RESEARCH ARTICLE OPEN ACCESS

Casting Selfish Nodes Into Non Selfish In Replica Allocation Over Mobile Ad Hoc Network

Jaya Krishna Katti*, Venkata Naga Sudheer**
*(M.Tech Final Year student, Department of CSE, QIS College of Engineering and Technology, Ongole)
** (Asst. Professor, Department of C.S.E, QIS College of Engineering and Technology, Ongole)

ABSTARCT
The nodes in Mobile Ad Hoc Network are resource constrained. This drawback and mobility nature of nodes can cause network partitioning and degradation of performance. Traditionally replica allocation techniques were used to overcome the drawback of performance degradation. It is assumed that all nodes in MANET share their memory resources fully. However, in reality the nodes may behave selfishly instead of cooperating in communication process. The selfishness exhibited by some of the nodes may lead to reduced quality of data accessibility over network. Recently Choi et al. studied the impact of selfish nodes from replication allocation perspective. Their solution is named “Selfish Replica Allocation”. They proposed a novel replica allocation and selfish node detection. In this paper we implemented those mechanisms for casting selfish nodes in non selfish from the perspective of replica allocation. We built a custom Java simulator to demonstrate the proof of concept. The empirical results reveal that the approach shows higher performance when compared with existing techniques in terms of communication cost, data accessibility and average query delay.
***Index Terms* –** Mobile ad hoc networks, selfish node, selfish replica allocation

I. INTRODUCTION

Due to the advent of innovative technologies in mobile devices, their usage has become ubiquitous. MANET (Mobile Ad Hoc Network) is a network of mobile nodes which are automatically configured without a fixed infrastructure. The existing technologies in the domain of mobile computing have paved the way for creation of MANETs [1], [2], [3]. MANETs are automatically established and they have real utility in real time applications [4] such as battle fields, natural disasters and so on. Therefore in emergency situations MANETs are suitable to address communication problems. Peer-to-peer mobile overlay is another interesting MANET that can be used for file sharing and searching [5], [6]. Network partitions are caused by node mobility as they move from one place to another place. Thus data present in a mobile node can become inaccessible. The data accessibility measure is used to know the level of data accessibility in MANETs [7]. To overcome the data inaccessibility problem due to node mobility, the data can be replicated in many nodes [1], [7], [8]. By making replicas, the node scan provides requested data in cooperative fashion. However, in reality nodes may behave selfishly to get rid of resource consumption or any other reason. Many replica allocation techniques came into existence. Recently Choi et al. [20] presented a selfish replica allocation model which is presented in fig. 1.

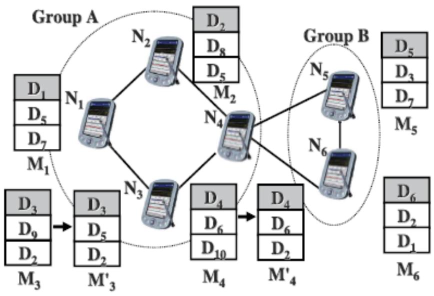

Fig. 1 – Selfish replica allocation

As can be seen in fig. 1, many nodes are interconnected to form MANET through wireless media. There is some replication allocation scheme named DCG as explored in [7]. Wireless links are represented by straight lines between nodes. Rectangle with an item in gray color indicates original data; white boxes represent replication allocations. The table1 shows access frequencies of data.

Data	Nodes					
	N1	N2	N3	N4	N5	N6
D1	0.65	0.25	0.17	0.22	0.31	0.24
D2	0.44	0.62	0.41	0.40	0.42	0.46
D3	0.35	0.44	0.50	0.25	0.35	0.37
D4	0.31	0.15	0.10	0.60	0.09	0.10
D5	0.51	0.41	0.43	0.38	0.71	0.20
D6	0.08	0.07	0.05	0.15	0.20	0.62
D7	0.38	0.32	0.37	0.33	0.40	0.32
D8	0.22	0.33	0.21	0.23	0.24	0.17
D9	0.18	0.16	0.19	0.17	0.24	0.21
D10	0.09	0.08	0.06	0.11	0.12	0.09

Table 1 – Access frequencies of nodes

As can be seen in table 1, the frequently accessed items by the nodes 3 and 4 are shown in gray color. To reduce duplication allocation DCG technique is employed. Delay in data access is resulted when a node behaves selfishly in a group. For instance, when node 3 misbehaves, the nodes in the group will be able to access data properly. In order to overcome the problems of query delay, improve data accessibility and response time, replication techniques are used in MENTs. Nodes in the MANET have fewer resources. For this reason they may behave selfishly to save energy by not cooperating with the objective of the network. When the nodes behave selfishly they can save energy and enjoy resources causing delay in data access [9], [10].

This paper has implemented the mechanisms proposed in [20] using a custom Java simulator which demonstrates selfish replica allocation. The remainder of this paper is organized into the following sections. Section II reviews literature relevant to replication allocation in MANETs. Section III presents proposed replica allocation strategy. Section IV shows experimental results while section V concludes the paper.

II. RELATED WORK

Mobile Ad Hoc Network is a network formed among mobile nodes without fixed infrastructure. MANET is of two types as explored in [11], [12], and [13]. One type of MANET is known as closed MANET where nodes exhibit expected behavior. Other type of MANET is known as open MANET where the nodes have some sort of guaranteed behavior. However, nodes in MANET can misbehave to save energy resources selfishly. This behavior of nodes in MANET was handled by many techniques [13]. This paper considers open MANET. All existing techniques employed to handle selfish nodes in MANET are categorized into three types. They are game theory based, reputation based and credit-payment based. Reputation based techniques allow the MANET to observe nodes for monitoring selfish behavior [14], [15]. Credits are evaluated in case of credit payment methods where nodes are rewarded when they behave well while participating in data access procedures such as data forwarding [9], [16]. The game theory based methods are employed by MANET nodes to have their own strategies to increase their profits [10], [17].

Packet forwarding is used in all the techniques for communication in MANETs. However, the focus of this paper is on selfish replica allocation. In [18] many trust models were presented. The selfishness is treated as refusal to forward data besides dropping packets. These techniques are not effective for replica allocation which gets rid of selfish behavior. In [7] also techniques were given for replication allocation. The techniques like DCG have shown highest accessibility of data. However, DCG is not suitable for MANET. In [8] problems like query delay and data inaccessibility were handled. In [19] cooperative caching is used in order to overcome the problem of selfish nodes. They features employed include Cache Path, Cache Data and Hybrid. In [20] a novel replica allocation technique was proposed. It could detect selfish nodes effectively besides allocating replicas successfully.

III. PROPOSED REPLICA ALLOCATION STRATEGY

In this section we describe the replica allocation strategy and algorithms used as proposed in [20]. The algorithms include SCF-tree building, selfish node detection and replica allocation. In order to detect selfish nodes, credit concept is used. Credit risk scores are used to determine the selfish behavior of nodes. The algorithms are employed in every node. Therefore every node is capable of detecting selfish nodes. Listing 1 shows pseudo code for detecting selfish nodes.

```
Algorithm 1. Pseudo code to detect selfish nodes
At every relocation period
/* node Ni detects selfish nodes with this algorithm */
detection (){
for (each connected node Nk ){
if (nCRki<δ)Nk is marked as non-selfish;
else Nk is marked as selfish;}
waits until replica allocation is done;
for (each connected node Nk){
if (Ni has allocated replica to Nk){
NDki = the number of allocated replica;
SSki = the total size of allocated replica;}
else{
NDki = 1;
SSki = the size of a data item;
} } }
```

Listing 1 – Selfish node detection algorithm

The algorithm presented in listing 1 runs in every node in MANET. In every relocation period, the node can make use of credit risk concept and detect selfish nodes. The algorithm 2 presented in listing 2 updates selfish features.

```
Algorithm 2. Pseudo code to update selfish features
At every query processing time
/*When Ni issues a query*/
update_SF(){
while (during the predefined time !){
if (an expected node Nk serves the query)
decrease Pki;
if (an unexpected node Nj serves the query){
NDji = NDji + 1;
SSji = SSji + (the size of a data item);
} }
if (an expected node Nk does not serve the query){
increase Pki ;
NDki = NDki - 1;
SSki = SSki - (the size of a data item);
} }
```

Listing 2 – Algorithm to update selfish feature

As can be seen in listing 2 at each and every query the credibility of nodes is computed and the selfish feature of the nodes is updated. The algorithm shown in listing 3 is used to build SCF tree.

```
Algorithm 3. Pseudo code to build SCF-tree
/* node Ni makes SCF-tree with a parameter of depth d*/
constructScfTree(){
append Ni to SCF-tree as root node;
checkChildnodesNiÞ;
return SCF-tree;}
Procedure checkChildnodes-NjÞ{
/*INajis a set of nodes that are adjacent nodes to Nj*/
for (each node Na 2 INaj){
if (distance between Na and the root > d )
continue;
else if -Na is an ancestor of Nj in TSCFi)
continue;
else{ append Na to TSCFi as a child of Nj;
checkChildnodes(Na); } } }
```

Listing 3 – SCF tree building algorithm

As can be seen in listing3, the algorithm is used to build a tree known as SCF tree which is best used in replication allocation successfully. The actual replication allocation mechanism is coded in listing 4.

```
Algorithm 4. Pseudo code for replica allocation
/_Ni executes this algorithm at relocation period _/
replica_allocation(){
Li = make priority(TSCFi );
for (each data item 2 IDi){
if (Ms is not full)
allocate replica of the data to Ms ;
else{/_Ms is full _/
allocate replica of the data to target node;
/_ the target node is selected from the Li
if (Mp is not full)
allocate replica of data to Mp; } }
while (during a relocation period){
if (Nk requests for the allocation of Dq)
replica_allocation_for_others (Nk;Dq); } }
Procedure make_priority ðTSCFi ) {
for (all vertices in TSCFi ){
select a vertex in TSCFi in order of BFS;
append the selected vertex id to the Li }
```

Listing 4 – Replica allocation algorithm

As can be seen in listing 4, identification of all data items is done by the algorithm for replication. Replicas are placed at each node which is organized in the descending order by access frequencies of nodes. Initially breadth first search is used to obtain priorities and based on the priorities replica allocation is carried out.

IV. EXPERIMENTAL RESULTS

We built a prototype application that is a custom Java simulator to demonstrate the proof of concept of casting selfish nodes to non selfish. The application is built in Java platform. The environment used to implement the prototype includes a PC with 2 GB RAM and Core 2 dual processor running Windows XP operating system. The simulations are made using the customer simulator application. The simulation demonstrates the creation of MANET, detecting selfish nodes and handling the problem of selfishness associated with nodes. The simulation parameters are presented in table 2.

Parameter(unit)	Value(default)
No. of nodes	40
No. of data items	40
Radius of communication range	1~19(7)
Size of network	50*50
Size of memory space	2~40(10)
Percentage of selfish nodes	0~100(70)
Maximum velocity of	1

Table 2 – Simulation Parameters

As can be seen in table 2, the simulation parameters are presented. The parameters considered include number of nodes, number of data items, radius communication range, size of network, size of memory space, percentage of selfish nodes and maximum velocity.

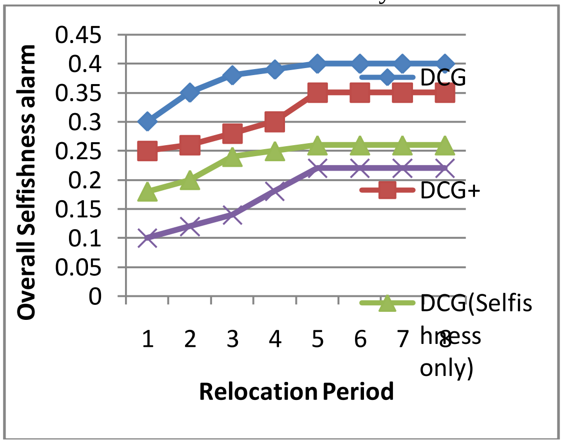

Fig. 2 –Relocation period vs. overall selfishness alarm

As seen in fig. 2, relocation period is presented in X axis while the Y axis represents overall selfishness alarm. The results show that the DCG+ shows very less selfishness alarm.

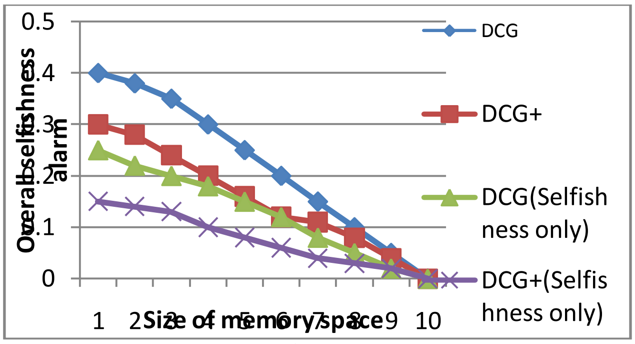

Fig. 3 –Size of memory space vs. overall selfishness alarm

As can be seen in fig. 3, the size of memory space is shown in horizontal axis while the vertical axis represents overall selfishness alarm. The results show that the DCG+ shows very less selfishness alarm.

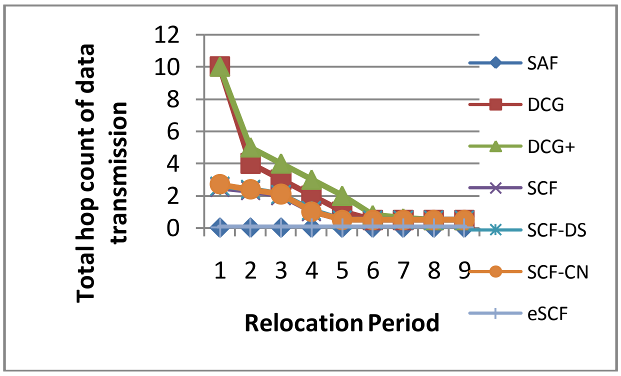

Fig. 4 –Varying relocation period vs. communication cost

In fig. 4, the horizontal axis represents relocation period while the vertical axis shows total hop count of data transmission.

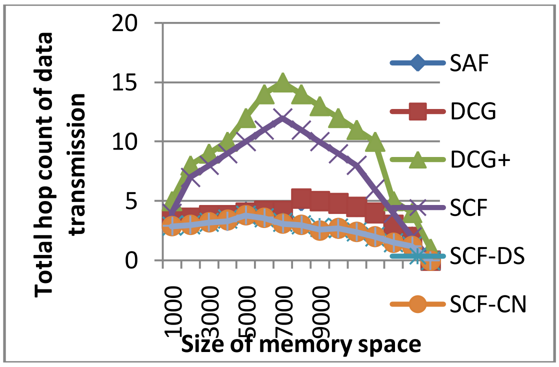

Fig. 5 –Varying size of memory space vs. communication cost

As seen in fig. 5, size of memory space is represented by horizontal axis while the vertical axis represents total hop count of data transmission. As shown in result, the increase in size of memory space increases communication cost. However, the debugging starts after a while. After some time, the replicas are in many nodes. Thus the delay in communication is reduced. When compared with other techniques, the communication cost of SCF is less.

V. CONCLUSION

In this paper we implemented mechanisms for selfish node detection and replication allocation to reduce selfish behavior in MANET. These mechanisms were originally proposed in [20]. The main focus of the paper is selfish replica allocation. It does mean that the mechanisms identity selfish nodes and convert them into non-selfish nodes. The algorithms used in this paper were proposed by Choi et al. [20]. The mechanisms take care of selfish node allocation and also handling selfish nodes in the MANET. We implemented a custom Java simulator to demonstrate the proof of concept. The results revealed that the mechanisms could reduce delay in response, improve data accessibility besides improving the performance of network.

REFERENCES

[1] 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.

[2] T. Hara and S.K. Madria, "Consistency Management Strategies for Data Replication in Mobile Ad Hoc Networks," IEEE Trans. Mobile Computing, vol. 8, no. 7, pp. 950-967, July 2009.

[3] S.-Y. Wu and Y.-T. Chang, "A User-Centered Approach to Active Replica Management in Mobile Environments," IEEE Trans. Mobile Computing, vol. 5, no. 11, pp. 1606-1619, Nov. 2006.

[4] P. Padmanabhan, L. Gruenwald, A. Vallur, and M. Atiquzzaman, "A Survey of Data Replication Techniques for Mobile Ad Hoc Network Databases," The Int'l J. Very Large Data Bases, vol. 17, no. 5, pp. 1143-1164, 2008.

[5] G. Ding and B. Bhargava, "Peer-to-Peer File-Sharing over Mobile Ad Hoc Networks," Proc. IEEE Ann. Conf. Pervasive Computing and Comm. Workshops, pp. 104-108, 2004.

[6] M. Li, W.-C. Lee, and A. Sivasubramaniam, "Efficient Peer-to-Peer Information Sharing over Mobile Ad Hoc Networks," Proc. World Wide Web (WWW) Workshop Emerging Applications for Wireless and Mobile Access, pp. 2-6, 2004.

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

[8] L. Yin and G. Cao, "Balancing the Tradeoffs between Data Accessibility and Query Delay in Ad Hoc Networks," Proc.

IEEE Int'l Symp. Reliable Distributed Systems, pp. 289-298, 2004.
[9] L. Anderegg and S. Eidenbenz, "Ad Hoc-VCG: A Truthful and Cost-Efficient Routing Protocol for Mobile Ad Hoc Networks with Selfish Agents," Proc. ACM MobiCom, pp. 245-259, 2003.
[10] 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.
[11] K. Balakrishnan, J. Deng, and P.K. Varshney, "TWOACK: Preventing Selfishness in Mobile Ad Hoc Networks," Proc. IEEE Wireless Comm. and Networking, pp. 2137-2142, 2005.
[12] H. Miranda and L. Rodrigues, "Friends and Foes: Preventing Selfishness in Open Mobile Ad hoc Networks," Proc. IEEE Int'l Conf. Distributed Computing Systems Workshops, pp. 440-445, 2003.
[13] Y. Yoo and D.P. Agrawal, "Why Does It Pay to be Selfish in a MANET," IEEE Wireless Comm., vol. 13, no. 6, pp. 87-97, Dec. 2006.
[14] Y. Liu and Y. Yang, "Reputation Propagation and Agreement in Mobile Ad-Hoc Networks," Proc. IEEE Wireless Comm. and Networking Conf., pp. 1510-1515, 2003.
[15] S. Marti, T. Giuli, K. Lai, and M. Baker, "Mitigating Routing Misbehavior in Mobile Ad hoc Networks," Proc. ACM MobiCom, pp. 255-265, 2000.
[16] W. Wang, X.-Y. Li, and Y. Wang, "Truthful Multicast Routing in Selfish Wireless Networks," Proc. ACM MobiCom, pp. 245-259, 2004.
[17] V. Srinivasan, P. Nuggehalli, C. Chiasserini, and R. Rao, "Cooperation in Wireless Ad Hoc Networks," Proc. IEEE INFOCOM, pp. 808-817, 2003.
[18] H. Li and M. Singhal, "Trust Management in Distributed Systems," Computer, vol. 40, no. 2, pp. 45-53, Feb. 2007.
[19] G. Cao, L. Yin, and C.R. Das, "Cooperative Cache-Based Data Access in Ad Hoc Networks," Computer, vol. 37, no. 2, pp. 32-39, Feb. 2004.
[20] Jae-Ho Choi, Kyu-Sun Shim, SangKeun Lee, and Kun-Lung Wu, Fellow, IEEE, "Handling Selfishness in Replica Allocation over a Mobile Ad Hoc Network", IEEE TRANSACTIONS ON MOBILE COMPUTING, VOL. 11, NO. 2, FEBRUARY 2012.

K. Jaya Krishna was born in Ongole, Prakasam Dt, Andhra Pradesh, India. He received MCA from Anna University, Tamilnadu, India. Presently, he is pursuing M.Tech in C.S.E from QIS College of Engineering& Technology, vengamukkapalem, Ongole, Prakasam Dt, Andhra Pradesh, India. His Research interest includes Computer Networks, Data Mining.

T. Venkata Naga Sudheer was born in Marlapadu(V), Prakasam Dt, Andhra Pradesh, India. He received B.Tech from university of JNTUH, Hydrabad, and M.Tech from S.V University, Tirupathi, Andrapradesh, India. Presently, he is working as a Asst.Professor in C.S.E from QIS College of Engineering & Technology, vengamukkapalem, Ongole, Prakasam Dt, Andhra Pradesh, India. His Research interest includes Computer Networks and Data Mining.