1998 International Computer Symposium
Workshop on Computer Networks, Internet, and Multimedia
December 17-19, 1998, N.C.K.U., Tainan, Taiwan, R.O.C.

Adaptive Clustering in a Hierarchical Ad Hoc Network

Ting-Chao Hou, and Tzu-Jane Tsai

Department of Electrical Engineering,
National Chung Cheng University, Chia-Yi, Taiwan 621, R.O.C.
Email:{tch@ee.ccu.edu.tw, p8502@eeipx2.ee.ccu.edu.tw}

ABSTRACT

This paper describes a two-level hierarchical architecture for ad hoc networks to resolve the routing scalability problem. Mobile hosts are organized into nonoverlapping clusters and clusterheads are elected to form the upper level backbone. We design a self-organizing dynamic reclustering algorithm to maintain the cluster structure in face of topological changes. A network model which is composed of hexagon units is proposed for simulation study. Under this simulation environment, we analyze four reclustering approaches and show that our cluster maintenance approach provides a stable cluster structure in a mobile ad hoc network.

1. INTRODUCTION

The wireless systems we use today, such as GSM (Global System for Mobile Communications), have fixed base stations and centralized administration. However, in some situations, mobile users may want to communicate with others under circumstances in which no established infrastructure exists. These situations are supposed to come about in a building, on a campus, and absolutely in the battlefield or a rescue environment. This kind of network, which does not depend on any pre-existing backbones or entities, is called an "ad hoc" network [1].

Most of the recent research work in ad hoc networks considers routing related issues on a flat (one-level) network architecture. The most serious problem in a flat network is that routing messages would increase exponentially if the network size grows. Instead of treating an ad hoc network as a flat network, the MCDS (minimum connected dominating set) [2] and hierarchical spine routing [3] propose a two-level hierarchical routing architecture to solve the scalability problem of flat routing algorithms. The authors present four approximation algorithms to compute the MCDS in [2], and describe the hierarchical spine routing (basic spine routing within each cluster and link-state routing at the cluster level) in [3]. This spine-backbone architecture is mostly suitable in the situation where nodes are geographically distributed in groups.

In this paper, we consider an ad hoc network which consists of a larger number (above 30) of mobile hosts. In order to enhance the scalability and take advantage of the backbone, which facilitates multicasting, the hierarchical architecture idea is adopted. We also design our algorithms in such a way that they are independent of the underlying topology. The main difference between the approaches discussed above and our approach is described as follows. We think to achieve high throughput with low communication overhead does not only depend on a good routing protocol, but also depends on other factors, such as clustering, resource allocation, etc. So we design a two-level hierarchical architecture by dividing the nodes into several clusters (for the benefit of code spatial reuse) and select one clusterhead within each cluster to form the backbone. One code is dedicated to the control phase on the backbone, and other codes are reserved to transfer packets in the data phase. Based on this hierarchical architecture, special route discovery and maintenance algorithms can be designed. Since they are simple and efficient, a route can be selected quickly with low routing control messages in a mobile environment.

2. NETWORK MODEL

Most of current wireless systems have mobility support such as base stations and access points. Fig. 1, for example, shows a general cellular network with each mobile host being one hop away from the wired infrastructure. In an ad hoc network bereft of pre-designated base stations or routers, however, for the sake of battery-power reduction and network-capacity increase, the environment considered is a multihop wireless network. Fig. 2 illustrates a simple ad hoc network of four mobile hosts which have the same transmission range (indicated by the circles). Not all hosts have a direct link to the others. Host A cannot directly send packets that will reach host D, since host D is not within the range of node A's wireless transmitter. Node A may ask host C to forward packets to host D because host C is within both A's range and D's range. Due to the mobility of all the hosts involved, the routing problem in a multihop environment is apparently more complicated than in wired networks.

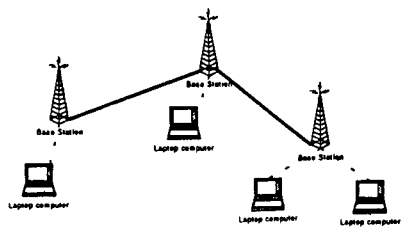

Figure 1. An example cellular network. (single-hop)

To reduce the high overhead in routing updates in a flat mobile network, we consider grouping mobile hosts in clusters to form a two-level hierarchical network. A distributed clustering algorithm is used to assign each mobile

host to a cluster and to select a clusterhead. Each clusterhead serves as a routing server for its cluster members. In addition to the membership information of the cluster, the clusterhead also exchanges topology information with other clusterheads to build a network topology table; and run the route discovery algorithm for its members. In other words, the clusterheads form the nucleus of a backbone network where routing functions are performed. A mobile host queries its clusterhead to obtain the route information to its intended destination. Note that the clusterheads alone do not constitute the backbone network. Some intermediate relay nodes are required to link a pair of clusterheads together.

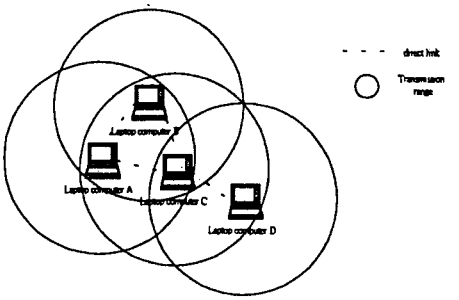

Figure 2. A simple ad hoc network of four mobile hosts

As Fig. 3 shows, three cases are possible between clusterheads on the backbone network. The first case is where there exists one intermediate relay node (if more than one, node with lowest ID is picked up), which we name it a gateway node, between the two clusterheads. The second case is where two gateway nodes exist between the two clusterheads. And the third case includes all scenarios where one or more clusterhead is between two clusterheads. Information transmission is divided into two phases. The control phase is responsible for the exchange of routing related messages. The data phase is used for data transport. Once a mobile host obtains the routing information from its clusterhead, it sets up the route in the flat network structure so that an efficient (e.g., shortest path) route can be used.

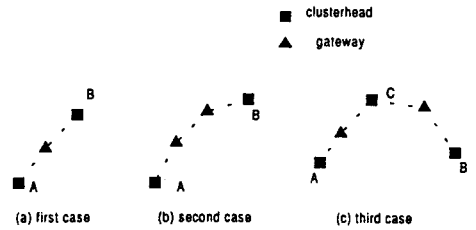

Figure 3. Backbone Network Formation.

2.1 Channel Access

We cluster the mobile hosts into several groups and properly arrange codes (graph-coloring problem) for the benefit of "spatial reuse" of the channel spectrum, like the frequency reuse property in the cellular network. A hybrid TDMA (time division multiple access) scheme is adopted for channel access in each cluster in our system. Cluster heads on the backbone has the responsibility to negotiate the code arrangement between clusters, and be the supervisor of the time division multiple access within their clusters. About the resource allocation, we will go into details in the following resource reservation discussion.

A major challenge in the design of the ad hoc network is the ability to account for resources so that channel reservations can be placed on them. We separate this problem into two parts, one is the code assignment among clusters, and the other is TDMA within each cluster. Transmitter-based code assignment is used in this paper, so inter-cluster collision can be avoided [4].

A. Code Assignment among Clusters

Besides the code with spatial reuse we have mentioned in the previous section, there also exists a problem to resolve when the cluster sizes are different! Intuitively, if the traffic load is uniformly distributed, a larger cluster with more nodes needs more codes; a smaller cluster with fewer nodes would need fewer codes. Using the control phase on the backbone, clusterheads negotiate with the others about the code arrangement among clusters in the network. Large clusters may get more than one code, and small clusters may get one code only. For example, suppose there are four clusters and three codes in the ad hoc network, and cluster 1 has 14 nodes, cluster 2 has 8 nodes, cluster 3 has 6 nodes, and cluster 4 has 2 nodes. We assume the traffic is uniformly distributed, since 3 codes divides 30 nodes results in an average of 10 nodes to share one code, the result of code arrangement is that: cluster 1 has two codes, cluster 2, 3 have one, and cluster 4 has one code, too.

This code assignment scheme has two advantages. First, no matter what kind of distribution the traffic is, it can work efficiently. And second, due to that this code assignment is independent of the cluster structure, it can process well no matter what kind of clustering algorithms are. But another problem remains: When a cluster has more than one code, how do the mobile hosts within this cluster access these codes? We have thought about two ways as Fig. 4 shows. If two codes, A and B, are allocated to this large cluster, we either (a) partition this cluster into two sub-clusters, or (b) all nodes within this cluster can access the two codes. Given consideration to the code spatial-reuse and transmitter tuning frequency, we adopt the first solution to partition a cluster into several sub-clusters.

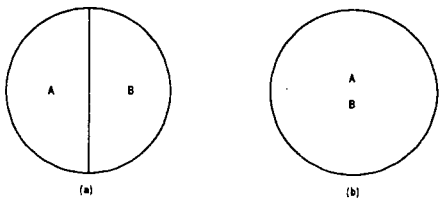

Figure 4. Two codes within a large cluster.

B. TDMA Within Clusters

Within each cluster (or sub-cluster), the MAC layer is implemented using a TDMA scheme. Time is divided into

slots which are grouped into cycles, and the cycle is defined as the maximum interval that two consecutive real-time packets can be separated [4]. At least one time slot per cycle is assigned for each node so that real-time connections will not be blocked. So we divide one cycle into two parts, one is fixed-assigned part and the other is demand-assigned part. Fig. 5 is a channel access example of a cluster with five nodes. Part 1 is fixed-assigned and has five time slots. Part 2 is demand-assigned and has (CYCLE-5) time slots. Since nodes may change their clusters, Part 1 format is readjusted after each join or leave.

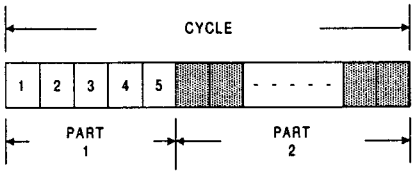

Figure 5. Channel access cycle within a cluster.

3. DISTRIBUTED CLUSTERING ALGORITHMS

The objective of the clustering algorithm is to find an interconnected set of clusters covering the entire node population [4]. A good clustering scheme for the mobile ad hoc network should preserve its cluster structure as long as possible while nodes are moving and the topology is changing. In this section, we probe into the implementation of clustering algorithms and cluster maintenance algorithms.

We can find an interconnected set by selecting the lowest ID node in each cluster to be the clusterhead, or the node with maximum effective degree. The following assumptions, which are common to most radio data link protocols, are made [4].

A1) Every node has a unique ID and knows the ID's of its one-hop neighbors. This can be provided by a physical layer protocol that locates and identifies radio nodes.

A2) A message sent by a node is received correctly within a finite time by all of its one-hop neighbors.

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

3.1Node-ID Based Clustering Algorithm

Clusters are constructed based on node ID. The following algorithm proposed in [4] partitions the multihop network into multiple non-overlapping clusters.

Distributed Clustering Algorithm (DCA)

Γ_i : the set of IDs of my one-hop neighbors, including myself

```
{   if (my_id is the smallest in Γi)
    {   my_cid = i;
        broadcasts cluster(my_id, my_cid);
        Γi = Γi - {my_id};
    }
    for (;;)
    {   on receiving cluster (id, cid);
        {   set the cluster ID of node id to cid;
            if (id==cid and (my_cid is UNKNOWN or
            my_cid>cid))
                my_cid = cid;
            Γi = Γi - {my_cid};
            if (my_id is the smallest in Γi)
            {   if (my_cid is UNKNOWN)
                    my_cid=my_id;
                broadcast cluster (my_id, my_cid);
                Γi = Γi - {my_id};
            }
        }
        if (Γi = ∅ ) stop;
    }
}
```

Figure 6. The Distributed Clustering Algorithm (node-ID based).

Consider the example topology in Fig. 7. After completing the DCA clustering, six clusters are formed in the network as shown in Fig. 8.

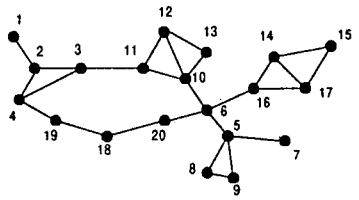

Figure 7. An example of network topology.

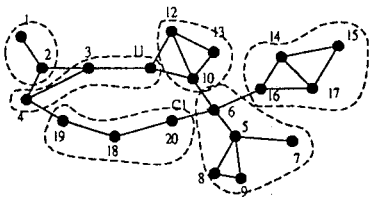

Figure 8. Clustering results of the DCA algorithm (node-ID based).

3.2 Node-Degree Based Clustering Algorithm

We propose a Maximum Degree Clustering Algorithm (MDCA) to partition the multihop network into some nonoverlapping clusters. In this algorithm, clusters are constructed based on the node degree. After the network initialization, in the same way as described in [4], every node knows the ID's of its one-hop neighbors and counts his own degree (the number of direct links one node has). Then, they organize themselves in clusters following the MDCA. The MDCA chooses the node with the maximum degree among his one-hop neighbors to be the clusterhead and forms a backbone network with other clusterheads. With the same network topology in Fig. 7, after completing the MDCA, five clusters are formed as shown in Fig. 10.

Maximum Degree Clustering Algorithm (MDCA)

Γ_i : the set of ID's of my one-hop neighbors, including myself

```
{   Qi = the set of node IDs with maximum degree in Γi
    if (my_id is the smallest in Qi)
    {   my_cid = my_id;
        broadcasts cluster(my_id, my_cid);
        Γi = Γi - {my_id};
```

```
    }
    for (;;)
    {     on receiving cluster (id, cid);
          {     set the cluster ID of node id to cid;
                if (id==cid and (my_cid is UNKNOWN or
                my_cid.degree>cid.degree or
                (my_cid.degree==cid.degree and my_cid>cid) ))
                      my_cid = cid;
                Γi = Γi - {my_cid};
                Qi = the set of node IDs with maximum degree in
                current Γi
                if (my_cid is the smallest in Qi )
                {     if (my_cid is UNKNOWN)
                            my_cid=my_id;
                      broadcast cluster (my_id, my_cid);
                      Γi = Γi - {my_id};
                }
          }
          if ( Γi = Ø )  stop;
    }
}
```

Figure 9. The MDCA algorithm (node-degree-based).

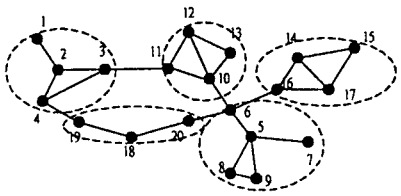

Figure 10. Clustering results of the MDCA (node-degree based).

The node-ID based DCA clustering algorithm is simpler than the MDCA, and the execution time is also shorter. On the other hand, the MDCA creates less clusters as the example shows, so the size (or dimension) of the upper level backbone is smaller.

4. Cluster Maintenance

In the presence of mobility, the network topology changes when a node disconnects from its current neighboring node or connects to a new neighboring node. A topology change makes it necessary to re-examine the cluster structure and also rearrange the cluster structure, if necessary. Frequent reclusterings affect the system performance. Therefore, we design a reclustering algorithm, shown in Fig. 11, to keep the cluster structure as stable as possible, i.e., to minimize the number of node transitions from one cluster to another. We let the clusterhead and its neighbors stay in the original cluster, remove the nodes that leave, and add new joining nodes. So clusterheads and ordinary nodes execute different reclustering operations.

Two assumptions are considered:

A1) No resource limitation: When a node wants to join a cluster, whether there exist enough resource of code or time slot to use is not considered here.

A2) Since clusterheads are responsible for serving their cluster members for routing, clusterheads need to have the whole topology information of the ad hoc network, while ordinary nodes do not. The only function an ordinary node needs to do is to send a handoff message to the new clusterhead that it is going to join.

The following reclustering algorithm (DBRA), Fig. 11, is degree-based and it assumes that a clustering algorithm MDCA is run before it.

Degree-Based Reclustering Algorithm (DBRA)

S_i : the set of IDs of my one-hop neighbors (including old and new ones) and myself.

Orphan nodes: nodes that do not belong to any clusters during the execution of the DRBA.

```
<case 1> I am originally a clusterhead.
{  if (I still have the maximum degree in my old cluster or all of the old
       and new member nodes still have direct links to me)
   {  /* I continue to be the clusterhead */
      for (;;)
      {  if ( a node j wants to join)
         {  /* it is an ordinary node without belonging to a clusterhead
            or it is a single clusterhead */
            if (a handoff message is received)
            {     add node j into my member list;
                  Si = Si - {j};        }
         }
         if (cluster A is my one-hop neighbors now)
         {  /* make sure the algorithm will converge */
            add cluster A members into my member list;
            Si = Si - {x | x ∈ A};      }
         if (a node k wants to leave)
         {  delete node k from my member list;
            Si = Si - {k};              }
         Si = Si - {my_id};
         if ( Si = Ø )        stop;
      }
   }
}
else { /* this cluster is to be decomposed */
       broadcast decomposition message to my members;
       run MDCA with other orphan nodes (the nodes within existing
       clusters ignore this procedure);  }
 }
}
<case 2> I am originally an ordinary node.
{  if ((node my_cid) is not my one-hop neighbor now)
   {  Q = {x | node x is a clusterhead, and my one-hop neighbor now}
      Q'= {q | q ∈ Q, node q with maximum degree in Q}
      if (Q' != Ø )
         join cluster q, q = min{Q'};
      else
         run MDCA with other orphan nodes;
   }
  else    /* node my_cid is still a one-hop neighbor. */
     do nothing;
  if (a decomposition message is received)
  {  run MDCA with other orphan nodes (the nodes within existing
     clusters ignore this procedure);
  }
}
```

Figure 11. Degree-based reclustering algorithm.

An ID-based reclustering algorithm can also be constructed similarly by considering the node IDs in forming the clusters instead of the degrees. The other way to do cluster maintenance is to run the distributed clustering algorithm (DCA or MDCA) periodically. And the interval between consecutive executions of the clustering algorithm depends on the mobility of the nodes. In contrast, in the reclustering algorithm we propose here, the reclustering procedures for all nodes are not executed periodically. When there is an event (a node joins or leaves a cluster) occurring, the reclustering algorithm is executed for related nodes to reduce the volume of control messages. Furthermore, every clusterhead will check whether there exist other clusters that are fully covered within his transmission range. If this is true, they will be merged together so that the number of clusters in this system is controlled to be small.

5. Simulation Model and Results

In previous work [4], an ad hoc network is simulated by placing N nodes randomly in an area of R × R square units. Each square unit describes the position that a mobile node is currently located. This chosen R is made large enough relative to N to form a two-dimensional Poisson point process. Each node moves in one of the four directions allowed by the square-unit model. But no matter how large the R is, due to the square unit description, the path of a mobile node has a zigzag pattern with right angle turns. To allow for a smoother turn in directions, we propose a new network model by placing N nodes in a square area composed of R × R hexagon units as Fig. 12 shows (R=4 here). Each node can move in one of the six directions in angles of $\{0, \pi/3, 2\pi/3, \pi, 4\pi/3, 5\pi/3\}$. This model is more suitable for ad hoc networks in a battleground or the disaster scene. In our simulation, we choose N=60 and R=250.

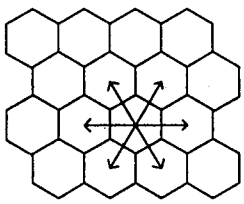

Figure 12. Network Model with hexagon units.

Two mobility models are considered. One is the random path model and the other is the Markov path model (a path model with memory). In the random path model, a node moves uniformly in one of the six directions whenever a move is required. In the one-state Markov path model, a node has a higher probability in moving in the same direction as the previous move. For example, Fig. 13 shows the probability assignment in six different directions.

The nodes can be either stationary or moving. We assume 20% of nodes are stationary and 80% move with a speed within the range of [3.6 km/hr, 36 km/hr]. Simulations include two kinds of mobility model, random path and Markov path. The performance metric chosen is the *cluster instability rate*, which is defined as the number of nodes which changes their clusters per time slot. One thing needs to be noticed is that the meaning of cluster instability rate is not only an instability indicator of the cluster structure but also an instability indicator of the upper level backbone. If a clusterhead is replaced by another node, the cluster members will change their cluster IDs and result in unstable cluster structure.

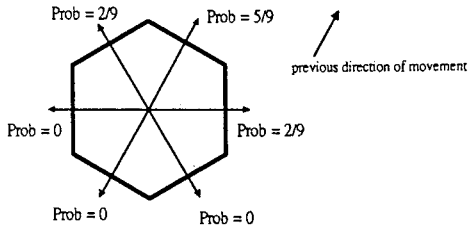

Figure 13. Mobility Model with memory characteristic.

Fig. 14-17 show the instability performance of the four cluster maintenance approaches, ID-based periodic clustering (run in every time slot), ID-based reclustering algorithm, degree-based periodic clustering (run in every time slot), and degree-based reclustering algorithm. In these figures, we call them respectively as IBPC, IBRA, DBPC and DBRA. From Fig. 14 and 15, we observe that the ID-based reclustering approaches generally have better performance than the degree-based reclustering approaches especially when the Markov mobility model is used. For both ID-based reclustering approaches, the number of nodes which change their clusters decreases as the transmission range increases (e.g., transmission range > 20). In contrast, there is no obvious relationship between the degree-based rereclustering algorithms and the transmission range. Because the DBPC approach has bad performance in cluster instability rate (mostly greater than 2), we don't show its full results in these figures. Among these four maintenance approaches, the IBRA algorithm clearly has the best performance.

Fig. 16 and 17 show the relationship of the cluster instability rate with the average mobility rate. We change the speed distribution of the 80% moving nodes, calculate the average speed value, and observe the cluster stability performance. Transmission range is fixed at 85 units. We also observe that the ID-based reclustering algorithm (IBRA) has the better stability than others, and its cluster instability rate increases the slowest as the average mobility rate increases.

6. Conclusion & Future Work

We have presented a two-level hierarchical architecture for ad hoc networks which utilize the self-organizing clustering idea. This architecture can enhance the routing scalability and take advantage of the backbone, which can support flows, multicasting, and even fault-tolerant routing for mobile computers [2]. The proposed cluster maintenance algorithm is simple and has much better mobility stability than continually running the "clustering" algorithm. Furthermore, in order to overcome the limitation of the number of orthogonal codes, we design an adaptive resource

allocation between clusters and within clusters. Therefore, no matter what distribution of mobile computers and traffic load, QoS-guaranteed service may be supported.

For future work, we are investigating the hierarchical routing algorithm which runs in this two-level network architecture. How to find a fault-tolerant, longer-lived, Qos-guaranteed route is the main goal we want to achieve.

7. REFERENCES

[1] B. Johnson and D. A. Maltz, "Truly seamless wireless and mobile host networking: Protocols for Adaptive Wireless and Mobile Networking," IEEE Personal Commun., pp. 34-42, Feb. 1996.

[2] Das and V. Bhaarghavan, "Routing in Ad-Hoc Networks Using Minimun Connected Dominating Sets," in Proc. IEEE ICC'97, pp. 376-380.

[3] B. Das, R. Sivakumar and V. Bharghavan, "Routing in Ad Hoc Networks Using a Spine," in Proc. IEEE Computer Communications and Networks, 1997, pp. 34-39.

[4] C. R. Lin and M. Gerla, " Adaptive Clustering for Mobile Wireless Networks," IEEE JSAC, vol. 15, no. 7, Sep. 1997, pp. 1265-1275.

[5] C. R. Lin and M. Gerla, " Multimedia Transport in Multihop Dynamiç Packet Radio Networks," in Proc. IEEE International Conference on Network Protocols, 1995, pp. 209-216.

[6] C. R. Lin and M. Gerla, "A Distributed Control Scheme in Multi-hop Packet Radio Networks for Voice/Data Traffic Support," in Proc. IEEE ICC'95, pp. 1238-1242.

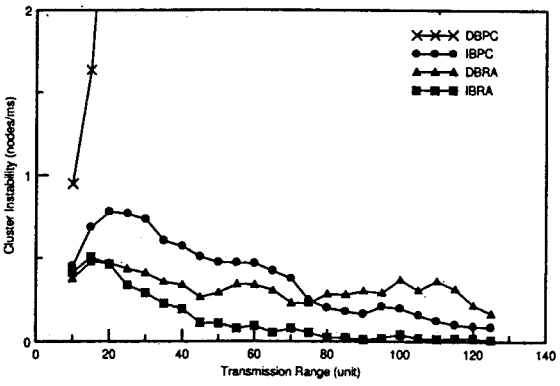

Figure 14. Cluster instability vs. transmission range. (Random Path)

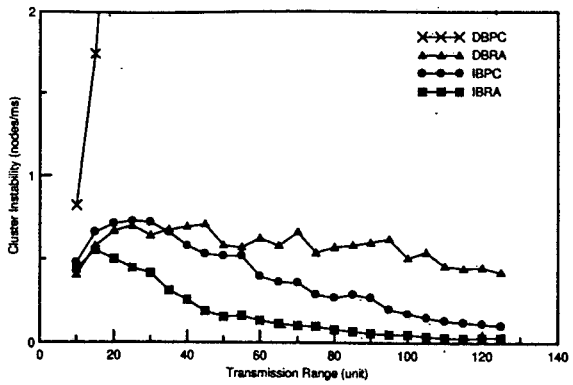

Figure 15. Cluster instability vs. transmission range. (Markov Path)

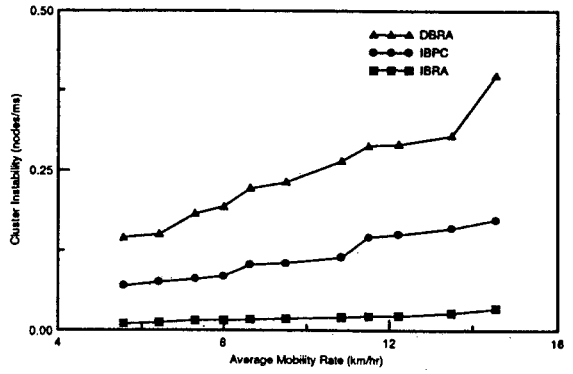

Figure 16. Cluster instability of average mobility rate. (Random Path)

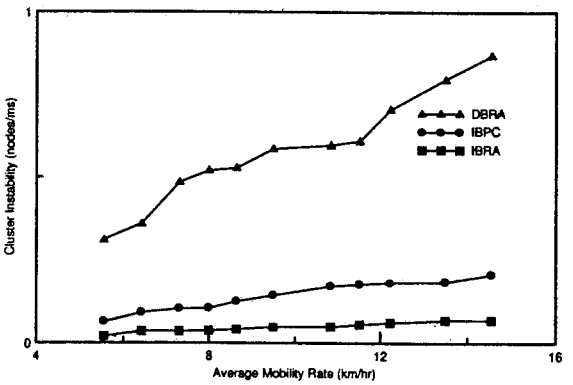

Figure 17. Cluster instability vs. average mobility rate. (Markov Path)