Scalable Routing Strategies for Multi-Hop Ad-hoc Wireless Network

Atsushi Iwata, Ching-Chuan Chiang, Guangyu Pei, Mario Gerla and Tsu-wei Chen
Computer Science Department
University of California, Los Angeles

Abstract

In this paper we consider a large population of mobile stations which are interconnected by a multihop wireless net. The applications of this wireless infrastructure range from ad hoc networking (eg, collaborative, distributed computing) to disaster recovery (fire, flood, earthquake), law enforcement (eg, crowd control), search and rescue and battlefield. Key characteristics of this system are the large number of users, their mobility and the ability to operate without the support of a fixed (wired or wireless) infrastructure. The last feature sets this system apart from existing cellular systems and in fact makes its design much more challenging.

In this environment, we investigate routing strategies which scale well to large number and can handle mobility. In addition to large number and mobility, we address also the need to support multimedia communications, with low latency requirements for interactive traffic and QoS support for real time streams (voice/video). In the wireless routing area, several schemes have already been proposed and implemented (eg, hierarchical routing, on-demand routing etc). We discuss the pro's and con's of the existing schemes, and introduce two new schemes - Fisheye state routing (FSR) and Hierarchical State Routing (HSR) - which offer some competitive advantages over the existing schemes. We compare the performance of existing and proposed schemes via simulation.

1 Introduction

In this paper we consider a large population of mobile stations which are interconnected by a multihop wireless network. A key feature of multihop wireless networks, which sets them apart from the more traditional and ubiquitous cellular radio systems, is the ability to operate without a fixed, wired communications infrastructure, and to be rapidely deployed to support emergency requirements, short term needs, coverage in undeveloped areas or other special services not available from public providers.

The applications of this wireless infrastructure range from civilian (such as ad hoc networking for collaborative, distributed computing) to disaster recovery (fire, flood, earthquake), law enforcement (e.g., crowd control, search and rescue) and military (automated battlefield). Key characteristics of these systems are the large number of users, their mobility and the need to support multimedia communications. The last requirement stems from the fact that in mobile, wireless scenarios human to human communications play a very important role (more than in typical wired network connectivity scenarios). For example, in battlefield the most critical applications are voice and imaging (albeit not the most bandwidth demanding). As a consequence, real time traffic support, with QoS constraints is important, and so is multicasting, when the application calls for team collaboration. Another critical requirement is low latency access to distributed resources (e.g., distributed database access for situation awareness in the battlefield).

In this environment, we investigate several routing strategies with focus on solutions which scale well to large number and can handle mobility. This problem is practically relevant since it is very likely that in the near future most of the commercial laptops and PDAs will be equipped with radios enabling them to form ad hoc "virtual" wireless networks. The problem is also very challenging, because of the combination of large number of users and mobility. In fact, large wireless networks can be effectively handled with conventional hierarchical routing, if nodes are stationary. When nodes move, however, the hierarchical partitioning must be dynamically changed - a significant challenge. Mobile IP [1] solutions can handle mobility well when there is a fixed infrastructure supporting the concept of fixed "Home Agent". When all nodes move (including the Home Agent) such strategy is less obvious.

If we consider prior work in the wireless routing area, we note that several schemes have already been proposed and implemented (e.g., hierarchical routing, on-demand routing etc). In section 2 we discuss the advantages and limitations of the existing schemes. We then introduce in section 3 two new schemes - Fisheye state routing (FSR) and Hierarchical state routing (HSR) - which overcome some of the limitations of the existing schemes. In section 4 we present network model and simulation results. Section 5 concludes the paper.

2 Background and prior work

Existing wireless routing schemes can be classified into three broad categories:

(a) global, precomputed routing: routes to **all** destinations are computed a priori and are maintained in the background via a periodic update process. Most of the conventional routing schemes, including Distance Vector and Link State, fall in this category.

(b) on-demand routing: the route to a specific destination is computed only when needed

(c) flooding: packets are broadcast to all destinations, with the expectation that one packet at least will reach the intended destination. Scoping may be used to limit the overhead of flooding.

In the sequel we review in more detail the contributions in the first two categories, exposing some of the limitations of the existing schemes.

2.1 Global routing schemes

Global routing schemes can be subdivided into two further categories: flat and hierarchical.

In the "flat routing"category, many protocols have been proposed to support mobile ad-hoc wireless routing. Some proposals are based on the results developed in traditional wired network. For example, Perkins' Destination-Sequenced Distance Vector (DSDV) [12] is based on Distributed Bellman-Ford (DBF), Garcia's Wireless Routing Protocol (WRP) [13] [14] is based on a loop-free path-finding algorithm, etc. These routing schemes provide solutions for flat networks. That is, a node has to maintain a routing table containing all the nodes in the network. This is fine if the user population is small. However, as the number of mobile hosts increases, so does the overhead. This makes most of the wireless routing schemes based on traditional routing algorithms not scalable to large networks.

To permit scaling, hierarchical techniques can be used. In the following, we first briefly review the hierarchical schemes proposed in wired networks. Then, we describe a hierarchical scheme recently proposed for wireless networks.

2.1.1 Hierarchical routing for wired network

As the size of the routing table grows linearly (one entry per node) with the number of nodes, the storage, processing and line overhead associated with routing table maintenance and updating becomes prohibitive in very large networks.

In order to make dynamic routing more scalable, one potential solution is hierarchical clustering and routing. The first hierarchical routing proposal was by McQuillian [2]. The characteristics and performance of the hierarchical approach were later evaluated and optimized by F.Kamoun and L.Kleinrock in [3]. Their scheme was designed to minimize storage of routing tables, and to propagate and update them very effectively without introducing significant performance degradation. Drawing from the work in [2] [3], C.Alaettinoglu proposed a routing scheme based on the distributed Bellman-Ford algorithm [6], and Ramamoorthy et al. [4][5] proposed a scheme based on link state algorithm.

The most recent effort in hierarchical clustering and routing is "Private Network Network Interface (PNNI)" routing [7], a hierarchical link state routing protocol for ATM networks which has been standardized by the ATM Forum. PNNI [7] uses the same concept as [2] [3] to create hierachical clustering. It enhances the existing link state routing, Open Shortest Path First (OSPF) [8] , so that it can support several hierarchy levels. It also includes multiple QOS link state parameters to support QOS sensitive routing, which is a key for multimedia communication [29].

The basic concepts of most hierarchical clustering algorithms are quite similar. Taking PNNI as an example, Figure 1 illustrates the structure of the PNNI hierarchical routing scheme. In this figure, the solid circles, e.g. A.1, A.2, A.3 and B.1, correspond to groups of subnetwork, called "peer groups", which are equivalent to "clusters". Above the first layer, dotted circles, A, B, and X, show logical peer groups of higher level, consisting of several logical group nodes (LGN) which represent lower level peer groups. Each node maintains a hierarchical view of the network topology and can compute the same route to destination. For example, node A.1.2, can stamp the route [A.1.2,A.1.1][A.1,A.3][A,B], in the SETUP signaling packet for a

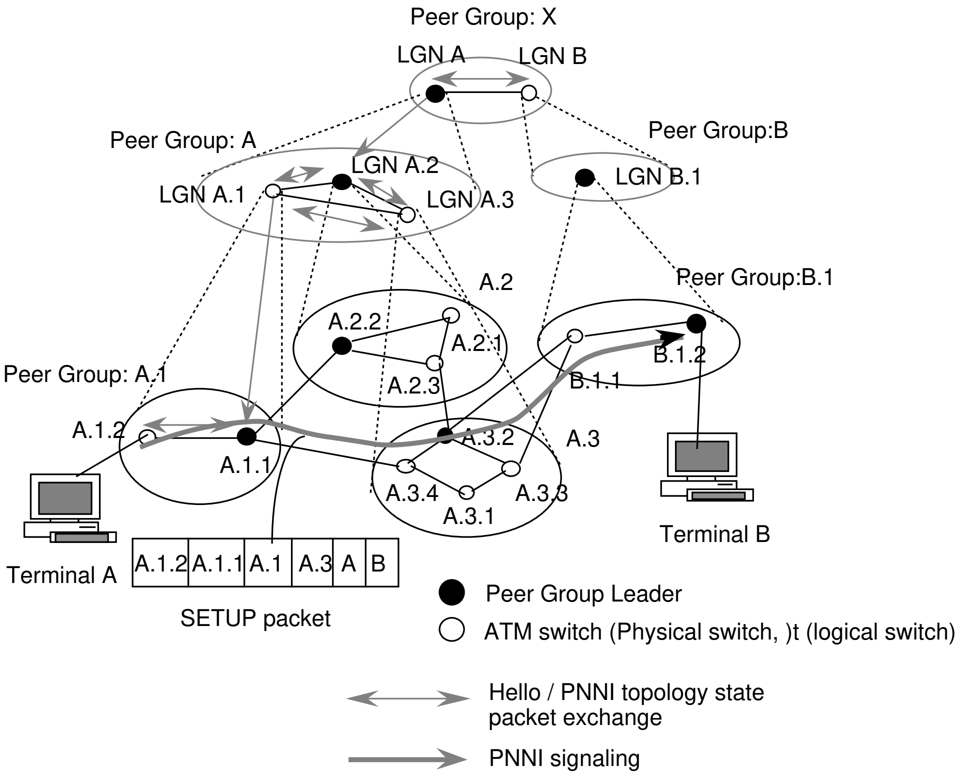

Figure 1: An example of PNNI Routing

connection to B.1.2.

Several enhancements have been introduced in the basic hierarchical routing scheme. For example, in order to prevent loops, Murthy and Garcia-Luna-Aceves propose a hierarchical routing algorithm called "Hierarchical Information Path-based Routing (HIPR)" [10], which combines McQuillian's clustering scheme and the loop-free path finding algorithm [9]. To further reduce hierarchical routing overhead, Behrens proposes a hierarchical link vector routing algorithm [11], where nodes propagate incremental routing information only about those links which they actually use to reach any destination (i.e., nodes keep a partial link state information).

The above hierarchical schemes were developed for wired networks. If we now consider mobile, multi-hop wireless networks, we immediately realize that having a pre-assigned hierarchical address is not feasible, since nodes move around . Thus, the hierarchical structure must be constantly adjusted to reflect topology changes. In the following section, we review a solution addressing this problem.

2.1.2 Hierarchical routing for wireless network

A hierarchical clustering and routing approach specifically designed for wireless networks was recently proposed in [22] [23]. The proposal addresses the link and network layers only, and is independent of the physical/MAC layer, thus supporting various MAC layer implementations.

In this model, the network contains two kinds of nodes, endpoints and switches. Only end points can be sources and destinations for user data traffic, and only switches can perform routing functions. Endpoints choose the most affordable switches by checking radio link quality. Autonomously, they group themselves into cells around those switches (cluster heads). This procedure is called "cell formation". Each endpoint is within one hop of the switch with which it is affiliated. These switches also organize themselves hierarchically into clusters, each of which functions as a multihop packet-radio network. In turn, clusters may themselves form higher-level clusters, and so on. This procedure is called "hierarchical clustering". (A switch is a level 0th cluster). As nodes move, clusters may split or merge, altering cluster membership.

To support data transfer between mobile nodes, it is necessary to keep track of the location of mobile nodes. This scheme uses both paging and query/response with a location manager to locate nodes. Each cluster has a location manager which keeps track of node locations within the cluster and assists in locating nodes outside the cluster. Each node has a roaming level which is specified with respect to the clustering hierarchy and which implicitly defines a roaming cluster. Paging is used to locate a mobile node within its current roaming cluster. When a node moves outside of its current roaming cluster, it sends a location update to the location manager. This update propagates to the highest level from which inter-cluster movement is visible. By combining these hierarchical topology management and location management functions, hierarchical routing is extended to the mobile environment.

This scheme, however, creates implementation problems which are potentially complex to resolve. First, it does allocate Cluster IDs dynamically. This allocation must be unique - not an easy task in multi-hop mobile environment, where the hierarchical topology must be often reconfigured. Second, each cluster can dynamically merge and split, based on the number of nodes in the cluster. This scheme may cause a frequent changes of cluster head, degrading the network performance significantly. Since the diameter of this cluster is variable, it is also difficult to predict how long it takes to propagate clustering control messages among nodes, and therefore it is difficult to bound the convergence time of the clustering algorithm. Third, the paging and query/response approach to locate mobile nodes may lead to control message overhead. Fourth, if the location manager moves from the current location, this scheme proposes a migration to another location server. This scheme requires a complex consistency maintenance of the location databases between new and original server.

2.2 On demand routing schemes

On-demand routing is the most recent entry in the class of wireless routing schemes. It is based on a query-reply approach. Examples include the Lightweight Mobile Routing (LMR) protocol [15], Ad-hoc On Demand Distance Vector Routing (AODV) [18], Temporally-Ordered Routing Algorithms (TORA) [16] [17], Dynamic Source Routing Protocol (DSR) [19] and ABR ([42]). Typically, on-demand routing aims at providing solutions for networks with fast changing topologies. IETF's MANET working group is also focusing on the on-demand routing solution for an Ad Hoc Network Standard [18] [17] [19].

There are many different schemes for discovering routes in On Demand algorithms. Most algorithms, however, employ a scheme derived from LAN bridge routing , ie, route discovery and backward learning. The source in search for a path floods a query into the network. The transit nodes upon receiving the query "learn" the path to the source (backward learning) and enter the route in the forwarding table. The intended destination eventually receives the query and can thus respond using the path traced by the query. This permits to establish a full duplex path. The query packet is dropped on its way to destination if it encounters a node which already has a route to such destination. After the path has been computed, any link failure will trigger another query/response so that the routes are always kept up to date.

An alternate scheme for tracing on demand paths (also inspired to LAN bridge routing) is source routing. In this case, the query packet picks up the IDs of the intermediate nodes. The destination can then retrieve the entire path to the source from the query header, and can use it to respond to the source thus establishing a path in the reverse direction.

On-demand routing does scale well to large population as it does not regulary maintain a routing table for all destinations. Instead, as the name suggests, a route to a destination is computed only when there is a need. Thus, routing table storage is greatly reduced, if the traffic pattern is sparse. However, on-demand routing introduces the less desirable initial latency which makes it not very efficient for interactive traffic (e.g., distributed database query applications). It is also impossible to know in advance the quality of paths to all destinations (e.g., bandwidth, delay etc.) - a feature which can be very effective in call acceptance and path selection of QoS oriented connections.

Zone routing [20] [21] can be viewed an extention of on-demand routing, since it is based on a hybrid of on-demand routing and conventional routing . In fact, zone routing represents a first step towards hierarchical on-demand routing. For routing inside the zone, any routing scheme, including Distributed Bellman-Ford (DBF) routing or Link State (LS) routing, can be applied. For interzone routing, on-demand routing is used. The advantage of zone routing is its scalability, as it reduces the need for routing table storage. At the same time, the efficiency of global routing is preserved within each zone. However, for interzone routing, the on-demand solution poses the usual problems of connection latency and QoS reporting.

3 Scalable wireless routing scheme

In this section we develop novel solutions which specifically address the joint large scale and mobility requirements, overcoming some of the limitations of the existing schemes.

As discussed in previous sections, flat routing schemes [12] [13] [14] cannot scale to a large network. On-demand routing [15] [18] [16] [17] [19] and zone routing [20] [21] do scale to a large network, but have latency and QoS support limitations. Hierarchical routing solutions [2] [3] [6] [4] [5] [7] for wired networks can be a good starting point, but must be extended to support dynamic mobility management. The wireless hierarchical routing proposal in [22] [23] is a first attempt in this direction. However, the protocol is overhead prone and quite complex to maintain.

Our proposed approach is based on global routing principles. We explore two different directions, addressing scalability and mobility support in both, namely:

(a) Fisheye State Routing (FSR), and

(b) Hierarchical State Routing (HSR)

Both schemes are intended as network layer protocols. They are designed to be independent of MAC layer and radio layer platforms, and can work with a variety of these platforms. Since link state (LS) based routing converges more quickly after a topological change, and is more suitable for Quality of Service (QoS) based routing, the LS routing approach will be chosen as a basis for both protocols. Section 3.1 and 3.2 discuss each scheme in detail.

3.1 Fisheye state routing (FSR) scheme

In [28], Kleinrock and Stevens proposed the fisheye technique to reduce the size of information required to represent graphical data. The original idea is to maintain high detail for information within a range of a certain point of interest; and less detail as the distance from the point of interest increases. For routing, this fisheye approach can be interpreted as maintaining a highly accurate network information about the immediate neighborhood of a node, with progressively less detail as we move away from the node.

Our Fisheye State Routing scheme is built on top of another recently proposed routing scheme called "Global State Routing" (GSR) [27], which computes accurate routing decisions using global network information. Fisheye state routing improves the scalability of GSR as it filters routing information selectively, based on distance from its origin. We first review GSR and then introduce the fisheye concept.

GSR is functionally similar to Link State Routing in that it creates a topology map at each node. The key difference between GSR and traditional LS is the way routing information is disseminated. In LS, link state packets are generated and flooded into the network whenever a node detects a topology change. GSR does not flood the link state packets. Instead, nodes in GSR maintain a link state table based on the up-to-date information received from neighboring nodes, and periodically exchange it with their local neighbors only. Information is disseminated as the link state with larger sequence numbers replaces the one with smaller sequence numbers. In this respect, GSR is similar to Distributed Bellman Ford (or more precisely, DSDV [12]) where the distances are updated according to the time stamp or sequence number.

In a wireless environment, a radio link between mobile nodes may experience frequent disconnects and reconnects. This makes it difficult to maintain topology data base consistency if the LS protocol is used. GSR solves this problem by using periodical exchange of the entire topology database with sequence numbers for each entry. GSR greatly reduces the number of control messages required to maintain uptodate topology and link state information throughout the network [27].

The drawback of GSR is the large size update message which consumes considerable amount of bandwidth. This is where the Fisheye technique comes to help, and reduces the size of update messages without seriously affecting routing accuracy.

3.1.1 Fisheye Protocol overview

Figure 2 illustrates the application of fisheye in a mobile, wireless network. In this figure, we show the scope of fisheye for the center node. The small circles with a number inside represent the mobile hosts in the network. The large circles define the fisheye scope of the center node. The scope of fisheye is defined as the nodes that can be reached within a certain number of hops. In our case, three scopes are shown and they represent the scope of 1-hop, 2-hop and 3-hop. Nodes that are located within scopes of different hop distances are plotted as black, grey and white, representing scope of 1-hop, 2-hop and 3-hop, respectively.

The reduction of update message size is achieved by updating the network information for nearby nodes at a higher frequency than for remote nodes which are outside the fisheye scope. The routing update procedure scans through the update message and filters out entries that have hop distance larger than the fisheye scope. Figure 3 depicts this operation. In this figure, entries in bold face indicate the messages selected for dissemination into the network. The rest of the entries will still be sent out later, at a much lower frequency. As a result, a considerable fraction of link state entries are suppressed so that the message size is reduced.

In summary, FSR scales well to large networks, by keeping link state exchange O/H low. Yet, it retains one routing entry for each destination (as flat routing does), avoiding the extra work of "finding" the destination (through a home agent, for example) and maintaining low fisrt packet transimission latency. As mobility increases, routes to remote destinations become less accurate. However, when a packet approaches its destination, it finds increasingly accurate routing instructions as it enters zones with higher refresh rate. Link state updates to remote destinations must be issued with a frequency which increases with mobility, so that paths to such destinations are not interrupted (albeit they may be non-optimal).

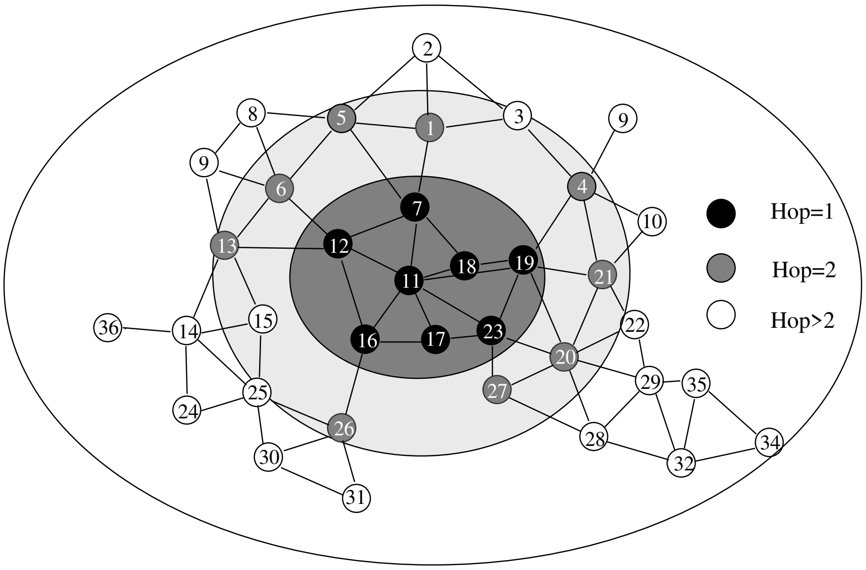

Figure 2: Scope of fisheye

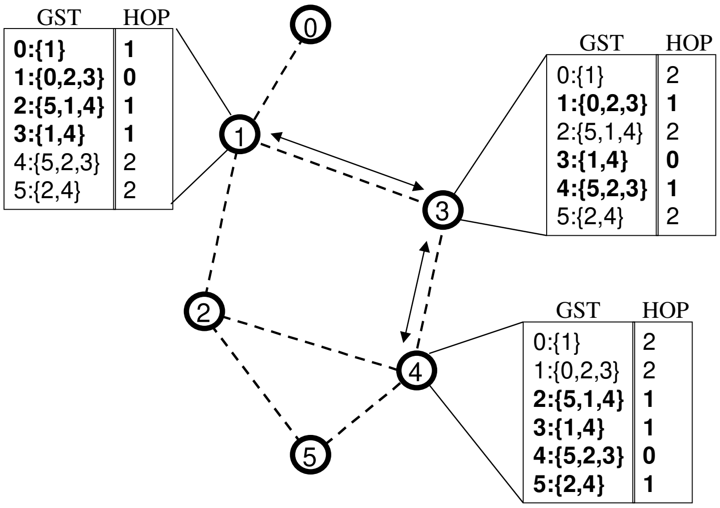

Figure 3: Message reduction using fisheye

3.2 Hierarchical state routing scheme

Hierarchical partitioning is common practice in multi-hop mobile wireless networks. For example, several clustering algorithms [24] [25] have been proposed to partition the multi-hop network so that by using different spreading codes across clusters interference is reduced and spatial reuse of channels is enhanced. As the number of nodes grows in this environment, there is further incentive to partition the network in order to reduce routing overhead. The main drawback of hierarchical partitioning in a mobile network is location managenment. In this paper, we propose a scheme which combines distributed multi-level hierarchical clustering and distributed location (membership) management.

The proposed clustering scheme maintains a hierarchical topology, where elected cluster-heads in a lower level become members of the next higher level and form other higher level clusters at this level, and so on. The multilevel clustering algorithm consists of two steps, physical level clustering and logical level clustering. Physical level clustering aims at the efficient utilization of radio channel resources, which are related to MAC layer performance. Logical level clustering aims at reducing network-layer routing overhead (i.e., routing table storage, processing and transmission).

The proposed location (membership) management scheme tracks mobile nodes, while keeping the control message overhead low. Membership management is based on a distributed location server approach akin to the mobile IP concept. The following sections give more details of these schemes.

3.2.1 Physical level (MAC level) clustering

Figure 4 shows an example of physical and logical clustering. $Level = 0$ of Fig. 4 indicates physical level clusters (C0-1, C0-2,C0-3,and C0-4; C means a cluster and 0 means a used level.) Physical level clustering provides an effective way to allocate wireless channels among different clusters. A variety of clustering algorithms have been proposed and can be used for this purpose [24] [25]. Based on previous research results [25], Lowest Node ID algorithm is used to create clusters . Spread-spectrum radios can be utilized to permit code division multiple access (CDMA) and spatial reuse across clusters. Within a cluster, the Medium Access Control (MAC) layer can be implemented by using a variety of different schemes (polling,

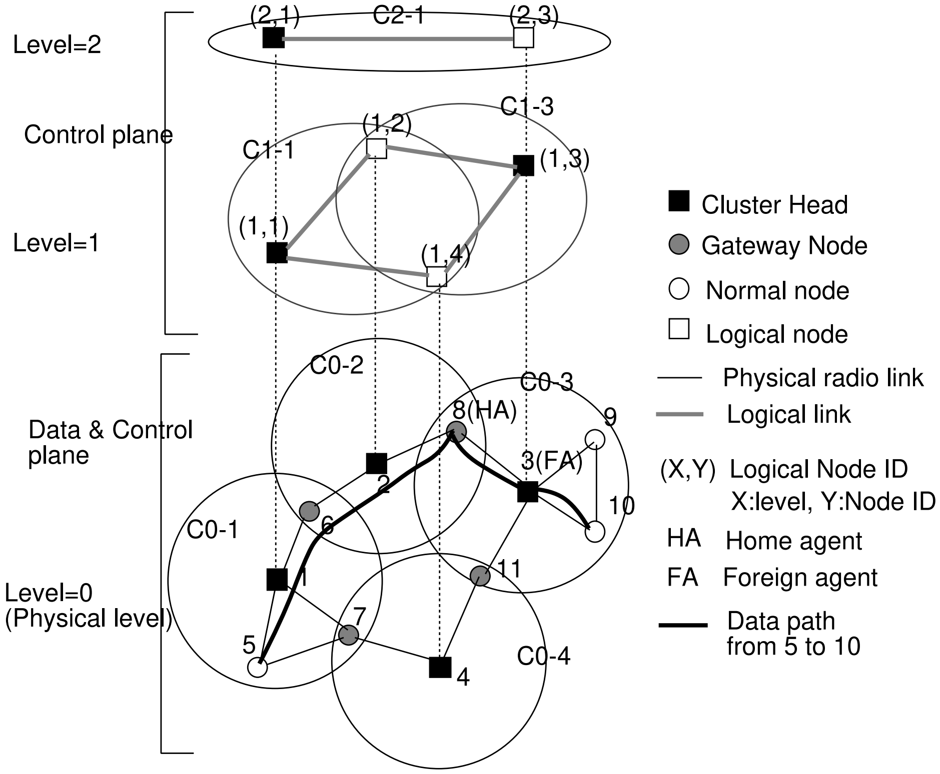

Figure 4: An example of physical/logical clustering

MACA, CSMA, TDMA etc.) [26]. Considering the example in Fig. 4, each cluster overlaps with neighbor clusters [24] [25]. There are three kinds of nodes in this clustering algorithm, cluster-head node (e.g. Node 1, 2, 3, and 4), gateway node (e.g. Node 6, 7, 8, and 11), and regular node (e.g. 5, 9, and 10). The cluster-head node acts as a local coordinator of transmissions within the cluster. It differs from the base station concept in current cellular system in that it is dynamically selected among the set of stations.

The node addresses shown in Fig. 4 are MAC layer addresses (i.e., they are hardwired and unique to each node). In addition to MAC addresses, nodes are assigned IP addresses of the type $< subnet, \; host >$. Each subnet corresponds to a particular user group (e.g., tank batallion in the battlefield, search team in a search and rescue operation, etc.). The notion of subnet is important because each subnet is associated with a home agent, as explained later. Also, a different mobility pattern can be defined independently for each subnet. This allows us to define the movements of different formations (e.g., members of a police patrol). The transport layer delivers to the network a packet with the IP address. The network resolves the

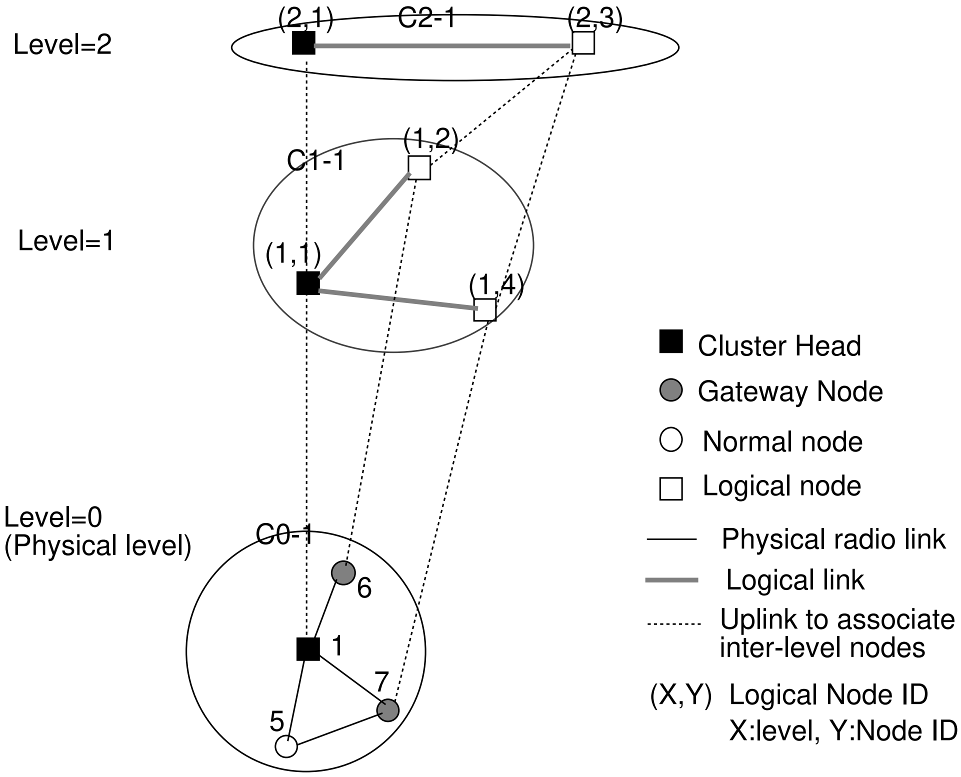

Figure 5: View of the network from node 5

IP address into a hierarchical (logical) address which includes the MAC address. Details will be discussed in the following sections.

Within a physical cluster, each node detects neighbor link state information and broadcasts it to other nodes within the same cluster. The cluster head summarizes link state information within its cluster and passes it up to a higher level cluster.

3.2.2 Logical level clustering

Logical level clusters exist above the physical clusters. Referring to Fig. 4, $Level = 1$ *and* 2 indicates logical level clusters (C1-1 and C1-3 in level 1, C2-1 in level 2).

Elected lower level clusterheads become "logical" nodes on the next higher logical level (e.g. Node 1 is elected as a cluster head and becomes a logical node (2,1) of level 1st cluster C1-1). These logical nodes get new IDs, consisting of level ID and physical level (i.e., MAC layer) ID. For example, node (2,1) means logical level 2 and MAC address 1. This way, the logical node ID is uniquely defined across the entire network, and can be easily adapted to dynamic topology change.

The logical nodes are connected by logical links, which can be established with "tunnels" between lower level clusters. They can be recursively organized in higher level logical clusters using the same algorithm as the physical level clustering.

Logical nodes within the cluster exchange logical link state information as well as summarized lower level cluster information. After obtaining the link state information at this logical level, each logical node floods it down to nodes within the lower level cluster. As a result, each physical node has a "hierarchical" topology information, as opposed to full topology view as in flat LS schemes.

Figure 5 shows an example of hierarchical topology viewed from node 5 in Figure 4. Node 5 can see link state information for C0-1 for level 0th, C1-1 for level 1st, and C2-1 for level 2nd. Figure 6 shows an example of hierarchical routing table of node 5, consisting of level-by-level link state information with inter-level link (uplink) state information. Level-by-level link state information has three tables for level 0th, 1st, and 2nd, and uplink state information has two tables for level 0th-1st and level 1st-2nd.

Given (1) the number of nodes are n^L, (2) the average number of nodes in each cluster or logical cluster are n, and (3) the number of hierarchical levels are L, then flat link state routing requires $O(n^(2L))$ entries, and the proposed hierarchical routing requires only $O(n^2 \times L)$ entries in the hierarchical map. Thus, a number of nodes that each node can see is greately reduced by introducing the hierarchical topology. Here, we assume that the number of nodes per cluster, n is the same for all levels. This follows the reslts of an optimization study by Kamoun and Kleinrock [3].

3.2.3 Location (Membership) management of mobile nodes

A node does not know which logical cluster a particular destination belongs to, except for those destinations within the same lowest level cluster. The distributed location server assists in finding the destination. The approach is similar to mobile IP, except that here the home agent may also move. Physical nodes are grouped into user IP subnetworks. Typically, nodes in the same subnetwork have common characteristics, eg, tanks in the same batallion, professionals on the move belonging to the same company, students within same class,etc. The user subnetwork is a "virtual" IP subnetwork which spans several physical clusters. Note that the subnet address is totally distinct from the MAC address. Each virtual subnetwork has at least one home agent

Hierarchical link state table at node 5

Intra-level link state Table

Level=0

	1	5	6	7
1	0	1	1	1
5	1	0	0	1
6	1	0	0	0
7	1	1	0	0

Level=1

	(1,1)	(1,2)	(1,4)
(1,1)	0	2	2
(1,2)	2	0	0
(1,4)	2	0	0

Level=2

	(2,1)	(2,3)
(2,1)	0	4
(2,3)	4	0

Inter-level link (Uplink) state table

Level=0-1

	(1,2)	(1,4)
1	0	0
5	0	0
6	1	0
7	0	1

Level=1-2

	(2,3)
(1,1)	0
(1,2)	2
(1,4)	2

Figure 6: An example of hierarchical link state table of node 5

to manage membership. Each member knows the physical ID of the home agent (it is listed in the routing table), and registers its own current hierarchical address at the home agent, periodically as well as whenever the member moves to a new cluster. The registered addresses are managed by soft state timer which obsoletes the entries when it time-outs.

When a node wants to send a packet to a destination node of which it knows the IP address, it extracts first the subnet address subfield from it. From the routing table it finds the hierarchical address of the corresponding home agent. It then sends the packet to the home agent using the hierarchical address. The home agent establishes a "tunneling path" to the foreign agent, which is actually the lowest level clusterhead communicated by the destination node to the home agent. (Note that a home agent has reachability information to the foreign agent, but not to every possible destination.) If any intermediate node on the path to the agent knows (e.g., has cached) the location of the destination node, it may forward the packet directly to the foreign agent or the destination on behalf of the home agent. Furthermore, different from mobile IP concept, the home agent can move from cluster to cluster, and the foreign agent can also change quite frequently. To permit efficient address resolution, home agent reachability is monitored by neighor clusterheads, which propagate this reachability information to the entire network by piggybacking it on the routing tables.

Figure 7 shows an example of the location management table at node 5 for both approaches.

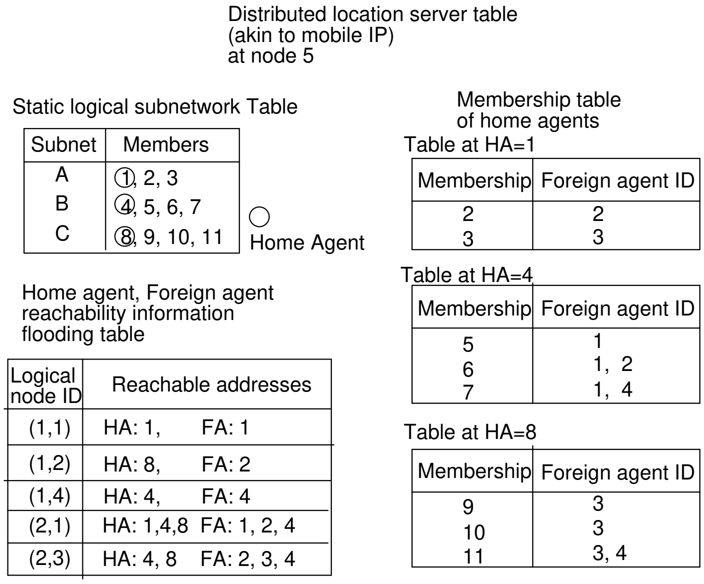

Figure 7: An example of location management table at node 5

3.3 Control overhead analysis

In this section, we analyze the control overhead (O/H) of the proposed schemes, Fisheye state routing (FSR) and Hierarchical state routing (HSR), and compare them with other routing schemes: Distributed Bellman-Ford (DBF), Link State (LS) and On-demand (OD) routing. The control O/H is studied under five aspects: (1) Memory overhead (O/H) (MO): the memory space required to store the routing information; (2) Line O/H (LO): the aggregate volume of control bytes exchanged by a node in each time slot; (3) Control Packet O/H (CO): the average number of routing packets exchanged by a node in each time slot; (4) Convergence Time (CT): the times requires to detect a link change; (5) Average delay and Hop counts (DH): the average delay and average hop count required for a packet to travel from source to destination, Table 1 shows the results on comparisons and explains the parameters used for the analysis.

MO: Memory O/H Memory O/H is defined to be the memory space required to store the routing information.

As shown in Table 1, FSR has the same memory O/H, $O(N^2)$ as LS, where N is defined to be the number of nodes in the network. They both maintain the network topology for the whole network and use Dijkstra's algorithm to compute shortest path routes.

HSR has memory O/H of $O(n^2 \cdot L) + O(S) + O(N/S) + O(N/n)$, where S, n, N/n, and L

Protocol	MO	LO	CO	CT
FSR	$O(N^2)$	$O({n_1}^2)+\sum_{i=2}^{D}\{O(n_1 \cdot n_i)/(i-1)I\}$	$O(1)$	$O(D \cdot I)$
HSR	$O(n^2 \cdot L)+O(S)+O(N/S)+O(N/n)$	$\{O(n^2 \cdot L) + O(S) + O(N/n)\}/I+O(1)/J$	$O(n \cdot L)/I+O(1)/J$	$O(D)$
LS	$O(N^2)$	$O(N^2)/I$	$O(N)/I$	$O(D)$
DBF	$O(N)$	$O(N)/I$	$O(1)$	$O(N \cdot I)$
OD	$O(a)$	$O(a)$	$O(a)$	N/A

N: The number of nodes in the network
n: The average number of logical nodes in the cluster
D: The maximum hop distance, the diameter, in the network
L: The maximum number of hierarchical levels
S: The number of virtual IP subnetworks
N/n: The number of physical level clusters
I: The average update interval
J: The registration interval to home agents from each node
a: The number of active paths per node

Table 1: Control overhead analysis

are defined to be the number of virtual IP subnetworks, the average number of logical nodes in the cluster, the number of physical level clusters, and the maximum number of hierarchical levels, respectively. The first element $O(n^2 \cdot L)$ corresponds to the one for each node to have a hierarchical connectivity table with physical and logical nodes along the hierarchical path, where n logical nodes can be seen on all levels up to L. The shortest path must be contructed over this set of nodes. The second element $O(S)$ corresponds to one entry for each of the S home agents, which manage the membership of their virtual IP subnetworks. The entry contains the hierarchical address of the home agents to send a packet at first to the associated home agent. It is used to request the address for the final destination. The third element $O(N/S)$ corresponds to the table in the home agent to store the location of its N/S members. The fourth element $O(N/n)$ is the table that each home agent keep track of logical subnetwork members in its respective clusters with the N/n foreign agents, which are elected one per physical cluster.

The HSR's O/H is usually much lower than LS's $O(N^2)$, as $O(n^2 \cdot L)$ is much smaller than $O(N^2)$ and $O(S)+O(N/S)+O(N/n)$ is also much smaller than $O(N^2)$.

The memory O/H of DBF is $O(N)$, as each node calculates next hop addresses for every possible destinations N, based on distrance vector algorithm, and store these entries.

The memory O/H of OD is $O(a)$, where a is defined to be the number of active paths per nodes, i.e. paths which support the ongoing sessions. This is in contrast to FSR and HSR which are independent of the number of active paths. Since the memory O/H of OD does not depend on the total number of nodes in the network, in case of sparse communication the number of active paths is small, requiring much less memory than FSR and HSR. However, as the number of active paths increase, the memory O/H of OD increases linearly with it, eventually equaling that of DBF, when there is a path to each possible destination $O(N)$.

LO: Line O/H Line O/H is defined to be the aggregate volume of control bytes exchanged by a node in each time slot.

The line O/H of FSR is $O({n_1}^2) + \sum_{i=2}^{D}\{O(n_1 \cdot n_i)/(i-1)I\} = O({n_1}^2) + O(n_1 \cdot n_2)/I + ... + O(n_1 \cdot n_D)/(D-1)I$, where n_1, n_2 and n_D are defined to be the number of nodes for one-hop scope, two-hop scope, and D-hop scope, and D is defined to be the maximum hop distance or diameter in the network, and I is defined to be a routing update interval. Here, N-hop $(1 \leq N \leq D)$ nodes correspond to the nodes which are located in the distance of N hop from my node. FSR distributes N-hop scope topology information in each $(N-1) \cdot I$ time slot, so that frequency of remote host update can be reduced.

Within one-hop scope, each node has n_1 neighbors, and exchanges one-hop neighbor $(O(n_1))$ information with n_1 neighbors in each timeslot. The total amount of control information within one-hop neighbor is given by $O({n_1}^2)$. For the rest of nodes, FSR updates them at a longer update interval and exchanges them with n_1 neighbors, thus the cost of updating the remote hosts is $O(n_1 \cdot n_2)/I + ... + O(n_1 \cdot n_k)/(k-1)I + ...$, where I is the long update interval used for remote hosts update. Since n_1 is a very small value and $O(n_1 \cdot n_k)/(k-1)I$ is also small due to long update interval, the O/H of FSR is smaller than LS's $O(N^2)/I$, and is close to the order of DBF's $O(N)/I$.

HSR has the O/H of $\{O(n^2 \cdot L) + O(S) + O(N/n)\}/I + O(1)/J$, which consists of two components: distribution of link state information and registration of membership to home agents. Regarding to link state distribuion, HSR distributes reachability information for S home agents and N/n foreign agents, and hierarchical node connectivity information $(O(n^2 \cdot L))$ in hierarchical link state manner with average update interval I. I can be calculated based on both of periodical update and event-driven udpate. Regarding to membership registration, each node registers own location to home agents in update interval J, which depends on the mobility, and the control O/H of the worst case scenario would be $(O(1)/J)$. As the moving speed of node gets faster, the update interval J has to be small, and its overhead becomes large.

The O/H of HSR is much smaller than LS's $O(N^2)/I$, as $O(n^2 \cdot L)/I$ is much smaller than $O(N^2)/I$ and $\{O(S) + O(N/n)\}/I + O(1)/J$ is also smaller than $O(N^2)/I$.

On the other hand, the line O/H of OD is also $O(a)$, as the same as the memory O/H. Since the O/H of OD is independent of the total number of nodes in the network, in case of sparse communication the number of active paths is small, it requires much less line O/H than FSR and HSR. However, as the number of active paths increases, the line O/H also increases linearly with it, eventually equaling that of DBF, when there is a path to each possible destination $O(N)$.

CO: Control Packet O/H Control Packet O/H (CO) is defined to be the average number of routing packets exchanged by a node in each time slot.

Similar to the Line O/H, HSR and LS transmit one short packet for each link state update. The control packet O/H for LS can be as high as $O(N)/I$. The O/H of HSR can be as high as $O(n \cdot L)/I + O(1)/J$. $(n \cdot L)$ logical nodes transmit one short packet for each logical neighbor link state update in average interval I, and each physical node also sends a regitration packet to each home agent in interval J, which depends on the mobility. It means that many numbers of short control packets are generated for both LS and HSR.

In case of OD, the control packet O/H can be as high as $O(a)$, and the control packet size is constant and small, due to the fact that the control packet is used for finding a path to a specific destination on demand. This O/H also becomes high when the mobility is high.

On the other hand, FSR and DBF broadcast their routing messages in group $(O(1))$, so fewer, but longer control packets can be used to optimize the MAC throughput.

CT: Convergence time Convergence Time (CT) is defined to be the times requires to detect a link change.

HSR and LS are the best performance of the convergence time $O(D)$ due to the nature that both scheme compute route based on global network information and can detect a breakage on a link at most within the network diameter D by receiving the flooded information of this link, although the Line O/H and and Control packet O/H have to be increased.

FSR is the next good performance convergence time ($O(D \cdot L)$), which is superior than that of DBF's $O(N \cdot I)$. This is because DBF tries to compute an alternate path to bypass the broken link. If the alternate path does not exist, DBF cannot detect it until the hop count for that node is iterated to the value of infinity, which can be as large as N. However, when the loop-free DBF protocol, or Destination-Sequenced Distance Vector (DSDV) is used, the CT is similar to FSR's $O(D \cdot L)$.

The convergence time of OD is not applicable due to the nature of the protocol.

DH: Average delay and Hop counts Average delay is defined to be the end-to-end delay that is obtained by accumulating the delay, which is experienced on each transit node from source to destination to forward a packet. Hop counts is also defined to be the end-to-end hop counts required for a packet to traverse from source to destination.

Since DBF, LS, FSR and HSR are topology distribution protocols which update the topology change periodically (as DBF), or whenever such a change is detected (as LS, FSR and HSR), each node always knows the next hop node toward the destination. Therefore the average delay of DBF, LS and FSR is close to smallest delay for taking a shortest path, and thus the hop counts of them are also smallest value.

HSR instead takes a longer path than DBF, LS, and FSR, due to the fact that each node sends a packet to home agent, which redirects the packet to final destination cluster. The average delay of HSR is a little bit larger delay than DBF, LS and FSR. The hop counts of HSR could be double of those values of DBF, LS and FSR.

On the other hand, in case of OD, it requires two phases: path finding phase and packet forwarding phase. Since initial path finding requires a flooding mechanisms to find a best path, then the average delay of this phase is close to the half of the average diameter of the network. The average delay of the second packet forwarding phase is similar to DBF, LS and FSR, because the obtained path would be close to the shortest path. However, as the mobility is higher, the possiblity that the obtained path may be wrong also increases. Then additional path finding phase are achieved and the delay gets much longer. Therefore, summation of both delays would be more than double of that of DBF, LS and FSR. The average hop counts would be similar to DBF, LS and FSR.

In summary, as seen in the memory O/H analysis, FSR has a similar range of memory O/H as LS, which is higher than DBF, but reduces the line O/H and control packet O/H lower than LS so that channel resources can be used effectively, and has a similar performance of the average delay and hop counts to DBF and LS, while giving a slower convergence time than LS.

HSR has more advantage than FSR. HSR has a similar memory, and line O/H as DBF, which is much less than FSR. It also has the similar control packet O/H and similar convergence time, to LS, which gives more control overhead than FSR, but gives more accurate information due to smaller convergence time. Although it has a similar range of average delay to DBF and LS, the hop counts are larger than DBF, LS, and HSR due to the nature of its protocol.

4 Performance evaluation

4.1 MAC layer model

This section descibes the MAC layer model used in our simulation. We use a cluster infrastructure [24]. Within each cluster, the MAC protocol is selected so as to provide efficient transfer of packets between neighbor nodes. There are several options for MAC protocols, including IEEE 802.11, MACA (multiple Access Collision Avoidance) and FAMA (Floor Acquisition Multiple Access), which have been equipped with an RTS-CTS exchange to share the channel [26]. In our experiments we have selected polling. Namely, the cluster head polls the neighbor nodes to allocate the channel.

Polling was chosen here for several reasons. First, polling is consistent with the IEEE 802.11 standard access scheme (Point Coordination Function). Secondly, polling gives priority to the cluster head, which is desirable since only gateway nodes and cluster heads are involved in our hierarchical routing protocol. Third, polling permits easy support of real time connections (which can be scheduled at periodic intervals by the cluster head). Fourth, in our experiments each cluster has on average six neighbors (which is the optimal value in a uniform multihop architecture); thus polling latency is not a critical concern. For large cluster size the polling scheme can replaced by a polling/random access scheme, to reduce latency.

For the sake of simplicity we also assume that nodes (and in particular gateway nodes) can receive on multiple codes simultaneously (e.g., using multiple receivers). This property does not enhance communications within a cluster, since all wireless nodes are tuned to the same code anyway. It does, however, permit conflict free communications between clusters through gateway nodes.

4.2 Network model

The multihop, mobile wireless network simulator was developed using the parallel simulation language Maisie/PARSEC [45] and exploiting the wireless simulation platform built upon Maisie/PARSEC [43]. The simulator is very detailed. It models all the control message exchanges at the MAC layer (e.g., polling) and the network layer (Hierarchical Routing Protocol control messages). Thus, the simulator enables us to monitor the traffic O/H of the protocols. The network consists of 100 mobile hosts roaming randomly in all directions at a predefined average speed in a 1000x1000 meter square. A reflecting boundary is assumed. Radio transmission range is 120 meters. Free space propagation channel model is assumed . Data rate is 2Mb/s. Packet length is 10 kbit for data, 2 kbit for cluster head neighboring list broadcast, and 500 bits for MAC control packets. Thus, tansmission time is 5ms for data packet, 1 ms for neighboring list and 0.25 ms for control packet. Buffer size at each node is 15 packets.

4.3 Simulation Results

In this section we evaluate and compare the various routing schemes using the PARSEC simulation platform. The performance measures of interest in this study are: (a) control O/H generated by the routing update mechanisms; (b) average delay; (c) average number of hops. The variables are: number of pairs communicating with eachother (this is a good indication of the "sparseness" in the traffic pattern), and; node mobility. Ideally, we would have liked to vary also the number of nodes and explore its impact on performance, especially since we are testing for scalability. Unfortunately, the parallel version of PARSEC for our routing schemes is not yet ready. The sequential version of the routing module has a limit of 100 nodes. Thus, all of our runs are for 100 nodes.

Traffic load corresponds to an interactive environment. Several sessions are established (in most cases, 100 sessions) between different source/destination pairs. Within each session, data packets are generated following a Poisson process with average interval of 2.5 s. This amounts to a traffic volume of 4Kbps per source/destination pair, recalling that data packet length is

10 kbits. In all, this load (even with 500 pairs, which is the maximum we considered in our experiments) can be confortably managed by the network in a static configuration, using any of the routing schemes so far described. With mobility, however, routes may become invalid, causing packets to be dropped and leading to throughput degradation.

The first experiment reports the control O/H caused by routing update messages in the various schemes (see Figure 4.3 and Figure 4.3). In Figure 4.3 we show the O/H as a function of number of communicating pairs, for a node speed of 60Km/hr. Tables are refreshed every 2 sec for DSDV and HSR, and are timed out after 1 sec for On Demand. The O/H is measured in Mbits/cluster. The O/H in DSDV and HSR is constant with number of pairs, as expected, since background updating is independent of user traffic. On Demand O/H, on the other hand, increases almost linearly with the number of pairs, up to 100 pairs (these pairs have distinct destinations). Beyond 100 pairs, destinations are repeated and therefore the same route is reused by multiple sources to reach the same destination. Thus, the O/H increase is less than linear beyond 100 pairs since some path have already been discovered. Recalling that the maximum throughput achievable in a single cluster is 2 Mbps (ignoring MAC layer O/H), we note that both HSR and On Demand have acceptable O/H ($< 10\%$ in the entire range between 10 and 100 nodes). DSDV, on the other hand, is quite "heavy", introducing more than 50% of line overhead! This is because DSDV propagates full routing tables (with 100 entries). HSR uses much smaller tables (10 entries on average), while On Demand propagates only single entry tables whenever needed. It is clear that already at 100 nodes a flat routing scheme such as DSDV is untenable if the network is mobile and therefore requires rapid refresh.

In Figure 4.3 we report the control O/H as a function of node speed. On Demand O/H is constant since the updates are independent of speed. HSR and DSDV both exhibit increasing O/H with speed - update rate must be increased with speed to keep accurate routes. Again, DSDV O/H is prohibitive over the entire range between 20 and 90 Km/hr, while for On Demand and HSR, the penalty is quite reasonable ($< 5\%$).

The next experiment reports average packet delay as a function of mobility. In Figure 4.3 we note that for DSDV and HSR the delay is almost constant (less than 100 ms). Actually, as speed increases, DSDV and HSR progressively lose track of routes and thus drop packets. However, the dropped packets are not accounted for in the delay computation. Moreover,

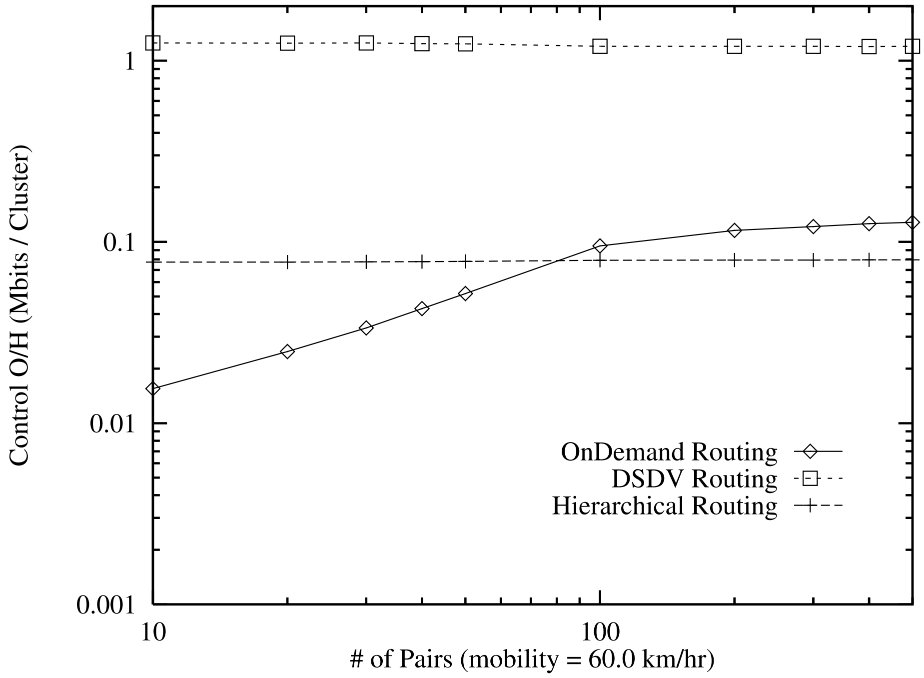

Figure 8: Control O/H vs. The number of traffic pairs

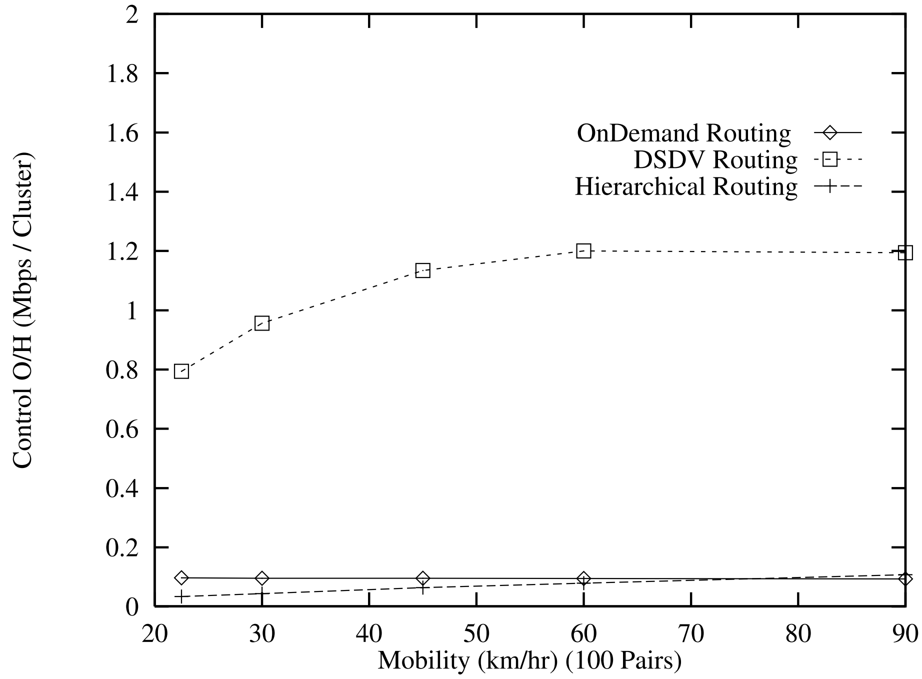

Figure 9: Control O/H vs. Mobility (100 pairs)

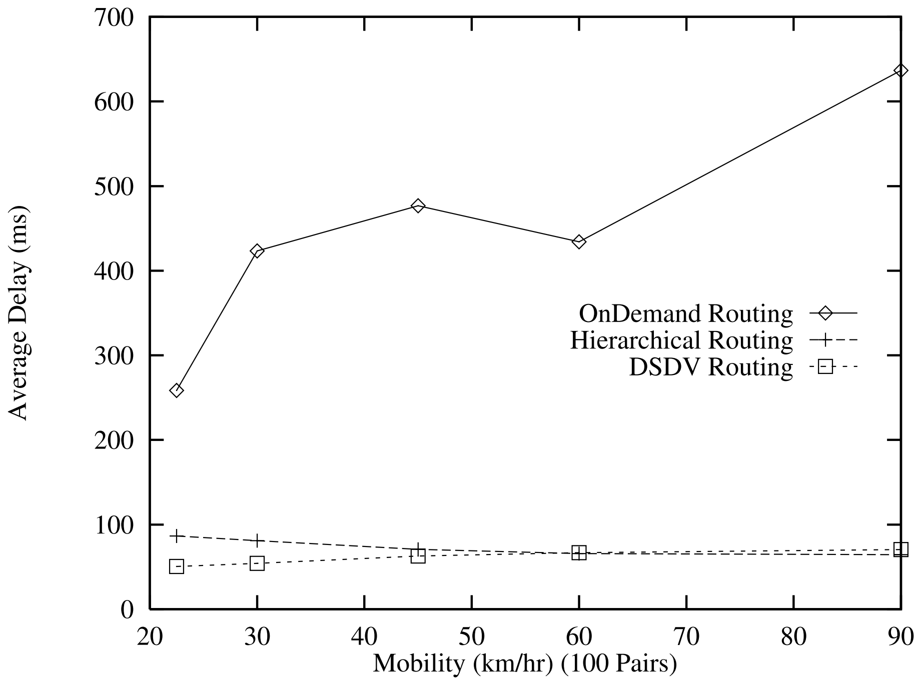

Figure 10: Average delay vs. Mobility (100 pairs)

packets to remote destinations are the most likely do be dropped . This is particularly true for HSR which experiences long paths due to home agent redirection and thus shows an overall decrease of average delay with mobility. For On Demand routing, on the oher hand, packet delay more than doubles (up to 600 ms) as speed increases from 20 to 90 Km/hr. This is due to the fact that if the path to destination is lost (because of mobility, in this case), On Demand routing will not drop the packet, rather it will initiate a search to find a new path at the cost of additional delay.

Average number of hops as a function of mobility is reported in Figure 4.3. We note that the hop length for HSR is almost twice that of DSDV and On Demand. This is due to the fact that the packet if first forwarded to the home agent and than from the home agent to the destination. In most cases, this amounts to doubling the path length. Hop length decreases in HSR as mobility increases because packets tend to be most frequently dropped on the second leg (from home agent to destination) thus leading to shorter overall average for higher mobility. On Demand average hop length slightly increases with speed because of the recomputation of the route at mid point when the original route is invalidated by node movements, leading by

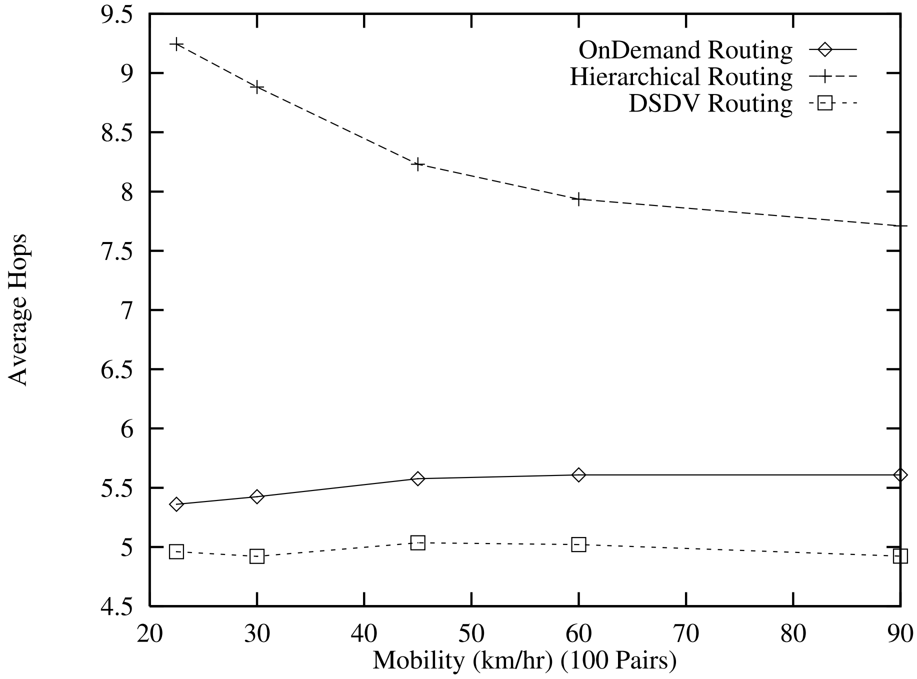

Figure 11: Average hops vs. Mobility (100 pairs)

suboptimal routing. DSDV hop length is constant since it traces the shortest path in all cases (unless the packet is dropped).

Figure 4.3 and Figure 4.3 illustrate the tradeoffs between throughput and control O/H in HSR when the route refresh rate is varied. In Figure 4.3 (at 90 Km/hr) we note that the O/H increases linearly with refresh rate until the network becomes saturated with control packets, and starts dropping them. The throughput first increases rapidely with refresh rate, owing to more accurate routes and lower packet drops due to lack of route. Eventually, throughput peaks and then starts decreasing as the network becomes saturated and data packets are dropped because of buffer overflow. The optimum refresh rate is the rate yielding the max throughput value. Figure 4.3 reports the "optimal" HSR refresh rate as a function of speed.

5 Conclusion

We have introduced two novel routing schemes Fisheye State Routing (FSR) and Hierarchical State Routing (HSR) for large, mobile wireless networks. The schemes are extensions of the conventional table driven routing schemes, but improve scalability by reducing update traffic

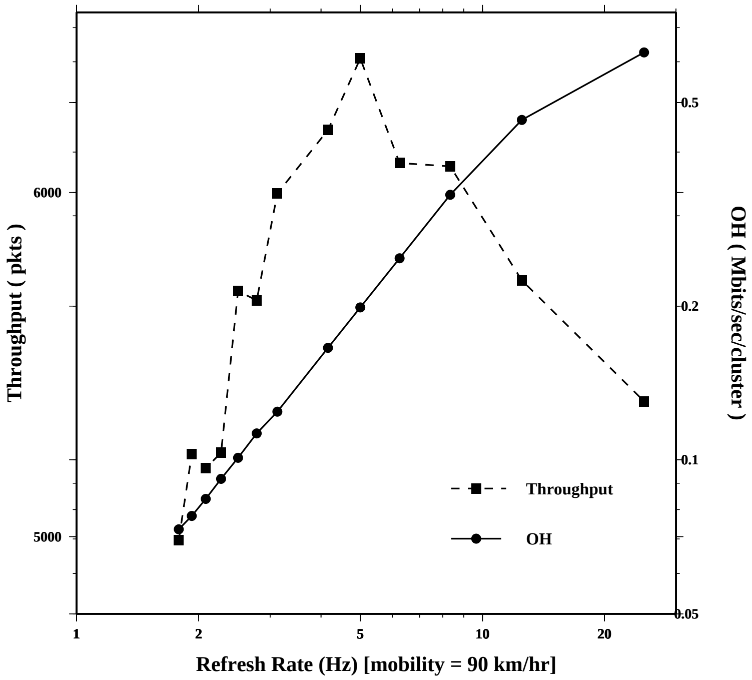

Figure 12: Throughput vs. Refresh rate (90 km/hr) for soft state evaluation

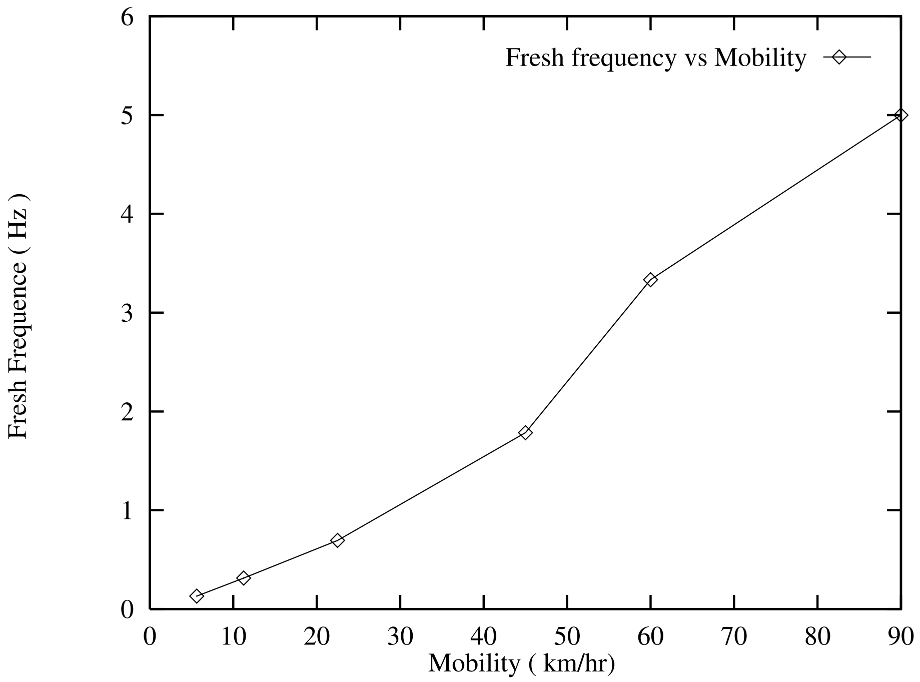

Figure 13: Refresh rate of HSR vs Mobility

O/H. FSR achieves control traffic reduction by selectively adjusting routing update frequencies, while HSR reduces the size of update messages by using a hierarchical addressing approach. FSR maintains a flat addressing scheme and topology map. This makes it easy to locate destinations, but limits scalability because of routing table storage and processing O/H. HSR, in contrast, resolves the routing table scalability problem by using the hierarchical approach. However, it must face the difficult problem of "finding" the destination. We have resolved this problem with a home agent/foreign agent technique which extends the Mobile IP concept to the multihop, mobile environment (no fixed base stations).

When compared with flat, table driven routing schemes (such as DSDV) the proposed solutions exhibit a much better scalability, at the cost of routing inaccuracy and increased complexity (eg, home agent). The scalability advantage is clearly shown by the simulation results.

We have also compared our scalable schemes with recently proposed On Demand routing schemes. Admittedly, HSR and FSR suffer of some disadvantges with respect to On Demand routing, most notably: (a) if a route becomes invalid because of mobility, packets are dropped until the new route is establish via the background routing update process (in contrast, in On Demand, the packet is buffered until the new route is discovered); (b) routing table storage O/H is higher (FSR); (c) protocol complexity is higher (eg., home agent in HSR). On the other hand, HSR and FSR provide the following advantages: (a) lower latency for access to non frequently used destinations; (b) lower control traffic O/H in dense traffic situations (avoiding the flood type search for each destination); (c) QoS advertising prior to connection establishment (this is particularly useful for acceptance control in real time traffic environments)

Via simulation, we have compared HSR, DSDV and On Demand routing. We have explored only a small set of the properties and tradeoffs. Yet, the simulation results clearly indicate the inadequacy of flat, table driven routing as the number of nodes grows. Also, clear is the increase of On Demand control overhead as the number of connections grows (ie, the traffic pattern becomes dense). Higher delays are noticed in On Demand, while higher packet loss rates are observed in HSR, as expected.

The main conclusion that can be drawn from these studies and experiments is that all the above schemes offer different, competitive and complementary advantages and are suited for different applications. A promising direction of future research is the integration of hierarchical, table driven (perhaps fisheye) concepts with on demand routing concepts to generate routing strategies that can perform consistently well across various application domains.

References

[1] C.E. Perkins. "Mobile IP: Design Principles and Practices", Addison Wesley, 1997

[2] J.McQuillian. "Adaptive routing algorithms for distributed computer networks", *BBN Rep.2831*, Bolt Beranek and Newman Inc., Cambridge MA., May 1974

[3] L. Kleinrock and F.Kamoun. "Hierarchical routing for large networks - Performance evaluation and optimization", *Computer Networks*, Vol.1, No. 3, pp.155-174, Jan. 1977

[4] C.V. Ramamoorthy and W.T.Tsai. "An adaptive hierarchical routing algorithm", *IEEE COMPSAC83*, pp.93-104, Nov. 1983

[5] W.T. Tsai, C.V. Ramamoorthy, W.K.Tsai, and O.Nishiguchi. "An adaptive hierarchical routing protocol", *IEEE Transactions on Computers*, Vol.38, No.8, pp.1059-1075, Aug. 1989

[6] C. Zlaettinoglu and Udaya Shankar. "The Viewserver Hiearchy for Interdomain Routing: Protocols and Evaluation" *IEEE Journal on Selected Areas in Communications*, Vol.13, No.8, Oct. 1995

[7] "Private Network-Network Interface Specification Version 1.0 (PNNI 1.0)", *The ATM Forum*, Mar. 1996

[8] J. Moy. "OSPF Version 2", *IETF, RFC 2178*, Jul. 1997

[9] J.J. Garcia-Luna-Aceves and S.Murthy, "A path finding algorithm for loop-free routing", *IEEE/ACM Transactions on Networking*, Vol.5, No.1, PP.148-160, Feb. 1997

[10] S. Murthy and J.J.Garcia-Luna-Aceves, "Loop-free internet routing using hierarchical routing trees", *IEEE Infocom'97*, Apr.1997

[11] J. Bhhrens and J.J. Garcia-Luna-Aceves. "Hierarchical Routing Using Link Vectors" *IEEE Infocom'98*, March 1998

[12] C.E. Perkins and P. Bhagwat. "Highly dynamic destination-sequenced distance-vector routing (DSDV) for mobile computers.", *ACM SIGCOMM'94*, pp. 234–244, 1994.

[13] S. Murthy and J.J. Garcia-Luna-Aceves. "A Routing Protocol for Packet Radio Networks.", *Proc. IEEE Mobicom*, pp. 86–95, November 1995.

[14] S. Murthy and J.J. Garcia-Luna-Aceves. "An Efficient Routing Protocol for Wireless Networks" *ACM Mobile Networks and Applications Journal*, Special Issue on Routing in Mobile Communication Networks, 1996

[15] M. S. Corson and A. Ephremides. "A destributed routing algorithm for mobile wireless networks." *ACM-Baltzer Journal of Wireless Networks*, **1**:61–81, January 1995.

[16] V. D. Park and M. S. Corson. "A Highly Adaptive destributed routing algorithm for mobile wireless networks." *IEEE Infocom*, 1997.

[17] V. Park and S. Corson. "Temporally-Ordered Routing Algorithms (TORA) Version 1 Functional Specification" *IETF, Internet Draft*, draft-ietf-manet-tora-spec-00.txt, Nov. 1997

[18] C. Perkins. "Ad Hoc On Demand Distance Vector (AODV) Routing" *IETF, Internet Draft*, draft-ietf-manet-aodv-00.txt, Nov. 1997

[19] J. Broch, D. Johnson and D. Malts. "The Dynamic Source Routing Protocol for Mobile Ad Hoc Networks" *IETF, Internet Draft*, draft-ietf-manet-dsr-00.txt

[20] Z. J. Hass. "A new routing protocol for the reconfigurable wireless networks.", *IEEE 6th ICUPC*, October 1996.

[21] Z. J. Hass. "The Zone Routing Protocol (ZRP) for Ad Hoc Networks", *IETF, Internet Draft*, draft-zone-routing-protocol-00.txt

[22] M. Bergamo et al. "System Design Specification for Mobile Multimedia Wireless Network (MMWN) (Draft)", *DARPA project DAAB07-95-C-D156*, Oct.30, 1996

[23] K. K. Kasera and R. Ramanathan. "A Location Management Protocol for Hierarchically Organized Multihop Mobile Wireless Networks" *IEEE IEEE 6th International Conference on Universal Person Communications (ICUPC)'97*, vol.1, pp.158-62, Oct. 1997

[24] M. Gerla and J. Tsai. "Multicluster, mobile, multimedia radio network", *ACM-Baltzer Journal of Wireless Networks*, pp.255-265, 1995

[25] C-C. Chiang. "Routing in Clustered Multihop, Mobile Wireless Networks", *The 11th International Conference on Information Networking (ICOIN11)*, 1996

[26] C.R. Lin and M. Gerla "Asynchronous multimedia multihop wireless networks", *IEEE INFOCOM '97*, April 1997

[27] T.-W. Chen, M. Gerla. "Global State Routing: A New Routing Schemes for Ad-hoc Wireless Networks", *IEEE ICC'98*, Jun. 1998, (to be appeared)

[28] L. Kleinrock and K. Stevens. ""Fisheye: A Lenslike Computer Display Transformation"." Technical report, UCLA, Computer Science Department, 1971.

[29] A. Iwata and et al. "ATM routing algorithms with multiple QOS Requirements for Multimedia Internetworking." *IEICE Transactions on Communications*, volume E79-B(8), pp. 999–1007, August 1996.

[30] Ching-Chuan Chiang and Mario Gerla and Lixia Zhang, "Shared Tree Wireless Network Multicast", *IEEE Sixth International Conference on Computer Communications and Networks (IC3N'97)*, pp. 28–33, 1997.

[31] Stephen E. Deering and David R. Cheriton, "Multicast Routing in Datagram Internetworks and Extended LANs", *ACM Transactions on Computer Systems*, Vol. 8, No. 2, pp. 85–111, 1990.

[32] Stephen Deering and Deborah Estrin and Dino Farinacci and Van Jacobson and Ching-Gung Liu and Liming Wei "The PIM Architecture for Wide-Area Multicast Routing", *IEEE/ACM Transactions on Networking*, Vol. 4, No. 2, 1996.

[33] Steven Deering and Deborah Estrin and Dino Farinacci and Van Jacobson and Ahmed Helmy and Liming Wei, "Protocol Independent Multicast Version 2, Dense Mode Specification", *Internet Draft*, 1997.

[34] Tony Ballardie and Paul Francis and Jon Crowcroft, "Core Based Trees (CBT) An Architecture for Scalable Inter-Domain Multicast Routing", *ACM SIGCOMM*, pp. 85–95, 1993.

[35] Kenneth L. Calvert and Ellen W. Zegura, "Core Selection Methods for Multicast Routing" *GIT-CC-95/15*, 1995.

[36] Arup Acharya and Ajay Bakre and B. R. Badrinath, "IP Multicast Extensions for Mobile Internetworking" *IEEE INFOCOM*, 1996.

[37] M. Scott Corson and Stephen G. Batsell, "A Reservation-Based Multicast (RBM) Routing Protocol for Mobile Networks: Initial Route Construction phase" *ACM/Baltzer Journal of Wireless Networks*, Vol. 1, No. 4, 1995.

[38] Elena Pagani and Gian Paolo Rossi, "Reliable Broadcast in Mobile Multihop Packets Networks" *ACM MOBICOM*, 1997.

[39] Ching-Chuan Chiang and Mario Gerla and Lixia Zhang, "Forwarding Group Multicast Protocol (FGMP) for Multihop, Mobile Wireless Networks" *Special Issue of Cluster Computing: the Journal of Networks, Software Tools and Applications*, 1998.

[40] S. Floyd and V. Jacobson and S. McCanne and Ching-Gung Liu and Lixia Zhang, "A reliable multicast framework for light-weight sessions and application level framing", *ACM SIGCOMM*, pp.342–356, 1995.

[41] L. Zhang, S. Deering, D. Estrin, S. Shenker, and D. Zappala. "RSVP: A new Resource ReSerVation Protocol." *IEEE Network*, pp. 8–18, September 1993.

[42] C.-K. Toh, "Associativity Based Routing For Ad Hoc Mobile Networks," *Wireless Personal Communications Journal*, Special Issue on Mobile Networking and Computing Systems, Vol. 4, No. 2, pp.103-139, March 1997.

[43] R. Bagrodia, X. Zeng and M. Gerla, "GloMoSim: a Library for the Parallel Simulation of Large-scale Wireless Networks", *Proc. of PADS'98*.

[44] R. Sedgewick. *Weighted Graphs*, chapter 31. Addision-Wesley, 1983.

[45] R. L. Bagrodia and Wen-Toh Liao, "Maisie: A Language for the Design of Efficient Discrete-Event Simulations" *IEEE Transactions on Software Engineering*, Vol.20, pp.225-38, April 1994.