Connectivity Based *k*-hop Clustering in Wireless Networks

Geng Chen[1], Fabian Garcia Nocetti[2], Julio Solano Gonzalez[2] and Ivan Stojmenovic[1,2]
[1]SITE, University of Ottawa, Ottawa, Ontario K1N 6N5, Canada ivan@site.uottawa.ca
[2]DISCA, IIMAS, UNAM, A.P. 20-726, Del. A. Obregon, Mexico D.F. 01000, Mexico

Abstract

In this paper we describe several new clustering algorithms for nodes in a mobile ad hoc network. We propose to combine two known approaches into a single clustering algorithm which considers connectivity as a primary and lower ID as secondary criterion for selecting clusterheads. The goal is to minimize number of clusters, which leads toward dominating sets of smaller sizes (this is important for applications in broadcasting and Bluetooth formation). We also describe algorithms for modifying cluster structure in the presence of topological changes. Next, we generalize the cluster definition so that a cluster contains all nodes that are at distance at most k hops from the clusterhead. The efficiency of four clustering algorithms (k-lowestID and k-CONID, k=1 and k=2) is tested by measuring the average number of created clusters, the number of border nodes, and the cluster size in random unit graphs. The most interesting experimental result is stability of the ratio of the sum of CHs and border nodes in the set. It was constantly 60-70% for 1-lowestID and 46-56% for 1-ConID, for any value of n (number of nodes) and d (average node degree). Similar conclusions and similar number were obtained for k=2. We also proposed an unified framework for most existing and new clustering algorithm where a properly defined weight at each node is the only difference in otherwise the same algorithm. Finally, we propose a framework for generating random unit graphs with obstacles.

1. Introduction

Mobile ad hoc networks consist of wireless hosts that communicate with each other in the absence of a fixed infrastructure. Examples include battlefield scenarios, disaster relief and short-term scenarios such as public events. Routes between two hosts in the network may consist of hops through other hosts in the network. The task of finding and maintaining routes in an ad hoc network is nontrivial since host mobility causes frequent unpredictable topological changes. In highly mobile situation, the flooding scheme is the most reliable for sending data packets. However, since the link channel and battery power resources are very scarce, more efficient schemes must be devised. These schemes require up to date information about the location of nodes. Storage is not a critical issue since memory continues to get less expensive each year. The savings in communication bandwidth and energy come from reporting only to nodes that need a particular information. To reduce the transmission overhead for the update of routing tables after topological changes, it was proposed to divide all nodes into clusters. The overhead of cluster formation and maintenance can not be ignored. In the general cluster-based schemes for ad hoc networks, clusters are formed at first, and one clusterhead (CH) is elected for each cluster, in the fully distributed fashion [GT]. In cluster based approaches [GT, KHC, KK, L1, RBS, S, TRTN], the sender must know the location information of the cluster within which the destination is located. Routing algorithm may consist of routing from source to its CH, from the CH to the CH of destination node, and from the later node to the destination. Communication between CHs involves intermediate nodes in their clusters. To reduce the power consumption in CH nodes, the information about all CHs may be replicated in all the nodes of the network. Therefore each node stores the information about all the clusters (more precisely, about CHs) in the network. Each node knows the content (i.e. the list of nodes) only for its own cluster. The sender may forward the directly towards destination's CH, and does not need to 'consult' its CH. Moreover, the routing paths do not necessarily have to pass through any of the CHs, since the message can be rerouted toward the next cluster as soon as it enters any of the clusters.

Ad hoc networks are best modeled by *unit* graphs constructed in the following way. Two nodes *A* and *B* in the network are neighbors if the Euclidean distance between them is at most *R*, where *R* is the transmission radius which is same for every node.

The efficiency of a distributed clustering algorithm is measured by the number of clusters and border nodes that it produces. Our goal is to minimize that number, with application in broadcasting [SSZ] (minimize number of message retransmisisons) and scatternet formation in Bluetooth [LS, BKNR] (minimize number of piconets) as justification ([SSZ, BKNR] apply modified clustering to solve their respective problems).

Section 2 gives literature review, section 3 describes combined higher connectivity lower ID clustering algorithm, section 5 describes a new cluster update structure. Performance evaluation is given in section 5. Section 6 describes a unified framework for clustering algorithms, while section 7 discusses the generation of random unit graphs with obstacles. Preliminary version of this paper was published in [GS]. A variant of *k-lowestID* scheme was proposed independently in [APVH], which also shows that minimum *d*-hop dominating set problem is NP-complete.

2. Literature review

Hierarchical network structure is an effective way to organize a network comprising a large number of nodes. In a single hierarchy, nodes are divided into clusters, which may or may not have clusterheads. It is suitable for networks with few hundred nodes. A multi-level hierarchy [L1, L2, SW] has nodes organized in a tree-like fashion with several levels of clusterheads. A three level hierarchy employs ordinary nodes, clusterheads and super-clusterheads, and is suitable for networks with few thousand nodes. In this paper we shall study only single level hierarchies.

Early literature [EFB, L1, KK, S, TRTN] on clustered networks assumes that the CHs are predetermined (military chiefs, for instance) and that ordinary nodes simply join themselves to a primary cluster and two or three secondary ones. The only references that actually discuss the clustering problem are [EWB, GT, KVCP, LG, P, S, RBS]. Shacham [S] discussed only regular graph structures while [RBS] employs a cluster controller or leader (therefore algorithm is not distributed).

A multi-cluster, multihop packet radio network architecture is presented by Gerla and Tsai [GT]. Nodes are organized into cluster by using one of two existing distributed clustering algorithms. In the lowest-ID algorithm [EWB], a node which only hears nodes with ID higher that itself is a clusterhead (CH). The lowest-ID node that a node hears is its clusterhead, unless the lowest-ID specifically gives up its role as CH (deferring to a yet lower ID node). A node which can hear two or more CHs is a 'gateway'. Otherwise, a node is an ordinary node.

We observe that in some cases the algorithm, as described, may fail to cluster the nodes. Suppose that the x-axis of each node serves as its ID (that is, nodes have IDs whose relative sizes follow, by chance, the x-axis). Then the CHs are only nodes which do not have any neighbor on their left. However, many graphs have only one such node (for instance, the interval graphs, e.g. cars moving on a highway) and the rule produces only one CH for the whole graph. Another important example is the triangular graphs (e.g. graph of base stations in wireless phone network). In general, the rule does not even guaranty finding a CH to each node in any predefined hop distance from it, if IDs are ordered as their x-axis.

The second clustering algorithm used in [GT] is a modified version of algorithm from [P], in which the highest degree node in a neighborhood becomes the clusterhead. More precisely, such nodes are elected as CHs, and their neighbors are then covered. The process then continues for the remaining uncovered nodes. An uncovered node is elected as a clusterhead if it is has the highest degree among all its uncovered neighbors. Although the algorithm is expected to perform well on many randomly defined graphs (as reported in [GT]), it may not produce any CH for graphs which do not have any node with the highest number of neighbors (like above mentioned interval and triangular graphs). Thus the algorithm must be completed by adding nontrivial tie resolution rules.

Both algorithms in [GT] have the properties that clusterheads are not directly linked, and each clusterhead is directly linked to every other node in its cluster. Thus each node is either clusterhead itself or is directly linked to one or more clusterheads. Such clusters are refereed to as 1-clusters. The role of clusterheads in [GT] is to control channel access (using a combination of TDMA within the cluster and CDMA among clusters), perform power measurements, maintain time division frame synchronization, and to guaranty bandwidth for real time traffic. [GT] uses any existing routing algorithm (e.g. Bellman-Ford) for sending messages between nodes (that is, routing decisions do not depend on cluster organization).

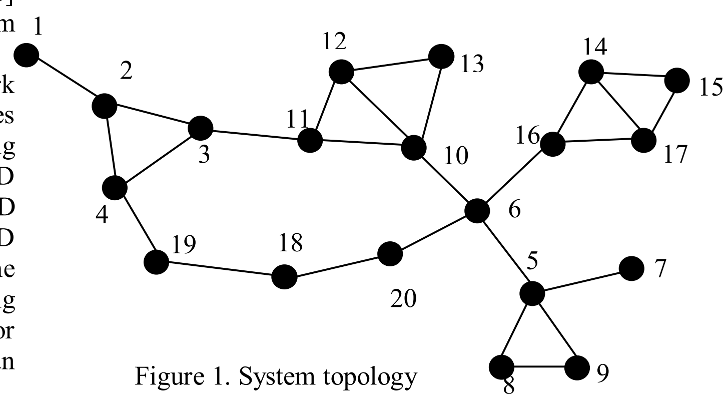

Figure 1. System topology

Lin and Gerla [LG] described a modified version of lowest ID algorithm that resolved the problems mentioned above. Each node in the network broadcasts its clustering decision exactly once. The distributed clustering algorithm [LG] is initiated by all nodes whose ID is lowest among all their neighbors (local lowest ID nodes). They broadcast their decision to create clusters (with them as CHs) to all their neighbors. Each node may hear the broadcasts by its neighbors and select the lowest ID among neighboring CHs, if any. If all neighbors which have lower ID sent their decisions and none declared itself a CH, the node decides to create its own CH and broadcasts its ID as cluster ID. Otherwise, it chooses neighboring CH with lowest ID, and broadcasts such decision. Thus each node broadcasts its clustering decisions after all its neighbors with lower IDs

Proceedings of the 35th Annual Hawaii International Conference on System Sciences (HICSS-35'02)
0-7695-1435-9/02 $17.00 © 2002 IEEE

have already done so. Every node can determine its cluster and only one cluster, and transmits only one message during the algorithm. For example, the clustering algorithm for the topology in Fig. 1 produces clusters as indicated in Fig. 2 (CHs are in bold).

The algorithm creates non-overlapping clusters by requesting nodes to select one of several neighboring CHs. We note that the node may still be linked to other CHs, and thus the clustering organization is essentially overlapping. The maintenance of clusters is performed in the following way. Within each cluster, nodes must be able to communicate with each other in at most two hops. Incoming nodes that preserve the property may join the cluster. When a link is disconnected, the highest connectivity node and its neighbors stay in the original cluster. Thus this node effectively takes over the clusterhead role from the lowest ID node. Other nodes from the former cluster shall either join another cluster or form their own cluster. This may lead to single node clusters. Thus it seems that additional procedures for merging or rearranging clusters may be desirable. Further, clusterhead role in [LG] is only important for clustering formation, and is not used in routing decisions.

than one cluster. Each node maintains a list of all its neighbors, a list of clusters, a list of boundary nodes in the network, and the routing table with next hop to each destination. Therefore for larger networks the amount of information to be updated at each node when they move is significant and imposes significant overhead on the communication bandwidth. 'The memory requirements are substantial, so are the control traffic required to maintain and update routing tables…Cluster management results in further overheads' [T, p. 132]. In the network of boundary nodes [KVCP], two boundary nodes will have a link between them if they have common clusters. Routing from one node to another consists of routing inside a cluster and routing from cluster to cluster. More precisely, the routing algorithm in [KVCP] is a shortest path algorithm (e.g. Dijkstra's algorithm) that runs on this connected network of boundary nodes ([KVCP], p. 58). Therefore the routing algorithm in [KVCP] does not require the information about clusters since it makes decision solely based on the list of boundary nodes and their connectivity. Procedures for maintaining cluster information thus merely serve to update the list of boundary nodes.

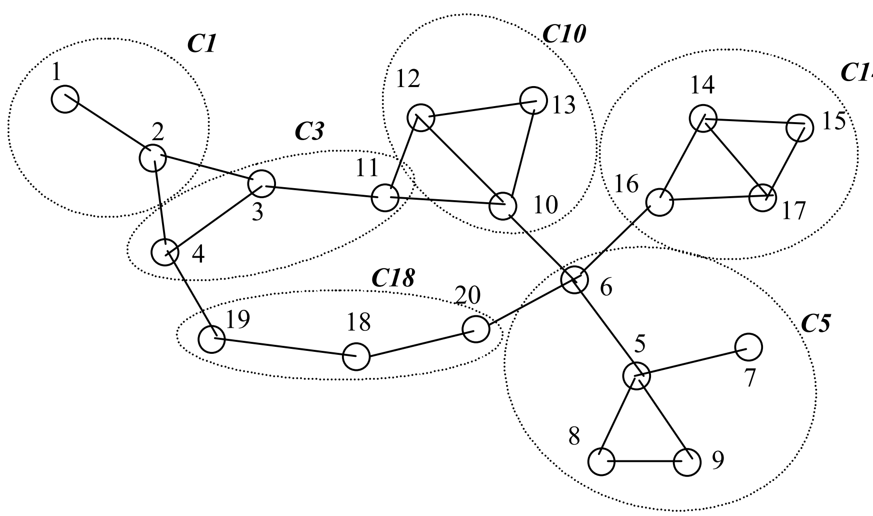

Figure 2. Lowest ID clustering

Krishna, Vaidya, Chatterjee and Pradhan [KVCP] recently proposed a cluster based approach (with single level of hierarchy) for routing in dynamic network. A *k*-cluster is defined in [KVCP] to be a subset of nodes which are mutually 'reachable' by a path of length at most *k* for some fixed *k*. A *k*-cluster with *k=1* is a clique. Each maximal clique in the graph serves as a cluster, and clusters have no clusterheads. The graph is divided into a number of overlapping clusters. [KVCP] presented algorithms for creation of clusters, as well as algorithms to maintain them in the presence of various network events. A node is *boundary* node if it belongs to more

The clustering and routing algorithms in [KVCP] are not fully distributed, and do not adapt well in case of 'sleeping' nodes. In fact, the temporary inactivity of any node requires the update of information in all nodes of network. In case of interval graphs (nodes on a highway) where each node may hear only its two neighbors, each edge is a separate cluster; that is, the number of clusters is equal to number of nodes. Each node of triangular graphs belongs to six clusters, and each cluster is a triangle consisting of three nodes. Even after redundant clusters are removed, the number of clusters is equal to number of nodes. Therefore the information about clusters for many graphs exceeds even the amount of information needed to simply store routing tables to each node in non-clustered approaches.

Sivakumar, Das and Bhargavan [SDB] proposed a series of routing algorithms for ad hoc wireless network. The idea is to identify a subnetwork that forms a minimum connected dominating set (*MCDS*) based on clustering. Each node in the subnetwork is called a spine (the corresponding notion of internal nodes). Their algorithm for determining spine nodes requires 2-hop neighborhood information, and involves running minimum spanning tree algorithm on weighted edges. It is a variation of [LG], with proper weight function for choosing CHs, called spine nodes in [SDB]. To choose CHs, record *(dCH,d,ID)* is formed, where *dCH* is number of nodes assigned to given CH (it is 0 if node is not CH). It is the primary key; degree *d* is secondary key, and *Id* number is the ternary key. The algorithm [SDB] has lower time complexity but higher message complexity than the algorithm in [KVCP]. Further, in order to compute a routing table, each *MCDS* node needs to know entire network topology. An all pairs shortest path algorithm is actually running on *G,* not on the reduced subnetwork of *MDCS* nodes. Therefore, it may loose part of original goal of network centralization.

Basagni [B1] proposed to use nodes' weights instead of *lowestID* or node degrees in clusterhead (CH) decisions. The algorithm is a variation of algorithm by Lin and Gerla [LG], where *lowestID* is replaced by largest weight as a criterion for CH decision. Weight is defined by mobility related parameters, such as speed. Algorithm is modified to initially cluster nodes even while they move, and to update the clustering afterwards.

In [B2], Basagni further generalized [B1], by allowing each CH to have at most *k* neighboring CHs (instead of none), and by reducing number of reallocations by introducing threshold parameter *h* (that is, there is no reallocation unless CH candidate has weight more than by h greater than the weight of current CH). The simulation measures the clustering stability, i.e. the number of elections and reaffiliations per tick. The first set of experiments is with weights associated to nodes' speed, while the second one has weights represented by nodes' transmission powers.

In [B3], Basagni described an algorithm for finding a maximal weighted independent set in wireless networks, and is based on clustering algorithm from [B1,B2]. It proves that the (worst case) time complexity is proportional to the number of clusters created, that (worst and average case) message complexity is proportional to the number of nodes, and that the average time complexity is logarithmic in number of nodes for random graphs. In [B4], Basagni selects node weights in clustering process so that the CHs created in the process result in a maximum weighted independent set.

A breadth first search based clustering scheme where CHs are not directly linked to each node within their clusters, and each cluster size is between *k* and *2k*, is given in [BK].

There are two approaches for routing in clustered networks. The strict hierarchical routing [KK, TRTN] approach uses tier routing within a cluster and link-state routing among clusters. In other words, shortest paths between CHs are precomputed and followed by given packet cluster by cluster. In a quasi-hierachical routing approach [KK], tier routing is enhanced by including the minimum distance to other radios in the cluster, and to other clusters. Thus the packet is sent directly toward the destination's cluster, using available information. We observe that fully distributed routing algorithms, which use only information about all neighbors and destination (or destination's CH) will not differ significantly. Upon entering destination's cluster, the packet is simply redirected toward destination.

Kim, Ha and Choi [KHC] define *k*-cluster as the set of all the nodes within distance at most *k* hops from a given node, referred to as the clusterhead of the *k*-cluster. We shall adopt the same definition in our paper. Border nodes are nodes that belong to two or more clusters. Clusters are formed by using the lowest ID algorithm [GT]. [KHC] measured the ratio of border nodes in a cluster over number of cluster members (for *n*=30 nodes) and found decreasing ratio. They propose *k*-hop cluster-based dynamic source routing scheme, in which the sender can transmit its data packets to its destinations after acquiring the route to the destination by performing route discovery procedure similar to that of dynamic source routing [BMJHJ]. The route discovery time is reduced by flooding the discovery packet to border hosts only, if the destination is not in the current cluster.

Section 8 describes recent clustering based flooding and Bluetooth scatternet formation algorithms.

2. A combined higher connectivity lower ID clustering algorithm

We shall refer to the algorithm of Lin and Gerla [LG] as the 1-lowestID clustering algorithm. First, we will generalize the same distributed algorithm to define *k*-clusters, and will call the clustering algorithm *k*-lowestID one. One of the nodes initiates the clustering process by flooding request for clustering to all the other nodes. Assume that all nodes are aware of their *k*-hop neighbors (that is, neighbors at distance at most *k* hops). All nodes whose *ID* is lowest among all their k-hop neighbors (local lowest *ID* nodes) broadcast their decision to create clusters (with them as CHs) to all their *k*-hop neighbors. Thus their decision (and similarly the decisions of other nodes later on) is retransmitted by other nodes until all nodes at distance up to k hops are reached. If all k-hop neighbors which have lower *ID* broadcasted their decisions and none declared itself a CH, the node decides to create its own CH and broadcasts its *ID* as cluster *ID.* Otherwise, it chooses a k-hop neighboring CH with lowest *ID,* and broadcasts such decision. Thus each node broadcasts its clustering decisions after all its k-hop neighbors with lower *ID*s have already done so. Every node can determine its cluster and only one cluster, and initiates the broadcast for only one message during the algorithm.

The lowest *ID* algorithm does not take into account the connectivity of nodes, and therefore may produce more clusters than necessary. The pure connectivity based algorithm (when *ID* is replaced by node degrees) does not work properly because of numerous ties between nodes. We propose to use node degree as the primary key, and *ID* as the secondary key in cluster decisions. Node degree is the connectivity measure for 1-clusters.

We generalize the connectivity to count all k-hop neighbors of given node. For k=1, the connectivity is equivalent to node degree. Therefore, whenever the connectivities are same, we compare *ID* to make the decision. The clustering algorithm, refereed to as the k-CONID (k-hop connectivity *ID*) algorithm, works as follows. Each node is assigned a pair *did=(d,ID),* containing its connectivity d and *ID,* which will be also called clusterhead priority. *Let did'= (d', ID')* and did"= (d", ID"). Then *did'>did"* if *d'> d"* or *d'=d"* and *ID' < ID".* That is, a node has clusterhead priority over the other if it has higher connectivity or, in case of equal connectivity, has lower *ID.* One of reasons to reduce the number of clusters and border nodes is to reduce the overhead of broadcasting task, where message initiated at a source is retransmitted by only CHs and border node. Such application of highest degree clustering for k=1 is given in [SSZ].

One of the nodes initiates the clustering process by flooding request for clustering to all the other nodes. All nodes whose clusterhead priority is largest among all their k-hop broadcast their decision to create clusters (with them as CHs) to all their k-hop neighbors. If all k-hop neighbors that have larger clusterhead priority broadcasted their decisions and none declared itself a CH, the node decides to create its own CH and broadcasts its *did* as cluster ID. Otherwise, it chooses a k-hop neighboring CH with largest clusterhead priority, and broadcasts such decision. Thus each node broadcasts its clustering decisions after all its k-hop neighbors with larger *did* have already done so. Every node can determine its cluster and only one cluster, and initiates the broadcast for only one message during the algorithm. However, we will assume that clusters may overlap, and thus each node belongs to all clusters whose CH is at k-hop distance from the node. Nodes that belong to more than one cluster are border nodes. For example, the clustering algorithm applied on the topology in Fig. 1 produces clusters as indicated in Fig. 3.

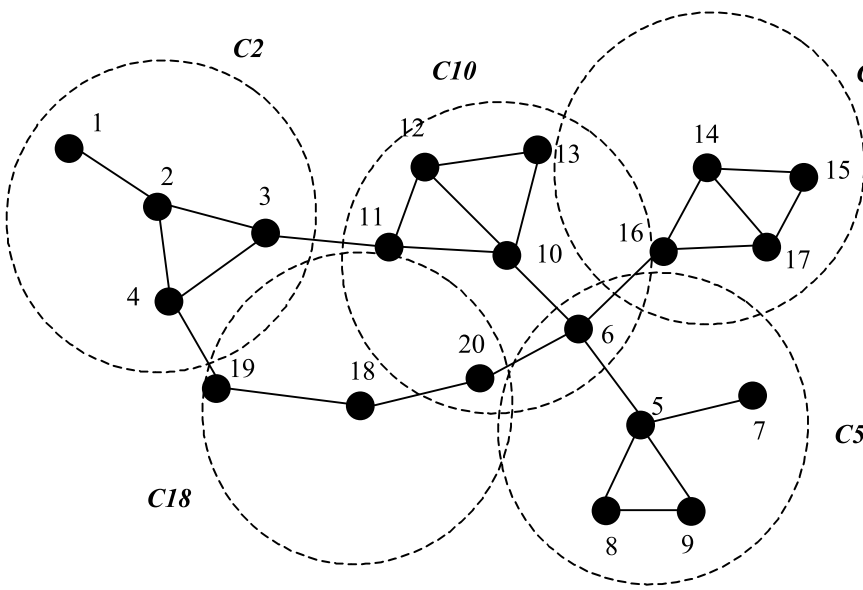

Figure 3. *CONID* clustering

4. Updating cluster structure

In this section, we shall describe algorithms for modifying cluster structure in the presence of topological changes. The maintenance procedures by Lin and Gerla [LG] are modified here. There are four cases to consider: a node switches on and joins the network, a node switches off and leaves the network, a link is disconnected, and a link between two existing nodes is formed after they moved closer to each other.

When a node switches on, it checks whether it is at distance up to k hops from any of existing clusterheads, and, if so, joins these clusters. Otherwise, the node creates a new cluster with itself as clusterhead, and invites its k-hop neighbors to join the cluster. This procedure does not differ from [LG] (for k=1).

When a node switches off, no change is made if the node was not a CH. In case of CH failure, nodes in the cluster elect a new CH using the number of k-hop neighbors within the cluster as the main criterion (overall number of k-hop neighbors as secondary, and *ID* as ternary criterion). Nodes that are not included in the new cluster repeat this procedure until all of them are included in a cluster. This procedure may result in splitting a cluster into two or more new clusters, and is similar to the one proposed in [LG] (for k=1).

If an existing link is disconnected, no change is made if the two nodes belong to separate clusters. Otherwise, all nodes in their cluster are informed, and their CH verifies whether all nodes in the cluster are still k-hop neighbors. If so, no change is made. Otherwise, nodes with hop count from CH greater than k (in some cases they may even become disconnected) create new cluster(s).

Finally, if a new link is created between two nodes *A* and *B,* there are several cases to consider. If none of *A* and *B* is a clusterhead, no change in the structure is made. However, this is a correct procedure only for k=1. For $k>1$, two clusterhead may reduce their mutual distance to at most k-hops, which violates the definition of k-clusters. The details of update procedure in this case are not trivial, and are omitted. In practice, it is not expected to use values of k that are greater than 2, and details for k=2 may be described in a straightforward way. If both *A* and *B* are clusterheads, first decide which of them preserves the role, using the same criteria outlined above. Nodes from the other cluster join the winning CH if they are its new k-hop neighbors (note that for k=1 no such node exists). Otherwise, they create new cluster(s).

As observed for updates described in [LG], the described maintenance procedures may, after repeated use, produce a poor quality of cluster structure. After any of these events the cluster structure is modified, and its quality is evaluated by each node. The quality of a cluster may be measured by its size (the number of nodes), and the ratio of border nodes in it. We propose three possible evaluation results: excellent quality, moderate quality, and poor quality. The two thresholds may be decided by some criteria that may depend on the total number of nodes and also on the movement pattern and expected frequency of restructuring clusters for selected thresholds. For example, a cluster containing only clusterhead and border nodes is of poor quality. After any of above described basic update procedures, nodes that belong to new clusters of poor quality will initialize a global restructuring process, following one of algorithms from previous two sections. In order to reduce the amount of overhead involved, clusters of excellent quality will reject and ignore the request. This will restrict the changes to the neighborhood of poor clusters only. Clusters of moderate quality participate in the restructuring, hoping to improve their quality. Some clusters may be unable to improve their poor quality without the cooperation from clusters of excellent quality. They will then accept the global favorable status and refrain from further requests until a new change in their cluster emerges.

Note that a recent paper [HT] also proposes some mobility and access based cluster update schemes.

5. Performance evaluation

The efficiency of the clustering algorithms is tested by measuring the average number of created clusters, the average ratio of border nodes, and the average cluster size. Parameters that define a networking context, in case of static nodes, are network size n (the number of nodes), and network connectivity d (the average degree of a node, that is the average number of neighbors of a node), which is related to the transmission range. The experiments were carried using random unit graphs. Each of n nodes is chosen by selecting its x and y coordinates at random in the interval $[0,100)$. The radius R is then increased until the graph becomes connected. It is further increased to increase network connectivity. In order to control the average node degree d, we sort all $n(n-1)/2$ (potential) edges in the network by their length, in increasing order. The radius R that corresponds to chosen value of d is equal to the length of $nd/2$-th edge in the sorted order. The advantage of this simple method of generating unit random graphs is in its full randomness, while the disadvantage is the difficulty in generating such graphs for small values of d. The DFS traversal was used to test whether a graph is connected. We experimented with the following network sizes: n= 50, 100, 200, 500, 1000. For each selected network size and degree, we generated ten random unit graphs. The minimum average degree tested was d=4 for n=50, 100 and 200 and d=5 for n=500, 1000. The maximum average degree tested was 12.

Tables 1 and 2 show the experimental results obtained for *1-lowestID, 1-CONID, 2-lowestID* and *2-CONID* algorithms. CH% denotes the ratio of clusterhead nodes (that is, the number of clusters divided by the total number of nodes). Similarly, B% denotes the ratio of border nodes, where border nodes are nodes that belong to more than one cluster (that is, which are at distance at most k hops from at least two CHs). Table 1 gives results for *1-lowestID* and *1-CONID* algorithms, for n=200 nodes. Their ratios of CHs and border nodes are given in the middle column. The average size of a cluster (the average number of nodes in a cluster) is denoted by AS in Table 2. The number of nodes and degree and denoted by n and d, respectively. C% is the ratio of the number of clusters created by *2-CONID* and

2-lowestID algorithms. Similarly, A% is the ratio of AS numbers in both algorithms.

The results clearly indicate significant advantage of *CONID* algorithm over *lowestID* one. The number of clusters generated by *1-CONID* algorithm is between 17% and 27% lower than the number of clusters generated by *1-lowestID* algorithm, while the average size is greater by 8%-23%. The differences between *2-CONID* and *2-lowestID* algorithm are smaller, but still significant. The number of clusters generated by *2-CONID* is 6%-25% smaller and the average cluster size is 2%-12% greater in *2-CONID* algorithm compared to *2-lowestID* one.

A closer analysis of the obtained results reveals several linear relations. For example, the number of clusters obtained by any algorithm is a linear function of the size of the network (when degree is fixed), or of the network degree (when size is fixed). Diagram 1 shows one of these relations. The most interesting experimental result is stability of the ratio of the sum of CHs and border nodes in the set. It was constantly 46-56% for *1-ConID* and 60-70% for *1-lowestID,* for any value of *n* and *d,* with ratio in 70-88% range. Similar conclusions and even ratios were obtained for k=2. Therefore increasing *k* did not reduce the ratio of CHs and border nodes together. However, it did reduce ratio of CHs, with the ratio of border nodes being increased by almost same amount.

	LowestID			*ConID/LowestID %*	*ConID*		
d	CH	B	CH+B	%	CH+B	CH	B
4	0.30	0.33	0.66	77	0.51	0.25	0.26
5	0.27	0.37	0.64	73	0.47	0.21	0.26
6	0.24	0.40	0.64	80	0.51	0.19	0.32
7	0.22	0.43	0.65	80	0.52	0.17	0.35
8	0.20	0.45	0.65	77	0.50	0.16	0.34
9	0.19	0.49	0.68	81	0.55	0.15	0.40
10	0.17	0.47	0.64	81	0.52	0.15	0.37
11	0.16	0.50	0.66	83	0.55	0.13	0.41
12	0.15	0.54	0.69	80	0.55	0.12	0.43

Table 1. Ratios of CHs and border nodes in *LowestID* and *ConID* algorithms for *n*=200 nodes

N	D	2-LOWEST ID			2-CONID			COMPARISON	
		CH%	AS	B%	CH%	AS	B%	C%	A%
100	4	0.17	8.7	0.36	0.15	9.25	0.26	0.86	1.07
100	5	0.15	10.66	0.42	0.13	11.05	0.32	0.88	1.05
100	6	0.13	12.64	0.45	0.12	13.38	0.43	0.93	1.07
100	7	0.11	15.39	0.5	0.1	15.66	0.41	0.92	1.03
100	8	0.11	17.43	0.55	0.1	17.38	0.44	0.92	1.01
100	9	0.1	20.39	0.63	0.08	20.74	0.44	0.83	1.03
100	10	0.09	21.9	0.55	0.08	22.53	0.42	0.87	1.04
100	11	0.09	24.63	0.64	0.07	26.35	0.4	0.75	1.08
100	12	0.08	26.45	0.66	0.06	29.49	0.48	0.76	1.12
1000	5	0.14	12.78	0.51	0.12	13.34	0.40	0.87	1.05
1000	6	0.12	15.28	0.54	0.10	16.15	0.42	0.84	1.07
1000	7	0.10	17.80	0.55	0.10	18.53	0.50	0.91	1.05
1000	8	0.09	21.24	0.58	0.08	21.72	0.48	0.90	1.03
1000	9	0.09	23.04	0.59	0.07	24.48	0.49	0.86	1.07
1000	10	0.08	25.28	0.64	0.07	27.90	0.46	0.78	1.11
1000	11	0.07	28.93	0.66	0.06	30.82	0.50	0.85	1.07
1000	12	0.07	32.80	0.68	0.06	33.74	0.54	0.88	1.04

Table 2. Ratios of CHs and border nodes and cluster sizes in *2-LowestID* and *2-ConID*

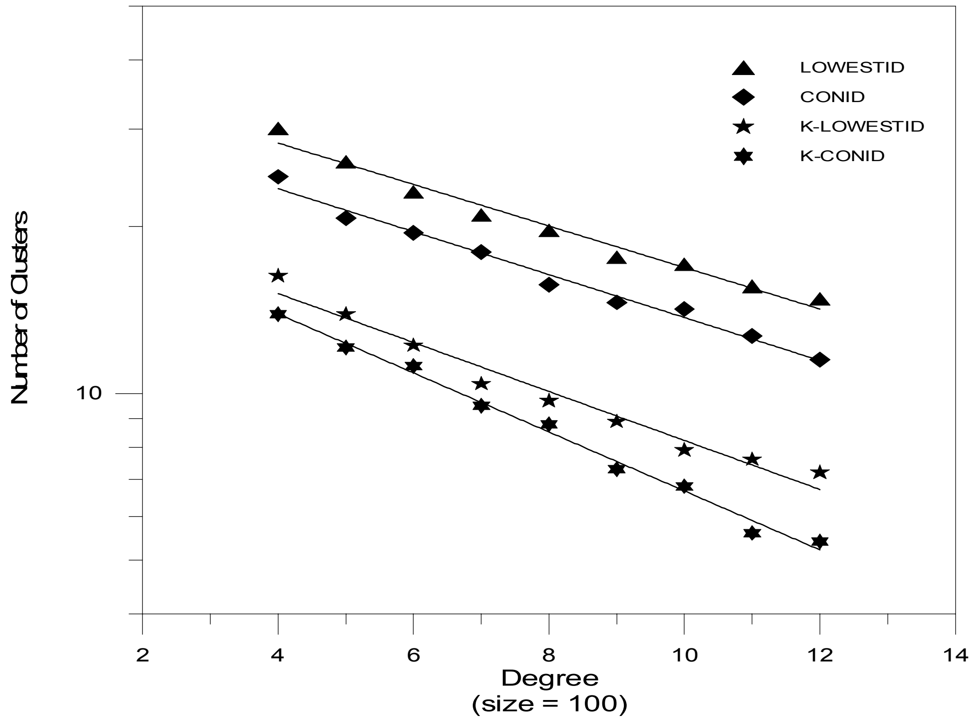

Diagram 1. The average number of clusters in *k-LowestID* and *k-ConID, k*=1 and *k*=2.

6. Unified framework for clustering

We intend to study unified framework for clustering algorithm in wireless networks, where each node has a weight that indicated its suitability for CH role, and weight is decided by a generalized formula that will take a number of components into the account. For example, the weight can be defined as:

*Weight=a*speed+b*degree + c*power+d*energy-left.*

The parameters *a, b ,c, d* depend on particular application. They can be positive or negative. The listed variables need to be expressed in proper units, possibly normalized, or inverse of stated meaning, again depending on application. The *speed* reflects node mobility (it is 0 for static nodes), *degree* indicated connectivity of node, *power* reflects transmission radius a node can use, and *energy-left* measured the amount of energy left at given node. More components can be added. For instance, the weight used in [SDB] includes component *dCH,* the number of already assigned nodes to given cluster (all initiated at 0).

The weight can be applied also for *k*-hop clustering, but value *k>1* appears to be practical only for static nodes, such as in sensor networks.

7. Random unit graphs with obstacles

The experiments on clustering in literature used either unit graph or random graph, where each edge is selected or not selected for the graph based on a randomly generated number and desired density. In order to address the issue of obstacles, and possibility of two nodes that are just beyond transmission radius to still communicate, we propose to consider a model where the existence of each edge depends on transmission radius and randomly generated number, with high probability of edge existence for distance below *R* (and increasing with reduced *R*), and low probability for distances more than *R* (decreasing with increasing *R*). There are variety of formulas that can be used for generating random unit graphs with obstacles. We suggest here one such formula. Let *d* be the distance between two nodes, and let *R* be the transmission radius. Suppose that we want to address issue of obstacles and increased visibility, but we also want to preserve graph density. Generate a random number *x* in interval (0,1), and consider values $x(d/r)^2$assigned to each possible edge. Decide the amount of impact of obstacles and extended visibility, which depends on terrain. Based on that, calculate the

Proceedings of the 35th Annual Hawaii International Conference on System Sciences (HICSS-35'02)
0-7695-1435-9/02 $17.00 © 2002 IEEE

desired number of obstacles p, which is then the same as the number of additional edges due to visibility added. Then delete p largest values for $x(d/r)^2$, for $d \le R$, and add p smallest such values when $d>R$. The graph density is preserved. Note that this idea may be varied. For instance, if we do not want to add any edge beyond transmission radius, then this is applied only when $d \le R$, and the relation between graph density and R is chosen such that there are p more edges than desired number for given density. After deleting p edges, density is back to desired level. Further, the quadratic dependence on distance can be replaced by other degree, to emphasise more or less the importance of distance for connectivity.

8. Clustering based flooding and Bluetooth scatternet formation algorithms

This section completes literature review by discussing in more detail two important applications of cluster formation algorithms, for broadcasting (that is, flooding) in wireless networks, and for Bluetooth scatternet formation.

In a broadcasting task, one node wishes to send the same message to all other nodes in the wireless network. Since nodes in wireless network normally share a common channel, a message sent by any node is received by all nodes located within the transmission radius, assumed to be equal for all nodes. Flooding can be achieved easily if each node receiving the message will retransmit it. However, this will cause unnecessary collisions and bandwidth waste, with many nodes not receiving the message as a consequence. Instead, it is sufficient if only nodes that form a connected dominating set retransmit the message. A dominating set *D(S)* of a set *S* is a set of nodes such that each node from *S* either belongs to *D(S)* or has a neighboring node that belongs to *D(S).* Broadcasting by retransmitting from nodes belonging to a connected dominating set was proposed in [LK, SSZ]. The problem of finding a connected dominating set of minimal size is NP-complete, even if a node has global knowledge about the network [LK]. All clusterheads and border nodes of a cluster structure define a connected dominating set. To minimize the size of the set, [SSZ] proposed to apply node degree as the primary key in clusterhead decisions. The scheme [AWF] does not apply degree as primary key, but instead reduces the size of border nodes set. After clustering process is completed, each clusterhead contacts neighboring clusterheads (up to 3-hops away) in order to eliminate some border nodes, and use only essential border nodes to preserve overall connectivity. Note that [SSZ] described some other broadcasting schemes, based on dominating set definition [WL], while [LK] applied yet another heuristic scheme.

A piconet in Bluetooth network organization consists of a master node and up to seven slaves, and each master-slave communication uses its own frequency hopping pattern. The standard allows multiple roles for the same device. A node can be master in one piconet and a slave in one or more other piconets. The network topology resulting by the connection of piconets is called a scatternet. Nodes use time division to switch between piconets. Since each switch causes delay (e.g. scheduling and synchronization time), an efficient scatternet formation protocol can be one that minimizes the roles assigned to the nodes, without losing network connectivity, and minimizes number of piconets. A clustering based formation algorithm was proposed in [BKNR]. It uses 2-hop communication, but seems to fail to connect piconets in some scenarios, as shown in [LS].

The formation algorithm [LS] applies a variant of clustering scheme [LG], combined with edges belonging to *RNG* (relative neighborhood graphs) used to preserve connectivity of piconets. An edge *UV* belongs to *RNG* iff $|UW|>|UV|$ or $|VW|>|UV|$ for any common neighbor *W* of *U* and *V*. *RNG* is a sparse connected subgraph of connected unit graph (see [SSS] for more details on RNG).

The algorithm [LS] marks all nodes initially as undecided and repeatedly creates piconets until no more undecided nodes remain. Each node *X* maintains *key(X)=(connectivity, id),* where *connectivity* is the number of its undecided neighbors. To create one piconet, choose any undecided node *X* such that *key(X)>key(Y)* for any undecided neighbor *Y* of *X*. *X* becomes master of a piconet. If it has up to 7 undecided neighbors, all of them become its slaves, and also decided nodes. Otherwise divide the area around X into 7 equal angular ranges, and choose one node (if any) from each region. If region contains an *RNG* edge, choose the shortest such edge. Otherwise choose the shortest edge, with the other endpoint becoming slave and decided. Delete all remaining edges from each region. Update all keys for all nodes appropriately. When all nodes become decided, the piconet structure needs to become connected. It is done by creating some new piconets. For each remaining (never deleted) edge *AB* that connects two slaves and is an *RNG* edge create a piconet with one of nodes, say *A,* being master, and the other, *B,* being the only slave. More details are given in [LS].

References

[APVH] A.D. Amis, R. Prakash, T.H.P. Vuong, D.T. Huynh, Max-min d-cluster formation in wireless ad hoc networks, Proc. IEEE INFOCOM, 2000.

[AWF] K.M. Alzoubi, P.J. Wan and O. Frieder, New distributed algorithm for connected dominating set in wireless ad hoc networks, Proc. IEEE Hawaii Int. Conf. System Sciences, 2002.

[BKNR] K. Balaji, S. Kapoor, A.A. Nanavati, L. Ramachandran, Scatternet formation in Bluetooth, 2001.

[B1] S. Basagni, Distributed clustering for ad hoc networks, proc. Int. Symp. Parallel Algorithms, Architectures and Networks ISPAN'99, Perth, Australia, June 1999, 310-315.

[B2] S. Basagni, Distributed and mobility-adaptive clustering for multimedia support in multi-hop wireless networks, Proc. IEEE VTC, Sept. 1999.

[B3] S. Basagni, A distributed algorithm for finding a maximal weighted independent set in wireless networks, Telecommunication Systems, to appear.

[B4] S. Basagni, A distributed algorithm for finding a maximal weighted independent set in wireless networks, Telecommunication Systems, to appear.

[BK] S. Banerjee and S. Khuller, A clustering scheme for hierarchical control in multi-hop wireless networks, Proc. IEEE INFOCOM, 2001.

[EFB] A. Ephremides, J.A. Flynn and D.J. Baker, The design and simulation of a mobile radio network with distributed control, IEEE J. of Selected Areas in Communication, 2, 1, 1984, 226-237.

[EWB] A. Ephremides, J.E. Wieselthier and D.J. Baker, A design concept for reliable mobile radio networks with frequency hoping signaling, Proc. IEEE 75, 1987, 56-73.

[GS] Geng Chen and Ivan Stojmenovic, Clustering and routing in wireless ad hoc networks, SITE, University of Ottawa, TR-99-05, June 1999.

[GT] M. Gerla and J.T.C. Tsai, Multicluster, mobile, multimedia radio network, Wireless Networks, 1, 1995, 255-265.

[HT] T.C. Hou and T.J. Tsai, An access-based clustering protocol for multihop wireless ad hoc networks, IEEE J. Selected Areas in Comm., 19, 7, 2001, 1201-1210.

[K] F. Kamoun, Design considerations for large computer communication networks, UCLA-ENG-7642, Comp. Sc. Dept, UCLA, 1976.

[KHC] D. Kim, S. Ha and Y. Choi, K-hop cluster-based dynamic source routing in wireless ad-hoc packet radio networks, IEEE VTC, 1998, 224-228.

[KK] I. Kleinrock, F. Kamoun, Hierarchical routing for large networks, Comp. Netw., 1, 1977, 155-174.

[KK1] F. Kamoun, L. Kleinrock, Stochastic performance evaluation of hierachical routing for large networks, Comp. Netw., 3, 1979, 337-353.

[KVCP] P. Krishna, N.N. Vaidya, M. Chatterjee, D.K. Pradhan, A cluster-based approach for routing in dynamic networks, ACM SIGCOMM Computer Communication Review, 49, 1997, 49-64.

[L1] G. Lauer, Packet-radio routing, in: Routing in Communication Networks, ed. M. Steenstrup, Prentice-hall, 1995.

[L2] G. Lauer, Hierarchical routing design for SURAN, Proc. ICC, 1986, 93-101.

[L3] G. Lauer, Address servers in hierachical networks, Proc. ICC, 1988, 443-451.

[LG] C.R. Lin and M. Gerla, Adaptive clustering for mobile wireless networks, IEEE J. on Selected Areas in Comm., 15, 7, 1997, 1265-1275.

[LK] H. Lim and C. Kim, Flooding in wireless ad hoc networks, Computer Communication J., 24, 3-4, 353-363, Feb. 2001.

[LS] X. Li and I. Stojmenovic, Clustering and RNG based Bluetooth scatternet formation, 2001.

[P] A.K. Parekh, Selecting routers in ad hoc wireless networks, ITS, 1994.

[RBS] C.V. Ramamoorthy, A. Bhide, J. Srivastava, Reliable clustering techniques for large mobile packet radio networks, Proc. IEEE INFOCOM, 218-226, 1987.

[S] N. Shacham, Organization of dynamic radio networks by overlapping clusters, Performance, December, 1984.

[SDB] R. Sivakumar, B. Das and V. Bharghavan, Spine routing in ad hoc network, ACM/Baltzer Cluster Computing Journal, 1998.

[SSS] M. Seddigh, J. Solano and I. Stojmenovic, RNG and internal node based broadcasting algorithms for wireless one-to-one networks, ACM Mobile Computing and Communications Review, Vol. 5, No. 2, 2001, www.site.uottawa.ca/~ivan .

[SSZ] Stojmenovic I., M. Seddigh, J. Zunic, Internal node based broadcasting in wireless networks, CD-ROM Proc. of the 34th Annual IEEE Hawaii Int. Conf. on System Sciences, January 2001.

[SW] N. Shacham and J. Westcott, Future directions in packet radio architectures and protocols, Proc. IEEE, 75, 1, 1987, 83-98.

[T] C.K. Toh, Associativity-based routing for ad-hoc mobile networks, Wireless Personal Communications 4, 1997, 103-139.

[TRTN] W.T. Tsai, C.V. Ramamoorthy, W.K. Tsai, O. Nishiguchi, An adaptive hierarchical routing protocol, IEEE Tr. Comm. 38, 1989, 1059-1075.

[WL] J. Wu and H. Li, On calculating connected dominating set for efficient routing in ad hoc wireless networks, Proc. DIAL M, Seattle, Aug. 1999, 7-14.

This research is partially supported by NSERC and REDII-CONACyT research grants.