A Weight Based Distributed Clustering Algorithm for Mobile ad hoc Networks*

Mainak Chatterjee, Sajal K. Das, and Damla Turgut

Center for Research in Wireless Mobility and Networking (CReWMaN)
Department of Computer Science and Engineering
University of Texas at Arlington
Arlington, TX 76019-0015
{chat,das,turgut}@cse.uta.edu

Abstract. In this paper, we propose a distributed clustering algorithm for a multi-hop packet radio network. These types of networks, also known as *ad hoc* networks, are dynamic in nature due to the mobility of the nodes. The association and dissociation of nodes to and from *clusters* perturb the stability of the network topology, and hence a reconfiguration of the system is often unavoidable. However, it is vital to keep the topology stable as long as possible. The *clusterheads*, which form a *dominant set* in the network, determine the topology and its stability. Our weight based distributed clustering algorithm takes into consideration the ideal degree, transmission power, mobility and battery power of a mobile node. We try to keep the number of nodes in a cluster around a pre-defined threshold to facilitate the optimal operation of the medium access control (MAC) protocol. The non-periodic procedure for clusterhead election gradually improves the load balance factor (LBF) which is a measure of the load distribution among the clusterheads. For lowering the computation and communication costs, the clustering algorithm is invoked on-demand which aims to maintain the connectivity of the network at the cost of load imbalance. Simulation experiments are conducted to evaluate the performance of our algorithm in terms of the number of clusterheads, *reaffiliation* frequency and dominant set updates. Results show that the our algorithm performs better than the existing algorithms and is also tunable to different types of ad hoc networks.

1 Introduction

Mobile multi-hop radio networks, also called *ad hoc* or *peer-to-peer* networks, play a critical role in places where a wired (central) backbone is neither available nor economical to build. Deployment of cellular networks takes time and cannot be set up in times of utmost emergency. Typical examples of ad hoc networks are law enforcement operations, battle field communications, disaster recovery

* This work is partially supported by Texas Advanced Research Program grant TARP-003594-013, Texas Telecommunications Engineering Consortium (TxTEC) and Nortel Networks, Richardson, Texas.

M. Valero, V.K. Prasanna, and S. Vajapeyam (Eds.): HiPC 2000, LNCS 1970, pp. 511–521, 2000.

situations, and so on. Such situations demand a network where all the nodes including the base stations are potentially mobile, and communication must be supported untethered between any two nodes.

A multi-cluster, multi-hop packet radio network for wireless systems should be able to dynamically adapt itself with the changing network configurations. Certain nodes, known as *clusterheads*, are responsible for the formation of *clusters* (analogous to *cells* in a cellular network) and maintenance of the topology of the network. The set of clusterheads is known as a *dominant set*. A clusterhead does the resource allocation to all the nodes belonging to its cluster. Due to the dynamic nature of the mobile nodes, their association and dissociation to and from clusters perturb the stability of the network and thus reconfiguration of clusterheads is often unavoidable. This is an important issue since frequent clusterhead changes adversely affect the performance of other protocols such as scheduling, routing and resource allocation that rely on it. Choosing clusterheads optimally is an NP-hard problem [2]. Thus, existing solutions to this problem are based on heuristic approaches and none attempts to retain the topology of the network [2,4]. We believe a good clustering scheme should preserve its structure as much as possible when the topology is dynamically changing. Otherwise, re-computation of clusterheads and frequent information exchange among the participating nodes will result in high computation cost overhead.

The concept of dividing a geographical region into smaller zones has been presented implicitly in the literature as *clustering* [9]. A natural way to map a "standard" cellular architecture into a multi-hop packet radio network is via the concept of a virtual cellular network (VCN) [4]. Any node can become a clusterhead if it has the necessary functionality, such as processing and transmission power. Nodes register with the nearest clusterhead and become member of that cluster. Clusters may change dynamically, reflecting the mobility of the nodes. The focus of the existing research has been on just partitioning the network into clusters [2,3,7,8,6], without taking into consideration the efficient functioning of all the system components. The lack of rigorous methodologies for the design and analysis of peer-to-peer mobile networks has motivated in-depth research in this area. There exist solutions for efficiently interconnecting the nodes in such a way that the latency of the system is minimized while throughput is maximized [6]. Most of the approaches [2,4,6] for finding the clusterheads do not produce an optimal solution with respect to battery usage, load balancing and MAC functionality.

This paper proposes a weight based distributed clustering algorithm which takes into consideration the number of nodes a clusterhead can handle ideally (without any severe degradation in the performance), transmission power, mobility and battery power of the nodes. Unlike existing schemes which are invoked periodically resulting in high communication overhead, our algorithm is adaptively invoked based on the mobility of the nodes. More precisely, the clusterhead election procedure is delayed as long as possible to reduce the computation cost. We show by simulation experiments that our method yields better results as

compared to the existing heuristics in terms of the number of reaffiliations and dominant set updates.

2 Previous Work

To the best of our knowledge, three heuristics have been proposed to choose clusterheads in ad hoc networks. They include (i) Highest-Degree heuristic (ii) Lowest-ID heuristic and (iii) Node-Weight heuristic. In the assumed graph model of the network, the mobile terminals are represented as nodes and there exists an edge between two nodes if they can communicate with each other directly (i.e., one node lies within the transmission range of another). The performance of these heuristics were shown in [3,6] by simulation experiments where mobile nodes were randomly placed in a square grid and moved with different speeds in different directions. These heuristics are summarized below.

Highest-Degree Heuristic: This approach is a modified version of [10] which computes the degree of a node based on the distance between that node from others. A node x is considered to be a neighbor of another node y if x lies within the transmission range of y. The node with the maximum degree is chosen as a clusterhead and any tie is broken by the node ids which are unique. The neighbors of a clusterhead become members of that cluster and can no longer participate in the election process. This heuristic is also known as the highest-connectivity algorithm. Experiments demonstrate that the system has a low rate of clusterhead change but the throughput is low under this scheme. Typically, each cluster was assigned some resources which was shared among the members of that cluster on a round-robin basis [6,7,8]. As the number of nodes in a cluster is increased, the throughput of each user drops and hence a gradual degradation in the system performance is observed. This is the inherent drawback of this heuristic since the number of nodes in a cluster is not bounded.

Lowest-ID Heuristic: Gerla and Tsai [6] proposed a simple heuristic by assigning a unique id to each node and choosing the node with the minimum id as a clusterhead. However, the clusterhead can delegate its duties to the next node with the minimum id in its cluster. A node is called a *gateway* if it lies within the transmission range of two or more clusters. For this heuristic, the system performance is better compared with the Highest-Degree heuristic in terms of the throughput. Since the environment under consideration is mobile, it is unlikely that node degrees remain stable resulting in frequent clusterhead updates. The drawback of this heuristic is its bias towards nodes with smaller ids which leads to the battery drainage of certain nodes. Moreover, it does not attempt to balance the load uniformly across all the nodes.

Node-Weight Heuristic: Basagni et al. [2,3] assigned node-weights based on the suitability of a node being a clusterhead. A node is chosen to be a clusterhead if its weight is higher than any of its neighbor's node-weights. The smaller id is chosen in case of a tie. To verify the performance of the system [2], the nodes were assigned weights which varied linearly with their speeds but with negative slope. Results proved that the number of updates required is smaller than the

Highest-Degree and Lowest-ID heuristics. Since node weights were varied in each simulation cycle, computing the clusterheads becomes very expensive and there are no optimizations on the system parameters such as throughput and power control.

3 Our Approach

None of the above three heuristics leads to an optimal selection of clusterheads since each deals with only a subset of parameters which impose constraints on the system. For example, a clusterhead may not be able handle a large number of nodes due to resource limitations even if these nodes are its neighbors and lie well within its transmission range. Thus, the load handling capacity of the clusterhead puts an upper bound on the node-degree. In other words, simply covering the area with the minimum number of clusterheads will put more burden on each of the clusterheads. This will of course maximize the resource utilization. On the other hand, we could have all the nodes share the same responsibility and act as clusterheads. However, more clusterheads result in extra number of hops for a packet when it gets routed from the source to the destination, since the packet has to go via a larger number of clusterheads. Thus, this solution leads to higher latency, more power consumption and more information processing per node. The other alternative is to split the whole area into zones, the size of which can be determined by the transmission range of the nodes. This can put a lower bound on the number of clusterheads required. Ideally, to reach this lower bound, a uniform distribution of the nodes is necessary over the entire area. Also, the total number of nodes per unit area should be restricted so that the clusterhead in a zone can handle all the nodes therein. However, the zone based clustering is not a viable solution due to the following reasons.

The clusterheads would typically be centrally located in the zone, and if they move, new clusterheads have to be selected. It might so happen that none of the other nodes in that zone are centrally located. Therefore, to find a new node which can act as a clusterhead with the other nodes within its transmission range might be difficult. Another problem arises due to non-uniform distribution of the nodes over the whole area. If a certain zone becomes densely populated, the clusterhead might not be able to handle all the traffic generated by the nodes because there is an inherent limitation on the number of nodes a clusterhead can handle. We propose to select the minimum number of clusterheads which can support all the nodes in the system satisfying the above constraints.

In summary, choosing an optimal number of clusterheads which will yield high throughput but incur as low latency as possible, is still an important problem. As the search for better heuristics for this problem continues, we propose the use of a *combined weight* metric, that takes into account several system parameters like the ideal node-degree, transmission power, mobility and the battery power of the nodes.

3.1 Basis for Our Algorithm

The following features are considered in our clustering algorithm.

- The clusterhead election procedure is invoked as rarely as possible. This reduces system updates and hence computation and communication cost.
- Each clusterhead can ideally support M (a pre-defined threshold) nodes to ensure efficient MAC functioning. A high throughput of the system can be achieved by limiting or optimizing the number of nodes in each cluster.
- The battery power can be efficiently used within certain transmission range. Consumption of the battery power is more if a node acts as a clusterhead rather than an ordinary node.
- Mobility is an important factor in deciding the clusterheads. *Reaffiliation* occurs when one of the ordinary nodes moves out of a cluster and joins another existing cluster. In this case, the amount of information exchange between the node and the corresponding clusterhead, is local and relatively small. The information update in the event of a change in the dominant set is much more than a reaffiliation.
- A clusterhead is able to communicate better to its neighbors if they are closer to the clusterhead within the transmission range. This is due to signal attenuation with increasing distance.

3.2 Proposed Algorithm

Based on the preceding discussions, we propose an algorithm which effectively combines all the system parameters with certain weighing factors, the values of which can be chosen according to the system needs. For example, power control is very important in CDMA networks, thus the weight of that factor can be made larger. The flexibility of changing the weight factors helps us apply our algorithm to various networks. The procedure for clusterhead election is presented below. Its output is a set of nodes (dominant set) which forms the clusterheads for the network. According to our notation, the number of nodes that a clusterhead can handle ideally is M. The clusterhead election procedure is invoked at the time of system activation and also when the current dominant set is unable to cover all the nodes.

Clusterhead Election Procedure

Step 1: Find the neighbors of each node v (i.e., nodes within its transmission range). This determines the *degree*, d_v.
Step 2: Compute the *degree-difference*, $D_v = |d_v - M|$, for every node v.
Step 3: For all v, compute the *sum of the distances*, P_v, with all its neighbors.
Step 4: Compute the running average of the speed for every node. This gives a measure of its mobility and is denoted by M_v.
Step 5: Compute the total time, T_v, for which a node has been a clusterhead. T_v indicates how much battery power has been consumed which is more for a clusterhead than for an ordinary node.

Step 6: Calculate a *combined weight* $I_v = c_1 D_v + c_2 P_v + c_3 M_v + c_4 T_v$, for each node v. The coefficients c_1, c_2, c_3 and c_4 are the weighing factors for the corresponding system parameters.

Step 7: Choose v with the minimum I_v as the clusterhead. The neighbors of the chosen clusterhead can no longer participate in the election procedure.

Step 8: Repeat Steps 2 - 7 for the remaining nodes not yet assigned to any cluster.

3.3 System Activation and Update Policy

When a system is brought up, every node v broadcasts its id which is registered by all other nodes lying within v's transmission range, *tx_range*. It is assumed that a node receiving a broadcast from another node can estimate their mutual distance from the strength of the signal received. Thus, every node is made aware of its geographically neighboring nodes and their corresponding distances. Once the neighbors list is ready, our algorithm chooses the clusterhead for the first time. All the non-clusterhead nodes know the clusterhead they are attached to; similarly the clusterheads know their members. The topology of the system is constantly changing due to the movement of all the nodes, thus the system needs to be updated which may result in the formation of a new set of clusters. It may also result in nodes changing their point of attachment from one clusterhead to another within the existing dominant set, which is called *reaffiliation*. The frequency of update and hence reaffiliation is an important issue. If the system is updated periodically at a high frequency, then the *latest* topology of the system can be used to find the clusterheads which will yield a good dominant set. However, this will lead to high computational cost resulting in the loss of battery power or energy. If the frequency of update is low, there are chances that current topological information will be lost resulting in sessions terminated midway.

Every mobile node in any cellular system (GSM or CDMA) periodically exchanges control information with the base station. Similar idea is applied here, where all the nodes continuously monitor their signal strength as received from the clusterhead. When the mutual separation between the node and its clusterhead increases, the signal strength decreases. In that case, the mobile has to notify its current clusterhead that it is no longer able to attach itself to that clusterhead. The clusterhead tries to hand-over the node to a neighboring cluster and the member lists are updated. If the node goes into a region not covered by any clusterhead, then the clusterhead election procedure is invoked and the new dominant set is obtained.

The objective of our clusterhead election procedure is to minimize the number of changes in dominant set update. Once the neighbors list for all nodes are created, the degree-difference D_v is calculated for each node v. Also, P_v is computed for each node by summing up the distances of its neighbors. The mobility M_v is calculated by averaging the speed of the node. The total amount of time, T_v, a node remained as a clusterhead is also calculated. All these parameters are normalized, which means that their values are made to lie in a pre-defined range. The corresponding weights c_1, c_2, c_3 or c_4, which sum upto 1, are kept fixed for

a given system. The weighing factors also give the flexibility of adjusting the effective contribution of each of the parameters in calculating the *combined weight* I_v. For example, in a system where battery power is more important, the weight c_4 associated with T_v can be made higher. The node with the minimum total weight, I_v, is elected as a clusterhead and its neighbors are no longer eligible to participate in the remaining part of the election process which continues until every node is found to be either a clusterhead or a neighbor of a clusterhead.

3.4 Balancing the Loads

It is not desirable to have any clusterheads to be overly loaded while some others are lightly loaded. At the same time, it is difficult to maintain a perfectly balanced system at all times due to frequent detachment and attachment of the nodes from and to the clusterheads. As a measure of how well balanced the clusterheads are, we define the *load balancing factor* (LBF) which is inversely proportional to the variance of the cardinality of the clusters. In other words,

$$LBF = \left(\frac{\sum_i x_i - \mu}{n_c} \right)^{-1/2}$$

where n_c is the number of clusterheads, x_i is the cardinality of cluster i, and μ is the average number of nodes a clusterhead has. Thus, $\mu = \frac{N - n_c}{n_c}$, N being the total number of nodes in the system.

3.5 Connecting the Clusters

As a logical extension to clustering, we investigate the connectivity of the nodes which is essential for any routing algorithm. *Connectivity* can be defined as the probability that any node is reachable from any other node. For a single component graph, any node is reachable from any other node and the connectivity becomes 1. For two clusters to communicate with each other, we assume that the clusterheads are capable of operating in *dual* power mode. It uses low power to communicate with its members within its transmission range and high power to communicate with the neighboring clusterheads because of higher range.

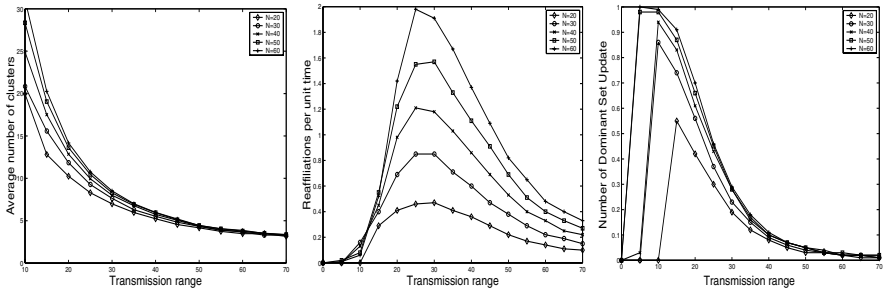

Fig. 1. *max_disp*=5

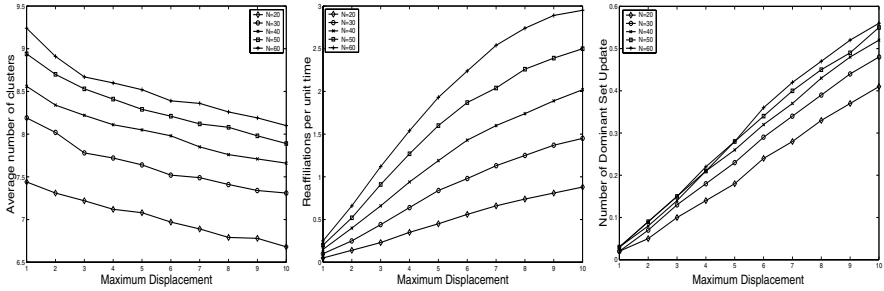

Fig. 2. *tx_range*=30

4 Simulation Study

We simulate a system with N nodes on a 100×100 grid. The nodes could move in all possible directions with displacement varying uniformly between 0 to a maximum value (*max_disp*), per unit time. To measure the performance of our system, we identify three metrics: (i) the number of clusterheads, (ii) the number of reaffiliations, and (iii) the number of dominant set updates. Every time a dominant set is identified, its cardinality gives the number of clusterheads. The reaffiliation count is incremented when a node gets dissociated from its clusterhead and becomes a member of another cluster within the current dominant set. The dominant set update takes place when a node can no longer be a neighbor of any of the existing clusterheads. These three parameters are studied for varying number of nodes in the system, transmission range and maximum displacement. We also study how the load balance factor changes as the system evolves and how well connected the nodes are.

4.1 Summary of Experimental Results

In our simulation, N was varied between 20 and 60, and the transmission range was varied between 0 and 70. The nodes moved randomly in all possible directions with a maximum displacement of 10 along each of the coordinates. Thus, the maximum Euclidean displacement possible is $10\sqrt{2}$. We assume that each clusterhead can ideally handle 10 nodes in its cluster in terms of resource allocation. Therefore, the *ideal degree* was fixed at $M = 10$ for the entire experiment. Due to this, the weight associated with D_v was rather high. The next higher weight was given to P_v, which is the sum of the distances. Mobility and battery power were given low weights. The values used for simulation were $c_1 = 0.7$, $c_2 = 0.2$, $c_3 = 0.05$ and $c_4 = 0.05$. Note that these values are arbitrary at this time and should be adjusted according to the system requirements.

Figure 1 shows the variation of three parameters, namely average number of clusterheads, reaffiliation per unit time and the number of dominant set update with varying transmission range and a constant *max_disp* of 5. We observe that the average number of clusterheads decreases with the increase in the transmission range because a clusterhead with a large transmission range covers a larger

area. For low transmission range, the nodes in a cluster are relatively close to the clusterhead, and a detachment is unlikely. The number of reaffiliations increases as the transmission range increases, and reaches a peak when transmission range is between 20 and 30. Further increase in the transmission range results in a decrease in the reaffiliations since the nodes, in spite of their random motion, tend to stay inside the large area covered by the clusterhead. The dominant set updates is high for a small transmission range because the cluster area is small and the probability of a node moving out of its cluster is high. As the transmission range increases, the number of dominant set updates decreases because the nodes stay within their cluster in spite of their movements. Figures 2 shows the variation of the same parameters but for varying *max_disp* and constant transmission range of 30. The average number of clusterheads is almost the same for different values of *max_disp*, particularly for larger values of N. This is because, no matter what the mobility is, it simply results in a different configuration, but the cluster size remains the same. We also see how the reaffiliations change per unit time with respect to the maximum displacement. As the displacement becomes larger, the nodes tend to move further from their clusterhead, detaching themselves from the clusterhead, causing more reaffiliation per unit time and more dominant set updates.

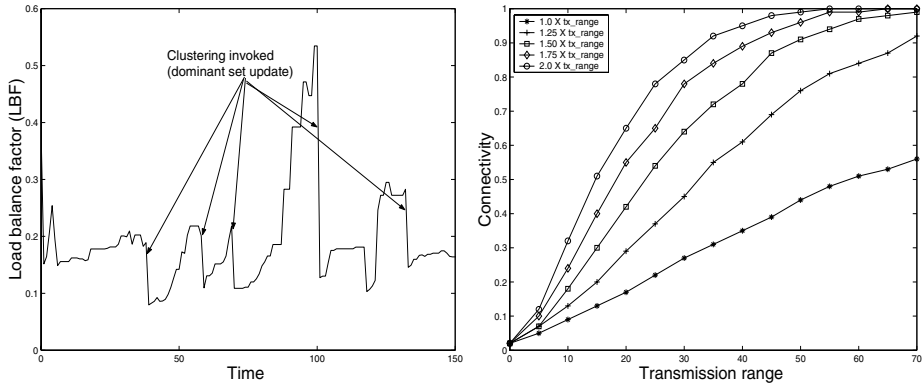

Fig. 3. Load distribution and connectivity

The non-periodic invocation of the clustering algorithm and the reachability of one node from another can be observed from Figure 3. There is a *gradual* increase in the load balance factor (LBF) due to the diffusion of the nodes among clusters. The improvement in LBF does not increase indefinitely because the nodes tend to move away from all possible clusterheads and the clustering algorithm has to be invoked to ensure connectivity. The clustering algorithm tries to connect all the nodes at the cost of load imbalance which is represented by the *sharp* decrease in LBF. As mentioned earlier, the lower power is used to communicate within the cluster whereas the higher power which effectively gives a higher transmission range is used to communicate with the neighboring clusterheads. To obtain the higher transmission range, we scaled the lower transmission

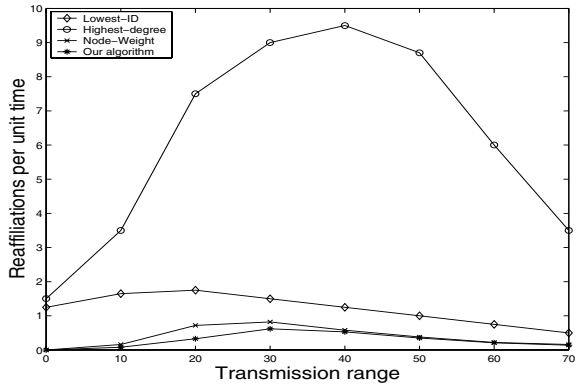

Fig. 4. Comparison of reaffiliations, N=30

range by a constant factor. Simulation was conducted for $N = 50$ and the constant factor was varied from 1.0 to 2.0 with increments of 0.25. It is observed that a well connected graph can be obtained at the cost of high power.

Figure 4 shows the relative performance of the Highest-Degree, Lowest-ID, Node-Weight heuristics and our algorithm in terms of reaffiliations per unit time. The number of reaffiliations for our algorithm is at most half the number obtained from the Lowest-ID. The main reason is that the frequency of invoking the clustering algorithm is lower in our case resulting in longer duration of stability of the network. Our algorithm performs marginally better than the Node-Weight heuristics which, however, does not give the basis of assigning the weights to the nodes. Our algorithm describes a linear model which takes into consideration the four important system parameters in deciding the suitability of the nodes acting as clusterheads. It also provides the flexibility of adjusting the weighing factors according to the system needs.

5 Conclusions

We propose a weight based distributed clustering algorithm which can dynamically adapt itself with the ever changing topology of ad hoc networks. Our algorithm has the flexibility of assigning different weights and takes into an account a combined effect of the ideal degree, transmission power, mobility and battery power of the nodes. The algorithm is executed only when there is a demand, i.e., when a node is no longer able to attach itself to any of the existing clusterheads. We see that there is a pattern of how the load balance factor changes to distribute the load and also ensure maximum connectivity. Our algorithm performs significantly better than both of the Highest-Degree and the Lowest-ID heuristics. In particular, the number of reaffiliations for our algorithm is about 50% of the number obtained from Lowest-ID heuristic. Though our approach performs marginally better than the Node-Weight heuristic, it considers more realistic system parameters and has the flexibility of adjusting the weighing factors.

References

1. D.J. Baker, A. Ephremides, and J.A. Flynn, "The design and simulation of a mobile radio network with distributed control", IEEE Journal on Selected Areas in Communications, vol. SAC-2, January 1984, pp. 226-237.
2. S. Basagni, I. Chlamtac, and A. Farago, "A Generalized Clustering Algorithm for Peer-to-Peer Networks", Workshop on Algorithmic Aspects of Communication (satellite workshop of ICALP), Bologna, Italy, July 1997.
3. S. Basagni, "Distributed Clustering for Ad Hoc Networks", International Symposium on Parallel Architectures, Algorithms and Networks", Perth, June 1999, pp. 310-315.
4. I. Chlamtac and A. Farago, "A New Approach to the Design and Analysis of Peer-to-Peer Mobile Networks", Wireless Networks, 5(3), Aug 1999, pp. 149-156.
5. A. Ephremides, J.E. Wieselthier, and D.J. Baker, "A design concept for reliable mobile radio networks with frequency hopping signaling", Proceedings of IEEE, vol. 75, Jan 1987, pp. 56-73.
6. M. Gerla and J. T.C. Tsai, "Multicluster, Mobile, Multimedia Radio Network", Wireless Networks, 1(3) 1995, pp. 255-265.
7. C.-H.R. Lin and M. Gerla, "A Distributed Control Scheme in Multi-hop Packet Radio Networks for Voice/Data Traffic Support", IEEE GLOBECOM, pp. 1238-1242, 1995.
8. C.-H. R. Lin and M. Gerla, "A Distributed Architecture for Multimedia in Dynamic Wireless Networks", IEEE GLOBECOM, pp. 1468-1472, 1995.
9. M. Joa-Ng and I.-T. Lu, "A Peer-to-Peer Zone-based Two-level Link State Routing for Mobile Ad Hoc Networks", *IEEE Journal on Selected Areas in Comm.*, Aug. 1999, pp. 1415-1425.
10. A.K. Parekh, "Selecting routers in ad-hoc wireless networks", ITS, 1994.