CLUSTER-BASED LOCATION-SERVICES FOR SCALABLE AD HOC NETWORK ROUTING

S. Sivavakeesar and G. Pavlou
Center for communication systems research, University of Surrey, Guildford, Surrey GU2 7XH, United Kingdom

Abstract: We propose a location-service to assist location-based routing protocols, realised through our 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. We evaluate our proposed scheme through simulations, and the results indicate that our protocol scales well with increasing node-count.

Key words: Ad hoc Networks, Scalable Location Management, Hierarchical Clustering Protocol, Scalable Routing.

1. INTRODUCTION

A mobile ad hoc network (MANET) is a network without any pre-existing infrastructure, and this paper considers the problem of routing in such networks of large-scale. In multihop MANETs, the routing protocol is the key to efficient operation. 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[1,8]. We use the concept of *virtual-clusters* we 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 (i.e., CHs handle the location-updates on behalf of their members). 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 II examines related previous work, and presents our motivation. The novel associativity-based clustering protocol and the proposed location-service technique are described in section III. Section IV evaluates the proposed scheme through simulations, and demonstrates that our location-service results in less signaling traffic in comparison to other similar methods. Section V presents our conclusions and future work.

2. RELATED WORK AND OUR MOTIVATION

This section briefly explains the basic principles and problems associated with location-based routing protocols. It is then followed by a brief review of key location-service techniques and popular clustering protocols in the literature.

2.1 Basic Principles 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 neighbours[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. Due to the communication complexity and periodic flooding 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. 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. 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 III.

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.

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

In order to make our clustering mechanism scalable, we make use of the notion of *virtual-clusters* we introduced in[1]. The idea is that a geographical area is divided into equal regions of circular shape 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 centres[1].

Our clustering protocol does not involve any extra control traffic; instead, periodic *HELLO* messages – as in other similar location-management approaches – is 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 kth *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 kth cluster by calculating the following:

1. 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 kth *virtual-cluster*, whose center’s Cartesian co-ordinates are (x_{ck}, y_{ck}), its distance at time ‘t’ can be calculated by:

$$d_{xk}(t) = \sqrt{(x_{xk}(t) - x_{ck})^2 + (y_{xk}(t) - y_{ck})^2} \qquad (1)$$

2. 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 jth *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).

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

$$N_{mean_X} = \frac{R_{mean_X}}{ASSOCIATIVITY\_TICK\_PERIOD} \tag{3}$$

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

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 III.B.

3.2 Homezone-based Hierarchical 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.

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

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 make its own location-updates, if it has not become a member of any cluster. In our scheme, it is assumed that whenever any node's speed increases beyond μ_{TH}, it will not be considered as a member of any cluster, and thus required to initiate its own location-updates. 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"[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.

4. EVALUATION THROUGH SIMULATION

We chose GLS – another similar location-management approach – in our attempt to compare the performance of our strategy. 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 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}. Each scenario was run for a 300 simulated seconds. Due to space limitation, we analyze the scalability of our location-service only 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 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 were selected for each run. The transmission range of each node is 100 m, and the wireless link capacity 2 Mbps. Traffic is generated using random CBR connections having a payload size of 512 bytes. These CBR connections are randomly generated so that at any moment the total number of source-destination pairs is kept constant – and each session lasts for a time-period which is uniformly distributed between 40 and 50 seconds.

Fig. 1 shows the average control cost (routing related cost) incurred per node for different CBR traffic sessions. Here the control cost includes periodic location-updates, *HELLO* packets, location queries and responses. As it can be seen, the control cost is lower in the case of our location-service, and hence our scheme outperforms GLS. Fig. 2 demonstrates the fact that in our location-management strategy the location-update packets don't traverse the unnecessary parts of the network. This is determined by considering the average number of location-updates that are relayed by each node in the network. As it can be seen from Fig. 2, although location-updates are much lower than that of GLS, the average number of location-update packets relayed by any node in our scheme slightly increases with the number of nodes. This can be attributed to the hash-function selected for simulation purposes. As the number of nodes within the network increases, the terrain-area size is also increased. With this increase, the total number of *virtual-clusters* or home-zones within the considered area also increases. As a result, some nodes may select far-away *virtual-clusters* as their home-zones.

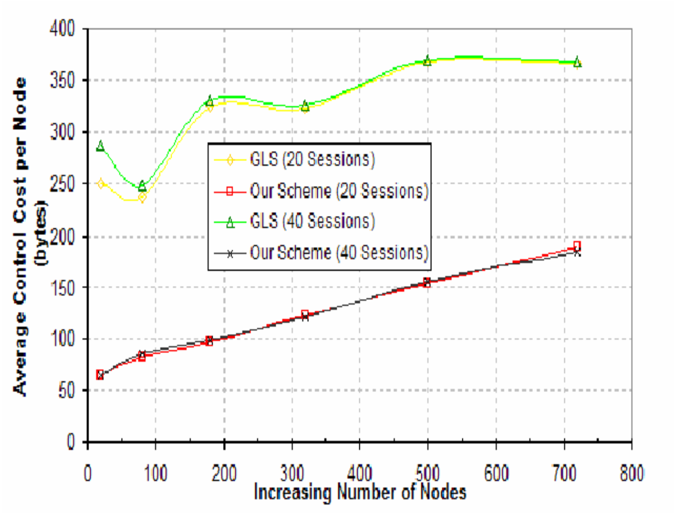

Figure 1. Average control cost incurred per node as a function of increasing number of node-count

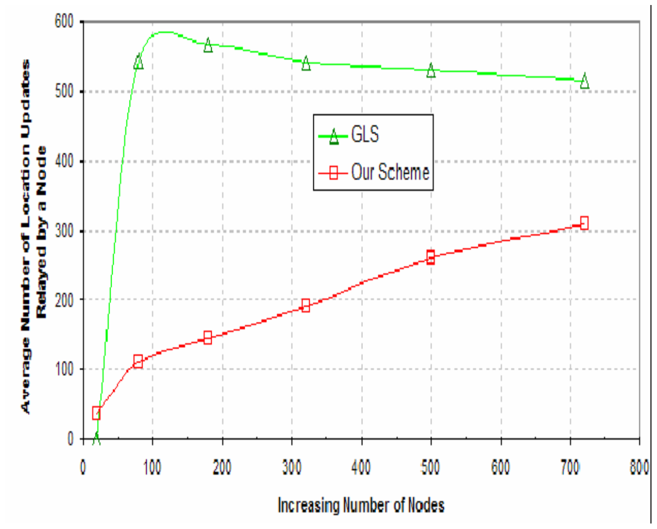

Figure 2. Average number of location-updates packets relayed by a node as a function of increasing number of nodes

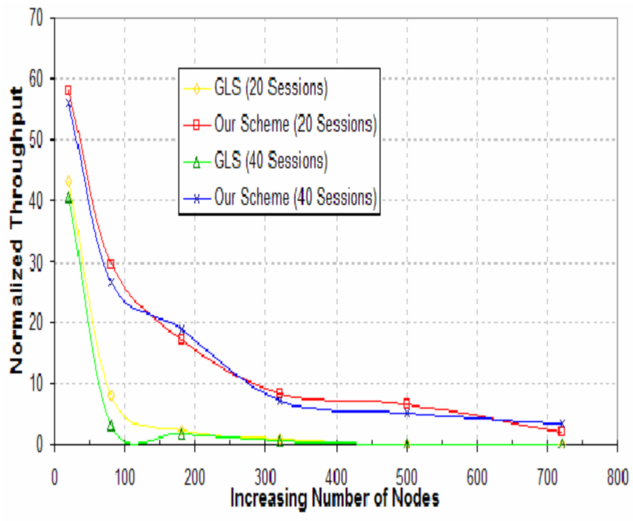

Figure 3. Normalized throughput as a function of increasing number of nodes and increasing number of sessions

Fig. 3 depicts the normalized throughput of both schemes as a function of increasing number of nodes under different traffic scenarios. The normalized throughput is defined as the total number of packets actually delivered to their respective destinations divided by the total number of packets generated within the whole network. Although the throughput in our scheme is higher than that of GLS, the throughputs in both schemes tend to decrease as the number of nodes increases. This is due to the fact the link capacity was 2 Mbps, and it poses the main bottleneck in the scenarios considered.

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.

ACKNOWLEDGEMENT

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.

REFERENCES

1. S.Sivavakeesar, and G.Pavlou, "Stable Clustering Through Mobility Prediction for Large-Scale Multihop Intelligent Ad Hoc Networks", IEEE WCNC, Mar. 2004.
2. X.Hong, M.Gerlo, Y.Yi, K.Xu, and T.J.Kwon, "Scalable Ad Hoc Routing in Large, Dense Wireless Networks Using Clustering and Landmarks", *Proc. Int'l Conf. on Communications.* (ICC 2002), vol. 25, no. 1, Apr. 2002, pp. 3179 – 3185.

3. S-C.M.Woo, and S.Singh, “Scalable Routing Protocol for Ad Hoc Networks”, Wireless Networks, Kluwer Academic Publishers, vol. 7, no. 5, Sep. 2001, pp. 513 – 529.
4. S.J.Philip, and C.Qiao, “ELF: Efficient Location Forwarding in Ad Hoc Networks”, *Proc. Global Telecommunications Conference.* (Globecom 2003), vol. 22, no. 1, Dec. 2003, pp. 913 – 918.
5. J.Li, J.Jannoti, D.S.J.De Couto, D.R.Karger, and R.Morris, “A Scalable Location Service for Geographic Ad Hoc Routing”, *Proc. 6th Int'l Conf. on Mobile Computing and Networking* (Mobicom 2000), Aug. 2000, pp. 120 – 130.
6. J.Sucec and I.Marsic, “Location Management for Hierarchically Organised Mobile Ad hoc Networks”, Proc. IEEE WCNC 2002, March 2002, pp. 603-607.
7. M.Mauve, J.Widmer, and H.Hartenstein, “A Survey on Position-Based Routing in Mobile Ad Hoc Networks”, IEEE Network, vol. 15, no. 6, Nov./Dec. 2001, pp. 30 – 39.
8. X.Hong, K,Xu, and M.Gerla, “Scalable Routing Protocols for Mobile Ad Hoc Networks”, IEEE Network, vol. 16, no. 4, Jul./Aug. 2002, pp. 11 – 21.
9. C-K.Toh, G.Lin, and M.Delwar, “Implementation and Evaluation of an Adaptive Routing Protocol for Infrastructureless Mobile Networks”, *Proc. Int'l Conf. on* Computer Communications & Networks (IEEE IC3N), Las Vegas, 2000.
10. R.Sivakumar, P.Sinha, V.Bharghavan, “CEDAR: A Core-Extraction Distributed Ad hoc Routing Algorithm”, IEEE Journal on Selected Areas in Communications, vol. 17, no. 8, August 1999, pp 1 – 12.
11. J.Sucec, and I.Marsic, “Clustering Overhead for Hierarchical Routing in Mobile Ad Hoc Networks”, Proc. IEEE INFOCOM'2002, June 2002, pp. 1698 – 1706.
12. X.Zengu, R.Bagrodia, and M.Gerla, “GloMoSim: A Library for Parallel Simulations of Large-scale Wireless Networks”, Proceedings of the 12th Workshop on Parallel and Distributed Simulations, May 1998.