ISSN: 0975-766X
CODEN: IJPTFI
Available Online through *Research Article*
www.ijptonline.com

FAULT TOLERANT PERMISSION BASED CLUSTERED MUTUAL EXCLUSION ALGORITHM FOR MANETS

M Vanitha*[1], D Narasimhan[2]
[1]Assistant Professor, Department of CSE, SASTRA University, Kumbakonam – 612001, Tamilnadu, India
[2] Senior Assistant Professor, Department of Mathematics, SASTRA University, Kumbakonam – 612001, Tamilnadu, India.
Email: vanitha_mu@src.sastra.edu

Received on: 19.10.2016 *Accepted on: 20.11.2016*

Abstract

Resource sharing is foremost features of distributed systems. Concurrent access of a resource by more than one node may lead to inconsistency and it can be avoided by mutually exclusive access. Only few mutual exclusion algorithmsexist for mobile ad-hoc networks and they are generally classified into permission-based and token-based algorithms. High message complexity is the main drawback of those algorithms. In this paper permission based algorithm is focused subsequently the message complexity is reduced by clustering technique. Network is partitioned into clusters and a cluster head handle the mutex problem. This approach will be less efficient if the cluster head leads to failure. We proposed a virtual cluster head election algorithm to handle cluster head failure which makes our mutex algorithm as a fault tolerant one.The proposed algorithm reduces the message complexity as well as response time, synchronization delay and it is also ascertained by simulation results.

Keywords: Mutual Exclusion, Permission based algorithm, Fault tolerance, Distributed system.

1. Introduction

A Mobile ad-hoc network (MANET) is an infrastructure-less assortment of mobile devices connected in a wireless fashion. The only way for communication among nodes is message passing. In many distributed applications, a shared resource may be accessed by more than one process simultaneously and the processes are allowed to access the resource one by one to prevent conflict/corruption of data due to simultaneous access.This is called mutual exclusion (mutex). The fragment of the program which access shared resource is known as Critical Section (CS). The mutual exclusionalgorithms in distributed systems are classified into two categories: permission based andtoken based. In

permission based algorithms[1,2], each process can access the shared resource only after receivingapproval from others by exchanging messages. The drawbacksof this algorithm are each process should maintain the address of other processes from which permission to be granted and needs more message exchange. Lamport[12] was the first who discussed the permission based mutual exclusion algorithm based on logical clocks. Each node maintains a request queue in ascending order of timestamp. A timestamp is an integer value used to determine the events collation in a distributed system. If a node wants to access a shared resource, send request message to all other nodes with its timestamp and add the request to its own request queue. Requests in the queue are processed in FIFO order. This approach requires (N-1) request messages, (N-1) reply messages and (N-1) release messages, totally 3(N-1) messages. This algorithm is fair but requires more number of messages, where N is the number of nodes in the network. Ricart and Agarwala[15] algorithm reduced the total number of communications when compared to Lamport. He combined release message with reply message and it requires only 2(N-1) messages to access a shared resource. Wu et al.[13] discussed the first permission based mutex algorithm for MANETs. To reduce total number of messages he proposed look-ahead technique. This technique will consider only nodes competing for shared resource for getting permission, maintains an information set for this purpose and FIFO order is relaxed. This algorithm is fault tolerant and scalable. In token based algorithms[3,4,17], the processes are assumed to be in a logical ring, a token circulated among the processes and the process is allowed to access shared resource only after availability of a token.

The shortcomings of this algorithm are if the token is lost, it must be regenerated and if a token holding process get crashed the algorithm fails. Erciyeset al.[14] proposed a hierarchical mutex algorithm for MANETs.Nodes in the network are gathered into clusters and each cluster is coordinated by a cluster head. Members of each cluster can access shared resource after getting permission from cluster head. Each cluster head can reply to its requesting member after getting approval from its superior cluster head.

The drawbacks of this approach are failureof cluster head or cluster member. In this paper we focus on cluster head failure and the steps to be followed to elect a new clustered head. Various election algorithms like Bully algorithm[7], Ringalgorithm[8],modified Bully algorithm[9,10], Chang and Robert's algorithm[11]exist. The weakness of all existing algorithms is that it needs the involvement of all nodes and leads to more message exchange as well as increase network traffic[18]. The present paper emphases on permission based algorithm and reduction of overall communications using

clustering technique[5,6]. A new approach to select a new cluster head and the technique to reduce the message complexity and increase the performance of the algorithm are also proposed.

2. Basic Concept of Proposed Scheme

2.1. Permission-Based clustered Mutual Exclusion Algorithm

The MANET consists of 'N' independent mobile nodes. All the nodes are connected in a wireless fashion. Given network is initially grouped into clusters. Clustering can be done using Merging Clustering Algorithm (MCA) and a cluster head (CH) is elected for each clusters. Cluster Heads are further logically connected by a ring topology using Backbone Formation Algorithm (BFA)[14]. AODV is used for routing purpose.

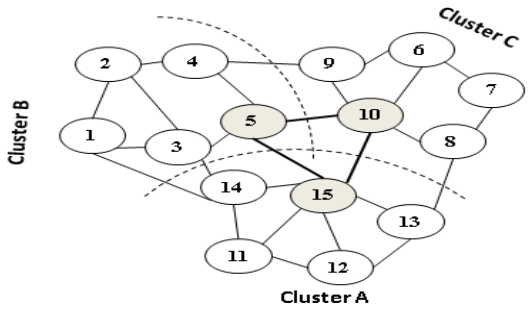

Fig. 1 MANET partitioned into Clusters

Partitioning of nodes into clusters in a Mobile ad-hoc networkare shown in Fig.1. Each node has a unique identifier, distinct timestamp and the clocks are synchronized using Lamport's logical clocks[12]. New clusters will not be formed after initialization. The given network consists of totally 15 nodes. It is divided into 3 clusters of equal size namely cluster A,B and C respectively. Each cluster contains a cluster head (CH) which is the node with highest id. Node 15 is the cluster head for cluster A, node 5 for cluster B and node 10 for cluster C. A Node who wants to access a shared resource sends a request message to only its respective cluster head along with its id, timestamp and set its status to 'waiting for reply', initially all nodes status will be set to 'idle'. The cluster head maintain a request queue to store its members request in the order of timestamp and the request are processed in FIFO order. When the CH receives a request message from its member, if it has critical section (CS) and if the timestamp in incoming message is lesser than the ones in request queue, CH send 'ok' message to its member and wait for 'release' message, set its status to 'waiting for release'. If the timestamp is not less, the request will be added to queue. If the CH doesn't contains CS it store the request in its queue, forward the request to next cluster head and wait until the reply, set its status to 'waiting for reply'. If the request message of a CH reaches itself, it indicates that no one is in CS and the CH can use the shared resource.

Whenever the cluster head receives a request message from a different cluster head, if it has critical section (CS) and if the incoming timestamp is lesser than the ones in request queue, send 'ok' message and wait for 'release' message, set its status to 'waiting for release'. If received timestamp is not less, store the request in its queue. If it doesn't contain CS, store the request in its queue, forward the request to next cluster head and wait until the reply, set its status to 'waiting for reply'.

After receiving 'ok' message from CH, a member can access shared resource, set its status to 'in CS' and after usage send a release message to CH, set status to 'idle'. If a cluster head receives 'release' message from a member, allocate the CS to next one in the waiting queue. If a CH receives ok message from another CH, send ok message to the node in the head of the queue.In this way the CH will represent its members and handle the mutex problem. The clustering approach reduces the total number of messages when compared to other mutex algorithms, since the message complexity is reduced the response time will also considerably reduced. The algorithm is fair, safe, and free from starvation, deadlock. The drawback of clustered mutex algorithm is that, if a cluster head gets failed, the request cannot be handled properly and the performance of the algorithm gets decreased. The existing algorithm to elect a leader involves all the existing process and requires more messages. We proposed a virtual cluster head election algorithm totake over the CH failure and elect a new cluster head to coordinate the members with less message complexity.

3. Proposed Technique

The prime motivation of our proposed algorithm is to guarantee that the new cluster head can be elected with less number of messages, less number of participants in the election process and also reduce the single point of failure.

The proposed fault tolerant permission based clustered mutex algorithm comprisesa virtual cluster head election algorithm to tolerate cluster head failure.

The set up consists of a Main Cluster Head (MCH), two backup heads namely Primary Cluster Head (PCH) and Secondary Cluster Head (SCH). Node with highest ID is selected as MCH and the next two highest ID nodes within the cluster will be selected as PCH, SCH. The main cluster head coordinate the cluster members i.e., handle the CS request and primary, secondary cluster heads observethe main cluster head. A virtual ID is given to cluster heads, the members will send CS request message to this ID and initially virtual ID is mapped to main cluster head. The cluster members are not aware of this virtual group.

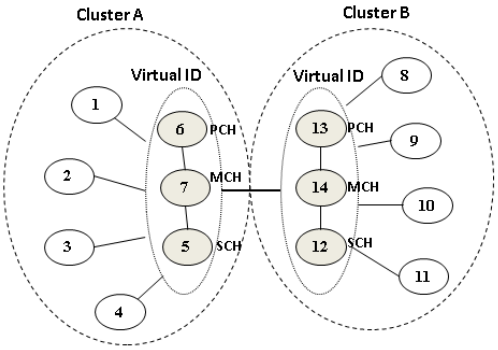

Fig. 2 Virtual Cluster Head Election Algorithm

Fig.2 shows the model of virtual cluster head election algorithm. Nodes are first grouped into cluster then cluster heads are selected. For cluster A, the highest ID node is 7 and it is selected as Main Cluster Head, next highest ID 6 is selected as Primary Cluster Head and next ID 5 is chosen as Secondary Cluster Head. A virtual ID is created and it is mapped to MCH. Node 5 and 6 continuously monitor node 7. The bold line between two clusters indicate the connection between cluster heads i.e., cluster A's virtual head ID to cluster B virtual head ID. The primary and secondary cluster head constantly monitor the main cluster head by sending 'Are You Alive (AYA)' message periodically and the MCH responds by sending 'Yes I am Alive (YIA)' message. If there is no response from MCH (best case), primary will take over the job of main cluster head, map its ID to virtual ID and intimate to secondary cluster head as a new head and it is represented in Fig.3. If the failure is detected by secondary cluster head (worst case) it will intimate to primary cluster head about main cluster head and the primary will take over further operation, denoted in Fig.4. If primary doesn't respond properly, the secondary cluster head will coordinate the members. In this way the election is done by using only three nodes in the cluster.This will reduce the participant'snumber in the election process and also the number of messages needed for declaration of a new cluster head.

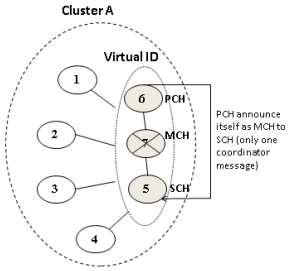

Fig. 3 Failure of Main Cluster Head noticed by Primary Cluster Head

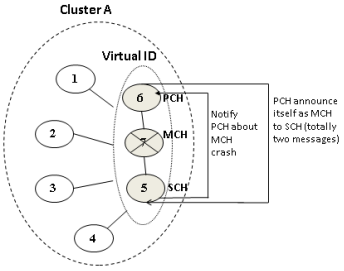

Fig. 4 Failure of Main Cluster Head noticed by Secondary Cluster Head

4. Simulation Results and Analysis

The proposed fault tolerant permission based clustered mutex algorithm has been simulated using NS2 (Network Simulator)[16] and matched with existing algorithms. From the results obtained we observed that our proposed algorithm

yields good performance metrics when compared to the existing one. The clustering technique will considerably reduce the amount of communications per CS entry and the virtual cluster head election algorithm always maintain constant number of participants. Since minimum numbers of nodes were participated in the election process, the total number of election message and coordinator message also gets reduced. The response time and synchronization delay will be decreased when the number of nodes gets increased.

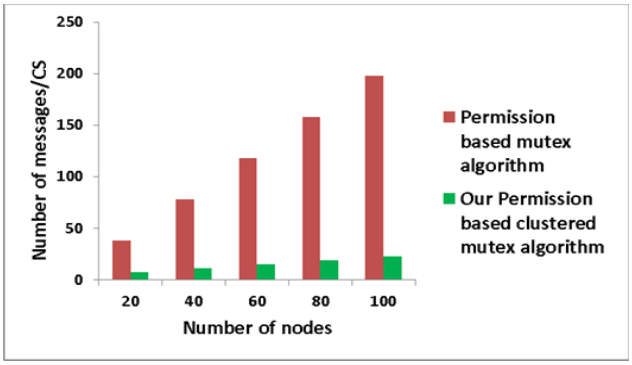

Fig. 5 Total number of messages per CS entry Vs Number of nodes

Fig.5 shows the total amount of communications required per Critical Section entry of Permission based mutex algorithm and our proposed one. In Permission based 2(N-1) number of messages will be used where as our proposed permission based clustered mutex algorithm requires only K+3D messages where N represents the total nodes, K represents the total clusters, it depends upon cluster size and D represents diameter of the cluster. Scalability of the graph: cluster size 5 and diameter one (since the algorithm assumes the mesh topology inside the cluster).

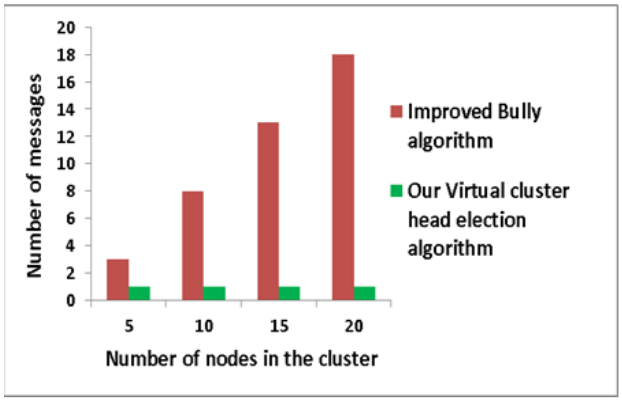

Fig. 6 Total number of messages Vs Number of nodes for Best Case Scenario of election algorithm

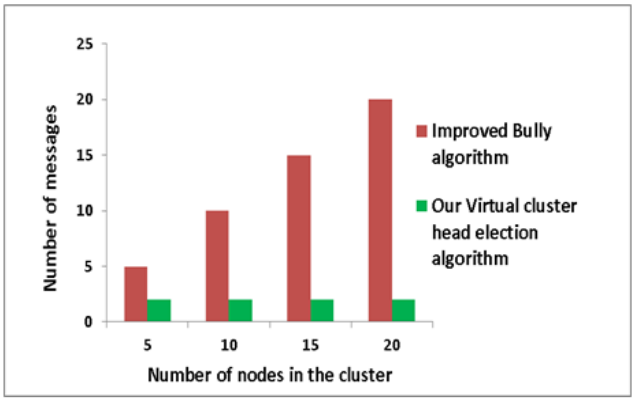

Fig. 7 Total number of messages Vs Number of nodes for Worst Case Scenario of election algorithm

Fig.6 and Fig.7 represents best case and worst case scenario of the total number of messages exchanged for conducting election as well as declaration of coordinator in Improved Bully algorithm and our proposed virtual cluster head election algorithm. In Bully algorithm N-2 and N messages are required for Best case and Worst case respectively. Thealgorithm proposed requires only one message in best case and two messages in worst case.

5. Conclusion

A fault tolerant permission based clustered mutex algorithm for Mobile Ad-hoc networks has been proposed. Data or resource sharing is one of the centralobjectives of distributed systems. When more than one distributed nodes tries to access a common resource it leads to inconsistency. To prevent this so many mutex algorithm exists. The proposed algorithm considerably reduced the overall communication required per mutually exclusive access and also has considerable response time, synchronization delay when compared to existing ones. We first include clustering approach to reduce the message complexity and since cluster head plays a vital role, it may leads to failure. To handle cluster head failure a virtual cluster head election algorithm has been proposed. It reduces the number of participants in the election process so that only constant numbers of messages are required for conducting election and declaration of new cluster head. Simulation results are shown in the graphs and it is proved that the algorithm proposed is considerably reduced the message complexity. Moreover the proposed algorithm can be used for any type of networks.

References

1. A.S. Tanenbaum and M.V. Steen, "Distributed systems: principlesand paradigms," 2nd edition, Prentice Hall of India, 2006.
2. Anchal, C. Ramakrishna, P. Saini, "A survey on permission based distributed mutual exclusion algorithms in mobile ad-hoc networks," International Conference on Computer Network and Information Technology, 20-21 March, 2014.
3. F. Mueller,"Fault tolerance for token-based synchronizationprotocols," Workshop on Fault-Tolerant Parallel and DistributedSystems, IEEE, April 2001.
4. Sung-Hoon Park, Seoun-Hyung Lee, "Token-Based Mutual Exclusion Algorithm in Mobile Cellular Networks,"International Conference on Advanced Information Networking and Applications Workshops, 2012.
5. Garcia-Molina,"Elections in a distributed computingsystem," IEEE Transactions on Computers, January 1982.
6. Greg, N. F., Nancy A. L., "Electing a leader in a synchronousring," Journal of ACM, 1987.
7. Candido Caballero-Gil, Pino Caballero-Gil, and JezabelMolina-Gil, "Self-Organized Clustering Architecture for Vehicular Ad Hoc Networks," International Journal of Distributed Sensor Networks, Volume 2015, Article ID 384869.

8. Anchal, Poonam Saini and C. RamaKrishna, "An Efficient Permission-Cum-Cluster BasedDistributed Mutual Exclusion Algorithm for MobileAdhoc Networks," International Conference on Parallel, Distributed and Grid Computing, 2014.
9. Mamun, Q.E.K.; Masum, S.M.; Mustafa, M.A.R.,"ModifiedBully Algorithm for ElectingCoordinator in DistributedSystems," International Conference on Software Engineering,Parallel and Distributed Systems, 2004.
10. Basim, A.,Laith, H.B, Mohammad, A., "Reducing MassagePassing and Time Complexity in Bully Election AlgorithmsUsing Two Successors," International Journal of Electronicsand Electrical Engineering, March 2013.
11. Chang, E., Roberts, R., "An improved algorithm fordecentralized extrema-finding in circular configurations ofprocesses," Communications of the ACM, 1979.
12. Lamport, "Time, clocks, and the ordering of events in a distributed system," Communicationsof the ACM, vol. 21, no. 7, pp. 558-565, 1978.
13. W. Wu, J. Cao, and J. Yang, "A scalable mutual exclusion algorithmfor mobile ad hocnetworks," Proceedings of 14th IEEE InternationalConference on Computer Communications and Networks, pp. 165-170, November 2005.
14. K. Erciyes and O. Dagdeviren, "A distributed mutual exclusionalgorithm for mobile ad hoc networks," International Journal ofComputer Networks and Communications, vol. 4, no. 2, pp. 129–148, March 2012.
15. G. Ricart and A. K. Agrawala, "An optimal algorithm for mutualexclusion in computer networks," Communications of the ACM, vol.24, no. 1, pp. 9–17, January 1981.
16. NS2 simulator, http://www.nsnam.org.
17. B. Gopinathan, S.Subramanian and R.Nedunchezhian, " Efficient and Robust Token based DME Algorithm in MANET," Indian Journal of Science and Technology, Vol.8(36), DOI:10.17485/ijst/2015/v8i36/83183, December 2015.
18. SungHoon Park, "A Safe Election Protocol based on an UnreliableFailure Detector in Distributed Systems", Indian Journal of Science and Technology, Vol.8(34),DOI: 10.17485/ijst/2015/v8i34/86665, December 2015.