*, vol. 72, pp. 126–147, 2012.

[24] S. Lim, C. Yu, and C. R. Das, "A realistic mobility model for wireless networks of scale-free node connectivity," *Int. J. Mobile Commun.*, vol. 8, no. 3, pp. 351–369, 2010.