CHAPTER 4

MARI TOPOLOGY FORMATION AND MANAGEMENT

Minimizing energy consumption is very important as well as a difficult challenge in mobile adhoc networking. Wireless network interface is often the device's single largest power consumer. Since the network interface may often be idle, turning a node off when not in use could save a considerable amount of power. In practice, however, this approach is not straightforward. A node must arrange to turn itself ON to send packets, to receive packets addressed to it and to participate in any higher-level routing and control protocols. The requirement of cooperation between power saving [76] and routing protocols is particularly acute in the case of multi-hop adhoc wireless networks, where nodes must forward packets for one another [56,90,91].

4.1. PROPOSED TOPOLOGY MANAGEMENT SCHEME

This thesis proposes a novel topology management scheme for adhoc wireless networks mainly intended for power management and secure routing. It is called, the MARI topology. Nodes in this scheme are classified into three categories based on their power level. They are:

1. MARI nodes: MARI nodes are selected in such a way that they have the maximum power level among their one-hop neighbors and all non-MARI nodes in
the one-hop neighborhood are within the transmission range of MARI nodes. These MARI nodes [115] have the routing intelligence i.e. they make decisions related to routing, such as path finding. Every MARI node has a group of member nodes connected to it, usually in its one-hop neighborhood. The responsibility of every MARI node is to make necessary communication with any member node connected to itself or with other MARI nodes (through Gateways) within the network, for both transmission and reception of the packets. MARI nodes are selected or formed by a procedure that is explained later in this chapter.

2. **Gateway nodes:** The Gateway nodes [118] having sufficient power level are selected by the MARI nodes such that they can be used to forward packets between MARI nodes. Any two adjacent MARI nodes (within two-hop distance usually) in the network are connected through the concerned Gateway node only. Gateway nodes do not have routing intelligence. The MARI nodes select these Gateway nodes, according to the procedure outlined later. *The MARI and the Gateway nodes stay continuously awake to route the packets of other member nodes.*

3. **Member nodes:** A member node is a non-MARI and non-Gateway node. These are the nodes, which want to communicate with each other [118,119]. Every member node is connected to one of the MARI nodes (some kind of belonging or bonding) through which it transmits or receives the packets. The member nodes wake up only at certain specified time epochs, and for very short periods, during any given beacon period $T$. When a member node wakes up and if it does not have to transmit or receive data, then it goes to sleep mode again, after a brief period.
This is the main principle behind the power-efficient operation of the network. The wake-up time epochs of each member node are determined apriority (pre-determined). In an actual network, these wake-up time epochs and wake-up durations may be determined as per the expected need. In our simulation of the operation of an adhoc wireless network, this is accomplished with the help of pseudo-random number generator. Also, these wake-up time epochs of a member node are known to its corresponding MARI node and its one-hop neighbor nodes, through the WAKEUP messages/packets that are exchanged at the beginning of a beacon period. Thus, the member node can remain in power saving sleep mode most of the time [72,99], especially when it is not actively sending or receiving packets. The packets are routed over the virtual backbone consisting of MARI nodes and Gateway nodes, which are awake continuously. This is the main power-saving advantage of the topology that is suggested and nurtured in this thesis.

If a member node is in sleep mode at a point of time when packets destined for it reach the respective MARI node, then the packets are not lost but are buffered at the respective MARI node. When the member node wakes up, it can retrieve these packets from the buffering MARI node. This is made easy since the MARI node knows the wake-up time epochs of its member nodes. This topology management scheme makes the routing simple. That is because only the routing table and information pertaining to the nodes in the virtual backbone are considered, for routing from any member node to any other member node.
Figure 4.1.1 shows an example adhoc wireless network with 100 nodes, which are distributed randomly. This was generated using pseudo random number generator, on MATLAB. Each node is represented by an address, called its id, which is essentially its node number.

Fig.4.1.2. Shows the adhoc wireless network with MARI nodes, Gateway nodes and member nodes. This structure is obtained after going through the procedure of topology formation, as explained later in this chapter. It also shows the connectivity of all the MARI nodes and Gateways to form a virtual backbone for routing the packets of member nodes. Here the number of MARI nodes is 9 and the number of Gateway nodes are 12 and all the remaining nodes (79 nodes) are member nodes. It also shows that the number of nodes in the virtual backbone...
path is only approximately 20% of the total nodes [119]. This is given in the table 4.1.1.

Fig.4.1.2. Random node distribution of nodes in an Adhoc wireless network with MARI nodes, Gateway nodes and member nodes (Total 100 nodes)

From the table 4.1.1, the lists of nodes are:

MARI nodes are: 2, 26, 33, 41, 48, 52, 72, 80 and 85 (Total 9 nodes)

Gateway nodes are: 3, 12, 16, 18, 19, 22, 27, 54, 64, 93, 94 and 100 (Total 12 nodes)

Some more definitions are as follows.

Sleep Cycle period \((T_m - T_{m+1}, m = 1, 2, \ldots, s)\) is the time period during which a member node remains in the power efficient sleep mode. There is a number of such sleep cycles during any given beacon period. A member node wakes up once in every fixed time sleep cycle duration. Usually, a beacon period is subdivided into a number of sleep cycles as illustrated later.
Table 4.1.1 List of MARI nodes, Gateway nodes and Member nodes

An Undecided Node is one, which is not connected to any MARI node, because of which; it is out of transmission range from any MARI node.

The time axis is divided into beacon periods each of which is of duration $T$. Initially all nodes are Undecided nodes. The topology is to be evolved or formed. We assume that each node at the beginning, while awake, broadcasts to all its one-hop neighbors a WAKEUP packet that contains:

- Node's id (8-bits)
- Its status (i.e., whether the node is a MARI node, Gateway, member, or undecided) (2-bits)
- It's current POWER level, (8-bits)
- It's current MARI node, if exists, (8-bits)
- Its wakeup counter which is designated as $w_i$, for node $i$ (7-bits)
it's MARI node etc.), if available.(5+n*18)

Explanation about each field:

(1) Each Node is represented by 8-bits, so a network can have max of 256 nodes.

(2) The status of the node is represented by 2-bits, if the value of two bits is:

   11-MARI node, 10- Gateway node, 01- Member node and 00- Undecided node.

(3) The energy level is indicated by 10-bits, and its value is given from 1.0WH to 20WH (i.e 25W to 1000W) to each node.

(4) Current MARI node is represented by 8-bits, this field is valid in the 2nd field value is 10 or 01, otherwise it has no meaning. Initially its value is ‘00’ to indicate that, there is no MARI node, in the first time WAKEUP packet.

(5) Wake up counter (7-bits) indicates the wakeup time epoch of the node‘i’ in each sleep cycle of each beacon period.

(6) Information about each neighbor may gives overhead of 18-bits each along with the common 5-bits.

Hence, the size of the WAKEUP packet from each node with an average of 10 neighbors is 220-bits. All the nodes, during the broadcast window, transmit this overhead information. For a network with 100 nodes the total overhead message is approximately 20 packets of nearly 1Kb each at first time. Once the neighbors, their power levels and status are known, the status packets are broadcasted by all the MARI nodes to their respective member nodes.
Every MARI node transmits ‘x-number of status packets, for x-number of neighbors’ (one packet to each neighbor). Each Status packet contains the following fields:

- MARI node id (8-bits).
- Neighbor node id (8-bits).
- Its status (2-bits)
- Current power level (8-bits)
- List of neighbors (8-bits) for each neighbor,

The total length of the each status packet (if the number of neighbors is 10) is: 106 bits. These status packets are transmitted by each MARI node to its all one-hop neighbors (members nodes and Gateway nodes) in the network. The total overhead is approximately 2Kb. This overhead is less compared with the WAKEUP packet overhead, since the total number of MARI nodes is only 10% to 20% of the total nodes in the network.

The Wake-up counter \( w \) has the information regarding the wake-up time epochs of node \( I \), during that beacon period. Such a broadcast is done by nodes, while awake, at the beginning of every beacon period, in order to evolve the topology again and dynamically. Usually, the WAKEUP packet is for exchanging information regarding the nodes and their one-hop neighbors. Once the information and status of every node is known, then every MARI nodes sends the STATUS packet to its one-hop neighbors. This STASUS packet sometimes also called as the second WAKEUP packet (some times it is al is for communicating the node-type
decisions among themselves). The STATUS packet is given usually by MARI nodes. This is how the topology gets formed. Thus, the topology of the network may change at the beginning of every beacon period.

Based on the WAKEUP packets received from neighbors, each node constructs a list of its neighbors, their MARI nodes, power levels, wakeup counters and other information about them.

A node can switch state from time to time between being a MARI node and being a member node, depending on its power level, as we shall see in detail later. When a node becomes a MARI node, only then its routing intelligence gets ‘activated’. A node becomes a Gateway, if its MARI node chooses it as a Gateway node. Gateway nodes are used to route the packets among MARI nodes. A node switches its state to ‘Undecided’, if it loses contact with its MARI node due to mobility. A node includes its current state in its WAKEUP messages/packets.

The network operational time is fragmented into beacon periods, each of which is of duration $T$. At the beginning of each beacon period, WAKEUP messages/packets are transmitted/exchanged, in order to evolve or form a suitable topology and virtual backbone, for managing the network. A given beacon period, say $(0-T)$, is subdivided into the broadcast window $(0-T_1)$ and $s$ number of sleep cycles, $(T_1-T_2)$, $(T_2-T_3)$, $\ldots$, $(T_m-T_{m+1})$, $\ldots$, $(T_s-T)$. These $s$ sleep cycles are of equal duration.

The maximum number of simultaneous node-to-node transmissions that are allowed in the network is an important issue in its management. That number can
be termed as the multi-access level, $l$. The multi-access level needed for certain performance optimizations can form very important issues for further research work and investigations. However, that is not in the scope of the present thesis, but highlighted towards further work in this area, for further doctoral theses and sponsored projects. In our work, we choose the uni-access communication. At any given time epoch, the number of node-to-node transmissions is one or zero. The reason is, in uni-access communication ($l = 1$), the energy levels drain rather slowly, of course at a cost in the throughput level. We have chosen this way mainly because the scope of this thesis is the successful introduction of our novel MARI topology and not the research on optimal multi-access level. In this framework, each sleep cycle is further divided into $M$ slots, where $M$ is the total number of member nodes in the network. Initially, that is for the sleep cycles of the first beacon period, these time slots are set to be of equal duration. In each such duration, only one member node can come to wake-up mode and communicate (transmit or receive) with its MARI node. That maximum wake-up duration of the member node is denoted by $\tau$, during its slot. The maximum wake-up duration $\tau$ need not be a constant value, it is rather adaptive. The assignment of slots and hence $t_{im}$, $i = 1, 2, \ldots$, can be adaptive. This is illustrated in Figure 4.5.1 and also in the text. In our simulations, we have used pseudo-random number generators combined with wake-up counters are used in order to assign the slots to member nodes (that is, to determine $t_{im}$'s and $\tau$ values, in the beginning of beacon
periods and these values are communicated through the WAKEUP messages/packets).

The following sections describe the procedure for becoming a MARI node, for withdrawal from being a MARI node [120], and also how a MARI node selects its Gateway nodes.

4.2. MARI NODE PLACEMENT

MARI nodes along with Gateway nodes form the virtual backbone for the network. This backbone is used for routing the packets of all member nodes from one to another, as required. Hence MARI nodes and Gateway nodes would need additional power for being awake continuously, for transmission, reception, processing and routing packets through the backbone. Thus, these MARI nodes should be selected in such a way that they have enough power level [120, 121] (see algorithm 4.2.1.).

- Undecided nodes periodically check if it has a maximum POWER level among its one-hop neighbors who have not joined to any MARI node (i.e. undecided neighbors). If a node has maximum power level among such one-hop neighbors, it becomes a MARI node and declares itself as a MARI node in the status field of the next WAKEUP packet.

- If an undecided node knows that its neighbor node has become a MARI node from a received STATUS message/packet, it changes it’s status to a member node. A node knows its status as a member node and identifies it’s current MARI node, in the STATUS packet. If more than one neighbors of an
undecided node became MARI, then the undecided node selects its MARI node from which it has received the STATUS packet earliest.

![Diagram of MARI node selection in a Random Adhoc Network](image)

**Figure 4.2.1 Illustration of MARI node selection in a Random Adhoc Network**

- **$P_m$** – member node power
- **$P_M$** – MARI node power

- There may be undecided nodes whose one-hop neighbors with POWER level more than that of the undecided node choose to join MARI nodes, as the MARI nodes have more POWER level than its one-hop neighbors. Such undecided nodes with maximum POWER level among one-hop undecided neighbors declare themselves as MARI nodes in the next, WAKEUP packet [116].

- A MARI node prepares a list of its member nodes, which joined the MARI node, from broadcast STATUS packets, received from its one-hop neighbors [117].
Algorithm 4.2.1 MARI placement algorithm for an adhoc wireless network (executed by all undecided nodes).

Max power = My power

for Each one-hop neighbor node $N_i$ do

if Status of node $N_i$ is MARI node then

My status = member node

My MARI node = $N_i$

else

if power of $N_i >$ Max POWER then

Max POWER = power of $N_i$

end if

end if

end for

if My status = Undecided AND My POWER > Max POWER then

My status = MARI node

end if

In this topology management scheme, we have assumed the channels are loss-less. If the channel is lossy and some beacons (‘beacon’ here means information exchanges during the beacon periods) are corrupted, this scheme is expected to select more number of MARI nodes than required, but the backbone formed by MARI nodes and Gateways will be connected quite well.
Table 4.2.1 shows the list of member nodes for the given MARI node 51. The number of member nodes for it is 10. The list of member nodes is, 5, 6, 11, 12, 13, 17, 34, 42, 45 and 59. All these Member nodes are one-hop neighbors of the MARI node 51.

4.3. MARI NODE WITHDRAWAL

MARI node would drain its energy more rapidly, as compared to member nodes, for obvious reasons. Before a MARI node loses major part of its POWER, the responsibility of that MARI node should be transferred to another appropriate member/Gateway node, which has sufficient POWER level. Generally MARI nodes should not be changed frequently; otherwise overheads will increase.

Table 4.2.1: List of Member nodes for the given MARI node 51
Figure 4.2.2 Flowchart for MARI Node Placement
When a MARI node observes that its POWER level is gone below a threshold, it will withdraw its status of MARI node. The withdrawal of MARI node is declared to its member nodes in the next WAKEUP message, declaring itself as a non-MARI node. That threshold can be set to 25% of POWER level of the initial value when the node decides to withdraw from being a MARI node to become a non-MARI node. The threshold value in a control parameter, which can play a pivotal role in the MARI topology network operation and management, and hence can be, is a subject of further study, research and development.

When a Gateway node or a member node comes to know that it is not able to contact its MARI node due to some reason or the other (for example, getting in to out of transmission range), it changes it’s status to ‘Undecided’, and it starts and undergoes the MARI node placement procedure.

4.4. GATEWAY NODE SELECTION

As the number of hops between any two MARI nodes is two or greater, Gateway nodes are required to forward packets between MARI nodes. Also as Gateways need to receive and transmit data packets to and from MARI nodes, they should have sufficient amount of POWER [117,118]. Fig 4.4.1 shows the Gateway selection and packet-flow in an Adhoc Wireless Network. The selection of Gateway nodes by the MARI nodes is illustrated in algorithm 4.4.1 and its corresponding flow chart is represented in Fig 4.4.2.
To determine the Gateway nodes, MARI node needs information of its two-hop neighborhood. This information is obtained from the WAKEUP packets that it receives from its one-hop neighbors and adjacent MARI nodes. But as the members of different MARI nodes are not synchronized, they (the MARI nodes) may miss some of the WAKEUP packets from members of adjacent MARI nodes. A MARI node periodically sends broadcast request packet STAY-AWAKE to its members, at appropriate times, to make them stay awake for the required brief periods as explained before, in every beacon period. This periodicity is about 6 to 10 times in a beacon period, to each member node, in our simulations.
• MARI node finds out all the MARI nodes within two hops. MARI node selects one (or more) of its members as Gateway. This Gateway should have the maximum power-level among all the member nodes within two-hops. The decision of Gateways is taken such that Gateways have more number of neighbors that are MARI nodes, this is in order to reduce the number of nodes in the virtual back-bone. This ensures that less number of Gateway nodes are required for the backbone.

• If any MARI node within two hops has already declared its Gateways, then there is no need to select Gateway for such a MARI node [122].

• MARI node determines the validity of its Gateways, i.e., the Gateway still has sufficient power level or not. Periodically a MARI node checks the power level of its Gateways, within each beacon period. If a MARI node finds that its Gateway has power level below a threshold, it will start a new Gateway selection using the above procedure [123].

Algorithm 4.4.1 Algorithm for Gateway node selection (Executed by the MARI nodes) Here $M_i$ is the MARI node which is executing this algorithm for selection of its Gateway node(s).
Transmit broadcast packet STAY-AWAKE

//Wait for one beacon period

for Each MARI node $M_i$ within two-hops do

    if $M_i$ has not decided Gateway for this MARI node then
        if $M_i$ is not neighbor of existing Gateways then
            $M_i$ Gateway = member which has maximum
            POWER among one-hop neighbors of $M_i$
        else
            $M_i$ Gateway node = existing Gateway node
        end if
    end if
end for

$M_i$ Gateway = Gateway of $M_i$ for this node
end if
end for

**Algorithm 4.4.1:** Algorithm for Gateway node selection (executed by MARI nodes periodically)
4.5. SCHEDULING OF SLEEP CYCLE

MARI nodes and Gateways continuously stay awake to forward packets of all nodes. Member nodes wake up at specified time-epochs, for specified amounts of brief time periods ($r$), within a beacon period $T$ (see figure 4.5.2), and if they do not have to transmit or receive data, they go to sleep again. There are a number of sleep cycle periods ($T_1, T_2$), ($T_2, T_3$)... ($T_n, T$) in a beacon period, covering the entire duration, also these are adjacent and non-overlapping. Each sleep cycle is further divided into $M$ number of equal slots, one for each member node. Each member node wakes up once in a sleep cycle, in its slot, at pre-defined time epoch. All nodes stay awake during period $(0, T_1)$, which is called broadcast window.
During this broadcast window, which is at the beginning of the beacon period, WAKEUP packets are exchanged, and that is done once or twice as required. Each node synchronizes its clock by using the time stamp in the WAKEUP packet from its MAR1 node. In our simulation on MATLAB, we have chosen that each member node determines its wake up time from its node id and a wakeup counter $w_i$, as explained below (see algorithm 4.5.1).

- During the initial period $(0, T_i)$ of the beacon period $(0, T)$, all nodes remain awake, transmit WAKEUP packet to all one-hop neighbors as broadcast messages, if any, and so as to keep every node updated about its one-hop neighborhood information. We call this period, $(0, T_i)$, the broadcast window. Thus, when a packet other than WAKEUP packet or STATUS packet, comes (at MAC layer) for transmission during broadcast window, the packet cannot be sent immediately, generally they may be data packets. The packet has to be buffered at the MAC layer and it will be transmitted after the end of the broadcast window. The WAKEUP packets during the broadcast window, from any node, are one or two.

- At the end of the broadcast window, i.e., at time $T_i$, all member nodes go to sleep mode, if they do not have any packets for transmission. A member node with id ‘i’ wakes up at time epoch $t_{im}$ in $m^{th}$ sleep cycle, that is during $(T_m, T_{m+1})$. It is awake for a period of $\tau$ or more depending on the necessity for transmission or receiving packets. In our simulations on MATLAB, node ‘i’ calculates its wake up time epoch $t_{im}$ from a pseudo-random number with
its node id 'i' and a wakeup counter \( w_i \) as seed to the pseudo-random number generator. Thus, the wake up time epoch of a node with id 'i' in \( m^{th} \) sleep cycle is given by \( t_{im} \):

\[
t_{im} = T_m + (T_{m+1} - T_m) \times \text{Rand}(i', w_i)
\]

where, \( \text{Rand}(i', w_i) \) is pseudo-random number in \((0,1)\) with \( i', w_i \) as seed.

- This wake-up time epoch \( t_{im} \) of node 'i', is also known to its MARI node and to its one-hop neighbors since each node knows its one-hop neighbor's id and its wakeup counter \( w_i \) from the WAKEUP packet received. All the nodes in adhoc network have identical pseudo-random number generators. So when the MARI node of node 'i' wants to send packets to node 'i', it will send at time \( t_{im} \). After the node 'i' receives packets, it goes to sleep again. So, packets intended for node 'i' from a neighbor node 'k' have to be buffered at the MAC layer at the MARI node of node 'i', until the time epoch \( t_{im} \). After a small time duration \( \tau \) from the epoch \( t_{im} \), if no packets are sent to node 'i', it goes to sleep mode again.

- If node 'i' wants to send packets to its MARI node, it will do so whenever the channel is free, otherwise it uses the back-off algorithm to sense the channel. After sending packets, the node enters the 'sleep mode' and wakes up at time epoch \( t_{(m+1)} \) or \( T \), whichever comes earlier.
Initial state

WAKEUP packet at appropriate time in (0, T₁)

For Each sleep cycle period (Tᵣ, Tᵣ₊₁)

Packets to transmit = n

Packet received at time tᵣ in (tᵣ₋₁, tᵣ + τ)

Yes

Wake up mode until tᵣ + τ

Go to Sleep mode

If tᵣ < tᵣ₋₁

Yes

Buffer the packets in the MARI node until tᵣ₋₁

No

Buffer the packets in the MARI node until tᵣ₋₁ + τ

Figure 4.5.1. Flow chart for sleep cycle scheduling
Algorithm 4.5.1 Algorithm for Sleep cycle scheduling (executed by all member
nodes for each beacon period).

Transmit WAKEUP packet at appropriate time in \((0, T_1)\)

for Each sleep cycle period \((T_m, T_{m+1}), m = 1 \ldots n\) do

wake-up at the pre-specified wake-up time epoch \(t_m\)

if there are no packets to transmit at \(t_m\) then

Go to sleep mode at time epoch \(t_m + \tau\) until time \(t_{(m+1)}\) which is the
next wake-up time epoch

if Packets received at any time \((t_r)\) in \((t_m, t_m + \tau)\) then

Remain in wake-up mode until \(t_m + \tau\) or longer depending
on the time required for packet transmission or receiving

end if

if \(t_r < t_m\) then

Buffer the packets its corresponding MARI node until time \(t_m\)

else

if \(t_r > t_m + \tau\)

Buffer the packets its corresponding MARI node until
time \(t_{(m+1)}\)

end if

else

Transmit the packet(s) at appropriate time(s)

Remain in wake-up mode until time \(t_m + \tau\) for
transmitting packets

end if

end for
If the node $k$ wants to send packets to its neighbor $i$ other than its MARI node, node $k$ wakes up at time $t_{km}$ (just as node $i$ wakes at time $t_{im}$), and sends packets to node $i$, through the MARI node of node $k$, intermediate MARI/Gateway nodes, MARI node of node $i$.

After each sleep cycle $(T_m, T_{m+1})$ in a beacon period, wakeup counter $w_i$ of node $i$ is updated using the amount of its wake-up time and packets sent or received during that time slot. This is how we have implemented the MARI topology network in our simulations on MATLAB. However, in actual use, one can determine the slot-assignment to member nodes, the times epochs $t_m$'s, the wake-up times $r$, by many heuristic ways with the aim of achieving safe, secure, fast and energy-efficient communication.

In large adhoc wireless networks, traffic passing through the backbone nodes, i.e., MARI nodes and Gateway nodes, is expected to be large when compared to local traffic among one-hop neighbors. MARI nodes and Gateway nodes do not have to wait for long periods of time for sending packets since they are always awake and have high channel-availability. Thus, the overall delay for routing packets can be reduced, by the use of this virtual backbone and MARI topology. On the other-hand, if the packets are to be transmitted to the neighbors directly, sender has to wait until the wake up time of receiver, which causes not only scheduling and operational difficulties, but also large delays.
On an average each packet suffers delay slightly more than \((T_{m+1} - T_m)/2\) at each hop. To reduce this delay, the number of sleep cycles can be increased. This will reduce the delay at last hop, as sender has to wait for less amount of time to deliver the packet to receiver (through their MARI nodes and concerned Gateway nodes) at each hop.

<table>
<thead>
<tr>
<th>Broadcast</th>
<th>Window</th>
<th>(kT)</th>
<th>(T_1)</th>
<th>(T_2)</th>
<th>(\ldots)</th>
<th>(T_m)</th>
<th>(\ldots)</th>
<th>(T_i)</th>
<th>((k+1)T)</th>
</tr>
</thead>
</table>

**Fig: 4.5.2a**

\[
\begin{align*}
    t_{1m} & \quad t_{1m} + \tau \\
    t_{lm} & \quad t_{lm} + \tau
\end{align*}
\]

\[
T_m \quad 1 \quad 2 \quad \ldots \quad i \quad \ldots \quad N-1 \quad T_{m+1}
\]

**Fig: 4.5.2b**

**Wake up time for node**

<table>
<thead>
<tr>
<th>Broadcast window</th>
<th>(kT)</th>
<th>(T_1)</th>
<th>(t_{i_1})</th>
<th>(T_2)</th>
<th>(t_{i_2})</th>
<th>(T_3)</th>
<th>(t_{i_3})</th>
<th>(T_4)</th>
<th>(t_{i_4})</th>
<th>((k+1)T)</th>
</tr>
</thead>
</table>

**Fig 4.5.2c.** Time epochs in a beacon period
4.6. ROUTING OVER VIRTUAL BACKBONE

To measure the effectiveness of the MARI Topology management scheme, we have designed a mobile agent based routing protocol. This routing protocol is executed only on MARI nodes. The Gateway nodes forward the packets by using NEXT-HOP-DESTINATION fields. When a MARI node has to send a packet to another MARI node, it sends the packet to the appropriate Gateway with address of the other MARI node in the NEXT-HOP-DESTINATION field. Thus, in this routing the Gateway node acts as a relay between two MARI nodes. The algorithm 4.6.1 explains the procedure for routing over the virtual backbone.

The routing algorithm is:

1. A member node \( S \), which needs to route to destination member node \( D \), sends a request to its MARI node \( M \), for routing.

2. The MARI node \( M \), checks from its routing table, if the destination \( D \) is one of its member nodes or not. If the destination is its member node, then there is no need to forward the packets to find the route. If not, then it sends sample packets for path finding (these are forwarding packets) to the destination MARI node \( M_D \), that is the MARI node of \( D \). This is done through many intermediate MARI nodes and Gateways. That is, \( M \), forwards the packets through several MARI nodes and Gateway nodes, through the backbone, until the destination MARI node \( M_D \) is identified. This is called forwarded routing.
(3) If the destination MARI node $M_n$ receives the forwarding packets and accepts
the route request, it acknowledges to the MARI node $M_x$, by a 'backward
packet'. This way, many number of MARI nodes, connected with the
destination MARI node, pass the routing information through the 'backward
packets' to the source MARI node.

(4) In the backward routing, the routing tables of the each MARI node along the
path are modified based on the information in the packets (backward packets).
When all the backward packets reach the source MARI node $M_x$, through the
different paths, The MARI node $M_x$ waits for time period $T_w$ for packets
coming from various paths. After time period $T_w$, MARI node $M_x$ selects the
best path among the received paths, based on shortest distance or any other
appropriate metric.

(5) It updates the routing table and informs node $S$ about path establishment.
Now node $S$ can send packets over the established path.

(6) The above procedure, thus, updates the routing tables of MARI nodes in the
network. This is done continuously, achieving efficiency in communication.
Algorithm 4.6.1: Routing over Virtual Backbone (executed by all MARI nodes)

// Assume S is the source node and D is the destination node, and $M_s$ and $M_D$ are the corresponding MARI nodes respectively.

if any route request packet is received at source MARI node $M_s$

    if destination node $D$ is neighbor of the source MARI node then

        inform node $D$ about route request and no need of forwarding packets for route request.

    else

        send the forward packets in all direction for the route request
        wait for acknowledgment, until wait time $T_w$

        if acknowledgment received then

            choose the shortest path among the acknowledgements received through backward packets from many number of paths.

        end if

    end if

end if

4.7. CONCLUSIONS

The MARI Topology is novel and is intended to bring much higher performance and QoS (Quality of Service) to Adhoc Wireless Networks, compared to the flat topologies that are presently in use. This chapter introduces the fundamental concepts and definitions related to MARI Topology networks. The nodes are categorized as, MARI nodes, Gateway nodes and member nodes. The first two node types are 'always awake' and jointly form the virtual backbone.
The member nodes stay in ‘sleep mode’ as much time as possible in order to reduce the power consumption. Member-to-member communication is done through intermediate MARI and Gateway nodes.

In this framework a number of procedures and algorithms for the formation of the virtual backbone, are designed and explained. This formation is supposed to take place wholly within the network. Some of these algorithms are,

(i) MARI node placement algorithm,
(ii) Gateway node selection algorithm
(iii) Sleep cycle scheduling
(iv) Routing over virtual backbone

Now what is intended further is the Performance and QoS evaluation of MARI Topology networks and possible comparisons with those of the existing flat-topology networks. This is taken up in the next chapter.

The research contributions of this chapter have been published/accepted by the author in [122,123].