Scalable Location Services for Hierarchically Organized Mobile Ad hoc Networks

Siva Sivavakeesar
Center for Communication Systems Research
University of Surrey
Guildford
Surrey GU2 7XH, United Kingdom
S.Sivavakeesar@surrey.ac.uk

George Pavlou
Center for Communication Systems Research
University of Surrey
Guildford
Surrey GU2 7XH, United Kingdom
G.Pavlou@surrey.ac.uk

ABSTRACT
This paper proposes a location service to assist location-based routing protocols, realized through a novel Associativity-Based clustering protocol. The main goal of our scheme, which employs hierarchical principles, is to minimize the control traffic associated with location-management. In location-based routing protocols, the control traffic is mainly due to location-updates, queries and responses. Our scheme employs a novel geographically-oriented clustering scheme in order to minimize control traffic without impairing performance. In our location management scheme, nodes are assigned home-zones, and are required to send their location-updates to their respective home-zones through a dominating-set. This strategy, unlike similar location-management approaches, minimizes inevitable superfluous flooding by every node, and prevents location updates and queries from traversing the entire network unnecessarily, hence conserving bandwidth and transmission power. The proposed scheme is evaluated through mathematical analysis and simulations, and the results indicate that our protocol scales well with increasing node-count, node-density and node-speed.

Categories and Subject Descriptors
C.2.1 [**Network Architecture and Design**]: Wireless communication – *ad hoc networks;* C.2.2 [**Network Protocols**]: Routing protocols – *location-based routing*

General Terms
Algorithms, Performance, Design.

Keywords
Location-based routing, location service, location-management, hierarchical clustering.

This work was undertaken in the context of the UK Engineering and Physical Sciences Research Council (EPSRC) Programmable Ad-hoc Networks (PAN - GR/S02129/01) research project.

MobiHoc'05, May 25–27, 2005, Urbana-Champaign, Illinois, USA.

1. INTRODUCTION
In multihop mobile ad hoc networks (MANETs), the routing protocol is the key to efficient operation, and this paper considers the problem of routing in such networks of large-scale. However, the design of an effective and efficient routing protocol in MANETs is extremely challenging because of mobility, limited battery energy, unpredictable behavior of radio channels, and time-varying bandwidth [2]. The absence of fixed infrastructure means that the mobile nodes (MNs) communicate directly with one another in a peer-to-peer fashion, and requires routing over multihop wireless paths. The main difficulty arises from the fact that multihop paths consist mainly of wireless links whose endpoints are likely to be moving independently of one another. Consequently, node mobility causes the frequent failure and re-activation of links, effecting a reaction from the network's routing algorithm to the changes in topology, thus increasing network control traffic and contributing to congestion. This routing task becomes extremely challenging when the network grows in size, and when two problems such as increasing node-density and large number of nodes have to be tackled. High node-density, where a node is within a radio-range of a large number of neighbors, often leads to superfluous forwarding of routing related control traffic, and large network size necessitates the maintenance of large routing tables. These two features are inter-related, and often affect the routing protocol scalability.

A considerable body of work has addressed on routing in MANETs, including a new generation of on-demand and efficient pro-active routing approaches [7][8]. These routing algorithms, however, tend to use flooding or broadcasts for route computation. While they can operate well in small networks, they incur heavy control traffic for discovery and maintenance of end-to-end routes, which forms a major bottleneck for large networks having node membership in the order of thousands over a large geographical area. In addition, flooding in MANETs does not work well due to the presence of hidden and exposed-terminals, and does not scale [10]. In recent years, a new family of protocols has been introduced for large-scale ad hoc networks that make use of the approximate location of nodes in the network for geography-based routing [3][4][7]. The amount of state information that needs to be stored by nodes in this case is minimal, because location-based routing does not use pre-computed routes for packet forwarding. As a result, link-breakage in a route does not affect the end-to-end session. These protocols, however, often need proper location services, and hence location-management plays a vital role. Previous work in this area has

shown that the asymptotic overhead of location-management is heavily dependent on the service primitives (location updates or registration, maintenance and discovery) supported by the location-management protocol of the location service [3][4]. However, the location-registration or update cost normally dominates other costs for all practical purposes, and thus novel schemes are required to limit this control traffic. In our location-management scheme, we try to achieve this with an introduction of stable geographically-oriented clustering protocol, which we name Associativity-based Clustering protocol. This protocol does not involve any extra control traffic, and periodic HELLO messages as in AODV or other location service approaches are enough [1][8][13]. We use the concept of *virtual-clusters* that was introduced in [1], and each *virtual-cluster* functions as a home-zone for a set of nodes. Nodes that reside within a *virtual-cluster* thus maintain approximate location information of a set of nodes, which select that *virtual-cluster* as their home-zone, in a distributed fashion. Since mobility is the main cause of uncertainty in ad hoc networks, our clustering protocol and algorithm takes this as the main criterion in order to select a relatively long-lived cluster head (CH) in each *virtual-cluster* [1]. Our strategy is to address the scalability issue both in dense and in large-scale networks. Scalability in dense networks is addressed efficiently by allowing only a few dominating set of nodes to make "summarized" composite periodic location-updates on behalf of a set of dominated nodes (the dominating-set in our context doesn't strictly follow the graph theory principles, and it refers to a set of CHs that can be reached by other neighbors not necessarily by single-hop but by single or k-hops at most, and dominated nodes are simply the members of a cluster). This is to minimize superfluous flooding by every node to the entire network, as in other location services [5]. Scalability in large-scale networks is addressed by strictly using geo-forwarding-based (location-based) unicasting as opposed to flooding even for location-registration process, and prevent location-updates, queries and replies from arbitrarily traversing unnecessary parts of the ad hoc network. As a result, the performance of our geo-forwarding-based routing strategy hardly degrades due to excessive control traffic, and from poor route convergence and routing-loops resulting from mobility. This paper thus deals with the problem of designing a location-update scheme in a scalable way to provide accurate destination information in order to enable efficient routing in mobile ad hoc networks.

The rest of this paper is organized as follows. Section 2 examines related previous work after explaining the basic principles of location-based routing strategy, and presents our motivation. The novel associativity-based clustering protocol and the proposed location service technique are described in section 3. Section 4 evaluates the proposed scheme through simulations, and demonstrates that our location service results in less signaling traffic in comparison to other similar methods. Section 5 presents our conclusions and future work.

2. RELATED WORK AND OUR MOTIVATION

2.1 Overview of Location-based Routing

In location-based routing, the forwarding decision by a node is primarily based on the position of a packet's destination and the position of the node's immediate one-hop neighbors [7]. The position of the one-hop neighbors is typically learned through one-hop broadcasts realized through periodic beaconing or HELLO transmissions. In order to learn the current position of a specific node, the help of a location service is needed. MNs first identify their location-servers, and register their current position with these servers through location-update packets. When a node needs to know the location of its desired destination partner, it contacts the appropriate location-server and obtains this information. A location-query packet is used by the querying node, and results in a location-response packet. There are three main packet forwarding strategies for location-based routing: greedy forwarding, restricted-directional flooding, and hierarchical approaches. In the first two strategies, a node forwards a given packet to only one (greedy-forwarding) or to more (restricted directional flooding) one-hop neighbors that are located closer to the destination than the forwarding node itself. Recovery strategies are needed in these two approaches, when there is no one-hop neighbor that is closer to the destination than the forwarding node itself. The third approach uses hierarchical principles for scalability reasons. There are four key location service strategies found in the literature: Distance routing effect algorithm for mobility (DREAM) approach, quorum-based location service, grid location service (GLS), and home-zone based location service. The details of these location service approaches are briefly described next.

2.2 Related Work on Location Service

According to the DREAM approach, each node tries to maintain location information about each other node in the network. This approach can be regarded as an *all-for-all* approach, whereby all nodes are involved, and every node maintains the location of all nodes. Each MN periodically floods location-updates, and uses two mechanisms to control the accuracy of its location information available to other nodes: 1) the frequency at which it sends location-updates and, 2) by specifying the "scope" of a location-update [7][8]. The frequency of location-updates is also coupled with the mobility rate of a node. Due to the communication complexity of location-updates, DREAM is considered the least scalable location service technique, and thus inappropriate for large-scale MANETs [7].

In the quorum-based location system, a subset of all mobile nodes is chosen to host location databases. A virtual backbone is then constructed between the nodes of the subset, using a non-location-based ad hoc routing mechanism. A MN sends its location-updates to the nearest backbone node, which then chooses a quorum of backbone nodes to host the location information. Whenever a node wants to find the location information of another node, the former sends a query to the nearest backbone node, which in turn contacts the nodes of a (usually different) quorum. Since by definition the intersection of two quorums is non-empty, the querying node is guaranteed to obtain at least one response with the desired location information. If several responses are received, the one representing the most recent location-update is chosen. The quorum-based location approach typically works as a *some-for-some* approach, with the backbone being a small subset of all available nodes and quorum being a small subset of the backbone nodes. This scheme, however, does not specify as to how the virtual backbone nodes are selected and managed. Further, the quorum system depends on a non-location-based ad hoc routing protocol for the virtual backbone, which

tremendously increases the implementation complexity. In the case of grid location service (GLS), the area that contains the ad hoc network is divided into a hierarchy of squares [5]. In this hierarchy, n-order squares contain exactly four (n-1)-order squares, forming quadtrees. To understand GLS, an arbitrary node v is considered. The set of nodes functioning as location-servers for v are based on the relation of their node ID to v and their location in the grid hierarchy. The density of location-servers for v in regions near v is high and low in the regions far from v. The frequency at which v updates location to nearby location-servers is high, while servers situated far from v receive updates at a low frequency. GLS ensures that for each grid-zone, a node can be selected unambiguously to function as the location-server for v. Since GLS requires all nodes to store information about some other nodes, it can be classified as an *all-for-some* approach. In GLS, the location updates of a particular node have to traverse the entire network, as the location-servers of a given node are spread throughout the network. In addition, whenever a querying node contacts the nearest location-server of another far-away node, whose location information is requested for, that query needs to be forwarded to the node being queried. This forwarding is based on the location information maintained by a far-away server, which is nearest to the querying node. In this case, the freshness of the information obtained is questionable, as nodes are required to update far-away location servers less frequently. This greatly depends on how quickly a particular entry in the location-server times-out or becomes stale. If, on the other hand, the frequency of updates increases, this may have a serious impact on scalability. Further, due to mobility, the role of location-servers for a particular node will keep on changing, and as a result the location-update and query packets may find it difficult to detect the appropriate location-servers. In the case of home-zone-based location service, the position C of the home-zone for a node can be derived by applying a well-known hash-function to the node identifier. All MNs within a circle with radius R centered at C have to maintain location information for the nodes. The home-zone approach is also an *all-for-some* approach. If the home-zone is sparsely populated, R may have to be increased, resulting in several tries with increasing R for updates as well as queries. In this way, increasing or decreasing the R depending on node-density is very complex when it comes to practical implementation. Although our scheme makes use of similar home-zone strategy, the way in which the location service is realized is both simple and scalable, as it will be explained in section 3.

2.3 Related Work on Clustering

The purpose of clustering is two-fold: the first is to create a network of hierarchy, and the second is to select a dominating-set of nodes i.e. the cluster heads. This results in scalable network, where the location-server functionality is equitably distributed among network nodes. Choosing CHs optimally is an NP-hard problem [1]. Thus existing solutions to this problem are based on heuristic (mostly greedy) approaches and none of them attempts to retain the topology of the network [1]. Also, almost none of them consider node mobility as the main criterion in the clustering process. As a result, they fail to guarantee a stable cluster formation. In a MANET that uses cluster-based services, network performance metrics such as throughput and delay are tightly coupled with the frequency of cluster reorganization. Therefore, stable cluster formation is essential for better Quality of Service (QoS). The most popular clustering approaches in the literature are the lowest identifier (Lowest-ID) and maximum-connectivity. But these two, along with others, do not provide a quantitative measure of cluster stability [1].

2.4 Our Motivation

As mentioned before, the location-based routing strategy is chosen in order to improve scalability. On the other hand, the location management cost should not be high. The adoption of a hierarchical strategy together with the use of a dominating-set demonstrates as to how the control traffic is minimized without compromising route computation accuracy. For analysis purposes, the ad hoc network is represented as an undirected graph G = (V, E), where V is the set of nodes in the graph, and E is the set of edges in the graph.

3. HOME-ZONE BASED HIERARCHICAL LOCATION MANAGEMENT

3.1 Associativity-based Stable Clustering

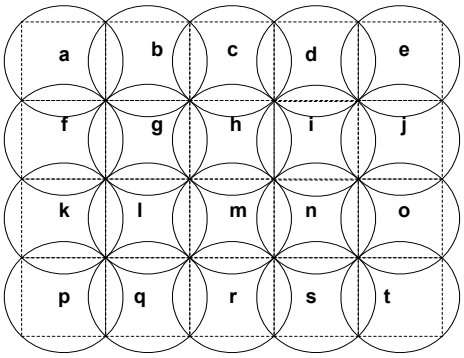

Figure 1. Concept of *Virtual-Clusters*.

In order to make our clustering mechanism scalable, we make use of the notion of *virtual-clusters* that was introduced in [1]. The idea is that a geographical area is divided into equal regions of circular shape, as depicted in Fig.1, in a systematic way so that each MN can determine the circle it resides in if location information is available. In our scheme, each *virtual-cluster* has a unique identifier based on the geographic location, which can be calculated using a publicly known function. Each MN should have a complete picture of the locations of these *virtual-clusters* and their centers [1].

Our clustering protocol does not involve any extra control traffic; instead, periodic HELLO messages – as in other similar location-management approaches – are enough. This clustering facilitates electing a dominating set i.e. the cluster heads (CHs). In order to maintain stable clusters, a new associativity-based criterion is used to elect CHs [9]. A node is elected as a cluster head, if it has the highest associativity-state with respect to its present *virtual-cluster*, and stays nearer to its *virtual-cluster*-centre (VCC), in comparison to other nodes in the same cluster. This implies spatial, temporal, and connection stability. Each MN periodically monitors its current speed, and whenever its speed is zero, it starts measuring its "associativity". This is because typically after an unstable migration period, there exists a period of stability, where

a mobile node spends some "dormant" or "residence time" within the *virtual-cluster* before it starts moving again [9].

When a MN is stationary, it measures its associativity with respect to a particular *virtual-cluster* by periodic "ticks" that takes place every ASSOCIATIVITY_TICK_PERIOD. This process however does not involve any transmissions. In this way, any node X within the k^{th} *virtual-cluster* that has its total number of ticks (n_{xk}) greater than $A_{threshold}$, will exhibit higher degree of "associativeness", and hence have greater "dormant time". If, however, the speed of the MN is monitored to be greater than μ_{TH} (a system parameter), its number of ticks immediately becomes zero. The heuristic used by our clustering scheme is given by equation (4). Any node X determines the criterion value (Ω_{xk}) in k^{th} cluster by calculating the following:

- Its distance from the centre (VCC) of a particular *virtual-cluster*. Assuming node X, whose location co-ordinates at time 't' are ($x_{xk}(t)$, $y_{xk}(t)$), in the k^{th} *virtual-cluster*, whose center's Cartesian co-ordinates are (x_{ck}, y_{ck}), its distance at time 't' can be calculated by :

$$d_{xk}(t) = \sqrt{\left(x_{Xk}(t) - x_{ck}(t)\right)^2 + \left(y_{Xk}(t) - y_{ck}(t)\right)^2} \quad (1)$$

- Each node stores the "residence" or "dormant time" in the last m number of clusters it has visited. This is basically the time period from the instance at which the MN's velocity is zero within a particular *virtual-cluster* and the instance at which it is more than μ_{TH}. Then the mean "dormant-time" in terms of number of "ticks" (N_{mean_X}) is calculated as follows. Assuming that the "dormant-time" of node X in the j^{th} *virtual-cluster* is R_{X_j}. Then node X's mean "dormant-time" (R_{mean_x}) is determined by considering its "dormant-times" in the last *m* number of *virtual-clusters* as given by equation (2). The mean "dormant-time" in terms of number of "ticks" is derived from equation (3). With this, the clustering criterion value (Ω_{xj}) for node X in *virtual-cluster* j is determined from equation (4).

$$R_{mean_x} = \frac{\sum_{j=1}^{j=m} R_{X_j}}{m} \quad (2)$$

$$N_{mean_X} = \frac{R_{mean_X}}{ASSOCIATIVITY\_TICK\_PERIOD} \quad (3)$$

$$\Omega_{xj} = \begin{cases} \dfrac{N_{mean_X} - n_{xj}}{d_{xj}(t)} & \forall d_{xj}(t) \neq 0, n_{xj} \neq 0 \\ \dfrac{N_{mean_X} - n_{xj}}{d_{\min}} & \forall d_{xj}(t) = 0, n_{xj} \neq 0 \\ \dfrac{1}{d_{xj}(t)} & \forall d_{xj}(t) \neq 0, n_{xj} = 0 \\ \dfrac{1}{d_{\min}} & \forall d_{xj}(t) = 0, n_{xj} = 0 \end{cases} \quad (4)$$

Accordingly, any node X that has the highest value for the clustering criterion Ω_x is elected in either a centralized way or in a distributed way, depending on whether the present CH is available or not [1]. Equation (4) tries to ensure that the resulting clusters are more stable, and have uniform coverage by the respective CHs. If the CH lies very near to a VCC, it can have a uniform coverage, and hence ensures that all member nodes of a *virtual-cluster* are connected to this CH directly or via k-hops, where k is bound by $D/(2R_{TX})$, and D is the diameter of the *virtual-cluster*, with R_{TX} being the transmission radius of a node. Ω_x is proportional to the expected "residence time", and inversely proportional to the distance from the respective VCC. The system parameter d_{min} ($\neq 0$) is the minimum value that $d_{xk}(t)$ can take. The structure of HELLO packet has been modified to include this criterion value, so that any node can know the neighbors' Ω-values. Depending on the current state and circumstances, any node can disseminate one of the following four different packet types: *JOIN*, *HELLO_CH*, *HELLO_NCH*, and *SUCCESSOR* [1]. These control packets, except periodic *HELLO* packets by CHs, are relayed by intermediate MNs only within the *virtual-cluster*, where they have originated from. The distributed operation and the bootstrapping process of this clustering protocol are similar to those we proposed in [1]. Whenever a present CH knows that it is going to leave the *virtual-cluster* it is currently serving, it will select a member node that has the highest value for the criterion value as its successor, and inform about it the cluster members through the SUCCESSOR packet. In this case, the CH is elected in a centralized manner. The present CH can decide it is going to leave the cluster, when its monitored speed at a moment exceeds μ_{TH}. Unlike in any other clustering algorithm, our algorithm has another unique feature in that whenever a CH leaves the *virtual-cluster*, it will loose its CH status. In this way this algorithm ensures that no other visiting MN can challenge an existing CH within a particular *virtual-cluster*, and thus causing transient instability. All aspects of our strategy ensure that stable CHs are elected, and thus results in stable clustering. This clustering scheme is thus fully distributed, where all the nodes share the same responsibility and act as CHs depending on the circumstances.

In our location management scheme, we maintain a two-level hierarchical topology, where elected cluster heads at the lowest level (level-0) become members of the next higher level (level-1) [8][11]. In our scheme, the CHs are needed, and thus are elected using the virtual clustering concept only at level-0. CH election is not triggered in level-1, where only the cluster-membership detail is maintained. Level-0 hierarchy is used for efficient location-updating, while level-1 hierarchy is used for resilience as explained in section 3.2.

3.2 Home-Zone Based Location Service

We make the following assumptions in our model: 1) the area to be covered is heavily populated with mobile nodes, 2) Heavy-traffic is expected within the network (i.e., multiple simultaneous communications among nodes are possible), 3) every node is equipped with GPS (Global Positioning System) capability that provides it with its current location, and 4) there exists a universal hash-function that maps every node to a specific home-zone based on the node's identifier [3][4].

A home-zone is basically a *virtual-cluster* that has a unique identifier. Any node that is present in that *virtual-cluster* is responsible for storing the current locations of all the nodes that select this cluster area as their home-zone, and hence functions as a location-server. Since our location-management scheme requires that all nodes store the location information on some other nodes, it can be classified as an *all-for-some* approach, which can scale well [7]. The static mapping of our hash-function, as given by equation (5), is to facilitate simplicity and distributed operation. This hash-function has to be selected such that, i) all MNs should be able to use the same function to determine the home-zone of a specific node, ii) every *virtual-cluster* has the same number of nodes for which the MNs residing within that cluster should maintain location information, iii) and the mapping functionality has to be time-invariant.

$$hf(NodeIdentifier) \rightarrow Home-Zone \quad (5)$$

In any location service, nodes are required to update their location information depending on their mobility. In our scheme, the update generation is mobility-driven as well as time-driven. The time-driven approach is to make sure that even if a node is stationary, periodical update is made to its home-zone. The unique aspect of our location-management scheme is that it tries to minimize the location-update cost with the help of dominant-set elected at level-0 using our clustering protocol. Dominant-set is basically a set of CHs elected at level-0 using our virtual-clustering principles. Accordingly, each node maintains four different table types: neighbor-table, location-cache, location-register, and forwarding-pointer-table. Each node has two different neighbor-tables maintained separately at each hierarchy level, and one of each of the other three types of tables maintained at level-0 only. Only the CH at level-0 has entries in its neighbor-table maintained at level-1. Neighbor-table at level-0 is used by every node to maintain the members of a particular *virtual-cluster* together with its one-hop neighbors, irrespective of their cluster identity. The periodic *HELLO* messages within a specific *virtual-cluster* are used to maintain this neighbor-table at level-0. As mentioned before, nodes of a specific cluster can relay *HELLO* packets of another node only when the latter resides within the same *virtual-cluster* [1]. However, when a bordering node receives a *HELLO* from a node of a different *virtual-cluster*, the former maintains the details of the latter in the neighbor-table for geo-forwarding purposes. Periodic *HELLO* messages by CHs have to be unicast by gateways between CHs of adjacent *virtual-clusters* to an extent that can be limited for scalability. This *HELLO* dissemination among neighboring CHs at level-1 facilitates maintenance of neighbor-table by CHs. Location-register of a node within a specific *virtual-cluster* has the location information of MNs whose home-zone is identical to the *virtual-cluster* of the former. The location-cache of a node is updated, whenever that node happens to know the location information of another non-member and non-home-zone node, for example, during location-discovery or data handling.

From the neighbor-table at level-0, any CH knows about its members, which may have different home-zones. In our scheme, the cluster head gathers the location information of its member nodes that have a common home-zone. Unless these nodes are highly mobile, the CH generates a "summarized" single location-update towards that common home-zone, on behalf of its member nodes. As a result, the need for every node, especially in a high-density network, to generate individual location-update packet is minimized. On the other hand, any node with high-mobility has to take care of its own location-updates whenever its speed increases beyond μ_{TH}. As specified in [1], in addition to its mere presence in a *virtual-cluster*, any node has to be included within the neighbor-table of the respective CH. In location-updates and data handling, the absolute locations of nodes are not needed; instead, the *virtual-cluster* id (VID) is enough. This is possible because of two reasons: 1) since these IDs are unique, from the VID any forwarding node we can obtain the co-ordinates of the corresponding VCC, and use it for geo-forwarding, 2) only for inter-cluster packet forwarding location-based routing is used, while at the local cluster-level proactive distance vector routing is used. Whenever a node in a home-zone receives the location-update which is meant for that home-zone, it can stop geo-forwarding that packet any more. Instead, it updates its location-register and informs other nodes within the same home-zone (i.e. *virtual-cluster*) through a periodic *HELLO* packet, which includes the location-register maintained by that node, so that other member nodes can update their location-registers. Within a *virtual-cluster,* efficient broadcast is utilized as opposed to flooding. This broadcast is based on reliable unicast realized through constant interaction between MAC-level and routing-level as used in "core-broadcast" of [10]. A node whose speed exceeds μ_{TH} makes its own updates. As long as such a node's total number of *virtual-cluster* boundary-crossing so far is less than a threshold, it makes use of "forwarding-pointer" concept for correct data forwarding [4][5]. Accordingly, whenever such node moves out of its present *virtual-cluster*, it leaves a "forwarding-pointer" in the previous cluster without initiating a location-update up to its home-zone. Hence, any packet that has been geo-forwarded based on the old cluster ID can still traverse the chain of forwarding pointers to locate the user at its present location in a different cluster.

As in any location management approach, whenever a node needs location information of another node, the former has to first find the location-server (home-zone) of the latter and initiate the location-discovery process. In our strategy, the querying node uses the same mapping hash-function to determine the home-zone of its desired communicating partner. It then geo-forwards the location-query packet to that home-zone. Location-response can be initiated by the node being queried or any intermediate node as long as it contains the "fresh" information about the node being queried, or any node in the home-zone of the node being queried. In order to enable "freshness" of location information, each entry in any of the four tables maintained by each node is subject to a time-out mechanism. The use of sequence numbers achieves the same effect, in addition to avoiding routing-loops that may be introduced by mobility [8]. In addition, due to the way location-servers (home-zones) are maintained in our location strategy, the location-update or location-query packets can be unicast using geo-forwarding principles as opposed to flooding as used in other similar approaches (for e.g. in GLS [5]). This, in turn, prevents location-update and query packets from traversing unnecessary parts of the MANET. This minimizes the control traffic, and conserves scarce bandwidth and transmission energy. In the worst case, when a querying MN has not received any location-response within the LOCATION_RESPONSE_TIME_OUT period, after having tried for MAX_LOC_QUERY_RETRIES, it will start gradually flooding its location-query in the network.

The maintenance of level-1 hierarchy is for resilience purposes. Accordingly, the level-1 cluster hierarchy members periodically update each other with the fresh location information of the nodes maintained in the four different tables. This update is subject to different scopes for scalability, as in the case of fisheye state routing (FSR) [8]. This is beneficial, in case there are no nodes in a specific *virtual-cluster*. The nodes that select such a *virtual-cluster* as their home-zone do not necessarily know about its emptiness, and they may continue to send location-updates either through their level-0 CHs or by themselves. By exchanging such information by CHs, adjacent clusters may maintain location-information (in location-cache) for nodes that select the empty *virtual-cluster* as their home-zone. With this approach, any location-query that is directed to the empty *virtual-cluster* for location information about nodes whose home-zone happens to be the empty *virtual-cluster*, can still receive location-response from adjacent clusters.

3.3 Mathematical Analysis

In the case of location-based routing schemes, the total cost (Φ_T) associated with the location management is due to three parts: i) location-update cost (Φ_{LU}), ii) location-maintenance cost (Φ_{LM}), and iii) location-discovery cost (Φ_{LQ}). The location-update cost covers signaling traffic involved when nodes send updates to their home-zones periodically or depending on their mobility. In our strategy, "summarized" location-updates are sent to the home-zones of member nodes by the respective CHs, as long as those nodes belong to a particular *virtual-cluster*. The location-maintenance cost of a node generally involves the control traffic associated with the following: i) sending forward-pointer packets to the previous cluster whenever MNs depart from it, ii) informing the new cluster about its arrival, and iii) collect location-information of the nodes that have selected the *virtual-cluster* that a node has just entered, as their home-zone. Ignoring the generation of "forwarding-pointers", the location-maintenance process is carried-out through our associativity-based clustering protocol, and the cost involved as part of location-maintenance is thus the cost of maintaining the level-0 cluster. In this analysis it is assumed that nodes at a moment are situated randomly throughout a fixed size area (A) in accordance with a two-dimensional uniform random variable distribution. Also, for the purpose of analyzing the location-management cost as part of location-update events, the random waypoint model for node mobility with zero pause-time is assumed.

The scalability of a routing protocol can be assessed in terms of i) increasing node count (|V|), ii) increasing average node-density (γ), iii) increasing average node speed (μ). In our scheme, the dominating-set (CHs) makes periodic location-updates on behalf of its member nodes, with only nodes of high-mobility making their own location-updates. The location-update packet is unicast up to the respective home-zone. Once the location-update has reached the respective home-zone, our clustering protocol ensures that all nodes in that *virtual-cluster* obtain the latest information about the node, and each member of the cluster updates its location-register. Hence, the location-update cost (Φ_{LU}) per node per second in our scheme is given by equation (6) [3][4][6].

$$\Phi_{LU} = \frac{H}{|V|}\left[\frac{u}{T_{LU}}\right]\Pr(\mu < \mu_{TH}) + f.u.\Pr(\mu \geq \mu_{TH}) \quad (6)$$

In equation (6) H is the total number of home-zones or *virtual-clusters* within the area A, u is the cost of sending a location-update packet to the home-zone (the unicast cost), η is the average number of nodes within any *virtual-cluster* ($H\eta=|V|$), T_{LU} is the location-update period, $\Pr(\mu<\mu_{TH})$ is the probability that the average speed of a node is so moderate that it will soon become member of any (level-0) cluster it visits within $D/(2R_{TX})$ rounds of communication and f is the average frequency at which a non-cluster-member node with high-mobility (i.e. $\mu\geq\mu_{TH}$) moves into a new *virtual-cluster* (i.e. the boundary-crossing rate). It has been stated elsewhere that for random graphs, average hop-count (h_{avg}) between an arbitrary pair of nodes is actually $\Theta\left(\sqrt{\frac{|V|}{\log|V|}}\right)$ [11]. We assume that each node selects its speed, chosen uniformly between [$\mu-\beta$, $\mu+\beta$] for some time t, where t is distributed exponentially with mean τ. Now the equation (6) can be written as:

$$\Phi_{LU} = \frac{H}{|V|}\left[\frac{u}{T_{LU}}\right]\left[\frac{\mu_{TH}^2}{4\beta}\right] + f.u\left[1-\frac{\mu_{TH}^2}{4\beta}\right] \quad (7)$$

We assume that a node moves around with a moderate speed, so that it is always a member of any level-0 cluster. In this case, the respective cluster head makes a "summarized" update periodically. Hence, equation of (6) transforms into:

$$\Phi_{LU} = \frac{H}{|V|}\left[\frac{u}{T_{LU}}\right] = \Theta\left(\sqrt{\frac{|V|}{\eta^2 T_{LU}\log|V|}}\right) \quad (8)$$

Equation (8) shows that our location-update cost, which is the dominant of the location-management cost, scales well with increasing number of nodes and increasing node-density (as $\gamma \propto \eta$ for a given geographical area), whereas in the case of non-hierarchical minimum cost routing-protocol, the total routing cost is shown to be $\Theta\left(\mu|V|^3\log|V|\right)$ [3]. In case the mobile speed is so high that any node hardly becomes a member of any *virtual-cluster* it visits, the equation (6) transforms into:

$$\Phi_{LU} = f.u = \Theta\left(\frac{\mu}{\sqrt{a}}\sqrt{\frac{|V|}{\log|V|}}\right) \quad (9)$$

In equation (9), in order to arrive at a value for f, the shape of a *virtual-cluster* is assumed to be square and its geographical area is taken as 'a' m^2. As mentioned before, the location-maintenance cost is actually the cost involved in maintaining hierarchical clusters (especially the level-0 cluster). Hence,

$$\Phi_{LM} = \Phi_{HELLO} + \Phi_{CL-F} + \Phi_{CL-M} \quad (10)$$

In equation (10), Φ_{HELLO} is the cost involved in *HELLO* packet transmissions in the clusters, $\Phi_{CL\text{-}F}$ is the cost involved in forming the hierarchical clusters, and $\Phi_{CL\text{-}M}$ is the cost involved in the maintenance of the cluster-hierarchy. The Φ_{HELLO}, $\Phi_{CL\text{-}F}$, and $\Phi_{CL\text{-}M}$ are proved to be $\Theta(1)$, $\Theta(\log|V|/T_H)$ and $\Theta(\log|V|/T_H)$ per

node per second respectively in [11], where T_H is the period of *HELLO* transmission. Hence, the location-maintenance cost per node per second is:

$$\Phi_{LM} = \Theta\left(\frac{\log|V|}{T_H}\right) \quad (11)$$

The location-query cost is incurred whenever a node receives a data packet for transmission, and it does not have the location information of its desired destination in its location-database. In our strategy, a location-query packet is initiated towards the home-zone of the node, and hence the cost per node is given by equation (12). It is assumed that once the querying node receives the location-response, it will cache the location-information in its location-cache, so that it does not need to trigger the location-discovery process for the same destination. However, the destination should periodically update the source node about its current-location through a location-notification packet [4].

$$\Phi_{LQ} = \Theta\left(\sqrt{\frac{|V|}{\log|V|}}\right) \quad (12)$$

With the estimation of individual cost components, the average total cost of our location-management strategy can be determined using equation (13). It is assumed that any node initiates a new session to a new destination at a rate of λ sessions per second according to a Poisson process.

$$\Phi_T = \Phi_{LU} + \Phi_{LM} + \lambda\Phi_{LQ} \quad (13)$$

Assuming a constant average speed (moderate) and constant *HELLO* transmission interval (T_H), equation (13) is bounded by equation (12). However, considering the fact that the location discovery process is not triggered every time the network layer accepts data from the application-layer (i.e., λ takes a very small value) the location-discovery cost (Φ_{LQ}) is minimal when compared to that of the location-update process. As a result, location-update traffic – both time and mobility driven – can be considered dominant. Hence, equation (13) is bounded by equation (8), and this result demonstrates that our location-management strategy based on our novel associativity-based clustering is scalable in both dense and in large-scale networks.

4. EVALUATION THROUGH SIMULATIONS

4.1 Scalability Improvement of Our Associativity-based Clustering Strategy

The scalability of our clustering protocol is assessed in terms of i) increasing node-count, ii) increasing average node-density, and iii) increasing average node speed. We chose Lowest-ID, maximum-connectivity (Max-Connect), LDV algorithms – the most popular clustering protocols found in the literature – in our attempt to compare the performance of our strategy. For this purpose, we implemented our algorithm along with the other three in GloMoSim [1][12]. The distance between any two VCCs in our scheme is 200m, and the diameter (D) of a *virtual-cluster* is 284m. Each node moves using a random waypoint model, with a constant speed chosen uniformly between zero and maximum speed, which is here taken as 10 ms^{-1}. The pause time takes a value that is exponentially distributed with mean 30 seconds. Each scenario was run for a 300 simulated seconds. Lowest-ID, LDV, and maximum-connectivity clustering algorithms form 2-hop clusters. Since it was necessary to ensure that clusters formed by all the schemes cover approximately equal area, the transmission range of each MN is set to 71m. The link capacity takes a value of 2 Mbps. The important simulation parameters for this case are listed in Table I. The simulation work attempts to compare the performance of our clustering algorithm with the Lowest-ID, maximum-connectivity, LDV clustering algorithms, in terms of the stability of clusters formed and control cost incurred. The cluster instability is measured by determining the number of times each MN either attempts to become a CH or gives up its role as a CH.

Table 1. Simulation Parameters

Parameter	Value
Speed Range	0 – 10 ms^{-1}
Transmission Range	71 m
Radius of a *Virtual-Cluster*	142 m
MAC protocol	IEEE 802.11
Simulation time	300 S
ASSOCIATIVITY_TICK_PERIOD	0.1 S
HELLO_INTERVAL for a CH	3 S
HELLO_INTERVAL for a non-CH	6 S

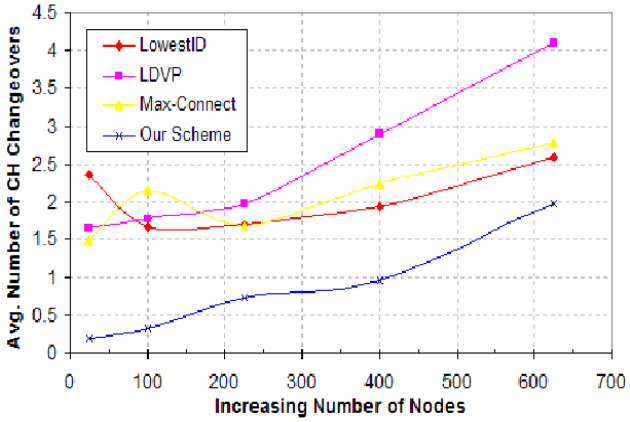

Figure 2. Cluster (In)Stability as a function of Number of increasing Node-Count.

In the first-set of simulations, the scalability of the clustering protocols is measured in terms of increasing node-count. In order to properly see the effect of increasing network nodes on the clustering algorithms, the terrain-area is also increased with an increase in the number of nodes, so that the average node-density is kept constant in this set of simulations. The number of nodes in this case is varied from 25, 100, 225, 400 and 625. The terrain-area size is varied such that the average node degree remains the same and accordingly 200X200 m^2, 400X400 m^2, 600X600 m^2, 800X800 m^2 and 1000X1000 m^2 are selected for each scenario.

Fig.2 shows the frequency of CH changes by each MN, and hence measures the (in)stability associated with each clustering algorithm as a function of increasing number of nodes. (The lower the frequency of CH changes, the more stable the cluster is). As it can be seen from Fig. 2, our clustering algorithm leads to more stable cluster formation. The average number of CH changes, which occurred per 100s, increases in the other three algorithms with the number of MNs. On the other hand, in the case of our clustering algorithm this increase is lower.

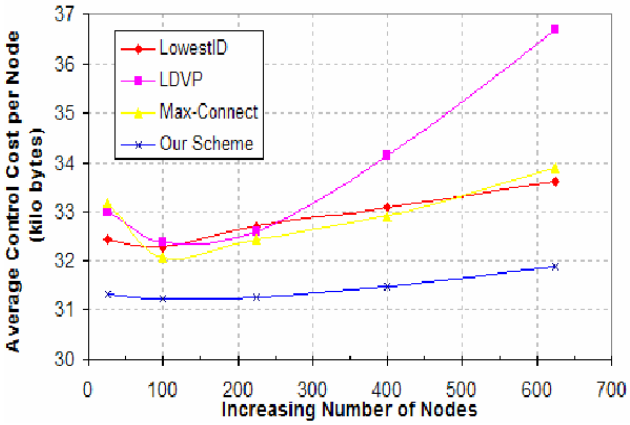

Figure 3. Average Control Cost incurred per Node as a function of increasing number of Node-Count.

Fig. 3 tries to compare the Lowest-ID, maximum-connectivity, LDV and our clustering algorithms in terms of the control cost incurred per node in kilo bytes, when the number of node increases. As it can be seen from Fig. 3, the control cost incurred per node in all the schemes tends to increase with increasing number of nodes. But in our scheme this increase is very small, and lower than those of the other three schemes. In the second-set of simulations, the scalability is measured in terms of increasing average node-density. In this case, the terrain-area is kept constant at 1000X1000 m^2, while the number of nodes in the given area is increased.

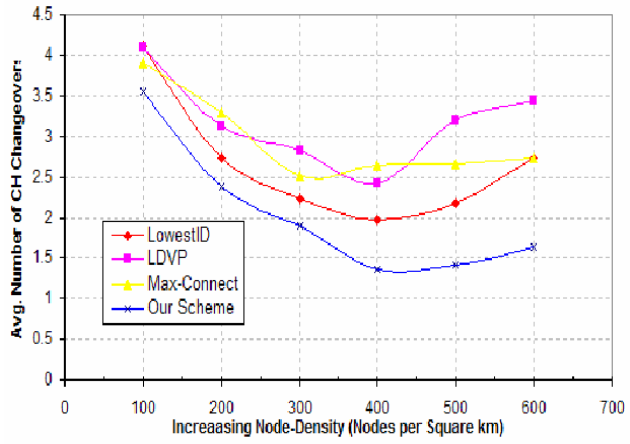

Figure 4. Cluster (In)Stability as a function of increasing Node-Density.

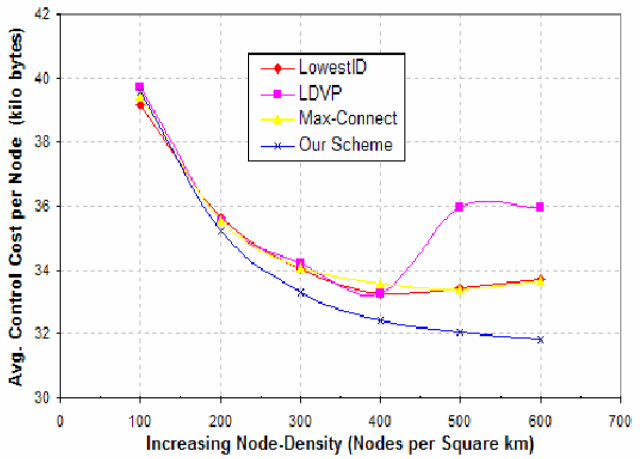

Figure 5. Average Control Cost incurred per node as a function of increasing Node-Density.

The measure of clustering stability as a function of increasing node-density is shown in Fig. 4. In all the four schemes, the cluster stability is very low when there is lower node-density. This can be due to the reason that when the node-degree is low, improper nodes are elected as cluster heads and as a result moving nodes may create transient instability. However, when the node-degree is moderate, all the clustering protocols try to converge quickly with the selection of proper heads. On the other hand, when the network becomes denser, the clustering algorithms take longer time to converge due to increased control traffic, and hence this affects the clustering stability. However, in our strategy, since the CH election heuristic takes associativity, and thus mobility, into consideration, only stable nodes are elected as CHs, and hence results in improved cluster stability. Further, in our strategy, there is a limit for the number of messages sent between nodes, and it is bounded by $\Theta(D/2R_{TX})$. This is due to the reason that our CH election heuristic ensures a node that lies very nearer to a *virtual-cluster*-center (VCC) to be elected as a CH with high probability. This condition ensures that any node is away from its respective CH by only a maximum of $D/2R_{TX}$ hops, and hence the number of messages sent from each node is limited to a multiple of $D/2R_{TX}$ in most cases. Messages are relayed by intermediate nodes only if they originate from the same *virtual-cluster* (i.e., relaying is spatially limited). Furthermore, unlike in other schemes, there is a control over the number of cluster heads elected in our scheme, and it is proportional to the number of *virtual-clusters* that we have in a given area. All these desirable features prevent arbitrary improper nodes to become CHs in our scheme, and thus help to improve cluster stability. Fig. 5 shows the clustering cost incurred by a node when the node-density increases. As can be seen, when the network is sparsely-connected, the average control cost incurred by a node tends to be high in all the four schemes. However it tends to decrease as the network becomes denser. While this control cost continues to decrease in our scheme, the same is not expected in the other three schemes. Instead, the control cost starts increasing after the node-density has reached a specific value (425 nodes per square km) in the network considered. The reason for this behavior is again same as the one given for Fig. 4.

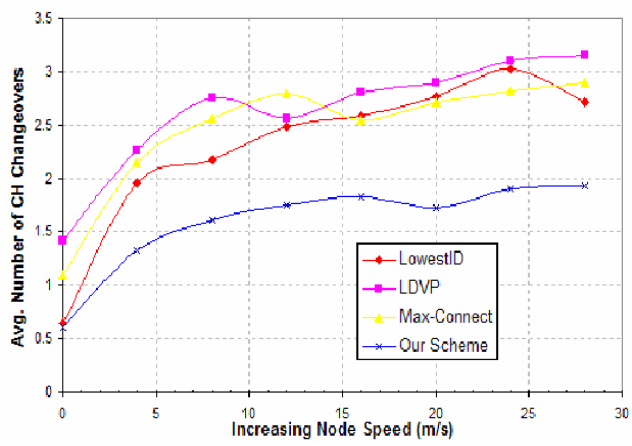

Figure 6. Cluster (In)Stability as a function of increasing Node-Speed.

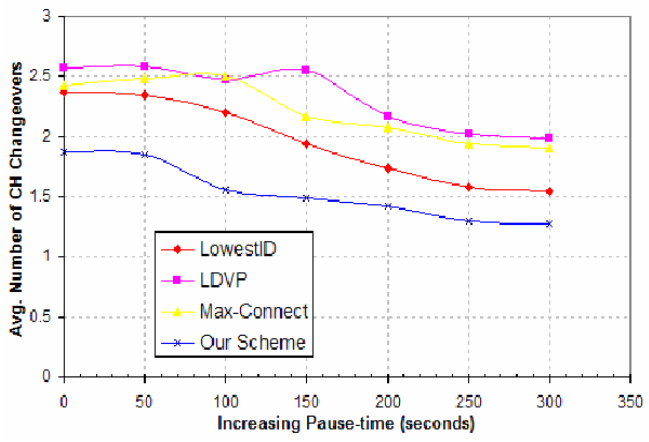

Figure 7. Cluster (In)Stability as a function of increasing Pause-time.

Figures 6 and 7 show the stability and scalability improvement of our strategy in terms of node-mobility. In the case of Fig. 6, the pause-time is exponentially distributed with mean value of 30 seconds, while the maximum speed of a mobile node is increased from 0 to 28 ms^{-1}. As can be seen from Fig. 6, although the stability is impaired by increasing node speed, the extent to which it is affected is very low in our scheme. Fig. 7 again tries to measure the stability of all four schemes in terms of node-mobility. However, in this case, the maximum speed of a node is kept constant at 10 ms^{-1}, while the pause-time is increased from 0 to 300 seconds (exponentially distributed). As pause-time increases, the stability of a cluster being formed in each scheme tends to increase in all four approaches; however the stability improvement is much high as far as our scheme is concerned.

4.2 Scalability Improvement of Our Location Service

In this section, we compare the performance of our location service only, and the scalability of our location service is assessed in terms of i) increasing node-count, ii) increasing average node-density, and iii) increasing average node-speed. We chose another similar location service approach known as GLS together with two on-demand routing protocols such as AODV and DSR [8][13][14]. We used simple greedy forwarding mechanism in both location service schemes. For this purpose, we implemented our location-management strategy based on associativity-based clustering protocol and GLS in GloMoSim [1][12]. The distance between any two VCCs in our scheme is 200m. Each node in this simulation has 100m radio range, and this smaller value is chosen to maximize the number of hops. With a larger number of hops, the effect of hops on the average response times is larger and helps us to investigate our algorithm in a realistic environment. Each node moves using a random waypoint model, with a constant speed chosen uniformly between zero and maximum speed, which varies from 0 to 20 ms^{-1}. Each scenario was run for a 300 simulated seconds. The important simulation parameters for this case are listed in Table 2. Traffic is generated using random CBR connections having a payload size of 512 bytes. These CBR connections are randomly generated such so that at any moment the total number of source-destination pairs is kept constant – and each session lasts for a time-period that is uniformly distributed between 40 and 50 seconds. We consider two performance metrics, which are normalized throughput per average control cost incurred (in per bytes) and the average end-to-end delay (in seconds). The normalized throughput is defined here as the total number of packets actually delivered to their respective destinations divided by the total number of packets generated within the whole network. The first metric is derived by dividing the normalized throughput by average routing related control cost incurred per node. The routing related control cost considers the amount of packets (non-data) generated or relayed by any node as part of an effort to route a data packet.

Table 2. Simulation Parameters

Parameter	Value
Node-Density (kept constant)	500 nodes per km^2
Speed Range	0 – 10 ms^{-1}
Transmission Range	100 m
Radius from VCC	142 m
MAC protocol	IEEE 802.11
Simulation time	300 S
ASSOCIATIVITY_TICK_PERIOD	0.1 S
HELLO_INTERVAL for a CH	3 S
HELLO_INTERVAL for a non-CH	6 S
Location Update Interval (periodic)	7 S
LOCATION_RESPONSE_TIME_OUT	4 S
MAX_LOC_QUERY_RETRIES	8

In the first-set of simulations, the scalability of our scheme is measured in terms of increasing node-count. In order to properly model increasing network sizes, the terrain-area is also increased with an increase in the number of nodes |V| so that the average node-density (γ) is kept constant. The number of nodes is varied

from 20, 80, 180, 320, 500 and 720. The terrain-area size is varied such so that the average node- degree remains the same and accordingly 200X200 m^2, 400X400 m^2, 600X600 m^2, 800X800 m^2, 1000X1000 m^2 and 1200X1200 m^2 are were selected for each run. In this run, the maximum speed and pause time of a node were kept constant and took values of 10 ms^{-1} and 30s respectively.

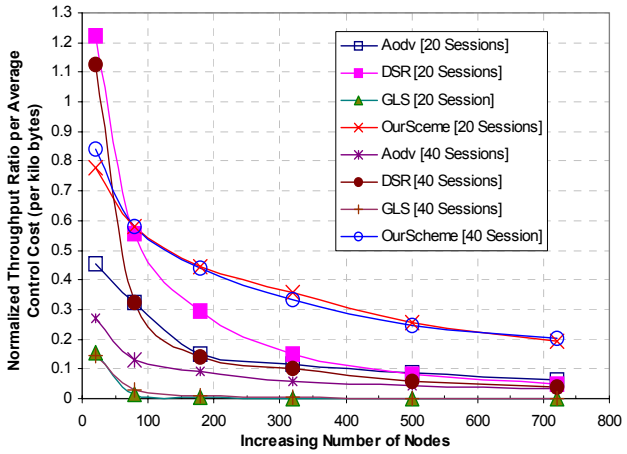

Figure 8. Normalized throughput per average Control Cost incurred as a function of increasing number of Node-Count and increasing number of Sessions.

Fig. 8 depicts our throughput metric of all the four routing approaches as a function of increasing number of nodes under different traffic scenarios. The routing strategy that leads to higher throughput delivery ratio while incurring less routing related cost would have a higher value for this metric, and hence the scheme that has higher value for this normalized throughput metric is preferred. As can be seen from Fig. 8, our location service with simple greedy forwarding works better as it minimizes the occurrence of control message flooding to a greater extent, and which in turn leads to better throughput performance. Although AODV and DSR perform well initially, however, their performances degrade as the network grows in size. This is specifically attributed to their reliance on flooding, which is used in the route (re)discovery process. Link breakage is possible in the scenario taken into consideration, as mobile nodes move at the speed of 10 ms^{-1}. GLS with simple greedy forwarding suffers due to its inefficient location-management approach as described in section 2.2. Although the throughput performance of our scheme is better for networks of larger size, the normalized throughput performances of all the four approaches tend to decrease as the network size increases. This is due to the fact that the link capacity was 2 Mbps, and it poses the main bottleneck in the scenario considered.

Fig. 9 depicts the end-to-end delay performances of AODV, DSR, GLS and our scheme (the latter two with simple greedy forwarding). Although the delay performance as expected tends to increase with network size in all four schemes, our location-management scheme with simple greedy forwarding has better delay performance. This is attributed to its efficient utilization of bandwidth. The delay of GLS is unexpectedly high, whereas DSR performs well initially in a small network; however, it degrades in larger network environment. The AODV performs moderately well. Since we used DCF of IEEE 802.11 as the underlying MAC, unnecessary flooding would result in unnecessary collisions and which in turn would lead to binary exponential backoff. The net effect of this phenomenon is the increase of end-to-end delay. Also, an important point to be noted at this juncture is that this delay is measured only for successfully delivered packets, and hence does not reflect the actual average delay, as in number of instances packets are dropped when buffer overflow occurs or after a number of unsuccessful retransmission attempts at MAC.

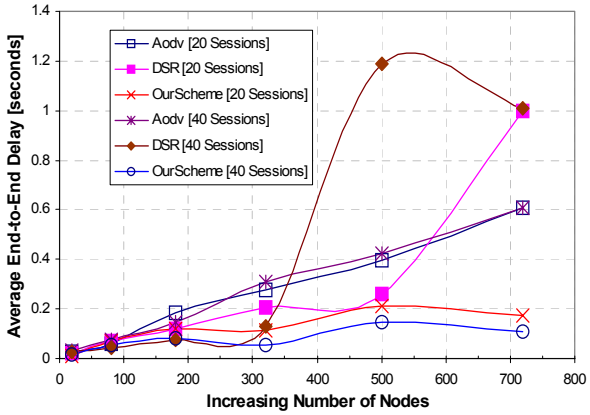

Figure 9. Average End-to-end Delay as a function of increasing number of Node-Count and increasing number of Sessions.

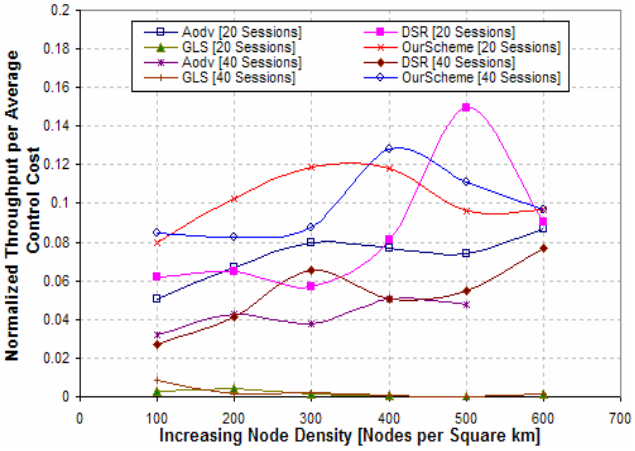

Figure 10. Normalized throughput per average Control Cost incurred as a function of increasing Node-Density and increasing number of Sessions.

In the second set of simulations, the scalability is assessed in terms of increasing node-density. In this case, the terrain-area is kept constant at 1000 X 1000 km^2, while the number of nodes in the given area is increased. In this run, the maximum speed and pause time of a node were kept constant and took values of 10 ms^{-1} and 30s respectively. Fig. 10 depicts our normalized throughput metric for all the four routing techniques, and our scheme

performs well. This is due to its conservation of bandwidth through the use of dominating-set. Since dominating-set performs "summarized" location-updates on behalf of cluster members, increase of node-density tends to improve the throughput performance. However, when the node-density reaches a certain threshold, the normalized throughput tends to drop. This can be attributed to IEEE 802.11 and is related to the fact that the wireless channel has a fixed capacity of 2 Mbps, which is saturated due to collisions when the node-density exceeds a threshold. The same explanation applies to Fig. 11, which depicts the average end-to-end delay. In this case the location-updates generated by such nodes have to traverse a long distance, and hence this slight increase is inevitable as node-count increases. However, with the selection of proper hash-function, if it can be ensured that nodes select only the nearby virtual-clusters as their home-zone, this slight increase can be very little.

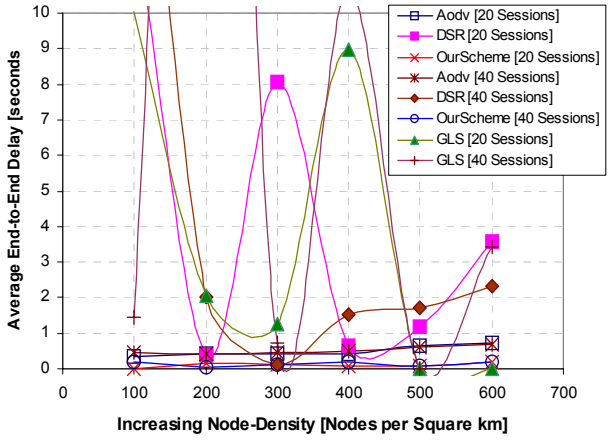

Figure 11. Average End-to-end Delay as a function of increasing Node-Density and increasing number of Sessions.

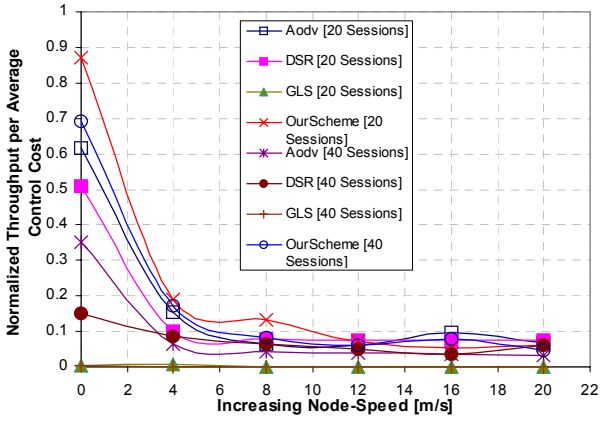

Figure 12. Normalized throughput per average Control Cost incurred as a function of increasing Node-Speed and increasing number of Sessions.

Third set of simulations assesses the scalability in terms of increasing node-mobility. In the case of Fig. 12, the pause-time is exponentially distributed with mean value of 30 seconds, while the maximum speed of a mobile node is increased from 0 to 20 ms^{-1}. As can be seen from Fig. 12, although the throughput performance is impaired by increasing node-speed, the extent to which it is affected is very low in our scheme. In addition to the use of unicasting, the mobility-driven location-update mechanism as described in section 3.2 results in comparable throughput increase in our scheme while incurring additional control cost. The same explanation applies to Fig. 13, which depicts the average end-to-end delay.

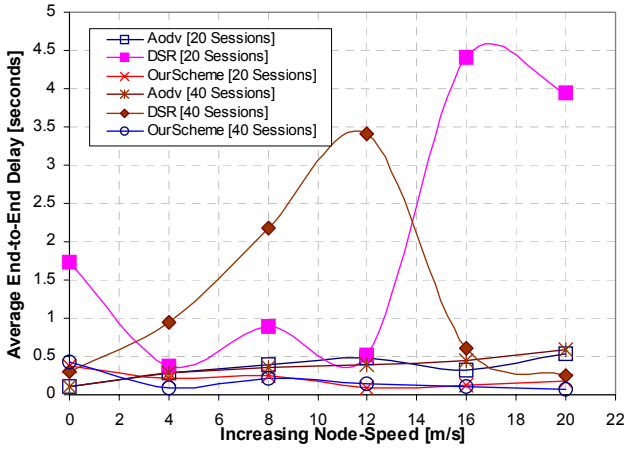

Figure 13. Average End-to-end Delay as a function of increasing Node-Speed and increasing number of Sessions.

5. CONCLUSIONS AND FUTURE WORK

In this paper we presented the design and performance of an efficient location service based on our novel associativity-based clustering strategy. By employing the dominating-set (CHs) to perform periodic location-updates on behalf of other nodes, we have demonstrated that our scheme leads to less control traffic when compared to the GLS. In addition, our location-management strategy conserves scarce resources such as battery energy and wireless bandwidth by preventing the location-updates, queries and responses from traversing the unnecessary parts of the ad hoc network. Mathematical analysis and simulation results confirmed the performance advantages of our scheme. In our future work, we have decided to construct longevity routes based on the proposed location service as our next step in realizing quality of service routing. We plan to report such findings in future papers.

6. REFERENCES

[1] Sivavakeesar, S.,, and Pavlou, G., Stable Clustering Through Mobility Prediction for Large-Scale Multihop Intelligent Ad Hoc Networks, *In Proceedings of the IEEE Wireless Communications and Networking Conference (WCNC'04)*, Georgia, USA, Mar. 2004, vol. 3, 1488 – 1493.

[2] Hong, X., Gerlo, M., Yi, Y., Xu, K., and Kwon, T.J., Scalable Ad Hoc Routing in Large, Dense Wireless Networks Using Clustering and Landmarks, *In Proceedings of the IEEE International Conference on Communications. (ICC'02),* Apr. 2002, vol. 25, no. 1, 3179 – 3185.

[3] Woo, S-C.M., and Singh, S., Scalable Routing Protocol for Ad Hoc Networks, Wireless Networks, Kluwer Academic Publishers, Sep. 2001, vol. 7, no. 5, 513 – 529.

[4] Philip, S.J., and Qiao, C., ELF: Efficient Location Forwarding in Ad Hoc Networks, *In Proceedings of the IEEE Global Telecommunications Conference (Globecom'03)*, Dec. 2003, vol. 22, no. 1, 913 – 918.

[5] Li, J., Jannoti, J., De Couto, D.S.J., Karger, D.R., and Morris, R., A Scalable Location Service for Geographic Ad Hoc Routing, *In Proceedings of the ACM. 6th International Conference on Mobile Computing and Networking (Mobicom'00)*, Aug. 2000, 120 – 130.

[6] Sucec, J., and Marsic, I., Location Management for Hierarchically Organized Mobile Ad hoc Networks, *In Proceedings of the IEEE Wireless Communications and Networking Conference (WCNC'02)*, Florida, USA, March 2002, 603-607.

[7] Mauve, M., Widmer, J., and Hartenstein, H., A Survey on Position-Based Routing in Mobile Ad Hoc Networks, IEEE Network, Nov./Dec. 2001, vol. 15, no. 6, 30 – 39.

[8] Hong, X., Xu, K., and Gerla, M., Scalable Routing Protocols for Mobile Ad Hoc Networks, IEEE Network, Jul./Aug. 2002, vol. 16, no. 4, 11 – 21.

[9] Toh, C-K., Lin, G., and Delwar, M., Implementation and Evaluation of an Adaptive Routing Protocol for Infrastructureless Mobile Networks, *In Proceedings of the IEEE International Conference on Computer Communications & Networks (IC3N'00),* Las Vegas, 2000.

[10] Sivakumar, R., Sinha, P., and Bharghavan, V., CEDAR: A Core-Extraction Distributed Ad hoc Routing Algorithm, IEEE Journal on Selected Areas in Communications, August 1999, vol. 17, no. 8, 1 – 12.

[11] Sucec, J., and Marsic, I., Clustering Overhead for Hierarchical Routing in Mobile Ad Hoc Networks, *In Proceedings of the IEEE Conference on Computer Communications (INFOCOM'02)*, New York, NY, USA, June 2002, 1698 – 1706.

[12] Zengu, X., Bagrodia, R., and Gerla, M., GloMoSim: A Library for Parallel Simulations of Large-scale Wireless Networks, *In the Proceedings of the 12th Workshop on Parallel and Distributed Simulations*, May 1998.

[13] Perkins, C.E., Belding-Royer, E.M., and Chakeres, I., Ad Hoc On Demand Distance Vector (AODV) Routing, *IETF Internet draft*, draft-perkins-manet-aodvbis-00.txt, Oct 2003 (Work in Progress).

[14] Johnson, D.B., Maltz, D.A., and Hu, Y-C., The Dynamic Source Routing Protocol for Mobile Ad Hoc Networks (DSR), *IETF Internet draft*, draft-ietf-manet-dsr-10.txt, Jul. 2004.