ASSIGNMENT - 1

WIRELESS MOBILE COMMUNICATION

1) MANET Routing protocols:

In Mobile Adhoc Network, nodes do not know the topology of their network, instead they have to discover it by their own as their own as the topology in the ad-hoc network is dynamic topology. The basic rule is that a new node whenever enters into an ad-hoc network, must announce its arrival and presence and should also listen to similar annocement broadcasts made by other mobile nodes.

PRO-ACTIVE ROUTING PROTOCOLS:

These are also known as table-driven routing protocols. Each mobile node maintain a seperate routing table which contains the information of the routes to all the possible destination mobile nodes.

Since the topology in the mobile ad-hoc network is dynamic, these routing tables are updated periodically as and when the network topology changes. It has a limitation that is doesn't work well for large network as the entries in routing table becomes too large since they need to maintain the route information to all possible nodes.

(i) Destination Sequenced Distance Vector Routing protocol (DSDV):

→ It is a pro-active / table driven routing protocol. It actually extends the distance vector routing protocol of the wired networks. It is based on Bellman-Ford routing algorithm.

→ Destination Sequence number added with every routing entry in the routing table maintained by each node. A node will include the new update in the table only if the entry consists of the new updated route to the destination with higher sequence number.

(ii) Global State Routing (GSR):

It is a pro-active / table driven routing protocol. It actually extends the link state routing of the wired networks. It is based on the Dijkstra's routing Algorithm. Link State routing protocol was not suited for mobile ad-hoc networks because in it, each nodes floods the link state routing information directly into the whole network i.e Global flooding which may lead to the congestion of control packets in the network.

Hence, as a solution Global State Routing protocol came into does'nt flood the link state routing packets globally into the network. In GSR, each of the Mobile node maintains one list and three tables namely, adjacency list, topology table, next hop table and distance table.

REACTIVE ROUTING PROTOCOLS :

These are also known as on-demand routing protocol. In this type of routing, the route is discovered only when it is required / needed. The process of route discovery occurs by flooding the route request packets throughout the mobile network. It consists of two major phases namely, route discovery and route maintainance.

(i) Dynamic Source Routing protocol (DSR) :

It is a reactive / on-demand routing protocol. In this type of routing, the route is discovered only when it is required / needed. The process of routing discovery occurs by flooding the route request packets throughout the mobile network. It consists of two phases :

→ Route Discovery :

This phase determines the most optimal path for the transmission of data packets between the source and

destination mobile nodes.

→ Route Maintainance:

This phase performs the Maintenance work of the route as the topology in the mobile ad-hoc network is dynamic in nature and hence, there are many cases of link breakage resulting in the network failure between the mobile nodes.

(ii) Ad-HOC on Demand Vector Routing protocol: ~~(AODC~~ (AODV):

It is a reactive /on-demand routing protocol. It is an extension of dynamic source routing protocol (DSR) and its helps to remove the disadvantage of dynamic source routing protocol. In DSR, after route discovery, when the source mobile node sends the data packet to the dustination mobile node, it also contains the complete path in its header, Hence, as the network

Size increases, the length of the complete path also increases and the data packet's header size also increases which makes the whole network slow.

Hence Ad-hoc on Demand Vector Routing protocol came as solution to it. The main difference lies in the way of storing the path, AODV stores the path in the routing table whereas DSR stores it in the data packet's header itself. It also operates in two phases in the similar fashion: Route discovery and Route Maintainance.

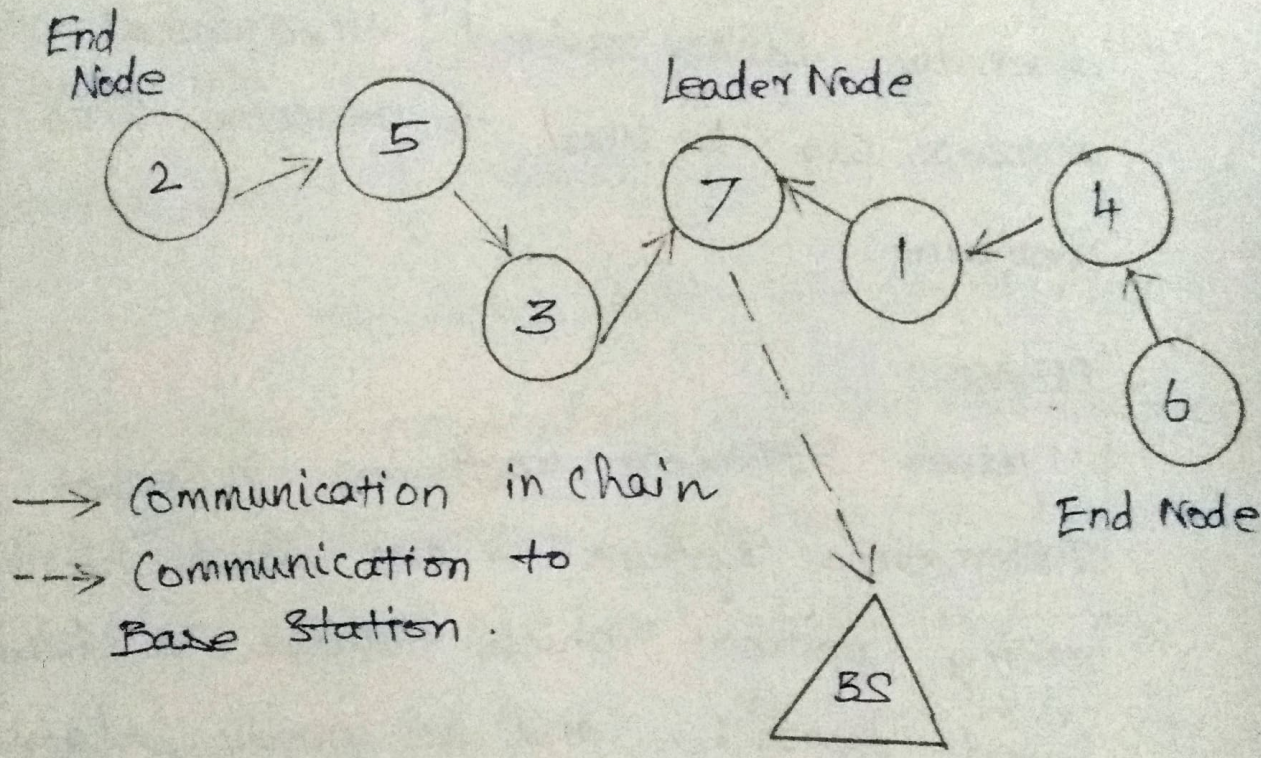
2) PEGASIS in WSN :

In Wireless Sensor Network energy efficiency plays a crucial role as the sensors are generally battery powered. Hierarchical routing protocols can be used to overcome this constraint.

PEGASIS :

power Efficient Gathering in Sensor Information Systems is one such hierarchical routing protocol which follows a chain based approach and a greedy algorithm. The sensor nodes organize themselves to form a chain. If any nodes dies inbetween the chain is reconstructed to bypass the dead node. A leader or Cluster head node is assigned and it takes care of transmitting data to the base station / Sink node. The main goal of pegasis is to receive and transmit data to and from

the neighbour and take turns being the cluster head for transmission to the Sink node.



PEGASIS in Netsim with MATLAB Interfacing:

PEGASIS algorithm is implemented in Netsim by Interfacing with MATLAB for the purpose of mathematical Calculation. The Sensor Coordinates are fed as input to MATLAB and PEGASIS algorithm that is implemented in MATLAB is used to dynamically form a Chain between

the nodes and to elect one of them as head node.

From MATLAB the order of devices in the chain and the head node id is retrieved to perform routing in Netsim.

All the above steps are performed during each Transmission and can be defined as per the implementation. Each time a node dies, the chain will be reconstructed. Also nodes take turn to become the head node in each iteration.

The codes required for the mathematical calculations done in MATLAB are written to a PEGASIS.m file.

A PEGASIS.c file is added to the DSR project which contains the following functions:

fn_Netsim_PEGASIS_CheckDestination()

This Function is used to determine whether the current device is the destination of a packet or an Intermediate node.

fn_Netsim_PEGASIS_GetnextHop()

This function handles routing in the Sensor network by determining the next hop device based on the chain that is formed as part of PEGASIS protocol.

fn_Netsim_PEGASIS_run()

This function makes a call to MATLAB interfacing function and passes the Inputs from Netsim to MATLAB and also retrieves the computed parameters from MATLAB workspace for further calculations in Netsim.

fn_Netsim_PEGASIS_form_Clusters()

This Function updates the information obtained from MATLAB to identify the head node and the neighbouring nodes in the PEGASIS Chain.

Static Routing:

Static Routing is defined in such a way that the Sensors send packets to neighbouring node in the PEGASIS chain which is closet to the head node. Once packet arrives at the head node it is fomarded to the destination or the Sink node.

To run this code 64-bit Version of MATLAB must be installed in the System.

(i) The downloaded project folder contains the folders Documentation, MATLAB_Code and Dynamic_Clustering_Workspace directory.

(ii) Import Dynamic_Clustering_Workspace by going to open Simulation → Workspace Options → more Options in Netsim Home window.

(iii) It displays a window where users need to give the path of the Workspace folder and click on ok.

(iv) Browse to the Dynamic_Clustering_Workspace folder and click on Select Folder.

(v) After this click on OK button in the Import workspace Window.

(vi) While Importing the Workspace, if the following warning message indicating Software Version mismatch is displayed, you can ignore it and proceed.

(vii) The imported Workspace will be set as the current Workspace automatically. To see the imported Workspace, Click on Open Simulation → Workspace options → More Options.

(viii) Create a User Variable with the name of MATLAB_PATH and provide the path of the installation directory of User's respective MATLAB Version.

(ix) Make Sure that the following directory is in the path (Environment Variable) <path where MATLAB is installed>\bin\win64.

(x) Open Command prompt as admin and execute the Command "matlab -regserver". This will register MATLAB as a COM automation Server and is required for Netgrim to Start MATLAB automation server during runtime.

(xi) place Clustering.m present in the MATLAB_Code folder inside the root directory of MATLAB.

For Example: "C:\Program Files\MATLAB\R2016a".

(xiii) Open the Source Codes in Visual Studio by going to open Simulation → Workspace Options and clicking on Open Code button.

(xiii) Under the DSR project in the Solution explorer you will able to see that MATLAB - interface.C and Dynamic - Clustering.C files which contains Source Code related to interactions with MATLAB and handling Clustering in Netsim respectively.

(xii) Based on whether you are using Netsim 32 bit or 64 bit setup you can Configure Visual Studio to build 32 bit or 64 bit DII files respectively.

(xv) Right Click on the DSR project in the Solution explorer and select Rebuild

(xvi) Upon Successful build modified libDSR.dll file gets automatically updated in the directory containing Netsim binaries

(xvii) Run Netsim as Administrative mode.

(xviii) Then Dynamic_Clustering_Workspace comes with a Sample configuration that is already saved. To open this example, go to open

Simulation and click on the Dynamic_Clustering _Example that is present under the list of experiments.

(ixx) The network Scenario consists of 100 Sensors deployed randomly in a 50x50m grid environment with a Sink node placed in the centre.

Run the Scenario:

Observe the Simulation. Netsim automatically intialises MATLAB and the plots will be displayed.