Mobility-Aware Access-Based Clustering in Mobile Wireless Ad Hoc Networks

by

Rajesh Palit

A Thesis Submitted in Partial Fulfillment of the Requirements for the Degree of

MASTER OF SCIENCE

in the Department of Electrical and Computer Engineering

Mobility-Aware Access-Based Clustering in Mobile Wireless Ad Hoc Networks

BY

Rajesh Palit

A Thesis/Practicum submitted to the Faculty of Graduate Studies of The University of Manitoba in partial fulfillment of the requirement of the degree

Of

MASTER OF SCIENCE

Rajesh Palit © 2004

Supervisor: Dr. Ekram Hossain

ABSTRACT

Wireless mobile ad hoc networks consist of mobile nodes which can communicate with each other in a peer-to-peer fashion (over single hop or multiple hops) without any fixed infrastructure such as access point or base station. In a multi-hop ad hoc wireless network, which changes its topology dynamically, efficient resource allocation, energy management, routing and end-to-end throughput performance can be achieved through adaptive clustering of the mobile nodes.

Most of the clustering approaches proposed in the literature primarily focus on the algorithmic aspects of clustering without considering the practical implementation issues and these are often reactive in nature. In this thesis, we propose a framework for Mobility-Aware Proactive Low Energy (MAPLE) clustering in ad hoc mobile wireless networks. The proposed approach addresses the problem of clustering in a medium-access control framework and enables proactive and energy-efficient clustering by exploiting the node mobility information.

MAPLE is compared to non-access-based clustering and access-based clustering protocols. Simulation results show that the proposed framework results in superior clustering performance in terms of stability, load distribution and control message overhead compared to other clustering approaches proposed in the literature.

This Page is Intentionally Blank

Table of Contents

List of Figures

List of Tables

List of Abbreviations

ABCP	Access-based Clustering Protocol
AODV	Ad hoc On-Demand Distance Vector
CABC	Channel Access-Based self-organized Clustering
CH	Cluster Head
CR	Collision Report
DTMC	Discrete-Time Markov Chain
HCN	Highest Connectivity based Clustering
ICMG	Inter-Control Message Guard
MAC	Medium Access Control
MANET	Mobile Ad hoc NETworks
MAPLE	Mobility-Aware Pro-active Low Energy clustering Framework
MBAC	Mobility-Based Adaptive Clustering
MCH	Message from Cluster Head
MFD	Medium Frame Delay
MMD	Max-Min D-Clustering
LCA	Linked Cluster Algorithm
LCC	Least Cluster Change
LID	Lowest ID based clustering
LVA	Link Vector Algorithm
PCBC	Power Control Based Clustering
QoS	Quality of Service
RBCA	Randomized Broadcast Channel Access
RCH	Reply from Cluster Head
RCM	Reply from Cluster Member
RTS	Request To Send
SFD	Short Frame Delay
SNR	Signal to Noise Ratio

TCP	Transmission Control Protocol
TORA	Temporally Ordered Routing Algorithm
TPMA	Three-Phase Multiple Access

Acknowledgement

I like to express my profound gratitude to my supervisor Dr. Ekram Hossain whose keen interest and experience has influenced me to carry out this thesis work. His constant supervision, constructive criticism at all stages of this work made it possible to complete this work.

I take this chance to acknowledge my deep indebtedness to my co-supervisor Dr. Parimala Thulasiraman whose enthusiastic support, scholarly guidance and valuable advice helped me greatly throughout my thesis work.

Special thanks to Dr. Jelena Misic for agreeing to be the external examiner of my thesis. I am also thankful to my colleagues of *Wireless Internet and Packet Radio Network Research Group*, Department of Electrical and Computer Engineering, University of Manitoba for their constructive criticism and suggestions.

Last but not the least, I would like to extend my sincere thanks to the Faculty of Graduate Studies, University of Manitoba and Natural Science and Engineering Research Council (NSERC) of Canada for their financial support.

Dedication

To my grandmother late Mrs. Moni Palit and to my family members and friends whose blessings, love and inspirations have been the greatest possession in my life.

Where the mind is without fear and the head is held high;
Where knowledge is free;
Where the world has not been broken up
into fragments by narrow domestic walls;
Where words come out from the depth of truth;
Where tireless striving stretches its arms towards perfection;
Where the clear stream of reason
has not lost its way into the dreary desert sand of dead habit;
Where the mind is led forward by thee into ever-widening thought and action—
Into that heaven of freedom, my Father, let my country awake.

Rabindranath Tagore, Gitanjali

Chapter 1

Introduction

1.1 Mobile Wireless Ad hoc Networks

Rapid developments in the portable electronic device technology have made the communication devices more compact, powerful and low-cost. Furthermore, the recent advances in wireless communication technology have spawned an increasing demand for various services to the nomadic users over mobile networks. The aim of the future-generation wireless mobile systems is to achieve seamless services across both wired and wireless networks under global user mobility. This is paving the way towards rapid development of infrastructure-less "self-organizing" mobile networks which are expected to complement the infrastructure-based networks in scenarios where the nature of the communication requires the mobile devices to be adaptive and self-organizing [1].

An ad hoc network (also known as a packet radio network or MANET) consists of a set of self-organizing mobile nodes which require no fixed infrastructure, and which communicate with each other over wireless links [2]. A wireless node can directly communicate with other nodes which are within its transmission range. If two nodes cannot communicate directly, an intermediate node(s) is used to relay or forward data from the source node to the destination node. By MANET we usually refer to a multi-hop ad hoc network where the distance between two nodes can be more than one hop.

A snapshot of a mobile wireless ad hoc network is given in Fig. 1.1. The wireless devices vary in their size, communications capabilities, computational power, memory, storage, mobility, and battery capacity (Table 1.1). This heterogeneity affects communication performance and the design of communication protocols. The mobile

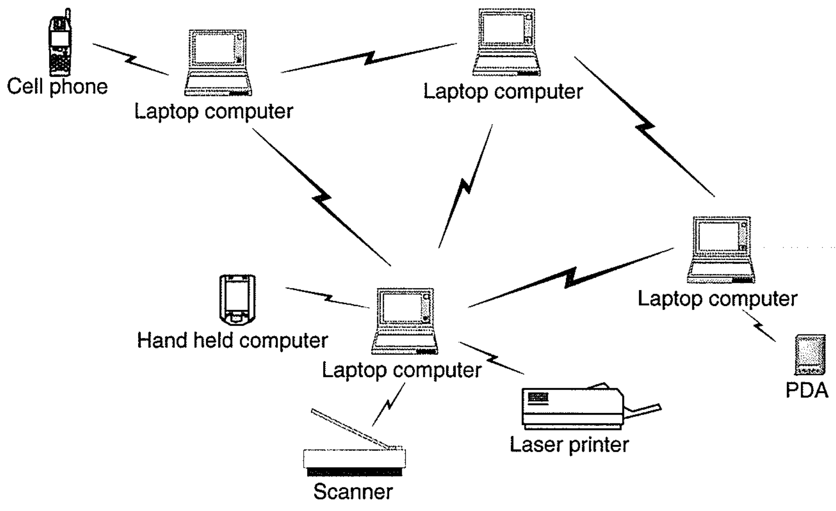


Figure 1.1. *A mobile wireless ad hoc network*

devices in an ad hoc network should not only detect the presence of the connectivity with neighboring devices or nodes, but also identify the devices' types and their corresponding attributes.

Probably the most common application requiring an ad hoc network is mobile conferencing. When a group of people gather outside their usual work environment where the wired network infrastructure is absent, they need to establish a network on the fly to access Internet or share data among themselves [3]. The other potential applications of wireless ad hoc networks include instant network infrastructure to support disaster recovery communication requirements, mobile patient monitoring, collaborative computing, vehicle-to-vehicle communication, distributed control, and micro-sensor networking [17]. The next generation cellular phones will connect to their neighboring phones or wireless equipment directly without the help of the base stations by forming wireless link instantly.

A micro-sensor network is a distributed network of thousands of collaborating tiny devices, which gather multidimensional observations of the environment [4]. Many of the necessary components and technologies for micro-sensor networks are already available. By collecting the multiple observations from different perspectives - a spatially distributed network of micro-sensor nodes returns a rich, high resolution, multidimensional picture of the environment. In micro-sensor networks, the nodes

Table 1.1. *Wireless devices.*

Devices	Example	Communication power	Computational power	Battery capacity	Mobility
Sensors	Temperature, humidity sensors	Low	Low	Low	Medium
Audio/video (remotely administrated)	Remote controller	Low	Low	High	Low
Computers	Pocket PC, laptop	Medium	High	Medium	Medium
Communicators	Mobile phone	Medium	Medium	Medium	High
Vehicular devices		High	High	High	High

are self-organized into ad hoc networks, and work in a distributed manner.

The topology of an ad hoc network changes over time due to the mobility of the nodes, efficient radio resource allocation, energy management, routing and end-to-end throughput performance are, therefore, challenging issues to the network designers [21]. By clustering the nodes in the entire network, better radio resource utilization can be achieved [6]. Routing based on clustering reduces the amount of information propagated in the network (e.g., by reducing topology update broadcasts) and the routing delays ([9]-[13]). Also, by carefully adjusting the transmission power, the mobile nodes in the clusters can conserve battery power and generate less interference to other transmissions in the network. This gives the benefit of spatial reuse of channel spectrum which in turn increases the network capacity.

A portion of nodes are dynamically selected as cluster heads or leaders using some distributed algorithms which are termed as clustering algorithms. If the maximum distance between any pair of nodes in an ad hoc network clusters is k hops, the resulting clusters are termed as k-hop cluster. A cluster head acts as a local coordinator of data transmissions within its cluster [15]. The nodes except the cluster heads are known as cluster members or ordinary nodes. Some of the nodes may reach more than one cluster head and these 'Gateway' nodes are used for inter-cluster communications.

The status of a newly turned on node is said to be non-clustered. An non-clustered node tries to join an existing cluster if there is any; otherwise it becomes a cluster head. The clustering algorithms (e.g., in [14]-[25]) can be broadly classified into two categories: cluster head-based and non-cluster head-based. In the former case, cluster heads are elected for individual clusters and for the latter case no cluster heads are elected. For large scale networks, cluster head-based approach is more performance efficient in terms of reducing clustering and traffic overheads [14].

The stability of the clusters (e.g., in terms of cluster head change rate), the control signaling overhead involved in maintaining the clusters, the energy consumption at the mobile nodes for cluster formation and maintenance and the load distribution among the mobile nodes primarily determine the performance of a clustering algorithm. Cluster heads store key information such as clustering topology or routing information. Frequent changes of cluster heads will incur a lot of extra message overhead to maintain the information. If the overhead for control message signaling is very high, the mobile nodes will quickly exhaust their limited battery power. Moreover, the capacity of data transmission will be affected due to cluster head changes. An ad hoc node may become non-clustered a number of times in its lifetime and form or join clusters. To achieve acceptable data transmission capacity, the cluster formation time of a node should be kept small which reduces the time during which a node remains non-clustered.

The goal of this thesis work is to explore the issues relating to clustering in mobile ad hoc networks and to design a clustering algorithm which is mobility aware and consumes less energy. We propose a novel framework for Mobility-Aware Proactive Low Energy (MAPLE) clustering of the mobile nodes (cluster head-based) in an ad hoc wireless network. MAPLE clustering differs from other clustering schemes in that it exploits the radio link level information on signal quality (e.g., variations in received signal strength) in a proactive manner to track the mobility pattern of the wireless nodes and to choose low cost link from a cluster member to cluster head. This results in low-energy clustering. Due to the proactive nature of the cluster formation, the number of link failures is also reduced. It uses a medium access control (MAC) framework where the contention in accessing the channel is significantly reduced by reserving the channel beforehand.

1.2 Thesis Organization

The reminder of this thesis is organized as following:

- The importance of clustering and related work is discussed in chapter 2.
- Chapter 3 presents the proposed clustering framework and the clustering algorithm.
- The performance analysis of non-access-based clustering algorithms is presented in chapter 4 followed by a comparison of performance of MAPLE with a non-access-based clustering algorithm.
- We compare the performance of MAPLE with the existing access-based algorithms in chapter 5.
- Conclusions and future work are presented in chapter 6.

Chapter 2

Importance of Clustering and Related Work

In this chapter, we discuss the impacts of clustering on radio resource management and protocol performance in a multi-hop wireless ad hoc network. We also present a survey of the different clustering mechanisms proposed in the literature.

2.1 Importance of Clustering

- *Clustering and Medium Access Control*: In a mobile ad hoc network, node mobility, vulnerability of the radio channel(s), and the lack of any central co-ordination give rise to the well known *hidden node* and *exposed node* problems. The medium access control protocols must handle these problems.
  A hidden node is a node which is out of range of a transmitter node (node **A** in Fig. 2.1), but in the range of a receiver node (node **B** in Fig. 2.1) [7]. A hidden node does not hear the data sent from a transmitter to a receiver (node **C** is hidden from node **A**). When node **C** transmits to node **D**, the transmission collides with that from node **A** to node **B**. Obviously, the hidden nodes lead to higher collision probability.
  An exposed node (node **C** is exposed to **B** in Fig. 2.2) is a node which is out of range of a receiver (node **A**), but in the range of the corresponding transmitter (node **B**). Node **C** defers transmission (to node **D**) upon detecting data from node **B**, even though a transmission from node **C** does not interfere with the reception at node **A**. The link utilization may be significantly impaired due to the exposed node problem.

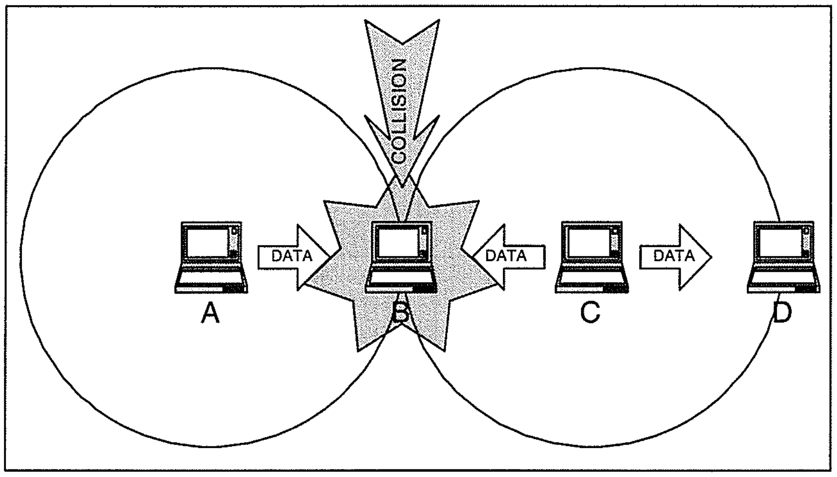


Figure 2.1. *Hidden node problem.*

The hidden node and the exposed node problems have a different look in the multi-channel environments [6]. Let us consider a scenario where two mobile nodes two-hops away from each other are trying to transmit data to a node which is one-hop away from both of the transmitting nodes. If both transmitting nodes use the same channel, the transmissions will be garbled. Even if the targets of these two transmitting nodes are different, there is still a chance of collision. This situation could be avoided if each mobile node would have information on the channels used by the other nodes which are two-hops away so that each transmitting node could use a channel which is different from another transmitting node at least two-hops away. However, realizing this solution in a way so that the least amount of channel bandwidth is used for control signaling is an NP-complete problem [6].

One solution to the above mentioned problem that does not incur huge control overhead is through clustering. By using clustering the network is divided into smaller groups so that the wireless channels can be reused across spatially distributed regions. Most hierarchical clustering architectures for mobile radio networks are based on the concept of cluster head (CH). The CH acts as a local coordinator for transmissions within the cluster, and is responsible for collecting information on channel use from neighboring clusters and selecting gateways to route packets to them. Therefore, the control message overhead for routing can

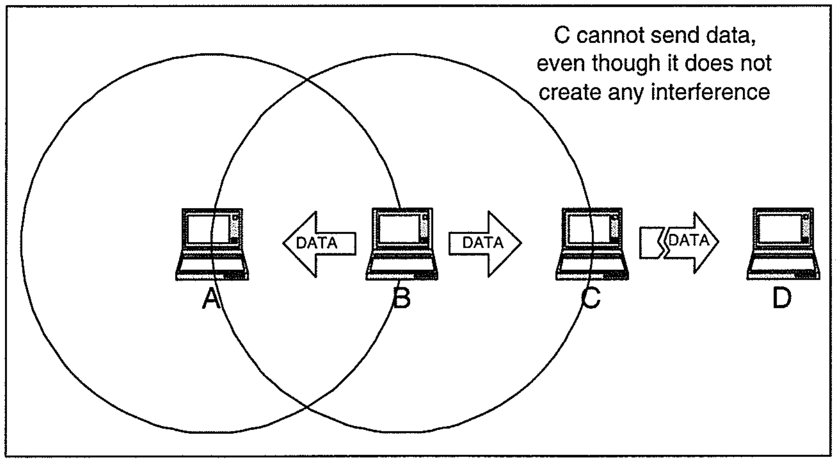

C cannot send data, even though it does not create any interference
DATA
DATA
DATA
A
B
C
D

Figure 2.2. *Exposed node problem.*

be significantly reduced. However, sometimes the cluster head may become a bottleneck of the cluster since it's a central point of administration and a failure in CH degrades the performance of the entire network.

Controlling transmission power is one of the most effective ways to reduce the interference among co-channel transmissions in a multichannel ad hoc wireless network [5]. Based on the clustering of the mobile nodes, the transmission power (and hence transmission range) of the mobile nodes can be controlled. Transmission power control not only reduces interference, but also saves valuable battery power resulting in more energy-efficient MAC protocols. Also, by adapting the transmission range the impact of mobility on the network performance can be reduced to some extent.

- *Clustering and Routing*: The tasks of a routing protocol are to find path/route between a source node and a destination node and maintain the route until the transmission session ends. The most desirable properties of a routing protocol are robustness and stability. There are two extreme routing mechanisms for mobile ad hoc networks – shortest-path routing that is suitable for low rate of topology change, and flooding which is suitable for high rate of topology change. Flooding increases communications overhead and shortest path techniques require each node to maintain up-to-date routing tables. Both of these techniques result in increased co-channel interference and degrade network throughput and response-time performances [9].

Improved system performance can be achieved by using routing protocols designed based on clustering ([9]-[13]). The goal of clustering is to partition the network logically in such a way that slow changes in the actual topology do not affect the logical structure of the network. It helps the routing process by allocating the tasks of establishing and maintaining routes among the mobile nodes. Some of the nodes in each cluster can act as gateways to communicate with the neighboring clusters. The cluster head, gateways and cluster members collectively participate in the routing process [17]. A proactive routing algorithm such as Link Vector Algorithm (LVA) [2] can be used for intra-cluster routing where each node in a cluster maintains topology information and routes to every node in its cluster. Routes to destinations outside of a node's cluster are established on a demand basis (i.e., by using reactive routing strategy such as AODV, TORA etc). This type of two level routing strategy reduces the time and control signaling overhead significantly. Clustering also reduces the effect of link failures due to the physical movements of the mobile nodes. If a link along a route fails, the nodes in the particular cluster may establish the link again.

Routing based on clustering may also result in efficient battery power usage at the mobile nodes. Let us consider three mobile nodes **A**, **B**, and **C** which use identical transmitters and receivers (Fig. 2.3) and let T_h be the minimum signal power required at the receiver for successful data reception. If node **A** tries to send data to node **C**, the transmission power must be at least $T_h d^n_{AC}$ (considering signal attenuation only due to path loss with path-loss exponent n, where n is generally ≥ 2). But if the transmission takes place via node **B**, then total power required for the transmission is $T_h d^n_{AB} + T_h d^n_{BC}$. Since $d^n_{AB} + d^n_{BC} < d^n_{AC}$, rather than transmitting directly to the destination node, relaying the transmission through an intermediate node (which serves as a 'gateway' within a cluster) may result in lower transmit power. This tradeoff between the transmission power and the number of hops along the route from the source node to the destination node (and hence transmission delay) should be considered while designing a clustering algorithm. Cluster-based routing which involves routing

Table 2.1. *Impact of clustering on protocol performance*

Clustering	Transport protocol (e.g., TCP)	Since routing and MAC protocol performances improve, transport protocol performance improves.
	Routing	Cluster-based routing incurs lower control signaling overhead and results in improved system performance.
	MAC	Clustering allows efficient channel resource management and battery power usage.

along the gateway nodes (for inter-cluster routing) may therefore result in better usage of battery power.

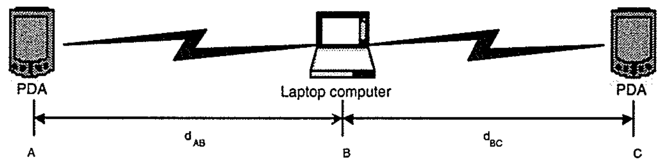


Figure 2.3. *Tradeoff between transmission power and number of hops en-route.*

Again, if a clustering mechanism can exploit the mobility information in a proactive manner, the number of link failures can be reduced and hence better routing performance can be achieved.

- *Clustering and Transport Protocol Performance*
  Since the performance of a transport layer protocol is largely impacted by the underlying network and radio link control/medium access control protocol, transport protocol performance improves as the routing and/or wireless channel access performances improve (Table 2.1).

2.2 Clustering Algorithms

An ad hoc network can be modeled as a graph $G = (V, E)$, where two nodes are connected by an edge if they can communicate with each other. The objective of a clustering algorithm is to find a feasible interconnected set of groups covering the entire node population. A set of nodes S in $G = (V, E)$ is called a D-hop dominating set if every node in V is at most D $(D > 1)$ hops away from a vertex in S. For a graph G and an integer $K \geq 0$, the problem of determining whether G has dominating set of size $\leq K$ was proven to be NP-complete ([19],[24]). For the special family of graphs known as unit disk graphs that represent ad hoc wireless networks, polynomial time and message complexity approximation solution to the K-clustering problem (where every two wireless hosts are at most K hops away from each other) can be found [24].

Several of the popular heuristic-based clustering algorithms, namely, the linked cluster algorithm (LCA) [14], the lowest ID (LID) algorithm [15], highest connectivity algorithm [10], least cluster change (LCC) algorithm [10], max-min D-clustering algorithm [19], mobility-based adaptive clustering algorithm (MBAC) [17], and access-based clustering protocol (ABCP) [21] will be discussed in the following sections.

2.2.1 Linked Cluster Algorithm (LCA)

The linked cluster algorithm in [14] was proposed as a survivability solution for HF Intra-Task Force (ITF) networks to organize radio-equipped mobile nodes into a reliable network structure and to maintain this structure in the face of arbitrary topological changes. The nodes in a HF ITF network communicate via radio links in the HF band (2-30 MHz) and one important characteristic of such a network is changing topology due to variations in the radio communication range of the nodes.

In the proposed architecture the network is organized into a set of node clusters and each node belongs to at least one cluster. Every cluster has its own cluster head which acts as a local controller for the nodes in that cluster. The cluster heads control the access of the radio channels. The cluster heads are linked via gateway nodes to connect the neighboring clusters and to provide global network connectivity. The LCA algorithm establishes (from any initial node configuration) and maintains (for mobility of nodes) the logical topology as long as the nodes have not moved so far apart that the network has become disconnected. The LCA algorithm is distributed and does not depend on the existence of any particular node. The algorithm has two logical stages - formation of clusters and linking of the clusters. At the completion of the LCA, each node becomes either an ordinary node, a gateway node or a cluster head.

The HF band is divided into M subbands and separate runs of the algorithm are performed consecutively for M epochs to obtain the corresponding sets of clusters. The algorithm is run for the ith subband of the HF channel at the ith epoch. During any epoch only one set of linked cluster is organized, the remaining $M - 1$ sets are unaffected. Each epoch is divided into two frames (*frame 1* and *frame 2*) and each frame is subdivided into N timeslots, where N is the total number of nodes (Fig. 2.4). The epochs repeat in a cyclic fashion providing a continual updating process. The necessary control messages are transmitted in a separate control channel.

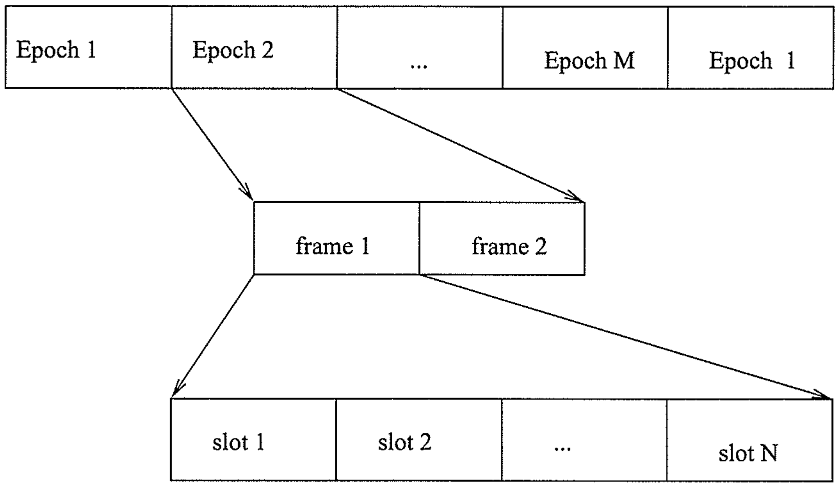


Figure 2.4. *Control channel schedule for LCA.*

During execution of the LCA algorithm each node maintains the following data structures:

- *heads_one_hop_away* is a list recording those cluster heads that are connected to a node.
- *heads_two_hops_away* is a list of the cluster heads, which are not directly connected, but connected to the neighbors of a node.

- *nodes_heard* is a list that includes all neighboring nodes to which a bi-directional link exists.
- *connectivity* is a matrix having binary entries. A value of 1/0 in the (i, j) position indicates the existence/absence of a link between nodes i and j.
- *own_head* is the identity of the cluster head for a given node.
- *node_status* indicates the status (i.e, *ordinary*, *gateway*, or *cluster head*) of a node.

In each epoch, the algorithm proceeds on a frame-by-frame basis. During the ith slot of *frame 1*, node i broadcasts its *nodes_heard* list (it has heard during the earlier slots of this frame). Therefore, at the end of *frame 1* node i can fill in elements (i, j) in the connectivity matrix where $j > i$. During *frame 2* each node broadcasts its full connectivity information in its assigned slot and node i determines the two-way connectivity of links (i, j) for $j < i$. At the end of *frame 2* all bidirectional links are determined. However, the global connectivity information is not available to every individual node.

Also, at the ith slot of *frame 2* node i also transmits its *node_status.* The cluster head selection rule is that the node with the highest ID number among a group of nodes is the first candidate to become a cluster head. At the end of *frame 2* each node is able to fill in its *heads_one_hop_away* and *heads_two_hops_away* lists and each node has at least one cluster head in its vicinity. Note that, there is only one cluster head if all nodes are within a distance of one hop from each other.

After the clusters are formed at the end of *frame 2*, a procedure *delete_heads* is used to eliminate redundant clusters. For example, if one cluster covers (i.e., overlaps) another, *delete_heads* eliminates the covered cluster head.

The clusters are linked by assigning *gateway* status to some nodes that connect adjacent clusters. If two cluster heads are linked directly there is no need for gateways. In case of overlapping clusters where the cluster heads are not directly linked, one gateway node is needed which can be chosen among the nodes in the common intersection region. The gateway selection is performed by procedure *linkup1* and the highest numbered node in the intersection region is chosen to become the gateway. In case of non-overlapping clusters, a gateway node pair must be formed to link the cluster heads and this is performed by procedure *linkup2.* The nodes in a pair with the largest sum of the ID numbers is chosen to be the gateway nodes.

The entire linked cluster algorithm can be described as follows:

```
Process at node i:

Begin
 own_head = self,
 node_status = clusterhead,
 heads_one_hop_away = empty,
 heads_two_hops_away = empty,
 nodes_heard = empty,
 connectivity = identity matrix.

 Repeat

  // The following tasks are performed during frame 1 and 2
  // in each epoch
  begin

   // Frame 1 events

   Node i broadcasts its nodes_heard list in slot i.
   In other slots node i receives nodes_heard list
     from its neighboring nodes.
   If node i receives nodes_heard list from node j
     put node j into node i's nodes_heard list.
   If node i was heard by node j
       set connectivity[i, j] = 1.

   // Frame 2 events

   Node i determines its node status with the information
```

```
      collected in frame 1.
   In slot i node i broadcasts row i of its connectivity
      matrix and node_status.

   If node i receives connectivity/status message from node j
     it fills in row j of its connectivity matrix.
   If (j < i)
     connectivity[i, j] = connectivity[j,i].
   If status of node j == clusterhead
     include j in heads_one_hop_away list.

  End
 Until (epoch ends)

 // Fill in heads_two_hops_away list

 Let node k is the cluster head for node j.
 If node i, which is bidirectionally linked to node j,
 is not connected to node k, then add k in heads_two_hops_away
 list of node i.

 The above procedure may produce a few unnecessary cluster heads
 under some circumstances. A procedure named delete_heads is
 invoked to remove these redundant cluster heads.

 Call procedure linkup1 to select the gateways for
 overlapping clusters.

 Call procedure linkup2 to select the gateways for
 non-overlapping clusters.

End
```

The fundamental assumption of the proposed algorithm is that each node maintains a common clock and knows the precise length of each time frame. Again, the number of nodes in the network is assumed to be known *a priori*. If the number of nodes cannot be bounded with certainty, a modification of the algorithm would be necessary to allow the occasional adjustment of the frame length. Another limitation of this algorithm is that it chooses the highest-ID node as cluster head which may result in an unbalanced load distribution. Since in each frame the nodes have to broadcast their *nodes_heard* list, the control message overhead is relatively high. LCA does not consider the node mobility, adaptive transmission range and power efficiency issues.

2.2.2 Max-Min D-Clustering Algorithm (MMD)

The max-min D-clustering algorithm proposed in [19] uses a load-balancing heuristic (max-min heuristic) to form D-hop clusters in a wireless ad hoc network so that a fair distribution of load among cluster heads can be ensured. In a D-hop cluster each node is at most D-hops away from the cluster head. The algorithm aims to avoid the clock synchronization overhead, limit the number of messages sent between nodes to $O(D)$, improve cluster stability and control the density of cluster heads as a function of D. Similar to the LCA, cluster heads are determined based on the node ID.

Execution of the max-min heuristic involves $2D$ rounds of information exchange and each node needs to maintain two arrays *WINNER* and *SENDER* each of size $2D$ node IDs. The *WINNER* and *SENDER* are the winning node ID and the node that sent the winning node ID, respectively, of a particular round. Initially each node sets its *WINNER* to be equal to its ID.

The heuristic has four logical stages - *floodmax*, *floodmin*, determination of cluster heads and linking of clusters. The *floodmax* phase consists of D rounds of information exchange and during each round each node broadcasts its present *WINNER* value to all of its one-hop neighbors and chooses the largest ID as the new *WINNER*. The nodes record the *WINNER* for each round. Therefore, *floodmax* propagates the largest node ID in each nodes D-neighborhood and the node IDs that it leaves at the end are elected as cluster heads. However, it may result in an unbalanced loading for the cluster heads. After *floodmax*, D rounds for the *floodmin* phase start to propagate smaller node IDs. In contrast to *floodmax* each node chooses the smallest rather than the largest value as its new *WINNER*. Any node ID that occurs at least once as a *WINNER* in both phases at an individual node is called a *node pair*. At the end of *floodmin*, each node determines its cluster head based on the entries in *WINNER* for the $2D$ rounds of flooding using the following rules:

- *Rule 1*: If a node has received its own ID in the second round of flooding, it declares itself a cluster head. Otherwise Rule 2 is applied.
- *Rule 2*: Among all node pairs, a node selects the minimum node pair to be the cluster head. If a node pair does not exist for a node then Rule 3 is applied.
- *Rule 3*: The maximum-ID node in the first round of flooding is elected as the cluster head for this node.

After cluster head selection each node broadcasts its elected cluster head to all of its neighbors. After hearing from all neighbors a node can determine whether it is a gateway node or not. If all neighbors of a node have same cluster head as its own cluster head, then the node is not a gateway node. If there exist some neighbors with different cluster heads, then the node is a gateway node.

To establish the backbone of the network the gateway nodes begin a converge-cast to link all nodes in the cluster to the cluster head and link the cluster head to other clusters. There are certain scenarios, where the max-min heuristic will generate a cluster head that is on the path between a node and its elected cluster head. During converge-cast, the cluster head receiving the message first adopts the node as one of its children and immediately sends a message to the node identifying itself as the new cluster head.

The time complexity of the heuristic is $O(D)$ rounds and the storage complexity is $O(D)$. Compared to the LCA, the max-min D-clustering algorithm was observed to produce fewer cluster heads, larger sized clusters and longer cluster head duration on the average. Also, the average cluster head duration and cluster member duration were observed to be higher for max-min D-clustering compared to those for a connectivity-based clustering. The cluster head duration was observed to increase with increasing network density. Max-min heuristic results in a backbone with multiple paths between neighboring cluster heads which provides fault tolerance in the network backbone.

Max-min heuristic does not consider node mobility, transmission power and power efficiency explicitly into consideration. More specifically, since it does not take the mobility or node failure into consideration topology changes may cause some nodes to be stranded without cluster heads when the execution of the heuristic is triggered late.

2.2.3 Lowest ID (LID) Clustering Algorithm

This is a two-hop clustering algorithm. While executing this algorithm, a mobile node periodically broadcasts the list of nodes that it can hear (including itself) (Fig. 2.5). A node which only hears nodes with ID higher than itself from the one-hop neighborhood, declares itself a cluster head. It then broadcasts its ID and cluster ID (i.e., ID of the cluster head). A node that can hear two or more cluster head is a gateway node, otherwise, it is an ordinary node or cluster member.

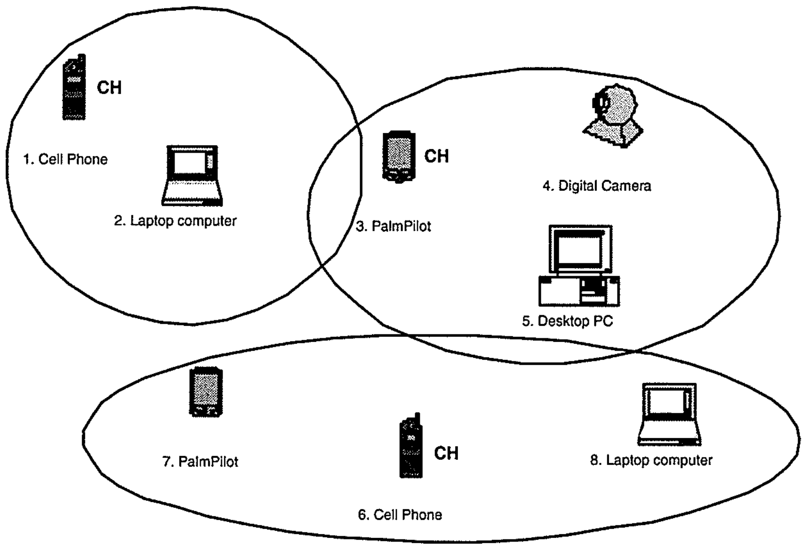


Figure 2.5. *Clustering using lowest-ID algorithm.*

The clustering algorithm can be described as follows [15]:

```
Process at node i:
Begin
 // Let S be the set of IDs of one-hop neighbors of node i
 // including itself.
 // A clustering message contains node ID and  cluster ID.

 Step 1: // This step is executed when node i has the minimum ID in S.

       1.a Set i as node i's own cluster ID.
       1.b Broadcast a cluster message with (i, i).
       1.c Remove i from S.

 Step 2: // This step is executed when a clustering message is received
         // at node i.

       2.a Update the clustering information table (at node i) with
           the received data and remove sender's node ID from S.

       2.b If the sender node is a CH and i's cluster ID is UNKNOWN or
           sender's cluster ID is less than i's cluster ID

             set the i's cluster ID to sender's ID.

       2.c If i is the minimum node ID in S and i's cluster ID
           is still UNKNOWN

            set the i's cluster ID to i,
            remove i from S,
            broadcast a cluster message (i, i).

       2.d Repeat step 2 until S is empty.
```

End

2.2.4 Highest Connectivity (HCN) Clustering Algorithm

Connectivity or degree of a mobile node refers to the number of nodes in its one hop neighborhood. Each node broadcasts the list of nodes that it can hear (including itself). A node is elected as a cluster head if it is the most highly connected node of all its 'uncovered' neighbor nodes (in case of a tie, the lowest ID node is chosen as the cluster head) (Fig. 2.6). A node which has not elected its cluster head yet is an 'uncovered' node, otherwise it is a 'covered' node. A node, which has already elected another node as its cluster head, gives up its role as a cluster head.

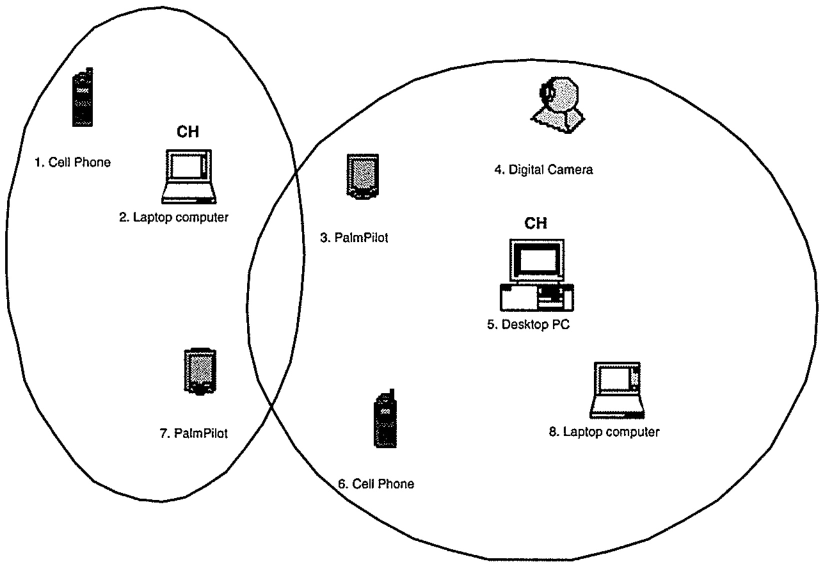


Figure 2.6. *Clustering using highest connectivity algorithm.*

A node gathers connectivity information about its neighbors and uses it for cluster formation. Compared to the LID-based clustering the message overhead is higher, because in LID-based clustering only the node ID information are used. However, for a connectivity-based clustering algorithm the cluster head change is more frequent, and therefore, the load distribution is more fair.

The algorithm can be described as follows:

```
Process at each node i:
Begin
 // Let S be the set of IDs of one-hop neighbors of node i
 // including itself.
 // A clustering message contains node ID and  cluster ID.

 Step 1: // This step is executed when node i has the highest
         // degree in the neighborhood. If there are more than
         // one node with the highest degree, then node i has the
         // lowest ID among these nodes.

         1.a Set i as node i's cluster ID.
         1.b Broadcast a clustering message with (i, i).
         1.c Remove i from S.

 Step 2: // This step is triggered when a cluster message is received
         // at node i.

         2.a Update the clustering information table (at node i) with
             the received data and remove sender's node ID from S.

         2.b If the sender node is a cluster head and node i's cluster ID
             is UNKNOWN or sender's degree is higher than that of node
             i's present cluster head

                set the i's cluster ID to sender's ID.

         2.c If i is the lowest ID which has highest degree in S and
             node i's cluster ID is still UNKNOWN
```

```
                set the i's cluster ID to i.
End
```

2.2.5 Least Cluster Change (LCC) Algorithm

Since frequent cluster changes adversely affect the performance of radio resource allocation and scheduling protocols, cluster stability is a major consideration for designing a clustering algorithm. With a view to increasing cluster stability, the LCC algorithm assumes that cluster heads may change only under either of the following two conditions: when two cluster heads come within the transmission range of each other, or when a node loses its membership in any other cluster.

The LCC mechanism can be described as follows:

- At the beginning, the lowest-id or highest connectivity clustering algorithm is used to form initial clusters.
- When a non-cluster head node in cluster i moves into cluster j, there are no changes for cluster i and cluster j in terms of cluster head (only the cluster membership changes).
- When a non-cluster head node moves out of its cluster and does not enter into any existing cluster, it becomes a new cluster head, forming a new cluster.
- When cluster head $c(i)$ from cluster i moves into cluster j, it challenges the corresponding cluster head $c(j)$. Either $c(i)$ or $c(j)$ will give up its cluster head position according to lowest-id or highest connectivity scheme.
- Nodes which become separated from a cluster, will recompute the clustering according to lowest-id or highest connectivity scheme.

The main objective of this algorithm is to minimize the number of cluster head changes. The stability of the clusters increases significantly compared to that due to only LID-algorithm or HCN-algorithm. The complexity of the algorithm depends on the technique that is used to form the clusters (i.e., LID or HCN). Since the number of cluster head changes is minimum here, the load distribution would be more unfair.

2.2.6 Mobility-Based Adaptive Clustering (MBAC)

The mobility-based adaptive clustering algorithm in [17] uses an estimate of path availability (which changes due to node mobility) for organizing clusters dynamically. This scheme does not use the concept of cluster head. The proposed adaptive clustering framework supports an adaptive hybrid routing mechanism which can be more responsive and effective when mobility rates are low and more efficient when mobility rates are high. The distributed asynchronous clustering algorithm maintains clusters which satisfy the (α, t) criterion, that is, there is a probabilistic bound α on the mutual availability of paths among all nodes in the cluster over a specified interval of time t.

Let $P^k_{m,n}(t)$ be the status of the path k from node n to node m at time t. $P^k_{m,n}(t) = 1$ if all the links in the path are active at time t, and $P^k_{m,n}(t) = 0$ if one or more links in the path are inactive at time t. The path availability $\pi^k_{m,n}(t)$ between the two nodes n and m at time $t \geq t_0$ is given by: $\pi^k_{m,n}(t) = Pr\left(P^k_{m,n}(t_0 + t) = 1 | P^k_{m,n}(t_0) = 1\right)$.

Path k is defined as an (α, t) path if and only if $\pi^k_{m,n}(t) \geq \alpha$. If nodes n and m are mutually reachable over (α, t) paths, they are said to be available. An (α, t) cluster is a set of (α, t) available nodes.

The cluster parameters α and t are tightly coupled. The parameter α controls the cluster's inherent stability. Larger values of t imply better cluster stability and reduce the computational requirements of cluster maintenance. However, since large values of t will reduce the path availability between nodes of a cluster for the same mobility patterns, they will tend to result in smaller clusters. The parameter α should be chosen considering the traffic intensity and QoS requirements of the connections routed through the clusters. Under the assumption that the path availability is an ergodic process, α represents the average portion of time an (α, t) path is available to carry data, and hence α determines the lower bound on the effective capacity of the path over an interval of length t. Modeling each node as an independent M/M/1 queue, and assuming that t is identical at each node in a cluster, the lower bound on path availability can be found as follows:

$$t \geq \frac{1}{\alpha C\mu - \lambda}$$

$$\Longrightarrow \alpha \ \geq \ \left(\frac{1+\lambda t}{\mu t C}\right) \tag{2.1}$$

where C is the link capacity (in bits/s), $1/\mu$ is the mean packet length (in bits), λ is the aggregate packet arrival rate, $\alpha C\mu$ is the effective service rate, and $\frac{1}{\alpha C\mu-\lambda}$ is the mean packet delay.

The (α, t) cluster algorithm is event-driven and requires the clustered nodes to determine whether or not the (α, t) criteria continues to be satisfied following a topological change. Nodes can asynchronously join, leave, or create clusters. A timer (referred to as the α timer) is maintained at each node which determines the maximum time t for which the node can guarantee path availability to each destination node in the cluster with probability $\geq \alpha$. If any one of the paths is found to be no longer an (α, t) path, the node leaves the cluster. The events that drive the (α, t) cluster algorithm are *node activation*, *link activation*, *link failure*, *expiration of the α timer* and *node deactivation.*

- *Node activation*: To join a cluster, an activating node (or source node) has to obtain the topology information for the cluster from its neighbors and determine the (α, t) availability of all the destination nodes in that cluster. If it is unable to join a cluster, it creates its own cluster (*orphan cluster*).
- *Link activation*: Link activation is triggered when an orphan node attempts to join a cluster. The node receives the cluster topology information and evaluates cluster feasibility and depending upon the outcome of the evaluation it either joins the cluster or returns to its orphan cluster status.
- *Link failure*: When triggered, this event causes a node to determine if the link failure has caused the loss of any (α, t) paths to the destinations in the cluster. A link failure can be detected by a node through the network-interface layer protocol or a topology update which reflects a link failure. The link failure information is forwarded by each node to the remaining cluster destinations and each node receiving this topology update reevaluates the (α, t) path availability. If the node detects that a destination has become unreachable, then it removes that destination from its routing table. If it finds that none of the nodes within the cluster is (α, t) reachable, it leaves the cluster.
- *Expiration of α timer*: Each node in a cluster periodically estimates the path

availability to each destination node in the cluster. This is controlled by the α timer and based on the topology information available at each node. The actions taken by a node upon the expiration of its α timer are similar to those due to link or node failure except that the α timer triggers an orphan node to reattempt to join a cluster in a way identical to link activation.

- *Node deactivation*: A node can gracefully deactivate or depart voluntarily from the cluster by announcing its departure through a topology update message to all nodes in the cluster. If a node fails suddenly or becomes disconnected from the cluster due to mobility, it becomes an orphan node and proceeds according to the rules of node activation.

To evaluate path availability, which is the basis for (α, t) cluster management, a random walk-based mobility model is assumed where each node's movement consists of a sequence of random length intervals called mobility epochs during which a node moves in a constant direction at a constant speed. To characterize the availability of a link between two nodes during a period of time (t_0, t_0+t), mobility distribution of a single node is first determined, and then it is extended to derive the joint mobility distribution which can be used to determine the link availability distribution. Assuming that the links along a path between two nodes fail independently, the path availability can be determined as the product of the individual link availability metrics.

The mobility profile of a node n can be expressed by three parameters: λ_n, μ_n, and σ_n^2, where μ_n and σ_n^2 are the mean and variances of the speed during each epoch, and $1/\lambda_n$ is the mean epoch length (where the epoch lengths and the speeds during each epoch length are assumed to be identically and independently (i.i.d.) distributed). If $\overrightarrow{R}_n(t)$ is the random mobility vector for node n, where the magnitude $R_n(t)$ and the phase angle θ_n represent the aggregate distance and direction of the mobile node, respectively, the distributions of $R_n(t)$ and θ_n are given by

$$\begin{aligned} Pr(\theta_n \leq \phi) &= \frac{1}{2\pi}\phi, 0 \leq \phi \leq 2\pi \\ Pr(R_n(t) \leq r) &\approx 1 - exp\left(\frac{-r^2}{\alpha_n}\right), 0 \leq r \leq \infty \end{aligned} \tag{2.2}$$

with $\alpha_n = (2t/\lambda_n)(\sigma_n^2 + \mu_n^2)$. Therefore, the random mobility vector has Rayleigh distributed magnitude and uniformly distributed direction.

Now, let $(\lambda_m, \mu_m, \sigma_m^2)$ and $(\lambda_n, \mu_n, \sigma_n^2)$ describe the mobility profiles for two mobile nodes m and n. If the random mobility vectors for these nodes are $\overrightarrow{R}_m(t)$ and $\overrightarrow{R}_n(t)$, respectively, the random mobility vector of node m with respect to node n is $\overrightarrow{R}_{m,n}(t) = \overrightarrow{R}_m(t) - \overrightarrow{R}_n(t)$ which is approximately Rayleigh distributed (with parameter $\alpha_{m,n} = \alpha_m + \alpha_n = \frac{2t}{\lambda_m}(\sigma_m^2 + \mu_m^2) + \frac{2t}{\lambda_n}(\sigma_n^2 + \mu_n^2))$ and has a uniformly distributed direction.

Based on the above joint node mobility model and the initial status and location the link availability (i.e., the probability that there is an active link between two nodes at time $t_0 + t$, given that there is an active link between them at time t_0) between nodes m and n can be determined. If node m becomes active at time t_0 within a uniform random distance from node n, the distribution of the link availability $A_{m,n}$ between node m and node n can be approximated as follows [17]:

$$A_{m,n} \approx 1 - \Phi\left(\frac{1}{2}, 2, \frac{-4R^2}{\alpha_{m,n}}\right) \tag{2.3}$$

where $\Phi(a, b, z)$ is the Kummer-confluent hypergeometric function and R is the transmission radius of a mobile node.

If a link activates between n and m at time t_0 (due to node mobility) such that m is located at a uniform random point at distance R from n, then the distribution of link availability is given by

$$A_{m,n}(t) = \frac{1}{2}\left(1 - I_0\left(\frac{-2R^2}{\alpha_{m,n}}\right) exp\left(\frac{-2R^2}{\alpha_{m,n}}\right)\right) \tag{2.4}$$

where I_0 is a modified Bessel function of the first kind.

Based on the link availability $A_{i,j}$ for link $(i,j) \in$ path k, the path availability between node m and n at time $t_0 + t$ can be found as

$$\pi_{m,n}^k(t) = \prod_{(i,j)\in k} A_{i,j}(t_0 + t). \tag{2.5}$$

With (α, t) clustering strategy, mean cluster size increases/decreases under low/high node mobility. Lower values of α increase the probability of a node being clustered. However, at high node mobility this probability may drop significantly. The mean node residence time within a given cluster and the mean cluster survival time generally decrease with increased node mobility. The control message processing rate per node does not increase monotonically with increase in node mobility. These control messages include routing updates and those required to join and leave clusters. With increase in node mobility, the control message processing rate increases initially due to increase in topology changes and node clustering activity. However, as mobility increases further, it decreases along with the mean cluster size.

2.2.7 Access-Based Clustering Protocol (ABCP)

In an attempt to minimize the clustering overhead resulting from the control signaling overhead in a hierarchical ad hoc network the access-based clustering proposed in [21] uses MAC layer process for cluster formation. The system model assumes one time-slotted control channel which is used for the exchange of control messages and multiple data channels which are used for data transport. With an intention to form a cluster each node accesses the control channel to send a cluster head declaration frame and upon successful transmission of this frame it becomes a cluster head. A node which receives the cluster head declaration from its neighbor before it declares itself as a cluster head becomes a member node. When a node becomes a cluster head with at least one cluster member, it remains to be a cluster head until it becomes inactive.

The scheme used by the nodes to access the control channel is a three-phase multiple access (TPMA) scheme consisting of *Request to Send* phase, *Collision Report (CR)* phase, and *Receiver Available* phase. The TPMA scheme provides a distributed method for local broadcast of control messages. The simple broadcast request-response coupled with first-come-first-serve selection constitute the access-based clustering protocol (ABCP). Each node has a unique ID and can act either as an ordinary node or as a cluster head and the ABCP for these two cases are as follows [21]:

```
Begin
// This algorithm is divided into two cases:
// ordinary node case and cluster head case.
// When a node is turned on, it becomes active
// with the role of an ordinary node.

//Ordinary node case:
```

1. At the beginning, the cluster ID of the ordinary node is UNKNOWN. It tries to join in an existing cluster by sending REQ_TO_JOIN message and waits for response.

   If it receives a HELLO message from a cluster head, it sets its cluster ID to the sender's ID and sends a JOIN message confirming its membership.

   If it does not receive any response within a certain time period, it tries to become a cluster head by sending HELLO message.

2. If an ordinary node gets a DISCONNECT message from its cluster head (or the link between the node and its cluster head weakens), it sets its cluster ID to UNKNOWN and tries to become a member of another cluster.

3. An ordinary node becomes inactive simply by sending a DISCONNECT message.

//Clusterhead case:

1. When a cluster head receives a REQ_TO_JOIN message from an ordinary node, it welcomes the sender node with a HELLO message. It waits for an interval TIME_OUT_2 to receive a JOIN message. If it does not receive a JOIN message, it sends the HELLO message again.

2. A JOIN message comes from a node that may belong to the same cluster or another cluster. If it is from the same cluster, the cluster head adds the sender to the cluster head's member

```
      list. Otherwise it removes the node from the member list if
      there is any entry for that node. A sender of a HELLO or
      DISCONNECT message is also removed from the member list.

   3. A cluster head becomes inactive by sending a DISCONNECT
      message like an ordinary node.
End
```

ABCP incurs less control message overhead compared to the ID-based clustering protocol and has shorter execution time. This is because, in ABCP, the node which first transmits the HELLO message successfully becomes a cluster head and the other nodes that receive the HELLO messages become cluster member by sending JOIN messages. Therefore, each node has to send only one control message to complete the cluster initialization. For cluster maintenance, control message is exchanged between the cluster head and the cluster member. Also, since the number of control messages is independent of the number of nodes, ABCP scales well with respect to clustering overheads.

2.2.8 Power-Control-Based Clustering (PCBC)

In [18] a power-control-based two-hop clustering algorithm was proposed in which a cluster head can adjust the cluster size by exercising power control. To become a member of one or more clusters a mobile node tries to detect the pilot signal transmitted by the cluster heads. Pilot signal carry information such as node ID and transmission power level. A clustered node adjusts its transmission power based on the received pilot signal strength. If a node cannot detect pilot transmission from any cluster head, it can claim to be a new cluster head by broadcasting initializing pilot signals which are different from the normal pilot signals transmitted from a functional cluster head.

The main features of this clustering algorithm can be described as follows:

- *Initial clustering*: All the nodes send out their initialization pilot signals with maximum pilot power to acquire their neighborhood information. It uses LID algorithm to form initial cluster.

- *Cluster maintenance*: When a mobile node goes out of the clustered area, it transmits initialization pilots and attempts to become a cluster head. When a cluster head goes into a different cluster area, only one cluster head (the one with higher degree) survives.
- *Cluster head power control*: A cluster head adjusts the pilot signal level when it needs to change the size of the cluster. When necessary it also adjusts the power level so that the furthest node can hear it. If a node in the cluster reports high error rates, the cluster head increases the data transmission power level. It can adjust the power level of a node by closed loop power control.
- *Cluster member power control*: A cluster member can use both an open loop power control and a closed loop power control. If a node experiences high error rate, it reports to the cluster head so that a closed loop power control can be initiated.

Adaptive transmission power control can result in substantial energy saving. Also it can provide better channel utilization. By controlling the transmission power at the cluster heads a more adaptive network infrastructure can be achieved.

2.2.9 Channel Access-Based Clustering (CABC)

CABC [25] uses a Randomized Broadcast Channel Access (RBCA) algorithm to maximize the worst case control channel efficiency. The control channel is slotted and each slot contains a sensing period (SP) and an acknowledgement period (AP) in addition to the actual control packet period (PP).

When the node senses the channel, if no busy tone is detected, the node concludes that it has won the contention for this slot, and starts emitting PRIMARY busy tone until the end of the sensing period to notify its neighbors that it is transmitting in the slot. If the node senses the PRIMARY tone, it knows that it has lost in contention to one of its neighbors and decides not to transmit in the slot. It also starts emitting SECONDARY busy tone until the end of the sensing period.

The purpose of the SECONDARY busy tone is to notify its neighbors that it is receiving one packet in the packet period and they should refrain from transmitting another packet (which will result in collision). As a result, those nodes which have the current node as a common neighbor with the winner can avoid collision.

If the node senses that there is already a SECONDARY busy tone when it starts listening to the channel, it postpones its packet transmission and remains silent in the rest of the sensing and the packet period. If the node did not transmit in the packet period but received no packet successfully as well, it starts emitting busy tone in the acknowledgement period to notify the winner that the packet transmitted was not broadcast successfully.

If the current node was the winner and transmitted packet in the packet period, it will listen to the channel throughout the acknowledgement period. If it detects busy tone, it concludes that its packet did not get through successfully and keeps the control message in its buffer. If no busy tone is detected, the packet was broadcast properly and the control message is removed from the control buffer of the node.

The algorithm consists of the following steps:

- If the cluster ID of the node is NULL, it listens to the control channel for duration t_b to discover any nearby cluster head. If it does not receive any cluster head beacon message or contending message within this time, then it starts contending in the next slot.
- If this node contends to be a cluster head and loses contention or receives a cluster head beacon message or cluster head contending message, it sets its cluster ID to the sender of the received message.
- If this node is a member of a cluster it periodically sends a beacon message with its own ID and cluster head's ID or if the node is a cluster head it periodically (period $= t_b$) broadcasts cluster head beacon message.
- If this node is a cluster head and receives a beacon message from a node which indicates it as cluster head but is not included in the member list, then this node adds that node into its member list.
- If this node is not a cluster head and determines that it lost contact with its cluster head, it sets its cluster ID to NULL. If the node is a cluster head and has lost contact with any of its member, it removes that node from its member list. If the list becomes empty, it sets its cluster ID to NULL to join any nearby cluster.

The randomized broadcast channel access method is the heart of CABC. It can

Table 2.2. *Qualitative performance comparison among the clustering algorithms.*

Algorithm	Cluster size	Stability	Load distribution	Message complexity	Power awareness	Asynchronous/ synchronous
LCA	No control	Medium	Less fair	High	No	Syn
LID	No control	Medium	Less fair	Low	No	Asyn
HCN	No control	Low	Less fair	High	No	Asyn
MMD	Controlled	High	More fair	High	No	Asyn
LCC	No control	High	More fair	Low	No	Asyn
MBAC	Controlled	Controlled	More fair	High	No	Asyn
ABCP	No control	High	More fair	High	No	Syn
PCBC	Controlled	High	More fair	Low	Yes	Asyn
CABC	No Control	High	More fair	Low	No	Syn

quickly form stable clusters in a network with less overhead. The maintenance overhead of CABC is also considerably low in presence of node mobility.

2.3 Summary

At the beginning of the chapter, we have explained the importance of cluster in the upper layers of protocol stack. Then we presented the salient features of various non-access-based and access-based clustering algorithms. Among them, only MBAC algorithm is proactive to the node mobility, all other algorithms discussed here are reactive. Except PCBC, all the clustering algorithms use static transmission power (hence fixed transmission range) at the mobile nodes, which indicates that these algorithms are not energy efficient. The size of the clusters can be controlled in only MMD and MBAC algorithms. We compare the different clustering algorithms qualitatively in Table 2.3.

From the above discussion, we see that no single algorithm possesses the three important attributes: pro-activeness, mobility awareness, and power awareness. Moreover, there is no control over cluster size in most of the techniques. This leads to our proposed clustering framework described in the next chapter.

Chapter 3

A Framework for Mobility-Aware Proactive Low Energy (MAPLE) Clustering

As we have discussed in chapter 2, none of the existing algorithm can address the important issues of clustering. In this chapter, we propose a new framework for Mobility-Aware Proactive Low Energy (MAPLE) clustering of ad hoc networks. At first we describe the structure of the control channel and control channel access mechanism. Collision resolution and mobility tracking techniques of MAPLE are later discussed. We also examine the impact of different network parameters on MAPLE at the end of this chapter.

3.1 Channel Access and Control Message Signaling

We consider an ad hoc mobile wireless network where each node has the same maximum transmission power and each node can dynamically adjust the transmission power. Control messages in the network are transmitted through a single broadcast channel which is shared among the ad hoc nodes. The channel access time is divided into frames. A group or cluster of nodes use a particular frame to transmit information which are collectively referred to as a *control message*. The cluster heads use these frames to communicate with their corresponding cluster members.

The wireless nodes in a non-clustered (without clustering) network periodically broadcast control messages or beacons to declare their presence in the network. In MAPLE, one cluster uses one frame in each cycle (one period) to transmit control message. A cycle consists of a fixed number (F) of frames. Each frame is divided into the following parts: MCH (Message from Cluster Head), MFD (Medium Frame Delay), RCM (Reply from Cluster Member) or mini frames, SFD (Short Frame Delay), and RCH (Reply from Cluster Head) (Fig. 3.1). There may be up to M mini-frames within a frame, so that at most M RCM can be transmitted within a frame time. Here, M specifies the maximum number of cluster members that a cluster head can support and this is a network design parameter.

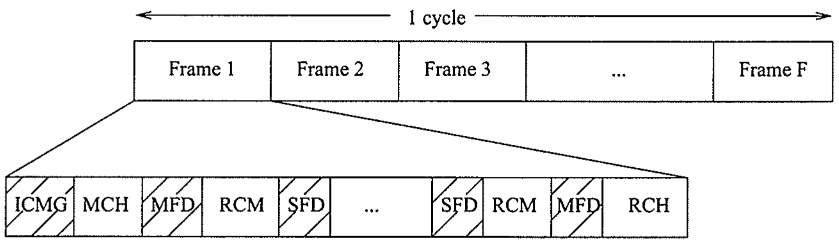


Figure 3.1. *Format of the control message frame in MAPLE clustering.*

Each cluster head reserves a particular frame to broadcast control or beacon messages. Other cluster heads which are geographically separated from this aforementioned cluster head can use the same frame, i.e., more than one cluster can use the same frame as long as their messages do not collide. There is an *inter-control message guard* (ICMG) time between two frames which absorbs the imprecision in frame timing due to propagation delay and enables synchronization.

A cluster head broadcasts MCH at the starting of its reserved frame. The nodes in that cluster (i.e., the cluster members) reply to this message in the mini frames of the same frame. In MAPLE, no extra signaling messages such as beaconing or neighborhood discovery messages are required either from the cluster heads or from the members. The MCH contains information such as message type (e.g., WELCOME, HELLO), node ID, cluster ID, transmission power level, flag for free mini-frames. A newly formed cluster head transmits a WELCOME message to announce a cluster formation. Once a cluster is formed, the cluster head broadcasts a HELLO message in each MCH. A cluster head transmits a GIVEUP message when it gives up its role as a cluster head. We do not consider HELLO as clustering overhead, because the nodes in an non-clustered network also need to broadcast those control messages. The WELCOME and GIVEUP messages are the clustering overhead. Since the nodes can transmit at different power levels, all messages specify the transmission power level as a fraction of the maximum transmission power level (i.e., the normalized transmission power level).

In the mini frame (RCM), a node provides the following information: whether it is joining, staying with or leaving its cluster head, normalized transmission power level and the available frame flag. By using the available frame flag, the members notify the status of the frames when they are idle so that the neighboring nodes can schedule their transmissions in the idle frames. Two RCMs are separated by the guard time SFD.

When a node wants to join a cluster, it transmits an RCM in one of the free mini-frames. If this RCM does not collide with other nodes' RCMs (which transmit RCMs during the same mini frame), the node can join that cluster. By using the RCH the cluster heads acknowledge the RCMs transmitted by the cluster members. The cluster head reserves the corresponding mini-frame for that node. In this way, the cluster members can communicate to the cluster head in response to the MCH through the reserved mini-frames. Due to this channel reservation technique, the number of contentions during channel access is significantly reduced. The MFDs (between MCH and the first RCM, before RCH) have two purposes: (i) to let the nodes process the received message and (ii) to report collisions if there is any.

A node that turns on, senses the medium for one cycle (i.e., for F frames) and gathers information from incoming messages. We will refer to this time as sensing period. At the end of sensing period, a node can face these three situations:

- there is one or more cluster head in the neighborhood and there are free mini-frames in which the newly turned-on node can send RCM to join a cluster,
- there is no cluster head in the neighborhood, and
- there are cluster heads in the neighboring area, but no mini-frame is free in which the new node can send an RCM.

If it receives more than one MCH from the neighboring cluster heads, it selects the cluster head which is the nearest (in terms of signal strength) and not already fully occupied by cluster members. In the last two situations, the node becomes a cluster head and sends an MCH(WELCOME) message in an idle frame.

The new node gathers information about the idle frames of its neighbors during the sensing period. An idle frame, not used by any other nodes in the neighborhood is always selected by a node when it desires to be a cluster head. The number of frames, F in a cycle can be chosen large enough to meet this criterion. Once a cluster head chooses a frame to broadcast its control message, it continues to use the same frame to transmit the control message until it suffers a collision.

The nodes which are in the transmission range of more than one cluster head are referred to as *Gateway* nodes and are used for inter-cluster data exchange. If a node can reach more than one cluster heads, it keeps information about another cluster head (referred to as auxiliary cluster head) except its own cluster head, which is nearest to it and has room to welcome a new node. If the current cluster head dies or fails, or it gives up its role as a cluster head or moves away due to mobility, the member node selects the auxiliary cluster head (identified by NextClusterHeadID) as its primary cluster head.

Since the cluster size (i.e., maximum number of nodes a cluster head can support) is defined in this framework, two cluster heads can coexist in the same region. This type of situation arises when the node density is high. However, a node may give up its role as a cluster head either when it encounters another cluster head in its neighborhood or it has only a few nodes (having alternatives cluster heads, i.e., secondary cluster heads) in its cluster. In our simulation environment when a cluster head has no member, it tries to join a neighboring cluster head.

3.2 MAPLE Clustering Algorithm

Each mobile node keeps the following information to participate in the cluster formation process: *ClusterHeadID* (for current cluster head) and *NextClusterHeadID* (for auxiliary cluster head), a *NodeTimer* and *busy frame flag*. The timer is updated after each frame. By using the *busy frame flag*, the nodes keep track of the busy and idle frames.

The following three modules run on each mobile node:

```
// To transmit MCH message (after sensing period)

 for each node
    if (NodeStatus == ClusterHead) and
        (NextEventTime == NodeTimer)
       check whether to give up cluster head role
       if (giving up is feasible)
         broadcast GIVEUP message
        else
         broadcast HELLO/UPDATE message
       endif
     else if (NodeStatus == UNKNOWN) and
          (NextClusterID == UNKNOWN)
        declare itself a cluster head
        broadcast WELCOME message
    endif
 endfor


 on receiveing an MCH message
 begin
    if (collision occurs) then
        send collision report in first MFD
    elseif (message type == GIVEUP) and
            (sender == cluster head)
       set node status to UNKNOWN
    else
       set busy frame flag
       gather information to select auxiliary cluster head
       if (NodeStatus == UNKNOWN) and
```

```
          (sender == NextClusterHeadID)
         set cluster head and
         set node status to cluster member
         send JOIN message
       elseif (sender == NextClusterHeadID) and
          (it is posible to minimize transmit power)
         migrate to next cluster
       elseif (NodeStatus == ClusterMember)
                 and (sender == ClusterHead)
         send UPDATE message
       elseif (NodeStatus == ClusterMember)
                 and (sender <> ClusterHead)
         set node status as GateWay
       endif
    end if
end


on receiving a RCM message begin
   if (collision in mini-frames)
     send report in the second MFD
   else
     update info
     if (sender is another  cluster head)
        and (NodeStatus == ClusterMember)
       set  GateWay
     end if
   end if
end
```

The flowchart of MAPLE clustering algorithm is given in (Fig. 3.2):

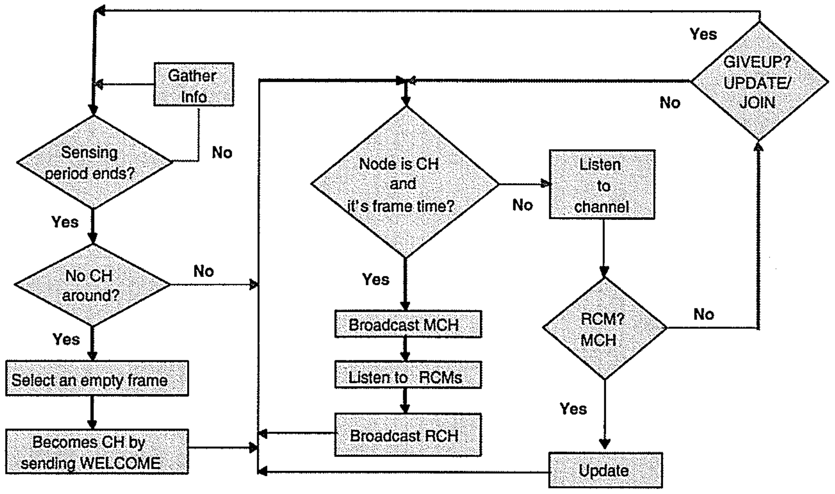


Figure 3.2. *Flowchart of MAPLE clustering algorithm.*

3.3 Collision Resolution

Collisions among the control messages may take place under two possible scenarios which we refer to as *direct* and *indirect*. A *direct* collision occurs when a member from one cluster comes closer to the cluster head of another cluster and the control messages for both the clusters have the same frame timing. Therefore, the control messages collide at this node (Fig. 3.3(a)). *Indirect* collision occurs when control messages corresponding to two or more clusters with same frame timing collide at a node in a neighboring third cluster (Fig. 3.3(b)). Note that, both these type of collisions will occur after cluster formation, and therefore, affect the overhead required for cluster maintenance.

If a cluster ('tagged' cluster) is surrounded by x other clusters in its neighborhood, assuming that the selection of frame timing is done independently among the cluster heads, the probability that k clusters among them choose the same frame time is given by

$$P_f(k) = \binom{x}{k} \left(\frac{1}{F}\right)^k \left(1 - \frac{1}{F}\right)^{x-k}. \tag{3.1}$$

When $k \geq 2$, RCM messages from nodes in different clusters may collide at a node in the 'tagged' cluster. Due to this collision, the node in the 'tagged' cluster will not be able to update the information about the neighboring nodes during collision.

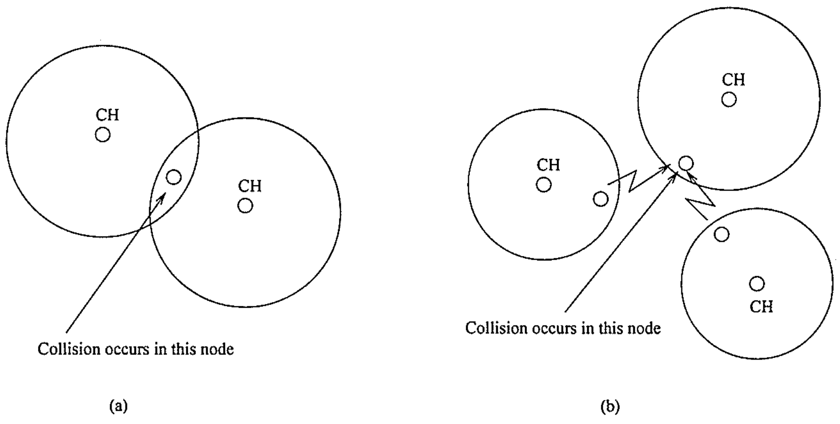


Figure 3.3. Direct *and* indirect *collision scenarios.*

We consider a special case when $k = 2$. In this case, the probability that the collision (among the RCM messages) occur in i mini-frames is

$$P^k_{mini}(i) = \frac{\begin{pmatrix} M \\ y \end{pmatrix}\begin{pmatrix} M-y \\ z-i \end{pmatrix}\begin{pmatrix} y \\ i \end{pmatrix}}{\begin{pmatrix} M \\ y \end{pmatrix}\begin{pmatrix} M \\ z \end{pmatrix}} P_f(k), \quad k=2 \tag{3.2}$$

where y and z are the number of nodes that are in the neighborhood of the node experiencing collision and M is the number of mini-frames.

We numerically evaluate $P^k_{mini}(i)$ for different values of y, z and i (Fig. 3.4). With $F = 64$, $M = 16$, $x = 6$, $k = 2$, $y = 4$, and $z = 4$, it is observed that $P_f(k) = 0.3\%$ and $P^k_{mini}(1) = 0.14\%$. With higher values of k and i, the probability P^k_{mini} will be even smaller.

In case of direct collision, one of the clusters needs to change its frame timing. The node, at which the collision occurs, transmits the collision report in the first MFD and also informs its cluster head of the collision through the RCM. Although the cluster messages collide, the node knows the timing of the mini-frame in which it can communicate with the corresponding cluster head. The other cluster head that receives the collision report but does not receive the RCM reporting the collision, changes its channel access timing.

Indirect collision can be resolved in either of two ways – by reallocating the mini-frames among the nodes in one of the clusters or by changing the frame timing of

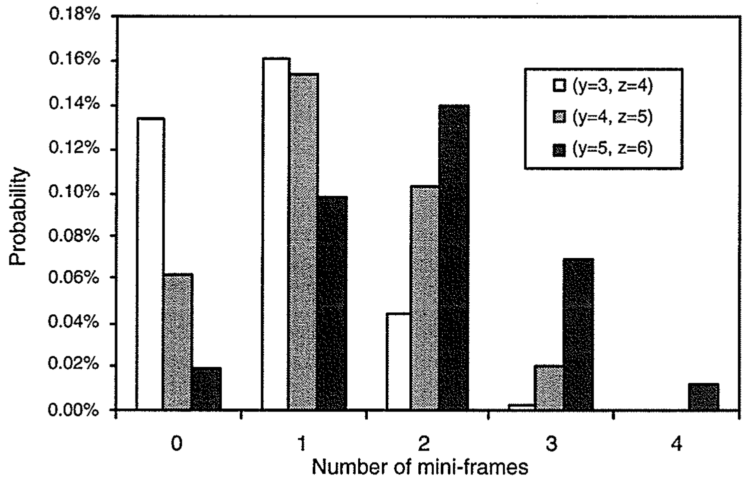


Figure 3.4. *Probability of collision in a mini-frame (for $F = 64$, $N = 16$, $x = 6$, $k = 2$).*

the colliding clusters. For reallocation of the mini-frames, if a cluster member detects collision, it can attempt to retransmit RCM during another mini-frame. If the number of mini-frames in which collision occurs (among RCMs) is observed to be more than half of the empty mini-frames, the node can request its cluster head to change the frame timing.

3.4 Mobility Tracking Using Received Signal Strength

Suppose node **B** transmits to node **A** using normalized transmission power r ($0 < r \leq 1$). Let the received signal strength at node **A** at time t_1 and t_2 be S_1 and S_2, respectively, and let S_{th} be the minimum required signal power at the receiver for successful decoding of the message. Let the distance between node **A** and node **B** at these time instants be d_1 and d_2, respectively.

Now, assuming that the distance-dependent path-loss is the major source of signal attenuation, we obtain

$$\frac{S_{th}}{S} \approx \frac{\frac{R_{max}}{d^{\delta}_{max}}}{\frac{rR_{max}}{d^{\delta}}} = \frac{d^{\delta}}{rd^{\delta}_{max}} \tag{3.3}$$

where δ is the path-loss exponent, d_{max} is the maximum transmission range, R_{max} is the transmission power required to successfully transmit a message at a distance d_{max} and S is the received signal strength at a distance d. Therefore, the values of d_1 and d_2 can be estimated as follows:

$$d_1 = \left(\frac{rS_{th}}{S_1}\right)^{\frac{1}{\delta}} \quad and \quad d_2 = \left(\frac{rS_{th}}{S_2}\right)^{\frac{1}{\delta}}. \tag{3.4}$$

Now, using the current estimate of the normalized radial distance from the cluster head d_i (as given by (3.4)), a node predicts its next position as follows:

$$\begin{aligned} d_{i+1} &= d_i + \alpha_i \times T \\ \alpha_{i+1} &= (1-\beta) \times (d_{i+1} - d_i) + \beta \times \alpha_i \end{aligned} \tag{3.5}$$

where β is the memory variable used to keep history of the mobility pattern of the node, T = cycle time and $\alpha_0 = 1 - d_0$. The value of β is assumed to be 0.4 in this analysis. A node can use this prediction to make a proactive decision. For example, if a node predicts that in the next cycle it would be out of the current cluster, then it joins another cluster, if there is any, otherwise, it becomes a cluster head.

3.5 Relationship Between N and F

The definitions of some of the parameters used in this section are as follows: T = duration of one cycle consisting of F frames, M = maximum number of member nodes supported by a cluster head, a = percentage of frames that can be used for clustering, C = total number of clusters, h = maximum number of cycles that a wireless node can wait to join a cluster, n = average number of new nodes that try to join a cluster in one cycle (this is caused, for example, due to the mobility of the nodes), m = average number of nodes in a cluster (i.e., total number of nodes/total number of clusters), N = maximum number of nodes that can be supported (i.e., the maximum number of nodes that can be placed in a region where each node is within the range of others transmission radii. All of these N nodes share the same physical wireless channel).

Now consider a control message in which s mini-frames are empty (i.e., there are already $M - s$ cluster members). Suppose there are n new nodes trying to join a cluster in each cycle (the cluster heads can keep track of this value) and these nodes send reply to the control message in the s free mini-frames to join the cluster. Since the free mini-frames are selected randomly, some of the replies (from different nodes) would collide in the mini-frames. The nodes which can successfully transmit in the mini-frames become cluster members.

For simplicity, if we assume that a new node chooses a free mini-frame with equal probability and the node tries to join a particular cluster, then the probability (p) that a new node can join a cluster successfully is, $p = (1 - \frac{1}{s})^{n-1}$ and the average number of attempts required is given by $1/p$. In case of low mobility of the nodes, the value of n is smaller, therefore, the value of p becomes higher. The higher the value of p the lower is the delay in joining a cluster.

According to the definition of h, $\frac{1}{p} \leq h$, from which it is easy to find that

$$M - m \geq g \tag{3.6}$$

where $g = \frac{1}{1-(\frac{1}{h})^{\frac{1}{n-1}}}$. Our aim is to find the maximum number of nodes that can be supported for some specific values of F, N, a and M. To support maximum number of nodes, maximum number clusters should be formed. We can say that $a \times F$ frames are used in this situation. Assuming uniform cluster sizes, we can define m as follows:

$$m = \frac{N - C}{C} = \frac{N - aF}{aF} = \frac{N}{aF} - 1. \tag{3.7}$$

Therefore, using (3.6), the relationship among N, M, F, a and h can be found to be

$$N \leq aF(M - g + 1). \tag{3.8}$$

Typical results on the variations in maximum value of N with n for different values of h are shown in Fig. 3.5. When the value of n is quite small (e.g., when the nodes are relatively static), the maximum value of N becomes almost independent of h. As expected, for a particular value for the maximum cluster joining delay, the maximum value of N decreases with increasing n. Note that, (3.8) would be useful to control the network size when the cluster joining delay is to be bounded.

3.6 Evaluation of Mini-Frame Access Performance

In wireless networks with mobile nodes, the distance between the receiver and senders may vary significantly. The data packets from different senders arrive at the receiver

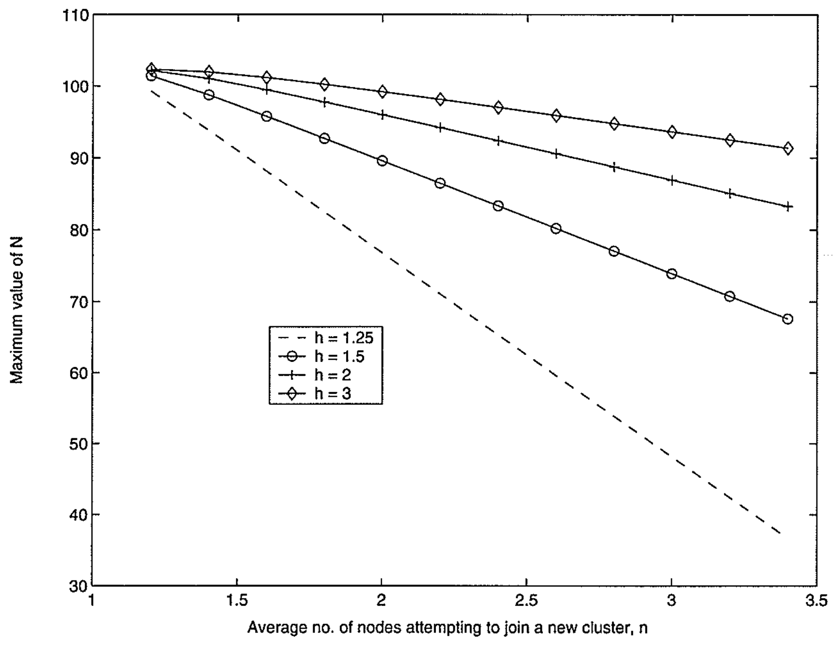


Figure 3.5. *Maximum number of mobile nodes under bounded cluster joining delay (for $F = 32$, $M = 16$, $a = 10\%$).*

with different power level due to shadow fading and Rayleigh fading. The packet with higher power level has a good chance of being received correctly at the receiver even when the packets arrive at the same time frame at the receiver ([28]-[29]). This phenomenon is known as *capture effect.* If we consider capture effect, the probability of successful transmission of data packets increases greatly.

In ad hoc networks, due to mobility some of the wireless nodes move out of their cluster and try to join another neighboring cluster. As a result, during a control frame time, a cluster head finds that some of its existing members move away from its cluster and some new nodes want to join in. The number of nodes that move in or out, depends on the average speed as well as transmission power of the nodes in the network. The higher the speed, the more number of nodes move in or out in each cycle. The incoming nodes select different empty mini-frames in the control frame randomly. If more than one nodes select the same mini-frame at the same time, then collision occurs. If we do not consider capture effect, all of the contending nodes fail to join cluster in that frame. But if the capture effect is considered, there is a good chance that the cluster head receives or detects a data packet from one of the transmitting nodes, thus one node becomes member of that cluster. We are interested in analyzing the mini-frame access with capture and without capture.

At first we will observe the arrival and departure patterns of mobile wireless nodes to cluster heads in each of their control frame for different node speeds. We then compare the arrival and departure patterns in presence of capture and without capture. Finally, we examine the impact of capture by comparing the probability of rejection by the cluster heads and average number of attempts by the ordinary nodes to join a cluster head. The simulation environment is same as we used for comparing access based clustering algorithms. To integrate mobility, we used one-step Markov path model [16]. The simulation area was 1000 $m \times 1000$ m. We put 300 wireless nodes with 100 m transmission range in the simulation area. The speeds of the nodes were varied 5 m/s to 40 m/s.

- *Arrival and Departure Probability:*

  We kept the frequency of arrival and departure of different number of nodes to each cluster head in each of their control frames. For example, if n_i is the frequency of arriving i number of nodes then the probability of arriving i number of nodes in each control frame is $\frac{n_i}{\sum n_i}$. The calculation is the same for departure patterns. Since the probabilities of arriving and departing higher number of nodes are very low we sum up those probabilities at the tail (Fig. 3.6). We see from Fig. 3.6 that for probability of arriving one node is always higher. As the speed goes up, the probabilities of arriving more than one node increase. When the node speed is less than 20 m/s, the probability that the arriving batch size will be greater than 3 is negligible. At speed 40 m/s, the probability that the arriving batch size is greater than 4 is about 4.5%. The patterns of departure process is same as arrival process as shown in Fig. 3.7. As the node speed increases the batch size of the departing nodes increases. The probability of departing one node is always higher. Since only a portion of the arriving nodes are accepted by the cluster heads, the departure probabilities in Fig. 3.7 are always lower than those in Fig. 3.6.

- *Comparison of Arrival and Departure Probability:*

  To compare the arrival and departure patterns in the both situations (with and

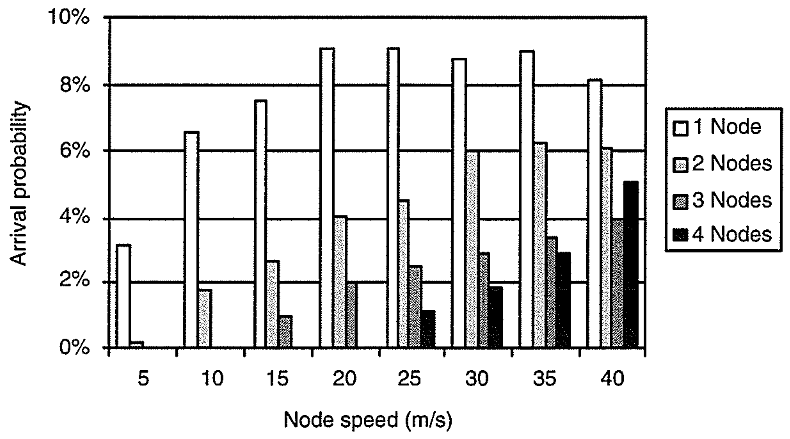


Figure 3.6. *Probability of node arrival in the clusters at different speed*

without capture), we examined the arrival and departure patterns at same node speed of 20 m/s. The arrival and departure patterns are given in Figs. 3.8- 3.9, respectively.

Since the probability of rejection in without capture is higher as we will see in the next section, more nodes try to join in a cluster in each frame time. As a result, the probabilities of arrival for different batch size is higher if we do not consider capture effect. Once the nodes join a cluster, the provability of departing the cluster depends solely on the mobility pattern of the nodes. Therefore, for departure process the probabilities remain more or less same for all batch sizes.

- *Rejection Probability:*

When a node tries to join a cluster, it may succeed in joining that cluster or it may fail to join if collision occurs. We measured the probability of rejection, $Prob_{rej}$, by dividing the number of failures to join a cluster by number of total attempts made by the nodes to join a cluster. If we do not consider capture effect and collision occurs in a mini frame time, all the contending nodes fail to join the cluster i.e., are rejected by the cluster head. But one of them becomes successful and accepted by the cluster head if we consider capture effect.

Typical variations in the rejection probability are shown in Fig. 3.10 for both

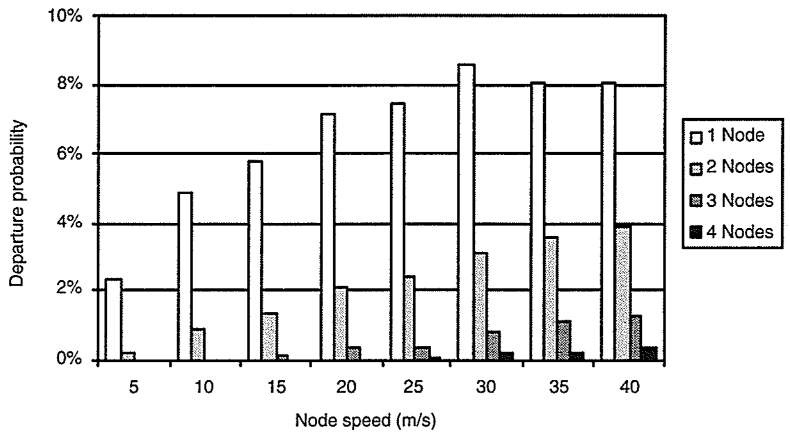


Figure 3.7. *Probability of node departure from the clusters at different speed*

situations. As expected, rejection probability is higher Without Capture.

The probability of success in joining a cluster is $1 - Prob_{rej}$. Now, we can define the average number of attempts to join a cluster by $\frac{1}{1-Prob_{rej}}$. The trend is shown in Fig. 3.11.

The goal of this entire analysis was to see whether the ordinary nodes in the ad hoc networks can join an existing cluster head immediately after it is disconnected from another cluster head. If the ordinary nodes spend much time in contending to join a cluster, the throughput performance of data transmission will be very poor.

In Fig. 3.11, we see the number of attempts to join a cluster about 1 for moderate speeds. At node speed 72 Kmph, the average attempts for each nodes is around 1.25 which is tolerable. Another observation is that the probability of rejection in absence of capture is higher. So the performance is improved if capture is used.

3.7 Summary

We have presented our proposed clustering algorithm in this chapter. We have shown the relationship between N and F and the performance of mini-frame access by

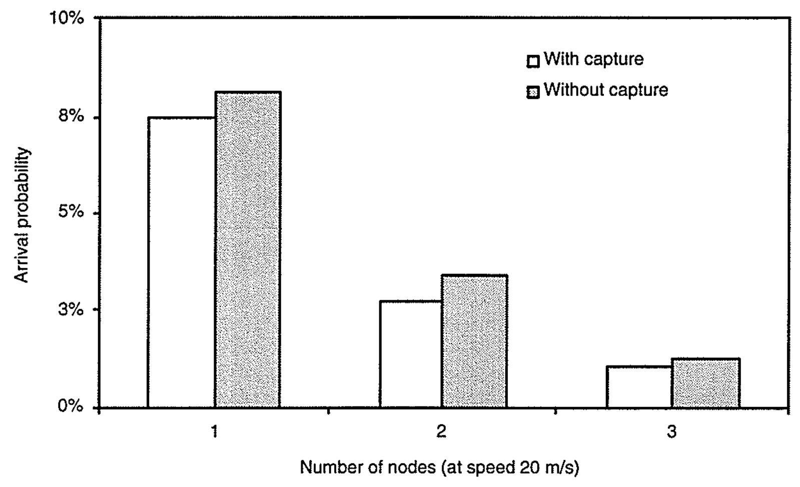


Figure 3.8. *Arrival probability with capture and without capture (for node speed 20 m/s).*

the wireless nodes through simulation. We see that the number of supported node by MAPLE decreases with number of new node arrival in each control frame. It indicates that in case of higher mobility the number of supported nodes decreases. it further implies that at high mobility MAPLE constructs small sized cluster. In MAPLE we have mini-frames in each control frame with which the cluster members are attached. We have shown that accessing the mini-frames will not be a bottleneck in MAPLE as the rejection probability is adequately lower in presence of capture and without capture.

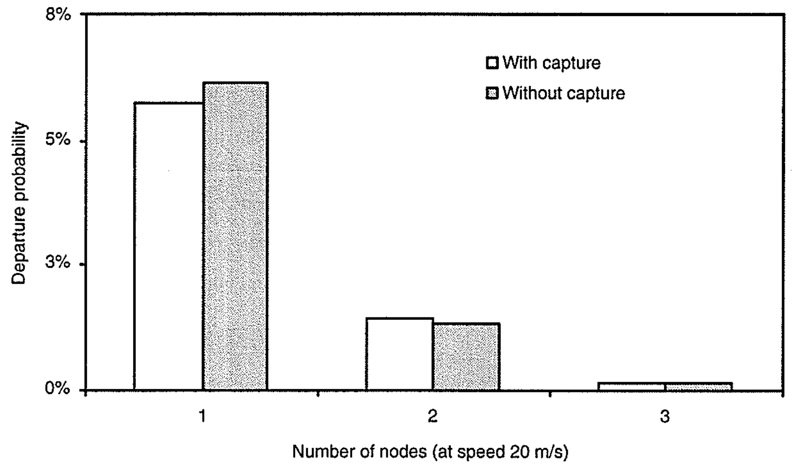


Figure 3.9. *Departure probability with capture and without capture (for node speed 20 m/s).*

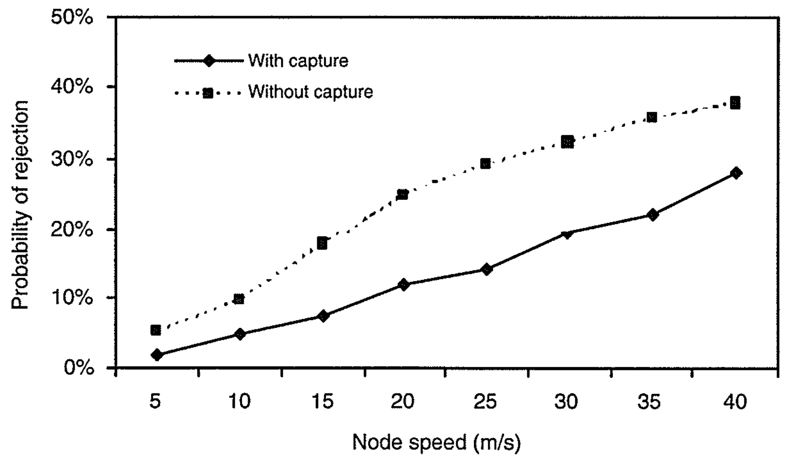


Figure 3.10. *Probability of rejection with capture and without capture for different node speed.*

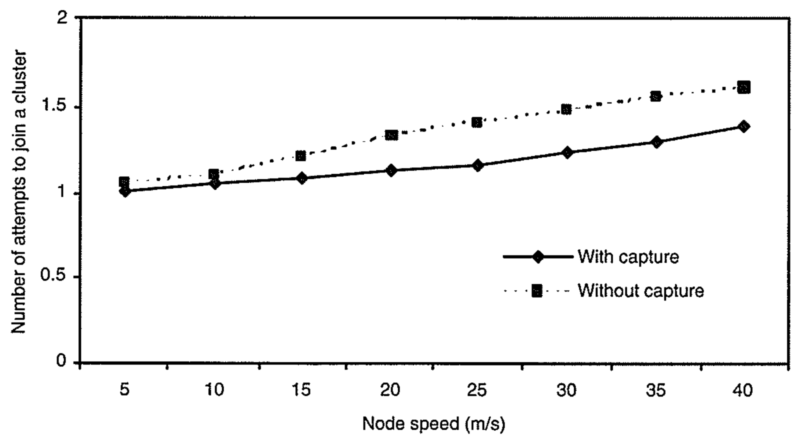


Figure 3.11. *Average number of attempts taken by the nodes at different node speed with capture and without capture.*

Chapter 4

Performance Comparison with Non-Access-Based Clustering

In chapter 2, we discussed various clustering techniques proposed in the literature. This chapter presents the performance of mainly three non-access-based clustering techniques in section 4.2. Our goal is to select one potential non-access-based clustering algorithm, with which we compare MAPLE. In section 4.3, we compare the performance of MAPLE with LCC-LID.

Before going to performance comparison, we present some metrics by which we can compare the performance of different clustering techniques.

4.1 Performance Metrics

It is desirable that the cluster configuration does not change rapidly when only few nodes are moving and the topology is slowly changing. For more stable clustering the mobility information of the mobile nodes should be exploited. For example, a node with high mobility should not be chosen as a cluster head. Rather it can be treated as a single node cluster.

Also, since cluster heads involve more computations they deplete the battery power more rapidly compared to the other mobile nodes, therefore, during selection of cluster heads, a clustering algorithm should give preference to nodes which have spent lesser amount of time being cluster heads. This will result in a more uniform cluster head load distribution among the nodes.

Since most portable devices are powered by batteries with very limited life time, energy efficiency is one of the biggest factors that need to be considered for designing ad hoc network protocols.

As mentioned earlier, a mobile node needs to broadcast its new cluster information to its neighbors periodically. Therefore, the clustering algorithm should try to minimize the overhead due to control signaling.

The stability, the control signaling overhead involved in maintaining the clusters, and the load distribution among the nodes primarily determine the performance of a clustering algorithm. Generally the following metrics have been used in the literature to evaluate the performance of a clustering algorithm:

- *Number of cluster head changes*: If a cluster head moves out of a cluster, a new cluster head needs to be selected. Frequent changes of cluster head may incur significant control overhead.
  Let $y_i(t) = \{1, 0\}$ denote whether node i is a cluster head or not at time t. Then the instability of node i as a cluster head during a last period of time T can be measured as the number of times the node changes its role between a cluster head and a cluster member as follows [26]:

$$z_i(t) = \frac{1}{T}\sum_{k=1}^{n_T} |y_i\,(t-(k-1)\Delta t) - y_i\,(t-k\Delta t)\,| \tag{4.1}$$

  where $n_T = \frac{T}{\Delta t}$. Then the stability, $s_i(t)$ of node i at time t can be determined as follows:

$$s_i(t) = \exp\{-z_i(t)\}. \tag{4.2}$$

  The higher the value of $s_i(t)$ $(0 \leq s_i(t) \leq 1)$, the better is the stability.

- *Number of cluster membership changes*: Each time a node changes its cluster, it has to broadcast this change. Therefore, overhead due to control signaling increases with increasing number of cluster membership changes.

- *Number of link failures among the mobile nodes within a cluster*: Link failures within a cluster generally trigger re-clustering which adversely affects system performance. The number of link failures within a cluster can be reduced by exploiting the mobility information of the nodes in a proactive manner during clustering.

- *Load distribution*: If $q_i(t)$ $(0 \leq q_i(t) \leq 1)$ is the fraction of time a node remains a cluster head and $g(t)$ $(0 \leq g(t) \leq 1)$ is the granularity of cluster heads of the system defined as
$$g(t) = \frac{1}{N}\sum_{i=1}^{N} q_i(t) \tag{4.3}$$
the cluster head load distribution $d(t)$ $(0 \leq d(t) \leq 1)$ of the system can be defined as follows [26]
$$d(t) = 1 - \sqrt{\frac{1}{N}\sum_{i=1}^{N} (q_i(t) - g(t))^2}. \tag{4.4}$$
A value of $d(t)$ close to 1 indicates that the load distribution among the nodes is more fair over a certain period of time.
- *Impact of transmission power and node density*: It is also important to observe the impacts of mobile nodes' transmission power (and hence transmission range) and node density in the network on the performance behavior of a clustering algorithm.

4.2 Performance Analysis of LID, HCN, and LCC

To observe the impacts of transmission range, node density and node mobility on the general clustering performance we evaluate the performances of the LID, HCN and LCC (LCC-LID and LCC-HCN) algorithms in a unified simulation framework. Since the quantitative performances of MBAC, ABCP and PCBC are more dependent on the specific signaling procedure and/or mobility patterns of the nodes, we have not included these algorithms in our simulation framework.

Total area of simulation is assumed to be 1000 m × 1000 m and the total number of nodes is varied from 100 to 800. During the course of a simulation the transmission range of each node is kept fixed; however, it is varied from 25 m to 500 m to examine the effects of variable transmission range on the general clustering performance. The mean speed of all nodes is varied from 1 m/s to 10 m/s (i.e., 3.6 *kmph* to 36 *kmph*). The values of *minspeed*, *mindir*, *maxdir* are assumed to be 0, 0, 2π, respectively. The value of *maxspeed* for each node is generated as an exponential random variable around the mean speed. The interval of each epoch is 0.25 *sec.* If a node crossed the simulation boundary due to mobility, it is bounced back to the simulation area.

- Effect of Transmission Range: For all the four techniques, the average cluster size increases as the transmission ranges of the nodes increase. The average cluster sizes for the least cluster change (LCC) algorithms are higher than those for normal LID and HCN algorithms (Fig. 4.1). Compared to the LID-based algorithms, the average cluster size is observed to be higher for HCN-based algorithms. For inter-cluster routing longer transmission range (and hence larger cluster size) helps in minimizing the delay involved in end-to-end transmission. However, as the number of nodes in a cluster increases the cluster members have to maintain comparatively longer look up tables for intra-cluster routing. Moreover, if the cluster size increases the workload on the cluster head also increases which may not be desirable. In such a case, a significant overhead will be involved in changing cluster heads.

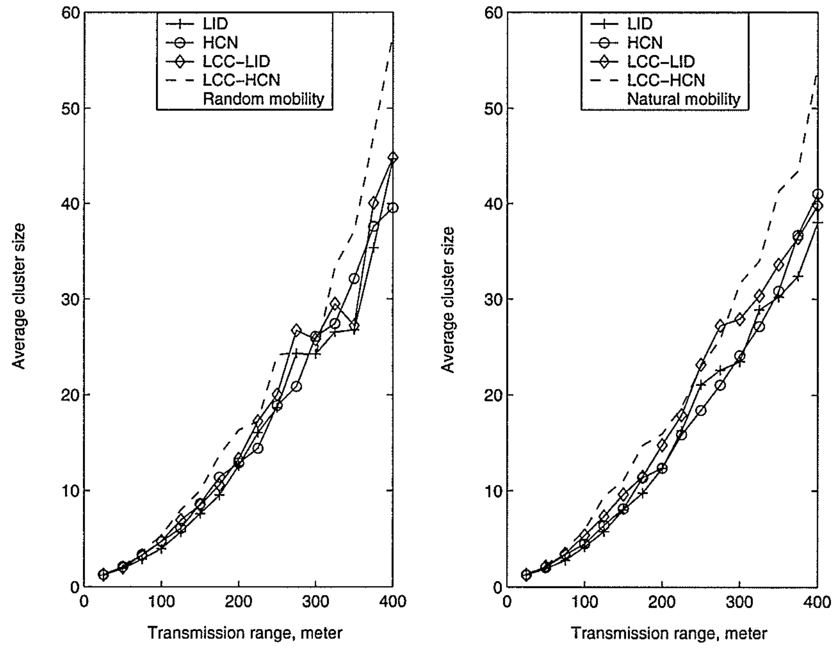


Figure 4.1. *Impact of transmission range on average cluster size (for N = 250, average node speed = 2.5 m/s).*

Compared to the pure LID and HCN-based algorithms, both of the LCC algorithms provide better stability across different transmission ranges (Fig. 4.2). Since the LCC algorithm is designed to reduce the number of the cluster head changes, the average duration for which a node remains cluster head is higher for the LCC algorithms. The normal HCN algorithm is observed to provide the worst stability. The simulation results reveal that the stability decreases up to a certain transmission range then it again increases.

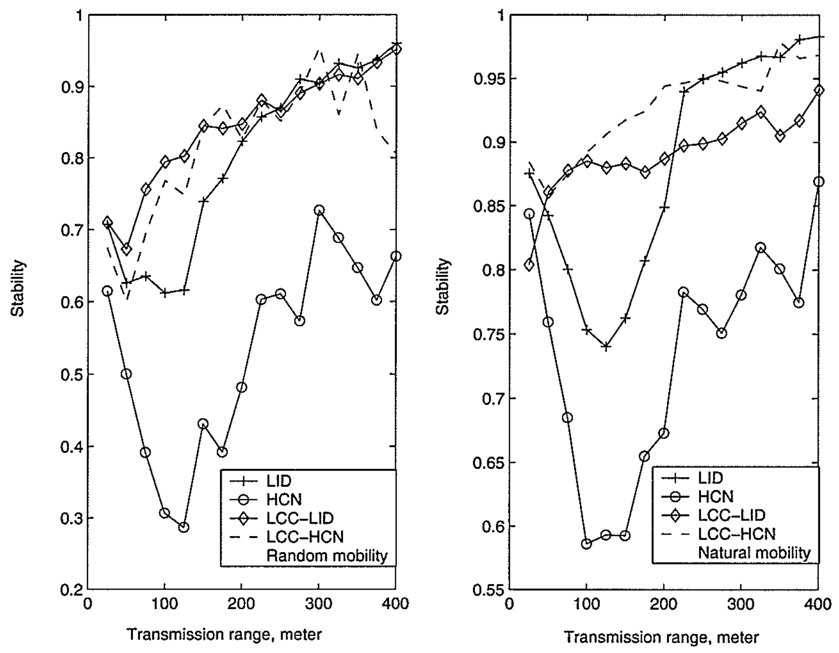


Figure 4.2. *Impact of transmission range on cluster stability (for N = 250, average node speed = 2.5 m/s).*

As the transmission range increases, the probability that a mobile node experiences change in its neighborhood increases. As a result, more cluster head changes take place for normal LID and HCN algorithms. But the LCC is immune to neighborhood changes unless a node becomes disconnected from its cluster head. Therefore, in case of LCC the stability increases as the transmission range increases. Note that, at very low transmission range the cluster size is very low especially if the node density is not very high. Small clusters experience higher stability and for this reason we observe that for all the algorithms stability is high at very low transmission ranges. Better stability is observed for the natural mobility model compared to that for the random mobility model. For all the algorithms, as the transmission range increases, the load distribution metric increases except for very small transmission range. Actually at very low transmission range, the cluster size is very small (about one). Therefore, in general, as the range increases, the load distribution becomes more fair (Fig. 4.3). The HCN based algorithms perform better since they are not biased by node ID. In LID or LCA, node ID rather than the neighborhood change biases the cluster head selection, and therefore, the load distribution may become more uneven.

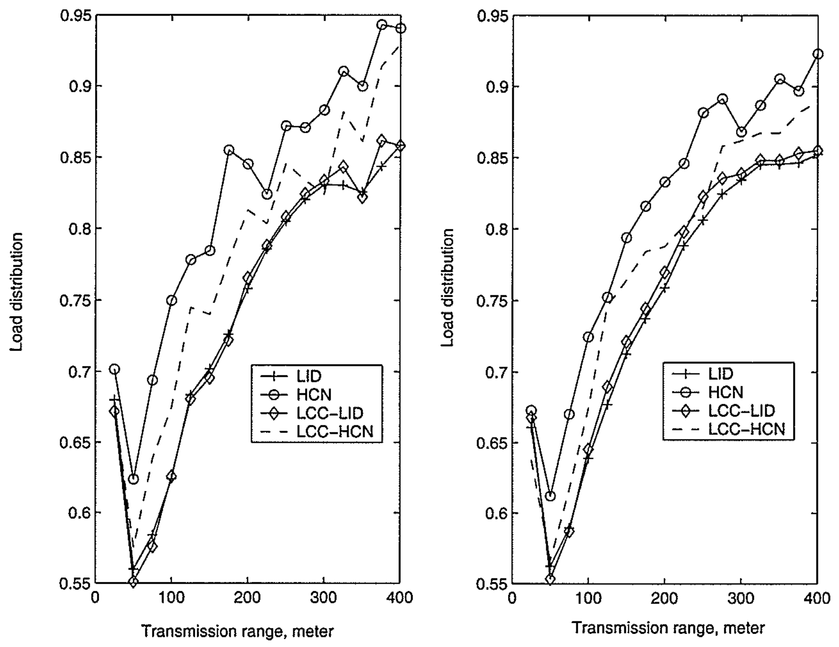


Figure 4.3. *Impact of transmission range on load distribution (for $N = 250$, average node speed = 2.5 m/s).*

It is observed that the HCN based LCC algorithm performs better in terms of both stability and load distribution. Better stability and load distribution are achieved with longer transmission range. However, longer transmission range has adverse effect on power consumption and may cause more interference.

- Effect of Node Density:
  As the node density increases, the average cluster size increases. It is observed

that the average cluster size is the largest for LCC-HCN algorithm while it is the smallest for LID algorithm (Fig. 4.4). Also, with normal LCC or HCN algorithms, the stability of the clusters decreases quite rapidly as node density increases. With higher node density, changes in neighborhood of the mobile nodes take place more frequently and since the connectivity of a node is very sensitive to the neighborhood changes, normal HCN algorithms perform the worst in stability measure as the node density increases (Fig. 4.5). The stability measures are observed to be higher for the natural mobility case compared to those in the case of random mobility (Fig. 4.5).

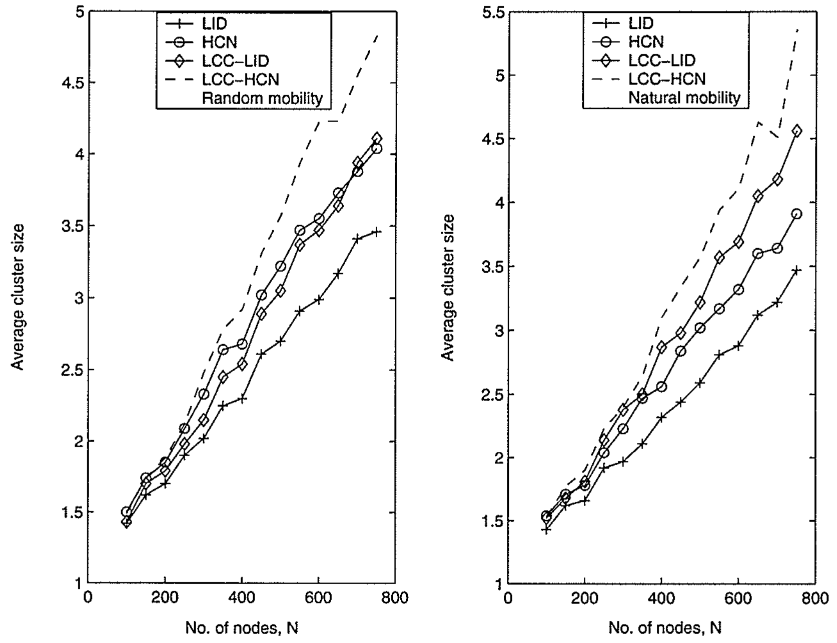


Figure 4.4. *Impact of node density on average cluster size (for transmission range = 50 m, average node speed = 2.5 m/s).*

Generally, the load distribution becomes more fair as node density increases. Since with the HCN-based algorithms more cluster head changes take place with increasing node density, the load distribution is observed to be the best for these algorithms (Fig. 4.6). The LID based LCC algorithm performs the worst due to fewer number of cluster head changes and their biasness towards lowest-ID nodes in selecting cluster heads. The load distribution measures are

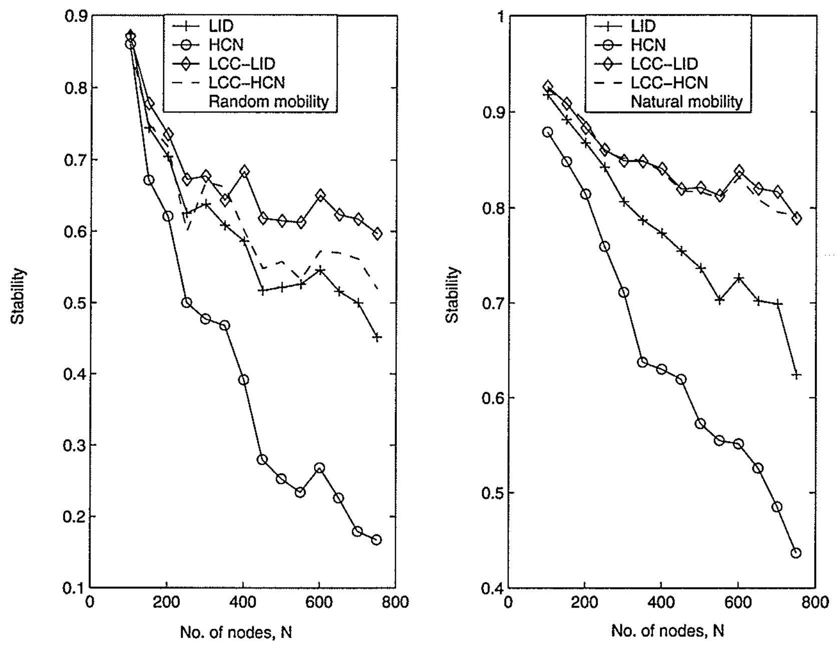


Figure 4.5. *Impact of node density on cluster stability (for transmission range = 50 m, average node speed = 2.5 m/s).*

observed to be lower for the natural mobility case compared to those in the case of random mobility.

- Effect of Node Mobility: In general, under different node speeds the HCN-based algorithms are observed to result in larger clusters compared to those due to the LID-based algorithms (Fig. 4.7). Again, LCC-based algorithms result in larger cluster sizes compared to the normal LID or HCN-based algorithms.

  As the mobility/speed of the nodes increases, stability of the clusters decreases in general. The LID-based algorithms provide better stability compared to the HCN-based algorithms as user mobility increases (Fig. 4.8). The performance trend for load distribution is just opposite to that for stability. The HCN-based algorithms provide a more fair load distribution (Fig. 4.9). The HCN-LCC algorithm seems to be a good choice if we consider both the stability and load distribution metrics.

The LCC algorithm is developed to minimize the cluster head changes and is known to be the most stable two-hop clustering algorithm. With the LID-based

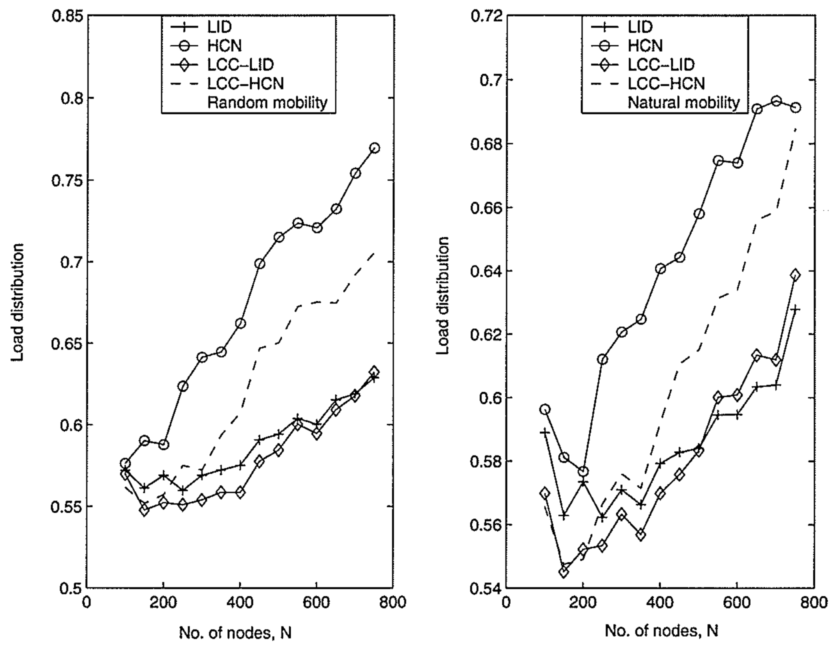


Figure 4.6. *Impact of node density on load distribution (for transmission range = 50 m, average node speed = 2.5 m/s).*

clustering, the cluster head load is not uniformly distributed among all the nodes. The lower the node ID the more likely it is for the node to become a cluster head. Since ordering of the node ID plays an important role in this approach, the number of cluster heads in a network varies with the distribution of the node ID. The highest connectivity mechanism aims at reducing the number of clusters at a given time by favoring nodes with largest number of neighbors when it comes to electing cluster heads. As the connectivity of a node may change rapidly as it moves, the connectivity-based algorithm also tends to be less stable.

One of the major limitations of all the above algorithms is that, none of these attempts to use the node mobility information for clustering. In case of link failure due to mobility, the clusters are recomputed for the entire (or some part) of the network.

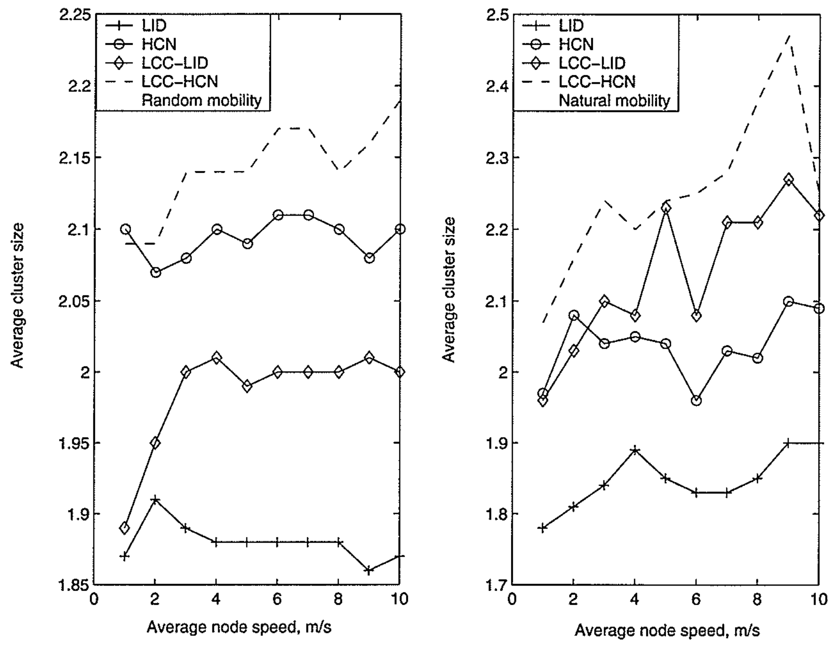


Figure 4.7. *Impact of average node speed on average cluster size (for N = 50, transmission range = 50 m).*

4.3 Performance Comparison Between MAPLE Clustering and LCC-LID-Based Clustering

We compare the performance of MAPLE clustering with that of LCC-LID clustering which is a non-access-based clustering algorithm. As we have discussed above, LCC-LID combines the good features of both LID and LCC-based clustering.

- *Control Message Overhead:*

  The performance of MAPLE clustering in terms of the average number of control messages per node (during the simulation period) is illustrated through Fig. 4.10. In LCC-LID, a mobile node needs to negotiate with its neighboring nodes to form a cluster. Therefore, for LCC-LID, whenever a cluster is formed, the number of control messages is measured to be equal to the number of one-hop neighbors.

  As is evident from Fig. 4.10, control message overhead for MAPLE clustering is significantly lower compared to that for LCC-LID-based clustering. This

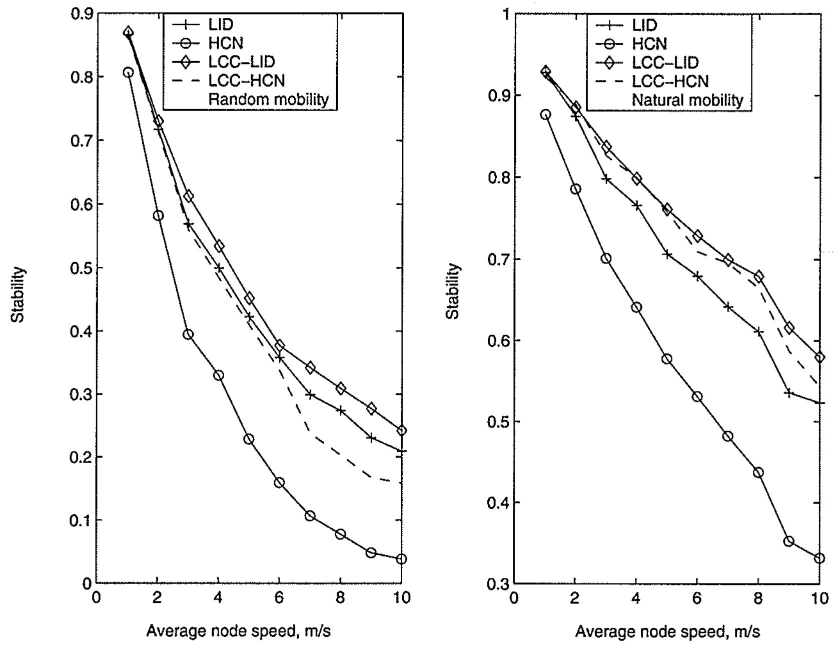


Figure 4.8. *Impact of average node speed on cluster stability (for N = 50, transmission range = 50 m).*

is due to the fact that, in LCC-LID-based clustering a node needs to receive control messages from all the neighboring nodes before it can decide on its role (i.e., whether to become cluster head or cluster member). Then the node needs to transmit the status message to all of its neighbors. However, in MAPLE clustering, a node senses the medium for one cycle and if it does not receive any message from any existing cluster head, it transmits a message. If it receives a message from a cluster head, it joins the cluster. In MAPLE clustering, a node can become a cluster head only by transmitting an MCH(WELCOME) and a node can join a cluster by transmitting an RCM(JOIN).

Also, with MAPLE clustering, as the node density increases, unlike the LCC-LID-based clustering, the control message overhead does not increase significantly. Therefore, MAPLE clustering can be considered to be more *scalable* than LCC-LID-based clustering.

The control message overhead increases with increasing node speed. However, the rate of increase in control overhead is observed to be lower in MAPLE

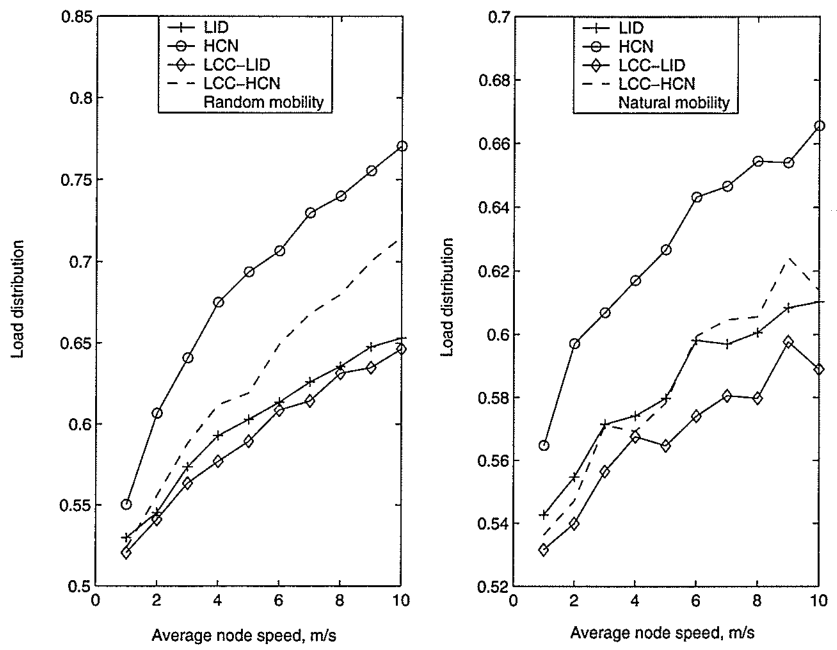


Figure 4.9. *Impact of average node speed on load distribution (for N = 50, transmission range = 50 m).*

clustering compared to that for LCC-LID-based clustering (Fig. 4.11).

- *Average Number of Link Failures:*

  A link failure occurs when a node moves out of its cluster and the link between the node and its cluster head breaks. A link failure can be prevented if a node joins another cluster before the link between the node and the corresponding cluster head breaks. With MAPLE clustering, the average number of link failures is smaller compared to that for LCC-LID clustering (Fig. 4.12). In LCC-LID-based clustering there is no mechanism to predict the link failures and a mobile node joins another cluster only after it experiences a link failure. MAPLE clustering, on the other hand, reduces the number of link failures due to its proactive nature. That is, a node predicts a link failure beforehand and joins another cluster by leaving the present one. As the mean node speed increases, the average number of link failures per node also increases (Fig. 4.13).

- *Stability and Load Distribution:*

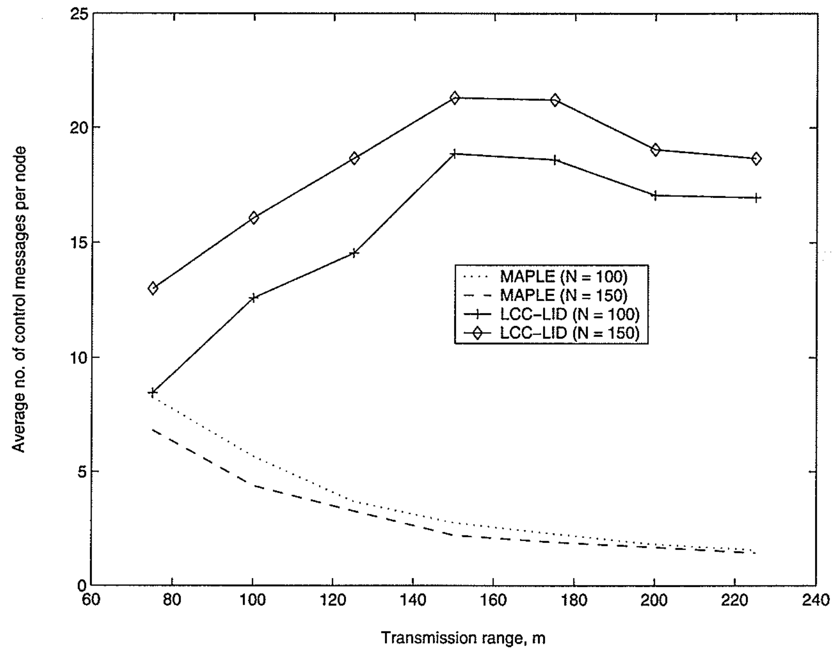


Figure 4.10. *Control message overhead for MAPLE and LCC-LID for varying transmission range (TR).*

One of the major drawbacks of LID-based clustering is its instability and LCC-based clustering can be used to make LID based clustering more stable. In LCC-LID-based clustering, cluster heads do not change very frequently. A change in cluster head occurs only in two situations: when two cluster heads collide or a cluster member is disconnected from its cluster head [9]. In MAPLE clustering, cluster heads also change quite infrequently. In MAPLE clustering, a cluster head gives up its role only when there is a cluster head in its one-hop neighborhood and its members have alternative cluster heads.

Figs. 4.14-4.15 illustrate the stability and the load distribution performances of MAPLE clustering in comparison to LCC-LID-based clustering. We notice that, at smaller transmission ranges MAPLE clustering provides better stability performance compared to that due to LCC-LID-based clustering. However, when the transmission range is relatively large, both the schemes provide more or less similar stability performance.

We notice that the load distribution is better with MAPLE clustering (Fig. 4.15).

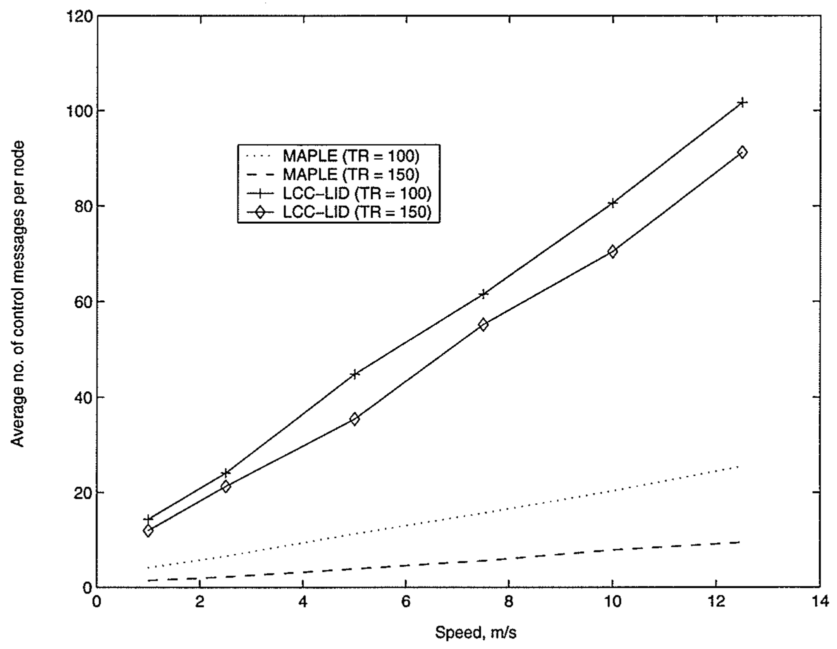


Figure 4.11. *Control message overhead for MAPLE and LCC-LID for varying node speed.*

This is because MAPLE clustering does not rely on node ID and each of the nodes has equal potential to become a cluster head. Here, the cluster formation depends on the spatial distribution of the nodes and how the nodes access the channel.

Also, to reduce the link cost (i.e., transmission power to reach the cluster head), a node occasionally selects another cluster head from its neighborhood, if there is any. This causes some cluster heads to give up their role giving chances to other nodes to become cluster heads. This also contributes to the more uniform load distribution.

- *Average Normalized Transmission Power:*

  In MAPLE clustering, the mobile nodes always choose the cluster heads which can be reached with the minimum transmission power. Therefore, the overall energy consumption is smaller. Also, since a reservation-based channel access mechanism is used, the medium access is more organized and there are fewer number of collisions among the nodes during channel access. This results in

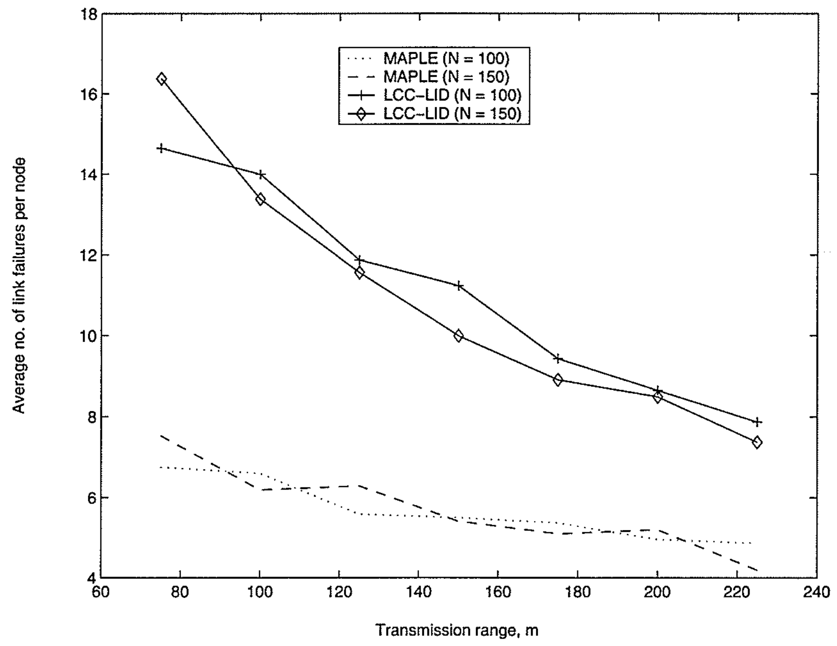


Figure 4.12. *Average number of link failures for MAPLE and LCC-LID.*

conservation of battery power in the nodes.

The average normalized transmission power is sensitive to the propagation environment. With MAPLE clustering, the saving in the average normalized transmission power per node increases as the value of the path-loss exponent δ increases (Fig. 4.16). We observe that, with $N = 100$, and $\delta = 4$, the average normalized transmission power is about 18%. This implies that, with MAPLE clustering the average power consumption is only 18% of that required when all the nodes use the same transmission power and do not employ any power saving scheme.

4.4 Summary

Simulation results indicated LCC-LID to be one of the best non-access-based clustering techniques. We therefore compared the performance of MAPLE with LCC-LID. We concluded that MAPLE results in more bandwidth, energy-efficient and robust clustering compared to LCC-LID-based clustering.

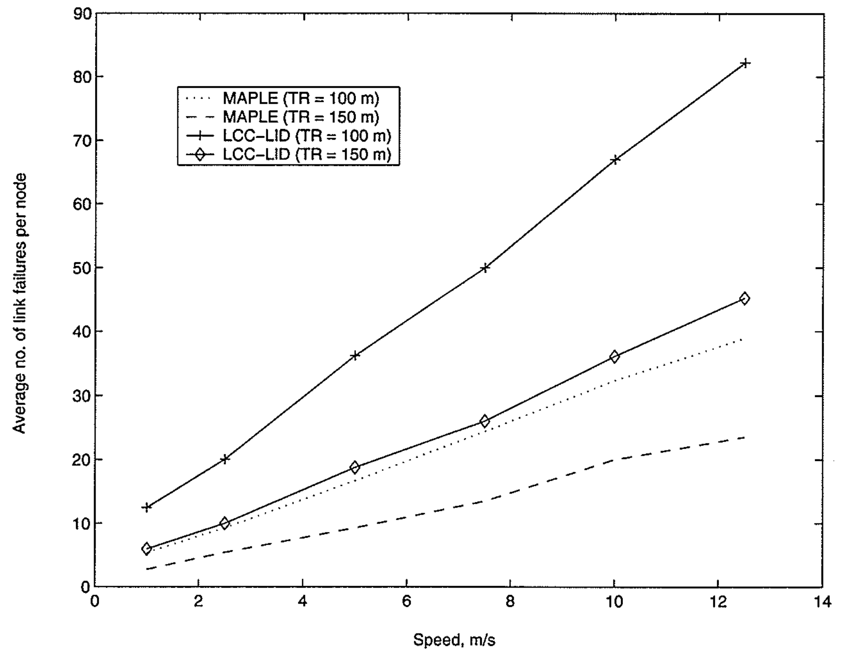


Figure 4.13. *Impact of user mobility on link failure.*

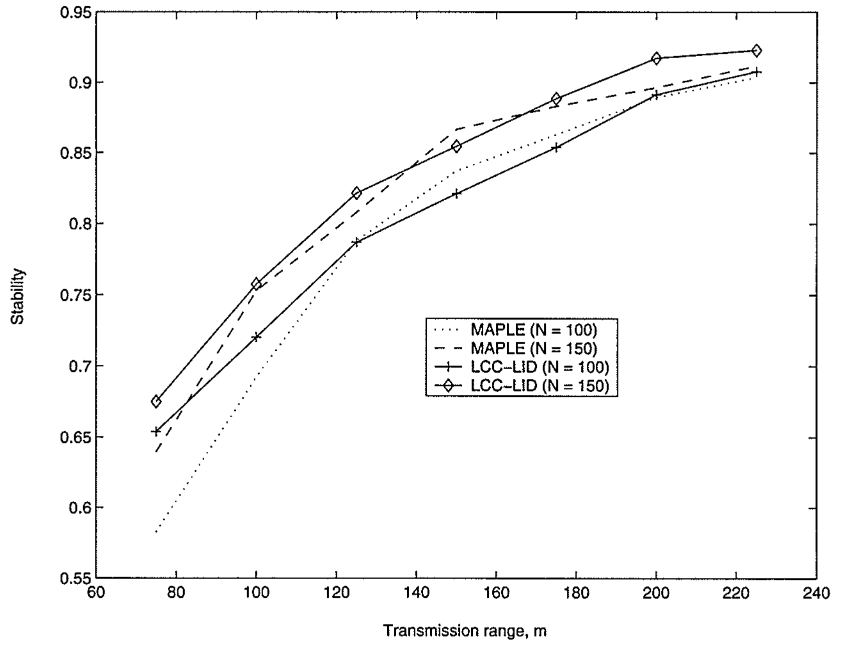


Figure 4.14. *Stability for MAPLE and LCC-LID.*

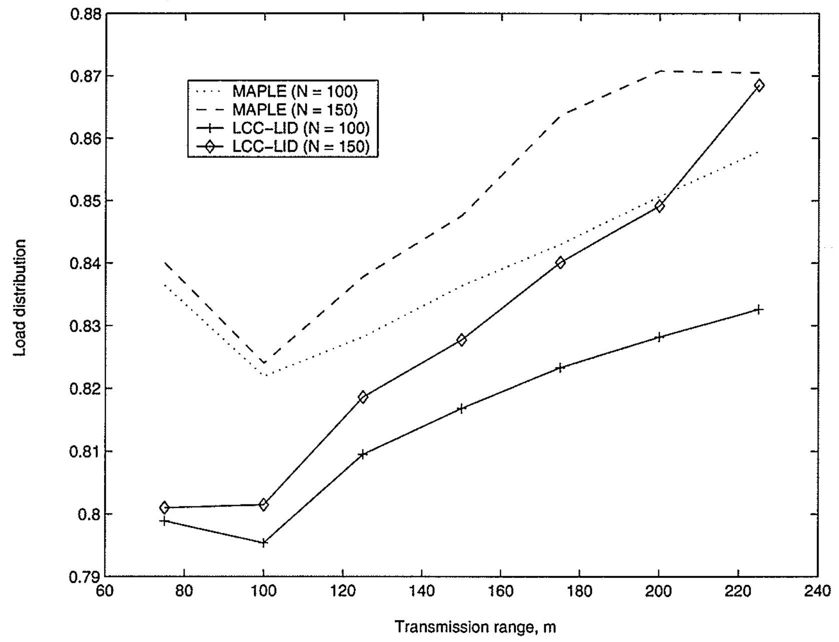


Figure 4.15. *Load distribution for MAPLE and LCC-LID.*

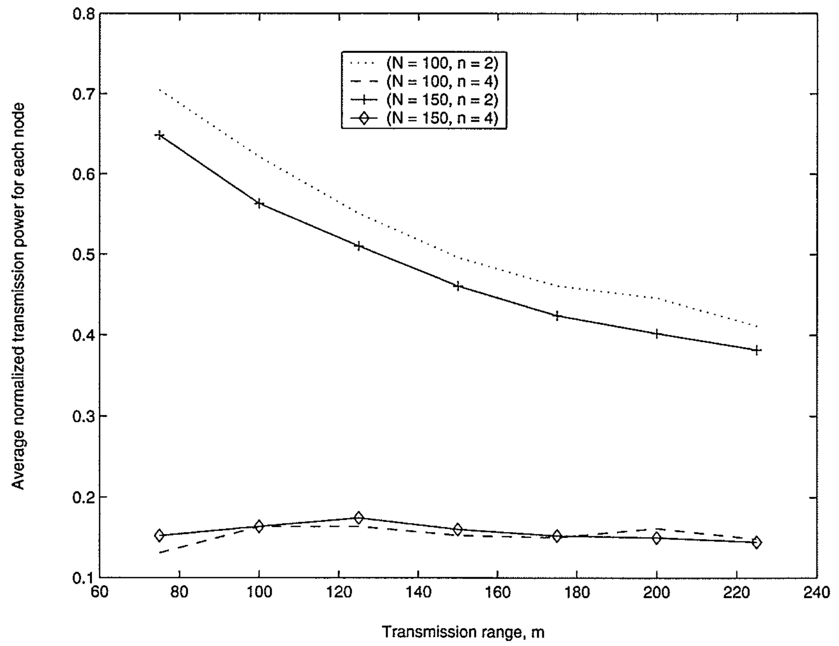


Figure 4.16. *Average amount of transmission power consumed per mobile per control message (for MAPLE clustering).*

Chapter 5

Performance Comparison with Access-Based Clustering

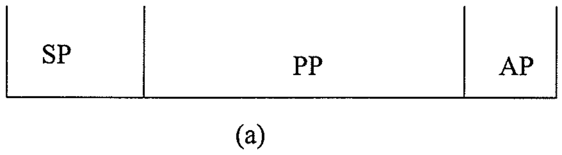
Access-based clustering in a mobile ad hoc network generally builds on a broadcast mechanism in the shared radio channel to transmit control messages for cluster formation. The topology information of the network is not essential for clustering and the changing topology does not necessitate topology update information to be propagated to maintain the cluster structures. The mobile nodes in the network contend for access in the shared broadcast channel when they intend to transmit control messages for cluster formation. A node cannot receive a packet when it is transmitting and a node cannot receive more than one packet simultaneously.

The access-based clustering techniques CABC and ABCP were discussed in chapter 2. The channel access mechanisms of these two clustering techniques are discussed in the following sections. We later compare the performance of these two access-based clustering algorithms to MAPLE.

5.1 Channel Access Technique in CABC

CABC uses Randomized Broadcast Channel Access (RBCA) algorithm the key parameters of which are optimized for the worst case scenarios. In RBCA, the control channel is slotted and each slot can transmit only one control packet. In order to incorporate the multiple access capability, the slot contains a sensing period (SP) and an acknowledgement period (AP) in addition to the actual control packet period (PP) (Fig. 5.1(a)).

Any mobile node willing to broadcast a control message contends in an upcoming slot in the channel. For this, it generates a backoff value δ $(0 < \delta < 1)$ from the distribution $f(\delta) = (r+1)\delta^r$. The optimal value of r is calculated at the network initialization time. After generating δ, the node waits $\delta\beta$ (where β is the normalized duration of SP w.r.t. PP) before sensing the control channel. The details of the protocol for accessing the common control channel can be found in [25].



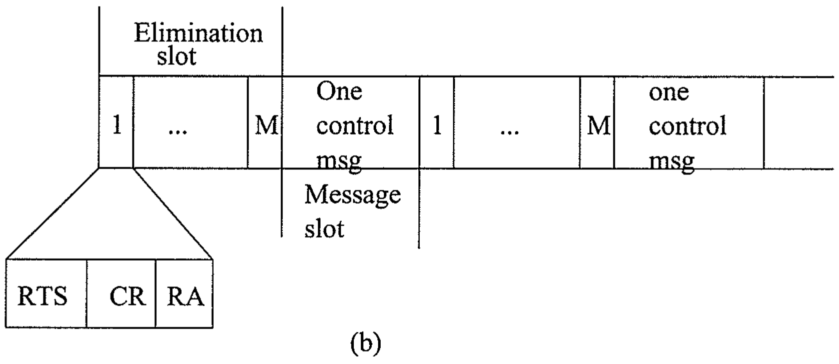


Figure 5.1. *Channel structure in (a) CABC and in (b) ABCP.*

The key parameters for optimizing the performance of the broadcast algorithm are r and β. The optimal value of r (to maximize the slot success probability) for a node can be determined based on the theory of order statistics when the maximum number of one-hop and two-hop neighbors are known.

5.2 Channel Access Technique in ABCP

In ABCP, the mobile nodes use Three-phase Multiple Access (TPMA) Protocol during cluster formation.

The control channel is divided into fixed size frames, which is composed of an elimination slot followed by a message slot (Fig. 5.1(b)). The elimination slot is further subdivided into M mini-slots. Each mini-slot goes through three phases: *request to send (RTS)* phase for the nodes to indicate that they are willing to transfer message in the current frame, *collision report (CR)* phase which follows the RTS phase is used by the nodes to report collisions if they sense any during the RTS phase, and the *receiver available (RA)* phase when nodes receiving only one RTS indication acknowledge the RTS request. Each node follows the following protocol for accessing the common control channel:

- If the control buffer has a new packet to deliver, wait until the next frame and send RTS indication in the first mini-slot of elimination slot.
- If more than one RTS are received in a mini-slot, report collision during CR phase.
- If RTS was sent during a mini-slot, check the CR indication; if CR is detected attempt in the next mini-slot with probability p or abandon contending in the current frame with probability $1 - p$.
- If only one RTS is received, send acknowledgement during RA phase of the mini-slot.

The choice of the values for M and p affects the performance of the TPMA protocol. If M is chosen too small, many of the frames will go empty because no winner may be selected with the number of mini-slots. Increasing the number of mini-slots will increase the frame length and will degrade the time efficiency. If the number of nodes in the network is high, the value of p has to be small to increase the probability of finding a winner in a frame. However, since the number of active nodes in the network is not fixed, the optimal value of p is difficult to obtain. Choosing p for worst-case scenario will result in excessive contention resolution time under normal load conditions.

5.3 Limitations of ABCP and CABC

The TPMA protocol is highly inefficient because of the way it tries to resolve contentions among the contending mobile stations. The technique is not scalable at all for a large ad hoc network. Furthermore, in the TPMA, to determine the optimal contending probability and the number of mini-slots in the elimination slot, total number of contenders in the broadcasting zone has to be known. If this number is not known, the contention resolution time may become significantly higher than expected. RBCA improves over TPMA and it scales with the size of the network.

However, RBCA has the following limitations:

- To optimize the key parameters, the exact values of the maximum number of one-hop and two-hop neighbors have to be known. If the values chosen are not accurate, the optimality expressions do not hold anymore and we get values for the backoff parameter r and the sensing period length (SP) which might be overly optimistic (when the maximum number of one-hop and/or two-hop neighbors are in fact higher) or overly conservative (when the maximum number of one-hop and/or two-hop neighbors are less than the considered values).
- To achieve optimal performance, all the nodes have to know the exact number of one-hop and two-hop neighbors each time it tries to contend in the channel to broadcast its control packet. This is impractical to implement, especially, when the nodes are on move. Maintaining the information about the one-hop and two-hop neighbors will cause extra messaging and will diminish the benefits of the access-based clustering in reducing control overhead.

Both ABCP and CABC suffer from the following limitations:

- Mobility of the nodes is not considered in the cluster head selection process. If a highly mobile node is selected, cluster stability will be greatly impaired.
- The clustering algorithms do not consider the power and energy efficiency of the clustering process.
- The clustering algorithms may result in highly irregular cluster structure with respect to cluster sizes which may create high degree of load imbalance across the network.

Improved mechanisms for access-based clustering can be designed by incorporating the mobility and power efficiency awareness in the clustering process. When a node joins a new cluster or moves out of its current cluster, it may selectively join the cluster for which the received signal energy from the cluster head is the highest. The node then can adjust its power level to reduce power consumption. The proposed clustering mechanism, as described in the following sub-section, is based on these observations.

5.4 Simulation Model and Assumptions

We use a discrete-event simulator, written in PARSEC [30], to evaluate the performances of all the above three clustering schemes. The one-step Markov path model [16] is adopted in which each mobile node has a higher probability in moving in the same direction as the previous move. The probability assignments in eight directions are assumed to be same as in [25]. There is only one control channel which is used for exchanging the control messages during clustering and the channel is assumed to be error free.

The mobile nodes are placed randomly within a 1000m $\times$ 1000m area. When a node reaches the boundary of the area, it is bounced back. Each mobile node is assumed to be moving with the same speed (v). The values of the parameters p and M (in ABCP) are assumed to be 5/8 and 10, respectively. The value of M is assumed to be 16 for MAPLE clustering. The simulation time is 25 minutes.

5.5 Performance Results and Discussions

- *Cluster Head Change Rate:*
  The cluster head change rate increases with increasing node mobility (Fig. 5.2). We observe that MAPLE performs significantly better than both CABC and ABCP. In fact, both CABC and ABCP perform very closely, because the cluster head selection mechanisms for these two cases are nearly the same. Since none of CABC and ABCP considers the mobility trend of the nodes, many high mobility nodes may become cluster heads. These cluster heads lose connectivity with their members within relatively small period of time. On the other hand, in MAPLE, since the mobility tracking mechanism is intertwined with the access-based cluster head selection process, high mobility nodes are less likely to be selected as cluster heads. As a result, the stability of the clusters formed with MAPLE is expected to be much better.
  Cluster head change rate decreases with increasing transmission range, because with larger transmission range the clusters formed tend to be more stable. The cluster head change rate has been observed to be better with MAPLE clustering over a wide range of transmission range (Fig. 5.3).

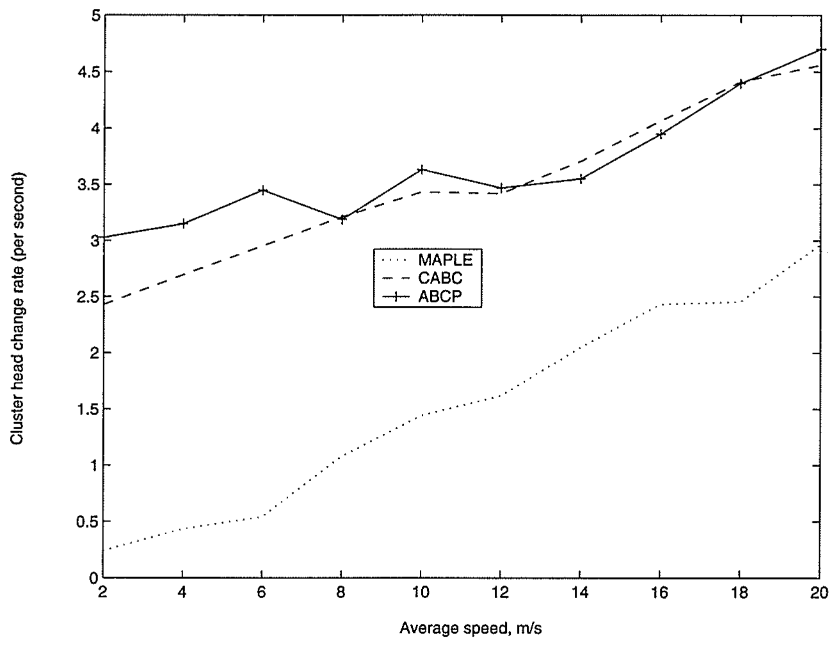


Figure 5.2. *Variations in cluster head change rate with node speed (for N = 150, transmission range = 150 m).*

- *Cluster Maintenance Overhead:*

  Compared to CABC, ABCP incurs significantly higher maintenance overhead and it rises sharply with increasing node speed (Fig. 5.4) and MAPLE incurs less overhead compared to CABC (Fig. 5.5). In ABCP high volume of messages in the form of REQ_TO_JOIN, HELLO and JOIN messages are transmitted each time a node becomes unclustered. In CABC, the only control message involved in clustering is the LCM message (regular beacon messages are not considered as clustering messages) and in MAPLE it is only the WELCOME message (periodic MCHs broadcast by the clusterheads are not considered as clustering messages).

  When the node mobility is high, in case of MAPLE clustering, the pre-selected auxiliary cluster heads also move away from the mobile nodes and at the same time, the mobility tracking becomes less accurate. More number of nodes start declaring themselves as cluster heads, and consequently, the overhead increases (Fig. 5.5).

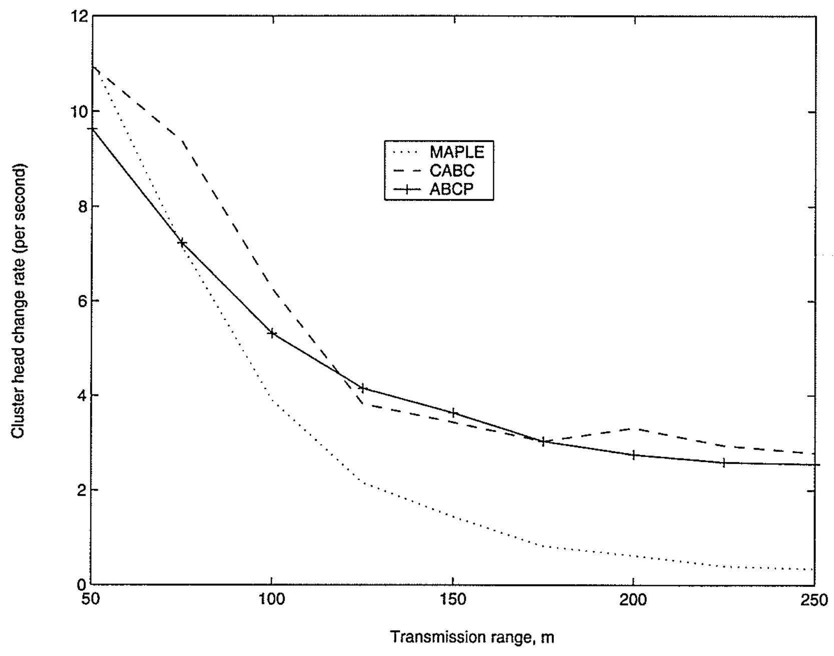


Figure 5.3. *Variations in cluster head change rate with transmission range (for N = 150, v = 10 m/s).*

- *Average Non-clustered Duration:*

  For ABCP the average non-clustered duration, expressed as the percentage of the total simulation time, has been observed to be longer compared to that for CABC. MAPLE clustering outperforms both the ABCP and the CABC schemes.

  As node mobility increases, more nodes become disconnected from their cluster heads and some cluster heads become non-clustered by losing their members. In ABCP, the non-clustered nodes have to exchange HELLO and JOIN messages by contending through the channel. This exacerbates the contention resolution problem and makes the non-clustered nodes wait a longer amount of time before they become clustered again.

  In CABC, non-clustered mobile nodes wait for beaconing time and if no LCM or header beacon message is received, they start contending for transmitting the LCM itself. In CABC, the significantly lower contention resolution time makes the joining process much faster. Moreover, any newly generated LCM message

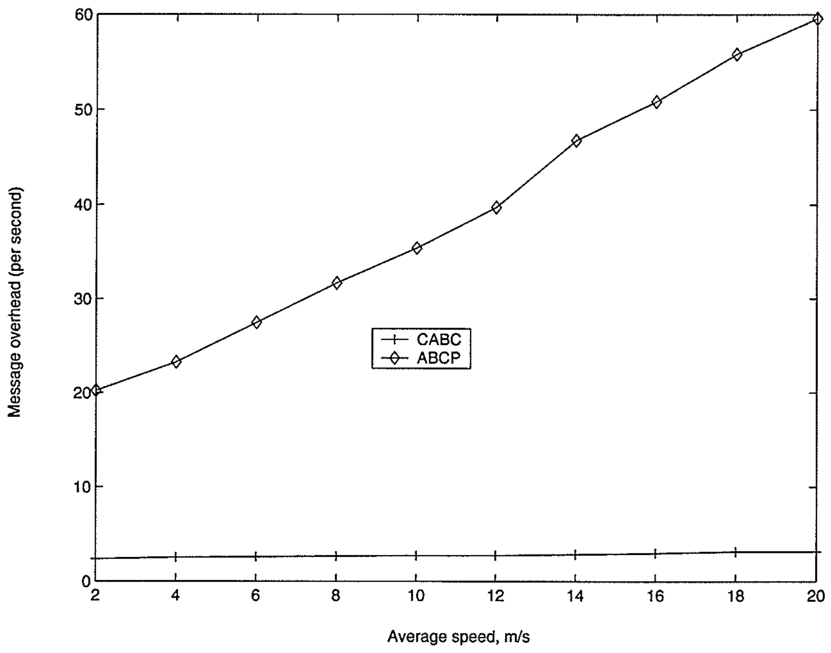


Figure 5.4. *Cluster maintenance overhead for CABC and ABCP for different node speed (for N = 150, transmission range = 150 m).*

does not need to contend with any joining message from the cluster members, since the joining messages are piggybacked inside the beacon messages. However, this piggybacked joining message may take extra time (depending on when the next beacon message is scheduled). For this duration, a mobile node trying to join a cluster is effectively non-clustered, because the corresponding cluster head is unaware of the fact that the node has picked it as the cluster head.

In MAPLE clustering, since a node keeps track of a secondary cluster head (which is the nearest one among the neighboring cluster heads except its own cluster head), a node can select a cluster head immediately after it moves away from its current cluster head. Non-clustered time will accumulate only if the node does not have any other cluster head in its range. But, in this scenario, as the node already knows that there is no cluster head to join in its range, it does not need to wait further listening to the channel and it simply has to spend some time to find a suitable frame to declare itself a cluster head. By following this intelligent cluster switching scheme, MAPLE clustering exhibits

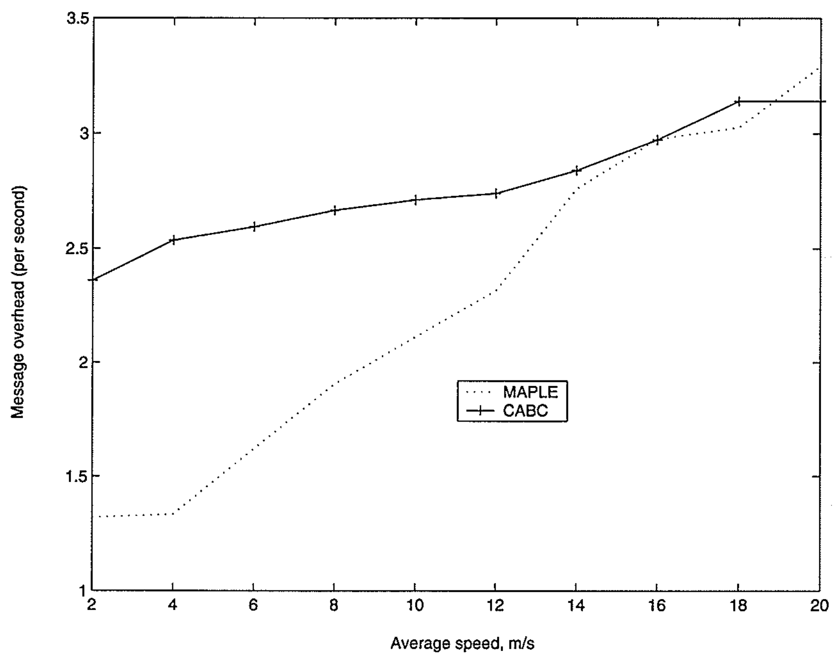


Figure 5.5. *Cluster maintenance overhead with MAPLE and CABC clustering for different node speed (for N = 150, transmission range = 150 m).*

less non-clustered duration for the nodes compared to that for CABC (Fig. 5.6).

- *Load Distribution:*

  Among the three schemes, the MAPLE clustering achieves the best load distribution among the nodes in the network (Fig. 5.7). In CABC, cluster sizes can be very irregular–some clusters being extremely large in size, while some others ending up having one or two cluster members. This irregular cluster structure causes high degree of load imbalance among the cluster heads and across the network as a whole. On the other hand, with MAPLE clustering, the maximum cluster size is fixed and the resulting clusters are much more uniform in size. For this reason MAPLE clustering provides superior performance in terms of load distribution. All the three clustering schemes achieve better load distribution when the transmission range increases.

  During simulating CABC, we assume that the maximum number of one-hop and two-hop neighbors are known in advance and optimization for it was done accordingly. But, in a practical set up, these values may not be available *a*

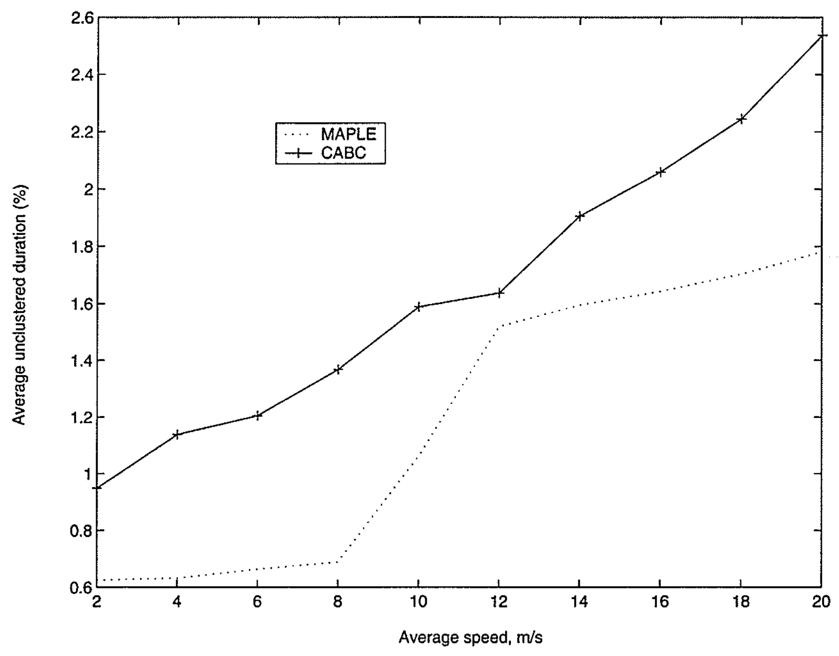


Figure 5.6. *Average non-clustered duration for MAPLE and CABC for different node speed (for N = 150, transmission range = 150 m).*

priori. Therefore, the performance results for CABC as presented above may become worse in a practical ad hoc networking scenario.

5.6 Summary

A comparative performance evaluation on the three access-based clustering schemes ABCP, CABC and MAPLE has been presented in this chapter. Simulation results show that MAPLE eliminates some of the major problems with the ABCP and CABC. It benefits from the proactive approach which utilize link level information for mobility tracking. MAPLE provides superior stability and incurs less clustering overhead. Average non-clustered duration has been observed to be lower and cluster sizes are more uniform (hence better load distribution) in MAPLE. However, the performance of MAPLE clustering may suffer in an overly dense network or when the node mobility is very high.

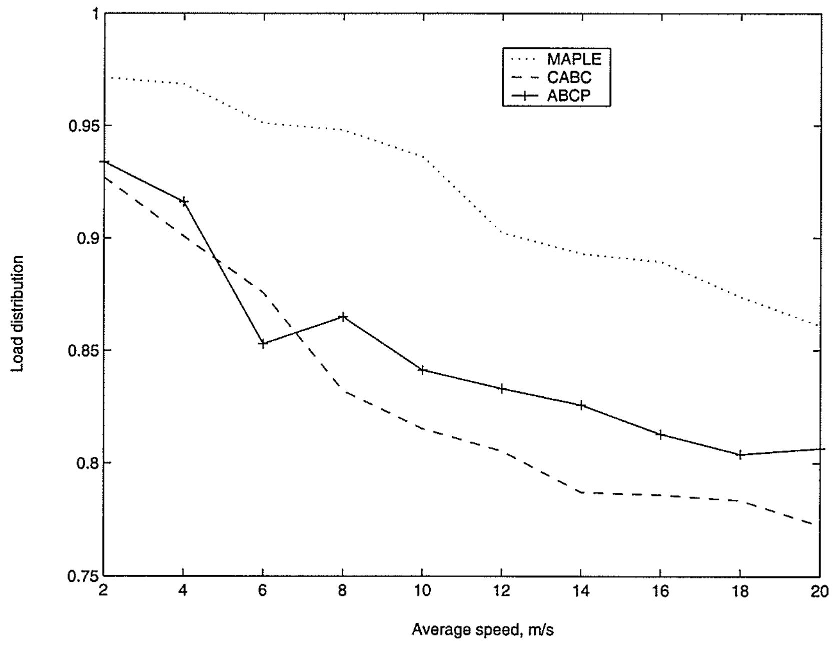


Figure 5.7. *Variations in load distribution with node speed (for N = 150, transmission range = 150 m).*

Chapter 6

Conclusion and Future Work

6.1 Conclusion

In this thesis, we deployed an access-based clustering algorithm that exploits the radio link level information in a proactive manner.

The goals of this thesis were to:

- study the importance of clustering in ad hoc wireless networks;
- explore the issues related to clustering;
- study the strength and weaknesses of existing clustering algorithms;
- examine the impact of different network parameters such as transmission range of the nodes, mobility etc on clustering;
- suggest an improved clustering technique.

We reached our goals as follows.

In chapter 2, we discussed the importance of clustering in ad hoc wireless networks and discussed major clustering algorithms proposed in the literature. We concluded that there is no single algorithm that studies the important issues of clustering: proactiveness, mobility awareness, and power awareness. This led us to develop a new clustering algorithm, called MAPLE.

MAPLE is an access-based, proactive and energy efficient clustering algorithm. A novel channel reservation technique is used to reduce the number of contentions among the nodes accessing the channel. Simulation results show that the MAPLE framework results in stable, low energy, robust and efficient clustering in the ad hoc networks. We also presented some analytical results to show the impact of different parameters related to MAPLE clustering. We estimated the maximum number of nodes that can be supported by MAPLE in an ad hoc network.

Chapters 4 and 5 presented comparative performance analysis of non-access-based and access-based clustering algorithms to MAPLE respectively. We first conducted extensive simulations on three non-access-based clustering algorithms to select the best algorithm for a fair comparison with MAPLE. LCC-LID was the chosen algorithm. Simulation results indicated that MAPLE results in more bandwidth, energy efficient, and robust clustering compared to LCC-LID.

Access-based clustering protocol (ABCP) and channel access-based clustering (CABC) were compared to MAPLE in chapter 5. Simulation results show that MAPLE offers superior performance in terms of stability, clustering overhead and load distribution.

In MAPLE, each of the cluster members has a pre-assigned mini-frame which lies in the control frame. Before joining a cluster, a node needs to contend for accessing a mini frame to become a member of a cluster. In chapter 5, we have shown an elaborate simulation study to measure the performance of mini-frame access. We consider two scenarios, the first one considers capture effect and the another is without capture effect. We have concluded that in both cases the rejection rate by the cluster head (during the joining of a node) is low which in turn indicates that the nodes need not stay non-clustered for a significant amount of time.

6.2 Future Work

The ultimate goal of clustering is to improve the performance of routing and upper layer protocols. The impact of clustering on these layers have to be investigated in future. There are some cluster based routing algorithms [31] with which we can compare the non-cluster based routing algorithms. We can see the impact of clustering by running the routing algorithm on top of clustering framework. In particular, MAPLE can be combined with an existing routing algorithm to study the benefit of clustering.

In this thesis, we assumed that all the transmissions are error free. That is, once a packet is transmitted, it reaches the destination without any error. However, in reality, the wireless channels are very much error prune. In the future, we can examine the performance of different clustering techniques in presence of wireless transmission errors.

Bibliography

[1] J.-P. Hubaux, T. Gross, J.-Y. L. Boudec, and M. Vetterli, "Towards self-organized mobile ad hoc networks: The terminodes project," *IEEE Communications Magazine*, pp. 118-124, Jan. 2001.

[2] C.-K. Toh, *Ad Hoc Mobile Wireless Networks: Protocols and Systems*, Prentice Hall, 2001.

[3] Charles E. Perkins, *Ad Hoc Networking*, Addison-Wesley, 2001.

[4] R. Min, M. Bhardwaj, SH Cho, N. Ickes, E. Shih, A. Sinha, A. Wang, and A. Chandraksan, "Energy-centric enabling technologies for wireless sensor networks," *IEEE Wireless Communications*, Aug. 2002.

[5] V. Rodoplu and T. H. Meng, "Minimum energy mobile wireless networks," *IEEE Journal on Selected Areas in Communications*, pp. 1333-1344, Aug. 1999.

[6] J. J. Garcia-Luna-Aceves and J. Raju, "Distributed assignment of codes for multihop packet-radio networks," in *Proc. IEEE MILCOM'97*, Monterey, California, Nov. 1997.

[7] H. S. Chhaya and S. Gupta, "Performance of asynchronous data transfer methods for IEEE 802.11 MAC protocol," *IEEE Personal Communications Magazine*, pp. 8-15, Mar. 1996.

[8] R. Krishnan, R. Ramanathan, and M. Steenstrup, "Optimization algorithms for large self-structuring networks," in *Proc. IEEE INFOCOM '1999*, vol. 1, pp. 71-78, New York, USA, Mar. 1999.

[9] C.-C. Chiang, H.-K. Wu, and M. Gerla, "Routing in clustered multihop, mobile wireless networks with fading channel," Technical Report, Computer Science Department, UCLA.

[10] M. Gerla and J. T.-C. Tsai, "Multicluster, mobile, multimedia radio network," *ACM/Baltzer Wireless Networks*, vol. 1, no. 3, pp. 255-265, 1995.

[11] R. Ramanathan and M. Steenstrup, "Hierarchically-organized multihop mobile wireless networks for quality-of-service support," *Mobile Networks and Applications*, vol. 3, no. 1, pp. 101-119, 1998.

[12] P. Krishna, N. H. Vaidya, M. Chatterjee, and D. K. Pradhan, "A cluster-based approach for routing in dynamic networks," *Computer Communications Review*, vol. 17, no. 2, Apr. 1997.

http://www.acm.org/sigs/sigcomm/ccr/archive/1997/apr97/ccr-9704-krishna.pdf

[13] P. Sinha, R. Sivakumar, and V. Bharghavan, "CEDAR: A core-extraction distributed routing algorithm," in *Proc. IEEE INFOCOM '1999*, vol. 1, pp. 202-209, New York, USA, Mar. 1999.

[14] D. J. Baker and A. Ephremides, "The architectural organization of a mobile radio network via a distributed algorithm," *IEEE Transactions on Communications*, COM-29(11), pp. 1694-1701, Nov. 1981.

[15] C. L. Richard and M. Gerla, "Adaptive clustering for mobile wireless networks," *IEEE Journal on Selected Areas in Communications*, vol. 15, no. 7, pp.1265-1275, Sept. 1997.

[16] T.-C. Hou and T.-J. Tsai, "Adaptive clustering in a hierarchical ad hoc network," in *Proc. Intl. Computer Symp.*, pp. 171-176, National Cheng Kung University, Taiwan, Dec. 1998.

[17] A. B. McDonald and T. F. Znati, "A mobility-based framework for adaptive clustering in wireless ad hoc networks," *IEEE Journal on Selected Areas in Communications*, vol. 17, no. 8, pp. 1466-1487, Aug. 1999.

[18] T. J. Kwon and M. Gerla, "Clustering with power control," in *Proc. IEEE MILCOM'99*, vol. 2, pp. 1424-1428, Atlantic City, NJ, USA, Oct. 1999.

[19] A. D. Amis, R. Prakash, T. H. P. Vuong and D. T. Huynh, "Max-min D-cluster formation in wireless ad hoc networks," in *Proc. IEEE INFOCOM '00*, pp. 32-41, Tel Aviv, Mar. 2000.

[20] A. D. Amis and R. Prakash, "Load-balancing clusters in wireless ad hoc networks," in *Proc. 3rd IEEE Symposium on Application-Specific Systems and Software Engineering Technology*, pp. 25-32, Richardson, Texas, USA, Mar. 2000.

[21] T.-C. Hou and T.-J. Tsai, "An access-based clustering protocol for multihop wireless ad hoc networks," *IEEE Journal on Selected Areas in Communications*, vol. 19, no. 7, Jul. 2001.

[22] J. Gao, L. J. Guibas, J. Hershberger, L. Zhang and A.Zhu,"Discrete mobile centers," in *Proc. 17th Annual Symposium on Computational Geometry (SCG)*, pp. 188-196, Medford, MA USA, Jun. 2001.

[23] S. Banerjee and S. Khuller, "A clustering scheme for hierarchical control in multi-hop wireless networks," in *Proc. IEEE INFOCOM '01*, pp. 1028-1037, Anchorage, Alaska, USA, Apr. 2001.

[24] Y. Fernandess and D. Malkhi, "K-clustering in wireless ad hoc networks," in *Proc. ACM POMC '2002*, pp. 31-37, Toulose, France, Oct. 2002.

[25] Z. Cai, M. Lu, and X. Wang, "Channel access-based self-organized clustering

in ad hoc networks," *IEEE Transactions on Mobile Computing*, vol. 2, no. 2, Apr.-Jun. 2003.

[26] A. Siddiqui and R. Prakash, "Modeling, performance measurement, and control of clustering mechanisms for multi-cluster mobile ad hoc networks," Technical Report (UTDCS-16-01), Department of Computer Science, University of Texas at Dallas.

[27] T. Camp, J. Boleng, and V. Davies, "A survey of mobility models for ad hoc network research," *Wireless Communications and Mobile Computing*, vol. 2, pp. 483-502, 2002.

[28] Wuyi Yue, "The effect of capture on performance of multichannel slotted ALOHA systems," *IEEE Transactions on Communications*, vol. 39, no. 6, pp. 818-822, Jun. 1991.

[29] M. Zorzi and R Rao, "Slotted ALOHA with capture in a mobile radio environment," in *Proc. International Zurich Seminar*, pp. 391-399, Mar. 1994.

[30] R. Bagrodia, R. Meyer, M. Takai, Y. Chen, X. Zeng, J. Martin, and H. Y. Song, "PARSEC: A parallel simulation environment for complex systems," *IEEE Computer*, vol. 31, no. 10, pp. 77-85, Oct. 1998.

[31] M. Jiang, J. Li, and Y.C. Tay, "Cluster based routing protocol (CBRP)," draft-ietf-manet-cbrp-spec-01.txt, Internet Draft, IETF, Aug. 1999.

VITA

Surname:	Palit	***Given Names:***	Rajesh
Place of Birth:	Bangladesh	***Date of Birth:***	Feb. 10, 1977

Educational Institutions Attended

Bangladesh University of Engineering and Technology (BUET)	1995 to 2000

Degrees Awarded

B.Sc. in Computer Science and Engineering (BUET)	2000

Honors and Awards

University of Manitoba Graduate Fellowship	2003-04
Alfred Rea Tucker Memorial Scholarship	2003
University of Manitoba Students' Union Scholarship	2003

Journal Publications

1. R. Palit, E. Hossain, and P. Thulasiraman, "MAPLE: A framework for mobility-aware proactive low energy clustering in ad hoc mobile wireless networks," submitted to *Ad Hoc Networks Journal* (Elsevier).
2. Rajesh Palit and M Abdus Sattar, "Representation of bangla characters in computer system", in *Bangladesh Journal of Computer and Information Technology*, Vol 7, No. 1, Dhaka, Bangladesh, Dec. 1999.

Conference Publications

1. R. Palit, E. Hossain, and P. Thulasiraman, "Mobility-aware proactive low energy clustering in ad hoc mobile networks," submitted to the *IEEE Global Communications Conference 2004 (GlobeCom'04)*, Dallas, Texas, US, Nov.-Dec. 2004.
2. M. Rashid, R. Palit, and E. Hossain, "On access-based self-organized clustering in ad hoc mobile wireless networks," submitted to the *IEEE Global Communications Conference 2004 (GlobeCom'04)*, Dallas, Texas, US, Nov.-Dec. 2004.
3. Abu Wasif, Rajesh Palit, MM Rashid, and C. M. Rahman, "Increasing the performance of classification trees by using a mixed criterion of attribute depen-

dency and gain ratio," in *Proc. of the Int. Conf. on Computer and Information Technology (ICCIT'98)*, Dhaka, Dec. 1998.

4. Abu Wasif, Rajesh Palit, MM Rashid, and C. M. Rahman, "Construction of decision trees by using the criterion of class-dependency," in *Proc. of the Int. Conf. on Computer and Information Technology (ICCIT'98)*, Dhaka, Dec. 1998.

Book Chapters

1. Ekram Hossain, Rajesh Palit, and Parimala Thulasiraman, "Clustering in mobile wireless ad hoc networks: issues and approaches," invited book chapter in *Wireless Communications and Mobile Computing Systems*, Editor M. Guizani, Kluwer Academic Publishers, 2004.