# Effect of Unstable Routing in Location-Aware Mobile Ad Hoc Networks on a Geographic DHT Protocol

Dae-Woong Kim, Woo-Ram Park, and Chan-Ik Park

Department of Computer Science and Engineering
Pohang University of Science and Technology
Pohang, Gyungbuk 790-784, Republic of Korea
{woong, wizrampa, cipark}@postech.ac.kr

## Abstract

*Geographic distributed hash table (DHT) protocols assume that the set of $\langle$key, value$\rangle$ pairs, called indexes, should be distributed among nodes. In geographic DHT protocols, the overhead of index redistribution due to node mobility may be high enough to impact the normal lookup operation if each node contains a large number of indexes. In our previous work, we proposed an efficient lookup protocol, called Double Indirect Access (DIA), that dispenses with index redistribution to improve lookup performance. We also evaluated the performance of DIA by using our own simulator which could show only limited simulation results with the assumptions that all nodes are connected and the packets are transferred instantly without delay. In this paper, we implement DIA into the packet-level simulator ns2 in order to evaluate how DIA is affected by the underlying unstable routing in MANET. The metrics of the effectiveness are considered lookup success rate and bandwidth consumption, and the simulation results show that DIA performs better than a conventional geographic DHT protocol.*

## 1 Introduction

Distributed hash tables (DHTs) such as Chord [8], Tapestry [9], and CAN [6] are considered one of the most promising lookup methods for P2P object sharing.

Mobile devices such as cellular phones, MP3 players, PDAs, and PMPs are currently converging. Moreover, they will be equipped with storage devices, localization devices such as GPS, and wireless communication devices such as Wi-Fi. Therefore, sharing of a large number of objects such as music files, pictures, documents, and movie clips in location-aware MANETs will be a potential application.

A geographic DHT uses a rendezvous mechanism based on the geographic hashing mapping rule: It hashes a key into a geographic coordinate, and the $\langle$key, value$\rangle$ pair is stored at the node closest to the location to which its key hashes as shown in Figure 1.

In our previous work [4], we proposed an efficient lookup protocol, called Double Indirect Access (DIA), that dispenses with index redistribution to improve lookup performance. The main idea was to determine the mapping from an index to a node not by the node's position, but by the node's static identifier that is obtained by hashing its MAC address into a geographic coordinate. However, a key lookup request will be routed to some node based on the key's hash value, failing to locate We included another level of indirection to resolve this inconsistency: The node to which a key lookup request has been routed is named as an indirection server, which is responsible for relaying the lookup request to the node storing the corresponding index. In order for an indirection server to find out the correct destination node for the lookup request, it maintains a list of nodes' static identifiers whose values (i.e., geographic coordinates) are close to the location of the indirection server. Moreover, for a fair comparison to DIA, we designed a conventional geographic DHT protocol, named as Single Indirect Access (SIA).

We evaluated, in our previous work, the performance of DIA by using our own simulator written in C language. However, our previous simulator did not include the details of the MAC and physical layer and could show only limited simulation results with the assumptions that all nodes are connected and the packets are transferred instantly without delay.

Due to unstable routing in MANET, two critical operations of DIA, the construction of an overlay network and the corresponding IS-list redistribution, may not be completed properly. In this paper, we implements DIA and SIA

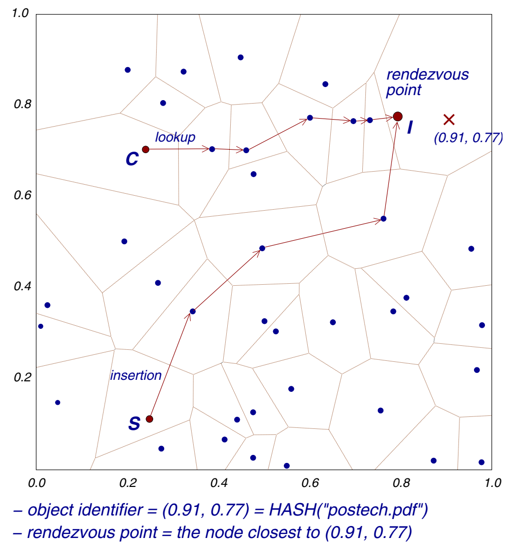

1.0
0.8
0.6
0.4
0.2
0.0
0.2
0.4
0.6
0.8
1.0
rendezvous point
lookup
C
I
(0.91, 0.77)
insertion
S
– object identifier = (0.91, 0.77) = HASH("postech.pdf")
– rendezvous point = the node closest to (0.91, 0.77)

**Figure 1. A rendezvous mechanism based on geographic hashing. A storage node $S$ publishes its local object with the name "postech.pdf" by hashing the name into the identifier (.91, .77) and by inserting the pair $\langle$(.91, .77), S$\rangle$ into the rendezvous point $I$. A client node $C$ hashes "postech.pdf" into (.91, .77) and issues a lookup request for the object, which will be routed to the rendezvous point $I$.**

into the packet-level simulator ns2 version 2.29 in order to evaluate how DIA is affected by the underlying routing in MANET. The metric of the effectiveness is considered the lookup success rate. Simulation results show that DIA is less sensitive to instability of the underlying routing in MANET than SIA under reasonable assumptions of node density, node mobility, and p-id update threshold distance. This is due to its relatively constant packet length and duplicated IS-list entries among neighbors.

Another significant metric is the bandwidth consumed by IS-list redistribution packets in DIA, and index redistribution and handover packets in SIA. As expected, the bandwidth consumption of SIA increases as the number of objects increases, whereas that of DIA is independent of the number of objects.

The remainder of this paper is organized as follows. Section 2 explains P2P object lookup protocols. Section 3 and Section 4 introduce SIA and DIA respectively. Section 5 presents simulations comparing the performance of DIA and SIA. Finally, we conclude this paper with Section 6.

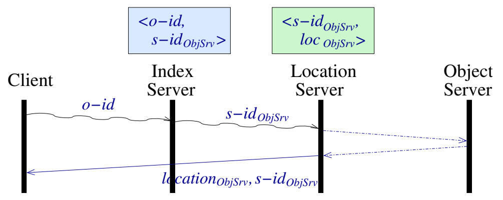


**Figure 2. An object lookup in Single Indirect Access is performed in two steps.**

## 2 P2P Object Lookup Protocol

Each mobile node $p_i$ in a P2P object lookup protocol in a location-aware MANET could have two types of node identifiers in a given two-dimensional region $\mathcal{R}$: (1) The static identifier, called *s-id*, is obtained by hashing the node's MAC address into a geographic coordinate, therefore it is fixed. (2) The physical identifier, called *p-id*, is related to the node's geographic location, therefore it is subject to change due to the node mobility. An object identifier, called *o-id*, of an object $o_i$ is obtained by hashing the object's name to a geographic coordinate.

All nodes in a P2P system are assumed to have the same functions in general, however, each node has a specific role during an operation such as lookup, insert, or retrieval. A *client* of an object is a node that issues a lookup request for the object. An *object server* of an object is a node storing the object. An *index (entry)* of an object is the mapping between its o-id and the s-id of its object server, i.e., $\langle$o-id, s-id$\rangle$, and an *index server* of the object is a node holding the corresponding index entry. A *location entry* of a node is the mapping between its s-id and geographic location, i.e., $\langle$s-id, location$\rangle$, and the *location server* of the node is a node holding the corresponding location entry.

DIA and SIA utilize the Voronoi diagram [5] and its dual structure, the Delaunay triangulation as their topologies. Voronoi diagram (Delaunay triangulation) is used as the overlay network since it is conceptually simple and clear; that is, a node whose identifier is closest to a key is responsible for the key. However, repeated computation of Voronoi diagram can be a severe burden to lightweight mobile devices. The overlay network of two-dimensional torus as in CAN system could be an alternative to Delaunay triangulations.

## 3 Single Indirect Access

For a fair comparison to DIA, we have designed a conventional geographic DHT protocol, named as Single Indirect Access (SIA), by applying two major modifications to an existing protocol GHT [7] as follows: (1) To guarantee that a key lookup will be routed correctly to the node closest to the key, an explicit overlay network is maintained. (2) To eliminate the overhead of object redistribution, an index entry stores the reference to an object, not the object itself.

In SIA, each node has two identifiers, p-id (physical identifier) and s-id (static identifier), and nodes self-organize into their overlay network whose topology is a Delaunay triangulation of the set of their p-ids.

For each point in a key space $\mathcal{R}$, there exists only one index server (and one location server) that is responsible for the point. For an o-id (object identifier) $o \in v(p_i)$, where $v(p_i)$ is the Voronoi polygon of $p_i$, $p_i$ is the index server of the o-id $o$. Similarly, for a s-id $s_j \in v(p_i)$, $p_i$ is the location server of the s-id $s_j$.

Let $P_{p_i}$ be the neighbor list of an index server $p_i$, which is the set of p-ids of the index server itself and its neighbors. To find out its zone, each node $p_i$ keeps track of the Voronoi diagram of $P_{p_i}$. The local version of Voronoi polygon of $p_j$ computed at $p_i$ is denoted by $v_{P_{p_i}}(p_j)$ or $v_P(p_j|p_i)$ and given by

$$\begin{aligned} v_P(p_j|p_i) \quad &= \quad \{\mathbf{x} \in \mathcal{R} \mid ||\mathbf{x} - \mathbf{p}_j|| \leq ||\mathbf{x} - \mathbf{p}_k|| \\ &\qquad \text{for } k \neq j,\ p_j,\ p_k \in P_{p_i}\}. \end{aligned}$$

The corresponding Voronoi diagram is given by

$$V(P_{p_i}) = \{v_P(p_j|p_i) \mid p_j \in P_{p_i}\}.$$

As shown in Figure **??**, the process of object lookup is executed in two steps: (1) When a client issues a lookup request for an object with o-id, the request is routed to the index server whose p-id is closest to the o-id. Then, the index server finds out the s-id of the object server of the object by searching its local indexes and issues a location lookup request to find the location of the object server. (2) The location request is routed to the location server whose p-id is closest to the s-id of the object server. Upon receiving the location request, the location server checks the age of location information of the object server. In case that the location information is not too old to be sufficiently accurate, the location server sends the lookup result that contains the s-id and geographic location of the object server to the client. Otherwise, the location request should be routed to the object server to obtain more accurate location information.

## 3.1 Index Redistribution

In a geographic DHT protocol, the topology changes when the node or its neighbors move. This implies that some ⟨key, value⟩ pairs should be migrated from node to node as illustrated in Figure **??**.

Since nodes move constantly in a MANET, the p-ids of the nodes change constantly. To alleviate the overhead

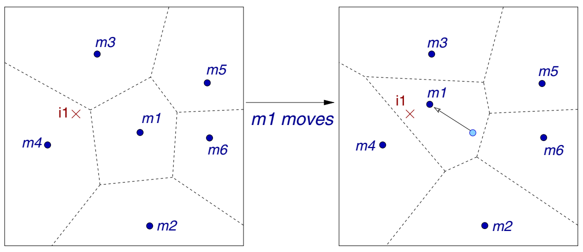


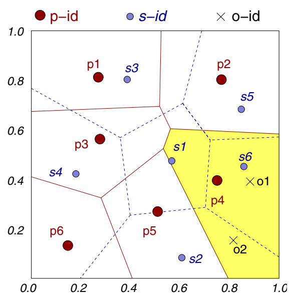
**Figure 3. The topology changes when a node $m1$ moves, and the index $i1$ should be migrated from $m4$ to $m1$.**

of frequently updating Delaunay triangulations, we allow a node to update its p-id only if it moves over a given distance, *p-id update threshold distance*, denoted by $d_{pid}$.

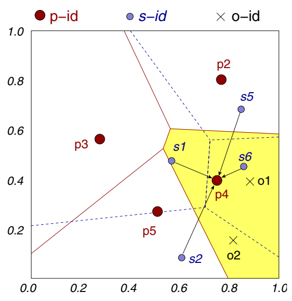
In SIA, the value of $d_{pid}$ has some effects on the overhead of updating the overlay network, the overhead of maintaining the overlay network links, and the efficiency of lookup/insert operations as follows: (1) The larger the $d_{pid}$, the less frequent the update of the overlay network. (2) The larger the $d_{pid}$, the larger the overhead of maintaining the overlay network links. A larger $d_{pid}$ enlarges the physical distance between two overlay neighbors, so that live beacons, which the neighboring nodes exchange with each other to maintain overlay links, will experience longer paths. (3) The larger $d_{pid}$, the larger the overhead of lookup/insert operations. Since a node responsible for the o-id $(o_{ix}, o_{iy})$ is the node whose p-id is closest to the o-id, an lookup/insert message for the o-id is routed to the geographical point $(o_{ix}, o_{iy})$. A larger $d_{pid}$ enlarges the physical distance between the point $(o_{ix}, o_{iy})$ and the node responsible for the point. The optimal value of $d_{pid}$ will minimize the sum of the above three overheads. The in-depth study of this tradeoff is left as future work.

## 3.2 Index Handover

The index redistribution operation of SIA should be completed before the next invocation of topology change and index redistribution. However, a packet delivery cannot be completed instantly and takes some delay. If an index redistribution packet is not transferred in time, then the receiving node may not be responsible for some indexes in that packet any longer, and those indexes should be forwarded repeatedly until they arrive at their new index servers. The above operation is called *index handover* and is distinguished from index redistribution.



(a) Stack of two DHTs



(b) Local view at $p_4$

| p-id | Neighbor list | IS-list |
|---|---|---|
| $p_1$ | $P_{p_1} = \{p_1, p_2, p_3\}$ | $S_{p_1} = \{s_3, s_4\}$ |
| $p_2$ | $P_{p_2} = \{p_2, p_1, p_3, p_4\}$ | $S_{p_2} = \{s_1, s_3, s_5\}$ |
| $p_3$ | $P_{p_3} = \{p_3, p_1, p_2, p_4, p_5, p_6\}$ | $S_{p_3} = \{s_1, s_3, s_4\}$ |
| $p_4$ | $P_{p_4} = \{p_4, p_2, p_3, p_5\}$ | $S_{p_4} = \{s_1, s_2, s_5, s_6\}$ |
| $p_5$ | $P_{p_5} = \{p_5, p_3, p_4, p_6\}$ | $S_{p_5} = \{s_1, s_2, s_4\}$ |
| $p_6$ | $P_{p_6} = \{p_6, p_3, p_5\}$ | $S_{p_6} = \{s_2, s_4\}$ |

(c) Neighbor list and IS-list

**Figure 4. An example of deploying Double Indirect Access. The solid lines and dashed lines depict the Voronoi diagrams of the p-id set and the s-id set respectively.**

## 4 Double Indirect Access

Two global Voronoi diagrams are exploited in Double Indirect Access: one of indirection servers and the other of index servers. Each node joins DIA with its s-id and acts as an index server. Moreover, it joins DIA with its p-id and acts as a distributed indirection server as well as a location server. Figure 4(a) shows the stack of two Voronoi diagrams: one of six indirection servers (which are also location servers) $p_1, \ldots, p_6$ and the other of six index servers $s_1, \ldots, s_6$.

DIA is similar to SIA except that an index entry of an object with o-id is stored on the index server whose s-id, not p-id, is closest to the o-id. Since indexes are distributed among index servers according to their s-ids and the s-ids do not change, index redistribution does not occur even when nodes move over the p-id update threshold.

Each indirection server computes two local Voronoi diagrams. Let $P_{p_i}$ be the neighbor list of an indirection server $p_i$, which is the set of p-ids of the indirection server itself and its neighbor nodes. First, $p_i$ computes the local Voronoi diagram of $P_{p_i}$

$$V(P_{p_i}) = \{v_P(p_j|p_i) \mid p_j \in P_{p_i}\}.$$

In Figure 4(b), the solid lines depict the local Voronoi diagram of the p-id set computed at $p_4$ (i.e., $V(P_{p_4})$).

The index server list, called *IS-list* of an indirection server $p_i$ is defined to be the set

$$S_{p_i} = \{s_j \mid v_S(s_j) \cap v_P(p_i) \neq \emptyset \text{ for } j = 1, \ldots, n\}.$$

An indirection server $p_i$ also computes the local Voronoi diagram of $S_{p_i}$

$$V(S_{p_i}) = \{v_S(s_j|p_i) \mid s_j \in S_{p_i}\}.$$

In Figure 4(b), the dashed lines show the local Voronoi diagram of the s-id set computed at $p_4$ (i.e., $V(S_{p_4})$). Every index server lookup request with an o-id $o \in v(p_i)$ is routed to the indirection server $p_i$ so that $p_i$ should be able to find out which index server is responsible for the o-id $o$. Therefore, the indirection server $p_i$ must store the s-id $s$ of every index server whose Voronoi cell overlaps its own Voronoi cell; i.e., $v_S(s|p_i) \cap v_P(p_i|p_i) \neq \emptyset$. For example, in Figure 4(b), the IS-list of $p_4$ includes four index servers $s_1$, $s_2$, $s_5$, and $s_6$ (i.e., $S_{p_4} = \{s_1, s_2, s_5, s_6\}$), since each of the four Voronoi cells overlaps the Voronoi cell of $p_4$.

As shown in Figure 5, The process of object lookup is executed in four steps: (1) When a client issues an index server lookup request for an object with o-id, the request is routed to the indirection server whose p-id is closest to the o-id. The indirection server, with the help of its local Voronoi diagram of the s-id set, finds out the index server whose s-id is closest to the o-id, and then it issues a location lookup request for the index server with the s-id. (2) The location request will be routed to the location server whose p-id is closest to the s-id of the index server. Since the location server can find out the current location of the index server, it can send the object server lookup request to the index server. (3) Upon receiving the lookup request, the index server finds out the s-id of the object server by searching its local indexes, and then it issues a location lookup request for the object server. (4) The location request will be routed to the location server whose p-id is closest to the s-id of the object server. The location server checks the age of location information of the object server. In case that the

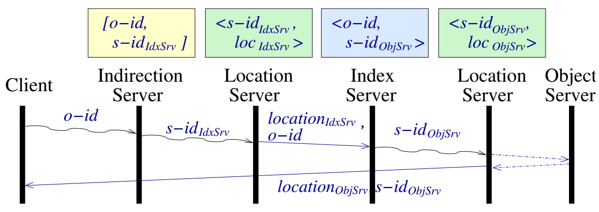


**Figure 5. An object lookup in Double Indirect Access is performed in four steps.**

location information is not too dated to be sufficiently accurate, the location server sends the lookup result that contains the s-id and geographic location of the object server to the client. Otherwise, the location request should be routed to the object server to get more accurate location information.

### 4.1 IS-List Redistribution

Whenever a node updates its p-id and its local Delaunay triangulations of the p-id set, the node itself and its neighbor nodes should exchange IS-list entries with the help of their own local Voronoi diagrams.

Each IS-list entry is stored redundantly since the Voronoi cell of a s-id usually overlaps many Voronoi cells of p-ids. Therefore, a node uses its own local Voronoi diagrams to prevent unnecessary packet transmissions in the operation of IS-list redistribution. For more details of this IS-list redistribution algorithm, please refer to [4].

## 5 Experimental Results

In this paper, we evaluate the performance of Single Indirect Access (SIA) and Double Indirect Access (DIA) by using the discrete event simulator ns-2. The simulator in our previous work, which was written in C, did not include the details of the MAC and physical layer. Its simplicity enabled us to reduce the execution time of simulations at the sake of more detailed and realistic performance results. Due to unstable routing in MANET, two critical operations of DIA, the construction of the overlay network and the corresponding IS-list redistribution, may not be completed properly. In order to evaluate how DIA is affected by the underlying routing in MANET, we implemented SIA and DIA into the ns2 version 2.29.

### 5.1 Simulation Environment

All simulations are performed in a rectangular area of size 1,000 by 1,000 meters. The number of nodes is set to 100. Each node moves based on the random waypoint model [1]. The node speed is chosen uniformly between minimum and maximum speed. The minimum speed is set to 0.0 m/s and the default maximum speed is 30.0 m/s. The pause time is fixed to 0 second. The default p-id update threshold $d_{pid}$ is set to 75 meters, and the simulation duration is 60 seconds. The average number of objects per node varies from 1 to 1,024, and each object identifier is randomly chosen in the rectangular area. The size of each index entry and IS-list entry is set to 24 bytes and 18 bytes respectively. We assume 2 Mbps and $10^{24}$ Mbps 802.11 MAC layers and the GPSR routing protocol [2]. The simulation code of the GPSR routing protocol is obtained from [3], which is modified from the original code of [2].

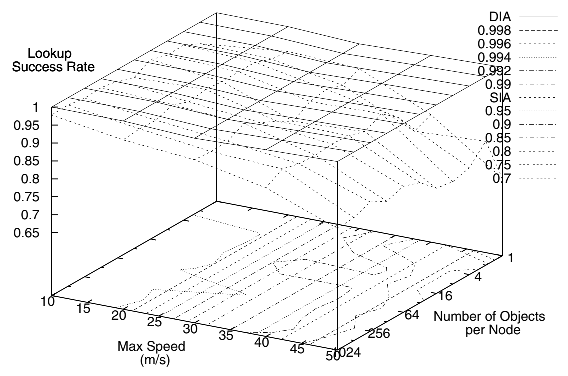


**Figure 6. The effect of the node speed and the average number of objects per node on the lookup success rate when $10^{24}$ Mbps MAC layer is assumed. The number represents the lookup success rate. For example, the line of 0.99 represents the point with lookup success rate of 0.99.**

The $10^{24}$ Mbps MAC layer is not realistic. However, it enables us to understand the effects of bandwidth limitation when compared to 2 Mbps MAC layer.

### 5.2 Simulation Results

The metric of the effectiveness is considered the lookup success rate which is the ratio of total valid index entries (or IS-list entries) stored in all index servers (or indirect servers) to the total index entries (or IS-list entries) in the system. In DIA (or SIA), when a node moves beyond the distance threshold, $d_{pid}$, the node itself and its neighbors should exchange IS-list entries (or index entries). If those entries are not delivered correctly, the associated indirect servers (or index servers) cannot maintain a correct list of index entries, resulting in lookup failure.

Figures 6 and 7 plot the lookup success rate of SIA and DIA after 60 seconds has elapsed in simulation time when $10^{24}$ Mbps and 2 Mbps MAC layers are assumed respectively. With $10^{24}$ Mbps MAC layer, the lookup success rate of SIA decreases from 97.8 % to 81.4 % when the maximum speed increases from 10 m/s to 50 m/s. This is because the number of Delaunay triangulation updates is proportional to node speed. Without any bandwidth limitation, the number of objects has little impact on the success rate.

Whereas, with 2 Mbps MAC layer, the lookup success rate falls not only when the average node speed increases but also when the number of objects increases. When the

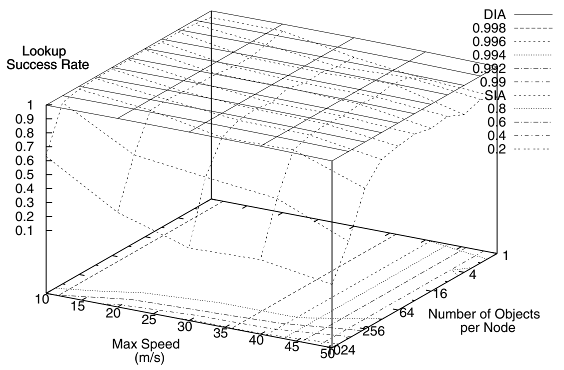


**Figure 7. The effect of the node speed and the average number of objects per node on the lookup success rate when $2$ Mbps MAC layer is assumed.**

**Table 3. The statistics of packet transmissions at router layer for IS-list redistribution in DIA under various values of the maximum node speed,** *Max Speed***.**

| *Max* | $10^{24}$ Mbps MAC layer | | | | 2 Mbps MAC layer | | | |
|---|---|---|---|---|---|---|---|---|
| *Speed* | Sent | Forw | Recv | Drop | Sent | Forw | Recv | Drop |
| 10 m/s | 1217 | 24 | 1219 | 0 | 1221 | 37 | 1242 | 0 |
| 20 m/s | 2664 | 64 | 2695 | 0 | 2669 | 79 | 2679 | 0 |
| 30 m/s | 4454 | 284 | 4623 | 4 | 4589 | 118 | 4589 | 0 |
| 40 m/s | 6081 | 355 | 6245 | 6 | 6226 | 151 | 6199 | 1 |
| 50 m/s | 7539 | 236 | 7549 | 4 | 7955 | 406 | 8038 | 7 |

maximum speed is 50 m/s and the average number of objects per node increases from 256 to 512, the success rate collapses from 65.7 % to 34.5 %. This is due to bandwidth limitation: The index redistribution packets created at router layer cannot be transferred at MAC layer. Note that the average length of index redistribution packets is proportional to the number of objects.

Table 1 shows the effect of the number of objects on the number of index redistribution packets which have been sent, forwarded, received, and dropped at router layer when the maximum speed is 30 m/s. A node forwards an index redistribution packet if it is not the packet's destination.

With $10^{24}$ Mbps MAC layer, the number of packets sent grows as *#obj*$_{node}$ increases. However, with 2 Mbps MAC layer, the number of packets sent collapses rapidly when *#obj*$_{node}$ is 1024. This is because the number and the length of packets increase as *#obj*$_{node}$ grows so that bandwidth limitation prevents the packets from being transferred in time. Consequently, the number of valid index entries stored at each node decreases and the number of packets that are newly created falls rapidly. Furthermore, with 2 Mbps MAC layer, the ratio of the number of index handover packets to the number of the index redistribution packets increases as *#obj*$_{node}$ rises, which indicates that the index redistribution and handover packets are not transferred in time.

Table 2 shows the effect of node speed on the number of the index redistribution packets which have been sent, forwarded, received, and dropped at router layer. As expected, with $10^{24}$ Mbps MAC layer, the number of packets sent increases as the node speed rises. At high speeds such as 30 m/s, 40 m/s, and 50 m/s, the number of the index redistribution packets sent is larger with $10^{24}$ Mbps MAC layer because the number of valid indexes stored at each node is larger than with 2 Mbps MAC layer.

In DIA, when the maximum speed is 50 m/s and the number of objects per node is 1024, the lookup success rates are 100.0 % and 98.8 % with $10^{24}$ Mbps and 2 Mbps MAC layers, respectively. The IS-list redistribution operation is observed to be completed in a short time due to the reasons explained as follows: Since the number of IS-list entries per packet is small and is independent of the number of objects, the packet delivery time will be short and IS-list redistribution will take less time than index redistribution. Furthermore, each IS-list entry is stored redundantly since the Voronoi cell of a s-id usually overlaps many Voronoi cells of p-ids. It is shown from an experiment that the average number of IS-list entries per node is 4.315 and its variance is 4.917. Thus, there will be a high chance for neighboring nodes to have multiple IS-list entries in common. The redundancy of IS-list entries increases the robustness of the operation of IS-list redistribution, thus making DIA less sensitive to instability of the underlying routing in MANET. For example, when a node $n$ changes its p-id and its Voronoi cell becomes to overlap the Voronoi cell of $sid$, it can successfully receive $sid$ if at least one of the indirection servers responsible for $sid$ stores $sid$ and sends it to $n$.

Table 3 shows the effect of node speed on the number of IS-list redistribution packets which have been sent, forwarded, received, and dropped at router layer when the network bandwidth is $10^{24}$ Mbps and 2 Mbps. The difference between those numbers from $10^{24}$ Mbps and 2 Mbps MAC layers is smaller compared to that of SIA, which supports that DIA is less sensitive to the available network bandwidth than SIA.

It is shown that, due to its relatively constant packet length and duplicated IS-list entries among neighbors, DIA is less sensitive to instability of the underlying routing in MANET than SIA under reasonable assumptions of node density, node mobility, and p-id update threshold distance.

**Table 1. The statistics of packet transmissions at router layer for index redistribution in SIA under various values of average number of objects per node, $\#obj_{node}$. The numbers in parenthesis mean the numbers of index handover packet transmissions. The maximum speed is set to 30 m/s.**

| *#obj* | $10^{24}$ Mbps MAC layer | | | | 2 Mbps MAC layer | | | |
|---|---|---|---|---|---|---|---|---|
| *pernode* | Sent | Forw | Recv | Drop | Sent | Forw | Recv | Drop |
| 1 | 963 ( 6) | 39 (0) | 989 ( 6) | 1 (0) | 1016 ( 3) | 13 ( 0) | 1008 ( 3) | 0 ( 0) |
| 2 | 1765 ( 2) | 59 (0) | 1780 ( 2) | 1 (0) | 1853 ( 16) | 23 ( 0) | 1831 ( 16) | 0 ( 0) |
| 4 | 2768 ( 2) | 88 (0) | 2776 ( 2) | 1 (0) | 2864 ( 14) | 39 ( 4) | 2824 ( 15) | 0 ( 0) |
| 8 | 3878 (10) | 171 (0) | 3913 (10) | 3 (0) | 4021 ( 30) | 96 ( 5) | 4001 ( 31) | 1 ( 0) |
| 16 | 4974 (12) | 208 (0) | 5001 (11) | 3 (0) | 5197 ( 37) | 122 ( 6) | 5123 ( 37) | 0 ( 0) |
| 32 | 5958 (35) | 236 (0) | 5968 (35) | 3 (0) | 6038 ( 58) | 158 ( 8) | 5958 ( 58) | 0 ( 0) |
| 64 | 6682 (44) | 252 (0) | 6665 (44) | 1 (0) | 6835 ( 91) | 229 ( 9) | 6766 ( 93) | 0 ( 0) |
| 128 | 7214 (50) | 329 (0) | 7251 (50) | 4 (0) | 7406 ( 170) | 266 ( 19) | 7302 ( 174) | 0 ( 0) |
| 256 | 7524 (77) | 346 (0) | 7560 (76) | 3 (0) | 7777 ( 343) | 349 ( 21) | 7638 ( 341) | 0 ( 0) |
| 512 | 7768 (61) | 355 (0) | 7794 (61) | 3 (0) | 7905 (1388) | 1024 (288) | 7536 (1341) | 19 ( 7) |
| 1024 | 7976 (51) | 409 (0) | 8052 (51) | 5 (0) | 4754 (1549) | 2036 (802) | 4147 (1572) | 165 (55) |

**Table 2. The statistics of packet transmissions at router layer for index redistribution in SIA under various values of maximum node speed. The number in parenthesis is the number of index handover packet transmissions. The number of objects per node is set to 1024.**

| *Max* | $10^{24}$ Mbps MAC layer | | | | 2 Mbps MAC layer | | | |
|---|---|---|---|---|---|---|---|---|
| *Speed* | Sent | Forw | Recv | Drop | Sent | Forw | Recv | Drop |
| 10 m/s | 1984 ( 6) | 199 ( 0) | 2133 ( 6) | 3 (0) | 2222 ( 184) | 487 ( 32) | 1942 ( 170) | 45 ( 5) |
| 20 m/s | 4693 ( 32) | 278 ( 6) | 4793 ( 32) | 2 (0) | 4413 (1416) | 1345 ( 405) | 3916 (1256) | 99 (21) |
| 30 m/s | 7976 ( 51) | 409 ( 0) | 8052 ( 51) | 5 (0) | 4754 (1549) | 2036 ( 802) | 4147 (1572) | 165 (55) |
| 40 m/s | 11161 (235) | 795 (17) | 11244 (235) | 7 (0) | 6713 (2164) | 2802 ( 928) | 6323 (2004) | 191 (52) |
| 50 m/s | 13738 (377) | 893 (57) | 13663 (407) | 15 (1) | 6926 (2785) | 3608 (1569) | 6713 (2898) | 198 (80) |

Another significant metric is the bandwidth consumed by IS-list redistribution packets in DIA, and index redistribution and handover packets in SIA. Figure 8 plots the bandwidth consumption at MAC layer under various average numbers of objects per node. *DIA_IS-list* refers to the bandwidth consumed by IS-list redistribution packets generated at DIA agents. Similarly, *SIA_index* refers to the bandwidth consumed by index redistribution and handover packetes generated at SIA agents. Both *DIA_total* and *SIA_total* include the bandwidth consumed by all packets including ARP (Address Resolution Protocol), RTS (Ready to Send), CTS (Clear to Send), ACK (Acknowledgement) packets generated at MAC layer, and live beacon packets generated at GPSR agents. As expected, the bandwidth SIA consumes increases as the number of objects increases, whereas the bandwidth DIA consumes is independent of the number of objects. *DIA_total* and *SIA_total* cross each other when the average number of objects per node is about 8. Moreover, the difference between *SIA_total* and *SIA_index* remains steady, independent of the average number of objects per node.

Figure 8 plots the bandwidth consumption at MAC layer under various values of the maximum node speed. As expected, DIA consumes less bandwidth and the consumption grows with a much lower gradient as speed increases than SIA. An anomaly occurs when 2 Mbps MAC layer is used; *SIA_index* is higher at 30 m/s than at 40 m/s. This is because the network becomes saturated at 40 m/s and index redistribution or handover packets cannot be transferred in time so that the number of valid index entries stored at each node decreases rapidly. Consequently, the number of index redistribution packets that are newly created also decreases.

## 6 Conclusions

In this paper, we newly implemented the IS-redistribution algorithm of DIA and the index redistribution and handover algorithms of SIA into the packet-level simulator ns2 in order to evaluate how DIA is affected by the underlying unstable routing in MANET. The metrics of the effectiveness are considered the lookup success rate and the bandwidth consumed by IS-list redistribution packets in DIA, and index redistribution and handover packets in SIA. Simulation results showed that DIA is less sensitive to instability of the underlying routing in MANET

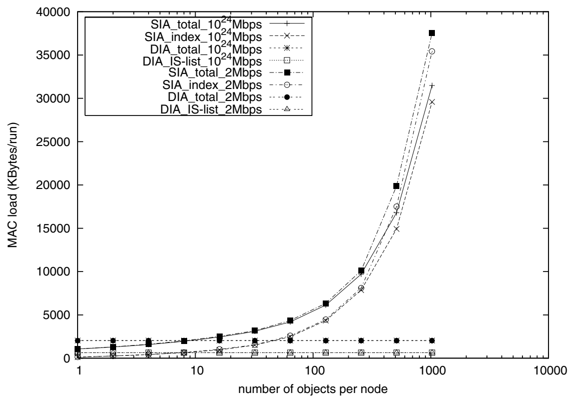


**Figure 8. The effect of the number of objects on the MAC load when the maximum node speed is 30 m/s.** *DIA_IS-list* **refers to the MAC load occurred by IS-list redistribution packets.** *SIA_index* **refers to the MAC load occurred by IS-list redistribution or handover packets. Both** *DIA_total* **and** *SIA_total* **additionally include the MAC load occurred by ARP, RTS, CTS, ACK, and GPSR.**

than SIA, due to its relatively constant packet length and duplicated IS-list entries among neighbors, under reasonable assumptions of node density, node mobility, and p-id update threshold distance. Furthermore, DIA consumes less bandwidth and the consumption grows with a much lower gradient as speed increases than SIA.

Future works include the following research topics: (1) Designing and implementing a distributed Delaunay triangulation update algorithm – Currently, Delaunay triangulation is computed globally, and the overhead of maintaining and updating overlay topology is unknown. (2) Implementing the entire process of object lookup/insert – The statistics related with object lookup/insert such as response time may be another significant metric. To fully evaluate our design, we need to implement every step of the object lookup and insert.

## Acknowledgements

The authors would like to thank the Ministry of Education of Korea for its support toward the Electrical and Computer Engineering Division at POSTECH through its BK21 program. This work was also supported in part by the MIC (Ministry of Information and Communication), Korea, under the ITRC (Information Technology Research Center) support program supervised by the IITA (Institute of Information Technology Assessment) (IITA-2006-C1090-0603-0045) and by the IT R&D program of MIC/IITA [2005-S-065-03, Development of Wearable Personal Station].

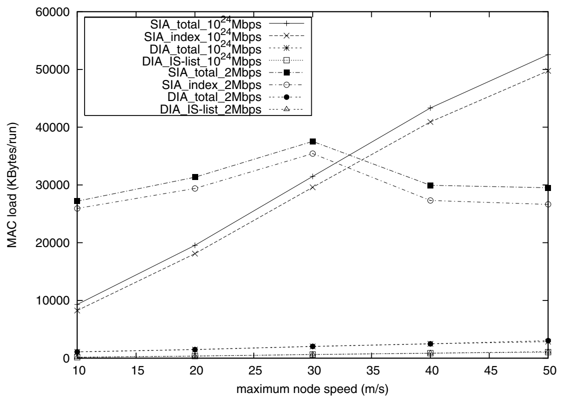


**Figure 9. The effect of the node speed on the MAC load when the average number of objects per node is 1024.**

## References

[1] J. Broch, D. A. Maltz, D. B. Johnson, Y.-C. Hu, and J. Jetcheva. A performance comparison of multi-hop wireless ad hoc network routing protocols. In *Proc. ACM MobiCom*, pages 85–97, San Diego, CA, October 1998.

[2] B. Karp and H. T. Kung. Gpsr: greedy perimeter stateless routing for wireless networks. In *Proc. ACM MobiCom*, pages 243–254, 2000.

[3] W. Kiess, H. Fübler, J. Widmer, and M. Mauve. Hierarchical Location Service for Mobile Ad-Hoc Networks. *ACM SIGMOBILE MC2R*, 8(4):47–58, October 2004.

[4] D.-W. Kim and C.-I. Park. Double indirect access: Efficient peer-to-peer object lookup protocol in location-aware mobile ad hoc networks. *IEICE Transactions on Communications*, E90-B(4), 2007.

[5] A. Okabe, B. Boots, K. Sugiharax, and S. N. Chiu. *Spatial Tessellations : Concepts and Applications of Voronoi Diagrams.* John Wiley and Sons Ltd., 2 edition, July 2000.

[6] S. Ratnasamy, P. Francis, M. Handley, R. Karp, and S. Shenker. A scalable content-addressable network. In *Proc. ACM SIGCOMM*, San Diego, 2001.

[7] S. Ratnasamy, B. Karp, S. Shenker, D. Estrin, R. Govindan, L. Yin, and F. Yu. Data-centric storage in sensornets with ght, a geographic hash table. *Mob. Netw. Appl.*, 8(4):427–442, August 2003.

[8] I. Stoica, R. Morris, D. Karger, M. F. Kaashoek, and H. Balakrishnan. Chord: A scalable peer-to-peer lookup service for internet applications. In *Proc. ACM SIGCOMM*, San Diego, California, August 2001.

[9] B. Zhao, J. Kubiatowicz, and A. Joseph. Tapestry: An infrastructure for fault-tolerant wide-area location and routing. Technical Report UCB/CSD-01-1141, Computer Science Division, U. C. Berkeley, April 2001.