INTERNATIONAL JOURNAL OF ENGINEERING AND MANAGEMENT SCIENCES

www.scienceandnature.org

IMPROVED WEIGHTED CLUSTERING ALGORITHM FOR MOBILE AD HOC NETWORKS

Sudhakar Pandey, Narendra Kumar Shukla
J K Institute of Applied Physics and Technology, University of Allahabad, Allahabad

ABSTRACT
The proposed weight-based distributed clustering algorithm takes into consideration the ideal degree, transmission power, mobility, and battery power of mobile nodes. 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 cluster head election is invoked on-demand, and is aimed to reduce the computation and communication costs. The cluster heads, operating in "dual" power mode, connects the clusters which help in routing messages from a node to any other node.

Key words: ad hoc networks, clusters, dominant set, load balancing

INTRODUCTION

Current wireless cellular networks solely rely on the wired backbone by which all base stations are connected, implying that networks are fixed and constrained to a geographical area with a pre-defined boundary. Deployment of such networks takes time and cannot be set up in times of utmost emergency. Therefore, 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 eco- nomical to build, such as law enforcement operations, battle field communications, disaster recovery situations, and so on. Such situations demand a network where all the nodes includ- ing the base stations are potentially mobile, and communica- tion must be supported untethered between any two nodes.

A multi-cluster, multi-hop packet radio network archi- tecture for wireless systems should be able to dynamically adapt itself with the changing network configurations. Cer- tain nodes, known as *clusterheads*, are responsible for the formation of *clusters* each consisting of a number of nodes (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 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 clus- terheads optimally is an NP-hard problem [4]. Hence exist- ing solutions to this problem are based on heuristic (mostly greedy) approaches and none attempts to retain the stability of the network topology [4,5]. We believe a good cluster- ing scheme should preserve its structure as much as possible when nodes are moving and/or the topology is slowly chang- ing. Otherwise, re-computation of clusterheads and frequent information exchange among the participating nodes will re- sult in high computation overhead.

In this paper, we propose a weight based distributed clus- tering algorithm which takes into consideration the number of nodes a clusterhead can handle ideally (without any se- vere degradation in the performance), transmission power, mobility, and battery power of the nodes. Unlike other ex- isting schemes which are invoked periodically resulting in high communication overhead, our algorithm is *adaptively* in- voked based on the mobility of the nodes. More precisely, the clusterhead election procedure is delayed as long as possi- ble to reduce the computation cost. Balancing the loads be- tween clusterheads is another desirable feature of any clus- tering algorithm, however, it is very difficult to maintain a completely balanced system due to the dynamic nature of the nodes. Our algorithm achieves *load balancing* by specifying a pre-defined threshold on the number of nodes that a cluster- head can handle ideally. This ensures that none of the clus- terheads are overloaded at any instance of time. We define *load balancing factor* (LBF) to measure the degree of load balancing among the clusterheads. Connecting the nodes is another important issue since the nodes need to communi- cate with each other.

Preliminaries

The network formed by the nodes and the links can be repre- sented by an undirected graph $G = (V, E)$, where V repre- sents the set of nodes v_i and E represents the set of links e_i. Note that the cardinality of V remains the same but the cardi- nality of E always changes with the creation and deletion of links. More formally, we look for the set of vertices S V (G),such that

$$\bigcup_{v \in S} N[v] = V(G).$$

Here, $N[v]$ is the *neighborhood* of node v, defined as

$$N[v] = \bigcup_{v' \in V,\ v' \neq v} \{v' | \text{dist}(v, v') < tx_{\text{range}}\},$$

where tx_{range} is the transmission range of v. The neighbor- hood of a clusterhead is the set of nodes which lie within its transmission range. The set S is called a *dominating set* such that every vertex of G belongs to S or has a neighbor in S.

Design Philosophy

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 a new algorithm which is based on 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. Depending on specific applications, any or all of these parameters can be used in the metric to elect the clus- terheads. We could have a fully distributed system where all the nodes in the mobile network share the same responsibil- ity and act as clusterheads. However, more clusterheads re- sult 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 larger number of clusterheads. Thus this solution leads to higher latency, more power consumption and more informa- tion processing per node.

On the other hand, to maximize the resource utilization, we can choose to have the minimum number of clusterheads to cover the whole geographical area over which the nodes are distributed. The whole area can be split up 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 en- tire 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 elected. It might so happen that none of the other nodes in that zone are centrally located.

BASIS FOR ALGORITHM

To decide how well suited a node is for being a clusterhead, we take into account its degree, transmission power, mobility and battery power. The following features are considered in our clustering algorithm:

- The clusterhead election procedure is not *periodic* and is invoked as rarely as possible. This reduces system updates and hence computation and communication costs. The clustering algorithm is not invoked if the relative distances between the nodes and their clusterheads do not change.
- Each clusterhead can ideally support only δ (a pre- defined threshold) nodes to ensure efficient medium access control (MAC) functioning. If the clusterhead tries to serve more nodes than it is capable of, the system efficiency suffers in the sense that the nodes will incur more delay because they have to wait longer for their turn (as in TDMA) to get their share of the resource. A high system throughput can be achieved by limiting or optimizing the *degree* of each clusterhead.
- The *battery power* can be efficiently used within certain transmission range, i.e., it will take less power for a node to communicate with other nodes if they are within close distance to each other. A clusterhead consumes more bat- tery power than an ordinary node since a clusterhead has extra responsibilities to carry out for its members.
- *Mobility* is an important factor in deciding the cluster- heads. In order to avoid frequent clusterhead changes, it is desirable to elect a clusterhead that does not move very quickly. When the clusterhead moves fast, the nodes may be detached from the clusterhead and as a result, a *reaf- filiation* occurs. Reaffiliation takes place 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 cluster- head 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 with its neigh- bors having closer *distances* from it within the transmis- sion range [12]. As the nodes move away from the cluster-head, the communication may become difficult due mainly to signal attenuation with increasing distance

CLUSTER HEAD ELECTION PROCEDURE

The procedure consists of eight steps as described below:

Step 1. Find the neighbors of each node v (i.e., nodes within its transmission range) which defines its *degree*, d_v, as

$$d_v = |N(v)| = \sum_{v' \in V,\ v' \neq v} \{\text{dist}(v, v') < tx_{\text{range}}\}.$$

Step 2. Compute the *degree-difference*, for every node v.

Step 3. For every node, compute the *sum of the distances*, D_v, with all its neighbors, as

$$\Delta_v = |d_v - \delta|,$$
$$D_v = \sum_{v' \in N(v)} \{\text{dist}(v, v')\}$$

Step 4. Compute the running average of the speed for every node till current time T. This gives a measure of mobility and is denoted by M_v, as

Step 5. Compute the cumulative time, P_v, during which a node v acts as a clusterhead. P_v implies how much bat- tery power has been consumed which is assumed more for a clusterhead than an ordinary node.

$$M_v = \frac{1}{T} \sum_{t=1}^{T} \sqrt{(X_t - X_{t-1})^2 + (Y_t - Y_{t-1})^2},$$

Step 6. Calculate the *combined weight* W_v for each node v, where

$$W_v = w_1\Delta_v + w_2 D_v + w_3 M_v + w_4 P_v,$$

where w_1 , w_2 , w_3 and w_4 are the *weighing factors* for the corresponding system parameters.

Step 7. Choose that node with the smallest W_v as the cluster- head. All the neighbors of the chosen clusterhead are no longer allowed to participate in the election procedure.

Step 8. Repeat steps 2–7 for the remaining nodes not yet se- lected as a clusterhead or assigned to a cluster.

AN ILLUSTRATIVE EXAMPLE

We demonstrate our weighted clustering algorithm with the help of figures 1–6. All numeric values, as obtained from executing WCA on the 15 nodes as shown in figure 1, are tabulated in table 1. Figure 1 shows the initial configuration of the nodes in the network with individual node ids. Dotted circles with equal radius represent the fixed transmission range for each node. A node can hear broadcast beacons from the nodes which are within its transmission range. An edge between two nodes in figure 2 signifies that the nodes are neighbors of each other. The degree, d_v , which is the total number of neighbors a node has is shown in step 1. The degree difference, A_v ,of each node with ideal node degree S = 2 is computed in step 2. Sum of the distances, Dv , for each node is calculated as step 3, where the unit distance has been chosen arbitrarily.The arrows in figure 3 represent the speed and direction of movement associated with every node. A longer arrow rep- resents faster movement and a shorter arrow indicates slower movement. The values for M_v (step 4), are chosen randomly.M_v = 0 implies that a node does not move at all. We choose some arbitrary values for P_v which represent the amount of time a node has acted as a clusterhead. This corresponds to step 5 in our algorithm. After the values of all the components are identified, we compute the weighted metric, W_v , for every node as proposed in step 6 in our algorithm. The weights con- sidered are w_1 = 0.7, w_2 = 0.2, w_3 = 0.05 and w_4 = 0.05. Note that these weighing factors are chosen arbitrarily such that w_1 + w_2 + w_3 + w_4 = 1. The contribution of the indi- vidual components can be tuned by choosing the appropriate combination of the weighing factors. Figure 4 shows how a node with minimum W_v is selected as the clusterhead in a dis- tributed fashion as stated in step 7 in our algorithm. The solid nodes represent the clusterheads elected for the network. Note that no two clusterheads are immediate neighbors. Figure 5 shows the initial clusters formed by execution of the cluster- ing algorithm. We observe that the total number of neighbors served by each clusterhead is close to the predefined ideal de- gree, S = 2. Figure 6 shows the achieved connectivity in the network. As discussed earlier, the connectivity is accom-plished through the higher power (as a result of dual mode power) transmission range of a clusterhead. It can be noted that a single component graph is obtained in this case which means that there is a path from a node to any other node.

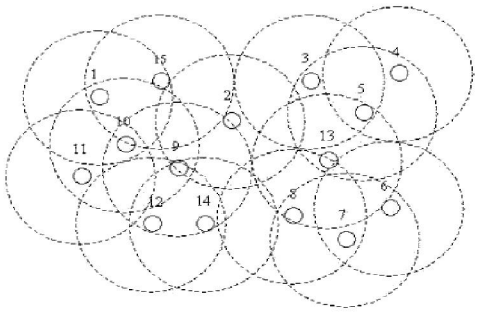

Figure 1. Initial configuration of nodes.

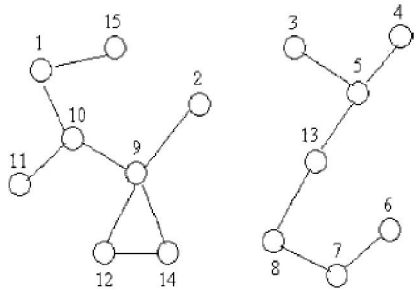

Figure 2. Neighbors identified.

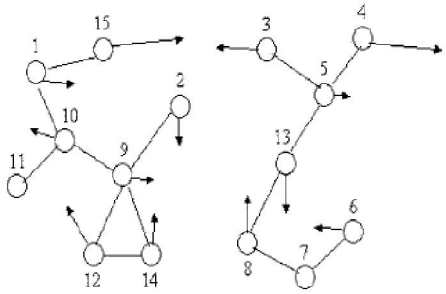

Figure 3. Velocity of the nodes.

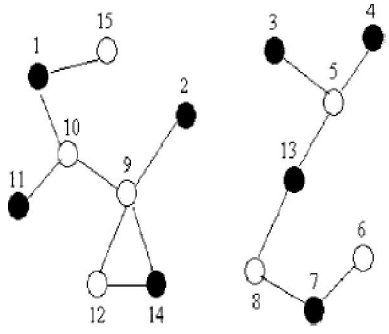

Figure 4. Clusterheads identified.

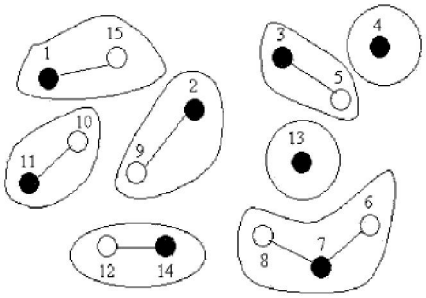

Figure 5. Clusters identified.

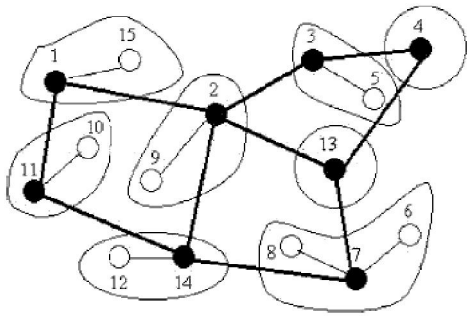

Figure 6. Connectivity achieved.

COMPLEXITY DUE TO DISTRIBUTIVENESS

The time required for the selection of the node with minimum W_v depends on the implementation of the algorithm. In a centralized system with a central server, the minimum W_v can be found in linear time with respect to the number of nodes. But it is not possible to have a centralized server in ad hoc networks. So, we proceed with a distributed solution in which all the nodes broadcast their ids along with W_v val- ues. A node receives broadcasts from its neighbors and stores the information. This stored information is again exchanged with the immediate neighbors and the process continues till all the nodes become aware of the node with the smallest W_v. The time required for the nodes to gather information about all other nodes will depend on the *diameter* of the underlying graph. It is to be noted that this procedure yields the *global* minima of W_v s unlike the Lowest-ID algorithm which finds only the local minima of ids.

It can be argued that the existing heuristics discussed in section 2 are all special cases of our algorithm. The Highest- Degree heuristic considers only the degree of a node and dis- regards all other system parameters ($w_2 = w_3 = w_4 = 0$). In Lowest-ID heuristic, the assignment of the ids are random. We can assume that the ids being assigned are based on mobility. The lowest id is assigned to the least mobile node and highest id for the most mobile. In that case, ($w_1 = w_2 = w_4 = 0$). The Node-Weight heuristic simply assigns weights to the nodes which are equivalent to W_v in our case. The basis for suitability of nodes being clusterheads is ignored there. However, in our approach, we define and formulate the parameters for choosing a clusterhead and we show how the weight W_v are calculated.

SYSTEM ACTIVATION AND UPDATE POLICY

When a system is initially brought up, every node *v* broad- casts its id which is registered by all other nodes lying within *v*'s transmission range, *tx_range*, as can be seen from figure 1. It is assumed that a node receiving a broadcast from another node can estimate their mutual distance from the strength of the signal received. GPS (Global Positioning System) can be another solution since it is mainly used to obtain the ge- ographical location of nodes. Even though GPS might make the problem relatively simpler, but there is always a cost associated with the deployment of GPS since every mobile node must be a GPS receiver. Based on the received signal strength, every node is made aware of its neighboring nodes and their corresponding distances. Note that these neighboring nodes are only the geographical neighbors and do not necessarily mean neighbors within the same cluster. Once the neighbors list for each node is ready, our clustering algorithm chooses the clusterhead for the first time, as illustrated in figure 4. It can be noted that the mobility factor and the battery power would be the same for all the nodes when the system is ini- tialized. Effectively, W_v will have only two terms A_v and D_v contributing to it. Each node maintains its status (i.e., clus- terhead or not). A non-clusterhead node knows the cluster it belongs to and the corresponding clusterhead.

Due to the dynamic nature of the system considered, the nodes as well as the clusterheads tend to move in different directions, thus disorganizing the stability of the configured system. So, the system has to be updated from time to time. The update may result in formation of new clusters and pos- sible change of point of attachment of nodes from one clus- terhead to another within the existing dominant set. This is called *reaffiliation*. The frequency of update and hence reaf- filiation is an important issue. If the system is updated peri- odically 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 compu- tational cost resulting in the loss of battery power or energy. If the frequency of update is low, there are chances that cur- rent topological information will be lost resulting in sessions terminated midway.

Instead of continuously monitoring the clusterhead the nodes should monitor it dynamically based on the received signal strength I.e if the siganl strength is strong then monitor it after a long time and if weak then notify the clusterhead and the clusterhead tries to hand-over the node to a neighboring cluster (existing clusterhead in the domi- nant set). The clusterhead of the reaffiliated node updates its member list. If the node goes into a region not covered by any clusterhead, then the clusterhead election algorithm is invoked and the new dominant set is obtained.

The objective of our cluster head election algorithm is to minimize the number of changes in dominant set update.

Once the neighbors list for all nodes are created, the degree- difference A_v is calculated for each node v. Also, D_v is com- puted for each node by summing up the distances of its neigh- bors. The mobility M_v is calculated by averaging the speed of the node. The total amount of time, T_v, it remained as a clusterhead is also calculated. All these parameters are nor- malized, which means that their values are made to lie in a pre-defined region. The corresponding weights w_1, w_2, w_3 or w_4 are kept fixed for a given system. The weighing fac- tors also give the flexibility of adjusting the effective contri- bution of each of the parameters in calculating the *combined weight* W_v. For example, in a system where battery power is more important, the weight w_4 associated with T_v can be made larger. Note that the sum of these weighing factors is 1. The node with the minimum total weight, W_v, is elected as a clusterhead. The elected 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 some clusterhead.

I. LOAD-BALANCING

The load handled by a clusterhead depends on the number of nodes supported by it. A clusterhead, apart from supporting its members with the radio resources, has also to route mes- sages for other nodes belonging to different clusters. There- fore, it is not desirable to have any clusterhead overly loaded while some others are lightly loaded [1]. At the same time, it is difficult to maintain a perfectly load balanced system at all times due to frequent detachment and attachment of the nodes from and to the clusterheads. To quantitatively measure how well balanced the clusterheads are, we introduce a parameter called *load balancing factor* (LBF). As the load of a cluster- head can be represented by the cardinality of its cluster size, the variance of the cardinalities will signify the load distribu- tion. We define the LBF as the inverse of the variance of the cardinality of the clusters. Thus,

$$\text{LBF} = \frac{n_c}{\sum_i (x_i - \mu)^2},$$

where n_C is the number of clusterheads, x_i is the cardinality of cluster i, and $\mu = (N - n_C)/n_C$, (N being the total number of nodes in the system) is the average number of neighbors

of a clusterhead. Clearly, a higher value of LBF signifies a better load distribution and it tends to infinity for a perfectly balanced system.

II. CONNECTING THE CLUSTERS

As a logical extension to clustering, we investigate the con- nectivity of the nodes which is essential for any routing al- gorithm. Clustering ensures that the nodes within a cluster are able to communicate among themselves through the clus- terheads, each of which acts as the central node of a *star*, as shown in figure 5. But, inter-cluster communication is not possible if the clusters are not connected. For two clusters to communicate with each other, we assume that the cluster- heads are capable of operating in *dual* power mode. A cluster- head uses low power to communicate with the members in its transmission range, and high power to communicate with the neighboring clusterheads because of greater range. The links between the clusterheads are shown as solid lines in figure 6 We define *connectivity* as the probability that a node is reachable from any other node. For a single component graph, any node is reachable from any other node and the connectiv- ity is 1. If the network does not result in a single component graph, then we can say that all the nodes in the largest com- ponent can communicate with each other and the connectivity can be the ratio of the cardinality of the largest component to the cardinality of the graph. Thus,

$$\text{connectivity} = \frac{\text{size of largest component}}{N}.$$

The transmission range of a clusterhead can be made large enough by adjusting the power in such a way so as to yield a connected network.

CONCLUSION

We proposed a weight based distributed clustering algorithm (WCA) which can dynamically adapt itself with the ever changing topology of ad hoc networks. Our approach restricts the number of nodes to be catered by a clusterhead so that it does not degrade the MAC functioning. It has also the flex- ibility of assigning different weights and takes into 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 cluster- heads. Our clustering algorithm tries to distribute the load as much as possible. We observe that there is a pattern of how the LBF (load balance factor) changes to distribute the load. There is a gradual increase in the LBF due to the diffusion of the nodes among the clusters. The sharp decrease is due to the imbalance caused by the clustering algorithm to ensure that the nodes are connected, which helps in routing messages from any node to any other node. Hence, there is trade-off between the load handled by the clusterheads and the connec- tivity of the network. We conducted simulation experiments to measure the performance of our clustering algorithm and demonstrate that it performs significantly better than both of the Highest-Degree and the Lowest-ID heuristics. In particu- lar, the number of reaffiliations for WCA is about 50% of that obtained from the Lowest-ID heuristic. Though our approach performs marginally better than the Node-Weight heuristic, it considers more realistic system parameters and has the flexi- bility of adjusting the weighing factors.with little improvement in algorithm done we can hope to increase in battery performance.

REFERENCES

[1] Chatterjee M., Das S.K., Turgut D., "WCA: A Weighted Clustering

[2] Algorithm for Mobile Ad Hoc Networks," Cluster Computing Journal, vol. 5, no. 2, Apr. 2002, pp. 193-204.
[3] Abdul Rahman H. Hussein,Amer O. Abu Salem,Sufian Yousef,"A Flexible Weighted Clustering Algorithm Based on
[4] Battery Power for Mobile Ad Hoc Networks", (2008)
[5] Likun Zou, Qishan Zhang, Jianwei Liu , An Improved Weight-Based Clustering Algorithm in
[6] MANETs (2009).
[7] S. Basagni, I. Chlamtac and A. Farago, A generalized clustering algorithm for peer-to-peer networks, July.(1997).
[8] I. Chlamtac and A. Farago, A new approach to the design and analysis of peer-to-peer mobile networks, Wireless Networks (1999).