# Mitigating Selfishness to Improve Replica Allocation in MANET

Jim Solomon Raja.D[1], Immanuel John Raja.J[2]
Karunya University India

## Abstract

Mobile ad hoc networks are formed dynamically due to autonomous system of mobile nodes that are connected through wireless links without using an existing infrastructure or centralized administration. In such ad hoc network nodes have various limitations due to its ad hoc behavior. Hence resources like power, battery and computing ability are valuable in such type of ad hoc networks. Hence some mobile nodes decided not to cooperate with other mobile nodes and simply aim to save its resources to the maximum while using the network to forward its own packets, these types of mobile nodes are called "Selfish Nodes" this misleading is very common in ad hoc network because of its configuration setup. These nodes could be detected and excluded from the cooperative portion of the network, as they only consume resources but don't contribute to the infrastructure. In existing methods, there are no steps to handle false alarms and efficient detection of selfish nodes. In this paper, a new mechanism that minimizes the problem of selfish nodes with the help of Credit risk and Brain trapping function Model. Including Degree of selfishness in allocating replicas will considerably reduce communication cost and produce high data accessibility. A collaborative monitoring mechanism is also used to manage false alarms. Simulation results shows that the proposed system provides better detection efficiency, low false positive and delay constraint.

**Keywords— Access frequency,** Mobility, False alarm, Selfish nodes, Selfish replica allocation, Degree of selfishness, Credit risk.

## I. INTRODUCTION

Ad hoc networks consist of wireless mobile nodes that auto configure to form a network with no fixed infrastructure, they can be frequently set up as needed. Such mobile nodes cooperate in routing to allow each node to communicate within its wireless transmission range [2].In such type of mobile nodes depend on each other for routing and forwarding packets. To consume power and other resources, nodes depending to independent authorities may behave selfishly, and may not be willing to cooperate with other nodes[6]. A mobile node may be able to communicate with other nodes far away with the combination of intermediate nodes, transforming the packets to the destination. In this multi hop communication, each mobile node performs as both host and router. Routing protocols of ad hoc network such as DSR [3], AODV [4] have been designed to manage such environment. Fast deployment, inadequate configuration and network without any central governing authority make MANET suitable for emergency situations such as military conflicts, natural disasters and emergency medical situations [4]. Data items are usually replicated at mobile nodes, other than the original holders, to improve data accessibility to adapt with continuous network partitions. Selfish replica allocation is a another notation refers to a mobile node's non cooperative act, means the node refuses to participate fully in sharing its memory space with other mobile nodes in the network. It considers replica allocation techniques with the developed selfish node detection method. They are based on the concept of a self-centered friendship tree (SCF-tree) [1] and its aim is to achieve high data accessibility with low communication cost in the presence of malicious nodes. The SCF-tree is divine by human friendship handling in the world. In this paper, a model to detect and prevent selfish nodes that decline to cooperate but at the same time still utilize the network for their own benefits. Actually this model is also helped to find any misbehaving node attack in ad hoc network but in this paper the focus on replica allocation with the presence of selfish nodes. This paper is organized as follows. In section 2, it briefly explain the existing methods of the replica allocation. In section 3,It provide preliminaries and assumptions Section 4 gives overview of proposed model Section 5 illustrates Performance metrics. Section 6 concludes this paper.

## II. BACKGROUND

Three replica allocation methods were proposed belongs to the access frequency and network topology [1].

a. SAF (Static Access Frequency) [3] The mobile host stores the original data with the highest access frequency and replicas with the descending order of their access frequencies.

b. DAFN (Dynamic Access Frequency and Neighbourhood)
[2]The mobile host allocates replicas based on SAF. And the replica duplications among neighbours are eliminated as much
as possible.

c. DCG (Dynamic Connectivity based Grouping)[3] The mobile hosts are organized into groups using biconnected components.

By examining the frequencies of a specific data among the group members, store the data on the mobile nodes with the highest access frequency. Then the three methods are outstretched to adapt to an environment where each data item is updated periodically.

## III.PRELIMINARIES

### 1. System Model

To focus on the selfish replica allocation, it will not consider selfishness in data forwarding throughout this paper [12].

- Each node in a MANET has a unique identifier[1]. All nodes that are placed in a MANET are denoted by N= {N1,N2,…Nm}, where m is the total number of nodes.
- All data items are of equal size, and each data item is held by a particular node as its original node. Each data item has a unique identifier, and the set of all data items is denoted by D= {D1, D2,…Dn} where n is the total number of data items.
- Each node Ni has its own access frequency to data item. The access frequency does not change.
- Each node moves freely within the maximum velocity.

### 2. Node Behavior Model

- Type-1 node: The nodes are nonselfish nodes[1]. The nodes hold replicas allocated by other nodes within the limits of their memory space.
- Type-2 node: The nodes are fully selfish nodes[1]. The nodes do not hold replicas allocated by other nodes, but allocate replicas to other nodes for their accessibility.
- Type-3 node: The nodes are partially selfish nodes[1]. The nodes use their memory space partially for allocated replicas by other nodes. Their memory space may be divided logically into two parts: selfish and public area. These nodes allocate replicas to other nodes for their accessibility. The detection of the type-3 nodes is complex, because they are not always selfish.

## IV.PROPOSED STRATEGY

### 4.1 Overview

The strategy consists of three parts:

1) Detection and prevention of selfish nodes
2) SCF-tree with cooperative monitoring system
3) Allocating replica regarding Degree of selfishness

The ultimate aim of the SCF-tree-based replica allocation techniques is that it minimize the communication cost and achieving high data accessibility.

### 4.2 Detection and prevention of selfish nodes

It made plan on the real case that everyone want to live and exertion for its existence if anyone is sure that he will not going to die because of scarcity of resources then it will be more chances that it will not cheat others for resources. Here it consider the notion of credit risk (CR)[1] from bank management to detect selfish nodes.

Credit Risk (CR) = Expected risk/ Expected value (1)

each mobile node computes a CR score for each of the nodes to which it is connected. Each node shall calculate the "degree of selfishness" for all of its connected nodes according to the score.

TABLEI
PARAMETERS

| Parameters | Definition |
|---|---|
| Nk | The number of nodes shared their memory space |
| $SS_i^k$ | Memory space shared |
| $ND_i^k$ | The number of Nk's shared data items observed by a node Ni |
| Si | Size of memory space at node Ni |
| $P_i^k$ | Ratio of selfishness alarm of Nk on Ni |

The value of $CR_k^i$ is being affected if $CR_k^i$ is not normalized. By normalizing $nCR_k^i$ [1] stands for the normalized $CR_k^i$

$$nCR_i^k = \frac{p_i^k}{a * SS_i^k / S_i + (1-a) * ND_i^k / n_i}, \text{where } 0 \le a \le 1 \qquad [2]$$

Algorithm 1 describes how to detect selfish nodes. At each relocation period, node Ni detects selfish nodes based on nCRki and Fig1[3] shows that the exclusion of selfish nodes.

#### 4.2.1 Algorithm for detecting selfish nodes

Algorithm 1.Pseudo code to detect selfish nodes

```
00: At every relocation period
01: /*$N_i$ detects selfish nodes using this algorithm */
02: selfish node detection (){
03:     for (each connected mobile node $N_k$){
04:     if ($nCR_i^k < \delta$ )Nk is noted as non-selfish;
05:         else Nk is noted as selfish ;}
06:     wait till allocation of replica is done;
07:     for (each connected mobile node Nk){
08:         if (Ni has allocated replica to Nk){
09:     $ND_i^k$=the number of replica allocated;
10:     $SS_i^k$= the total size of replica allocated;}
11:         else {
12:         $ND_i^k$=1;
13:         $SS_i^k$= the size of shared data item
14:         }}}
```

#### 4.2.2 Algorithm for updating selfish features

Algorithm 2 .Pseudo code to update features of selfishness

```
00: At every query processing time
01: /* when Ni produces a query */
02: update_SF () {
03:     while (during the predefined time $\omega$ ) {
```

```
04:        if (an target node Nk serves the query)
05:        decrease $P_i^{k;}$
06:        if(an unexpected node Nj answer the request){
07:        $ND_i^j= ND_i^j+1$;
08:        $SS_i^j= SS_i^j$ +(the size of a data item);
09:        }}
10:        if( an expected node Nk does not answer the request){ increase $P_i^k$;
11:        $ND_i^k= ND_i^k-1$;
12:        $SS_i^k= SS_i^k$ - (the size of the data item) ;}}
```

As described in Algorithm 2, Ni maintains its $ND_i^k$ , $SS_i^k$ and $P_i^k$ during each query processing phase.

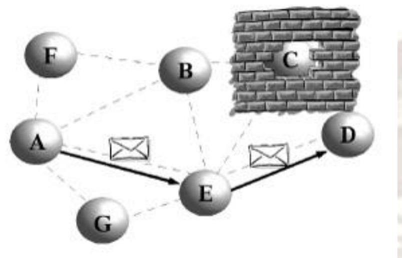

Fig. 1 Exclude selfish nodes from routing

### 4.3 The Brain Trapping Function Node (BTFN)

These nodes perform Brain Trapping functions for all nodes present in ad hoc network. It is used to improve detection of selfish nodes and minimize the effects of false alarms. The important parts of Brain Trapping Function nodes are

- *SDP module* :This Module has the ability of prevention of selfish node.
- *Turi machine* :It constitute of infinite memory capability to store virtual node.
- *Virtualization* Layer :This Layer is responsible for creating virtual node.

**SDP module working algorithm:**

```
Set mobile node = M //Total Mobile Nodes
Set source node = S //S Ɛ M
Set Destination Node = D // D Ɛ M
Set radio range = rr
```

#### 4.3.1. RREQ_B(S, D, rr)

```
00:{If ((rr<=300) && (next hop >0))
01:        {
02:        Compute path ()
03:        {rtable->insert(rtable->rt_nexthop);
04: if (dest==true)
05:        { send ack to S with rtable;
06:        Data_send(s_no, nexthop, type) }
07: else {
08:        destination not found;
09:        }}}
10: else {destination un-reachable ; }}
```

#### 4.3.2. Selfish_Node ()

```
00:{Check (incoming packet)
01:{        If (pkt == 'Routing')
02:{        Capture and update destination field ;
03:        Send route ACK to sender;
04:}
05:Else if (pkt == 'TCP')
06:{        Block TCP packet }
07:Else If {pkt =='UDP'}
08:{        Capture UDP packet;
09:Can't Send to Destination;
10:}
11:Else ( pkt =='other')
12:        {Drop; }
13:        Set inf_pkt= (scan_rate * s_max_ / selfish node);
14:        infected packet send's to all normal node
15:        Selfish_Broadcast (inf_pkt, nexthop)
16:        { Set priority = 1 //Higher priority
17:Send inf_pkt = 100 pkts/ms // greater than the limit
18:Find (number of pkt accepted node)
19:}
```

**Elimination of Selfishness Algorithm**

```
00:        Set IPS node = p ; // IPS node
01:        Set routing =AODV ;
02:        RREQ_B(p, n, rr) // broadcast for communication and send 4:request packet toall n nodes via p node
03 :{ If ((rr<=250) && (next hop >0))
04 :{        Set inf_rm = ( scan_rate *pkt s_max_ / selfish node);
05: If (inf_rm => 100)
06 :{ Selfish Node Block ; }
```

#### 4.3.3. Check_Selfishness (S,D,M)

```
00:        { If ((node € M) && (pkt < 100 pkts/ms)
01:        {pkt accepted by neighbour
02:Node infection remove via inf_rm parameter ;
03:        }}}
04:Node unreachable;
05:        }
06:Node out of range;
07:        }
```

### 4.4 Working of proposed model

The model consist of Brain Trapping Function Nodes(BTFN) shown in Fig 2, are created in ad hoc network the number of BTFN belongs to the generics like area, radio range strength, data importance. The BTFN is very effective because it takes concepts of various fields like theory of computation, neural network, artificial intelligence. Virtualization is the domain which gain fame day by day instead of old conventional methods .In this method, whenever any node would be reached to the state of the power off , it immediately send signal to nearest BTFN node requesting for virtual node creation on hearing the request only the nearest BTFN node respond to that node and saves the status plus all the information in its memory, there is no deficiency of memory capacity in it because this model uses turi machine having infinite memory capability for this purpose. Then BTFN node creating virtual node in adhoc environment this is somewhat similar concept to have more than one operating system virtually onto a single operating system .This could be done by virtual layer for the model. Now when the same node previously exhausted(in terms of resource) come again(power-on)detected by BTFN node based on unique id information that stored in memory table of BTFN node, it immediately send this information to other BTFN node so that all the BTFN node deleted this information from its memory continuously. Before removing virtual node the BTFN node which detects the power-on occurrence of node resend all the data, condition of particular node information

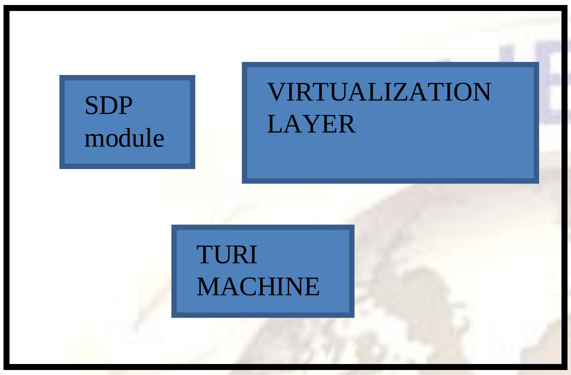

Fig. 2 Architecture of BTF node

to the same node. During the period the node is power off all the information exchanged in that time gap is up to date by BTFN node .Hence there is no discrepancy of old and new information mismatch. But some nodes are malicious in nature means they know that they are not going to die even knowing the truth they behave selfishly .For such types of nodes we have Credit risk[1] calculation and SDP modules that would take attention about such type of selfish nodes. It will considerably reduce false positives and false negatives due to network disconnection and minimize the impact of selfish nodes

### 4.5 SCF-tree with Collaborative monitoring system

The SCF-tree[1] based replica allocation methods are divine by human friendship handling in the world, where each person makes their own friends forming a web and forming friendship by himself/herself. He/she does not have to communicate these with others to insist the friendship. Then define $G_i^{ns}$ as the undirected graph $G_i^{ns} = (IN_i^{ns}, IL_i^{ns})$ which consists of a finite set of non selfish nodes detected by Ni, $IN_i^{ns}$ and a finite set of communication links among nodes $IN_i^{ns}$ , $IL_i^{ns}$ is derived by a smooth[1] over operation in graph theory. If there exists a path

$$N_a,\ N_{b,\dots\dots,}N_l \in IN_i^s = IN_i - IN_i^{ns} \qquad (3)$$

It assumes that all nodes are non-selfish nodes for simplicity. Since the SCF-tree consists of only nonselfish nodes, we need to calculate the degree of selfishness to employ real-world friendship handling to replica allocation in a MANET.

#### 4.5.1 Algorithm for build SCF-tree

```
00: /* Ni makes SCF-tree height d*/
01:        constructSCFTree(){
02:         add Ni to SCF-tree as the root node;
03:         checkChildnodes(Ni);
04:        return SCF-Tree;}
05: Procedure checkChildnodes(Nj){
06: /* $IN_j^a$ is a set of nodes that are adjacent nodes to $N_j^*$/
07:        for(each node Na ∈ $IN_j^a$){
08:        if (distance between Na and the root >d)
09:        continue;
10:        else if (Na is an ancestor of Nj in $T_i^{SCF}$)
11:        continue:
12:        else{ append Na to $T^{SCF}$ as a child of Nj;
13:        checkChild nodes(Na);
14 :        }}}
```

At every relocation period, each node changes its own tree based on the network topology of that moment.

#### 4.5.2 Implementing Collaborative Monitoring within SCF Tree

If one node has previously detected a selfish node using credit risk value, it can spread this information to other nodes when a contact begins. It implies that a node has a positive if it knows the selfish node. The node is overhearing the packets of the neighborhood. Thus, when it starts receiving packets from a new node it is considered to be a new contact. Then, the node transmits one message including all known positives it knows to this new contacted node. The number of messages required for this task is the overhead of this method. A collaborative node can have a positive when a contact occurs between the connected nodes. In the model one of the nodes is the selfish node. Then, the collaborative monitoring node can identify it using its monitoring and have a positive about this selfish node. Even so, a contact does not always simply detection. To model this fact, it introduce a probability of detection (pd).This probability depends on the effectiveness of the monitoring system and the type of contact (for example if the contact time is very low, the node does not have enough information to evaluate if the node is selfish or not). A node has two states: NOINFO, when the node doesn't know about the malicious node and POSITIVE when the node has information about the selfish nodes. All nodes have an initial state of NOINFO and they can update their initial state when a contact occurs. Using a contact rate $\lambda$ it can update the states of the particular node.

#### 4.5.3 Enforcement of Collaborative contact

Consider both nodes are collaborative. Then, if one of these nodes has one or more positives, it can spread this information to the other mobile node, hence from that moment, both nodes have these positives where c represents the number of collaborative nodes in the POSITIVE state. The degree of collaboration is a global parameter of the network to be evaluated. This value is used to reflect that either a message with the information about the selfish nodes is lost or that a node temporally does not collaborate.

### 4.5.4 Algorithm for finding the detection time of the selfish node

```
00:  pij denoting the transition rate from transient state si to
       absorbing state sj.
01:  Given a state si =(c) the following transitions can occur:
02:  (c) to (c+1): This case takes place when a collaborative
       node changes from NOINFO to POSITIVE state.
03: The transition probability is tc = (λpd + λpcc)(C − c).
04: λPd represents the probability of detection of a selfish node.
05: λPcC is the probability of transmission for the information of the selfish node.
06: Finally, factor (C − c) represents the number of pending nodes.
07: (c) to (c): This is the probability of no changes, and its value is t0 = 1− tc.
```

The collaborative monitoring method is used to reduce the detection time & cost of the each node and reduce query delay.

### 4.6. Allocating replica regarding Degree of selfishness

This technique takes into consider the degree of selfishness[1] in allocating replica. Less selfish nodes should be visited first at the same SCF-tree level. This one makes more frequently accessed data items rest on less selfish nodes. Consider that a node can use some part of its memory space selfishly[1], so divide memory space Mi for replica logically into two parts:

- Selfish area (Ms)
- Public area (Mp)

Each node may use its own memory space Mi freely as Ms and Mp. In each mobile node, Ms will be used for data of local interest to produce average query delay and Mp for public data is set to hold data for other nodes to improve data accessibility. Consequently, each mobile node allocates replicas in descending order of its own access frequency. Each node Ni executes the replica allocation algorithm at every relocation period after construct its SCF-tree. At first, a mobile node determines the priority for allocating replicas. The priority is based on Degree of selfishness. A mobile node allocates a replica to the expected node in its SCF-tree once during a single relocation phase; a node has at most one expected node for each replica. If its own Ms is not full, Ni allocates replica to its Ms first. If its own Ms becomes full, the node requests replica allocation to nodes in its SCF according to degree of selfishness of each node.

Procedures for replica allocation:

1. Each mobile node allocates replica at its interest.
2. When each mobile node receives a request for replica allocation from Nk during a relocation period, it make decision whether to accept the request according to the degree of selfishness.

#### 4.6.1 Algorithm for allocating replica regarding Degree of selfishness during relocation period

```
00: /*Ni executes this algorithm at replication period*/
01: replica_allocation(){
02: Li=make_priority(Degree of selfishness)
03:     for (each data item $ID_i$) {
04:             if (Ms is not full)
05:     allocate replica of the data item to Ms;
06:     else/*Ms is full*/
07: allocate replica of the data to the expected node;
08:     /*the expected node is selected from Li */
09:     if (Mp is not full)
10:     allocate replica of the data item to Mp;}}
11:     while (in a relocation period){
12:     if(Nk requests for the allocation of Dq)
13:     replica_allocation_for_others(Nk,Dq);}}
14:     Procedure make_priority (Degree of selfishness){
15:     for (all vertices in $T^{SCF)}${
16:     select a vertex in $T_i^{SCF}$ in order of Degree of
        selfishness;
17:     add the selected vertex id to Li;}
18:     return li;}
19:procedurereplica_allocation_for _others(Nk,Dq){
20:     if(Nk is in $T^{SCF}$ and Ni does not hold Dq){
21:     if (Mp is not full ) allocate Dq TO Mp;
22:     else (/*Mp is full*/
23:if(Ni holds any replica iem of local interest in Mp)
24:             replace the replica with Dq;
25:             else{
26:/* Nh is the node with the highest $nCR_i^h$
     amidst the nodes which allocated replica to Mp*/
27:     if ($nCR_i^h$> $nCR_i^k$)
28: replace the replica requested by Nh with Dq;}}}}
```

The new replica allocation method based on SCF tree with Degree of Selfishness and closer nodes allocate replica with low communication cost and produce high data accessibility compared to normal SCF tree replication.

## V. PERFORMANCE EVALUATION

In simulation, the number of mobile nodes is set to 40. Each mobile node has its own memory space and moves with a velocity from 0 -1 (m/s) over 50(m)*50(m) flatland. The movement pattern of nodes follows the random waypoint model. The default relocation period is set to 256 units of simulation time which we vary from 64 to 8,192 units of simulation time.Here75percent of nodes set to be type3 (partially selfish) and the remaining nodes to be type-2 (fully selfish). Type-3 nodes having 25, 50, and 75 percent of its memory space for the selfish space. The accessing frequency is assumed to follow Zipf distribution.

### 5.1 Performance metrics

1. **Overall selfishness alarm:** This is the ratio of the overall selfishness alarm of all nodes to all queries.
2. **Communication cost:** This is the total hop count of data transmission for selfish node detection and replica allocation.
3. **Average query delay:** This is the number of hops from a requester node to the nearest node with the requested data item.
4. **Data accessibility:** This is the ratio of the number of successful data requests to the total number of data requests.
5. **Degree of selfishness:** It is represented by the size of shared memory space and the number of shared data items.
6. **Probability of detection (pd):** Effectiveness of the selfish node detection by monitoring and the type of contact.
7. **Probability of collaboration (pc):** Effectiveness of collaboration between the nodes
8. **Detection time:** The time taken to detect the selfish nodes.

### 5.2 Parameter Setting in Our Strategy

For the selfishness detection algorithm, it uses the threshold δ and for the selfishness features update algorithm, it uses the predefined wait time ɯ and need to initialize the selfishness alarm $P_i^k$. In building the SCF-tree, it uses the depth d and set ɯ to 50 units of simulation time. $P_i^k$ is initialized to 0 and δ is set to 0.7.To evaluating the performance of collaborative monitoring, it uses three different sets of values for *pc* and *pd* were used. The first set (1, 0.8) is a full collaborative network with a high probability of detection, the second set has a reduced degree of collaboration(0.7) and finally the last set has a low probability of detection(0.3).It provide that the greater the number of nodes, the lesser the detection time and the greater the number of messages.

### 5.3 Simulation Results

To establish the effectiveness of our detection method hence many selfish nodes will be removed from the replica allocation stage and many reliable nodes will serve data requests from the network, our detection method is added in SCF tree to detects selfish nodes effectively and the detected selfish nodes are removed from replica allocation groups, can reduce the overall selfishness alarm effectively. In these technique 70 percent of total communication cost is caused by replica allocation/relocation and 30 percent is caused by selfish node detection, communication cost of our techniques is less sensitive than traditional replica allocation techniques. In Fig 3 by using improved detection method the number of successful requests being locally answered increases to a small extent. This is because when the number of nodes in the SCF-tree with degree of selfishness is very small, the local public memory may be used for data of local discretion temporarily, the data accessibility improves with the wide range of communication in Fig 4, and hence more nodes become connected.

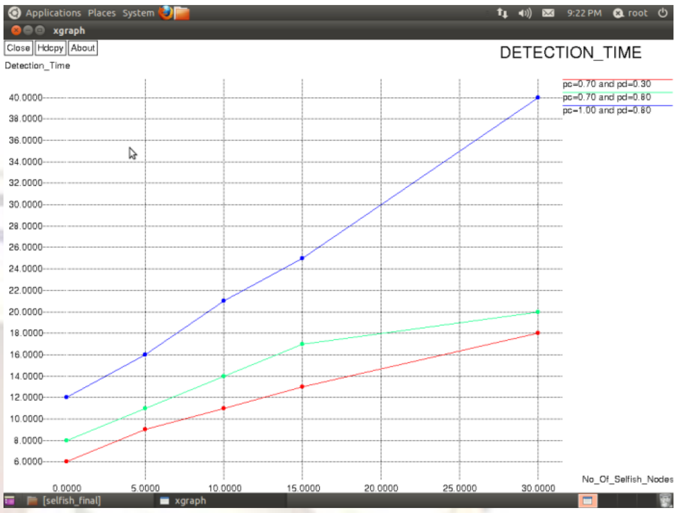

Fig. 3 Varying Detection time

It clearly shows that the Detection method and replica allocation with degree of selfishness works optimum.

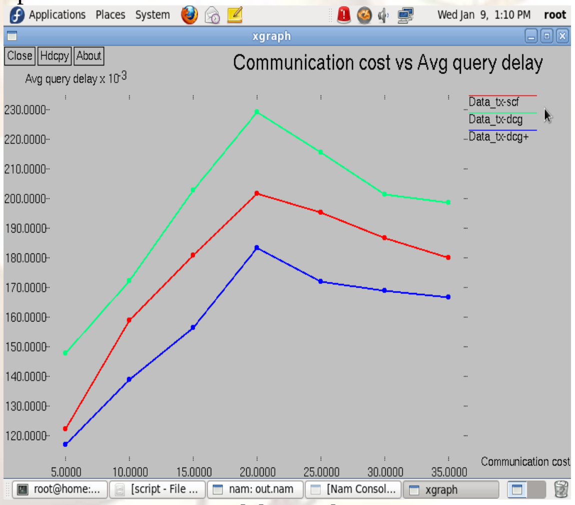

Fig.4 Varying Query delay and Communication cost

### 5.4 Related works

Three main schemes were proposed for overcoming selfishness, one which is reputation scheme, credit payment scheme and the other is acknowledgment based technique. Reputation systems have the following main components: Monitoring unit, which discovers the node's neighbour behaviour, reputation unit which validate the neighbouring nodes' behaviour construct a reputation table, alarming unit which sends and receives alarms, and path manager which manage routing and forwarding decisions based on reputation values. Such components can be either executed in each node or distributing among the mobile nodes. Two reputation schemes were considered in the literature. The first is

CONFIDANT [8], where alarm message and global reputation were employed to accomplish the aim of reputation sharing. Therefore, it could acquire other nodes experiment fully and exclude misbehaving nodes. The second one is CORE [7], which used a monitoring technique and reputation mechanism, where each node calculate a reputation value for each neighbour using a reputation mechanism that distinct between indirect and functional reputation. In this technique, the network's nodes show unwilling to set services to the selfish node if its reputation is lower than a threshold, which can lead to be excluded from the network. Bansal and Baker proposed an OCEAN [10] scheme, where each node consumes one-hop neighbour’s reputation, reputation modification couldn't be exchanged with each other. In OCEAN selfish nodes have second chance to get service and change some parameters to determine false accusation and discrepant reputation value. On the other hand, credit based schemes provide mechanisms that cause nodes to be well behaved. In these schemes, the concept of virtual electronic payments was proposed, where nodes are rewarded for messages forwarding through trading virtual electronic payments with source and neighbor nodes throughout the routing path to destination. In the credit based systems, Buttyan and Hubaux [8] introduced nuglets as credits for handle forwarding proceedings.

Two electronic payment models, message purse model and message trade model, were proposed. Intermediate nodes acquire nuglets when forwarding the message. In the latter, a relay node accepts messages from the previous node and sells messages to the next node in the path. The credit-based system in [5] uses message receipts and credit clearance service. When a node forwards a message, it holds a receipt and uploads it to the clearance service for credits.

The last category is acknowledgment based technique; it depends on the acceptation of an acknowledgment to conclude that a packet has been forwarded [9]. It proposed the 2ACK [6] system where nodes directly send acknowledgment to hops upstream to verify the co-operation between the nodes. This technique is fictile to coalition [11] of two or more consecutive nodes. Furthermore, conspire nodes can frame honest nodes by requesting not to receive the acknowledgment. The use of two-hop ACK approach for monitoring packets (RREP, RRER) and consider promiscuous-based overhearing technique for monitoring broadcast packets RREQ [12] suggest that the monitoring node compares the ratio of relay RREQ count between its neighbor and itself. If the ratio is minimum than a threshold, the neighbor node is regarded as malicious and its packet is dropped as the penalty.

TABLE II
COMPARATIVE STUDY OF EXISTING DETECTION METHODS

| Credit based system | Reputation based system | Acknowledgement based system |
|---|---|---|
| **Based on Virtual electronic payment.** | Based on Reputation metric. | Based on Acknowledgement |
| **It requires Costly Security modules for protecting the virtual money** | No such modules required. | No such modules required. |
| **Packet purse model(PPM),pack-et trade model(PTM) are exploits credit based approaches.** | Watchdog and Pathrater are considered as Reputation based approach. | Secure 2ACK Routing Protocol is used. |

## VI. CONCLUSION

The collaborative monitoring method and replica allocation with degree of selfishness address the problem of selfish nodes in the form of replica allocation. The selfish replica allocation could reduce the overall data accessibility in a MANET and does not handle false alarms. The proposed strategies are inspired by the real-world reflection from economics in terms of credit risk and in human friendship management in terms of selecting friends completely at its own discretion. The applied notion of credit risk and the collaborative monitoring method to detect selfish nodes outperforms the existing detection methods and every node in a MANET calculates credit risk information on other connected nodes individually to measure the degree of selfishness. The collaborative monitoring method is used to reduce the detection time & cost of the each node. Since existing replica allocation techniques failed to consider selfish nodes and also proposed novel replica allocation techniques. The research is currently going on the impact of different mobility patterns.The proposed strategies improves the data accessibility, reduces communication cost, and average query delay and also to reduce the detection time of the selfish nodes.

## ACKNOWLEDGEMENT

I would like to thank reference authors and also like to thank the anonymous reviewers, whose comments and suggestions have helped them to improve the quality of the original manuscript.

## REFERENCES

[1] Jae-Ho Choi, Kyu-Sun Shim, *"Handling Selfishness in Replica Allocation over a Mobile Ad Hoc Network"* SangKeun Lee, and Kun-Lung Wu, Fellow, IEEE.2012.

[2] T. Hara, "*Effective Replica Allocation in Ad Hoc Networks for Improving Data Accessibility*," Proc. IEEE INFOCOM, pp. 1568- 1576, 2001.

[3] T. Hara and S.K. Madria, "*Data Replication for Improving Data Accessibility in Ad Hoc Networks*," IEEE Trans. Mobile Computing, vol. 5, no. 11, pp. 1515-1532, Nov. 2006.

[4] L.J. Mester, "*What's the Point of Credit Scoring*?" Business Rev., pp. 3-16, Sept. 1997.

[5] Y. Liu and Y. Yang, "*Reputation Propagation and Agreement in Mobile Ad-Hoc Networks*," Proc. IEEE Wireless Comm. And Networking Conf., pp. 1510-1515,

[6] S. Marti, T. Giuli, K. Lai, and M. Baker, "*Mitigating Routing Misbehavior in Mobile Ad hoc Networks*," Proc. ACM MobiCom, pp. 255-265, 2000.

[7] L. Anderegg and S. Eidenbenz, "*Ad Hoc-VCG: A Truthful and Cost-Efficient Routing Protocol for* V. Srinivasan, P. Nuggehalli, C. Chiasserini, and R. Rao, "*Cooperation in Wireless Ad Hoc Networks*," Proc. IEEE INFOCOM, pp. 808-817, 2003.

[8] D. Hales, "*From Selfish Nodes to Cooperative Networks - Emergent Link-Based Incentives in Peer-to-Peer Networks*," Proc. IEEE Int'l Conf. Peer-to-Peer Computing, pp. 151-158, 2004.

[9] S.U. Khan and I. Ahmad, "*A Pure Nash Equilibrium-Based Game Theoretical Method for Data Replication across Multiple Servers*," IEEE Trans. Knowledge and Data Eng., vol. 21, no. 4, pp. 537-553, Apr. 2009.

[10] M.J. Osborne, *An Introduction to Game Theory*. Oxford Univ., 2003.

[11] B.-G. Chun, K. Chaudhuri, H. Wee, M. Barreno, C.H. Papadimitriou, and J. Kubiatowicz, "*Selfish Caching in Distributed Systems: A Game-Theoretic Analysis*," Proc.

[12] B.-G. Chun, K. Chaudhuri, H. Wee, M. Barreno, C.H. Papadimitriou, and J. Kubiatowicz, "*Selfish Caching in Distributed Systems: A Game-Theoretic Analysis*," Preoc. ACM Symp. Principles of Distributed Computing.