Selfish Node Handling In the context of Replica Allocation in MANET's

K.Navatha , N.Sravanthi, L.Sunitha, E. Venkata Ramana

Department of Computer Science Engineering, Vidya Vikas Institute of Technology, Hyderabad, India.

Abstract
In a Mobile Adhoc Network (MANET), the mobility and resource constraints of mobile nodes may lead to network partitioning and performance degradation. All mobile nodes should participate fully by sharing memory space to increase data accessibility. But, some of the nodes can act as selfish nodes, only for partial participation or fully selfish with other nodes. Such selfish nodes are handled in replica allocation.

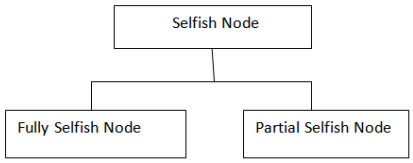
Key Terms
Selfish node, SCF-Tree, mobile ad-hoc network

1. Introduction

A Mobile Adhoc Network is a collection of autonomous wireless devices that move unpredictably, forms a temporary network without any fixed backbone infrastructure. In these networks each node acts as an end system and a router. These nodes are capable of both single hop & multi hop communication.
In MANET, the nodes are moving frequently. This lead to frequent network partitioning, causing some data to be often inaccessible to some of the nodes. Hence, data accessibility is often an important performance metric in MANET. Data are replicated at nodes other than owners to increase data accessibility to cope with frequent network partitions and also reduces query response time, if mobile nodes in a MANET have sufficient memory to store both all the replicas and the original data. Storing same replica by all nodes will lead to decrease of data accessibility. Hence, to maximize data accessibility, a node should not hold the same replica that is also held by many other nodes. This will increase its own query delay. A selfish node may not share its own memory space to store replica for the benefit of other nodes.

2. Behavior of Selfish Nodes in Manet

Selfishness for nodes are categorized into two types based on their behavior. Fully selfish nodes-The nodes do not hold replicas allocated by other nodes, but allocate replicas to other nodes for their accessibility. Partial selfish nodes-The nodes use their memory space partially for allocated replicas by other nodes. These nodes allocate replicas to other nodes for their accessibility. Each node in a MANET has limited memory locally and each node acts as a data provider, it provides several data items and as well as a data consumer. Each node holds data item replicas and maintains the replicas in local memory space.



The replicas are relocated in a specific period. There are 'm' nodes, N1,N2,…Nm. Any node can freely joins in a MANET.A mobile adhoc network is an undirected graph G=(IN,IL).Where 'IN' is a finite set of nodes and 'IL' is a finite set of communication links. Each node in a MANET has a unique identifier and they are denoted by N={N1,N2,…Nm}, where 'm' is the total number of nodes. Each node holds data items of equal size, and first data item in a memory is considered as its original data. Every data item has a unique identifier, denoted by D={D1,D2,…Dn}, where 'n' is the total number of data items. The remaining data items in a memory are treated as replicas for its particular node. Each node Ni has its own access frequency for data item and it does not change always. When a node Ni sends a request (query) for accessing of data item, first, the search has takes place in its own memory. The request is successful, when the node Ni holds the data item as its original data item (or) replica, otherwise the request is broadcasted. The request is also successful, when the node Ni gets reply from its adjacent nodes connected to Ni with one hop or multi hops. Otherwise, the request fails.

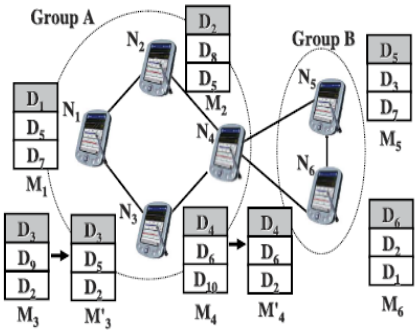


Fig.1. Selfish replica allocation

Manuscript received August 5, 2015
Manuscript revised August 20, 2015

In above figure, there are 6 nodes name as N1,N2,..N6 and their memory spaces are M1,M2,..M6. Each node access the frequency information from the access frequency table.

Table 1. Access frequency of nodes

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	0.45	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

Where each memory location contains 3 data items. First data item is a original one, and the remaining 2 data items are replica allocated.

In the above figure, Node 'N3' behaves 'selfish' by maintaining M3', instead of M3 to prefer the locally frequently accessed data for low query delay.

Due to the selfish behavior, D3, D5, D2, the three top most local frequent accessed items are maintained instead of D3, D9, D2. The nodes N1, N2, N4 in the above figure are no longer able to access D9. This will results in degradation of data accessibility.

Node 'N4' behaves partially selfish. This want to hold 'D2' locally as one of the locally frequently accessed data items. So, in this case N4 uses a part of its storage for its own frequently accessed data, where the remaining part is used for the benefit of overall data accessibility. So that N4 is decided to maintain M4' instead of M4.Data accessibility is degraded with the partial selfishness also. The nodes N1,N2,N3 are cannot access D10 because of partial selfishness in 'N4'.

3. Handling Selfish Nodes in Replica Allocation

To handle selfish nodes in MANET, 3 steps have to follow. They are

1. SELFISH NODE DETECTION
2. SCF-TREE CONSTRUCTION
3. REPLICA ALLOCATION

3.1. SELFISH Node Detection

In MANET, each node detects the selfish nodes based on credit risk.

Credit Risk=expected risk/expected value

```
10. Detection()
11. For(each connected Nk) {
12. If(nCRki > δ) Nk is selfish node
13. Else Nk is non selfish node
14. Wait until replica allocation is done;
15. For(each connected node Nk){
16. If(Ni has allocated replica to Nk)
17. NDik=Number of allocated replica;
18. SSik=Total size of allocated replica;
```

The size of shared memory space and the number of shared data items are used to represent 'expected risk' and the node specific features are used to represent 'expected value'. Algorithm for selfish node detection is given in List1.

```
10. Else {
11. NDik=1;
12. SSik=Size of  data item;}}}
```

List1: Algorithm for selfish node detection

Every node should execute this algorithm in order to detect the selfish node at relocation period. The following algorithm is to update selfish features in selfish node.

```
1.  SF_Update() {
2.  For(during the predefined time w) {
3.  If(the query is served by the expected node Nk)
4.   Pik--;
5.  If(the query is served by the unexpected node
    Nj)
6.  NDij +=1;
7.  SSij +=(Data item size);}}
8.  If(query is not served by the expected node Nk)
    {
9.   Pik++;
10. NDik -=1;
11. SSik -=(data item size); } }
```

List2: Algorithm for update selfish features

The algorithm in list2 is executed to update the selfishness features in selfish node.

3.2. SCF-TREE Construction

Selfish nodes are not participated in Self Centered Friendship Tree based replica allocation. Degree of selfishness should be measured by using credit risk score for each non selfish node participated.

```
1.  SCF-tree_Construct() {
2.  Append Ni to SCF-tree  as root node;
3.  Checkchildnodes(Ni);
4.  Return SCF-Tree;}
5.  Checkchildnodes(Nj) {
6.  For(each node Na belong to INja) {
7.  If (d < distance between Na and the root )
8.  Continue;
9.  Else if(Na is an ancestor of Nj in Tiscf)
10. Continue;
11. Else { append Na to Tiscf as a child of Nj;
12. Checkchildnodes(Na); } }}
```

List3: Algorithm For Scf-Tree Construction

3.2.1. Sample SCF-Tree Construction

In this example, we assume all the nodes are non-selfish in nature. Multiple roots are possible among the nodes in a MANET. First, select one node as root node for SCF-Tree. The neighbors of root nodes are its Childs. Later connect its neighbors as sub Childs and so on.

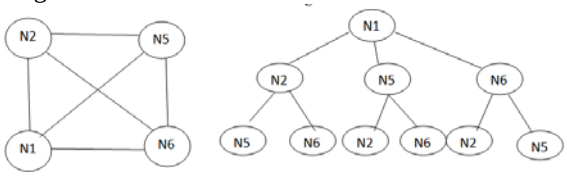


Fig 2.1: Graph

3.3. Replica Allocation

After construction of SCF-Tree, a node allocates replica at every relocation period.

```
1.  Allocate_replica()
2.  Li=make_priority(TiSCF);
3.  For(each data item belong to IDi) {
4.  If(Ms is not full)
5.  Allocate replica of the data to Ms;
6.  Else {
7.  Allocate replica of the data to the target
    node;
8.  If(Mp is not full)
9.  Allocate replica of the data to Mp;
10. }}while(during relocation period){
```

The following algorithm list out the steps for replica allocation. List 4.

```
11. If(Nk requests for the allocation of Dq)
12.  allocate_replica_others(Nk,Dq); }}
13. make_priority(TiSCF){
14. for(all vertices in TiSCF){
15. select a vertex in TiSCF in order of BFS;
16. append the selected vertex id to Li;}
17. return Li;}
18. allocate_replica_others(Nk,Dq){
19. if(Nk is in TiSCF and Ni does not hold Dq){
20. if(Mp is not full)allocate Dq to Mp;
21. else{
22. if(Ni holds any replica of local intrest in Mp)
23.replace the replica will Dq;
24.else { if(nCRik>nCRik) replace the replica
requested by Nh with Dq;}}}}
```

Fig 2.2:SCF-Tree of N1

4. Performance Evolution

The results are noted by made an experiment in a PC with 3GB RAM and CORE2 dual processor. The simulations are tested using NS2.Creation of MANET, detection of selfish node, selfish node handling while allocating replicas are shown in simulations. The parameters considered for simulations are shown in table2.

Table 2: Simulation Parameters

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 Nodes	1
Relocation period	64~8,192(256)
Zipf parameter	0.8

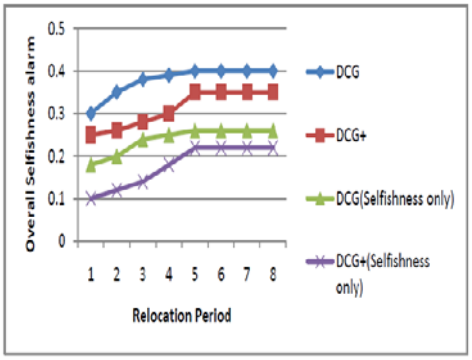


Fig 3: Relocation period Vs. overall selfishness alarm

In Fig.3. X-axis represents relocation period and Y-axis represents overall selfishness alarm. The results shows that overall selfishness alarm of DCG+ shows very less.

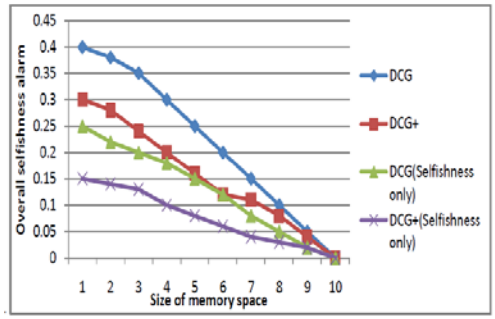


Fig4. Size of memory Vs. overall selfishness alarm

In Fig.4. X-axis represents Size of memory and Y-axis represents overall selfishness alarm. The result shows that overall selfishness alarm of DCG+ shows very less.

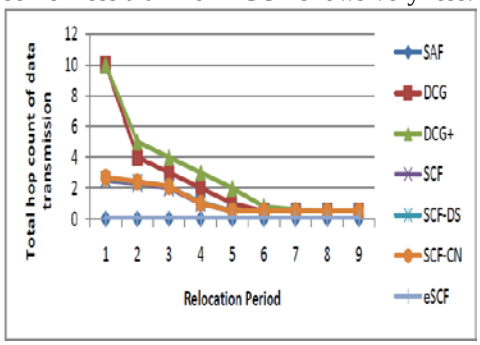


Fig 5. Varying relocation period Vs. hop count of data transmission

In Fig 5. X-axis represents Relocation period and Y-axis represents hop count of data transmission.

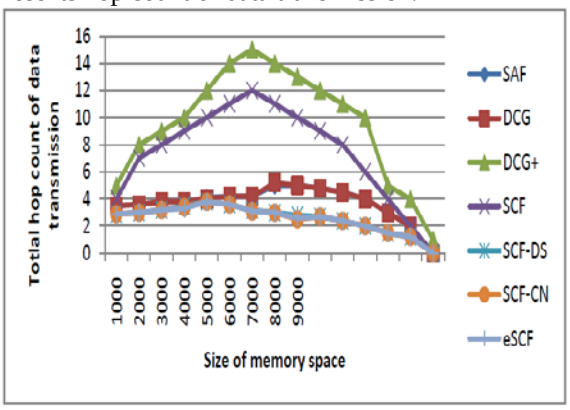


Fig 6. Varying size of memory Vs. hop count of data transmission

In Fig6. X-axis represents size of memory space and Y-axis represents hop count of data transmission. The results shows that, both size of memory and hop count of data transmission are directly proportional. Communication cost of SAF is very less when compared with other techniques.

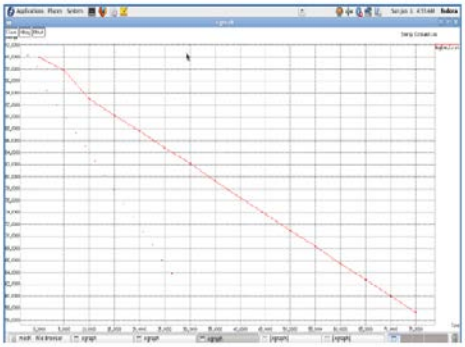


Fig 7. Time Vs. Energy Consumption

In Fig.7. X-axis represents time in milli seconds and Y-axis represents energy consumption. Time and energy are indirect proportional.

Conclusion

MANET is a network with collection of movable nodes. Some of the nodes are selfish in nature. These selfish nodes are making a problem in replica allocation. The selfish nodes are detected and handled by the algorithms mentioned. The simulation results are showing that the algorithms are capable of reducing query delay and improve the data accessibility and overall performance.

References

[1] A text book on Mobile Adhoc Networks: Current status and Future Trends by Jonathan Loo, Jaime Lloret Mauri, Jesús Hamilton Ortiz

[2] Brian B. Luu, Barry J. O'Brien, David G. Baran, and Rommie L. Hardy, "A Soldier-Robot AdHoc Network" Proceedings of the Fifth Annual IEEE International Conference on Pervasive Computing and Communications Workshops (PerComW'07)

[3] Murthy, S. and J.J. Garcia-Luna-Aceves, "An Efficient Routing Protocol for Wireless Networks", ACM Mobile Networks and App. J., Special Issue on Routing in Mobile Communication Networks, Oct. 1996, pp. 183-97

[4] Yongguang Zhang and Wenke Lee. Intrusion detection in wireless ad-hoc networks. In Mobile Computing and Networking, pages 275–283, 2000. also available as http://citeseer.nj.nec.com/zhang00intrusion.html.

[5] V.P.Sundararajan1, Dr.A.Shanmugam2, Modeling the Behavior of Selfish Forwarding Nodes to Stimulate Cooperation in MANET, International Journal of Network Security & Its Applications (IJNSA), Volume 2, Number 2, April 2010

[6] Djamel DJENOURI, NadjibBADACHE, "A Gradual Solution to Detect Selfish Nodes in Mobile Ad

hocNetworks,"Proc.14th European Conf.Research in computer networks, pp.355-370, 2009

[7] 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.

[8] Jae-Ho Choi, Kyu-Sun Shim, SangKeun Lee, and Kun-Lung Wu, Fellow, IEEE, "Handling Selfishness in Replica Allocationover a Mobile AdHoc Network", IEEE Transactions on mobilecomputing, vol.11, no. 2, Feb. 2012.

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

[10] 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.

[11] 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.