Available Online at www.ijcsmc.com

International Journal of Computer Science and Mobile Computing

A Monthly Journal of Computer Science and Information Technology

ISSN 2320–088X

IJCSMC, Vol. 3, Issue. 6, June 2014, pg.133 – 143

RESEARCH ARTICLE

A Modified Version of DMAC Clustering Algorithm in MANET

[1]Ira Nath, [2]Trisha Gorai
JIS College of Engineering
Kalyani, Nadia
Ira.nath@gmail.com

Abstract- A Mobile Ad-hoc network (MANET) is formed when a group of mobile wireless nodes collaborate between them to communicate through wireless links in the absence of the fixed infrastructure and any centralized control. These characteristics make it able to adapt and operate in difficult conditions. It is vital to keep the topology stable as long as possible. In this paper a new approach is adopted for cluster formation by modifying DMAC clustering algorithm. This algorithm can result in a more stable configuration, and thus yield better performance.

Keywords: MANET, Cluster, DMAC, Cluster head, Wireless

1. Introduction

A wireless ad hoc network is a decentralized wireless network. The network is ad hoc because each node is willing to forward data for other nodes, and so the determination of which nodes forward data is made dynamically based on the network connectivity. This is in contrast to wired networks in which routers perform the task of routing. It is also in contrast to managed (infrastructure) wireless networks, in which a special node known as an access point manages communication among other nodes. Minimal configuration and quick deployment make ad hoc networks suitable for emergency situations like natural disasters or military conflicts. The presence of a dynamic and adaptive routing protocol will enable ad hoc networks to be formed quickly. The decentralized nature of wireless ad hoc networks makes them suitable for a variety of applications where central nodes can't be relied on. Minimal configuration and quick deployment make ad hoc networks suitable for emergency situations like natural disasters or military conflicts. The presence of a dynamic and adaptive routing protocol will enable ad hoc networks to be formed quickly. A mobile ad hoc network (MANET), sometimes called a mobile mesh network, is a self-configuring network of mobile devices connected by wireless links. A MANET is an autonomous collection of mobile users those communicate over relatively bandwidth constrained wireless links.Each device in a MANET is free to move independently in any direction. Since the nodes are mobile, the network topology may change rapidly and unpredictably over time. The network is decentralized, where all network activity including

discovering the topology and delivering messages must be executed by the nodes itself, i.e., routing functionality will be incorporated into all mobile nodes.The primary challenge in building a MANET is equipping each device to continuously maintain the information required to properly route traffic. MANETs need efficient distributed algorithms to determine network organization, link scheduling, and routing. Significant examples include establishing survivable, efficient, dynamic communication for emergency/rescue operations, disaster relief efforts, and military networks.

In this paper, section 2 describes cluster formation and various popular clustering algorithms in MANET. Section 3 presents our modified algorithms. A case study is depicted in section 4. Conclusion is in section 5.

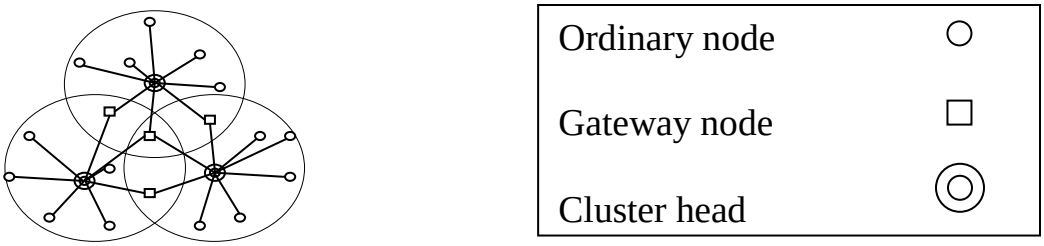

Fig1: Clustering in MANET

2. Cluster Formation

A mobile ad hoc network (MANET) consists of a number of mobile wireless nodes which do not rely on any wired infrastructure. Unlike infrastructure-based networks such as cellular networks, all the components of ad hoc networks are highly mobile. Due to this mobility, the topology of the network changes dynamically. In MANET, nodes rely solely on each other to establish communication links and act as routers to convey data packets. Since the data packet may need to travel from a source to a destination node through a set of intermediate nodes, ad hoc networks are also called "multihop networks". The dynamic nature of ad hoc networks poses challenges to routing protocols. Topology changes cause link breakages between mobile nodes, increasing packet loss ratio. Limited resources such as bandwidth diminish the quality of service in the application layer. Ad hoc networks need to be designed to be able to dynamically adapt to the changing network configurations. One of the ways to handle the topology changes and maintain a connected network is to entrust certain nodes with more responsibility. These nodes are called cluster heads and are responsible for the formation of clusters, each consisting of a number of ordinary nodes and possible gateway nodes. A cluster head is responsible for resource allocation to all the nodes belonging to its cluster. If a node wishes to transmit data outside the cluster, it can either go through the cluster head, or go through a gateway designated by the cluster head. A clustered network is usually more power efficient. Clustering can help with topology management, routing, and so on. 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 cluster heads is unavoidable. But a good clustering scheme should preserve its structure as much as possible when nodes are moving and/or the topology is slowly changing. Otherwise, re-computation of cluster heads and frequent information exchange among the participating nodes will result in high computation overhead. Naturally, there are many ways in which a network can be partitioned into clusters and cluster head chosen. This is reflected in the large number of clustering protocols developed by the ad hoc network community.

2.1Clustering Algorithms

2.1.1Lowest-ID algorithm

The Lowest-ID algorithm also known as identifier based clustering, provides three different roles for the nodes: original, gateway, and cluster head. The algorithm of electing a cluster head is as follows:

Each node is assigned a unique node ID during the setup process. The node assigned with lowest ID in its group becomes the cluster head. If a node lies in at least two different clusters, it becomes a gateway in which it acts as a bridge or a connection between the clusters it is part of. Cluster heads cannot be direct neighbors. Inter-cluster messages always have to go through a gateway node. To identify a gateway node candidate, the node v checks if any of its neighbor z is a cluster head. If v has multiple neighbors which are cluster heads, it becomes a gateway node. The creating Clusters function is executed until the entire network is partitioned. The node v broadcasts its ID to find its neighbors. If there are no neighbors, v becomes a cluster head itself. The finding neighbor algorithm is used to find the neighbors of v. Then v broadcasts its node –ID to all its neighbors. If a node z is in transmission range of v it is added to the neighbor list of v.

2.1.2Highest-degree algorithm

The Highest-Degree algorithm also known as connectivity-based clustering is one of the well known heuristics for cluster formation. The node of the network is classified as cluster head and ordinary nodes. The functionality of a cluster head node is to control the local traffic of the nodes in the cluster. The algorithm of electing a cluster head is as follows:

Each node is assigned a unique ID in the network. The node broadcasts its node ID to other nodes within its transmission range. Any node receiving the signal is included in the neighbors list of that node. The node with the most number of neighbors is elected as the cluster head. If there is a tie, the node with the lower node ID is chosen. The cluster heads cannot be direct neighbors. The steps are repeated until the remaining nodes in the network either become a cluster head or join a cluster. The creating clusters function identifies the cluster heads in the network. To be considered in the cluster head election process, the node neither can be marked as member of a cluster, nor can it be in a transmission range of a cluster head. The algorithm is executed until all the nodes are marked and the entire network is partitioned. The node v then broadcasts its node ID to its neighbors. If the node z is in the transmission range, it is added to the neighbor list of v.

2.1.3Weighted clustering algorithm (WCA)

Weighted clustering algorithm elects cluster heads based on the degree-difference, the sum of the distances, the mobility, and the battery power .For electing the cluster head the following procedure is computed first. Find the neighbor of each node v (i.e., nodes within its transmission range) which defines its degree is denoted by dv. Compute the degree-difference, $\Delta v = |dv - \delta|$, for every node v. For every node, compute the sum of the distances, Dv, with all its neighbors. Compute the running average of the speed for every node till current time T. This gives a measure of mobility and is denoted by Mv. Compute the cumulative time, Pv, during which a node v acts as a cluster head. Pv implies how much battery power has been consumed which is assumed more for a cluster head than an ordinary node. Calculate the combined weight Wv for each node v, where $Wv = w1\Delta v + w2Dv + w3Mv + w4Pv$, Where w1, w2, w3 and w4 are the weighing factors for the corresponding system parameters. Choose that node with the smallest Wv as the cluster head. All the neighbors of the chosen cluster head are no longer allowed to participate in the election procedure. The above procedure is repeated for the remaining nodes not yet selected as a cluster head or assigned to a cluster.

2.1.4Distributed Mobility-Adaptive Clustering (DMAC) algorithm

It is enhanced version of distributed clustering algorithm (DCA) which works best in quasi static network. DMAC is on the other hand works good in dynamically changing environment. In both algorithm weight is assigned to the nodes. There are at least three conditions that must hold for these two algorithms:

(a)Each partition must have a cluster head of its own.
(b)Each ordinary node will affiliate with one neighboring cluster head. The weight of the cluster head is larger than the weight of the nodes in the cluster.
(c)Cluster heads cannot be direct neighbors.

In the DCA algorithm the node with highest weight sends a CH message to all its neighbors. Other nodes in the cluster wait to receive a message. When it receives a CH message it checks if it already belongs to a cluster. If not it sends a JOIN message. In this way whole network is partitioned. The DCA algorithm only handles the setup phase of the clustering process. Its extension, DMAC algorithm, can handle both the setup and maintenance phases. As the network has a dynamic nature, the algorithm needs an ability to check whether the nodes are attached to or detached from a cluster at any time. DCA and DMAC are passive algorithms, which mean that after an active initialization process, they become message driven. Thus, the nodes in the network react to the messages received from the neighboring nodes and remain in sleep mode when no events occur. When a network is in the setup phase, new node joining to the network, the init procedure is executed to divide the network. The node v checks if any neighbor has larger weight than its own. If so, v joins that node's cluster and no longer participates in the election procedure. If not so v becomes cluster head. When the node v receives a JOIN message from node u, it checks whether its cluster contains u, If so, v will remove it from its cluster. On the other hand, if u was not in the cluster, v will add u to the cluster. If node v receives the CH message from node u it checks if the weight of u is larger than of its current cluster head. If this is the case, v tells its neighbors z that it is leaving the current cluster and joining u cluster. If there is a link failure, the node v checks if it is currently a cluster head and whether the failure link is within its cluster. If so, v removes it from the cluster. If v is not the cluster head, it checks whether the failed link was its cluster head. In this case, v is not part of any cluster, so it will execute the init function to attach itself to a cluster.

2.2Limitations

2.2.1Highest-Degree Algorithm

The system has a low rate of cluster head change but the throughput is low under the Highest-Degree heuristic. Typically, each cluster is assigned some resources which are shared among the members. As the number of nodes in a cluster is increased, the throughput drops and hence degradation occurs in the system.
As a cluster head can loose neighbor it may not be elected cluster head next time. So reaffiliation of nodes will occur as a result. All these drawbacks occur because this approach does not have any restriction on the upper bound on the number of nodes in a cluster. To be precise, the Highest-ID algorithm states that the node with the largest number neighbors should be elected as a cluster head. However, a cluster head may not be able handle a large number of nodes due to resource limitations.

2.2.2Lowest-ID Algorithm

The system performance is better compared with the Highest-ID in terms of throughput. However, the drawback is that since the nodes ids do not change with time, those with smaller ids are more likely to become cluster heads than nodes with larger ids. Thus, certain nodes are prone to power drainage due to serving as cluster heads for longer periods of time. If we fix this problem by changing node ids from time to time, the neighboring list of all the nodes need also to be changed.

2.2.3WCA Algorithm

The limitation of the algorithm lies in the procedure of finding the minimum weight node for electing the cluster head .A large number of information are stored and exchanged among the nodes to find the smallest weight. This becomes worse with the increase in network size. Further, the time is quiet high for the cluster setup. This is due to the need for computing so much of information for every node to calculate the combined weight. Whenever a reelection takes place the combined weight of every node needs to be calculated, resulting in further increase of the computation cost.

The major drawback of this algorithm lies with not retaining the property of the lowest weight value to be the cluster head. This happens as WCA does not re-cluster when a member node changes its attaching cluster head. That is, there may be a situation when a low weight node may enter into a cluster whose head is of higher value than this newly entered node.

2.2.4Limitations of DMAC Algorithm

In this algorithm the weights are assigned randomly by the user. The node with largest weight becomes the cluster head. Here the cluster head is not chosen on the basis of parameters like mobility, battery power etc. This algorithm does not measure the amount of time that a node may need to wait to receive responses from its neighbors. So it takes a large amount of time for setup of cluster. When a member node comes within the transmission proximity of another head node whose weight is more than its current head, then it joins the new head resulting in further re-affiliation. The DMAC algorithm assigns weights to each node of the cluster and the election of the cluster head is done on the basis of the largest weight among its neighbors. The major weakness of this algorithm lies with the lower weighted nodes. The nodes with the smaller weights wait for a CH message to decide which cluster head to join. Then, the node joins the cluster head, by sending a JOIN message.

3. Modified Work

3.1 Modification of DMAC Algorithm

We hereby propose some modifications on DMAC algorithm which will avoid the reconfiguration of the cluster on the basis of some criteria. For every new node with higher weight coming in the transmission range of a cluster head with less weight, the cluster head will not be reelected every time. Cluster head will be reelected if certain conditions are satisfied. If the weight difference of the new node and the cluster head is greater than some threshold amount then the new node will be cluster head. Otherwise some additional conditions will be checked like remaining battery power of the new node and the cluster head. If the remaining battery power difference is greater than some value then the new node will be cluster head otherwise no changes will occur.

3.2Advantage of modified DMAC Algorithm over DMAC algorithm

In modified DMAC algorithm init() is same as DMAC .so initial phase or setup phase is same as DMAC algorithm. But whenever a new node comes into transmission range of a cluster head two conditions are checked. First if the weight of the new node is greater than weight of the cluster head with a minimum value m. If this condition is true for the new node then the battery drainage of the two nodes are checked. The battery drainage of the new node is less than the battery drainage of the cluster head with a minimum value d or not. If the new node satisfies these two conditions then the new node becomes cluster head otherwise no changes occur. As the re election of cluster head decreases, cluster becomes more stable than DMAC algorithm. Also computation time overhead for re election decreases. The new cluster head election is done on basis of weight as well on battery drainage of the node, i.e. election is done on basis of some parameter. In ad-hoc network battery life is an important issue for the network. So this algorithm is advantageous over DMAC algorithm.

3.3Modified DMAC Algorithm

3.3.1 Variables & Formulas used

(a)clusterHeadValue(v) is a flag which is true only for the cluster heads of the network.
(b)t_{ch} = measures the amount of time ,a node acts as clusterhead.
© tn = measures the amount of time a node acts as member node.
(d)**B**atteryDrainage(v) is a variable which shows the available battery power of a node v.
(e)BatteryDrainage of a member node which works during 1 time unit=dn
(f)BatteryDrainage of a member node which works during t time unit=dn*t

(g)If a node acts as clusterhead for the same amount of time its battery drainage will be 5% more than an ordinary node (suppose)

So it will be
=dn*t+5/100*dn*t=21/20*d_n*t
(h)Total BatteryDrainage of a node which works during t time unit=dn*t_n+21/20 *d_n*t_{ch}
(i)m is a variable which denotes the threshold value of weight difference of two node which is assigned as 10.
(j)d is a variable which denotes the threshold value of the battery drainage difference of the two node which is assigned as 20 unit.

3.3.2Algorithm init() of MODIFIED DMAC

```
start
if !isCHMessageSent(z == 0)
possibleClusterHead =
w_max(z){neighborList(v)}
if nodeWeight(v) >
nodeWeight(possibleClusterHead)
possibleClusterHead.receivedJoin(v)
masterClusterHead(v) = possibleClusterHead
else
for v : neighbors(z)
neighbor(v).receivedCHMessage(v)
end for
isCHMessageSent(v) = true
end if
else
for v : neighbors(z)
neighbor(v).receivedCHMessage(v)
end for
isCHMessageSent(v) = true
end if
timeCalculation()
end
```

3.3.3.Algorithm 2 receivedJoin(u) of MODIFIED DMAC

```
start
if isCHMessageSent(v) == true
if clusterContent(v).contains(u)
clusterContent(v).remove(u)
else
clusterContent(v).add(u)
end if
else if isCHMessageSent(v) == true
init()
end if
end
```


3.3.4Algorithm 3 failureLink(u) of MODIFIED DMAC

```
start
if isCHMessageSent(v) == true and
clusterContent(v).contains(u)
clusterContent(v).remove(u)
else if masterClusterHead(v) == u
init()
end if
end
```

3.3.5Algorithm4: newLink(u) of MODIFIED DMAC

```
Start
 if (weight (u) – weight (v) > =m)
 {
barreryDrainage()
{ if (batteryDrainage(v)-batteryDrainage(u)> =d)
  masterClusterHead(v)=u

v.receivedCHmessage(u)
for v: neighbors(z)
neighbors(v) .receivedJoin(v)
 end for
if isCHmessagesent(v)==true CHmessagesent(v)=false
 If isCHmessagesent(u)==false                    isCHmessagesent(u)=tru
                if(clusterHeadValue(u)==false)
  clusterHeadValue(u)=true
  endif
end
```

3.3.6Algorithm5: timeCalculation(v) of MODIFIED DMAC

```
start
if(clusterHeadValue(v)&& masterClusterHead(v)==true)
tch+=1
if(clusterHeadValue(v)&& MmasterClusterHead(v)==false)
tn+=1
else
    if (clusterHeadValue(v)==true&&   masterClusterHead(v)==false)
    tn+=1
end
```

3.3.7Algorithm 6: battery Drainage() of MODIFIED DMAC

```
start
if(clusterHeadValue(v)&& masterClusterHead(v)==true)
   BatteryDrainage =21/20dn*tch

   if(clusterHeadValue(v)&& masterClusterHead(v)==false)
else
   if (clusterHeadValue(v)==true&&      masterClusterHead(v)==false)
   BatteryDrainage =  dn*tn
End
```


4. Case Study

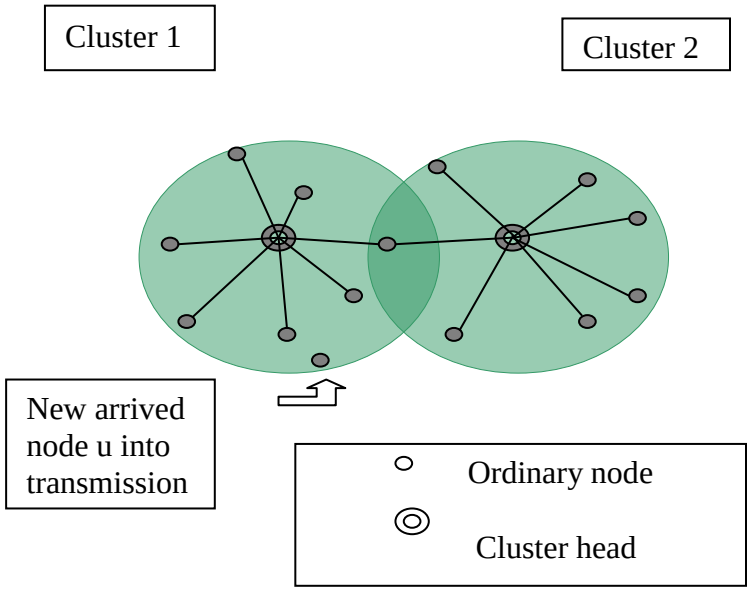

Fig2: Case Study

In the above figure a network is described which has two clusters. Every cluster has its own cluster head. The circles represent the transmission range of each cluster head. Here the circles are overlapped with each other which represent the common transmission range of the two cluster heads. The cluster heads are represented by the double circle and ordinary nodes are represented by circle. The node which lies within the common transmission range of the two cluster heads act as a gateway node. The inter cluster communication is done using the gateway node. Now if a new node u arrives within the transmission range of any cluster head then these following conditions can be possible:

(a)The weight of the new node u can be greater than the current cluster head node v and the weight difference can be greater than the threshold value m. In this case the new node u will be cluster head. The clusterHeadValue of u will set as true and the clusterheadValue of node v will set as false.

(b)The weight difference of new node u and the cluster head v can be less than m. In this case first the clusterHeadValue of the new node u is checked. If it is true then the remaining battery power of the two nodes is checked. If the battery power of the new node u is greater than d than of the remaining battery power of the cluster head node v then new node becomes the cluster head otherwise there will be no change.

(c)If the clusterHeadValue of the node u is false then the node u becomes the cluster head based on the assumption that two nodes working for the same time but one as cluster head and the other as ordinary node will have variable battery drainage i.e. the cluster head will have much battery drainage than the ordinary node So obviously the remPower of node u will be greater than the node v.

(d)The weight of new node u is less than the weight of cluster head node v. In this case no change will occur.

4.1Initial Cluster Setup

Cluster no	Cluster head	Member node
1	90	85,6576,70,35
2	60	15,29,35
3	175	120,140,100,95,110
4	150	155,165,135,105,137

To measure the performance of the algorithm 20 nodes are taken. The nodes are random and assuming they are moving to the outward direction. The nodes are communicating to each other. Now initially there are four clusters.
The threshold weight difference m=10
The threshold Battery drainage difference d=10 unit.

4.1.2Results

- Re election result for DMAC algorithm

- Now for displacement 1m/s there is only one re election of cluster head in DMAC as node with weight 70 is coming to cluster2. The new node becomes the new cluster head. Cluster head of cluster 2 changes from 60 to 70.
- When max displacement is 2m/s there are two re election as node 76,85 are coming into cluster 2.cluster head of cluster 2 changes from 70 to 76 then 76 to 85.
- When max displacement is 3m/s there are four re election. As nodes 90, 95, 100 are coming into cluster 2, cluster head of cluster 2 changes from 85 to 90 then 90 to 95, then 95 to 100. Node 105 is coming into cluster 1.cluster head of cluster 1 changes from 90 to 105.
- When max displacement is 4m/s there are eight re election. Nodes 135,155 are coming into cluster, Cluster head of cluster 1 changes from 105 to 135 then 135 to155.As nodes 110,120 are coming into cluster 2. Cluster head of cluster 2 changes from 100 to 110 then 110 to120. Nodes 165,175 are coming into cluster 3. Cluster head of cluster 3 changes from 150 to 165 then 165 to175. Nodes 140, 150 are coming into cluster 4. Cluster head of cluster 4 changes from 137 to 140 then 140 to150.

- Re election result for the modified DMAC algorithm
  In the modified DMAC algorithm, for displacement 1m/s there is only one re election of cluster head in DMAC as node with weight 70 is coming to cluster2. Cluster head of cluster 2 changes from 60 to 70.
- When max displacement is 2m/s there are one re election as node 76,85 are coming into cluster 2.cluster head of cluster 2 changes from 70 to 85. As the node 76 does not satisfy the threshold weight difference condition.
- When max displacement is 3m/s there are two re election. Node 105 is coming into cluster 1.cluster head of cluster 1 changes from 90 to 105.
  As nodes 90, 95, 100 are coming into cluster 2, cluster head of cluster changes from 85 to 95. The node 90 does not satisfy the battery drainage condition as it was previously a cluster head. Node weight 100 does not satisfy the threshold weight difference condition.
- When max displacement is 4m/s there are five re election.
  Nodes 135,155 are coming into cluster 1. Cluster head of cluster 1 changes from 105 to 135 then 135 to155. As nodes 110,120 are coming into cluster 2. Cluster head of cluster 2 changes from 95 to 110 then 110 to120. Nodes 165,175 are coming into cluster 3. Cluster head of cluster 3 changes from 150 to 165 .The node 175 does not satisfy the battery drainage condition as it was previously a cluster head. Cluster head of cluster 4 becomes node 137. As both the nodes 140,150 does not satisfies the conditions.
  Node weight 140 does not satisfy the threshold weight difference condition. The node 150 does not satisfy the battery drainage condition as it was previously a cluster head.

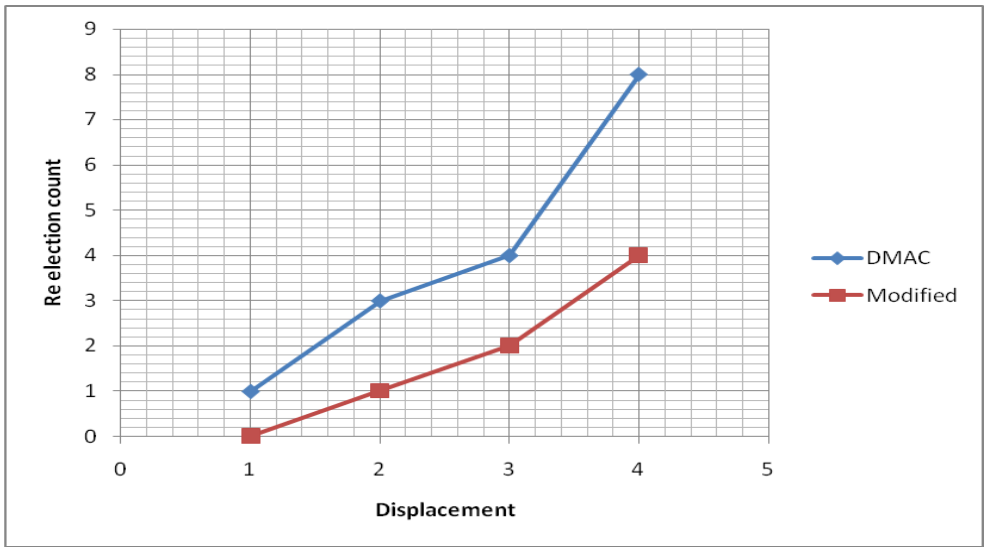

Fig3: Displacement vs. Relection Count for DMAC and ModifiedDMAC

This Modified DMAC algorithm gives a better performance when m is assigned a value of 12. Then re election count is zero for displacement 1m/s. As the new node as node with weight 70 is coming to cluster2. Cluster head of cluster 2 does not change.

- When max displacement is 2m/s there are one re election as node 76,85 are coming into cluster 2.cluster head of cluster 2 changes from 70 to 85. As the node 76 does not satisfy the threshold weight difference condition.
- When max displacement is 3m/s there are two re election. Node 105 is coming into cluster cluster head of cluster 1 changes from 90 to 105. As nodes 90,95,100 are coming into cluster 2, cluster head of cluster changes from 85 to 100. The node 90 does not satisfy the battery drainage condition as it was previously a cluster head.
- When max displacement is 4m/s there are four re elections. Nodes 135,155 are coming into cluster 1. Cluster head of cluster 1 changes from 105 to 135 then 135 to155. As nodes 110,120 are coming into cluster 2. Cluster head of cluster 2 changes from 100 to120. As the 110 node does not satisfy the threshold weight difference criteria. Nodes 165,175 are coming into cluster 3. Cluster head of cluster 3 changes from 150 to 165 .The node 175 does not satisfy the battery drainage condition as it was previously a cluster head .Cluster head of cluster 4 becomes node 137. As both the nodes 140,150 does not satisfies the conditions. Node weight 140 does not satisfy the threshold weight difference condition. The node 150 does not satisfy the battery drainage condition as it was previously a cluster head

5. Conclusion

The performance of the new algorithm is better than DMAC algorithm in the given example. This example is based on some random values. But in dynamically changing environment where a node can move in any direction the algorithm may produce different result. Here performance is not checked with variable transmission range or in terms of dominant set updates. It has to be tested by ns-2 simulator for confirmation of better performance.

References

[1] A. Amis and R. Prakash, Load-balancing clusters in wireless ad hoc networks, in: Proceedings of ASSET 2000, Richardson, TX, March 2000, pp. 25–32.
[2] D.J. Baker and A. Ephremides, A distributed algorithm for organizing mobile radio telecommunication networks, in: Proceedings of the 2nd International Conference on Distributed Computer Systems, April 1981, pp. 476–483.
[3] D.J. Baker and A. Ephremides, The architectural organization of a mobile radio network via a distributed algorithm, IEEE Transactions on Communications COM-29 11 (1981) 1694–1701.
[4] S. Basagni, I. Chlamtac and A. Farago, A generalized clustering algorithm for peer-to-peer networks, in: Proceedings of Workshop on Algorithmic Aspects of Communication (satellite workshop of ICALP), July 1997.
[5] S. Basagni, Distributed clustering for ad hoc networks, in: Proceedings of International Symposium on Parallel Architectures, Algorithms and Networks, June 1999, pp. 310–315.
[6] S. Basagni, Distributed and mobility-adaptive clustering for multimedia support in multi-hop wireless networks, in: Proceedings of Vehicular Technology Conference, VTC, Vol. 2, 1999-Fall, pp. 889–893.
[7] B. Bollbas, Random Graphs (Academic Press, 1985).
[8] M. Chatterjee, S.K. Das and D. Turgut, An on-demand weighted clustering algorithm (WCA) for ad hoc networks, in: Proceedings of IEEE GLOBECOM 2000, San Francisco, November 2000, pp. 1697–1701.
[9] I. Chlamtac and A. Farago, A new approach to the design and analysis of peer-to-peer mobile networks, Wireless Networks 5(3) (August 1999) 149–156.
[10] A. Ephremides, J.E. Wieselthier and D.J. Baker, A design concept for reliable mobile radio networks with frequency hopping signaling, in: Proceedings of IEEE, Vol. 75(1) (1987) 56–73.
[11] M. Gerla and J.T.C. Tsai, Multicluster, mobile, multimedia radio network, Wireless Networks 1(3) (1995) 255–265.
[12] http://www.bluetooth.com
[13] 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 Communications (August 1999) 1415–1425.

