ISSN:0975-9646

Energy Efficient Network Reconfiguration Algorithm Based on Cooperation for Wireless Ad hoc Network

Sachin Kumar Soni[1], Gajendra Singh[2]

[1]*M.Tech Scholor SSSIST Sehore*
[2]*HOD CE Department SSSIST Sehore*

Abstract— In a wireless multi hop ad hoc network, communication between distant nodes is done via intermediate hops working as forwarders. As forwarding the data packets consumes energy, intermediate hops may not wish to cooperate in communication. In this paper we are dealing with two major issues of wireless ad hoc network namely, cooperation and energy efficiency. We propose an algorithm that increases the level of cooperation of a wireless ad hoc network in an energy efficient manner, working over topology control in network along with load balancing and sleep scheduling

Keywords— **Ad hoc networks, cooperation, energy efficiency, load balancing, sleep scheduling, topology control.**

I. INTRODUCTION

A Wireless ad hoc network is an infrastructure less, decentralized network. It has no central object to govern the functioning of the network. In this network, each node also known as "hop" may act as router, forwarder, source and/or destination for a data packet. As compared to wired networks, ad hoc networks are more prone to malicious attacks and different kinds of failures. In order to manage the functionalities of an ad hoc network the nodes in it must be cooperative. Non-cooperative behavior of nodes affect throughput of the network.

In ad hoc network nodes communicate with each other by relaying data packets for other nodes when there is no direct link in between. To carry out this communication successfully the nodes must show their cooperation in relaying the packets. However, as nodes in ad hoc network have limited energy associated with them, they may choose not to forward any or all the packets for other nodes. But this misbehavior will consequently reduce the overall network throughput. Detection and penalization of misbehaving nodes is done by a protocol named CONFIDANT [3]. It detect malicious nodes by means of observation or reports about several types of attacks, thus allowing nodes to route around misbehaving nodes and to isolate them [3].

A wireless ad hoc network has finite energy, so to save energy and to increase the lifetime nodes may choose not to cooperate in relaying packet. This emphatically would bring down the throughput percentage. To manage the network in such a manner that the resources available, get properly utilized and nodes in the network do not drain out, energy efficient algorithms are needed. Along with it, sleep and wake-up strategy contributes to a level in saving a node's energy.

In this paper, we are trying to improve the level of cooperation, considering local topology control and load balancing, in an energy efficient manner with sleep scheduling working in parallel.

The remainder of this paper is organized as follows. In Section II, we present related works on cooperation in ad hoc networks, topology control, load balancing and sleep scheduling. Section III includes formulated problem. Section IV includes proposed solution of the listed problem. In Section V, we will discuss the work to be simulated.

II. RELATED WORK

The idea of this paper is all-embracing and do not particularly fall into one category of wireless ad hoc networks. It covers several issues such as cooperation, energy efficiency, load balancing, topology control and sleep scheduling.

The idea of the project starts with cooperation in network and its measurement.

A. Cooperation and its measurement

Cooperative communication is a way to improve the performance of relay transmission system working over ever-challenging wireless medium. In this domain, the work did include strategies which mainly follow tit-for-tat in order to maintain cooperation among nodes. [1] Nodes are assumed to be rational, i.e., their actions are strictly determined by self interest, and that each node is associated with a minimum lifetime constraint. The optimal throughput that each node should receive is defined to be the rational Pareto optimal operating point. Using which a distributed and scalable acceptance algorithm called Generous TIT-FOR-TAT (GTFT) has been proposed [2]. A market-based approach has also been worked upon in which nodes charge a price for relaying data packets and then use these incentives to relay their own packets. In both these approaches, the nodes which are detected as non-cooperating are punished either by isolating them from rest of the network or their packets are not forwarded by rest of the nodes.

B. Energy Efficiency

Energy is an important asset in ad hoc networks, as the working of whole network relies on the availability of energy. Energy efficiency can be achieved in many ways. The ways include [4] distributed algorithm where each node makes local decisions about its transmission power and these local decisions collectively guarantee global connectivity. Specifically, based on the directional information, a node grows it transmission power until it finds a neighbour node in every direction. The resulting network topology increases network lifetime by reducing transmission power and reduces traffic interference by having low node degrees.

C. Load Balancing

As Wireless ad hoc networks is infrastructure less, nodes working on their own in coordination with other nodes which makes them consume energy for being on the route of different Path in the network. Some of the nodes may act as forwarder or router in multiple numbers of paths while some may idle or do the minimal. These conditions create a huge difference between the residual powers of the nodes. They require a procedure where it is made sure that there will not be any particular set of nodes that only participate in communication but the network as a whole must participate. This comes under load balancing and different load balancing algorithms in theory choose different kind of parameters that are measured and balanced.

By taking network node taking into account three factors, residual battery capacity, the number of hops on the path and the length of the interface queue which gives an estimate of load on a node, that is the average number of packets, queued up at the interface queue to be delivered to the destination. By estimating these values and averaging those over time give an approximate look of the traffic in the networks and energy left with nodes. These estimates are added in a manner such that the main contributing factor becomes hop count and average queue length when energy of the nodes is high and residual energy become the parameter of choice when energy gets depleted. The defined parameters are:

1. Route Energy (RE): The route energy of a path is the minimum of residual energy of nodes (r_{ei}) falling on a route. Higher the route energy, lesser is the probability of route failure due to exhausted nodes.
2. Traffic queue (t_q): The traffic queue of a node is the number of packets queued up in the node's interface. Higher is its value, more occupied the node is.
3. Average Traffic Queue (ATQ): It is the mean of traffic queue of nodes from the source node to the destination node. It indicates load on a route and helps in determining the heavily loaded route.
4. Hop count (HC): The HC is the number of hops for a feasible path

D. Topology Control

Topology control is defined as a process where the topology of a network can be controlled by selective addition of nodes and links within a network. Due to varied usage of wireless ad hoc network in number of applications like battlefield, environmental modeling, emergency relief, topology control in ad hoc networks becomes a primary challenge where nodes can change their transmitting powers and can select different nodes for forwarding data. Topology control is defined as a process where the topology of a network can be controlled by selective addition of nodes and links within a network

The basic goal of Topology Control is

- Maintain connectivity using the minimum transmission power.
- Maintain connectivity by moving some "router" nodes to fill in the hole.
- Enable nodes to self-organize themselves into clusters.
- Load balance with power consideration.

E. Sleep Scheduling

Research results have shown that radio energy consumption of listening or idling in ad hoc networks can be significant. Hence by turning off the transceivers of idle devices or making nodes sleep, redundant energy can be saved more efficiently. Node scheduling is an efficient power saving method but it is required to be in coordination with the routing algorithm. It should allow as many nodes as possible to turn their radio receiver off most of the time, since even an idle circuit can consume almost as much energy as an active transmitter. Also, it should forward packets between any source and destination with minimally more delay than if all the nodes were awake.

III. PROBLEM FORMULATION

A. Assumptions

Starting with a static ad hoc network with known topology, we assume that all the nodes are working as in Unit Disk Graph (UDG) i.e., with same initial energy. Nodes which are not busy being a forwarder or a source/destination will move to sleep mode from the active mode according to a sleep scheduling algorithm.

B. Problem Definition

For a static ad hoc network with known topology, the goal is to devise an algorithm that increases the cooperation level among the static nodes in an energy efficient manner.

C. Problem Description

At any instant of time, the number of nodes and the network topology is known. To increase the cooperation level, one of the factors which we have to consider is that the nodes in the network do not drain out too soon. For that, an energy efficient algorithm is to be applied and some techniques must be used which may save energy of a node when it is in idle condition.

Initially, we are assuming the network with some number of nodes in sleep mode while other nodes in active mode and the network is in running state in which routing has already been done.

We propose a technique which can increase the cooperation level in an energy efficient manner. Along with it sleep scheduling is running in parallel so that overall energy efficiency is increased. We have tried to resolve above stated upshots while deliberating issues like load balancing and topology control.

IV. PROPOSED SOLUTION

A. Assumptions

The proposed protocol will run between two successive routing updates. And the messages in the network should get delivered in real time i.e., there should be no delay.

B. Protocol

Consider a static ad hoc network with known topology having "N" number of nodes, where in routing has already been performed. In the network, out of N nodes, some nodes are in sleeping state (say N_s) while the rest are active (say N_a) (Figure 1). So,

$N = N_s + N_a$ …………… (i)
where,
N = Total number of nodes
N_s = Number of nodes in sleeping state N_a = Number of nodes in active state

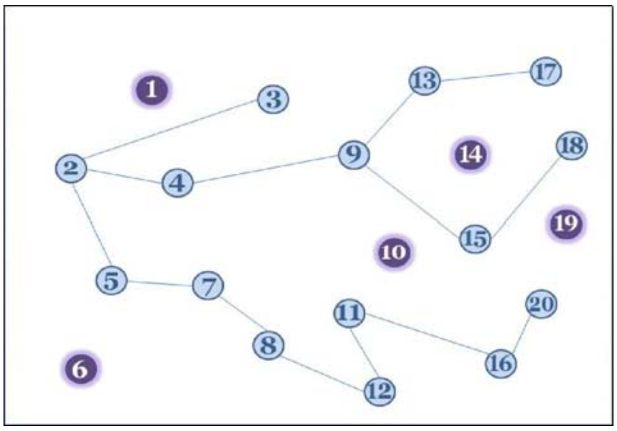


Figure 1: Initial Topology of the Network
(An example network with 20 nodes, where, 5 nodes numbered 1, 6, 10, 14 & 19 are in sleep mode while rests are awake.)

The protocol requires value of the cooperation to be precompiled at every of the node. For that, we will be using queues. Each node in the network is working with two queues attached with them, which are:
1) ***Input queue, Q_i*** - used to store incoming packets, and
2) ***Output queue, Q_o*** - used to store the packets which are to be transmitted by the node.

Cooperation of a node can be calculated using two queues associated with the nodes. Cooperation can be defined as number of packets forwarded by a node.

$$Qi = Pf\,i + Pr$$
$$Qo = Pf\,o + Po$$
$$Pf\,o = Qo - Po$$

Where
$Pf\,o$ = No. of Packets forwarded
Po = No. of self generated packets sent
$Pf\,i$ = Packets to be forwarded
Pr = No of packets of received packets
Pf o indicates the level of cooperation.
The protocol can be divided into five modules as following:

Module 1: Two-hop Communication
A hop is a component part of a signals journey from source to destination. When a data packet has to be transmitted, it often has to go after crossing many routers before it reaches final destination. Each time a packet is passed to next router hop count increases by 1. An ad hoc network uses two or more wireless hops to convey information from a source to destination. In particular, when only two hops are used, the communication is known as 2-hop communication.

Module 2: Cluster Formation
As the protocol starts each of the node distributes its level of cooperation in two-hop proximity. This is done through "2-Hop Prefatory Packet" which carries the node_ID along with calculated cooperation value and energy remaining with the node.

ID	Cooperation value	Remaining Energy	TTL

Figure 2: Two Hop Prefatory Packet

every node receiving this message maintains a table structure to have track of cooperation level of every of it's two hop neighbor. The node with highest level of cooperation becomes the cluster head. If the two nodes Are having same cooperation value then the cluster head will be elected on the basis of remaining energy. Each of the clusters contains all the nodes which are in two-hop connectivity with the cluster head. Also the clusters may be overlapping clusters.

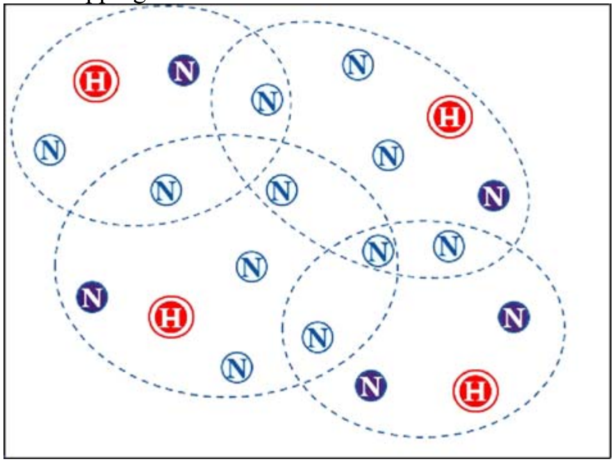


Figure 3: Cluster formation

Module 3: Inflicting Penalty
Each of the cluster head has some of the responsibilities associated with them. A cluster head has to:
1) Monitor the cooperation level of every node in its cluster,
2) Detect non-cooperative node and penalize them by forcefully sending them to sleep state for a certain period of time, and
3) Find alternate paths when nodes are penalized.

For this, every cluster head maintains a following table:

Node _ID	Cooperation Value	Status	Time to Sleep

Status field can have three values:
1. Node is in active State.
2. Node is forcibly sent to sleep state.

3. Node voluntary sent to sleep state.

It compares the value of cooperation of each node with every other and finds a node with lowest cooperation value. It then sends a *"Penalty Message"* to that node which includes TTS i.e. *time to sleep.*

ID	TTL	TTS

Figure 4: Penalty Message

***Module 4**: Identification of New Route*

Whenever a node with lowest cooperation level is punished and is forcibly sent to sleep state, cluster head manages the topology of the network. Suppose if there is any path from source to destination which had the punished node, cluster head has to find an alternate path. This may infix two conditions:

1) There may be one efficient path formed by active nodes. If it is so, then use it.
2) There are more than one alternate paths. This may further introduce three situations:
   a) One of them is efficient out of all the paths. If so, then use it.
   b) None of them is efficient. If it is so, then use load balancing mechanism to create an efficient path.
   c) If no node is in the proximity and even load balancing could not find an efficient path, then in this case routing algorithm is triggered by the cluster head.

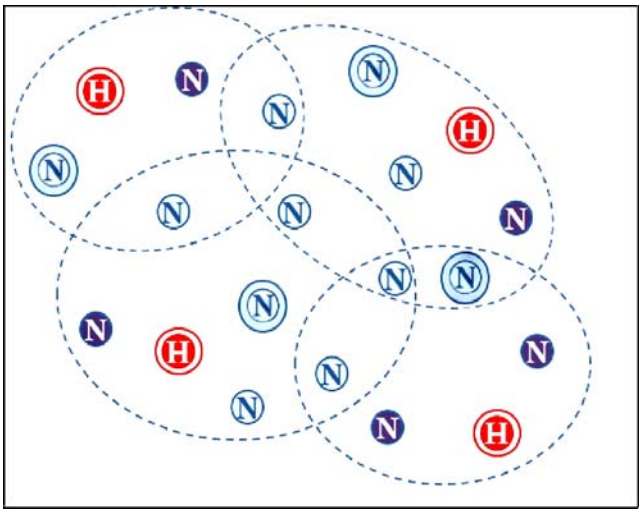


Figure 5: Punished Nodes (Double concentric circled N are punished nodes-with low cooperation value)

***Module 5:** Sleep Scheduling*

This module can be divided into two cases.

Case I: When node voluntarily goes to sleep state-

In ad hoc network it may happen that a node remains idle for a certain period of time. In that case, it voluntarily goes to sleep state to save energy. The fixed time interval after which node goes to sleep state is referred as *"Idle Time"*. Whenever a node sleeps it sends an "Alert Message" to its neighbors that include time for which node would be sleeping. This sleeping time is referred to as *"Sleep Interval"*.

ID	TTL	Sleep Interval

Figure 6: Sleep Alert

When in sleep state, a node periodically checks its *sleep buffer* which stores the packets received when the node is asleep. Sleep buffer is actually input queue but in case of sleeping nodes, we are referring it as *"sleep buffer"*. This periodic interval is named as *"sleep buffer checking interval"*.

Requesting to cooperate:

It may happen that during topology control and load balancing i.e., while identifying a new route any sleeping node may be required to act as intermediate hop. For that, cluster head sends request to the required hop. The request gets stored in the sleep buffer of the node.

But as adverted before that the clusters are overlapping, a sleeping node may receive requests from multiple cluster heads. This problem is resolved on *first come first serve* basis.

ID	TTL	Data

Figure 7: Requesting message (In this message the data part contains the requesting message)

Case II: When node is forcibly sent to sleep state-

The node with lowest cooperation value is punished and is forcibly sent to sleep state

When a punished node completes its punishment it sends an introductory packet to every node informing its availability.

ID	TTL	Data

Figure 8: Introductory packet (In this message the data part contains a message indicating availability of the node)

V. Simulation

The protocol designed is simulated in NS2 with the above drawn topology and tested the requirements of being called an energy efficient algorithm. The nodes will increase gradually as the network goes bigger.

Initial Topology-

NAM is a TCL based animation tool for viewing network simulation traces and real world packet traces. It is mainly intended as a companion animator to the ns simulator. NAM provides a visual interpretation of the network topology created. In our protocol, we are assuming an initial network topology with 25 nodes. Following is the scenario for our protocol which contains 25 static nodes:

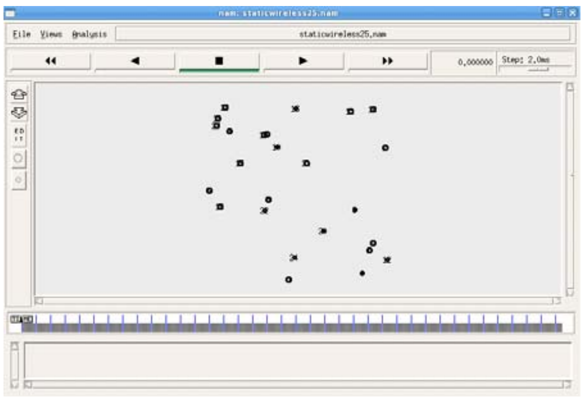


Figure 9: Implementation Scenario

Sensing-
Every node in ad hoc network transfers data packets via its neighbors. For discovering the neighbors, node has to sense the environment for relaying the packets. In the following figure it is shown that how a node senses its environment for discovering neighbors.

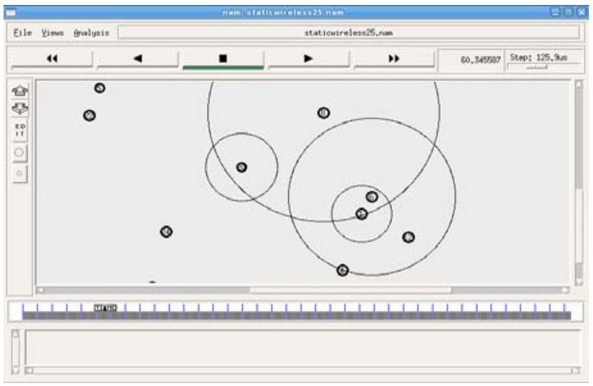


Figure 10: Sensing

Packet Transmission-
The following figure depicts how packet transmission occurs in the network. In the below exhibited figure node 15 is transmitting packet to node 7 and node 17 is transmitting a packet to node 9.

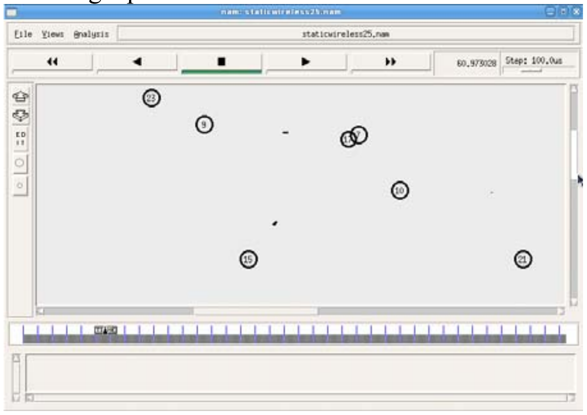
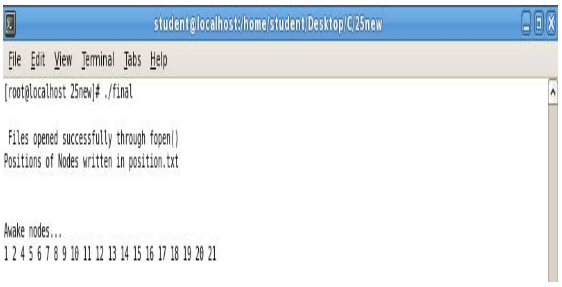


Figure 11: Packet Transmission

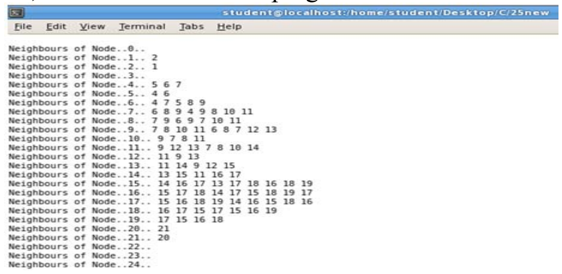
Simulation Result
Initially, in the network some of the nodes are in sleep state while others are active. Following snapshot lists the awake nodes out of 25 nodes.



```
[root@localhost 25new]# ./final


 Files opened successfully through fopen()
Positions of Nodes written in position.txt



Awake nodes...
1 2 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21
```

Figure 12: List of Awake Nodes

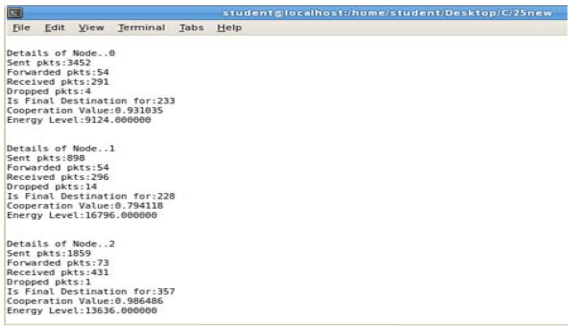
Before running the EECR protocol, list of neighbors within the two-hop vicinity is extracted from Traffic Generation file, stated above. Here, modified list of neighbors is shown, that excludes the sleeping nodes.



```
Neighbours of Node..0..
Neighbours of Node..1.. 2
Neighbours of Node..2.. 1
Neighbours of Node..3..
Neighbours of Node..4.. 5 6 7
Neighbours of Node..5.. 4 6
Neighbours of Node..6.. 4 7 5 8 9
Neighbours of Node..7.. 6 8 9 4 9 8 10 11
Neighbours of Node..8.. 7 9 6 9 7 10 11
Neighbours of Node..9.. 7 8 10 11 6 8 7 12 13
Neighbours of Node..10.. 9 7 8 11
Neighbours of Node..11.. 9 12 13 7 8 10 14
Neighbours of Node..12.. 11 9 13
Neighbours of Node..13.. 11 14 9 12 15
Neighbours of Node..14.. 13 15 11 16 17
Neighbours of Node..15.. 14 16 17 13 17 18 16 18 19
Neighbours of Node..16.. 15 17 18 14 17 15 18 19 17
Neighbours of Node..17.. 15 16 18 19 14 16 15 18 16
Neighbours of Node..18.. 16 17 15 17 15 16 19
Neighbours of Node..19.. 17 15 16 18
Neighbours of Node..20.. 21
Neighbours of Node..21.. 20
Neighbours of Node..22..
Neighbours of Node..23..
Neighbours of Node..24..
```

Figure 13: List of Neighbours

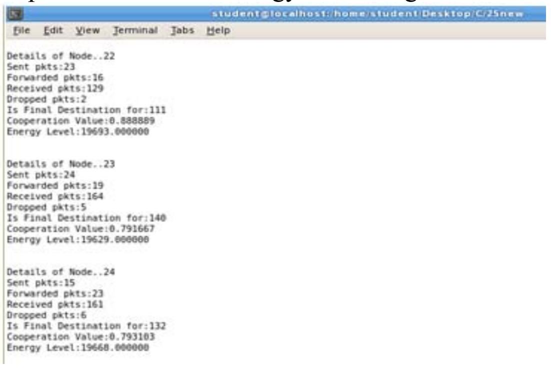
After finding active nodes, the detail of packet transmitted, received, forwarded and dropped is fetched from the trace file. Based on which cooperation level and remaining energy of each of the node is calculated separately.



```
Details of Node..0
Sent pkts:3452
Forwarded pkts:54
Received pkts:291
Dropped pkts:4
Is Final Destination for:233
Cooperation Value:0.931035
Energy Level:9124.000000


Details of Node..1
Sent pkts:898
Forwarded pkts:54
Received pkts:296
Dropped pkts:14
Is Final Destination for:228
Cooperation Value:0.794118
Energy Level:16796.000000


Details of Node..2
Sent pkts:1859
Forwarded pkts:73
Received pkts:431
Dropped pkts:1
Is Final Destination for:357
Cooperation Value:0.986486
Energy Level:13636.000000
```

Figure 14: Details of Node Traffic

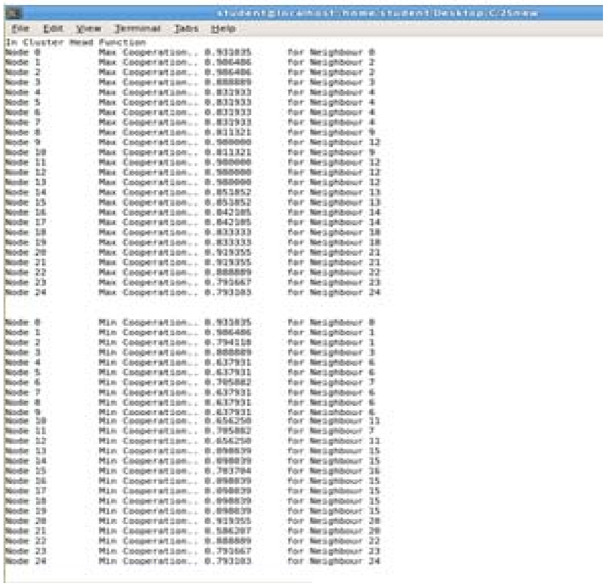
Similarly, for all the nodes value of cooperation and remaining energy is calculated. And then average of both, the cooperation level and energy level is figured.



```
Details of Node..22
Sent pkts:23
Forwarded pkts:16
Received pkts:129
Dropped pkts:2
Is Final Destination for:111
Cooperation Value:0.888889
Energy Level:19693.000000


Details of Node..23
Sent pkts:24
Forwarded pkts:19
Received pkts:164
Dropped pkts:5
Is Final Destination for:140
Cooperation Value:0.791667
Energy Level:19629.000000


Details of Node..24
Sent pkts:15
Forwarded pkts:23
Received pkts:161
Dropped pkts:6
Is Final Destination for:132
Cooperation Value:0.793103
Energy Level:19668.000000
```

Figure 15: Overall Cooperation and Remaining Energy of the Network

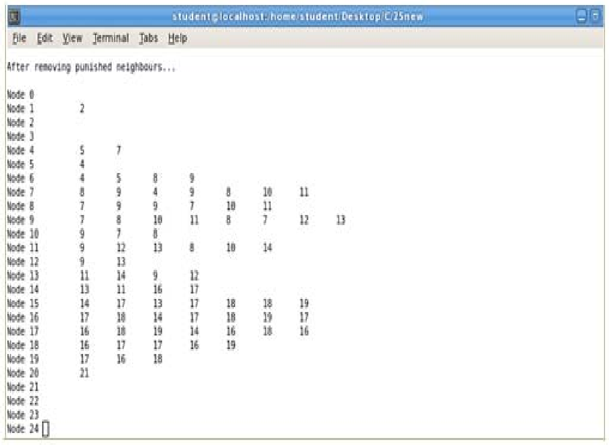
Now based on the level of cooperation, the cluster-heads are chosen and clusters are formed. Also, examining the value of cooperation can yield the nodes to be punished.



```
In Cluster Head Function
Node 0          Max Cooperation.. 0.933835      for Neighbour 0
Node 1          Max Cooperation.. 0.986486      for Neighbour 2
Node 2          Max Cooperation.. 0.986486      for Neighbour 2
Node 3          Max Cooperation.. 0.888889      for Neighbour 3
Node 4          Max Cooperation.. 0.833333      for Neighbour 4
Node 5          Max Cooperation.. 0.833333      for Neighbour 4
Node 6          Max Cooperation.. 0.833333      for Neighbour 4
Node 7          Max Cooperation.. 0.833333      for Neighbour 4
Node 8          Max Cooperation.. 0.811321      for Neighbour 9
Node 9          Max Cooperation.. 0.900000      for Neighbour 12
Node 10         Max Cooperation.. 0.811321      for Neighbour 9
Node 11         Max Cooperation.. 0.900000      for Neighbour 12
Node 12         Max Cooperation.. 0.900000      for Neighbour 12
Node 13         Max Cooperation.. 0.900000      for Neighbour 12
Node 14         Max Cooperation.. 0.851852      for Neighbour 13
Node 15         Max Cooperation.. 0.851852      for Neighbour 13
Node 16         Max Cooperation.. 0.842105      for Neighbour 14
Node 17         Max Cooperation.. 0.842105      for Neighbour 14
Node 18         Max Cooperation.. 0.833333      for Neighbour 18
Node 19         Max Cooperation.. 0.833333      for Neighbour 18
Node 20         Max Cooperation.. 0.919355      for Neighbour 21
Node 21         Max Cooperation.. 0.919355      for Neighbour 21
Node 22         Max Cooperation.. 0.888889      for Neighbour 22
Node 23         Max Cooperation.. 0.791667      for Neighbour 23
Node 24         Max Cooperation.. 0.793103      for Neighbour 24

Node 0          Min Cooperation.. 0.933835      for Neighbour 0
Node 1          Min Cooperation.. 0.986486      for Neighbour 1
Node 2          Min Cooperation.. 0.794118      for Neighbour 1
Node 3          Min Cooperation.. 0.888889      for Neighbour 3
Node 4          Min Cooperation.. 0.637931      for Neighbour 6
Node 5          Min Cooperation.. 0.637931      for Neighbour 6
Node 6          Min Cooperation.. 0.705882      for Neighbour 7
Node 7          Min Cooperation.. 0.637931      for Neighbour 6
Node 8          Min Cooperation.. 0.637931      for Neighbour 6
Node 9          Min Cooperation.. 0.637931      for Neighbour 6
Node 10         Min Cooperation.. 0.656250      for Neighbour 11
Node 11         Min Cooperation.. 0.705882      for Neighbour 7
Node 12         Min Cooperation.. 0.656250      for Neighbour 11
Node 13         Min Cooperation.. 0.898039      for Neighbour 15
Node 14         Min Cooperation.. 0.898039      for Neighbour 15
Node 15         Min Cooperation.. 0.783784      for Neighbour 16
Node 16         Min Cooperation.. 0.898039      for Neighbour 15
Node 17         Min Cooperation.. 0.898039      for Neighbour 15
Node 18         Min Cooperation.. 0.898039      for Neighbour 15
Node 19         Min Cooperation.. 0.898039      for Neighbour 15
Node 20         Min Cooperation.. 0.919355      for Neighbour 20
Node 21         Min Cooperation.. 0.586207      for Neighbour 20
Node 22         Min Cooperation.. 0.888889      for Neighbour 22
Node 23         Min Cooperation.. 0.791667      for Neighbour 23
Node 24         Min Cooperation.. 0.793103      for Neighbour 24
```

Figure 16: Nodes with Minimum and Maximum Level of Cooperation

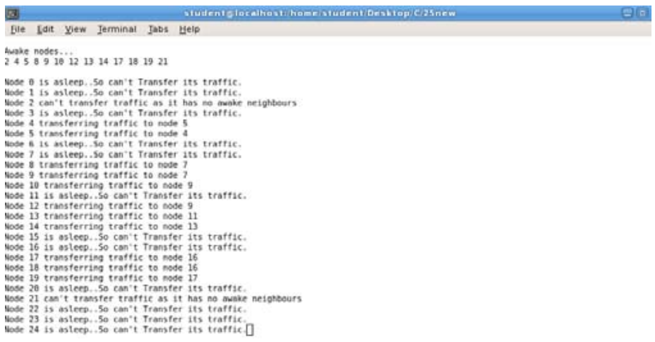
On penalizing the nodes, the resultant neighbor list is shown below.



```
After removing punished neighbours...

Node 0
Node 1      2
Node 2
Node 3
Node 4      5     7
Node 5      4
Node 6      4     5     8     9
Node 7      8     9     4     9     8     10    11
Node 8      7     9     9     7     10    11
Node 9      7     8     10    11    8     7     12    13
Node 10     9     7     8
Node 11     9     12    13    8     10    14
Node 12     9     13
Node 13     11    14    9     12
Node 14     13    11    16    17
Node 15     14    17    13    17    18    18    19
Node 16     17    18    14    17    18    19    17
Node 17     16    18    19    14    16    18    16
Node 18     16    17    17    16    19
Node 19     17    16    18
Node 20     21
Node 21
Node 22
Node 23
Node 24
```

Figure 17: New List of Neighbours

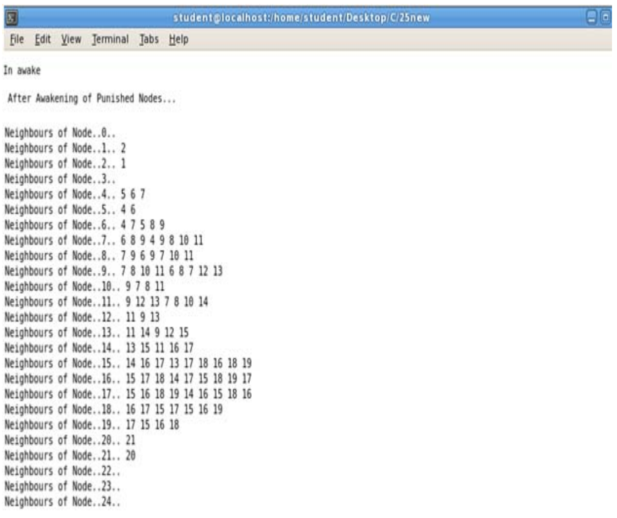
Penalization of nodes engendered broken links. So, to continue the traffic load balancing and topology control is to be applied.



```
Awake nodes...
2 4 5 8 9 10 12 13 14 17 18 19 21

Node 0 is asleep..So can't Transfer its traffic.
Node 1 is asleep..So can't Transfer its traffic.
Node 2 can't transfer traffic as it has no awake neighbours
Node 3 is asleep..So can't Transfer its traffic.
Node 4 transferring traffic to node 5
Node 5 transferring traffic to node 4
Node 6 is asleep..So can't Transfer its traffic.
Node 7 is asleep..So can't Transfer its traffic.
Node 8 transferring traffic to node 7
Node 9 transferring traffic to node 7
Node 10 transferring traffic to node 9
Node 11 is asleep..So can't Transfer its traffic.
Node 12 transferring traffic to node 9
Node 13 transferring traffic to node 11
Node 14 transferring traffic to node 13
Node 15 is asleep..So can't Transfer its traffic.
Node 16 is asleep..So can't Transfer its traffic.
Node 17 transferring traffic to node 16
Node 18 transferring traffic to node 16
Node 19 transferring traffic to node 17
Node 20 is asleep..So can't Transfer its traffic.
Node 21 can't transfer traffic as it has no awake neighbours
Node 22 is asleep..So can't Transfer its traffic.
Node 23 is asleep..So can't Transfer its traffic.
Node 24 is asleep..So can't Transfer its traffic.
```

Figure 18: Traffic Scenario

After performing load balancing, the punished nodes are released and hence they change their mode from inactive to active.



```
In awake

 After Awakening of Punished Nodes...

Neighbours of Node..0..
Neighbours of Node..1.. 2
Neighbours of Node..2.. 1
Neighbours of Node..3..
Neighbours of Node..4.. 5 6 7
Neighbours of Node..5.. 4 6
Neighbours of Node..6.. 4 7 5 8 9
Neighbours of Node..7.. 6 8 9 4 9 8 10 11
Neighbours of Node..8.. 7 9 6 9 7 10 11
Neighbours of Node..9.. 7 8 10 11 6 8 7 12 13
Neighbours of Node..10.. 9 7 8 11
Neighbours of Node..11.. 9 12 13 7 8 10 14
Neighbours of Node..12.. 11 9 13
Neighbours of Node..13.. 11 14 9 12 15
Neighbours of Node..14.. 13 15 11 16 17
Neighbours of Node..15.. 14 16 17 13 17 18 16 18 19
Neighbours of Node..16.. 15 17 18 14 17 15 18 19 17
Neighbours of Node..17.. 15 16 18 19 14 16 15 18 16
Neighbours of Node..18.. 16 17 15 17 15 16 19
Neighbours of Node..19.. 17 15 16 18
Neighbours of Node..20.. 21
Neighbours of Node..21.. 20
Neighbours of Node..22..
Neighbours of Node..23..
Neighbours of Node..24..
```

Figure 19: Traffic Scenario

Similarly, the above simulation is carried out with different number of nodes in the network.

Simulation Result Analysis

The EECR protocol is designed to increase the average cooperation level of a static wireless ad hoc network in an energy efficient manner. After simulating working of network without EECR and with EECR, following results came out.

Level of Cooperation

Number of nodes	Cooperation values without EECR	Cooperation values with EECR
5	0.200000	0.200000
10	0.669359	0.673333
15	0.669591	0.737056
20	0.763309	0.781117
25	0.772920	0.821487

Graphically,

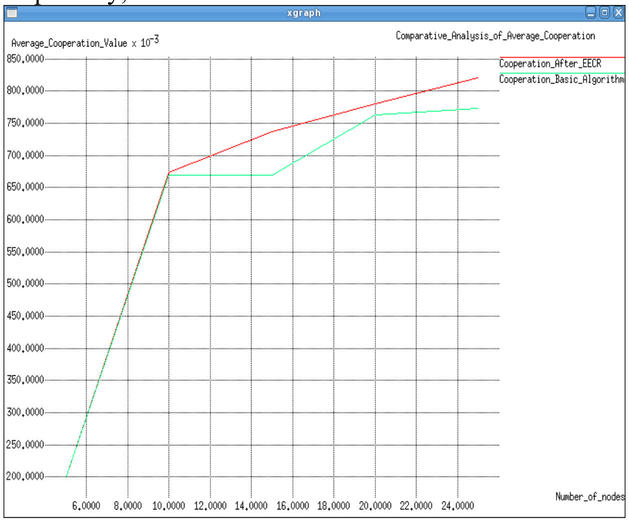


Figure 20: Comparative Analysis of Cooperation Level

Analysis-

By studying the above graph, we can conclude that

1. As the number of nodes increases, average cooperation level both, before and after running the EECR algorithm, shows a general increase.
2. After running the EECR algorithm, the overall cooperation level of the network showed an increase, as intended.
3. When number of nodes are 5, there is no change in overall cooperation level of the network.
4. When number of nodes are 15, a substantial change in overall cooperation level of the network is encountered.

Remaining Energy Level

Number of nodes	Energy values without EECR	Energy values with EECR
5	18387	18387
10	19542	19729
15	18902	19176
20	19140	19529
25	18507	19693

Graphically,

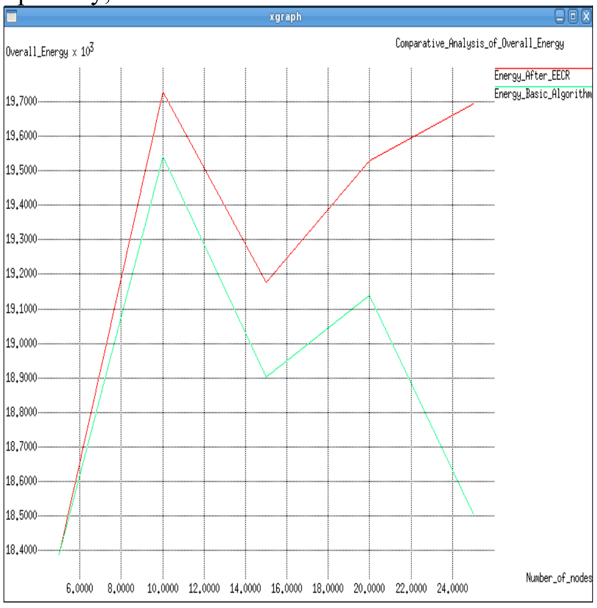


Figure 21: Comparative Analysis of Energy Level

Analysis-

By studying the above graph we can conclude that

1. As the number of nodes increases, average energy consumption for both, before and after running the EECR algorithm, show a fluctuation and no general trend is observed.
2. After running the EECR algorithm, the overall energy level of the network showed an increase, as intended.
3. Drastic change in remaining energy levels of nodes is observed when the numbers of nodes in network are 25.

VI. CONCLUSION

For an ad hoc network with 2-hop connectivity the design of this algorithm was aimed to increase the cooperation level of nodes while taking into consideration the energy constraint. It can be concluded from the simulation results that the EECR is resultant protocol that can increase the cooperation level in an energy efficient way. This is collectively accomplished with the help of many factors like topology control, load balancing and sleeps scheduling. in future, the devised protocol can be extended for mobile ad hoc network.

REFERENCES

[1] Vikram Srinivasan, "Cooperation In Wireless Ad Hoc Networks". Department of Electrical and Computer Engineering, University of California at San Diego..

[2] Peter Marbach, Member, IEEE, and Ying Qiu, "Cooperation in Wireless Ad Hoc Networks: A Market-Based Approach.

[3] Sonja Buchegger, "Performance Analysis of the CONFIDANT Protocol (Cooperation Of Nodes: Fairness In Dynamic Adhoc Networks)". IBM Zurich Research Laboratory Saumerstrasse, Switzerland.

[4] Roger Wattenhofer, "Distributed Topology Control for Power Efficient Operation in Multihop Wireless Ad Hoc Networks". Microsoft Research Redmond, WA 98052

[5] C. Siva Ram Murthy, B. S. Manoj, "Ad Hoc Wireless Networks, Architectures and Protocol". Prentice Hall Communication Engineering and Emerging Technologies series.

[6] E.L. Lloyd, R. Liu, M. V. Marathe, R. Ramanathan and S. S. Ravi, "Algorithmic Aspects of Topology Control Problems for Ad hoc Networks", MOBIHOC" 02, June 9-11, 2002, EPFL Lausanne, Switzerland, ACM, 2002.

[7] N. Li and J. Hou , "Design and Analysis of an MST-Based Topology Control Algorithm".

[8] Vamsi Paruchuri, Shivakumar Basavaraju, Arjan Durresi, Rajgopal Kannan and S.S. Iyengar, "Random Asynchronous Wakeup Protocol for Sensor Networks". Louisiana State University, Department of Computer Science