Abstract: This paper presents a distributed algorithm for adaptive movement of nodes in a MANET (Mobile Ad-hoc Network) to maintain the overall topology of the network. The proposed algorithm assumes the presence of a GPS receiver in each node. A node measures the relative position and velocity of its neighbors and checks whether a certain condition is satisfied or not. If the condition is satisfied it keeps its velocity unchanged otherwise a simple heuristic algorithm is used to modify the node velocity with respect to that of its neighbors so that the topology remains unchanged in subsequent beacon intervals. The simulation run of the algorithm is carried out on a few synthetically generated network scenarios and results thus obtained show the effectiveness of the proposed algorithm.

Keywords: Mobile Ad-Hoc Network, Topology Management, Distributed Algorithm

I. INTRODUCTION

A Mobile Ad-Hoc Network (MANET) is a group of wireless autonomous mobile nodes forming a typical environment to communicate with each other dynamically without any fixed infrastructure or administration [1]. In such a network the nodes can move in any arbitrary manner without any prediction. Each node acts as a usual trans-receiver, which can also find the network routes to aid the network to form a complete connected graph. As a result, routing and topology management has become an important issue in a MANET. Many researchers have developed efficient routing protocols which ensure to find the exact route to connect the transmitting mobile node to its intended caller, may be outside the transmitting range of the transmitter, via other node(s) without having much delay and unnecessary control overhead.

Existing routing protocols for MANET can be classified into four different basic categories namely flooding, proactive routing, reactive routing and dynamic cluster based routing [2]. However none of these routing schemes guarantees constant network connectivity during the movement and each of these schemes have constant route maintenance overhead. A particular node may even be disconnected in the worst case.

Centralized topology management schemes in [3 and 4] discuss a self-adaptive movement control algorithm, which ensures the retention of network connectivity even during the positional variation of the nodes. But in this case, a coordinator has to be elected and all other nodes should follow the instructions from the coordinator to maintain the topology. The main disadvantages of the centralized topology management scheme are increase in control overhead and non-scalability. Once the coordinator fails to perform, the whole network becomes non-functional.

In our paper, we have suggested a distributed topology management algorithm where all the nodes of the network will retain its topology throughout the operation so that the nodes can follow a fixed routing, which eliminates the routing overhead. Each node is provided with a GPS receiver that receives its position as well as the velocity (in both magnitude and direction) information. All the nodes can separately calculate its own velocity, on the basis of the position and velocity information received from its neighbors, for the next beacon interval in such a way that the overall topology of the network remains same. The algorithm was also simulated in a synthetically designed environment and encouraging results were obtained.

The paper is organized as follows. Next section provides the formal definition of the topology management problem. In the third
section we present the proposed distributed algorithm for maintaining the topology. Simulation results are presented in the fourth section. The next section presents a comparative study between the proposed algorithm and the centralized algorithm in [3]. We finally conclude the paper in section six. The appendix contains the proof of Lemma 1 and Lemma 2.

II. TOPOLOGY MANAGEMENT PROBLEM

Given the physical topology of a mobile ad-hoc network, the problem is to control the movements of the individual nodes so as to maintain a stable neighborhood topology so that the nodes are able to communicate amongst themselves without the need of any variable routing protocols.

Let us consider a MANET consisting of N number of nodes \( n_0, n_1, n_2, \ldots, n_{N-1} \). We assume that each node has a maximum transmission range of \( R_{\text{max}} \). Now, any two nodes \( n_i \) and \( n_j \) are called neighboring nodes if they can communicate amongst themselves without the need of any routing. So any two nodes \( n_i \) and \( n_j \) will be neighbors if the distance between them \( D(i,j) \leq R_{\text{max}} \). Hence the network topology will be maintained if \( D(i,j) \leq R_{\text{max}} \) for any two neighboring nodes \( n_i, n_j \) at any time \( t \).

III. PROPOSED ALGORITHM

Assumptions

The algorithm is based on following assumptions:

1) All nodes are enabled with GPS receivers. These receivers can furnish the current position and velocity information of an individual node.
2) All the nodes have a predefined maximum velocity, \( V_{\text{max}} \).
3) Acceleration and deceleration of the nodes are taken to be instantaneous.
4) It is assumed that if the nodes are within the appropriate range, the instruction message will never be lost in transit.
5) At the beginning, all the nodes have a configuration that creates a connected topology.
6) Each node has a unique identification number.

Neighborhood selection algorithm

Let, \( R_{\text{max}} \) be the maximum range of message communication of each individual node. Then at the beginning a node selects as its neighbor, all nodes which are at a distance less than \( R_{\text{max}} \) from that node. Next through Hello packets it sends its current position, velocity and node identification number to all its neighboring nodes and also receives the same from its neighbors. A node stores position and velocity information and identification number of its neighbors. This concludes the neighborhood selection procedure.

Movement Algorithm

Each node sends its current position and velocity information to all its neighboring nodes with the help of hello packets after a fixed duration \( T \). This time interval \( T \) is defined as the “Beacon Interval”. At the beginning, each node is assigned a velocity at random. Then in each subsequent beacon interval it calculates the current distance from its neighbors and also predicts the new distance from its neighbors after the end of the beacon interval based on the relative velocity between the nodes. If the new distance exceeds a predefined threshold distance \( R_{\text{th}} \) then it modifies its velocity in such a way that the relative velocity between the pair of nodes decreases and the relative distance does not exceed the threshold distance, otherwise it keeps its velocity intact. This procedure is followed by all the nodes and enables the network to maintain its topology. Next we analyze the algorithm in more detail.

![Fig.(1): Illustrating the movement algorithm.](image-url)
Let A be a node which calculates its velocity for its neighboring node B. Let, the current velocity of node A be \( \vec{V}_A \) and that of node B be \( \vec{V}_B \); the current distance between them be \( d_{AB} = d_A - d_B \); and the threshold distance be \( R_{th} \). Choice of \( R_{th} \) is given in Lemma 1 in appendix. The relative velocity between the two nodes is \( \vec{V}_{AB} = \vec{V}_B - \vec{V}_A \). After a time \( T \) the new predicted distance between them is given by,

\[
\left| d_{AB}\right|_{\text{New}} = d_{AB} + \vec{V}_{AB} \cdot T.
\]

Now two cases may arise, if \( \left| d_{AB}\right|_{\text{New}} < R_{th} \) then node A keeps its velocity intact. But if \( \left| d_{AB}\right|_{\text{New}} > R_{th} \) then node A decreases the relative velocity between the nodes by increasing its velocity, as given in Lemma 2 in appendix, such that the new distance \( \left| d_{AB}\right|_{\text{New}} \) becomes less than \( R_{th} \). In this way node A calculates its velocity for the next beacon interval for all its neighboring nodes. Node A also associates a weight for each calculated velocity as follows for \( i^{th} \) node,

\[
W_i = \left( d_{\text{new}} \text{ (predicted)} - R_{th} \right) + 1 \quad \text{for } d_{\text{new}} \text{ (predicted)} > R_{th} \\
= 1 \quad \text{otherwise.}
\]

Now, the node calculates its velocity for the next interval as \( \vec{V} = \sum W_i \vec{V}_i / \sum W_i \). Where \( \vec{V}_i \) is the velocity calculated for the \( i^{th} \) neighbor and \( W_i \) is its corresponding weight. By choosing the velocity in this manner the velocity calculated for the node for which the predicted distance exceeds \( R_{th} \) gets greater precedence over the velocities calculated for other nodes. In this way by adjusting its velocity after each beacon interval the network is able to maintain its topology.

**IV. SIMULATION RESULTS**

We simulated the algorithm assuming a hypothetical network and obtained encouraging results. The simulation was conducted using Matlab software in Windows environment. For our simulation we considered five nodes with random initial velocities. The following parameters were chosen for simulation. Maximum communication range \( R_{max} = 50 \text{ Km} \), maximum velocity of a node \( V_{max} = 80 \text{ Km/hr} \), the beacon interval \( T = 6 \text{ minutes} \). Therefore, the threshold distance \( R_{th} \) was chosen according to Lemma 1, \( R_{max} - 2V_{max} \cdot T = 34 \text{ Km} \). so \( R_{th} \) was taken as 30 Km. The initial positions and velocities of the nodes were as follows.

<table>
<thead>
<tr>
<th>Node</th>
<th>Position (Km)</th>
<th>Velocity (Km/hr)</th>
<th>Neighboring Nodes</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>(0,0)</td>
<td>50 10</td>
<td>2,3,4,5</td>
</tr>
<tr>
<td>2</td>
<td>(30,0)</td>
<td>55 50</td>
<td>1,4,5</td>
</tr>
<tr>
<td>3</td>
<td>(-30,0)</td>
<td>60 35</td>
<td>1,4,5</td>
</tr>
<tr>
<td>4</td>
<td>(0,30)</td>
<td>60 20</td>
<td>1,2,3</td>
</tr>
<tr>
<td>5</td>
<td>(0,-30)</td>
<td>20 60</td>
<td>1,2,3</td>
</tr>
</tbody>
</table>

The simulation was carried out for an interval of 60 hours=3600 minutes. The results obtained are shown in figure 2. Figure 2 shows the distance variation of the neighbors from node 1. Similar graphs were also obtained for other nodes. From the graph it is clear that the distance between the neighboring nodes never exceed the maximum communication range \( R_{max} = 50 \text{ Km} \). Hence this clearly shows the effectiveness of the proposed algorithm is maintaining the topology of a MANET.

**V. PERFORMANCE COMPARISON**

We also perform a comparative study between the distributed topology management algorithm presented in this paper and the centralized topology management algorithm presented in [3]. Both distributed and centralized algorithms are able to maintain the topology of a MANET, but the distributed algorithm is better since it completely eliminates the control overhead of the coordinator, which was present in the centralized approach. In the centralized topology management algorithm [3] the coordinator had to issue appropriate commands to the nodes in order to maintain the network topology and to ensure that a node always remains in contact with the coordinator. But this greatly increases the control overhead of the coordinator, and hence that of the network. Distributed algorithm is much superior as it eliminates all additional control overheads present in centralized algorithms. Thus, the control overhead of a node is reduced to just sending hello messages containing its current position and velocity, as received by their GPS receivers, to its neighboring nodes, after each beacon interval \( T \). Unlike the centralized algorithm in [3] which considered velocities of the nodes only in one
direction, the distributed algorithm presented here is able to maintain the topology even when the nodes possess velocities along any direction.

Fig.(2): Distance of the neighboring nodes from node 1.

VI. CONCLUSION

In this paper, we have presented a distributive algorithm to maintain the network topology in a MANET. The algorithm is effective even for nodes possessing velocities along any direction. Application of a distributed scheme reduces the control overhead as compared to a centralized approach and the network is not vulnerable to failure if a particular node becomes non-functional, as there is no concept of central coordinator. The simulation results show the effectiveness of the algorithm. Presently, we are working on a distributive scheme where the nodes will have better flexibility in choosing its velocity and still maintain the network topology.

REFERENCES


APPENDIX

Lemma 1: Choice of threshold distance $R_{th}$.

If $R_{max}$ be the maximum range of communication, $V_{max}$ be the maximum node velocity and $T$ be the beacon interval then the threshold distance $R_{th}$ should be chosen as $R_{th} < R_{max} - 2*V_{max}*T$

Proof: The maximum relative velocity between any pair of nodes is $2V_{max}$ (when they move in opposite direction with maximum velocity). The threshold distance must be so chosen that if initially the nodes are at a distance $R_{th}$ moving in opposite direction with maximum relative velocity, they must not cross the maximum range of communication $R_{max}$ after the beacon interval $T$. Hence $R_{th} + 2*V_{max}*T < R_{max}$ or

$$R_{th} < R_{max} - 2*V_{max}*T$$

Lemma 2: Choice of the relative velocity $V_{rel}$ if the new predicted distance between the nodes exceeds the threshold distance $R_{th}$.

If $d_x$ and $d_y$ be the current distance between the nodes along X and Y axis, $\theta$ be the angle between the direction their relative velocity vector and the X axis and $T$ be the beacon interval, then the new relative velocity must be chosen such that

$$0 < V_{rel} < \sqrt{(d_x \cos \theta + d_y \sin \theta)^2 + R_{th}^2 - d^2} - (d_x \cos \theta + d_y \sin \theta)/T$$

Proof: Let $\overline{d} = d_x \hat{a}_x + d_y \hat{a}_y$ be the current distance between the nodes (refer figure 1), where $\hat{a}_x$ and $\hat{a}_y$ are the unit vectors along X and Y axis respectively. Let $V_{rel}$ be the new chosen relative velocity. Then we must have,

$$| \overline{d} + V_{rel} * T | < R_{th}$$

Let $\overline{V_{rel}} = V_{rel} \cos \theta \hat{a}_x + V_{rel} \sin \theta \hat{a}_y$. Then we have

$$\sqrt{(d_x + V_{rel} \cos \theta * T)^2 + (d_y + V_{rel} \sin \theta * T)^2} < R_{th}$$

Squaring and solving for $V_{rel}$ we have,

$$0 < V_{rel} < \sqrt{(d_x \cos \theta + d_y \sin \theta)^2 + R_{th}^2 - d^2} - (d_x \cos \theta + d_y \sin \theta)/T$$

Where we have taken $V_{rel}$ as positive and have used the identity $d^2 = d_x^2 + d_y^2$. Hence the proof.