Design and evaluation of two scalable protocols for location management of mobile nodes in location based routing protocols in mobile Ad Hoc Networks

Amjad Osmani[1], Abolfazl Toroghi Haghighat[2]
(1,2) Department of Computer Engineering and Information Technology
(1) Saghez Branch, Islamic Azad University, Saghez, Iran
(2) Qazvin Branch, Islamic Azad University, Qazvin, Iran
a.osmani@iausaghez.ac.ir, haghighat@qiau.ac.ir

Received: 2011/01/10 ;Accepted: 2011/04/10 Pages: 55-74

Abstract

Heretofore several position-based routing protocols have been developed for mobile ad hoc networks. Many of these protocols assume that a location service is available which provides location information on the nodes in the network.Our solutions decrease location update without loss of query success rate or throughput and even increase those.Simulation results show that our methods are effective and the algorithms are distributed and can keep scalability in scenario of increasing nodes density.

Keywords: *Mobile Ad Hoc Wireless Networks; Mobility Management; Location Service*

1. Introduction

AD HOC networks consist of autonomous nodes that collaborate in order to transport information. Usually, these nodes act as end systems and routers at the same time [1]. Due to node mobility, the network topology changes frequently which makes the design of a scalable and robust routing protocol with low message overhead, one of the challenging task in this kind of networks. Routing a packet from a source to a destination in a mobile ad hoc network is a challenging problem, since nodes in the network may move and cause frequent, unpredictable topological changes [2]. Location services are used in mobile ad hoc and hybrid networks either to locate the geographic position of a given node in the network or to locate a data item. One of the main usages of position location services is in location based routing algorithms.

2. Related Work

Figure 1 shows the classification of the location services proposed so far. Location services can be divided into flooding-based and rendezvous-based approaches. Flooding-based protocols can be further divided into dissemination and reactive approaches. In the dissemination approach, each node periodically floods its location to all nodes in the network. Thus, when a given node requires location information on another node, the information is found in the node's location table, i.e., the dissemination services usually do not send query messages. They can be classified as an all for-all approach. In the reactive approach, nodes do not send update messages;

instead they query location information of a specific node only if needed. The location query is flooded to the whole network. The reactive services belong to all-for some category. In rendezvous-based approach, all nodes agree on the set of location servers. Reactive and dissemination services represent the two extremes of the update strategy and they are not scalable. We focus in the following on the rendezvous based services. Two approaches are used to select the location servers, quorum-based and hashing based [3, 4].

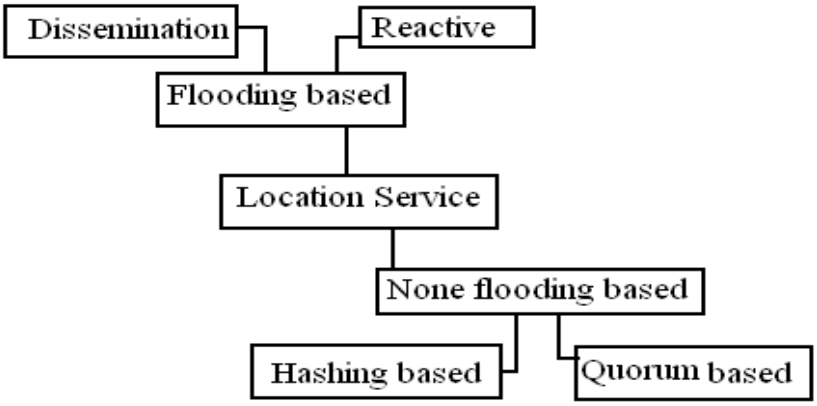


Figure 1. Location services classification

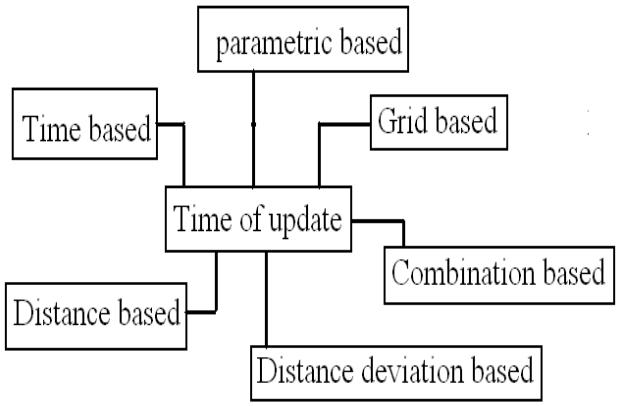


Figure 1. Time of update classification

One of the main problems in location service problem is time of sending of location update packets. As per available methods, we can classify those to: 1) Time based, 2) Distance based, 3) Distance deviation based, 4) Combination based, 5) Grid based and 6) parametric based methods. The proposed classification is available in figure 2.

2.1. Time based (periodic)

In this category, after a special time each node generates a packet (with new location information) and sends that. We can address ADLS [5], DQS [6] and DREAM [7]. It is possible that a node sends a packet (after a special time) but without long passed distance.

[7] has proposed the Distance Routing Effect Algorithm for Mobility (DREAM) in which nodes maintain a location table using the distance effect. Nodes maintain location information of all other nodes in the network proactively. However, the location of a node is updated (by the node) to its nearer neighbors more frequently than nodes that are farther. To send a packet to the destination a source node estimates the expected zone of the destination based on the destinations' location (using its location table) and floods data packets within the expected region. An intermediate node upon receiving the packet re-broadcasts the data packet if it is within the expected region and this continues until the destination receives the packet.

[5] proposes ADLS, an Adaptive Demand-driven Location Service – a multi-home region scheme that creates and maintains beyond a single home region for each node on-demand, based on the actual location demand for each node and the locality of the querying sources.

In [6] the author designs an adaptive location service on the basis of diamond quorum considering the gravity of locality of a mobile node in a MANET. In the protocol, the topology of the ad-hoc network is divided into several logical regions of 2-layered grid structure, and single home region is selected from each two dimensional region by using the mapping function. Then, the logically spread surface quorum system is composed from these selected N home regions, and the diamond quorum system (DQS) is constructed from this system. If one mobile node updates its location, a quorum is selected from the DQS by considering the gravity of locality of that mobile nodes, the location information of that mobile node is stored to the nodes in the selected quorum.

2.2. Distance Based

In this category, the nodes after a special distance generate a packet (with new location information) and send that. We can address GLS [8], GHLS [9] and KCLS [10]. It is possible that a node sends a packet (after a special distance) but without long passed time.

The GHLS protocol proposed in [9] shares the nature of geographic hashing with GHT [11]. However, GHT is proposed to support data storage in dense sensor networks which are typically static. Additionally, storage and replication strategies are fundamentally different for GHT since it stores sensed data where reliability and storage costs are a bigger concern.

The GLS (Grid Location Service) [8] divides the area containing the ad hoc network into a hierarchy of square, forming a quad-tree. Each node selects one node in each element of the quad-tree as a location server. Therefore the density of location servers for a node in areas close to the node is high and becomes exponentially less dense as the distance to the node increases. The update and request mechanisms of GLS require a chain of nodes based on node IDs is found and traversed to reach an actual location server for a given node. The chain leads from the updating or requesting node via some arbitrary and some dedicated nodes to a location server.

The KCLS protocol [10] is based on a single level k -hop cluster structure to provide location service. A k -hop cluster is a set of hosts under the cluster-head and any host in the cluster has a distance of equal to or less than k hops to the cluster-head. Every

cluster consists of one cluster-head, ordinary cluster members that are located inside of a cluster, and gateways which are located on the cluster border to connect the neighboring clusters. It is supposed that each host has a unique host ID. The cluster ID is determined by the host ID of the cluster-head.

2.3. Distance Deviation Based

In this category, the nodes after a special time (for example 1 second) check predicted location and real location and if distance deviation be greater than a special value then generate a packet (with new location information) and send that. We can address DRM [12], SaLAM [13], H2SaLAM [14] and SHGRID [14].

SaLAM [13] (Scalable @ Location Advertisement Management in MANET) has designed based on some phases such that some of them are used in SaLAM-ON-HGRID and some of that are used in SaLAM-ON-SLALoM and SaLAM-ON-SLURP. The phases are used to decreasing control overhead and solutions are not limited to a special network grid ordering.

H2SaLAM [14] uses a dynamic hierarchy between location servers such that solves ping-pong problem between rings. SHGRID [14] using one hop chain technique has prevented decreasing of data delivery rate and it has increased the delay experienced by data packets. Both of them use location prediction method, near to destination.

2.4. Combination Based

This category can be a composed of Time based and Distance based methods. We can address GrLS [15].

GrLS [15] consists of two components: individual location management and group location management. In the protocol, the network coverage area is partitioned into equal circle-shaped regions, which are selected as home regions by nodes. For each node with individual mobility, it sends location updates to the location servers in its home region and the location server handles all the location queries for it. For nodes with group mobility, group location management is designed, which consists of micro and macro group location management. With micro group location management, each group member is aware of the locations of all other group members. With macro group location management, a designated group leader updates its location to the location servers in a specified group home region and replies all the location queries for other group members in its group. Thus, the overhead of location updates to the home regions can be saved for all the other group members.

2.5. Grid Based

In this category, each time a node crosses a grid boundary, it generates a packet (with new location information) and sends that. We can address HGRID [16] and SLURP [17], SLALoM [18], HLS [19], SaLAM, H2SaLAM, SHGRID, NGRID [20].

In [18] they present an algorithm, called Hierarchical Location Service (HLS), which can efficiently provide position information about nodes in ad hoc networks. HLS is a

hierarchical architecture to maintain the location information of nodes. [18] also mention some update/query problem of location service would be happened in hierarchical manner, and [18] discuss how HLS can solve these problems when providing the location service.

In HGRID [16], Hierarchical leader nodes serve as location servers, and are updated by other nodes on crossing grid boundaries, via location update packets. A lower order leader notifies its leader only if the location update requires it to update its leader. Thus, while the leaders in the highest level of the hierarchy know the approximate locations of all the nodes in the network, location information in servers becomes more accurate as one traverse down the hierarchy. Location servers can now be queried by source nodes who wish to know the location of the destination, in an on demand fashion.

In SLURP [17], the network area is divided into a flat grid of squares. Node A selects its location servers by applying a hash function to A's ID and obtains the (x, y) coordinate of a point in the entire area. The square containing that point is called the home square for node A. All nodes in that square store A's exact location information. Every time node A moves to a different square, it updates its home square with new location information. For any node B wishes to communicate with node A, the same hash function is applied to node A's ID to obtain A's home square. A query packet is then forwarded to A's home square to retrieve A's location information. One of the major drawbacks of this design is that the query latency grows as the network size grows. Even if the querying node B is relatively close to the target node A, node B may still need to query A's home region that is far away. Furthermore, nodes may be far away from their home square and their updates may have to travel long distances.

2.6. Parametric Based

In this category, the nodes will check other parameters (instead of time and distance). We can address SaLAM and Column/Row [21].

In the Column/Row Location Service (CRLS) [21] the location of each node is propagated in the north–south direction, while any location queries are propagated in the east–west direction. When a node decides a location update is needed, it propagates the location update along the north–south direction, i.e., with the goal of reaching all the nodes in the same column in the geographic area. Each node selected as a location server in the north or south direction broadcasts the update to its one hop neighbors. The update contains the identifier of the next location server in the update direction. When a source node initiates a location request for a destination node, the query is propagated along the east–west direction, i.e., along its row of nodes in the geographic area. The query contains the time of the most recent location known to the source. If a node along the row has a cached location for the destination node that is more recent than the time in the query, it sends a reply packet via geographic forwarding back to the source.

The rest of this article is organized as following. In Section 3, we briefly review the basic concept of FLS. Section 4 details the FLS-algorithms design for distributed location service. Simulations are presented in Section 5. Section 6 concludes this paper through a summary.

3. Suggested Algorithms

3.1. Fuzzy Based Location Service (FLS)

We apply Fuzzy logic to our distributed location services (FLS (1, 2): Fuzzy-based Location Service).This section is divided to three parts: 1) grid ordering, 2) Location update, 3) location query based data transferring.

3.1.1. Hierarchical Partitioning

Our location services use a partitioned network based on unit square regions of side $r_t/\sqrt{2}$ (L0), where r_t is the transmission range of a node such that any two nodes within a L0 grid can directly communicate with each other. The grid hierarchy is defined as figure 3 shows which is grid based and hierarchical. Note that this may not be the only way to form clustering using unit grid (L0) regions and we just use it to setup our location service. We can simplify figure 3 in figure 4. We want to propose a system which is based on fuzzy logic to a better management of update operation in hierarchical location services.

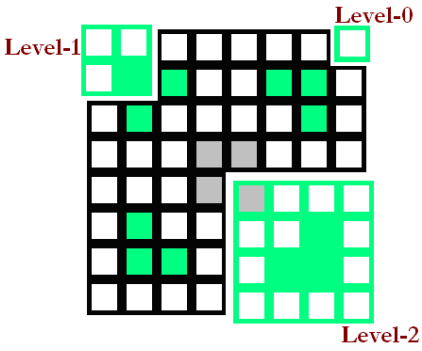


Figure 3. Squares area that are maintained by each level location server

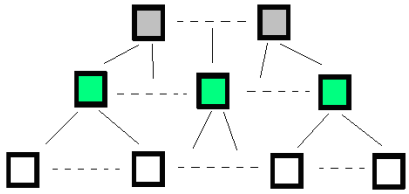


Figure 4. Location servers in hierarchically mode

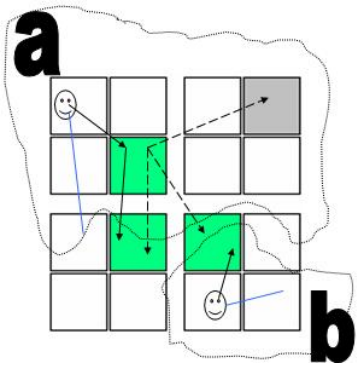


Figure 5. Two scenarios in Location-update-part

3.1.2. Location Update

In FLS (1, 2) each time a node crosses a L1 grid boundary, two location-update packets are generated, one to the L1 grid of its current L0 grid to indicate its arrival, and another packet to the L_1 grid of its previous L1 grid, indicating its departure from the

region and indicating location of its new leader in new L1 grid. Each packet contains information to make the location servers consistent in their view of the network. The location-update packets are processed at each level of the hierarchy in the following manner:

In scenario of b under figure 5, node u sends a location-update packet to its leader in L1 grid of its current L_0 grid and the leader that receives location-update packet updates its own location database. In scenario of a under figure 5, node u sends a location-update packet to its new leader in L1 grid of its new L0 grid and new leader that receives location-update packet updates its own location database and sends notification to L1 grid of previous L0 grid such that it makes a one-hop-chain between two leaders that if new location-update is not received (seldom) by hierarchical leader then new server can be achievable for data transfer . The new leader also checks the location-update packet to see whether the boundary under L1 grid of previous L0 grid has crossed and whether its hierarchical leader requires to be alerted then it uses fuzzy rules under Table 1 and if the consequent is one (it means trigger) it sends the location-update packet to its next hierarchical leader else waits to arrive adequate time to trigger (maybe another time). If the movement is within the area covered by the current hierarchical leader then the leader decides to stop the registration process. Thus the location registration process continues until the location-update packet reaches one of the four L leaders. In FLS (1, 2) each node has a counter that presents number of call-References (d_{NCNN}) that recently has had. Each time a node crosses a $L1$ grid boundary, it sends its d_{NCNN} and new location to new leader and new leader add d_{NCNN} to $\sum d$ .Addition to storing counter, each node stores the time of last call-Reference (T_{LCNN}). So each time each node that has crossed the L_1 grid of previous L_0 grid and hierarchical leader requires to be alerted, leader waits and uses fuzzy rules under Table 1.

We have two algorithms for this section. The first is Timer based and second is Cross based. At first we describe Timer based algorithm and then Cross based.

We want to aggregate the location update packets in first location server in each *L1* grid. For example if we have n location update packets that location server is needed to forward them to hierarchical server, we can show in here that our solution has lower overhead related to normal mode that server will send all packets to hierarchical server. So:

- *In normal mode*: we have *n* packets to forward so it means *n*(data + header)* that is *(n*data + n* header)*.
- *In our phase 2*: we have *1* packets to forward so it means *1*(n*data + header)* that is *(n*data +1* header)*.

Data means location of a mobile node and header is header section of a packet that is sent. It means leader aggregates location-update packets and each time that is essential, instead of all packets sends a location-update packet to hierarchical leader. It decreases the number of location-update packets that are generated but some of that are essential for responsiveness in operation of location- query while the other in another time maybe are used and we want to distinguish between them.

Timer based algorithm uses fuzzy rules in Table 1 and if probability of trigger is one then the algorithms decide for trigger else wait, where R is a random number between

zero and one. Under Table 1 e is neperian number and if $\sum d$ is high then probability of trigger is high and vice versa. We mainly focus on nodes that recently are referenced (by call). Recently means the time of last call-Reference is related to first and second fuzzy sets in figure 6. It means the nodes that recently are referenced with high probability, in future again are referenced relative to other. Maybe in future other nodes are referenced too, so the leader set a timer and every δ seconds add the value of $(\omega \times (N/2) + \lambda)$ to $\sum d$ (where N is the number of nodes that have entered to new L_1 grid up to now and (ω , λ) are for tuning of proposed value) and figure 7 is checked, if $\sum d$ is in trigger-mode then triggers an update packet, else waits to trigger in another time. It means the timer, in the time, increases weight of $\sum d$ such that other nodes which recently are not referenced do not be forgotten. So in FLS, firstly, update operation for low-essential nodes is delayed and secondly, location-update packets are aggregated and one packet is sent instead of all. The algorithms that we have introduced and proposed timer guaranty that we do not have significant loss of query success probability. In FLS (2) , if a leader decides to trigger to hierarchical leader then it will send update packet to hierarchical leader and other leaders that are in the same level and same hierarchical grid but in FLS (1) trigger-operation will stop in hierarchical leader. That is different of FLS (1, 2) in Location-update-part of location service.

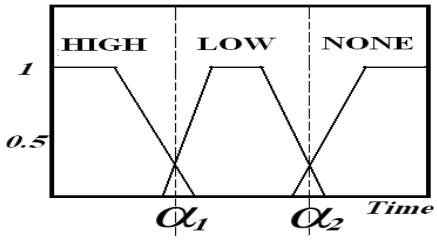


Figure 6. Membership function for the time of last call-reference

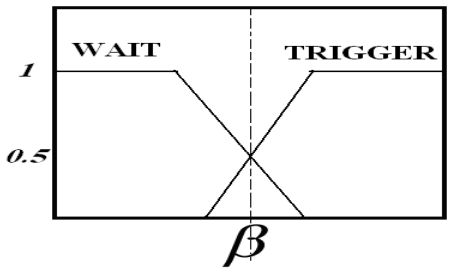


Figure 7. Membership function for the number of call-references

Table 1. Fuzzy Rules for Trigger or Not and Consequent

T_{LCNN}	$\sum d$	Probability of Trigger
HGH-URGENCY	Any	1
LOW-URGENCY	Any	$e^{-\left\lvert \frac{1}{\sum d} \right\rvert_{>R}}$
NON-URGENCY	TRIGGER	1
NON-URGENCY	WAIT	0 (means Wait)

In Cross based algorithm location server runs the algorithm in figure 8 and based on result of that will send location update packet to its hierarchical location server in L2 grid. The algorithm uses figure 9 to map *time* variable to *urgency* in *high* or *low*. Note that if *t(now) – t(last-call)* be between *zero and beta* then we say *high-urgency* else we

say *low-urgency*. Note that this can be using a kind of fuzzy logic. In here we describe the algorithm parameters:

T(now) is current time. *T(last-call)* is time of last call reference to receive data. *N(call)* is number of call reference heretofore. *N(cross-level-1)* is number of crossing of *L1* grids heretofore. Note that "Trigger update to hierarchical location server" in the algorithm means that sending a location update that is contained all previous hoarded location update packets such that packet is contained *(n*data +1* header)*.

if n(cross-level-1) is an even number
trigger update to hierarchical location server
else
if t(now) – t(last-call) means high-urgency
trigger update to hierarchical location server
else
if t(now) – t(last-call) means low-urgency
if $e^{-\left|1/N(Call)\right|} > e^{-\left|1/N(\text{cross-level-1})\right|}$
trigger update to hierarchical server
else
wait

Figure 8. The Cross based algorithm

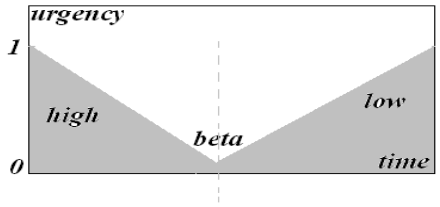


Figure 9. To map time variable to urgency in high or low

3.1.3. Location Query and Data Transferring

Considering FLS (1, 2) we suggest two methods for location query and data transferring. In FLS 1, as Fig. 10 shows, when s1 in scenario of b wants to send data to d1, it sends data to its leader then if leader does not find location of destination in its database it sends a query-packet to hierarchical leader and as soon as possible that receives reply packet it sends data to destination location. In FLS 2, as figure 10 shows, when s1 in scenario of a wants to send data to d1 ,it sends data to its leader then leader with considering specifications of FLS 2 if it finds location of destination in its database then it sends data to d1. If leader does not find the location of destination in its database then it sends a query-packet to hierarchical leader and as soon as possible that receives reply packet it sends data to destination location, but this situation is happened when destination is in one of other three grids with level of two.

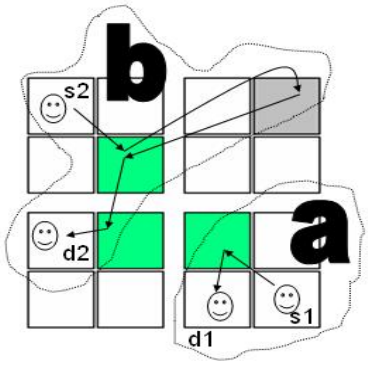


Figure 10. Two scenarios in location query and data transfer

3.2. Scalable Location Management (SLM)

This section is divided to three parts: 1) grid ordering, 2) Location update, 3) location query based data transferring.

3.2.1. Grid Ordering

We use a network area A that is divided to L0 grids (based on side of $r_t/2\sqrt{2}$) in lowest level, where r_t is the transmission range of a node. It then combines (K*K) L0 grid to form a L1 grid, where K is a variable and (K*K) are number of home regions in one L1 grid. Thus, due to this specific division of the terrain, every node has O(A/K*K) home regions in A.

In our method, the grids have two coordinates (g, h) and nodes have four coordinates (g, h, x, y).

You can see the grid ordering in figure 11, when network dimension is 1200m in 1200m and L0 dimension is 200m in 200m.

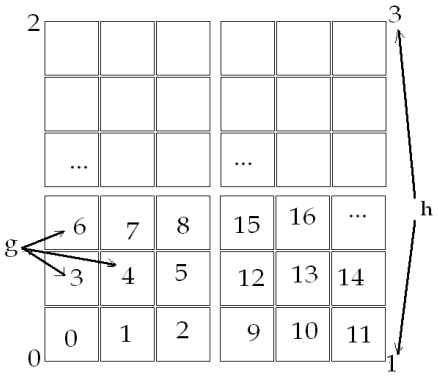


Figure 11. Grid ordering in SLM

3.2.2. Location Update

Each node uses a well-known hash function (f(x, y) = x mod y) to map nodes address (*IP* address or *id* or MAC address) to a L0 grid in each L1 grid (for determining location server home). Each node uses the hash function for sending update packets and also for sending discovery packets to destination location server home. The hash function is a many-to-one function. The algorithm to obtain location server home of destination with

its coordinates (home_h, home_g) in a L1 grid is in figure 12 where source's coordinates in that L1 is (g, h) .

when a node crosses a L0 grid boundary it runs algorithm in figure 13 and based on result of that decide for sending location update packet just to location server home in current L1 grid but in SLALoM[18] the location update packet is sent to 9 location server home such that one of them is in current L1 grid and the other are in 8 neighbor L1 grids.

```
function (destination_node_address)
{
    home_h=h;
    home_g=home_h * k * k +    f(destination_node_address, k*k);
}
```

Figure 12. The algorithm to obtain location server home coordinates

```
if (node crosses L1) then
  if (destination is in current L0) then
    trigger update with no-prediction mode
  else
    trigger update with prediction mode
  else
    if (node is in prediction mode) then
      goto a-label with probability of (1-p1)
    else
      a-label:
      if (destination is in current L0) then
        trigger update with no-prediction mode
      else
        trigger update with prediction mode
```

Figure 13. The algorithm to send location packet

The algorithm in figure 13 has two modes. The first is no-prediction mode that operates when a node's destination is in its current L0 or its distance from predicted value is greater than a threshold value (j). The second one is prediction mode. When a node is in prediction mode, it sends update packet to its server with probability of (1-P1). P1 is probability of that movement gradient (in formula 1) of the node is within the confine of the allowable for the node. Each node based on its number of call references in previous L0 grid crossing and current L0 grid crossing increases or decreases probability of P1.

The node can calculates movement gradient via formula 1:

$$\text{movement gradient} = \frac{Y - y}{X - x} \qquad (1)$$

Location server in L1 grid uses the formula 2 to prediction mobile nodes location (in figure 14):

$$\begin{cases} y2 = y1 + Vy \times (t2 - t1) \\ x2 = x1 + Vx \times (t2 - t1) \end{cases} \quad (2)$$

Where in formula 2 Vy and Vx are equal to formula 3:

$$\begin{cases} Vy = v \times SIN(\hat{\alpha}) \\ Vx = v \times COS(\hat{\alpha}) \end{cases} \quad (3)$$

Where v in formula 3 is equal to formula 4:

$$v = \frac{\sqrt{(Y - y)^2 + (X - x)^2}}{T - t} \quad (4)$$

And $\hat{\alpha}$ is equal to formula 5:

$$\hat{\alpha} = \begin{cases} ArcCOS\left(\frac{X - x}{\sqrt{(Y - y)^2 + (X - x)^2}}\right) \cdots\cdots\cdots\cdots When(Y - y \geq 0) \\ 2 \times \Pi - ArcCOS\left(\frac{X - x}{\sqrt{(Y - y)^2 + (X - x)^2}}\right) \cdots\cdots\cdots When(Y - y < 0) \end{cases} \quad (5)$$

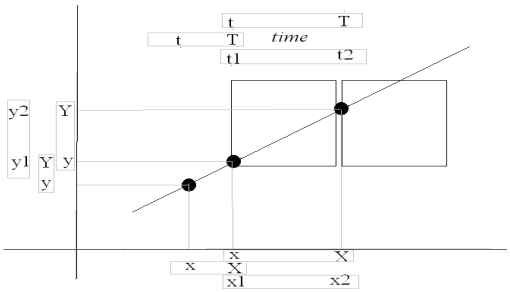


Figure 14. To prediction mobile node location

In this phase when a node crosses a L1 grid boundary sends location update packet to all location servers in all L1 grids like SLALoM but our tree is more efficient than multicast tree that is used in SLALoM and our tree decreases location update overhead. Our multicast tree uses five directions (Left, Right. Top, Bottom, NONE) or five arms and prunes idle arms to decrease update overhead. The algorithm for pruning based multicast tree is showed in Figure 15. Each location server for each mobile node stores the direction that it receives location update for that node and the name of that is P-direction or previous direction. So when it receives location update for that node in this time, it compares that with P-direction and the name of that is C-direction or current direction. Thus each location server for each mobile node stores location information and direction that it receives location update for that in its location database. Note that NONE means the node has entered in current L1 grid and location server in location server home of the node in current L1 grid records NONE for its direction because the node is in current L1 grid and has no direction.

//when an intermediate server receives a location update packet do:

```
//if  p-direct is top
 // if c-direct is left
        //forward   packet to next server homes in its top and right.
//elseif c-direct is right
        //forward packet to next server homes in its top and left.
//elseif c-direct is top
         // nothing
 //elseif c-direct is none
        //forward packet to next server homes in its top, left and right.
 //endif
//elseif  p-direct is buttom
 //if c-direct is left
       //forward packet to next server homes in its buttom and right.
//elseif c-direct is right
       //forward packet to next server homes in its buttom and left.
 //elseif c-direct is buttom
       // nothing
//elseif c-direct is none
      //forward packet to next server homes in its left, right and buttom.
//endif
//elseif p-direct is left
//if c-direct is top or buttom or left
       // nothing
//elseif c-direct is none
      //forward packet to next server homes in its left.
//endif
//elseif p-direct is right
//if c-direct is top or buttom or right
       // nothing
//elseif c-direct is none
      //forward packet to next server homes in its right.
 //endif
//endif
```

Figure 15. The algorithm to send location packets between servers

The mobile node when crosses a L1 grid boundary sends location update packet to its location server home in previous L1 grid but it can send location update packet to its location server home in new L1 grid whenever it crosses two L0 grids (consecutively) in new L1 grid. Also this phase can solve ping-pong [22] problem between L1 grids. Solving ping-pong problem between L1 grids is more important than between L0 grids because overhead of hand off is higher.

3.2.3. Location Query and Data Transferring

We can have three type nodes: source (sender node), intermediate node (forwarder) destination (receiver node).

The source node checks its location database and neighbors list and if finding result is not positive it sends a discovery packet to its location server in location server home in its current L1 grid and location server checks its location database and neighbors list. Of course if source receives a reply-packet it sends its data to another intermediate location server home in proposed multicast tree in phase 4. This is because of

specifications of multicast tree that we use in phase 4. Our multicast tree decrease location information advertisement. It means data moves on a waistline that is created by multicast tree between location server homes of each mobile node. And in last location server home the first location server that receives data it sends data to destination based on algorithm in figure 16. If the node is in prediction mode server predicts location of destination in L1 grid by formulas in phase 3 and sends data to destination.

4. Simulation Results

In this section, we study the performance of the proposed algorithms. When location information is available, geographic forwarding can be used in the place of establishing a route from a source node to a destination node. There are different methods for forwarding packets in geographic way, for instance flooding, restricted flooding, hierarchical methods and greedy forwarding. Greedy methods seem to be better for geographic routing, because these methods just need local information for choosing route. We use a composed method for geographic routing of packet through network. This method has two sections. Assume that we have three type nodes that have main roles in geographic routing in one hop forwarding to destination, node S that is source node, node I that is intermediate node and D which is destination node.

At first node S finds the closest node to D related itself between its neighbors.

But the first phase is based on a greedy algorithm and sometimes may fail for the sake of void-space (or hole-space) such that the node cannot find closest neighbor node and packet is dropped. In Ad-hoc networks this problem is completely temporary because of node mobility, consequently further attempts leads to success. In this work we use a simple way (in figure 17 and figure 18). The way is finding hindmost node between neighbors. This strategy leads to turn around the face. Since the I nodes keep records of visited nodes, loops are prevented.

```
if (node is in no-prediction mode) then
    send data to destination without prediction
else
    send data to destination with prediction of its location
```

Figure 16. The algorithm to send data

```
if  (s find closest node to d)
    //forward the packet to closest node to d
else
    //forward the packet to hindmost neighbor
endif
```

Figure 17. The algorithm to forward packets

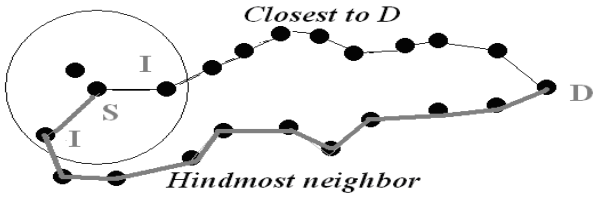


Figure 18. Forwarding of packets schema

4.1. FLS

We have evaluated the performance of the FLS (1, 2) in terms of location overhead, query-cost, location discovery count and query success probability. Table 2 shows the parameter values for the simulation. At beginning of setup of network, counter of each node sets to one and after that reach to $\beta / 2$ at everywhere of network, it will be set to one again.

Table 2. Simulation parameters for FLS

MAC model	IEEE 802.11
Simulation Time	300s
Each Side Of L0	100 m
$(\alpha 2 - \alpha 1)$ (Sec)	8
β	7
δ (Sec)	15
ω	1
λ	0
Transmission Range	144m

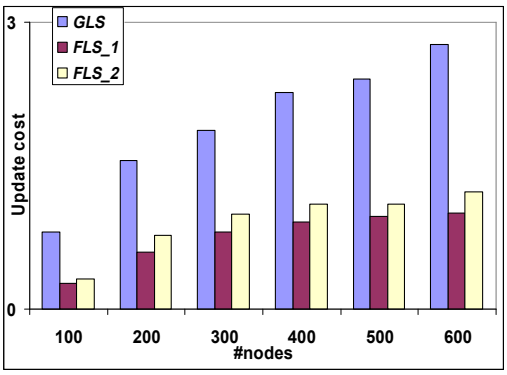


Figure 19. Average update cost vs. #nodes

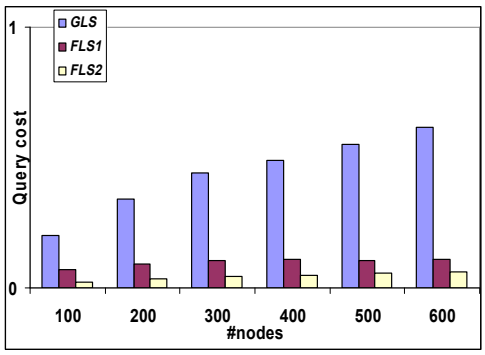


Figure 20. Average query cost vs. #nodes

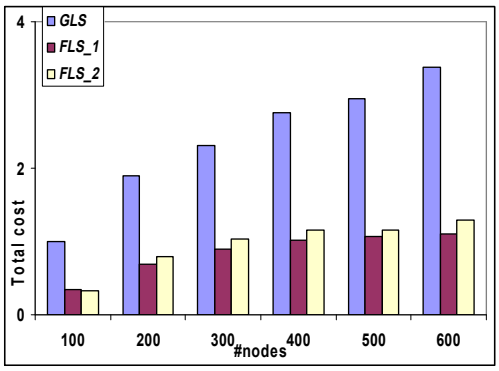


Figure 21. Average total cost vs. #nodes

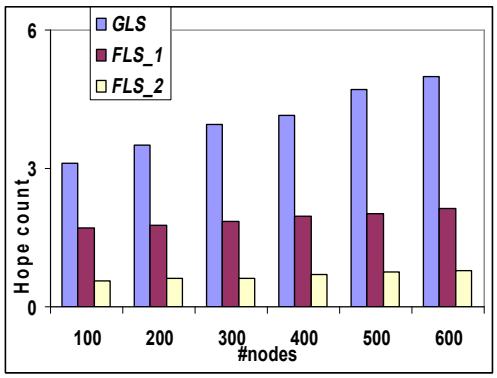


Figure 22. Average hop count vs. #nodes

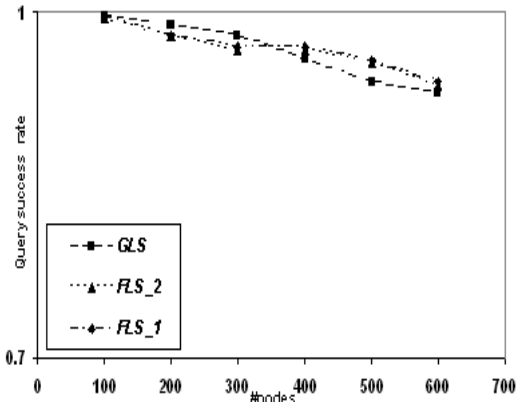


Figure 23. Query success rate vs. #nodes

Figure 19 shows average update overhead in packets per node per second for all the protocols. FLS (1, 2) have low overhead related to GLS and FLS 1 has overhead lower than FLS 2 because of different in last step of Location update process. We at all (as figure 19 shows) have decreased overhead of update operation. Figure 20 shows average query overhead for all three protocols. This term in FLS 1 is lower than FLS 2, because in FLS 2, if leader decide to trigger to hierarchical leader update packet is sent to leaders (including hierarchical leader and leaders that are in the same level and are in the same hierarchical grid) and request for location (and data transferring) goes to first leader and it (mostly) finds destination location and transfers data to destination. Figure 21 shows total cost for three protocols. It means average update cost and average query cost. Average update cost in FLS 1 is lower than FLS 2 and average query cost in FLS 2 is lower than FLS 1 but at all total cost in FLS 1 is lower than FLS 2. Figure 22 shows average hop count for location discovery for the three protocols. It means how much time in count (from an node to another node that is its neighbor) are required for finding location of destination .As figure 22 shows, in our protocols location-discovery-count is lower than GLS because of terms such as data and location-Query process are together in the way. Term of location-discovery -count in FLS 2 is lower than FLS 1 because of different in Location update-process, of course, decreasing of location-discovery-count mainly is related to last step of Location update-process in FLS 2 such that overhead

of update in FLS 2 is more than FLS 1. As you see in figure 19, while overhead of update in GLS is greater than FLS (1, 2) but discovery-cost in figure 20 in GLS is greater than FLS 1,2 and that is key point behind our algorithms where we have managed mobility using fuzzy logic system. Location-discovery-time in FLS (1, 2) is lower than GLS too, because location-discovery count is lower than GLS. Figure 23 shows the query success probability of location for the three protocols. Query success probability in our services at first is lower than GLS because our algorithms wait and immediately do not trigger location-update to hierarchical leader, so databases in hierarchical leaders are not consistent and this decreases query success probability related to GLS but that is not significant, as you see in Figure 23, because our algorithms using fuzzy logic and one-hop-chaining do not let to significant decreasing of success probability of location. In continue of increasing nodes success probability in our methods will be greater than GLS. One of the reasons can be significant increasing of location-update-packets in GLS related to FLS1,2 .Result of that is congestion and exact locations will not arrive (on time) to leaders ,so locations will not be precise related to FLS 1,2 (by increasing of nodes). We want to make an algorithm (in current work) such that location overhead be lower than GLS while reaching the good query success probability of location without significant loss of that related to GLS and we will work (in our other works) on simulation of other parts that are important for improving of mobility management and location service in mobile ad hoc networks.

4.2. SLM

Table 3 shows the parameter values for the simulation.

Table 3. Simulation parameters for SLM

MAC Model	IEEE 802.11
Simulation Time	900s
Each Side Of L0	125 M
Mobility Model	Random Waypoint
Number Of Connections	360
Traffic Pattern	Random
Transmission Range	350

Figure 24 shows comparison parameter of control overhead between SLM and SLALoM. SLM operates better than SLALoM, because our method in L1 grids just send update packet to one home of location and between L1s uses a pruning based multicast tree that keep location update traffic locally. Figure 25 shows comparison parameter of throughput. Our method increases parameter because overhead in the network is lower than SLALoM and dropped packets can be lower than that. Figure 26 shows comparison parameter of number of dropped packets. Dropped packets in our method are lower than SLALoM, because of lower overhead. Figure 27 shows comparison parameter of delay based on hops between source and destination. SLM increases the delay experienced by data packets related to SLALoM.

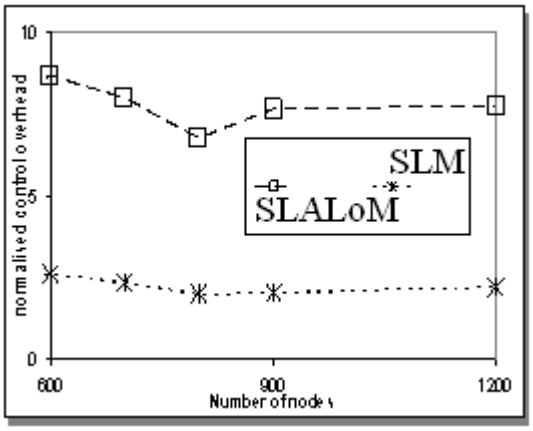


Figure 24. Control overhead vs. #nodes

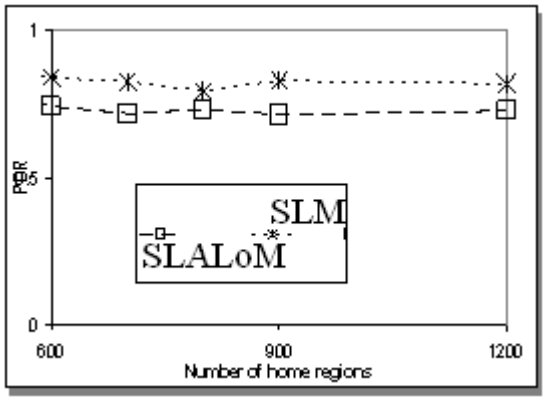


Figure 25. Throughput vs. #nodes

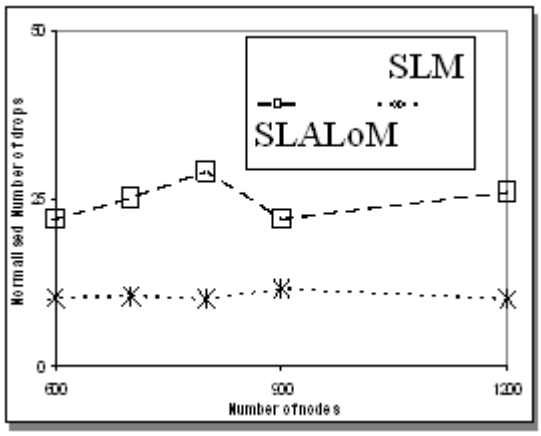


Figure 26. Number of dropped packets vs. #nodes

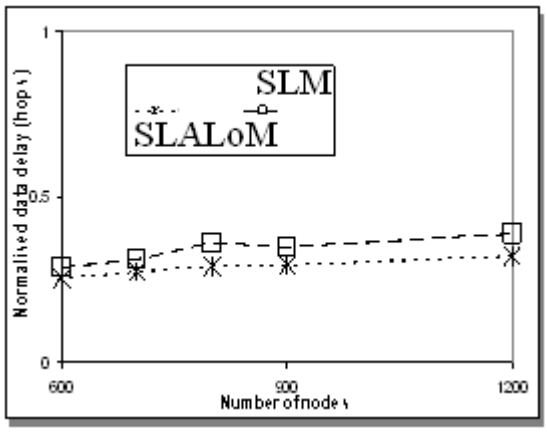
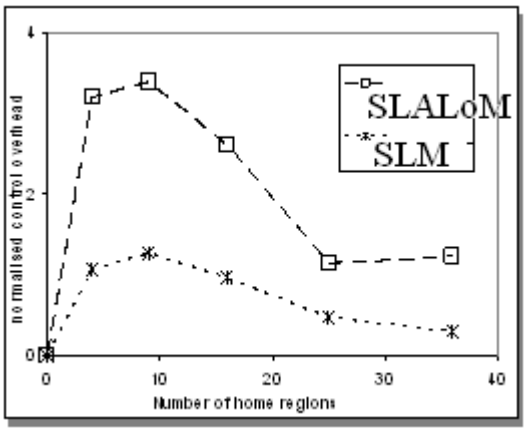


Figure 27. Delay vs. #nodes



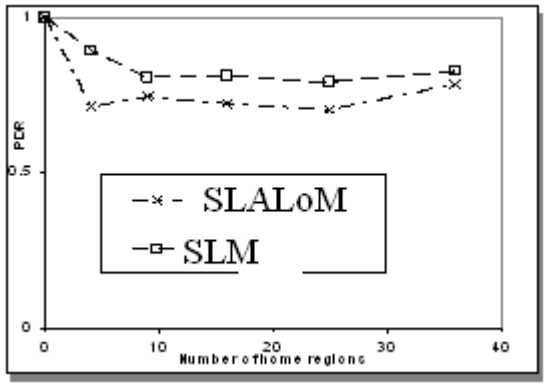


Figure 28. Control overhead vs. #home-regions ***Figure 29. Throughput vs. #home-regions***

Figure 28 shows comparison parameter of control overhead between SLM and SLALoM. Our method can decrease overhead with increasing number of home regions related to SLALoM. Figure 29 shows comparison parameter of throughput. Our method can increase parameter related to SLALoM because overhead in the network is lower than SLALoM and dropped packets can be lower than that.

5. Conclusion and Future Works

In this article, we have introduced some methods to better management of location update operation. Note that our algorithms are distributed and can keep scalability in scenario of increasing nodes density. FLS uses fuzzy logic before sending of location updates packets to hierarchical location servers. SLM uses a new composed method to update mobile nodes location when the nodes cross a grid boundary. Simulation results show that our methods decrease location-update-overhead without loss of throughput or query success rate.

References

1. Mauve, M., Widmer, J. and Hartensteinm H. (2001). A Survey on Position-Based Routing in Mobile Ad Hoc Networks, (IEEE) NEC Europe.
2. Camp, T., Bolengm J. and Davies, V. (2002). A survey of mobility models for ad hoc network research, (IEEE) WCMC.
3. Camp, T., Boleng, J., Wilcox, L. (2001). Location information services in mobile ad hoc networks, (IEEE) ICC.
4. Luo, X., Camp, T., Navidi, W. (2005). Predictive methods for location services in mobile ad hoc networks, (IEEE) IPDPS.
5. Seet, B-C., Pan, Y., Hsu, W-J., Lau, C-T. (2005). Multi-home region location service for wireless ad hoc networks: an adaptive demand-driven approach, (IEEE) WONS, (pp.258-263).
6. Bae, Ihn-Han. (2007). An Adaptive Location Service on the basis of Diamond Quorum for MANETs , (IEEE) ICNC.
7. Basagni, S., Chlamtac, I., Syrotiuk, V., Woodward, A. B. (1998). A distance routing effect algorithm for mobility (dream), (ACM) MOBICOM, (pp.76–84).
8. Li, J., Jannotti, J., De Couti, D. S. J., Karger, D. R., Morris, R. (2000). A scalable location service for geographic ad hoc routing, (ACM) MobiCom, (pp.120-130).
9. Das, S. M., Pucha, H., Hu, Y. C. (2005). Performance comparison of scalable location service for geographic ad hoc routing, (IEEE) INFOCOM, (pp.1228-1239).
10. Leng, S., Zhang, L., Rao, J., Yang, J. (2006). A novel k-hop cluster-based location service protocol for mobile ad hoc networks, (IEEE) ITST, (pp.695-700).
11. Ratnasamy, S., Karp, B., Yin, L., Yu, F., Estrin, D., Govindan, R., Shenker, S. (2002). GHT: A geographic hash table for data-centric storage in sensornets, (ACM) WSNA.
12. Kumar, V., Das, S. R. (2004). Performance of dead reckoning-based location service for mobile ad hoc networks, (John Wiley and Sons) Wireless Communications and Mobile Computing, (Vol 4, No 2, pp.189-202).
13. Osmani, A., Haghighat, A. T. (2010). SALAM: Scalable @Location Advertisement Management in Ad Hoc Networks, (IEEE) CICSYN.
14. Osmani, A., Haghighat, A. T., Khezri, S. (2010). Performance Improvement of Two Scalable Location Services in MANET, (IEEE) CICN.
15. Cheng, H., Cao, J., Chen, H. H. (2007). GrLS: Group-based location service in mobile ad hoc networks, (IEEE) ICC, (pp.4734-4740).

16. Philip, S. J., Ghosh, J., Qiao, C. (2005). Performance evaluation of a multilevel hierarchical location management protocol for ad hoc networks, (Elsevier) Computer Communications, (Vol 28, No 10, pp.1110-1122).
17. Woo, S-C. M., Singh, S. (2001). Scalable routing protocol for ad hoc networks, (Springer) Wireless Networks, (Vol 7, No 5, pp.513-529).
18. Cheng, C. T., Lemberg, H. L., Philip, S. J., van den Berg, E., Zhang, T. (2002). SLALoM: A scalable location management scheme for large mobile adhoc networks, (IEEE) WCNC, (pp.574-578).
19. Liu, T. N., Hwang, S. I. (2006). On design of an efficient hierarchical location service for ad hoc network, (IEEE) ISWPC, (pp.1-6).
20. Jomeiri, A., Dehghan, M. (2008). Perfomance improvement of a grid based location service in manet, (IEEE) UBICOMM, (pp.165-170).
21. Stojmenovic, I., Liu, D., Jia, X. (2008). A scalable quorum based location service in ad hoc and sensor networks, (Inderscience Publishers) International Journal of Communication Networks and Distributed Systems, (Vol 1, No 1, pp.71-94).
22. Flury, R., & Wattenhofer, R. (2006). MLS: An efficient location service for mobile ad hoc networks. In Proceedings of the 7th ACM Conference on Mobile Ad Hoc Networking and Computing (pp.226-237).