Adaptive Clustering for Mobile Wireless Networks†

Chunhung Richard Lin and *Mario Gerla*

Abstract

This paper describes a self-organizing, multihop, mobile radio network, which relies on a code division access scheme for multimedia support. In the proposed network architecture, nodes are organized into nonoverlapping *clusters*. The clusters are independently controlled and are dynamically reconfigured as nodes move. This network architecture has three main advantages. First, it provides spatial reuse of the bandwidth due to node clustering. Secondly, bandwidth can be shared or reserved in a controlled fashion in each cluster. Finally, the cluster algorithm is robust in the face of topological changes caused by node motion, node failure and node insertion/removal. Simulation shows that this architecture provides an efficient, stable infrastructure for the integration of different types of traffic in a dynamic radio network.

1. INTRODUCTION

Personal communications and mobile computing require a wireless network infrastructure which is fast deployable, possibly multihop, and capable of multimedia service support. The first infrastructure of this type was the Packet Radio Network (PRNET), developed in the 70's to address the battlefield and disaster recovery communication requirements [16, 17]. PRNET was totally asynchronous and was based on a completely distributed architecture. It handled datagram traffic reasonably well, but did not offer efficient multimedia support. Recently, under the WAMIS (Wireless Adaptive Mobile Information Systems) [1] and Glomo ARPA programs several mobile, multimedia, multihop (M^3) wireless network architectures have been developed, which require some form of synchronous, time division infrastructure. The synchronous time frame leads to efficient multimedia support implementations. However, it introduces more complexity and is less robust in the face of mobility and channel fading. Clearly there are complexity vs performance tradeoffs in introducing various degrees of synchronization into the network, from the completely asynchronous PRNET to the tightly synchronized cluster TDMA solution proposed in [7]. In this paper, we will evaluate these tradeoffs and will in fact propose a scheme with an intermediate degree of synchronization.

Another important wireless network feature addressed in this paper is multihopping, i.e. the ability of the radios to relay packets from one to another without the use of base stations. Most of the nomadic computing applications today are based on a single hop radio connection to

† This work was in part supported by the National Science Council, Taiwan, R.O.C., under Contract NSC-86-2213-E-194-024-T "QoS Support for Wireless, Mobile, Multimedia Networks", and in part by the U.S. Department of Justice/Federal Bureau of Investigation, ARPA/CSTO under Contract J-FBI-93-112 "Computer Aided Design of High Performance Wireless Networked Systems".

the wired network (Internet or ATM). Figure 1 shows the cellular model commonly used in the wireless networks. A, B, C, and D are fixed base stations connected by a wired backbone. Nodes 1 through 8 are mobile nodes. A mobile node is only one hop away from a base station. Communications between two mobile nodes must be through fixed base stations and the wired backbone.

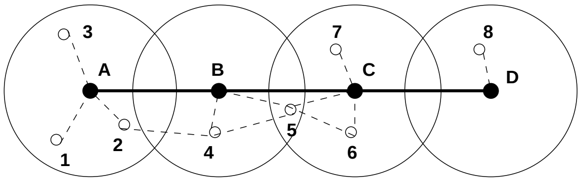


Figure 1: Conventional cellular networks (single-hop)

In parallel with (and separately from) the single hop cellular model, another type of model, based on radio to radio packet multihopping, has been emerging to serve a growing number of applications which rely on a fast deployable, wireless infrastructure. The classic examples are battlefield communications and (in the civilian sector) disaster recovery (fire, earthquake) and search and rescue. A recent addition to this set is the ‘‘ad hoc’’ personal communications network, which could be rapidly deployed on a campus, for example, to support collaborative computing and access to the Internet during special events (concerts, festivals etc). Multihopping through wireless repeaters strategically located on campus permits to reduce battery power and to increase network capacity. More precisely, by carefully limiting the power of radios, we conserve battery power. Furthermore, we also cause less interference to other transmissions further away; this gives the additional benefit of ‘‘spatial reuse’’ of channel spectrum, thus increasing the capacity of the system.

Interestingly, the multihop requirement may also arise in cellular networks. If a base station fails, a mobile node may not be able to access the wired network in a single hop. For example, in Figure 2, if base station B fails, node 4 must access base stations A or C through node 2 or node 5 which act as wireless multihop repeaters.

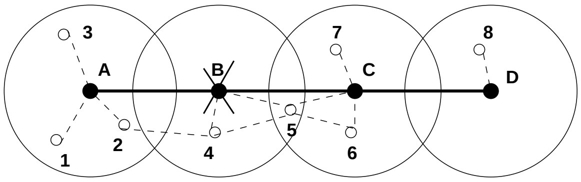


Figure 2: A multihop situation occurs when base station B fails.

In this paper, we consider a networking environment in which the users are mobile, the topology changes, interference occurs when multiple transmissions take place over (possibly different) links on the same or different codes, real-time multimedia traffic must be supported as well as datagram traffic, there is no stable communication infrastructure, and there is no central control. The kind of application scenarios that motivate this research include many that require instant infrastructure network support and multimedia network support. These include military applications (special operations, battlefield scenarios, etc.), disaster relief (fire, earthquake, flood), law enforcement situations, short term scenarios such as public events, etc.

For the above environment and scenarios, we develop an architecture and networking algorithms which support a rapidly deployable radio communications infrastructure. The network provides guaranteed Quality of Service (QoS) to real-time multimedia traffic among mobile users without requiring a fixed infrastructure (e.g., no base station). The last comment is worth emphasizing since much of the research in wireless communications has exploited the existence of "central" control of base stations. We deal with no such central system support in this research.

The paper is organized as follows. Section 2 presents the network architecture. Based on this architecture, section 3 introduces the protocol for packet transmission. Section 4 describes the QoS routing. Section 5 shows some system performance. Section 6 concludes the paper.

2. THE MULTICLUSTER ARCHITECTURE

A major challenge in multihop, multimedia networks is the ability to account for resources so that bandwidth reservations (in a deterministic or statistical sense) can be placed on them. We note that in cellular (single hop) networks such accountability is made easy by the fact that all stations learn of each other's requirements, either directly, or through a control station (e.g. base station in cellular systems). This solution can be extended to multihop networks by creating clusters of radios, in such a way that access can be controlled and bandwidth can be allocated in each cluster. The notion of cluster has been used also in earlier Packet Radio nets, but mainly for hierarchical routing rather than for resource allocation [3, 4].

Most hierarchical clustering architectures for mobile radio networks are based on the concept of *clusterhead* [3, 4, 7]. The clusterhead acts as a local coordinator of transmissions within the cluster. It differs from the base station concept in current cellular systems, in that it does not have special hardware and in fact is dynamically selected among the set of stations. However, it does extra work with respect to ordinary stations, and therefore it may become the bottleneck of the cluster. To overcome these difficulties, in our approach we eliminate the requirement for a clusterhead altogether and adopt a fully distributed approach for cluster formation and intra-cluster communications [13, 14].

The objective of the proposed clustering algorithm is to find an interconnected set of clusters covering the entire node population. Namely, the system topology is divided into small partitions (*clusters*) with independent control. A good clustering scheme will tend to preserve its structure when a few nodes are moving and the topology is slowly changing. Otherwise, high processing and communications overheads will be paid to reconstruct clusters. Within a cluster, it should be easy to schedule packet transmissions and to allocate the bandwidth to real time traffic. Across clusters, the spatial reuse of codes must be exploited. Since there is no notion of clusterhead, each node within a cluster is treated equally. This permits us to avoid vulnerable centers and hot spots of packet traffic flow.

2.1. The Clustering Algorithm

In order to support multimedia traffic, the wireless network layer must guarantee QoS (bandwidth and delay) to real time traffic components. Our approach to provide QoS to multimedia consists of the following two steps: (a) partitioning of the multihop network into clusters, so that controlled, accountable bandwidth sharing can be accomplished in each cluster; (b) establishment of Virtual Circuits with QoS guarantee. In this section we describe the implementation of both steps in the multicluster architecture.

The objective of the clustering algorithm is to partition the network into several clusters. Optimal cluster size is dictated by the tradeoff between spatial reuse of the channel (which drives toward small sizes), and delay minimization (which drives towards large sizes). Other constraints also apply, such as power consumption and geographical layout. Cluster size is controlled through the radio transmission power. For the cluster algorithm, we have so far assumed that transmission power is fixed and is uniform across the network.

Within each cluster, nodes can communicate with each other in at most two hops. The clusters can be constructed based on node ID. The following algorithm partitions the multihop network into some nonoverlapping clusters. We make the following operational assumptions underlying the construction of the algorithm in a radio network. These assumptions are common to most radio data link protocols [3, 4, 6, 7].

A1: Every node has a unique ID and knows the IDs of its 1-hop neighbors. This can be provided by a physical layer for mutual location and identification of radio nodes.

A2: A message sent by a node is received correctly within a finite time by all its 1-hop neighbors.

A3: Network topology does not change during the algorithm execution.

We can find from this algorithm (Figure 3) that each node only broadcasts one *cluster* message before the algorithm stops, and the time complexity is $O(|V|)$ where V is the set of nodes. The clustering algorithm converges very rapidly. In the worst case, the convergence is linear in the total number of nodes. Consider the topology in Figure 4. After clustering, in Figure 5, we can find six clusters in the system, which are {1,2}, {3,4,11}, {5,6,7,8,9}, {10,12,13}, {14,15,16,17}, {18,19,20}. To prove the correctness of the algorithm we have to show that: 1) every node eventually determines its cluster; 2) in a cluster, any two nodes are at most two hops away; 3) the algorithm terminates.

Lemma 1: Every node can determine its cluster and only one cluster.
Proof: The cluster ID of each node is either equal to its node ID or the lowest cluster ID of its neighbors. Every node has to decide its cluster ID once it becomes the lowest ID node in its locality. Hence, every node can determine its cluster and only one cluster.

Lemma 2: In a cluster, any two nodes are two hops away at most.
Proof: Consider nodes in the same cluster. Every node can reach the node which node ID is

Distributed Clustering Algorithm(Γ)

```
Γ: the set of ID's of my one-hop neighbors and myself
{
    if (my_id == min(Γ))
    {
        my_cid = my_id;
        broadcast cluster(my_id,my_cid);
        Γ = Γ − {my_id};
    }

    for (;;)
    {
        on receiving cluster(id, cid)
        {
            set the cluster ID of node id to cid;
            if (id==cid and (my_cid==UNKNOWN or my_cid>cid))
                my_cid = cid;
            Γ = Γ − {id};
            if (my_id == min(Γ))
            {
                if (my_cid==UNKNOWN) my_cid = my_id;
                broadcast cluster(my_id,my_cid);
                Γ = Γ − {my_id};
            }
        }
        if (Γ==∅) stop;
    }
}
```

Figure 3: Distributed clustering algorithm

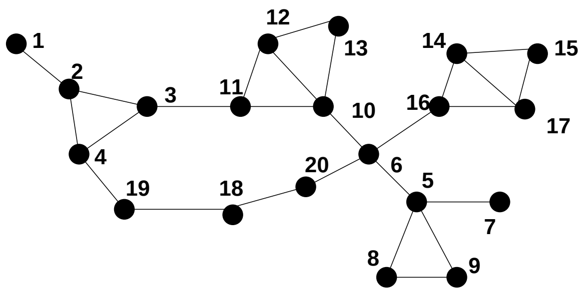


Figure 4: System topology

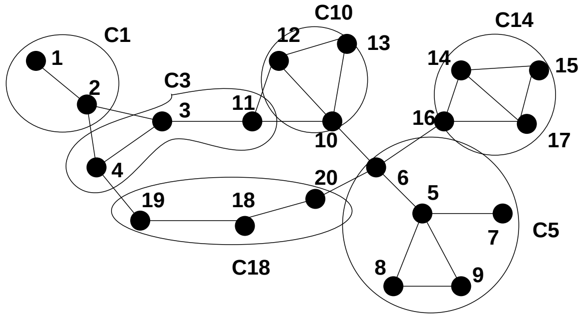


Figure 5: Clustering

equal to cluster ID in one hop. Thus, any two nodes are two hops away at most.

Theorem 1: Eventually the algorithm terminates.
Proof: Since every node can determine its cluster (*Lemma 1*), the set Γ will eventually become empty. Thus, the algorithm will terminate.

Theorem 2: Each node transmits only one message during the algorithm.
Proof: A node broadcasts messages only at the time it decides its cluster ID. Thus, only one message is sent out before the algorithm stops.

Theorem 3: The time complexity of the algorithm is $O(|V|)$.
Proof: From the distributed clustering algorithm, each message is processed by a fixed number of computation steps. From *Theorem 2*, there are only $|V|$ messages in the system. Thus, the time complexity is $O(|V|)$.

As we shall see in the following experiments, transmission power determines network topology and therefore has a direct impact on the performance of the cluster solution. If the power is large, average hop length between source-destination (SD) pairs is small. Thus, the system throughput will tend to increase. However, higher interference due to large power will tend to limit the throughput. If we decrease the power, the interference is also decreased. However, since average hop length between SD pairs is increased, the network will accumulate more packets among the store-and-forward paths. Congestion control becomes more critical in this situation. Clearly, it is important to choose a suitable power level in order to get high system performance.

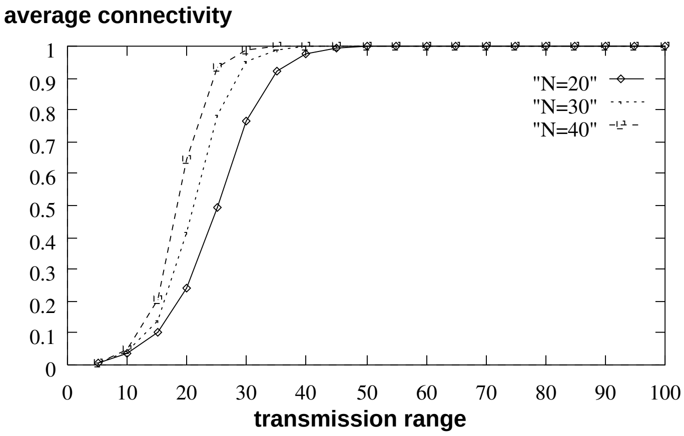


Figure 6: Connectivity property

We simulate the clustering algorithm by placing N nodes randomly in a 100×100 area and we measure some of its properties. We assume two nodes can hear each other if their distance is within a predefined transmission range. We begin studying the impact of transmission range on *connectivity*. The connectivity is defined as the fraction of node pairs which can communicate through single or multiple hops. We assume an ideal network model where a link can be established between any two nodes within transmission range of each other; and where a path can always be found (by the routing algorithm) between two nodes connected by a chain of links. Several random deployments of the N nodes are examined for each transmission range. Each deployment yields a different value of connectivity. In Figure 6, we report the average connectivity. From Figure 6, we note that in order to guarantee that all nodes can communicate with each other, the transmission range should be more than 30 for $N = 40$, and more than 40 for $N = 20$.

Next, we study the characteristics of the clusters generated by the algorithm. In the multi-cluster architecture, *repeaters*, for example node 2 in Figure 5, relay packets from one cluster to another. Every repeater is time-shared among the set of adjacent clusters, that is, its spreading code must be transmit to these clusters. So, the *order* of a repeater (i.e. how many clusters it can access, for example the order of node 2 is 2 and the order of node 6 is 4) should be small in order to maintain efficient operation, with as few code changes as possible (the minimal order of a repeater is 2). Figure 7 shows the average order of repeaters versus the transmission range. From this figure, we find that the typical order of the repeaters is either 2 or 3 based on this clustering algorithm.

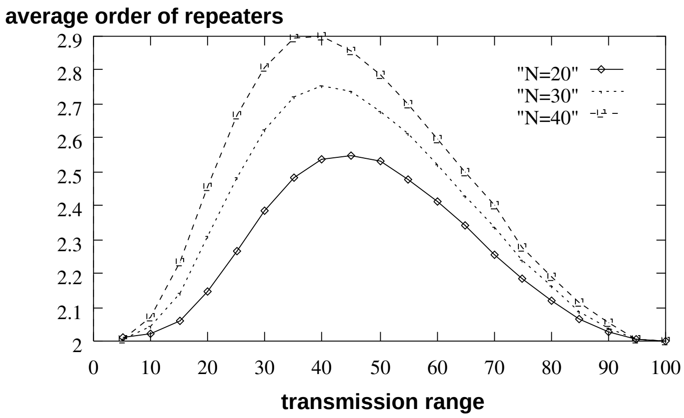


Figure 7: Average order of repeaters

Since the topology in WAMIS is dynamically changed, the reliability of packet routing is important to guarantee the integrity of network services. Thus, the existence of at least one path between a pair of nodes is required. The number of repeaters will affect the number of paths. Namely, the larger the fraction of nodes which are repeaters, the larger the number of alternate paths. In Figure 8, we note that more than 50% of nodes are repeaters if the transmission range is over the interval (30, 80).

2.2. Cluster Maintenance in the Presence of Mobility

In the *dynamic radio network*, 1) nodes can change location; 2) nodes can be removed; and 3) nodes can be added. A topological change occurs when a node disconnects and connects from/to all or part of its neighbors, thus, altering the cluster structure. System performance is affected by frequent cluster changes. Therefore, it is important to design a cluster maintenance scheme to keep the cluster infrastructure as stable as possible. In this respect, the proposed cluster algorithm is more robust than one reported in [7], since there are fewer restrictions on clusters. The cluster maintenance scheme was designed to minimize the number of node transitions from one cluster to another.

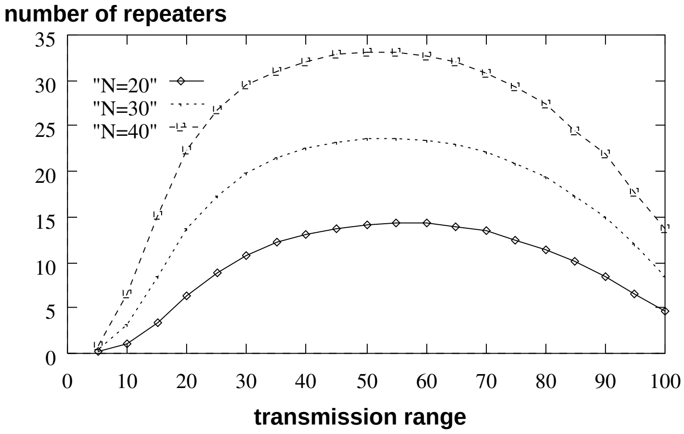


Figure 8: Number of repeaters

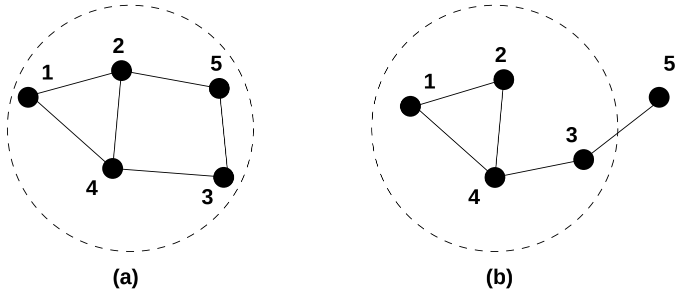


Figure 9: Re-clustering

Consider the example shown in Figure 9(a). There are 5 nodes in the cluster and the hop distance is no more than 2. Because of mobility, the topology changes to the configuration shown in Figure 9(b). At this time, $d(1,5) = d(2,5) = 3 > 2$, where $d(i, j)$ is the hop distance between node i and j. So the cluster needs to be reconfigured. Namely, we should decide which node(s) should be removed from the current cluster. We let the highest connectivity node and its neighbors to stay in the original cluster, and remove the other nodes. Recall that each node only keeps the information of its ‘‘locality’’, that is, one and two hop neighbors. Upon discovering that a member, say x, of its cluster is no longer in its locality, node y should check if the highest connectivity node is a one hop neighbor. If so y removes x from its cluster. Otherwise, y changes cluster.

Two steps are required to maintain the cluster architecture:

Step 1: Check if there is any member of my cluster has moved out of my locality.

Step 2: If Step 1 is successful, decide whether I should change cluster or remove the nodes not in my locality from my cluster.

Consider the example in Figure 9(b). Node 4 is the highest connectivity node. Thus, node 4 and its neighbors {1, 2, 3} do not change cluster. However, node 5 should either join another cluster or form a new cluster. If a node intends to join a cluster, it has to check first if all of members of this cluster are in its locality. Only in this case can it join the cluster.

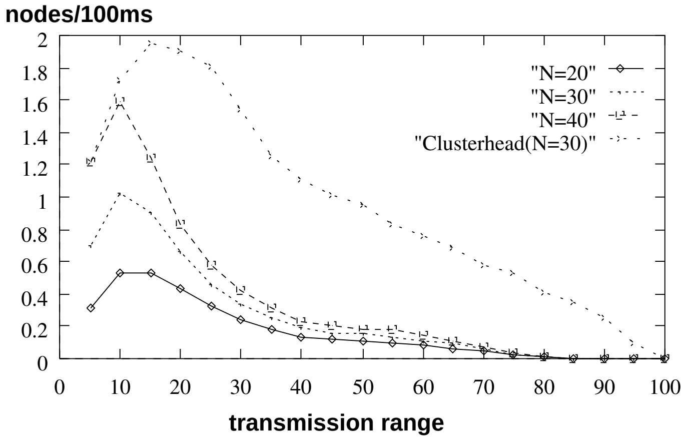


Figure 10: Stability of the multicluster architecture.

We measure the stability of the multicluster architecture by counting how many nodes immigrate from one cluster to another (or form a new cluster) within a 100 ms interval. Figure 10 shows the stability of the cluster maintenance algorithm. In our simulation, every 100 ms each nodes moves in a direction uniformly distributed over the interval (0, 2π), covering a distance of (0, 3) feet. From Figure 10, note that the average number of nodes which change clusters per 100 ms is relatively small over the (40, 50) transmission range (which is the region of interest). Note also that our cluster maintenance scheme based on node connectivity is more stable than the ‘‘clusterhead’’ scheme reported in [7].

2.3. Code Assignment

Each node has a transceiver which can either transmit or receive at any given time. In the spread-spectrum code-division system, the receiver should be set to the same code as the designated transmitter. For simplicity (and to be conservative) we assume no capture. That is, if two or more transmissions interfere at the same receiver, none is received, regardless of the code. We assume that there is a small set of ‘‘good’’ spread-spectrum codes which are low cross-correlation. Since the number of codes we can use is very limited, spatial reuse of codes will be important [11]. Thus each cluster is assigned a single code which is different from the codes using in the neighbor cluster. The problem of the code selection can be formulated as a graph coloring problem, and has been extensively studied in [11].

There are 3 options for using the dedicated code within a cluster:

(1) Receiver-based code assignment: every node within a cluster is assigned a common receiving code. All neighbor nodes send packets to a node using its code. In this scheme, a receiver only listens to one code, but both inter-cluster and intra-cluster collisions can occur.

(2) Transmitter-based code assignment: within a cluster, every node uses a common transmitting code so that there is no inter-cluster collision. If no two nodes in a cluster are transmitting simultaneously, there will be no intra-cluster collision.

(3) Another approach is to assign a common codes to all transmitter-receiver pairs within a cluster. This code assignment requires that some other codes be assigned for inter-cluster communications.

Following [11], we use transmitter-based code assignment. Receiver-based assignment cannot avoid inter-cluster collision and internal pair code assignment needs extra codes for inter-cluster communications. Since there is no inter-cluster collision with the chosen scheme, we only need to be concerned with collision avoidance within a cluster. Based on transmitter-based code assignment, when a node is not in transmitting mode, it randomly selects and listens to one of the codes used by its neighbors. In Figure 5, for example, node 6 will randomly listen to the codes in C_{10}, C_{14}, C_{18} and C_5.

2.4. Network Initialization

Initialization is carried out using a common ‘‘control’’ code in the same way as described in [7]. A node which does not yet belong to a cluster listens to the control code until timeout. Then, it transmits its own ID (using the control code) and repeats the procedure until it hears from one of the neighbors. Channel access in this phase is CSMA. This basic communications facility allows nodes to organize themselves in clusters following the algorithm just described. Once a cluster is formed, the cluster leader communicates with the neighbors (using the control code) to select the codes. Only when the code assignment is completed (i.e. each cluster has been assigned its code) can user data be accepted by the nodes and transmitted in the network.

3. TRANSPORT PROTOCOLS

In this section, we introduce the MAC, link and network layer protocols (with the exception of routing which is discussed in Sect. 5). The aim of our design is to support integrated traffic (i.e. datagram and real time) efficiently. We will assume fixed packet size through the paper.

3.1. Channel Access Scheme

Since our system assumes a common transmitting code in each cluster, there is no inter-cluster collision. The receiver must tune to the transmitter's code to receive the packet. Since the distance between any two nodes in the same cluster is at most two hops, it is relatively easy to maintain time (i.e. slot) synchronization within each cluster. So, the channel will be assumed slot synchronized. It must be noted that synchronization is required only within a cluster. This is much easier than maintaining slot synchronization across the entire network as in [7,14].

Within each cluster, the medium access control (MAC) layer is implemented using a TDMA scheme. Time is divided into slots which are grouped into frames. In Figure 11, assume that there are n nodes in a cluster. Each node is assigned a slot to transmit either control (such as ACKs and connectivity) or data information. In each frame, a free slot is reserved for a new node joining the cluster. Using the control code, the nodes in the cluster take turns to transmit periodically in the free slot, their cluster and code information for the purpose of ‘‘attracting’’ new

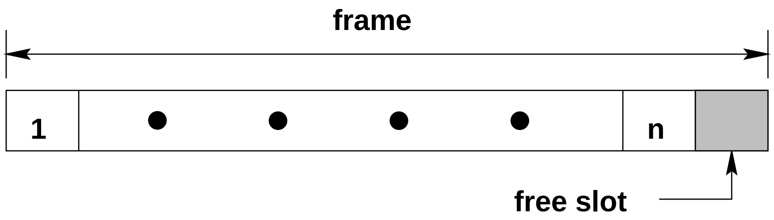


Figure 11: Channel access frame within a cluster

nodes or migrant nodes. When a node decides to join a cluster, the node listens to the channel for a period of time, and then uses this free slot to transmit packets temporarily. Since cluster switches are infrequently, one free slot will suffice. The frame is readjusted after each join/leave.

3.2. Acknowledgment for Datagram

Datagram traffic is error-sensitive. Thus it is important to design a reliable transmission for datagrams. In addition to forward error correction (FEC) at the link level, and end-to-end (TCP) acknowledgments, we have a link level acknowledgment scheme, which is conflict-free. As mentioned earlier, each cluster has a dedicated code for transmission. Since every node can only transmit packets in its assigned TDMA slots, we use an implicit acknowledgment scheme. Namely, upon receiving a packet successfully, the *intended receiver*, the node to which a packet is destined, piggybacks the ACK on its data packet at its assigned slot. The transmitter listens to the receiver's slot and code. If a time-out occurs, it retransmits the data packet. Figure 12 illustrates this implicit ACK scheme. Node x uses code a to transmit its packet to y, and then listens to code b for ACK. Node y receives the packet successfully. When its transmitting slot comes, y piggybacks an ACK for x on the packet which is transmitted to z.

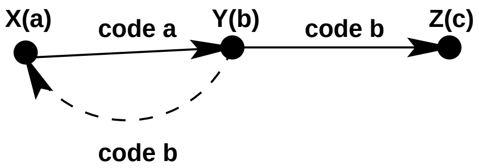


Figure 12: Implicit acknowledgment scheme

3.3. Bandwidth Reservation for Virtual-Circuit Traffic

A real-time connection is set up using a fast reservation approach. Namely, we assume that real-time packets arrive at constant time intervals. The first data packet in the multimedia stream makes the reservations along the path. Once the first data packet is accepted on a link, a transmission window is reserved (on that link) at appropriate time intervals for all the subsequent packets in the connection. The window is released when idle for a prespecified number of cycles. Conceptually, this scheme is an extension of PRMA (Packet Reservation Multiple Access) [8] to the multihop environment.

Each real-time connection is assigned to a VC (Virtual Circuit). The VC is an end-to-end path along which slots have been reserved. As we shall later see, the path and slots of a VC may change dynamically during the lifetime of a connection due to mobility. Each node schedules each of its slots to transmit either datagram or VC traffic. Since real time traffic (which is carried on a VC) needs guaranteed bandwidth during its active period, each node has to reserve its own slots to the VC at connection setup time.

When a node intends to setup a VC to its neighbor, it transmits the first packet of the session as a datagram packet in its TDMA slot. After successfully receiving the packet, the intended receiver will setup the reservation for receiving the next packet since the next transmission time is piggybacked on the current packet. Since the sender always uses the same code to transmit packets, the intended receiver only needs to lock on that code when the reserved slot comes. If the link is not broken due to mobility, the subsequent packets will be received successfully (assuming perfect channel). No ACK is necessary.

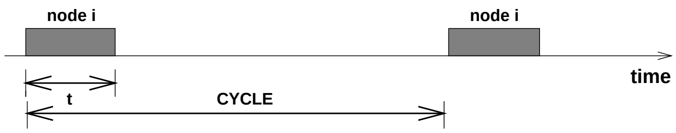


Figure 13: Bandwidth reservation

Let *CYCLE* be the maximum interval tolerated between two real-time packets. The first packet of a real-time session is treated as a data packet and is transmitted using TDMA. It has higher priority than data packets in the local queue. A real-time source schedules its next transmission after a time *CYCLE* following a successful transmission, and piggybacks the reservation with the current packet transmission. Figure 13 shows that node i successfully transmits the first real-time packet, and it reserves the time slot for the real-time session. The receiver has to listen to the sender's transmitting code when the reserved time slot comes. So, for real-time sources, transmission is always collision-free and the maximal delay is guaranteed. At the end of the real-time session (i.e. the reservation field is set to zero), the reservation is automatically canceled.

Because of the limitation of node bandwidth in a cluster, the number of real-time sessions which can pass through a node is restricted. Slots which are not reserved by voice traffic are accessed according to a TDMA protocol. Datagram packets become backlogged when real-time traffic starts building up. Consider the case when the bandwidth of a node is completely used by real-time session. That is, there are $\left\lfloor \frac{CYCLE}{(n+1) \cdot t} \right\rfloor$ real-time sessions (see Figure 13) over a node, where n is the number of members in the cluster. No other source can construct a VC which passes through the saturated node, until one of the VCs over the saturated node ends its transmission, and bandwidth becomes available. To avoid datagram traffic lockout, a limit is imposed on the maximum number of real-time sessions on each node. Also, it is assumed that CYCLE time is much longer than frame time, so that a node can transmit real time and datagram packets in alternate frames.

3.4. Mobility

As nodes move from one cluster to another, it is easy for a node to join a cluster. First, the node does not destroy the target cluster structure. Rather, it changes its own cluster ID, using the free slot to transmit packets in the new code. After a predefined time period (for example, 100 ms in our system), the nodes in a cluster will recompute the new TDMA frame format. In the same way, if a node is removed from a cluster, the frame is reduced. In the rare event that more than one node joins a cluster, collision may occur at the free slot. Using random retransmission and backoff, eventually all new nodes can make themselves known, and a new frame format is computed accordingly.

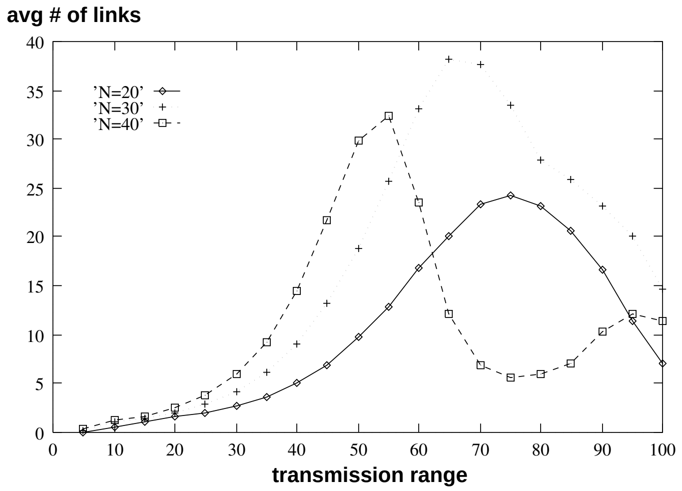


Figure 14: The average number of links between an adjacent cluster pair.

Typically, multiple links exist between adjacent clusters (see Figure 14). Therefore, as a node moves, the connectivity between clusters will not change very rapidly. This characteristic is important for the rerouting of VCs following node movements. When a new link has to be built for an existing real-time session due to topological change, the reservation procedure discussed in the previous section must be re-executed on the new route. More precisely, following the "fast VC reservation" concept introduced in [7], the first packet to be rerouted will trace the new path and make reservations on it. If bandwidth is not adequate on the new path, the connection is dropped. Alternatively, if layered coding is used, low priority packets can be dropped, as described in [7], thus reducing the required bandwidth, and making the new path feasible [7].

4. QoS ROUTING

Multimedia applications such as digital audio and video have much more stringent QoS requirements than traditional datagram applications. For a network to deliver QoS guarantees, it must reserve and control resources. Routing is the first step in resource reservation. The routing protocol first finds a path with sufficient resources. Then, the resource setup protocol makes the reservations along the path.

4.1. Bandwidth in the Cluster Infrastructure

The key resource for multimedia QoS support is bandwidth. Thus, we first must define bandwidth in our cluster infrastructure. Recall that a real-time packet is transmitted every cycle time. For the purpose of real time connection support, we can define “bandwidth” as the number of real-time connections that can pass through that node. Since in our scheme a node can at most transmit one packet per frame, the bandwidth of a node is given by:

$$Bandwidth = \left\lfloor \frac{cycle\ time}{frame\ time} \right\rfloor$$

The frame time of a cluster depends on how many nodes there are in the cluster, as we mentioned in chapter 3. Figure 15 shows the slots dedicated to node i in the cycle, which correspond to node i “bandwidth”. For a numerical example, consider Figure 16, where the cycle time is 24. Consider cluster C_1, where frame size is equal to 6 slots. Thus, the node bandwidth in C_1 is 24/6 = 4. Since there are 3 VCs passing through node C, the available bandwidth for node C is 1.

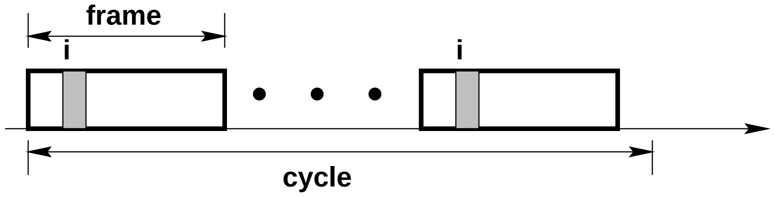
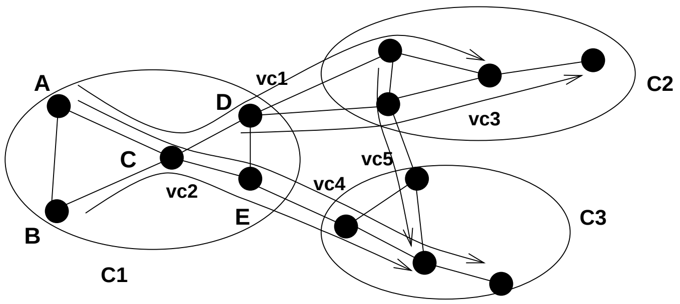


Figure 15: Node bandwidth



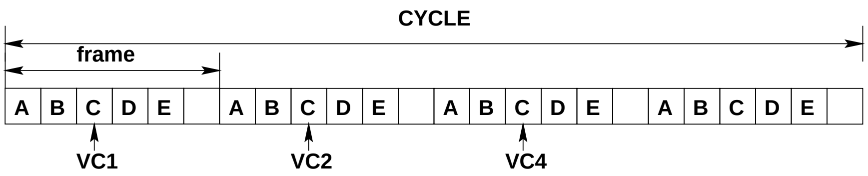


Figure 16: Bandwidth of node C in cluster C_1

4.2. QoS Routing Scheme

The goal of the bandwidth routing algorithm is to find the shortest path such that the free bandwidth is above the minimum requirement.

To compute the "bandwidth" constrained shortest path, we use the DSDV (Destination Sequenced Distance Vector) routing algorithm [15] which was proven to be loop-free. Loop freedom follows from the fact that the updates generated by a destination are sequentially numbered. In our shortest path computation, the weight of each link is equal to 1 (i.e. minimal hop distance routing). The bandwidth constraint is simply accounted for by setting to infinity the weights of all the links to/from a node with zero bandwidth. An advantage of this scheme is to distribute real time traffic evenly across the network. A cluster with small frame size will allow more connections to pass through it, since it has more "bandwidth" per node.

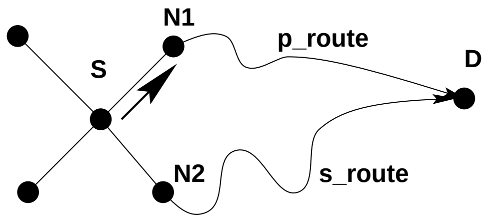


Figure 17: Standby routing

In addition to load balancing, our routing scheme also support the alternative paths. This is very important in a mobile environment, where links will fail because of mobility. In such an environment, routing optimality is of secondary importance. The routing protocol must be capable of finding new routes quickly when a topological change destroys existing routes. To this end, we propose to maintain secondary paths which can be used immediately where the primary path fails. In Figure 17, each node uses the primary route to route its packets. When the first link on the path (s, N1) fails, the secondary path (s,N2) becomes the primary path, and another standby path (s,N3) will be computed, as shown in Figure 18. It is worth emphasizing that these routes must use different immediate successors to avoid failing simultaneously.

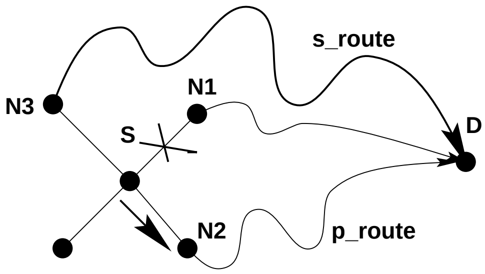


Figure 18: The primary route fails and the standby route becomes the primary route. Another standby route is constructed.

The secondary (standby) route is easily computed using the DSDV algorithm. Referring to Figure 17, each neighbor of node *S* periodically informs *S* of its distance to destination *D*. The neighbor with shortest distance yields the primary route. The runner up yields the secondary route. This scheme guarantees that the first link is difficult for the two paths. Furthermore, the standby route computation requires no extra table, message exchange or computation overhead.

Also, the standby route is loop free as the primary route is.

5. SYSTEM PERFORMANCE

The multicluster architecture has been evaluated using the MAISIE simulation platform [2]. Several sets of experiments were carried out in order to evaluate the performance as a function of traffic and system parameters, and to compare it with that of other schemes. Most of the simulation experiments share the same network layout and traffic pattern, which are described below.

The channel rate is 800 kbps (the nominal rate of the radio under development with the ARPA sponsored WAMIS project [1]). All data packets (datagram and real-time) are 4 kbits. The preamble for DS-SS acquisition is 500 bits. Thus, data packet transmission time (at 800 kbps) is 6 ms. The default *CYCLE* time is 100 ms.

The offered traffic consists of two components: real-time sessions and datagrams. A new real-time session is generated on average every second (Poisson arrival model) between a random pair of nodes. Session duration is exponential with 3 minutes average. One real time packet is transmitted very 100 ms. At 4 kbits per packet, this corresponds to 40 kbps. Datagrams are also generated between random node pairs, with average default interarrival time = 100 ms. Datagrams have lower priority than real-time packets on the transmission queue. In order to prevent lockout of datagrams, a fraction α of the *CYCLE* time can be reserved for datagrams. In our experiments, the default value is $\alpha = 0$.

5.1. Weighted End-to-end Throughput

The link throughput is defined as the sum of the throughputs on the links which are simultaneously active in the network. The performance measure based on link throughput however is biased towards small transmission range since in this case there is a large number of short links. There is also a large number of small clusters, many of which may actually be disconnected. In practice, we are more interested in end-to-end throughput rather than single link throughput, and would like to maintain connectivity among as many node pairs as possible in the system. To evaluate end-to-end throughput, we generate different topologies with N randomly distributed nodes in a 100×100 square area. We vary the transmission range and for each value we obtain different topologies. Then, we average our throughput measures over the random placements. We measure the end-to-end throughput accounting for possible network disconnection:

$$throughput = \sum_{i=1}^{DC} f_i \frac{LT_i}{L_i}$$

More precisely, we denote

DC = the total number of disconnected components

f_i = the fraction of node pairs in component $i = \dfrac{n_i(n_i - 1)/2}{N(N-1)/2}$ (n_i: the total number of nodes in component i)

LT_i = the total link throughput of component i

L_i = the average path length in component i

In this formula, $\frac{LT_i}{L_i}$ is the average end-to-end throughput in component i and f_i is the weight for this throughput. Simulation results are shown in Figure 19. We find that the optimal transmission range for $N = 30$ is within the interval (35, 45).

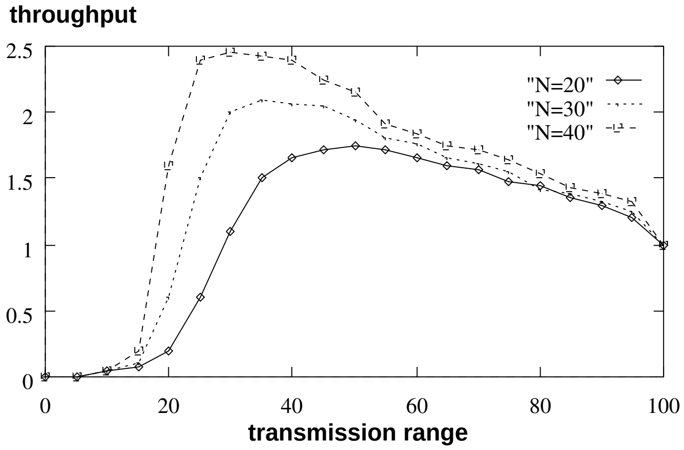


Figure 19: End-to-end throughput

5.2. Real-time and Datagram Traffic Mix

In the next experiment, we consider the effect of variable real-time load on datagram throughput. The topology connectivity is shown in Table 1. Namely, the real-time cycle is allowed to vary from ∞ (i.e. 0 pkt/sec) to 100 ms (i.e. 10 pkts/sec). Default values were used for datagram traffic (i.e. $\alpha = 0$ and $CYCLE$ = 100 ms). Figure 20 reports datagram and real-time throughput for varying traffic load. The datagram throughput is the aggregate throughput (sum over all source/destination pairs). The real-time throughput is the average throughput over a session. We note that, because of the priority given to real-time traffic, and because of the $\alpha = 0$ threshold assumption, the datagram throughput decreases very rapidly as the real-time traffic increases. This points out the importance of a careful selection of the threshold α for fair bandwidth sharing.

5.3. Standby Routing

The following set of experiments evaluates the system performance in a mobile environment with heavy real-time load ($CYCLE$ = 100). Due to mobility, real-time packets may be dropped on already established connections during path changes. The goal of this experiment is to assess the improvement introduced by the "standby" routing feature, i.e. the availability of an alternate route in case the preferred route fails. This feature is of critical importance when stations are mobile. In particular, it is expected to reduce loss rate. Two experiments were run, with speed 2 feet/sec and 8 feet/sec respectively. Table 2 and Table 3 show the results. Standby routing reduces packet loss by more than 50% in both cases.

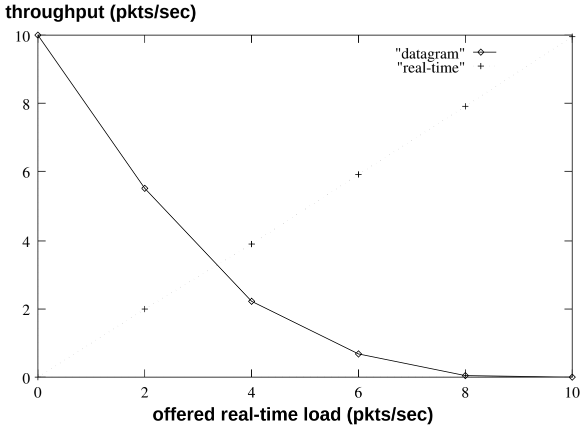


Figure 20: The throughput of mix traffic

	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19
0		x	x	x				x	x		x								x	x
1	x			x	x		x	x	x	x	x	x						x	x	
2	x			x					x		x								x	x
3	x	x	x						x	x	x								x	x
4		x				x		x		x				x		x	x	x		
5					x												x	x		
6		x								x		x			x					
7	x	x			x				x	x	x			x		x		x	x	
8	x	x	x	x				x		x	x								x	x
9		x		x	x		x	x	x		x	x						x	x	
10	x	x	x	x				x	x	x									x	x
11		x					x			x					x			x		
12														x		x				
13					x			x					x			x	x			
14							x					x						x		
15					x			x					x	x			x	x		
16					x	x								x		x		x		
17		x			x	x		x		x		x			x	x	x			
18	x	x	x	x				x	x	x	x									x
19	x		x	x					x		x								x	

Table 1: The system topology (N=20)

	with standby routing	without standby routing
Pkts lost	14 (0.77%)	53 (2.94%)
End-to-End Delay (msec)	290.65	425.45
Throughput (pkts/sec)	9.92	9.70

Table 2: The performance of standby routing (maximum speed = 2 feet/sec)

	with standby routing	without standby routing
Pkts lost	21 (1.17%)	104 (5.78%)
End-to-End Delay (msec)	298.80	721.44
Throughput (pkts/sec)	9.88	9.42

Table 3: The performance of standby routing (maximum speed = 8 feet/sec)

5.4. Scheme Comparison Synopsis

Table 4 and Table 5 compare the proposed Adaptive Cluster scheme with various schemes which have appeared in the literature, namely, PRNET [17], MACA/PR [14], and Cluster TDMA [7]. PRNET is a multihop packet radio network which uses CSMA for channel access. No clustering nor slot synchronization are needed. No separate initialization procedure is required. Simple Bellman Ford Distance Vector is used. Real Time traffic delivery is made more reliable by the use of "duct routing" [17]. Namely, multiple copies of the same packet are carried in parallel on alternate paths to destination. MACA/PR uses IEEE 802.11 (RTS/CTS) access scheme. It implements QoS routing and bandwidth reservations using a scheme similar to the one here proposed. It does not use clustering, code separation, nor slot synchronization. In this respect, it is totally asynchronous like PRNET. Cluster TDMA is an earlier, more complex version of the scheme here presented, with more overhead. The key differences are: clusterhead based clustering; fixed TDMA control slots and "on demand" data slots in the frame; use of multiple codes within each cluster; frame synchronization (i.e. slot alignment) across frames. In this experiment, *CYCLE* = 100. The schemes are ordered by increasing implementation complexity (from the totally asynchronous PRNET to the highly organized Cluster TDMA). Both throughput per connection and aggregate throughput are compared. The performance of our proposed Adaptive Cluster is very similar to that of Cluster TDMA. This is quite of interest, since Adaptive Cluster uses code separation but does not require intercluster synchronization (it only requires intra cluster synchronization). Performance is only moderately degraded by speed increase. MACA/PR appears to provide an attractive compromise between throughput efficiency and simplicity of implementation. On the positive side, MACA/PR exhibits the lowest end-to-end delay (which is critical for some real-time applications such as voice). On the negative side, MACA/PR can not achieve the aggregate throughput efficiency of Cluster type solutions (which benefit from code separation).

	PRNET	MACA/PR	Adaptive Clustering	Cluster TDMA
avg VC throughput (pkts/sec)	4.2	9.68	9.89	9.98
total VC throughput (pkts/sec)	15.54	34.65	97.34	107.89
end-to-end delay (ms)	72.02	48.20	254.23	298.15
avg pkt loss per VC	67.70%	3.15%	1.07%	0.01%

Table 4: Overall performance comparison (2 feet/sec)

	PRNET	MACA/PR	Adaptive Clustering	Cluster TDMA
avg VC throughput (pkts/sec)	4.6	9.69	9.88	9.97
total VC throughput (pkts/sec)	14.54	34.45	95.34	107.70
end-to-end delay (ms)	73.05	66.50	298.80	302.13
avg pkt loss per VC	67.30%	3.07%	1.18%	0.02%

Table 5: Overall performance comparison (8 feet/sec)

6. CONCLUSIONS

We presented an Adaptive Clustering architecture for multimedia support in a multihop mobile network. This architecture is not constrained by a fixed infrastructure, rather, it can be deployed in an environment without infrastructure at all. It can also tolerate mobility well because of the robustness of our adaptive cluster algorithm.

In order to reduce control overhead and overcome the limitation of the number of orthogonal codes, we use only one code within each cluster. Packet transmissions by data sources and real-time sources are interwoven, with top priority given to real-time sources. Within each cluster, TDMA permits to reserve bandwidth for real-time traffic. Taking advantage of Spread Spectrum, adjacent clusters can transmit at the same time on different codes. Simulation experiments have identified key tradeoffs between transmission range and throughput performance, and have shown the advantages of code separation and spatial reuse. The performance of the proposed cluster scheme is similar to that of Cluster TDMA [7], with less implementation complexity (less burdened by synchronization requirements).

REFERENCES

[1] A. Alwan, R. Bagrodia, N. Bambos, M. Gerla, L. Kleinrock, J. Short and J. Villasenor, "Adaptive mobile multimedia networks," *IEEE Personal Communications,* pp. 34-51, April 1996.

[2] R. Bagrodia and W. Liao, "Maisie: A language for the design of efficient discrete-event simulations", *IEEE Transaction on Software Engineering,* pp.225-38, 1994.

[3] D.J. Baker and A. Ephremides, "The architectural organization of a mobile radio network via a distributed algorithm," *IEEE Transactions on Communications,* pp. 1694-1701, Nov. 1981.

[4] D.J. Baker, J. Wieselthier and A. Ephremides, "A distributed algorithm for scheduling the activation of links in a self-organizing, mobile, radio network," *IEEE ICC'82,* pp. 2F.6.1-2F.6.5.

[5] N. Bambos, S.C. Chen, and G.J. Pottie, "Radio link admission algorithms for wireless networks with power control and active link quality protection," *Proceedings of IEEE INFOCOM '95,* 1995.

[6] I. Chlamtac and S. S. Pinter, "Distributed nodes organization algorithm for channel access in a multihop dynamic radio network," *IEEE Transactions on Computers,* pp. 728-737, June 1987.

[7] M. Gerla and J.T.-C. Tsai, "Multicluster, mobile, multimedia radio network," *ACM-Baltzer Journal of Wireless Networks,* Vol. 1, No. 3, pp. 255-265, 1995.

[8] D. Goodman and R.A. Valenzuela, K.T. Gayliard and B. Ramamurthi, "Packet reservation multiple access for local wireless communications," *IEEE Transactions on Communications,* pp. 885-890, Aug. 1989.

[9] T. Hou and V. Li, "Transmission range control in multihop packet radio networks," *IEEE Transactions on Communications,* pp. 38-44, Jan. 1986.

[10] L. Hu, "Topology control for multihop packet radio networks," *IEEE Transactions on Communications,* pp. 1474-1481, Oct. 1993.

[11] L. Hu, "Distributed code assignments for CDMA packet radio networks," *IEEE/ACM Transactions on Networking,* pp. 668-677, Dec. 1993.

[12] K. Keeton, B.A. Mah, S.Seshan, R.H. Katz, and D. Ferrari, "Providing connection-oriented network services to mobile hosts," *Proceedings of USENIX ASSOCIATION Mobile and Location-Independent Computing System,* pp. 83-102, August 1993.

[13] C.R. Lin and M. Gerla, "A distributed architecture for multimedia in a multihop dynamic packet radio network," *Proceedings of IEEE Globecom '95,* pp. 1468-1472, November, 1995.

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

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

[16] J. Jubin and J.D. Tornow, ''The DARPA packet radio network protocols,'' *Proceedings of the IEEE,* January, 1987.

[17] N. Shacham, E.J. Craighill and A.A. Poggio, ''Speech transport in packet-radio networks with mobile nodes,'' *IEEE Journal on Selected Areas in Communications,* pp. 1084-1097, Dec. 1983.