Performance Comparison of Scalable Location Services for Geographic Ad Hoc Routing

Saumitra M. Das, Himabindu Pucha and Y. Charlie Hu
School of Electrical and Computer Engineering
Purdue University
West Lafayette, IN 47907
{smdas, hpucha, ychu}@purdue.edu

***Abstract*— Geographic routing protocols allow stateless routing in mobile ad hoc networks (MANETs) by taking advantage of the location information of mobile nodes and thus are highly scalable. A central challenge in geographic routing protocols is the design of scalable distributed location services that track mobile node locations. A number of location services have been proposed, but little is known about the relative performance of these location services. In this paper, we perform a detailed performance comparison of three rendezvous-based location services that cover a range of design choices: a quorum-based protocol (XYLS) which disseminates each node's location to $O(\sqrt{N})$ nodes, a hierarchical protocol (GLS) which disseminates each node's location to $O(\log N)$ nodes, and a geographic hashing based protocol (GHLS) which disseminates each node's location to $O(1)$ nodes.**

We present a quantitative model of protocol overheads for predicting the performance tradeoffs of the protocols for static networks. We then analyze the performance impact of mobility on these location services. Finally, we compare the performance of routing protocols equipped with the three location services with two topology-based routing protocols, AODV and DSR, for a wide range of network sizes. Our study demonstrates that when practical MANET sizes are considered, robustness to mobility and the constant factors matter more than the asymptotic costs of location service protocols. In particular, while GLS scales better asymptotically, GHLS is far simpler, transmits fewer control packets, and delivers more data packets than GLS when used with geographic routing in MANETs of sizes considered practical today and in the near future. Similarly, although XYLS scales worse asymptotically than GLS, it transmits fewer control packets and delivers more data packets than GLS in large mobile networks.

I. Introduction

A mobile ad hoc network (MANET) consists of a collection of wireless mobile nodes dynamically forming a temporary network without the use of any existing network infrastructure or centralized administration. In such a network, each node operates not only as a host, but also as a router, forwarding packets for other mobile nodes.

A fundamental challenge in MANETs is the design of scalable and robust routing protocols. Existing routing protocols fall into two categories: (1) *Topology-based* routing protocols which assume no knowledge of the mobile nodes' positions. Examples include proactive protocols such as DSDV [1], reactive protocols such as DSR [2], AODV [3] and hybrid protocols such as ZRP [4]. These protocols rely on discovering and maintaining global states in order to route packets. As a result, they have limited scalability. (2) *Geographic* routing protocols that utilize the location information of nodes available from positioning systems such as GPS to forward packets. Since local information (neighbor's locations) is used for global routing, geographic routing protocols have the potential to scale to a larger number of nodes than topology-based protocols.

However, before routing a packet using a geographic routing protocol, the source node needs to retrieve the location of the destination node. As such, a central challenge in geographic routing protocols has been the design of efficient location services that can track the locations of mobile nodes and reply to location queries for them. Such a service has to be scalable to preserve the scalability of geographic routing, and it itself should ideally operate using geographic routing. Overall, a protocol that implements an effective location service needs to be efficient, scalable, robust, load balanced, and locality aware.

Numerous protocols for location services [5], [6], [7], [8], [9], [10], [11], [12], [13], [14], [15], [16] have been proposed to solve the location tracking and retrieval problem in geographic routing for MANETs. Although each protocol is proposed along with some theoretical and/or simulation analysis, little is known about the relative performance of these protocols, especially in a realistic setting, i.e., a mobile environment.

This paper is the first to provide a realistic, quantitative analysis comparing the performance of a variety of scalable location service protocols. We focus on rendezvous-based location services as they are inherently more scalable than flooding-based ones. We perform an in-depth performance comparison of three representative rendezvous-based location services which cover a range of design choices: a quorum-based protocol (XYLS) which disseminates each node's location to $O(\sqrt{N})$ nodes, a hierarchical hashing-based protocol (GLS) which disseminates each node's location to $O(\log N)$ nodes, and a geographic hashing-based protocol (GHLS) which disseminates each node's location to $O(1)$ nodes.

The major contributions of this paper are:

- We present the first detailed comparison of three scalable location services that focuses on design tradeoffs rather

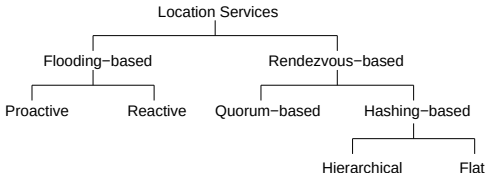


Fig. 1. A taxonomy of location services.

than protocol nuances.

- We present a quantitative model for measuring location service overhead.
- Since the quantitative model cannot predict the impact of node mobility and MAC layer interference, we perform a simulation evaluation of all three location services in a detailed simulator.
- We compare the performance of geographic forwarding when equipped with the three location services with two topology-based routing protocols, AODV [3] and DSR [2], for a wide range of network sizes to characterize the applicability of geographic routing protocols.

The rest of the paper is organized as follows. Section II presents a taxonomy of location services. Section III describes the three location services considered in our study. Section IV presents an analysis of the overhead of the three protocols. Section V describes the simulation methodology. Section VI presents the performance comparison of the three location services, and Section VII presents the scalability of geographic routing using these location services. Section VIII discusses related work and Section IX summarizes the paper's contributions.

II. A Taxonomy of Location Services

Figure 1 depicts a taxonomy of the location services proposed so far. At the top level, location services can be divided into *flooding-based* and *rendezvous-based* approaches. Flooding-based protocols can be further divided into proactive and reactive approaches. In the proactive flooding-based approach, each (destination) node periodically floods its location to other nodes in the network each of which maintains a location table recording the most recent locations of other nodes. The interval and range of such flooding can be optimized according to the node's mobility and the "distance effect". For example, flooding should be more frequent for nodes with higher mobility, and flooding to faraway nodes can be less frequent than to nearby nodes. DREAM [6] serves as a good example of proactive flooding-based location services. In reactive flooding-based approaches [5], if a node cannot find a recent location of a destination to which it is trying to send data packets, it floods a scoped query in the network in search of the destination.

In rendezvous-based protocols, all nodes (potential senders or receivers) in the network agree upon a mapping that maps each node's unique identifier to one or more other nodes in the network. The mapped-to nodes are the *location servers* for that node. They will be the rendezvous nodes where periodical location updates will be stored and location queries will be looked up. Two different approaches of performing the mapping, *quorum-based* and *hashing-based*, have been proposed. In the *quorum-based* approach [8], each location update of a node is sent to a subset (update quorum) of available nodes, and a location query for that node is sent to a potentially different subset (query quorum). The two subsets are designed such that their intersection is non-empty, and thus the query will be satisfied by some node in the update quorum. Several methods on how to generate quorum systems have been discussed in [8]. In this paper, we choose a design that is conceptually simple and efficient in terms of update and query complexities. In the so-called column-row quorum-based protocol [9], the location of each node is propagated in the north-south direction, while any location queries are propagated in the east-west direction. Effectively, each node's location is disseminated to $O(\sqrt{N})$ location servers.

In *hashing-based* protocols, location servers are chosen via a hashing function, either in the node identifier space or in the location space. Hashing-based protocols can be further divided into hierarchical or flat, depending on whether a hierarchy of recursively defined subareas are used. In *hierarchical hashing-based* protocols (for example, [13], [14]), the area in which nodes reside is recursively divided into a hierarchy of smaller and smaller grids. For each node, one or more nodes in each grid at each level of the hierarchy are chosen as its location servers. Location updates and queries traverse up and down the hierarchy. A major benefit of maintaining a hierarchy is that when the source and destination nodes are nearby, the query traversal is limited to the lower levels of the hierarchy. Since the height of the hierarchy is $O(\log N)$, effectively each node's location is disseminated to $O(\log N)$ location servers. The best example of hierarchical rendezvous-based location service is GLS [13], which is chosen in the comparative study in this paper.

In *flat hashing-based* protocols (for example, [10], [11], [12]), a well-known hash function is used to map each node's identifier to a *home region* consisting of one or more nodes within a fixed location in the network area. All nodes in the home region maintain the location information for the node and can reply to queries for that node; they serve as its location servers. Typically, the number of location servers in the home region is independent of the total number of nodes in the network, and thus effectively each node's location is disseminated to $O(1)$ location servers. In this paper, we propose a flat hashing-based protocol called *Geographic Hashing Location Service* (GHLS) that pushes the concept of geographic hashing to the extreme: the home region consists of only one node – the node whose location is closest to the hashed location. We use it as the representative of the flat hashing-based protocols.

Since flooding-based protocols scale poorly with the network size, we focus on rendezvous-based protocols in the rest of the paper.

III. Location Service Protocol Descriptions

In this section, we describe geographic forwarding and the three representative rendezvous-based location services under study, along with implementation decisions made in our

comparison study. For each protocol, we describe how it (1) chooses the location servers, (2) performs a location update, and (3) performs a location query.

A. Geographic Forwarding

All of the location services in our study are implemented by leveraging geographic forwarding. Each node determines its own geographic location – latitude and longitude – with the assistance of some global positioning systems such as GPS [17]. Our geographic forwarding is based on face routing [18]. Our implementation and simulation parameters are based on [19]. We modified the periodic beaconing mechanism to include 2-hop beaconing, i.e., in addition to its own location, each node also includes a list of its neighbors and their locations in the beacon. This two-hop beaconing is an integral part of GLS and is required for its operation. All the location services are evaluated using the same greedy geographic forwarding protocol.

B. Column-Row Location Service (XYLS)

XYLS is a proactive location service that disseminates locations in a direction such that a query can intersect it subsequently. It is similar to the column-row quorum-based location service originally proposed in [9].

1) Selecting location servers: At any given point of time, for each destination node, the nodes that are located along the north-south direction in the geographic area form the *update quorum* (location servers). Similarly, for each source node, the nodes that are located along the east-west direction form the *query quorum* for the node. Thus each query is expected to intersect one of the location servers.

2) Performing updates: 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. The thickness of the column controls the tradeoff between the update overhead and the robustness of the location service: the thicker the column, the higher the update overhead, but also the more likely a query will be satisfied. Our implementation constructs thicker columns using a low overhead approach: Each node selected as a location server in the north or south direction broadcasts the update to its one hop neighbors in addition to unicasting it to the next location server in the update direction. This increases the thickness of the column. An update is sent after every movement of distance d (update threshold). To facilitate caching of the location by location servers and forwarding nodes, each update packet has a timeout window which specifies the validity of the update. The timeout window is predicted based on the current mobility of the node and d. Updates are cached for the timeout period by all forwarding nodes.

3) Performing queries: 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.

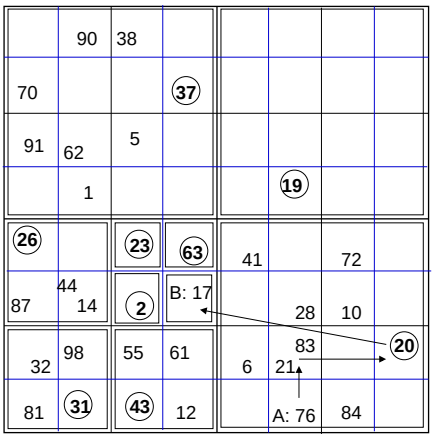


Fig. 2. Location update and query in GLS.

C. Grid Location Service (GLS)

GLS [13] is a hierarchical hashing-based location service. It is motivated by consistent hashing [20] and performs hashing in the node identifier space, as opposed to the geographic location space. The geographic area in which the nodes reside is recursively divided into a hierarchy of smaller and smaller squares, and the smallest square is referred to as an order-1 square. The four order-$(n-1)$ squares sharing the same order-n square as the parent square are *sibling* squares. For each node, one node from each sibling square of the node's square at each hierarchy level is selected as its location server. The geographic forwarding for GLS uses the two-hop beaconing protocol as described in Section III-A. This ensures that each node knows the location of all nodes in its order-2 square. Figure 2 (from [13]) shows an example of a hierarchy. The node *closest* to a node X in the circular identifier (ID) space is the node with the least ID greater than X. GLS assigns a unique, random ID to a node, by applying a strong hash function to the node's *unique name*. The unique name could be a host name, IP address, or MAC address. For each node B, GLS selects a location server in each sibling square at each hierarchy level as the node whose ID is closest to node B in that square. In the figure, the location servers at B's order-1 sibling squares are nodes 2, 23, 63, respectively. In B's order-2 sibling squares, the location servers are 26, 31, and 43, respectively.

1) Performing queries: We first describe how a location query is routed, assuming all nodes have updated their location servers with their identifiers and locations. At each step, a query makes its way to the closest node at successively higher hierarchical levels. If node A (76) requests for the location of node B (17), it looks up the same order-1 square for the best node, in this case, itself. It then looks for the best node in each of its order-1 sibling squares, in this case, node 21 which it knows about from two-hop beaconing. In the next step, node 21 finds node 20 which is the best node in node A's order-2 sibling squares, and node 20 has the location of node B since

it is one of node B's location servers in B's order-3 sibling squares. Recall 21 is the best node in its own order-2 square. This ensures that no nodes in that square have IDs between 17 and 21. Now consider a node X in 21's sibling order-2 squares. If X's ID is between 17 and 21 then X must have chosen node 21 as a location server since there are no other nodes closer to X than 21 in 21's order-2 square. Thus node 21 knows all nodes in its order-2 sibling squares that lie between 17 and itself, including the minimum such node, which is 20 in this case. Similarly, at the next step, node 20 knows all nodes in its order-3 sibling squares between 17 and itself. In other words, node 20 is a location server for such nodes, including node 17.

Once the query reaches node 20, it forwards the query to the destination node 17 so that the destination node can reply to the source using geographic forwarding with its current location. This detour to the destination node is necessary in GLS due to the distance effect in location updates as explained below. Note that a query or a query reply stays inside the smallest square containing the source and the destination.

2) Performing updates: Similar to how queries are routed, a node X inserts location updates to all of its location servers, i.e., the node whose ID is closest to X in each sibling square at each level, without knowing who the location servers are. Since there are 3 sibling squares and thus 3 location servers at each level of the hierarchy, and the hierarchy height is $O(\log_4 N)$, the total number of location servers per node is $O(3 \log_4 N)$. Each update packet from a node carries its ID, its current location, and a timeout window explained below. Similarly as in XYLS, updates are cached at forwarding nodes.

3) Location update interval: The location update packets for a node are sent out at a rate proportional to the node's speed, and the range of the updates takes into account the distance effect [6]. A node updates its location servers in order-i sibling squares after each movement of $2^{i-1} \cdot d$, where d is the update threshold. Similarly as in XYLS, a timeout is advertised except that the timeout calculated is based on the order of the location server to be updated (e.g., for location servers in order-2 sibling squares, the timeout is based on the time taken to travel $2 \cdot d$ distance). Since a node updates its high-order servers much less often, those servers can have fairly inaccurate location information, and thus they need to forward a query to the destination node which will then reply to the querying node with its current location.

4) Implementation changes: The ns-2 implementation of GLS provided by its authors [13] was faithfully ported to Glomosim [21]. To confirm the correctness of the porting, we ran simulations with the same parameters using the original ns-2 code provided by the authors and our Glomosim implementation, and the results from the two implementations matched closely.

D. Geographic Hashing Location Service (GHLS)

Compared to hierarchical hashing-based protocols, flat hashing-based protocols avoid the complexity of maintaining a hierarchy of grids and the consequent maintenance due to nodes moving across grid boundaries. Previous flat hashing-based protocols, however, still introduce a geographic region (either a rectangle or a circle surrounding the hashed location) as the home region for the location servers for each node. In this paper, we propose a flat hashing-based protocol called *Geographic Hashing Location Service* (GHLS) that pushes the concept of geographic hashing to the extreme: the home region consists of only one node – the node whose location is closest to the hashed location.

1) Selecting location servers: In GHLS, each node is assigned a single location server, by hashing the node's *unique name* into a location in the geographic area, and designating the node whose location is currently closest to the hashed location as the node's location server. The unique name could be a host name, IP address, or MAC address. Thus, each node may be a location server for zero or more nodes, whereas each node stores its location on only one location server.

2) Performing updates: Similarly as in GLS, the updates of the nodes are reactive to their movement patterns and cached at forwarding nodes. A key difference between the update protocols for GLS and GHLS is that GHLS does not have multiple timeout windows and update frequencies for multiple location servers since there is only one location server per node. An update is sent after every movement of distance d.

3) Performing queries: To look up the location for a destination node, a sending node hashes the node's unique name using the same hashing function to generate the location, and sends a query packet which will be forwarded towards the server location via geographic forwarding. Upon reaching the location server, the stored location information is sent via a query reply packet from the location server back to the source of the query packet, again via geographic forwarding.

4) Decoupling server and destination movement: Due to node mobility, location information stored on a server may need to be migrated to other nodes that become closer to the location. One way to achieve this is to rely on a new update packet to arrive from the destination node periodically to refresh this state. This is difficult as it requires each node to adapt its update frequency to the movement of its location server. GHLS solves this problem by decoupling the movement of location servers from the movement of the originators of the updates via a lightweight *handoff* procedure. Every node that acts as a location server executes the following procedure when it receives a beacon packet from another node: It checks whether the source of beacon packet is closer to the hashed location of each of the sources for whom it is acting as the location server. If the source of the beacon packet is a better match for a subset of locations it stores, the node hands off these locations to the source of the beacon packet.

5) Incorporating locality: A potential drawback of flat geographic hashing protocols is that a location server can potentially be far away from both the source and destination nodes, causing update and query operations to incur high overhead. To alleviate this problem, GHLS uses a hashing function that generates locations within a smaller region near the center of the whole region. We define the ratio of the

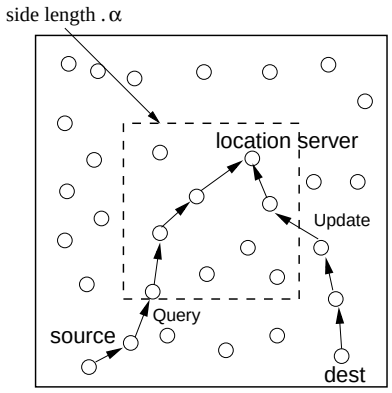


Fig. 3. Location update and query in GHLS. The location servers are inside the scaled location server region, surrounded by the dashed-line box.

side length of the scaled location server region and that of the whole area as the *scaling factor* α, and denote the scaled location server region as the α *region*. Figure 3 shows an example of a scaled location server region.

Using a scaled location server region can potentially create service load imbalance among the nodes in the whole network, i.e., higher load in the scaled region. We evaluate and compare the load balance in all three location services in our simulation study in Section VI.

6) Handling Node Failure: Consider a node X that is currently a location server in GHLS. If X fails, queries that arrive between the time that X fails and the next updates from nodes whose locations are stored at X arrive could potentially fail. However, caching of updates allow other nodes to answer queries despite failure of the location server. Additionally, the locations stored by X can also be piggy-backed in the beacon packets and stored at all the neighbors of X for added reliability.

IV. Analysis

In this section, we analyze the performance of the three location services under study. The main results are summarized in Table I.

A. Analysis for Static Networks

We first compare the amount of location service protocol packets initiated and transmitted in the three location services. The total number of location service protocol packets transmitted is referred to as the *LS protocol overhead*. To make the analysis tractable, we assume the network is static, the nodes are uniformly distributed in the geographic area, and location updates are not cached at forwarding nodes. To aid our analysis, we define the following parameters: Let the update frequency be f updates per second, the side length of order-1 square in GLS be l, and the height of the hierarchy be h, i.e., the whole area is an order-h square. Thus the side length of order-i square will be $2^{i-1} \cdot l$, and the side length of the whole area is $2^{h-1} \cdot l$. In GLS, l is chosen to be the transmission range of 802.11, i.e., 250m. In the following, we use this assumption as well as the notion of order-1 squares to simplify our analysis of all three protocols, even though the notion of order-1 squares does not exist in XYLS and GHLS.

TABLE I

Comparison of the three location services.

Metrics	XYLS	GLS	GHLS
Type	quorum	hier. hashing	flat hashing
Updates per interval	1	$3 \cdot (2 - \frac{1}{2}^{h-2})$	1
Avg. update transmissions per interval	2^{h+1}	$\frac{3}{2} \cdot (2+\sqrt{2}) \cdot (h-1)$	$\frac{1}{3}2^{h-1}\sqrt{2}$
Average query path length	$\frac{2}{3} \cdot 2^h$	$\frac{2}{3}2^{h-1}\sqrt{2}$	$\frac{1}{3}2^{h-1}\sqrt{2}$
Average query rep. path length	$\frac{1}{3} \cdot 2^{h-1}$	$\frac{1}{3}2^{h-1}\sqrt{2}$	$\frac{1}{3}2^{h-1}\sqrt{2}$
Robustness	high	medium	high

In XYLS, one update is sent at each update interval. The update is propagated along the column, which covers 2^{h-1} order-1 squares. Thus in theory, an update costs 2^{h-1} transmissions. However, since our implementation of XYLS update (or query) chooses the node closest to the starting node's x-coordinate (or y-coordinate), the average stride each hop makes is about half (the theoretically exact number is $\frac{4}{3\pi}$) of the transmission range, or half the side length of an order-1 square. Thus an update travels about 2^h hops in the north-south direction. Since a unicast and a broadcast are performed at each hop, the total number of transmissions is 2^{h+1}. For each query, two packets are sent along the two directions, only unicast is performed at each hop, and one of the two packets will not be forwarded further when it reaches a server. The expected number of transmissions per query is $\frac{2}{3} \cdot 2^h$ (via simple integration). A query reply travels about $\frac{1}{3} \cdot 2^{h-1}$ hops on average since it goes along one direction towards the source.

In GLS, at each update interval, each node sends 3 updates to the 3 location servers in order-1 sibling squares. At every two update intervals, 3 updates are sent to the 3 location servers in order-2 sibling squares, and so on. At every 2^{h-2} update intervals, 3 updates are sent to the 3 location servers in order-$(h-1)$ sibling squares. Thus the average number of update packets per interval from each node is $3 \cdot 1 + \frac{1}{2} \cdot 3 + \frac{1}{4} \cdot 3 + ... + \frac{1}{2^{h-2}} \cdot 3 = 3 \cdot (2 - \frac{1}{2}^{h-2})$. The numbers of hops traveled by the update packets to the three order-1 sibling squares are 1, 1, and $\sqrt{2}$, approximately, and the average number of hops traveled between larger squares grows proportionally to the side lengths of the squares. Thus the average number of hops traveled by all update packets per interval is $(2+\sqrt{2}) \cdot \frac{3}{2} + \frac{1}{2} \cdot (2+\sqrt{2}) \cdot 2 \cdot \frac{3}{2} + \frac{1}{4} \cdot (2+\sqrt{2}) \cdot 2^2 \cdot \frac{3}{2} + ... + \frac{1}{2^{h-2}} \cdot (2+\sqrt{2}) \cdot 2^{h-2} \cdot \frac{3}{2} = \frac{3}{2} \cdot (2+\sqrt{2}) \cdot (h-1)$. The factor of $\frac{3}{2}$ for each term is explained as follows. Let the lowest-order sibling squares containing the destination and the location server be order-k squares X and Y, respectively. The update packet is first forwarded in a straight line towards Y while inside X. Once crossing the boundary between X and Y, it is forwarded in a sequence of steps in traversing up the

hierarchy starting from an order-1 square and reaching the order-$(k-1)$ square containing the location server. Since the sequence of steps are directionless and the expected distances they travel are recursively doubled, they incur an approximate factor of 2 overhead compared to the direct distance between the starting point and finishing point inside square Y. The factor of 2 overhead while traversing inside Y and the straight line path inside X together contribute to the factor of $\frac{3}{2}$ for each term in the above average hop count formula for update packets.

For queries in GLS, the average number of hops they travel can be approximated as twice the average distance between two random points in a square of area $2^{h-1} \times 2^{h-1}$, which is about $\frac{2}{3}2^{h-1}\sqrt{2}$. The factor of two comes from the fact that the location server has to forward the query to the destination node. Similarly, query reply takes an average $\frac{1}{3}2^{h-1}\sqrt{2}$ hops as it travels from the destination to the source of the query.

In GHLS, one update is sent at each update interval. Assume the location server region is not scaled down, i.e., $\alpha = 1$. Again, the average number of hops it travels is the same as the average distance between two random points in a square of area $2^{h-1} \times 2^{h-1}$, which is about $\frac{1}{3}2^{h-1}\sqrt{2}$. Similarly, the average number of hops a random query or query reply travels is $\frac{1}{3}2^{h-1}\sqrt{2}$.

The following conclusions can be drawn from the above analysis: (1) GLS will lose to GHLS by a factor of 2 and will outperform XYLS by a factor of $\sqrt{2}$ in average query path length. (2) GLS and GHLS have comparable query reply path lengths and both lose to XYLS by a factor of $\sqrt{2}$. (3) The average numbers of update transmissions per interval for GHLS and XYLS grow as $O(2^h)$ while that of GLS grows as $O(h)$. Thus, theoretically, GLS should outperform XYLS and GHLS in terms of update transmissions as the network is scaled. To estimate the network size beyond which GLS outperforms GHLS and XYLS, in Figure 4 we plot the formulas for the average number of update transmissions per interval for all three protocols as a function of the grid hierarchy height h and the network size N assuming an adequate density of 100 nodes/km^2. The plots show that while GLS outperforms XYLS for all network sizes, it has a larger number of update transmissions than GHLS until the height of the hierarchy reaches beyond 7, i.e., when the network size reaches beyond 25,600. It is not clear whether ad hoc networks of such sizes can be realized or have applicability in any practical scenarios.

B. Effect of Mobility

We qualitatively compare the impact of mobility on the performance of the three location services. In all three approaches, each node in its role as a destination node needs to send a new update whenever it moves the threshold distance, or when the timeout window expires first due to the change of speed. Thus, there is a proportional increase or decrease of the amount of protocol packets and transmissions due to node mobility.

In XYLS, mobility has no additional effect on the communication complexity. In GHLS, a node in its role as a location server also needs to hand off its local location database to neighboring nodes. In GLS, because the hashing is in the node ID space, there is no need for handoff. However, since the hashing is performed individually for each sibling square at each level of the hierarchy, every time a node crosses a grid boundary, it has to update some of its location servers accordingly. In the worst case, when a node crosses a boundary between two highest level grids, it has to select and update all of its $O(\log N)$ location servers. Furthermore, to allow query resolution despite mobility of location servers, a node has to leave "forwarding pointers" via broadcasts to nodes in its previous order-1 square [13]. Such forwarding pointers are continuously propagated for newcomers by piggy-backing them on beacon packets. As a result, the size of the beacon packets tends to grow.

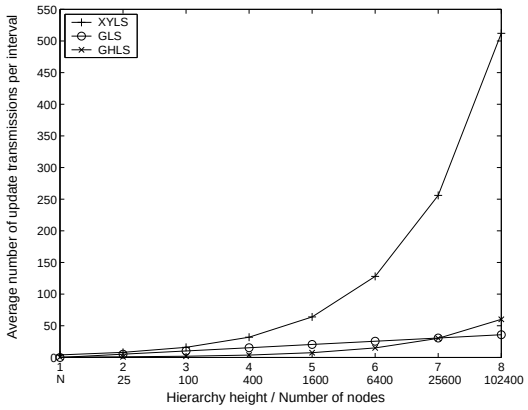


Fig. 4. Average number of update transmissions per interval as a function of the hierarchy height.

V. Experimental Methodology

We implemented all the protocols studied as well as geographic forwarding in the Glomosim [21] simulator. Glomosim is a widely used wireless network simulator with a comprehensive radio model. We use a 802.11 radio model with a nominal bit-rate of 11Mbps and a transmission range of 250m. For all three location service protocols, we chose a beacon period of 2 seconds. The mobility scenarios use the modified random way point mobility model [22]. For mobile networks, all simulations use a pause time of 0 second with nodes moving at speeds randomly chosen between 1 and 9m/s. The simulation duration is 300 seconds.

The following metrics are evaluated for the location service protocols: (1) *LS protocol overhead* – The number of location service (LS) protocol packets transmitted, with each hop-wise transmission of the protocol packet counted as one transmission. Note that the LS protocol overhead excludes the overhead due to beaconing. In Sections VI-D and VII, we evaluate the *normalized* LS protocol overhead (normalized by the received data packets); (2) *Packet delivery ratio* (PDR) – The ratio of the data packets delivered to the destinations to those generated by the CBR sources; (3) *Query success ratio* (QSR) – The ratio of the number of query replies received at the sources to the number of location queries initiated by the sources.

VI. Location Service Performance

In this section, we first evaluate the three location service protocols in a 600-node static and a 600-node mobile network to compare their relative performance and analyze the impact of mobility on the LS protocol overhead. We then study their scalability by evaluating their performance in networks ranging from 100 to 2000 nodes.

The query pattern in this section is chosen to study the efficiency of the query and update mechanism of all the protocols. Every node in the network initiates 15 queries to look up the location of randomly chosen (unless noted otherwise) destinations at times randomly distributed between 30 and 300 seconds. Thus the number of queries originated are $N \cdot 15$. Also, if a query is not successful, no retry is initiated. To measure the accuracy of the query reply, when the query reply is received, each source sends a single data packet of size 128 bytes to that destination using the replied location. The PDR obtained in this scenario is an indication of the accuracy of the destination location since for all location services, the same greedy geographic forwarding is used to route the data packets. For all protocols, the value of d (update threshold) was fixed at 200m. For GHLS, we also vary α as 0.5 and 1. The node density in all the scenarios is kept constant at 100 nodes/km^2.

A. Static Networks

We first compare the performance of the three protocols in a static network to separate the effects of mobility on the performance of the three protocols. Table II shows the various components of the LS protocol overhead for a static network and a mobile network of 600 nodes. The terrain is a square of 2500mx2500m. To construct a perfect grid, GLS pads the area to a virtual area of 4000mx4000m, and constructs a grid hierarchy of height 5. To decouple the LS protocol overhead due to the update frequency from that due to mobility, for the static network, we assume a constant update interval of 40 seconds for all three protocols. This interval is based on the average time taken to cross d=200m (update threshold) in the corresponding mobile network where nodes travel with random velocities between 1 m/s and 9m/s. For GLS, this interval corresponds to the update interval of the order-1 sibling location servers. The higher order servers are updated at proportionally lower rates (power-of-two multiples of 40 seconds).

Table II shows that in a static network, XYLS has the highest overhead among all three protocols. This is primarily due to the larger number of updates that are propagated up and down the column. GLS has a lower number of updates than XYLS. However, it has almost 5 times more updates than GHLS because it has to update multiple location servers at multiple levels. As a result, GHLS-1 and GHLS-0.5 have 2.55 and 3.11 times fewer update transmissions than GLS. The number of query transmissions by GLS is the highest among all three protocols since the query has to reach the actual node and takes detours along the way. The number of query reply transmissions in GLS and GHLS-1 are similar while XYLS is the lowest. GHLS-0.5 has a lower number of query reply transmissions than GHLS-1 due to the use of the α *region*.

B. Mobile Networks

To measure the impact of node mobility on the three location service protocols, we introduced mobility into the same 600-node scenario. Nodes now move according to a random waypoint model with velocities chosen randomly between 1 m/s and 9 m/s and with a pause time of 0 second. Table II shows a detailed breakdown of the LS protocol overhead for all three protocols.

1) Comparison with Static Networks: We first compare the overhead for each location service protocol in the mobile network relative to in the static network. The following observations can be made.

First, the number of updates sent for GLS increases by about 81.89% while the numbers for XYLS and GHLS both decrease by about 7.77%. The reason for the slight decrease in updates sent for XYLS and GHLS is as follows. In the mobile network, the nodes have different velocities randomly distributed between 1 m/s and 9 m/s. Thus for the same update distance (200m), different nodes send different numbers of updates. In contrast, in a static network, every node sends an update every 40 seconds. The 80% increase in the number of updates in GLS is caused by frequent grid boundary crossing by the mobile nodes. In GLS, whenever a node crosses grid boundaries, it needs to update its old (high-order) location servers as well as select new (low-order) servers since its position in the hierarchy has changed. This causes an increased number of updates to be initiated in a mobile network for GLS. We found that grid crossings account for approximately 50% (21,000 packets) of the total updates sent. These updates do not occur in a static network and occur less frequently when mobility is lower.

Second, the average number of transmissions per update in GLS grows from 3.39 in the static network to 7.91 in the mobile network, whereas the same metric stays roughly the same in XYLS and GHLS. The reason for this is that in GLS, node mobility leads to temporal node state inconsistencies which result in detours in routing update packets towards the location servers. Thus, mobility has a higher impact on the protocol overhead of GLS than on those of XYLS and GHLS.

Third, the number of queries sent for all three protocols are reduced by on average 2-6% in the mobile network. This is because mobility helps to improve the connectivity and the effectiveness of caching over time, resulting in more location queries being resolved using the cache at the source. However, the average number of transmissions per query for GLS increases by 12%. This shows the susceptibility of the GLS query process to node mobility. In particular, when location servers in GLS move, forwarding pointers may have to be followed to resolve a query thereby increasing the average number of transmissions per query. Similarly, in XYLS, the average number of transmissions per query increases by 21%. This can be explained as follows. Stale neighbor location information due to mobility hampers the accurate choice of

TABLE II

LS PROTOCOL OVERHEAD BREAKDOWN FOR A 600 NODE NETWORK.

	Static Network				Mobile Network			
	XYLS	GLS	GHLS-1	GHLS-0.5	XYLS	GLS	GHLS-1	GHLS-0.5
Beacon Ovhd.	89,400	89,400	89,400	89,400	89,400	89,400	89,400	89,400
LS Protocol Ovhd.	311,181	254,504	134,643	112,317	302,771	526,327	134,962	115,046
UPDATES								
Sent	19,200	24,466	4,800	4,800	17,708	44,502	4,427	4,427
Txed	184,319	83,099	32,611	26,752	166,798	351,940	30,651	24,396
QUERIES								
Sent	14,368	8,055	8,578	8,613	13,498	7,445	8,459	8,464
Txed	97,232	120,218	51,639	43,829	110,452	124,534	48,139	38,092
REPLIES								
Sent	6,893	7,073	8,386	8,308	7,191	6,793	8,190	8,361
Txed	29,630	51,187	50,392	41,736	25,521	45,980	45,815	37,235
HANDOFFS	-	-	1	0	-	-	10,357	15,323
FP UPDATES	-	0	-	-	-	3,873	-	-

the next hop in forwarding query packets. The impact is exacerbated by the fact that a query in XYLS is forwarded to the neighbor node that has the closest x-coordinate as the source at each hop in order to preserve the east-west direction of propagation, as explained in Section IV-A. Thus, at every hop, there is a possibility of choosing a less optimum next hop causing packets to take extra hops to reach the column. In contrast, the impact of mobility is much less in GHLS where queries take the maximum possible stride at each hop towards the location server. In fact, for both versions of GHLS, the average number of transmissions per query is slightly reduced in the mobile scenario as compared to in the static scenario.

Last, since all protocols use geographic forwarding for query replies, the reply overhead remains largely unchanged in the mobile scenario. In addition, mobility introduces the overhead of handoff packets in GHLS. Similarly, in GLS, when a node crosses grid boundaries, it broadcasts a one-hop forwarding pointer information packet (FP Update) to the nodes in its previous grid.

In summary, the increase in the average number of updates, in the average number of transmissions per update, and in the average number of transmissions per query in GLS in the mobile network suggests that GLS is the most susceptible to node mobility.

2) Comparison among the Three Protocols: Table II shows that while the protocol overhead of GLS is 1.89 times that of GHLS-1.0 and 0.81 times that of XYLS in the static network, these ratios increase to 3.89 and 1.74 in the mobile network. In particular, while the number of update transmissions of GLS is 2.54 times that of GHLS-1 and 0.45 times that of XYLS in the static network, these ratios increase to 11.48 and 2.11 in the mobile network. Thus GLS is significantly more adversely affected by mobility than XYLS and GHLS. This suggests that if for static networks, the network size has to be 25,000 nodes for GLS to tie GHLS in terms of update transmissions, then for mobile networks, the network size has to be much larger for GLS to catch up with GHLS in terms of update transmissions. In moving from the static network to the mobile network, the protocol overhead of XYLS stays roughly the same while that of GLS is more than doubled. As a result, XYLS has lower protocol overhead than GLS in the mobile scenario although it has higher protocol overhead than GLS in the static network.

TABLE III

AVERAGE QUERY AND QUERY REPLY HOP LENGTHS.

Protocol	XYLS	GLS	GHLS-1	GHLS-0.5
Query Length	4.53	10.42	5.05	4.03
Query Rep. Length	3.34	5.91	5.05	4.04

Table III shows the average hop lengths of resolved queries and of their replies. Note that for XYLS, the query length refers to the number of hops traveled by the query (out of the two sent in opposite directions) that reaches a location server. Since in GLS, queries have to travel from the location server to the destination, whereas for the other two protocols, they just need to travel to the location servers, the average query length in GLS is twice that in XYLS or GHLS. In GHLS, a query reply travels the same number of hops as the query. In GLS, the query reply travels back from the destination node to the source node, and thus the average length is about half of the query length. In XYLS, a query reply takes fewer hops on average than a query. This is because queries in XYLS do not take the maximum stride at each hop (explained in Section IV-A) while for query replies each next hop chosen takes the maximum possible stride back towards the source. Note that the query and query reply path lengths in Table III differ from those in Table II since they consider only *successfully resolved* queries.

3) Load Balance: When GHLS is used with $\alpha < 1$, nodes in the α region potentially bear more update and query traffic than a node outside. To measure this effect, we repeated the same 600-node experiment in the previous section (with mobility), e.g., keeping the same query rate, except the simulation is extended to 1500 seconds so that the results better reflect long term mobility. For all protocols, we logged the LS protocol packet transmissions inside and outside the α region where α is chosen to be 0.5. The measured results are shown in Table IV.

In XYLS, the *overhead ratio* (i.e. the ratio of the overhead outside to the overhead inside the α region) is the same as the ratio of their areas (i.e. 3:1). XYLS exhibits near perfect load

TABLE IV

LS OVERHEAD INSIDE AND OUTSIDE THE α = 0.5 REGION.

	LS Protocol Overhead	
	Inside	Outside
XYLS	334,605	1,044,216
GLS	729,951	1,308,840
GHLS-1	331,971	299,009
GHLS-0.5	416,635	119,828

TABLE V

EFFECT OF LOCALITY IN TRAFFIC PATTERNS.

	Queries	Replies	Updates	Total
Random				
XYLS	97,232	29,630	184,319	311,181
GLS	120,218	51,187	83,099	254,504
GHLS-1	51,639	50,392	32,611	134,643
GHLS-0.5	43,829	41,736	26,752	112,317
10/15 Local				
XYLS	73,088	16,988	184,319	274,395
GLS	60,756	30,675	83,099	174,530
GHLS-1	41,325	41,383	32,611	115,319
GHLS-0.5	34,541	34,446	26,752	95,739
15/15 Local				
XYLS	41,932	4,069	184,319	230,320
GLS	13,113	4,702	83,099	100,914
GHLS-1	22,003	21,332	32,611	75,946
GHLS-0.5	18,762	18,481	26,752	63,995

balance because independent of where the nodes are, their updates and queries are sent across the network along the x- and y-dimensions. For GLS, the overhead ratio is 1.7:1, suggesting an imbalance and higher load in the α region. This occurs because in GLS, some updates are sent to faraway location servers in diagonal grids resulting in more load in the center of the network. In GHLS-1, the overhead ratio is approximately 1:1 also suggesting an imbalance. This occurs because of the fact that lines connecting points chosen at random within a square are likely to cross the center of the square. GHLS-0.5 increases the overhead inside the α region by 25% compared to GHLS-1. However, it also reduces the overhead outside the region by 60%. Thus, the choice of α decides a tradeoff between the total overhead incurred in the whole network and the load inside the α region.

The most important observation made from Table IV is that despite the load in GHLS being more imbalanced than in XYLS and GLS, GHLS has significantly lower LS protocol overhead either outside (compared to XYLS) or both inside and outside (compared to GLS) the α region.

In GHLS-0.5, although the average storage load on each node in the α region is 4 location entries, while nodes outside store no entries, each stored entry is of size 20 bytes, making the storage load on the nodes inside rather insignificant. In contrast, GLS stores each entry on $O(3 \cdot \log_4 N)$ servers, and thus the average number of entries stored at any node in the network can easily exceed 4 for medium to large networks.

C. Impact of Locality in Traffic Patterns

In this section, we investigate the performance of the protocols under traffic patterns with locality. Such patterns are in favor of protocols such as GLS and XYLS which exhibit locality during query lookups. We consider three traffic patterns with increasing locality. The first pattern is the random traffic pattern used in the previous experiments. In the second pattern, denoted as *10/15 Local*, 10 out of the 15 queries each source initiates go to nearby nodes ($\leq$ 750m away) while the remaining 5 go to random nodes. In the third pattern, denoted as *15/15 Local*, all 15 queries go to nearby nodes ($\leq$ 750m away). The experiment uses a static network of 600 nodes as before. A static network was used since it is difficult to instrument locality of traffic as nodes move; to preserve locality of traffic as nodes move, the traffic pattern would have to be continuously changed based on the new positions of the nodes.

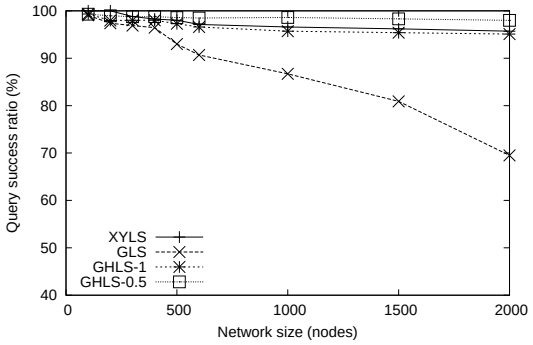
Table V shows the overhead breakdown for the three protocols under the three traffic patterns. As expected, due to their protocol features, the query overhead of GLS and XYLS decreases as the locality in the traffic increases. This is because in GLS, the queries and replies are restricted to the largest square containing the source and the destination and therefore incur lower overhead when traffic is local. In XYLS, the query and reply distances and consequently overhead are reduced since the columns of the destinations are nearby. Although GHLS has no specific feature to exploit locality apart from scaling the α region, its query overhead is reduced due to increased effectiveness of caching. This occurs because as the locality in the traffic increases, more nearby nodes look up locations of a similar set of destinations. Note that the increased locality in the traffic also improves the effectiveness of caching in both XYLS and GLS.

Overall, with extreme locality in the traffic (15/15 Local), GLS incurs lower query overhead than GHLS, which still has lower overhead than XYLS. The update overhead of GLS and XYLS, however, remain higher than that of GHLS since it is independent of the locality in the traffic. Because of the unchanged large gap between update overheads, GHLS has lower overall protocol overhead than GLS and XYLS. Note that this result was observed for very high query rate; on average every node issues a query every 20 seconds, which favors GLS and XYLS.

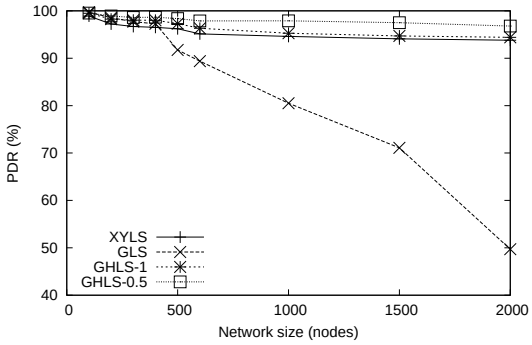
D. Scalability

In this section, we compare the scalability of the three location service protocols in terms of the network size. Figure 5 shows the performance of the three location services as the network size is scaled up from 100 to 2000 nodes. The same random waypoint model and the same query traffic pattern as in Section VI-B are used. A pause time of 0 second is chosen.

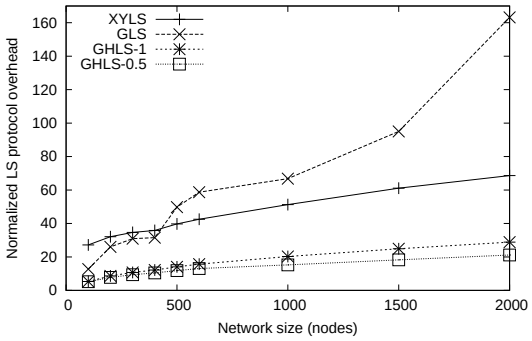
Figure 5(a) depicts the query success ratio of location queries as a function of the network size for all the location services. Similar to the study in [13], queries are not retransmitted, so a success means a success on the first try. The results show that for a small network size of 100 nodes, all protocols have identical query success ratios of close to 100%. As the network size increases, the QSR of GLS drops quickly, reaching 70% at 2000 nodes, whereas GHLS-0.5 achieves a



(a) Query success ratio



(b) PDR



(c) Normalized LS protocol overhead

Fig. 5. Query success ratio, PDR and Normalized LS protocol overhead for a pause time of 0 second.

consistently high QSR (above 98%). XYLS performs similarly to both versions of GHLS for small to medium networks but has a lower QSR than GHLS-0.5 for larger networks.

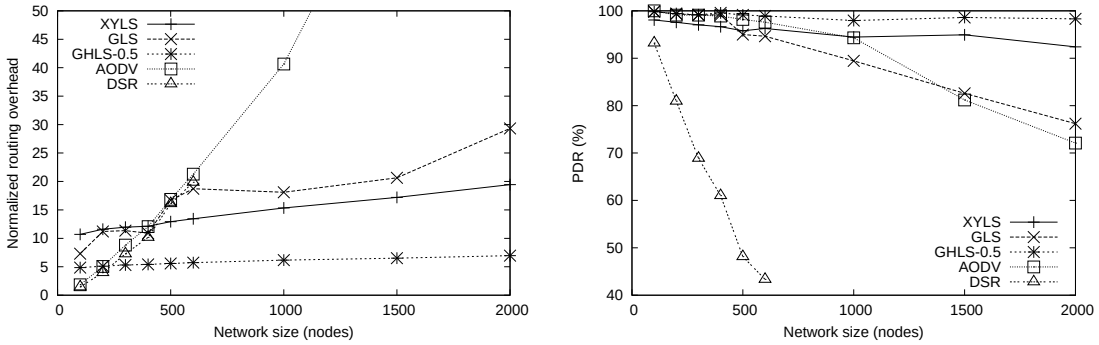
To measure the qualities of the location retrieved, each location query is triggered with a single data packet, and the packet delivery ratio of the data packets is measured. Note that we do not show the geographical deviation of the location received in the query reply from the actual location of the destination since geographic forwarding is quite robust against slight inaccuracies in the location of a destination. Figure 5(b) plots the PDRs under the three location services. It shows for all location services except GLS, the PDR curve closely resembles the corresponding QSR curve, suggesting almost all locations retrieved are accurate enough for successful data packet delivery. In GLS, as the network size increases beyond 600 nodes, the PDR drops significantly. This is caused by the corresponding sharp increase in the protocol overhead (Figure 5(c)) which results in increased congestion.

Figure 5(c) depicts the protocol overhead of all three location services. It shows that although the location services have similar overhead for a small network of 100 nodes, the overhead of both versions of GHLS is significantly lower than those of GLS and XYLS in larger networks, and the gap between GHLS and GLS widens as the network size is increased. More importantly, GHLS not only incurs lower protocol overhead, it also achieves higher QSR and PDR than GLS and XYLS. XYLS has lower overhead as well as higher PDR/QSR than GLS in larger networks.

VII. Geographic Routing Performance

In this section, we combine each location service with geographic forwarding to provide a data routing protocol. We then evaluate the data delivery performance of these protocols across a wide range of network sizes. The number of nodes in the network is increased from 100 to 2000 nodes while keeping the node density constant at 100 nodes/km^2. A pause time of 0 second is chosen. Half of the nodes in the network are sources of data traffic, each originating one connection to a random destination and sending 128-byte packets at a rate of 4 packets/second for a duration of 20 seconds. Similarly as in [13], to reduce the effects of caching and fully stress the routing protocols, we restrict the number of times a node can be chosen as a destination to 3. Unlike in the previous section, an exponential backoff of location queries is done by each protocol whenever replies are not received. Additionally, all protocols maintain a table of live connections so that once a CBR flow is initiated, the destination and source can coordinate with each other, avoiding location queries for the duration of the flow, same as in [13]. GHLS used an α of 0.5.

Our aim was to measure the performance of location-based protocols by using a normalized routing overhead metric that takes into account location service overhead as well as geographic beaconing overhead. This allows us to understand the applicability of location-based protocols to different network configurations. We used two representative non-location-based routing protocols, AODV and DSR, in addition to three routing



(a) Normalized routing overhead

(b) PDR (%)

Fig. 6. Normalized routing overhead and PDR for a pause time of 0 second. The routing overhead includes transmissions of all non-data packets.

protocols based on the three location services studied in this paper. Note that although this comparison is unfair to the non-location-based protocols, it gives us an idea as to at what network sizes, location-based routing protocols can provide a large enough gain to justify the costs of including positioning systems on the mobile nodes. The AODV implementation provided by the authors of AODV follows draft version 13 of the protocol and incorporates local repair and query localization to help its performance in large networks. We augmented DSR with a graph-based cache [23] to reduce its overhead so as to allow it to scale further than the results presented in [13].

Figure 6 shows the simulation results. The most important observations from the results are as follows. First, GHLS has the lowest routing overhead among all protocols for network sizes larger than 300 nodes, while achieving the highest packet delivery ratio. Second, AODV performs well when compared to location-based protocols. For network sizes up to 400 nodes, AODV has lower overhead than GLS and XYLS while delivering a similar amount of data packets. This occurs due to the reactive design of AODV in contrast to the proactive location update mechanism used in the location-based protocols. In addition, optimizations like query localization and local repair reduce the routing overhead of AODV in the presence of mobility, especially in large networks. The results for DSR show that although it performs better when augmented with a graph cache than the results presented in [13] which used a path cache, the overhead of source routing in large networks and the absence of local repair and query localization in contrast to AODV severely hamper its scalability. GHLS is the only location-based protocol that has comparable overhead to AODV and DSR for small networks.

VIII. Related Work

Several different classifications of location services have been proposed. In [24], location services are classified according to the number of servers and the extent of each server's usage. In [25], location services are classified as being reactive or proactive. A survey of location services appears in [26].

While our work is the first (to the best of our knowledge) to evaluate the performance tradeoffs between scalable rendezvous-based location services, several previous studies have either quantitatively compared flooding-based and rendezvous-based location services, or evaluated flooding-based services via simulations. Studies of individual protocols such as GLS have also been performed. In [24], the authors discuss the asymptotic complexities of several protocols such as greedy forwarding, DREAM [6], LAR [5], Terminodes [27], and Grid [13]. In [28], the authors compare three flooding-based location services (DREAM, SLS, and RLS). In SLS, each node periodically exchanges locations with its neighbors whereas RLS floods the network with location queries on-demand. Another work [29] proposes a proactive protocol LEAP and compared it with SLS, RLS, and GLS for small networks. In [30], the authors propose a reactive location service and compare it to DSR and GLS running over GPSR. In [7], a proactive location service with a prediction model utilizing the distance effect is proposed and compared with SLS, DREAM and GRSS [15].

The GHLS protocol proposed in this paper shares the nature of geographic hashing with GHT [31]. However, GHT is proposed to support data storage in statically deployed dense sensor networks. Additionally, design objectives in GHT are fundamentally different since it stores sensed data where reliability and storage costs are more important.

We note that ad hoc positioning systems such as [32] have been proposed which can route packets geographically without using GPS hardware by providing a node with an estimate of its virtual coordinates. However, a location service is still required to lookup another node's virtual coordinates. An evaluation of such positioning systems is outside the scope of this paper.

IX. Conclusions

In this paper, we presented a performance comparison of three scalable location services for geographic routing. The primary insight from our study is that when practical MANET sizes are considered, i.e., up to a few thousand nodes, robustness to mobility and the constant factors matter more than the asymptotic costs of location service protocols. In particular, we have shown that although asymptotically GLS is more scalable than both GHLS and XYLS in terms of protocol overhead, GHLS is far simpler and transmits fewer packets than GLS in networks of up to 25,000 nodes.

Similarly, although XYLS scales worse asymptotically than GLS, it transmits fewer control packets and delivers more data packets than GLS in large mobile networks. We also found that although XYLS has a comparable PDR to GHLS, it transmits more control packets.

Hierarchical hashing-based protocols try to reduce their overhead by taking advantage of localized mobility. However, if communication is not local, the query has to travel high up in the hierarchy, and thus such schemes have to deal with inconsistencies at faraway location servers which degrade performance in mobile networks. The primary reasons for GLS losing to the flat hashing scheme GHLS are: (1) Hashing in the node ID space (for both updates and queries) using geographic forwarding unavoidably runs into consistency handling, e.g. when nodes cross grid boundaries. In contrast, geographic hashing uses a simple handoff mechanism to deal with node mobility, avoiding any consistency handling. (2) Maintaining multiple location servers at each level of the hierarchy increases the number of updates by a factor of 3 every one interval, a factor of 6 every two intervals, a factor of 9 every four intervals, and so on. The average number of updates per interval approaches 6 quickly as the height of the hierarchy increases. (3) While queries for nearby nodes can benefit from the locality inherent in the hierarchy, they can also suffer the "gridding" effect: source and destination nodes near the boundaries of two high-order sibling squares may have to travel many steps up the common sub-hierarchy containing both nodes, which defeats the purpose of using hierarchy for query locality. We have also shown that even for traffic patterns with high locality, GHLS incurs lower total protocol overhead than GLS due to its low cost location updates as well as effective caching in query resolution.

We have shown that reactive routing protocols like AODV provide an efficient alternative to geographic routing for small to medium sized networks. In such networks, the beaconing overhead of geographic routing and the update and lookup cost of GLS and XYLS make them less attractive than a cheaper, non-location-based solution. In contrast, geographic forwarding equipped with GHLS provides a routing protocol that is efficient in networks of a wide variety of sizes including small networks.

ACKNOWLEDGMENTS

This work was supported in part by NSF grant ANI-0338856.

REFERENCES

[1] C. E. Perkins and P. Bhagwat, "Highly dynamic destination-sequenced distance-vector routing (DSDV) for mobile computers," in *Proc. of ACM SIGCOMM*, August 1994.

[2] D. B. Johnson and D. A. Maltz, "Dynamic source routing in ad hoc wireless networks," *Mobile Computing, Kluwer Academic*, 1996.

[3] C. E. Perkins and E. M. Royer, "Ad hoc on-demand distance vector routing," in *Proc. of IEEE WMCSA*, February 1999.

[4] Z. J. Haas and M. R. Pearlman, "The performance of query control schemes for the zone routing protocol," in *Proc. of ACM SIGCOMM*, August 1998.

[5] Y.-B. Ko and N. H. Vaidya, "Location-aided routing (LAR) in mobile ad hoc networks," in *Proc. of ACM MobiCom*, October 1998.

[6] S. Basagni, I. Chlamtac, V. R. Syrotiuk, and B. A. Woodward, "A distance routing effect algorithm for mobility (DREAM)," in *Proc. of ACM MobiCom*, October 1998.

[7] V. Kumar and S. R.Das, "Performance of dead reckoning-based location service for mobile ad hoc networks," *Wireless Communications and Mobile Computing Journal*, December 2003.

[8] Z. J. Haas and B. Liang, "Ad Hoc Mobility Management with Uniform Quorum Systems," *IEEE/ACM Transactions on Networking*, vol. 7, no. 2, pp. 228–240, April 1999.

[9] I. Stojmenovic, "A routing strategy and quorum based location update scheme for ad hoc wireless networks," SITE, University of Ottawa, Tech. Rep. TR-99-09, September 1999.

[10] S. Giordano and M. Hami, "Mobility management: The virtual home region," EPFL, Tech. Rep. SSC/037, October 1999.

[11] I. Stojmenovic, "Home region based location updates and destination search schemes in ad hoc wireless networks," SITE, University of Ottawa, Tech. Rep. TR-99-10, September 1999.

[12] S.-C. M. Woo and S. Singh, "Scalable routing protocol for ad hoc networks," *Wireless Networks*, vol. 7, no. 5, pp. 513–529, 2001.

[13] J. Li, J. Jannotti, D. S. J. D. Couto, D. R. Karger, and R. Morris, "A scalable location service for geographic ad hoc routing," in *Proc. of ACM MobiCom*, August 2000.

[14] Y. Xue, B. Li, and K. Nahrstedt, "A scalable location management scheme in mobile ad-hoc networks," in *Proc. of IEEE LCN*, November 2001.

[15] P. Hsiao, "Geographical region summary service for geographical routing," *ACM MC2R*, vol. 5, no. 4, pp. 25–39, January 2002.

[16] C. Cheng, H. Lemberg, S. Philip, E. V. den Berg, and T. Zhang, "SLALoM: A scalable location management scheme for large mobile ad-hoc networks," in *Proc. of IEEE WCNC*, March 2002.

[17] USCG Navigation Conter GPS page, "http://www.navcen.uscg.nil/gps/default.html", January 2000.

[18] P. Bose, P. Morin, I. Stojmenovic, and J. Urrutia, "Routing with guaranteed delivery in ad hoc wireless networks," in *Proc. of ACM DIALM Workshop*, August 1999.

[19] B. Karp and H. Kung, "GPSR: Greedy perimeter stateless routing for wireless networks," in *Proc. of ACM MobiCom*, August 2000.

[20] D. Karger, E. Lehman, T. Leighton, M. Levine, D. Levin, and R. Panigrahy, "Consistent Hashing and Random Trees: Tools for Relieving Hot Spots on the World Wide Web," in *Proc. of ACM STOC*, May 1997.

[21] X. Zeng, R. Bagrodia, and M. Gerla, "Glomosim: A library for parallel simulation of large-scale wireless networks," in *Proc. of PADS Workshop*, May 1998.

[22] J. Yoon, M. Liu, and B. Noble, "Random waypoint considered harmful," in *Proc. of IEEE INFOCOM*, April 2003.

[23] Y.-C. Hu and D. B. Johnson, "Caching strategies in on-demand routing protocols for wireless ad hoc networks," in *Proc. of ACM MobiCom*, August 2000.

[24] M. Mauve, J. Widmer, and H. Hartenstein, "A survey on position-based routing in mobile ad hoc networks," *IEEE Network*, pp. 30–39, November/December 1999.

[25] T. Camp, "Location information services in mobile ad hoc networks," The Colorado School of Mines, Tech. Rep. MCS-03-15, 2003.

[26] I. Stojmenovic, "Location updates for efficient routing in ad hoc wireless networks," *Handbook of Wireless Networks and Mobile Computing, Wiley*, 2002.

[27] J. Hubaux, T. Gross, J. L. Boudec, and M. Vetterli, "Towards self-organizing mobile ad-hoc networks: The Terminodes project," *IEEE Comm Mag*, vol. 39, no. 1, pp. 118–124, January 2001.

[28] T. Camp, J. Boleng, and L. Wilcox, "Location information services in mobile ad hoc networks," in *Proc. of IEEE ICC*, May 2002.

[29] X. Jiang and T. Camp, "An efficient location server for an ad hoc network," The Colorado School of Mines, Tech. Rep. MCS-03-06, May 2003.

[30] M. Kasemann, H. Fuler, H. Hartenstein, and M. Mauve, "A reactive location service for mobile ad-hoc networks," University of Mannheim, Tech. Rep. CS TR-14, November 2002.

[31] S. Ratnasamy, B. Karp, L. Yin, F. Yu, D. Estrin, R. Govindan, and S. Shenker, "GHT: A geographic hash table for data-centric storage in sensornets," in *Proc. of ACM WSNA*, September 2002.

[32] A. Rao, C. Papadimitriou, S. Shenker, and I. Stoica, "Geographic routing without location information," in *Proc. of ACM MobiCom*, September 2003.