Validity Region Sensitive Query Processing Strategies in Mobile Ad Hoc Networks

Byungkwan Jung[†] Sunho Lim[†] Jinseok Chae[§] Cong Pu[†]
[†]T[2]WISTOR: TTU Wireless Mobile Networking Laboratory
[†]Dept. of Computer Science, Texas Tech University, Lubbock, TX 79409, {byung.jung, sunho.lim, cong.pu}@ttu.edu
[§]Dept. of Computer Science and Engineering, Incheon National University, Incheon 22012, Korea, jschae@inu.ac.kr

***Abstract*—Due to the lack of centralized coordination and time-varying network topologies, designing an efficient query processing scheme is admittedly challenging in mobile ad hoc networks (MANETs). Each node often broadcasts a query, retrieves its queried data item, and periodically checks its freshness but a non-negligible communication overhead may incur. In light of this, we propose a set of query processing strategies by deploying a validity region to efficiently minimize the query traffic and update the freshness of the queried data. We first investigate simple rectangle and convex hull based validity regions and their corresponding query processing schemes, and then extend them by adaptively combining both techniques and considering an opportunistic overhearing. We conduct extensive simulation experiments using the OMNeT++ simulation package for performance comparison and analysis, in which an infrastructure-based query processing approach is also adjusted to be implemented in MANETs. The simulation results indicate that our convex hull based adaptive query processing approaches can reduce the number of queries but it increases the time spent in the validity region.**

***Index Terms*—Mobile ad hoc networks, query processing, validity region**

I. Introduction

A set of mobile devices (later nodes) cooperatively communicate among themselves directly or indirectly via multi-hop relays without a wired infrastructure in mobile ad hoc networks (MANETs), where nodes are equipped with sensing, computing, and communicating capabilities. MANETs have been applied to diverse applications in civilian and military environments. As a part of recently emerging Internet of Things (IoTs) [1], where a myriad of multi-scale sensors and heterogeneous nodes are seamlessly blended and connected with or without a wired infrastructure, MANETs will play an important role in realizing a ubiquitous computing and communication in various application domains, e.g., personal and home, enterprise, utilities, and mobile.

Due to the time-varying network topologies and limited amount of battery energy, however, query processing is challenging in MANETs. When a node broadcasts a query to collect the information from the query object (e.g., point of interests, POI), it may receive more than one query reply if multiple nodes answer the query. Unlike an infrastructure-based network, where a query is directly sent to and responded from a centralized server, the query is often flooded to the rest of the nodes through multi-hop relays. Thus, a blind query operation followed by a series of unconditional forwarding operations is inefficient and even harmful because of redundant query retransmissions, contentions, and collisions. This will consume a significant amount of battery energy because wireless communication could be responsible for more than half of the total energy consumption [2]. In addition, when a node intends to reply a unicast query result, unlike a wired network, all one-hop neighbor nodes can still overhear the query result, as if it is a broadcast packet [3]. Thus, it is essential to reduce the number of queries broadcasted without degrading the validity of query in MANETs.

To address these constraints, we propose four query processing strategies based on a *validity region* to reduce the number of queries broadcasted. A validity region is defined as an area where a query result remains the same as long as a query-issuing node is located within the area. Our contribution has two parts:

- We first analyze time-sensitive and -insensitive query types and their corresponding query operations, and investigate a validity region based query processing approach to efficiently minimize the query traffic and update the freshness of queried data.
- Second, we propose basic rectangle (*Rect*) and convex hull (*Conv*) based validity regions and their corresponding query processing strategies, and extend them by adaptively combining both techniques (*Adapt*) and considering an opportunistic overhearing (*Adapt:Ovhr*). We also modify an infrastructure-based query processing scheme to work in MANETs for the purpose of performance comparisons, called *Greedy*.

We conduct extensive simulation experiments using the OMNeT++ [4] for performance comparison and analysis in terms of the number of queries and time spent in the validity region. The simulation results indicate that the convex hull based approaches, Adapt and Adapt:Ovhr, can reduce the number of queries but they can increase the time spent in the validity region during the query processing operations.

The rest of paper is organized as follows. The prior query processing approaches are reviewed and analyzed in Section II. Both query types and validity region based query processing strategies are presented in Section III. Section IV is devoted to performance evaluation and presents simulation results and their analyses. Finally, we conclude the paper with future work in Section V.

II. Related Work

In this section, we categorize prior approaches in terms of top-k query, k nearest neighbor (kNN) query, and location-based spatial query operations.

First, a top-k query is to acquire the k number of data items that is ordered by the score of a target attribute. A query issuing node naively floods a query to entire network, and each node replies its data items that are locally evaluated and scored. Then the query issuing node selects the k highest scored data items from the replied data items. In [5], a two-phase top-k query is proposed to reduce the replied data traffic, in which the query issuing node floods a query, collects the score information of data items, and sets the k-th highest score as a threshold in the search phase. In the data collection phase, the query issuing node floods the query piggybacked with the threshold, and each node replies its corresponding data items scored higher or equal to the threshold. Each node maintains a routing table to further reduce the traffic of flooded queries and replied data items [6]. Since the routing table contains the ranking of scores of data items stored in the network, the query can be forwarded to the limited number of candidate nodes for replies.

Second, a k nearest neighbor (kNN) query is to search data items stored at the k nearest nodes located adjacent to a query point, where a query issuing node transmits a kNN query to. In [7], [8], [9], several kNN query processing strategies are proposed either with a tree-based search structure or without structure in wireless sensor networks (WSNs), where each node is aware of its location and neighbor nodes. The basic idea is that a search space is partitioned into multiple sub areas (i.e., sectors), where each local coordinator node collects and transmits partial data items or results to the query issuing node. In [10], variants of prior kNN query processing strategies (Explosion and Spiral) are proposed in MANETs, in which a geo-routing is deployed to forward the kNN query to the node located nearest to the query point.

Third, a location-based spatial query is to attain the information of static query objects (e.g., point of interests (POIs)) located around the mobile nodes. A query issuing node continuously sends a query to a server directly after its current location is updated, but this can lead to high network overhead. To avoid redundant queries, the server calculates and returns a validity region to the query issuing node. Then the node is able to decide whether to issue an additional query by verifying whether its current location is still within the validity region. A validity region based spatial query approach [11] and its following variants have been proposed in diverse networks. In particular, spatial query strategies have been proposed depending on the mobility of query issuing node and/or POIs in vehicular ad hoc networks (VANETs) [12].

In summary, prior search based query processing approaches may not be directly applied to MANETs, where the network topology varies because of the node mobility. Due to the lack of centralized coordination, prior validity region based querying approaches with a centralized server may not be practical in infrastructure-less MANETs.

III. The Proposed Query Processing Strategies

In this section, we first analyze a query type and its corresponding query operation and then propose two basic validity region based query processing strategies in MANETs. The proposed strategies are further enhanced by considering the update of validity region.

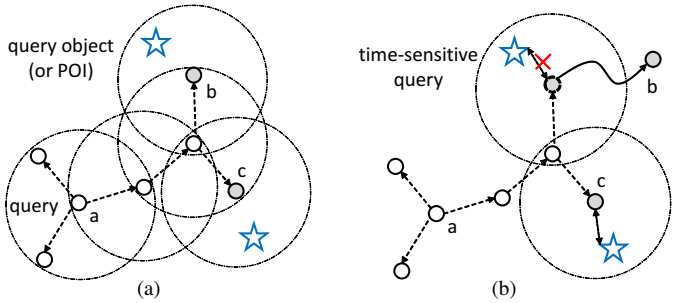


Fig. 1. An example of query operation in a MANET. Here, a query-replying node is shaded and a query object (or point-of-interest (POI)) is marked by a star. A direction of query packet propagated and the communication range are represented by a dashed arrow and a dash-dotted line, respectively.

A. To Query or Not To Query

When a node generates a location-based query (later in short, query), it broadcasts the query to collect the information from the query objects (or event) based on the geographical location where it currently resides. In Subfig. 1(a), a node (e.g., n_a) broadcasts a query and two nodes (e.g., n_b and n_c) closely located to query objects reply their query result. Since the information from the query objects may be time-varying and affect the validity of query result, we investigate two different query types: time-sensitive and time-insensitive queries.

First, a time-sensitive query targets a query object that contains a time-varying information as well as affects the validity of prior query result sent to a query-issuing node. The query-issuing node or query-replying node needs to periodically send a query or reply the updated query result, respectively to maintain the validity of query result. In Subfig. 1(b), both n_b and n_c periodically interact or measure the query object and forward their query result to n_a. For example, a set of wireless sensors are deployed in a wildfire monitoring system, where each sensor periodically senses the current temperature and sends it to a sink through multi-hop relays. Then the sink can actively monitor the specific region where nodes reply higher temperature, suspect a wildfire, and send back additional queries. A good deal of research effort has been made how to efficiently search the information from the query objects while minimizing the network traffic in wireless multi-hop networks, where k nearest neighbors (kNNs) or top-k query processing techniques [10], [5] are often deployed. Note that if a query-replying node moves away from a query object, then the query-issuing node needs to broadcast a query again to search another query-replying node to maintain the validity of query result. In Subfig. 1(b), n_a broadcasts a query again when it does not receive any further query result from n_b because of its mobility.

Second, a time-insensitive query targets a query object that contains a static information and does not change in a short period of time, such as geographical data. Since a query result depends on the location, the mobility of query-replying node does not affect the validity of query result. Thus, the query-

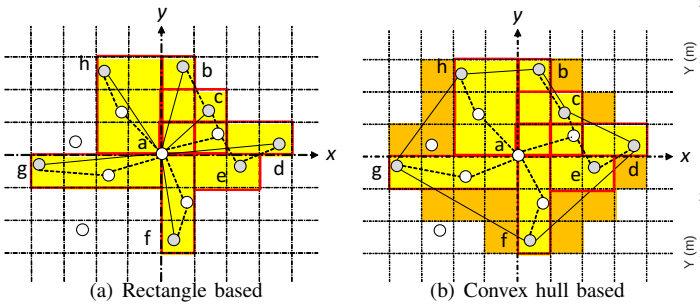


Fig. 2. A rectangle based validity region consists of cells (marked by yellow) located within the rectangles (marked by red) based on the locations of query-replying nodes (marked by gray circle). However, a convex hull based validity region has more cells (marked by orange for additional cells) than that of the rectangle based approach.

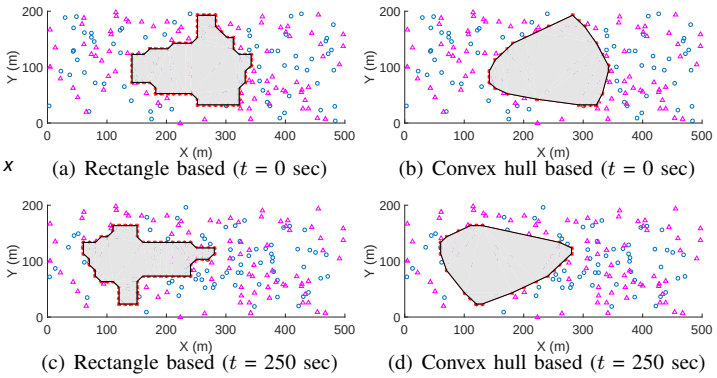


Fig. 3. A snapshot of validity region. We experiment with 100 nodes (marked by a circle) and POIs (marked by a triangle) in a network, where nodes are uniformly distributed and move according to the random waypoint mobility, 1.0 m/sec. We show a set of validity regions at the initial (t = 0 sec) and after t = 250 sec using rectangle and convex hull based approaches, respectively.

issuing node sends a query again only when it moves away from the query object and needs a new query result. In Subfig. 1(b), although n_b moves away from a query object, n_a does not broadcast an additional query. In this paper, we consider a time-insensitive query for a static information and present its corresponding query processing techniques in MANETs. A tracking of moving query object (or event) is out of scope of this paper.

B. Validity Region Sensitive Query Processing

Although the mobility of query-replying node does not affect the validity of query result in time-insensitive query, it becomes an issue when a query-issuing node needs to rebroadcast the same query. A periodic query rebroadcast may not be a best solution because of its redundancy that can lead to the flooding and significant energy consumption. In light of this, we deploy a validity region based query processing approach. The validity region is an area where a query result remains the same as long as a query-issuing node is located within the area. A network is virtually divided into a set of grids, where each square region is called *cell* and has the same size. Since each node is equipped with an on-board GPS, it knows the current location and corresponding cell. Thus, the query-issuing node can rebroadcast the query only when it moves out of the validity region.

In this paper, we propose how to approximate a validity region and its corresponding query processing to minimize the number of queries broadcasted: (i) rectangle based (*Rect*) and (ii) convex hull based (*Conv*) approaches. First, the basic idea of Rect is to set a validity region based on the locations of query-issuing and -replying nodes. When a node receives a query broadcasted from a query-issuing node, it replies a query result piggybacked with its current location. Whenever the query-issuing node receives a query result, it sets a rectangle having a diagonal line connecting from the current location of query-issuing node and to the location of query-replying node. All the corresponding cells that cover the rectangle are considered as a part of validity region, and the query-issuing nodes only store the list of cell *ids*. For example, suppose n_a generates a query, broadcasts it, and receives its query results from n_b to n_h as shown in Subfig. 2(a). Then n_a constructs a set of rectangles and finds the corresponding cells that cover the rectangles. Note that the rectangles constructed based on the locations of query-replying nodes, n_b and n_c, can have the overlapped cells. The Rect is simple but its overall shape of validity region is an irregular polygon that often contains either any length of sides or size of angles. In Subfigs. 3(a) and (c), we snapshot the shapes of validity region when a query-issuing node approximates. This implies that the query-issuing node can easily be out of the validity region while it is moving and thus, it may need to frequently rebroadcast a query.

Second, in the Conv, we deploy the convex hull algorithm to approximate a validity region, where a convex hull is the smallest convex polygon containing a given set of points in two-dimensional area [13]. Similar to the rectangle based approach, when a node receives a query and finds a queried data, it replies a query result piggybacked with its current location. After receiving all query results, a query-issuing node builds a polygon as a validity region based on the locations of query-replying nodes. Note that we use the Graham-Scan based method [13] to construct a validity region. We initially choose a location with the minimum x- and y- coordinates, sort the rest of locations based on the angle to the location in counterclockwise order. Then the initially chosen location is connected with a location with the minimum angle. We keep incrementally connecting the locations only if they are located to the counterclockwise of the line connecting previous two locations. For example, n_a builds a polygon with the locations of query-replying nodes (n_b to n_h) using the convex hull algorithm as shown in Subfig. 2(b). All the corresponding cells that cover the polygon are considered as the validity region. The Conv is also simple but its validity region includes more cells compared to that of the Rect. More importantly, the overall shape of validity region is less irregular and closer to a circle as shown in Subfigs. 3(b) and (d). This may reduce the number of queries generated and broadcasted while the query-issuing node is moving. The major operation of Conv is summarized in Fig. 4.

In summary, when a query-issuing node generates a query,

Notations:
• $Rpy[nid', POI_{data}, l_{nid}(x,y)]$: A reply packet forwarded back to the query-issuing node (nid') with the queried data (POI_{data}) and the current location of query-replying node (nid), $l_{nid}(x,y)$.
• k, k', k'': An index incremented by one and initially set by zero.
• $B_{loc,k}$: A buffer to save the received replying node's location, $l_{nid}(x,y)$, into the k^{th} slot.
• $R_{loc,k'}$: A set of locations forming a validity region.
• $G_{k''}$: A set of virtual grid ids corresponding with each location of $R_{loc,k'}$.
◇ When a query-issuing node, n_i, receives a reply packet from a query-replying node, n_j,
if $l_j(x,y) \notin B_{loc}$
Save $l_j(x,y)$ into $B_{loc,k}$;
Sort the $B_{loc,k}$ based on the angle in counterclockwise order;
Initiate the location with the minimum x- and y-coordinates;
for $\forall$ B_{loc} **do** /* Graham-Scan based [13] */
if $B_{loc,k}$ is located to the counterclockwise of the line connecting from $B_{loc,k-2}$ to $B_{loc,k-1}$
Add $N_{loc,k}$ to $R_{loc,k'}$;
Replace $B_{loc,k}$ with $B_{loc,k-1}$;
for $\forall$ R_{loc} **do**
Map $R_{loc,k'}$ into a virtual grid id and save it into $G_{k''}$;

Fig. 4. The pseudo code of convex hull based validity region.

Notations:
• $Rpy[nid', POI_{data}, l_{nid}(x,y)]$: Defined before.
• $Qry[nid, POI_{type}, h]$: A query packet flooded from a query-issuing node (n_{nid}) with a POI type (POI_{type}) and the number of hops (h).
◇ When a node, n_i, generates a query,
Broadcast a query packet, $Qry[i, POI_{type}, 0]$;
◇ When n_j receives a query packet from n_i,
if $++hop > h_{max}$
Drop the query packet and exit;
if POI_{data} is found
Send a reply packet, $Rpy[i, POI_{data}, l_j(x,y)]$, to n_i;
Rebroadcast the query packet;
◇ When n_i receives a reply packet from n_j,
Build a validity region with the received $l_j(x,y)$ based on the Rect or Conv; /* e.g., Fig. 4 */
◇ When n_i is located in out of bound of its validity region,
Broadcast a query packet again, $Qry[i, POI_{type}, 0]$;

Fig. 5. The pseudo code of proposed query processing.

the query is limited to be flooded upto the maximum number of hops (h_{max}). When a node closely located to the POI receives the query, it sends a reply directly or indirectly via multi-hop relays. Upon receiving the reply, the query-issuing node initiates to build a validity region based on the locations of replying nodes using the Rect or Conv. If the query-issuing node is out of bound of the validity region due to its mobility, it generates a query again. The major operation of query processing is also summarized in Fig. 5.

C. Enhancements

In the Conv, a query-issuing node may not create a validity region because the number of replies is not enough. Due to the network condition, the node may experience a delay even before creating the validity region. Since the node does not know when and how many replies will arrive, it may blindly wait for the replies which are in fact dropped during the transmission. A query can also be dropped during the transmission. Thus, we propose an adaptive approach by combining both Rect and Conv, called as *Adapt*, to promptly

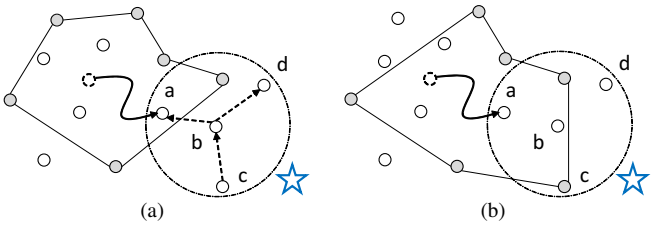


Fig. 6. A validity region is adaptively updated by opportunistically overhearing a reply forwarded to another querying-issuing node.

build the validity region. In the Adapt, when a query-issuing node receives the first reply, it immediately builds an initial validity region by conducting the Rect. Then the node keeps extending the validity region upon receiving the following replies while it receives enough number of replies to build a validity region based on the Conv.

Note that the proposed three approaches (Rect, Conv, and Adapt) use a static validity region and its shape does not change once a query-issuing node build the validity region. However, the node will ultimately move out of the bound of validity region because of its mobility and thus, it will generate a query again. In this paper, we also propose a scheme to update the validity region, called *Adapt:Ovhr*, to reduce the number of queries. The basic idea is to update the current validate region by opportunistically utilizing the overheard on-flying replies. In the Adapt:Ovhr, the node initially builds a validity region based on the Adapt, in which the validity region can be shaped combining by the Rect or Conv depending on the number of received replies. When the node overhears a reply forwarded back to another query-issuing node, if the reply contains the same POI what the node queried, it updates the current validity region by including the location of the query replying node. Here, the node filters the location and only considers it that is out of the current validity region. The benefit of this Adapt:Ovhr is that the node can update the validity region toward the way where it moves, stay more time within the validity region, and save the number of queries.

For example, a query-issuing node (e.g., n_a) has built a validity region and moves to a new location in Subfig. 6(a). Another query-issuing node (e.g., n_d) generates a query and a node (e.g., n_c) closely located to a POI sends a reply back to n_d. An intermediate node (e.g., n_b) forwards the received reply from n_c to n_d. n_a can overhear and check the reply whether it contains the same type of POI what n_a queried before. If so, n_a updates its validity region by including the location of query replying node, n_c, in Subfig. 6(b). In addition, as n_a moves, the location that becomes far away will be removed from the validity region.

D. Variant

In addition, we modify a safe exit algorithm [12] deployed in an infrastructure-based road network to work in MANETs, called *Greedy*, for the purpose of performance comparison in this paper. Here, the safe exit algorithm uses the Euclidean distance between a query-issuing node and its queried POI to determine when to re-generate a query. In the Greedy, we assume a virtual centralized server that is aware of the network

topology in a real-time manner. When a query-issuing node generates a query, it sends the query to the server directly. The server ideally replies all the locations of neighbor nodes located within the maximum distance of three hops (d_{max}, i.e., $h_{max} \times r$) from the node. Here, r is the radio transmission range. Then the node keeps recalculating the distances to the locations of query-replying nodes whenever it moves to see whether it is still in the validity region. Unlike the Greedy, the proposed strategies use a list of virtual cells corresponding to the validity region. Thus, a query-issuing node only remembers a list of cell ids included in its validity region, compares the current cell whether it is in the list, and determines when to re-generate a query.

IV. Performance Evaluation

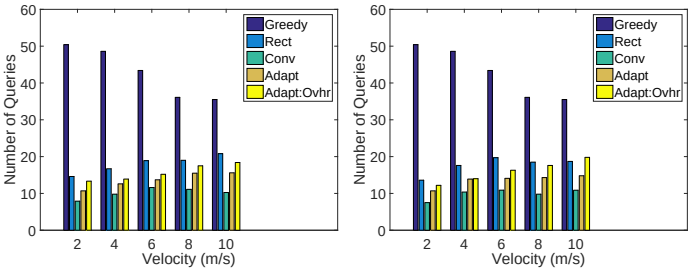
A. Simulation Testbed

We developed a customized discrete-event driven simulator by using the OMNeT++ [4]. A 200×500 m^2 rectangle network is deployed, where 100 nodes are uniformly distributed. Each node is equipped with an on-board GPS receiver and is aware of the current location. The radio transmission range (r) is assumed to be 50 m, and the two-ray ground propagation channel is assumed with a data rate of 2 Mbps. The POIs are uniformly distributed in the network, where the total number of POIs and POI types (POI_{type}) are set to 200 and 20, respectively. Each node generates a query with the maximum number of POI types (Q_{type}) for POIs, i.e., Q_{type} is set to three or eight. If Q_{type} is three, a query-issuing node queries with three types of POIs in the network, where any node who can find a POI with any one of three types can reply. The more Q_{type} is set, the more queried POIs the node receives. The query is limitedly flooded upto three hops (h_{max}) to reduce the network traffic and its queried data size is set to 16 Kbits. The random waypoint mobility model [14] is used to simulate the node mobility with a node speed from 2 to 10 m/s and a pause time of 0 to 50 seconds.

B. Simulation Results

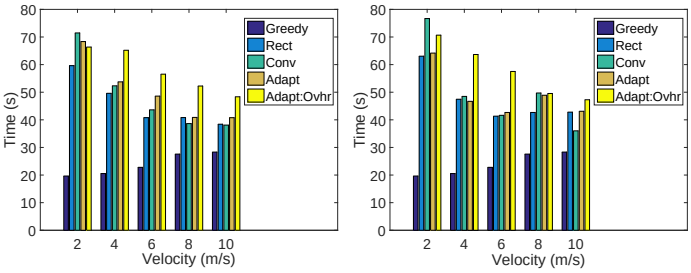
In this section, we evaluate the performance of proposed query processing strategies in terms of the number of queries and time period spent in the validity region as a function of number of POI types and node velocity. We compare five different strategies: rectangle based (*Rect*), convex hull based (*Conv*), adaptive (*Adapt*), adaptive with overhearing (*Adapt:Ovhr*), and virtual centralized server based (*Greedy*).

1) Number of Queries: We measure the number of queries by changing the number of query types and velocities in Fig. 7. In Subfig. 7(a), the Greedy shows the highest number of queries compared to other schemes. In the Greedy, a query-issuing node receives the replies from more number of neighbor nodes located within d_{max} in low velocity, compared to the case in high velocity. Whenever the query-issuing node moves, it re-computes the distance to each query replying nodes and re-generates a query if there is any distance over d_{max}. Since the query-issuing node computes the distance to more number of query replying nodes in low velocity, it



(a) Q_{type} = 3 (b) Q_{type} = 8

Fig. 7. The number of queries generated against the number of POI types and node mobility in terms of velocity.

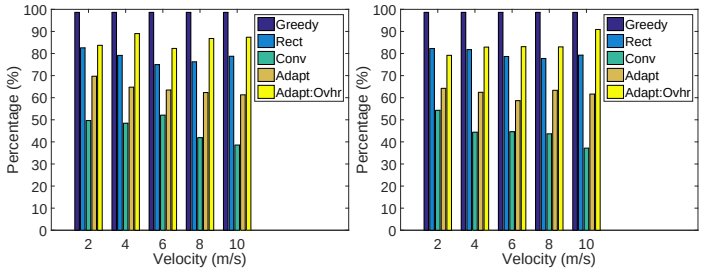


(a) Q_{type} = 3 (b) Q_{type} = 8

Fig. 8. The time period spent within the validity region against the number of POI types and node mobility in terms of velocity.

is frequently to be over d_{max} while it moves and generates more number of queries compared to the case in high velocity. Thus, the number of queries reduces as the velocity increases. Unlike the Greedy, the Rect, Conv, Adapt, and Adapt:Ovhr use a validity region based on the virtual cell and they are not sensitive to the distance to individual query replying nodes. The Rect shows higher number of queries than the rest of three schemes because it creates a validity region with an irregular polygon. A query-issuing node may easily move out of the bound of validity region. However, the Conv can build a validity region with less irregular polygon than that of the Rect and shows the lowest number of queries. In Subfig. 7(b), with higher Q_{type}, a query-issuing node can receive more number of query replies. The number of queries slightly reduces in the Rect, Conv, Adapt, and Adapt:Ovhr for entire velocities.

2) Average Time Spent in Validity Region: We measure the time period how long a query-issuing node stays within its validity region in Fig. 8. In Subfig. 8(a), the Conv shows the longest time period in low velocity, i.e., 2 m/s. Since a query-issuing node receives the replies from more number of neighbor nodes in low velocity, it can create a validity region with less irregular polygon compared to that of the Rect. Both Adapt and Adapt:Ovhr also show the similar performance because they are based on the Conv. As the velocity increases, the Adapt:Ovhr shows the longest time period because a query-issuing node adaptively expands its validity region toward the way where it moves by opportunistically overhearing on-flying replies. Thus, the query-issuing node can stay in the validity region for longer period. With higher Q_{type} in Subfig. 8(b), the time period increases for entire velocities because a query-issuing node receives more replies and it can create a validity region with regular polygon, e.g., closer to a circle. Note that the Greedy shows the shortest time period for entire velocities.



(a) Q_{type} = 3 (b) Q_{type} = 8

Fig. 9. The percentage of time period spent within the validity region against the number of POI types and node mobility in terms of velocity.

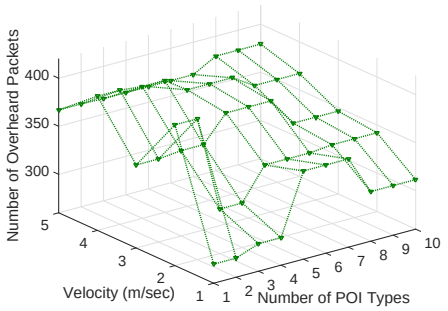


Fig. 10. The number of overheard packets against the number of POI types and node mobility in terms of velocity.

Since the distance from a query-issuing node to any query-replying node frequently is over d_{max}, the query-issuing node stays less time period in its validity region and generates more queries (see also Fig. 7).

3) Percent of Time Spent in Validity Region: We measure the percentage of time period how long a query-issuing node stays in its validity region for entire simulation time in Fig. 9. The Greedy shows the highest percentage for entire velocities and Q_{type} in both Subfigs. 9(a) and (b). In the Greedy, although a query-issuing node stays in its validity region for a short period time (see also Fig. 8), it re-generates a query and quickly builds another validity region through the virtual server. This can keep the query-issuing node staying in the validity region almost always. In Subfigs. 9(a) and (b), the Adapt:Ovhr shows the second highest percentage for entire velocities and Q_{type} because the validity region is adaptively extended based on the mobility of query-issuing node. Note that the Rect shows higher percentage than that of the Conv and Adapt:Conv. In the Rect, a query-issuing node can begin to create its validity region immediately after receiving the first reply. Then the query-issuing node incrementally expands the validity region whenever it receives any following reply. In the Conv and Adapt:Conv, however, a query-issuing node may experience a delay before creating its validity region because it has to wait to receive at least three replies based on the convex hull algorithm.

In addition, we measure the number of overheard packets by changing the number of POI types and velocities in the Adapt:Ovhr. As shown in Fig. 10, the node mobility helps to overhear more number of replies in the network, but the number of POI types does not affect much to the performance. As the velocity increases, each query-issuing node can opportunistically overhear more on-flying replies and stay longer within the validity region by adaptively extending it.

V. Concluding Remarks

In this paper, we investigated validity region sensitive query processing strategies to efficiently minimize the number of queries and update the freshness of queried data in MANETs. We first proposed basic rectangle and convex hull based validity regions and their corresponding query processing techniques. Then we extended them by combining both rectangle and convex hull techniques and considering an opportunistic overhearing. We also modified the safe exit algorithm to work in MANETs for performance comparison. The simulation results show that the convex hull based Adapt and Adapt:Ovhr have the competitive performance in terms of the number of queries and time period spent in the validity region. The Adapt:Ovhr can query-issuing nodes stay in their validity regions for 80 to 90% of the total simulation time. To see the full potential of the proposed techniques, we are currently investigating the performance impact of a group mobility (e.g. a platoon mobility), diverse query patterns and types (e.g., time-sensitive), and the size of virtual grid in the network.

Acknowledgment

This research was supported in part by International Cooperative Research Grant from Incheon National University (Incheon, Korea) in 2015.

References

[1] J. Gubbi, R. Buyya, S. Marusic, and M. Palaniswami, "Interet of Things (IoT): A Vision, Architectural Elements, and Future Directions," *Future Generation Computer Systems*, vol. 29, pp. 1645–1660, 2013.

[2] R. Kravets and P. Krishnan, "Power Management Techniques for Mobile Communication," in *Proc. ACM MOBICOM*, 1998, pp. 157–168.

[3] S. Lim, C. Yu, and C. R. Das, "RandomCast: An Energy Efficient Communication Scheme for Mobile Ad Hoc Networks," *IEEE Trans. on Mobile Computing*, vol. 8, no. 8, pp. 1039–1051, 2009.

[4] *OMNeT++*, http://www.omnetpp.org/.

[5] Y. Sasaki, T. Hara, and S. Nishio, "Two-Phase Top-k Query Processing in Mobile Ad Hoc Networks," in *Proc. Int'l Conf. on Network-based Information Systems*, 2011, pp. 42–49.

[6] 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 Management*, 2013, pp. 251–256.

[7] Y. Xu, T. Y. Fu, W. C. Lee, and J. Winter, "Processing k Nearest Neighbor Queries in Location-aware Sensor Networks," *Signal Processing*, vol. 87, no. 12, pp. 2861–2881, 2007.

[8] S. H. Wu, K. T. Chuang, C. M. Chen, and M. S. Chen, "Toward the Optimal Itinerary-based KNN Query Processing in Mobile Sensor Networks," *IEEE Trans. Knowledge and Data Engineering*, vol. 20, no. 12, pp. 1655–1668, 2008.

[9] T. Y. Fu, W. C. Peng, and W. C. Lee, "Parallelizing Itinerary-based KNN Query Processing in Wireless Sensor Networks," *IEEE Trans. Knowledge and Data Engineering*, vol. 22, no. 5, pp. 711–729, 2010.

[10] Y. Komai, Y. Sasaki, T. Hara, and S. Nishio, "kNN Query Processing Methods in Mobile Ad Hoc Networks," *IEEE Trans. on Mobile Computing*, vol. 13, no. 5, pp. 1090–1103, 2014.

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

[12] H.-J. Cho, S. J. Kwon, and T.-S. Chung, "A Safe Exit Algorithm for Continuous Nearest Neighbor Monitoring in Road Networks," in *Mobile Information Systems*, 2013, pp. 37–53.

[13] R. L. Graham, "An Efficient Algorith for Determining the Convex Hull of a Finite Planar Set," *Information Processing Letters*, vol. 1, no. 4, pp. 132–133, 1972.

[14] D. B. Johnson and D. A. Maltz, "Dynamic Source Routing in Ad Hoc Wireless Networks," in *Mobile Computing*, Kluwer 1996, pp. 153–181.