World Academy of Science, Engineering and Technology
International Journal of Electronics and Communication Engineering
Vol:4, No:4, 2010

An Energy Efficient Algorithm for Distributed Mutual Exclusion in Mobile Ad-hoc Networks

Sayani Sil and Sukanta Das

***Abstract*—This paper reports a distributed mutual exclusion algorithm for mobile Ad-hoc networks. The network is clustered hierarchically. The proposed algorithm considers the clustered network as a logical tree and develops a token passing scheme to get the mutual exclusion. The performance analysis and simulation results show that its message requirement is optimal, and thus the algorithm is energy efficient.**

***Keywords*-Critical section, Distributed mutual exclusion, Mobile Ad-hoc network, Token-based algorithms.**

I. Introduction

MUTUAL exclusion ensures that only one process can access shared resources at a time. If at that time, other process requests for those shared resources, then the requesting process has to wait until the resources have been released.

A distributed system consists of a collection of independent geographically dispersed autonomous computers appears to its users as a single coherent system which are connected by a communication network. Global or centralized controller is absent in this network. Segments of code to be executed mutually exclusively are referred to as critical sections (or critical regions) in the literature [4],[7],[12],[9]. A mobile ad-hoc network is a collection of mobile nodes that communicate through wireless link, and therefore, the topology of underlying network can dynamically be changed. In such mobile ad-hoc network (MANET), the complexity of mutual exclusion problem is much high since no static topology can be considered.

In this paper we have proposed a new energy efficient mutual exclusion algorithm for mobile ad-hoc network. The whole network is hierarchically clustered to get a logical tree structure of the network. The proposed algorithm is token based and works on the logical tree to obtain the mutual exclusion. We have addressed the issue of mobility of nodes extensively. Utilizing the hierarchical structure of the network, the proposed algorithm significantly reduces the message requirement per mutual exclusion. Therefore, overall energy requirement for message communication of the system is optimized.

The rest of the paper is organized as follows. *Section II* reports the survey on distributed mutual exclusion algorithms. *Section III* notes the properties of the system for which the algorithm will be proposed. The algorithm is proposed in *Section IV*. The performance of the proposed algorithm is analyzed in *Section V*.

The authors are with the department of Information Technology, Bengal Engineering and Science University, Shibpur, West Bengal, India, emails: sayanisil.02@gmail.com, sukanta@it.becs.ac.in

II. Distributed mutual exclusion algorithms

The mutual exclusion algorithms for distributed systems can broadly be classified into two categories according to their algorithmic principles - permission based algorithms and token based algorithms. The permission based algorithms (e.g. [1], [14], [13], [21] etc.) require two (or more) successive rounds of message exchanges among the sites. A process can enter in the critical section only after receiving permission from other process(es) in the system. Lamport's algorithm, requires approximately $3(n-1)$ messages per mutual exclusion invocation [10]. An algorithm, proposed by Ricart and Agrawala [1] for this problem, requires $2(n-1)$ messages per CS entry. In this algorithm only after receiving permission from all the other processes in the network one process can enter into the CS. Mamoru Maekawa proposed a distributed mutual exclusion algorithm [14] which requires only $3\sqrt{n}$ to $5\sqrt{n}$ messages per mutual exclusion. On the other hand, in token based mutual exclusion algorithms a unique token or a privilege message is shared among the nodes. Token gives the authority to a node to execute the CS. Suzuki and Kasami's token based algorithm, based on the concept of node privilege, requires n messages per CS entry [22]. In Raymonds algorithm, the nodes are arranged in an un-rooted tree structure [18] which normally is a minimal spanning tree of the network. The message complexity of the algorithm is $O(\log n)$ under light demand. Algorithm proposed by Pranay Chaudhuri, Mehmet Hakan Karaata [16] achieves the message complexity as $O(n^{1/3})$ per mutual exclusion. They assumed that the n node network is available in the form of a three-dimensional mesh.

However, all the algorithms, reported above, are developed for static networks. Those algorithms can not be efficiently utilized for MANET. B. R. Badrinath, Arup Acharya first proposed [3] a mutual exclusion algorithm for cellular network. In this algorithm, base station acts as a proxy for nodes attached to it and the request queue is maintained only by the base station. The base stations are arranged in a logical ring. J. Walter and S. Kini's algorithm is a token based algorithm [23] for ad-hoc networks. The algorithm defines the Direct Acyclic Graph (DAG) to map the physical topology of network. The algorithm maintains multiple path leading to the token holding node through DAG. They considered that the mobility of the nodes are slow and they does not considered the token loss. J. Walter, J. Welch and Vaidya assumed that communication channels are FIFO with no loss [8]. Each node dynamically chooses their neighbor with lowest elevation as its preferred next link to the token holder. This algorithm also acts with the same assumptions as [23]. R. Bladoni, A. Virgillito's algorithm

[2] is based on a dynamic logical ring and combines the two methods of token asking and token circulating. The algorithm reduces the power consumption by reducing the number of hops traversed per CS execution. This algorithm needs n number of message per CS entry under light load. N. Malpani, N. H. Vaidya proposed a parametric algorithm [15] with many variants. It uses dynamic logical ring and the size of the ring may vary at every round. The main idea of the algorithm is the method of choosing the next node to which the token will be sent. The variant's policies applied to determine the next node. The algorithms in [2] and [15] are not aware of the nodes mobility. Romain Mellier, Jean-Frederic Myoupo presented a token based mutual exclusion algorithm for multi hop mobile network and it needs $O(n)$ broadcast rounds [19]. Ranganath Atreya, Neeraj Mittal's algorithm [17] achieves the message complexity of $O(q)$ and a bit-message complexity of $O(bqr)$, where q is the maximum size of a quorum, b is the maximum number of processes from which a node can receive critical section requests, and r is the maximum size of a request while maintaining both synchronization delay and waiting time at two message hops.

In this paper, we also present an algorithm for MANET. We extensively address the issue of mobility in our work. Next we report the properties of the system under consideration, for which the algorithm is developed.

III. The properties of system under consideration

This section reports the properties of the system. We have few assumptions regarding the system. Moreover, the whole network is hierarchically clustered.

A. Assumptions

The system contains a number of independent nodes. The nodes are mobile and they communicate with each other only through message passing. Here we assume some characteristics of the system in which we run our algorithm.

Assumption 1: Communication is reliable, i.e., there is no message loss during transmission and the receiving node can receive message without any error or distortion.

Assumption 2: The message passing from one node to another node is time bounded and the time is negligible.

Assumption 3: The channel is a FIFO channel i.e, message passed through the channel in a first come first serve manner and the communication links are bidirectional links.

B. Clustering

For the execution of the proposed algorithm, the network is to be clustered hierarchically. We have utilized the hierarchical clustering technique proposed in [24],[20],[6],[11],[5]. The proposed algorithm is independent from clustering algorithms, but the message requirement per mutual exclusion may slightly vary with the choice of clustering algorithm. In this work, we have considered two level hierarchical clustering to get the best performance - level 1 consists of the cluster heads, and one cluster head is chosen as level 2 cluster head. Considering level 2 cluster head as root, the whole network looks like a

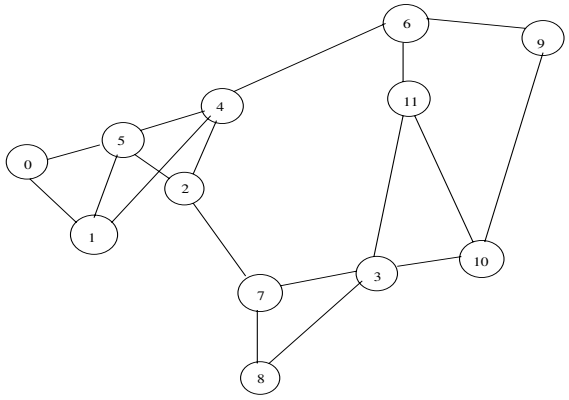

Fig. 1. A MANET with 12 nodes

logical tree. We consider, the nodes in a cluster can directly communicate with the cluster head, and it requires at most p messages for communication between the root and a cluster head. The performance of the algorithm will be better if p is low.

Consider the network of *Fig.1*. Suppose, node 3, 5 and 6 are selected cluster heads (level 1). Now, each cluster head advertises itself as a cluster head within its transmission range. As a result, 3 cluster will be formed - (3, 7, 8, 10), (5, 0, 1, 2, 4) and (6, 9, 11). Among the level 1 cluster heads (node 3, 5 and 6), node 6 is chosen as a level 2 cluster head (root). Here, the cluster heads 3 and 5 require two messages to communicate with the root (node 6). The tree like structure of *Fig.1* is shown in *Fig.2*.

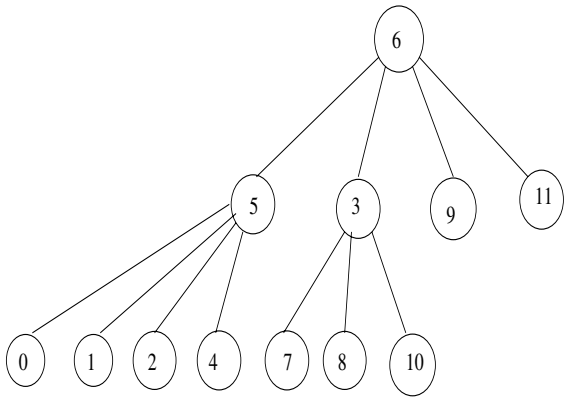

Fig. 2. Logical tree structure of the 12 node MANET

IV. The Algorithm

This section reports the distributed mutual exclusion algorithm in detail. The algorithm requires a number of data structure and control messages. We next report such requirements.

A. Data structure and control messages

Pointers: Each node maintains a pointer which either points to its higher level node or lower level node, if any. The pointer actually provides the direction where from the token can be received. Each non-cluster head member points to its cluster

head as the node can get the token from its cluster head only. On the other hand, a cluster head either points to a member of the cluster if the member is having the token, or points to the root.

Queue: Each node maintains a queue to store the request messages coming to the node or generated by the node for the token.

Request: A request message contains a special bit pattern and a node number.

Token: It is a control message which has a special bit sequence and a node ID. If the node ID is not null, then the token is carrying a *dummy request*. The dummy request(discuss latter) contains a node ID.

B. Overview of the scheme

The proposed algorithm is a token based algorithm. A node can execute the critical section (CS) if it gets the token. For simplicity we consider that the number of CS is 1. Since there is a single token in the network, only one node can execute the CS.

Each node generates a *Request* message while it wants to execute the CS. The hierarchical structure (tree) of the network helps a node to get the token. Each node points to some other node in the tree where from the node can get the token. The *Request* message is forwarded to the pointed node. While a node receives a request message, it forwards the message to some other node through the edges of tree if the node does not have the token. Otherwise, the node returns back the token if it has free token. Each node enqueues the request, including the self request, if the node can not immediately grant the request by returning the token. The node dequeues the request if token is passed to the requesting node. While the token is passed, the pointers are organized accordingly to point the new token location. After getting the token, the requesting node executes the CS, and after completion of execution the token remains with the node. In the proposed scheme, the token is not circulated if it is not requested.

To minimize the number of messages, the algorithm does not always forward the request message. If a node A has already requested the token and during the waiting time another node makes a request to A, then A enqueues but does not forward the request.

Consider, a non-cluster head node, say A generates a request message. The node A forwards the request to its cluster head, say C. If C has the free token, it immediately returns back token to A. If C has token but not free, then the request is queued up. Otherwise, the token can be received either from any non-cluster head member of that cluster, or from the root. The node C can get such direction from the pointer. The request is forwarded according to the content of the pointer. We utilize the concept of *dummy request* in our algorithm.

Dummy Request: To avoid the starvation and to maintain the fairness of the scheme, we introduce the concept of dummy request. While the token is passed to some other node, the node checks whether queue is empty or not. If queue is not empty, the node adds a dummy request in the token. When a node receives a token with dummy request, then the node pushes a new request in its queue considering the sending node as the requesting node. The token message, therefore, has a provision to add a dummy request.

Dummy request node ID - If one node needs to send dummy request then it put its own ID in the token message otherwise this value will be ”NULL”.

The following example illustrates the execution of the algorithm.

Example 1: Consider the network of *Fig.1* and the corresponding logical structure shown in *Fig.2*. Here we assume that the number of CS is 1 and the execution time of CS is considered as unit time. We also assume that the token is with node 0 and at time 0.51, the node makes a request and starts the execution of CS. During the execution, the nodes 7, 8 and 5 make requests at time 0.56 unit, 0.67 unit and 0.78 unit respectively. However, the token is released by node 0 at time 1.51 unit. Therefore, the requests are to be enqueued. The scenario is depicted in *Fig.3*.

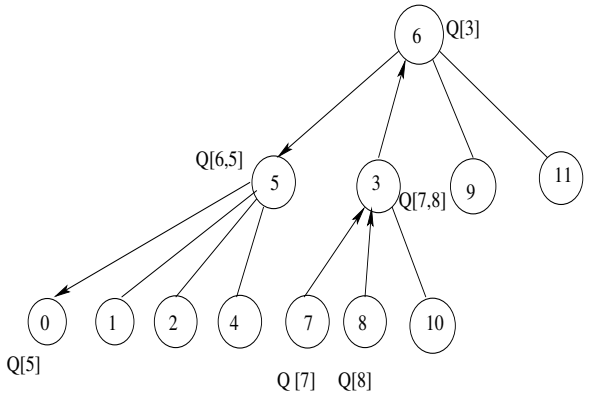

Fig. 3. Structure of tree after requests of node 7, 8 and 5

At time 0.56, node 7 sends a request to its cluster head node 3 and enqueues the request in its queue. Node 3 knows that the token is not held by any member of its cluster, so node 3 sends a request to the root, that is, node 6. Node 6 knows that it can get the token from node 5. So node 6 forwards the request to node 5 and enqueues the request as the request is made by node 3. Node 5 knows that the token originally hold by node 0 so it sends a request message to node 0 and enqueues the request of node 6. The node 0 enqueues the request of node 5.

Now at 0.67 unit time node 8 sends a request message to its cluster head node 3 and enqueues its own request in its queue. Node 3 enqueues the request of node 8 and it knows that it already sends a request message to node 6 for the token. So it does not make any further request to node 6.

At 0.78 unit time node 5 want to execute the critical section it en-queues its own request in its queue. Node 5 knows that it already send a request message to node 0 so it does not send any request message to node 0.

After completion of execution of node 0, it sends the token to node 5. Node 5 then dequeues a node from its queue. It is node 6, and after dequeue the queue is still non-empty. Then node 5 sends the token to node 6 with a dummy request. Node 6 will send the token to node 3 and also sends a dummy request

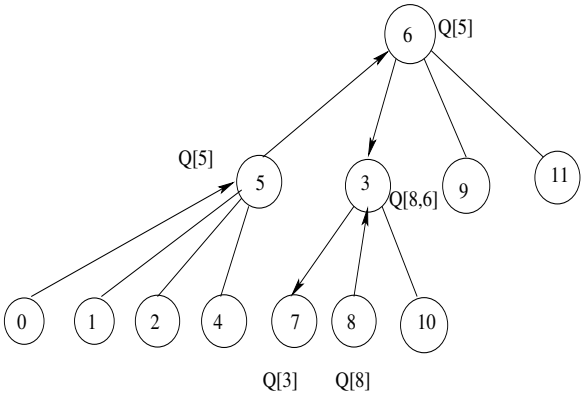

Fig. 4. Token is passed to node 7

with the token for the above reason. Node 3 sends the token to the first requesting node 7 with the dummy request. After receiving the token node 7 will start execution of the critical section. The scenario is depicted in *Fig.4*.

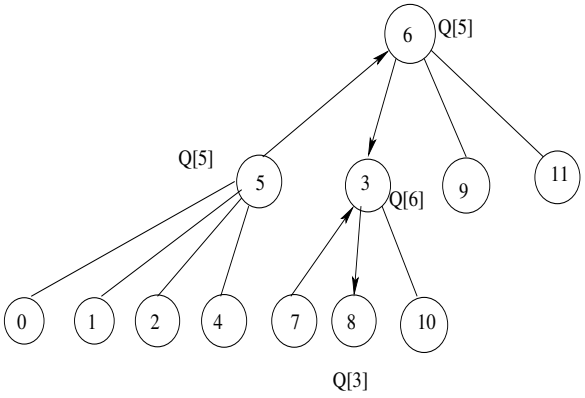

Fig. 5. Token is passed to node 8

After completion of execution of the node 7 it sends back the token to the node 3 without dummy request. Node 3 sends the token to the node 8 with a dummy request. After receiving the token node 8 will start execution of the critical section (*Fig.5*).

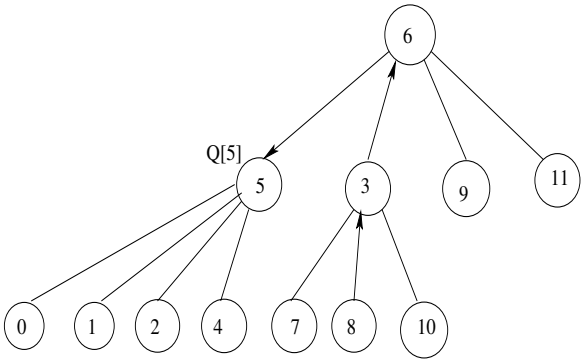

Fig. 6. Token is passed to node 5

After completion of execution of the node 8 it sends the token to the node 3. Node 3 sends the token to the first requesting node 6. Node 6 sends the token back to node 5. After receiving the token node 5 starts execution of the critical section. The scenario is noted in *Fig.6*.

C. Mobility

The mobility of nodes is an important property of a MANET. The proposed algorithm can tolerate the mobility in various ways. The following scenario may arise due to the node's movement.

1. A node moves within the cluster: It does not affect the proposed scheme as the logical tree structure of the network remains unchanged.

2. A node moves from one cluster to another cluster: In this case, three different actions are to be taken based on the node's type - whether the node is non-cluster head node, cluster head or root.

a. The node is non-cluster head node: If any non-cluster head node moves then that node will be considered as a new node and it will again send a request message to its new cluster head for the token. If that non-cluster head node holds the token, then the token as well as the queue will be destroyed. The corresponding cluster head will regenerate the token.

b. It is cluster head: If any cluster head moves to another cluster then a new cluster head will be selected from the non-cluster head nodes of its cluster. Other nodes will send the request message to the newly selected cluster head. The old cluster head deletes its whole queue. If the new cluster head or any of its member does not have the token then it will send the information to the root. Otherwise any member of the cluster which holds the token will send that information to its cluster head.

c. The node is root: If root moves to the another cluster then a new root will be selected from the cluster heads. The remaining cluster head will send a request message to the newly selected root. The cluster having the token will send this token information to the newly selected root. In general, if any node changes its cluster then it deletes its previous queue and if the node holds the token then it will also destroy the token.

4. A node is added in the network: The node will be under some cluster. A node will be added in the logical structure.

5. A node is departed from the network or goes down: A node will simply be deleted from the tree. If the node is having the token, the corresponding cluster head will regenerate the token.

D. Formal description

Here we formally present the algorithm. The algorithm assumes the network hierarchically clustered and the network looks like a logical tree. Here we consider only two levels of hierarchy. The steps of the algorithm are presented below.

Step 1: If a node wants to execute the critical section, then
If the node contains the token, it executes the CS.
Otherwise, form a request message, enqueue the request and

the pass the message to the pointed node.

Step 2: While a node receives the request message:

1) If the node has free token, the token is returned back.
2) Otherwise, the request is enqueued.
3) If the node has not made any request for the token, the request is forwarded to the pointed node considering the node as the requesting node.

Step 3: A node is having the token:

1) If token is received from other node and token is having the dummy request, the request is queued up.
2) If the queue is empty, the token remains with the node.
3) Otherwise, one element, say q is dequeued from the queue.
4) If q points to the node itself as the requesting node, the node execute the CS.
5) Otherwise, the token is sent back to requesting node as mentioned in q.
6) If the queue is still non-empty after dequeue, a dummy request is added with the token.

V. Performance analysis

The performance of the proposed algorithm is measured by the message requirement per mutual exclusion, as the message requirement directly determines the energy requirement. In best case, the node that wants to execute the CS is having the token. In that case, no message is exchanged. However, the worst case of our algorithm is - a non-cluster head node, say A requests for the token and the request is routed through root to some other non-cluster head node, say B, having the token. Obviously, this is a rear case in medium or high load, because according to our algorithm, the request of A does not reach to B if the cluster head of A or the root or the cluster head of B has already requested the token. Therefore, the worst case can occur in very low load only.

Consider, it requires at most p messages for the communication between a cluster head and the root. So, in worst case at most $2 + 2p$ messages are required for the request - 1 for A to the cluster head of A, p for the cluster head of A to the root, another p for the root to the cluster head of B, and 1 for cluster head of B to B. To pass the token to A from B, another $2 + 2p$ messages are required. Therefore, at most $4 + 4p$ messages are required per critical section entry. Here, the number of messages varies with p. The selection of root from among the cluster heads determines the value of p.

We have simulated the whole environment. A number of arbitrary networks are synthesized with the consideration that each node can have at most 4 neighbors. We assume, the number of CS is 1 and arrival pattern of requests for CS in the system follows the Poisson distribution. We consider the execution time of the CS as the unit of time for our simulation. The simulation result is reported in *Table I*. The first column reports the number of nodes in the MANET, where as the second column notes the the number of nodes that make a request for critical section entry. The last three columns depict the number of messages required per mutual exclusion. λ denotes the number of requests arrived in the system in unit time. We experimented with 100 different networks for same parameters, and average results are reported in the table. For the whole simulation, we have considered $p = 2$. The reported results show that the message requirement is almost same for different load.

TABLE I
Simulation results

# nodes	# nodes execute CS	Avg. no. of messages		
		Low load $\lambda = 0.5$	Medium load $\lambda = 1$	High load $\lambda = 2$
60	10	9.25	9.54	9.17
85	10	9.93	9.02	9.63
96	10	9.40	9.71	9.10
100	10	9.85	9.22	10.01
60	15	9.35	9.72	9.56
85	15	9.93	8.72	9.05
96	15	9.13	9.11	10.05
100	15	10.00	9.98	9.95

VI. Conclusion

In this paper, we have presented a token based mutual exclusion algorithm for mobile Ad-hoc network. The message complexity of the algorithm in worst case is $4+4p$, where p is the maximum message required for communication between a cluster head and the root.

References

[1] G.Ricart And A. K Agrawala. An optimal algorithm for mutual exclusion in computer networks. In *Commun.ACM*, Jan. 1981.
[2] R. Baldoni and A. Virgilito. A token-based mutual exclusion algorithm for mobile ad-hoc networks. In *Dipartimento di Informatica e Sistemistica, Universita di Roma La Sapienza, Viasalaria 113, 00198 Roma, Italia, Technical Report 28-01*.
[3] A. Acharya B.R.Badrinath and T. Imielinski. Structuring distributed algorithms for mobile hosts ,. In *Proc. of the 14th Intern. Conf. on Distr. Comp.*, 1994.
[4] P.J. Denning E.G. Co.man Jr. Operating systems theory, prentice-hall, new york. 1973.
[5] Mohsen Tolou Honary Hamid Shokrzadeh Farzad Tashtarian, A. T. Haghighat. A new energy-efficient clustering algorithm for wireless sensor networks. In *IEEE*, 2007.
[6] A. Arora H. Zhang. scalable self-configuration and self-healing in wireless networks. In *in: Proceedings of the 21st ACM Symposium on Principles of Distributed Computing (PODC 2002)*, July 2002.
[7] P.B. Hansen. Operating system concepts, prentice hall, new york,. 1973.
[8] J. Welch J. E. Walter and N.Vaidya. A mutual exclusion algorithm for mobile ad-hoc networks. In *accepted to the ACM and wareless networks journal special issue on Dialm papers*, 2001.
[9] A. Silberschatz L. Peterson. Operating systems concepts, addson-wesley, new york. 1986.
[10] L. Lamport. Time, clocks, and the ordering of events in a distributed system,. In *Communications of the ACM 21,*, 1978.
[11] Mohd Fadlee A. Rasid M. Hossein Fotouhi Ghazvini, Maryam Vahabi and Raja Syamsul Azmir Raja Abdullah. Optimizing energy consumption in hierarchical clustering algorithm for wireless sensor networks. In *IEEE*, 2007.
[12] R. Oldehoeft M. Maekawa, A.E. Oldehoeft. Operating systems: Advanced concepts, the benjamin/cummings, menlo park, ca,. 1987.
[13] Hsiou Mien Lien amd Shyan-Ming Yuan. A new approach of coiistructiiig informatioii structure for mutual exclusioii in distributed systems. In *IEEE*, 1994.
[14] M.Maekawa. A $\sqrt{N}$ algorithm for mutual exclusion in decentralised system. In *ACM Trans.Comput.Syst.*, May 1985.
[15] J. L. Welch N. Malpani, N. H. Vaidya. Distributed token circulation on mobile ad-hoc networks ,. In *Technical report, Intel Corporation 505 E. Huntland Dr. Suit 550, Austin TX 78752*.
[16] Mehmet Hakan Karaata Pranay Chaudhuri. An $o(n^{1/3})$ algorithm for distributed mutual exclusion. In *Journal of Systems Architecture*, 1998.

[17] Neeraj Mittal Ranganath Atreya. A quorum-based group mutual exclusion algorithm for a distributed system with dynamic group set. In *IEEE Transactions on parallel and distributed systems, Vol. 18, No. 10*, OCT 2007.

[18] KERRY RAYMOND. A tree-based algorithm for distributed mutual exclusion. In *ACM Transactions on Computer Systems, Vol. 7, No. 1,*, February 1989,.

[19] Jean-Frederic Myoupo Romain Mellier. A clustering mutual exclusion protocol for multi-hop mobile ad hoc networks. In *IEEE International Conference on Networks (ICON 2005), pp. 250-255, IEEE Press*, 2005.

[20] E. Coyle S. Bandyopadhyay. An energy efficient hierarchical clustering algorithm for wireless sensor networks. In *in: Proceedings of the 22nd Annual Joint Conference of the IEEE Computer and Communications Societies (INFOCOM 2003), San Francisco, California*, April 2003.

[21] Sandeep Lodha and Ajay Kshemkalyani. A fair distributed mutual exclusion algorithm. In *IEEE Transactions on Parallel and Distributed Systems, vol. 11*, 2000.

[22] I. Suzuki and T. Kasami. A distributed mutual exclusion algorithm. In *ACM TOCS*, 1985.

[23] J.E. Walter and S. Kini. Mutual exclusion on multihop, mobile wireless networks. In *Texas A and M Univ,College Section, TX 77843-3112, TR97-014*, Dec 9 1997.

[24] H. Balakrishnan W.B. Heinzelman, A.P. Chandrakasan. Application specific protocol architecture for wireless microsensor networks. In *IEEE Transactions on Wireless Networking*, 2002.