A Modified Leader Election Algorithm for MANET

Smita Bhoir
Computer Engineering Department
Ramrao Adik Institute of Technology
Mumbai, India
smitapatilbe@gmail.com

Amarsinh Vidhate
Computer Engineering Department
Ramrao Adik Institute of Technology
Mumbai, India
vidhate.amarsinh@gmail.com

***Abstract*— Distributed systems are the backbone of modern day computing services. A mobile ad hoc network (MANET) is a collection of mobile nodes that can communicate via message passing over wireless links. Communication takes place directly between nodes which are in transmission range of each other else communication is done through message relay. A MANET is different from existing distributed network because of their concurrent and frequently changing wireless link formation and removal, network partitioning and disconnections, limited bandwidth and energy and highly variable message delay. An election algorithm elects a leader to coordinate and organize a task in distributed systems that includes MANET also. In the case of a leader node departure or failure, nodes detecting the non-availability of the leader initiate a leader election process to select a new leader. This paper presents a comparative analysis of various leader election algorithms and a new leader election algorithm in analytical way which considers factors such as node's position, time complexity, message complexity, battery life and security.**

Keywords- Distributed System, Mobile Ad Hoc Networks, Leader Election

I. INTRODUCTION

The distributed system, collection of independent computers located at different geographical area, started the journey with a dream that there would be no centralized controller to operate the system. The individual computers interact with each other through only message passing and have equal responsibility to manage the system. The system appears to its users as a single coherent system. However it was identified soon that with time technology, even with today's technology the dream could not be fulfilled. The distributed algorithms, a new set of algorithms, which are different from the classical concept, were required to run different applications in such distributed environment. The people thought to assign an extra load to any one of the independent computers for the coordination of the system. The particular computer was referred a coordinator or leader. Introduction of the concept of the leader in a distributed system was a betray to the dream, but it was a practical requirement to utilize the already developed centralized algorithms in distributed environment. Leader in distributed systems handle mutual exclusion, act as coordinator, initiator, and perform some specific role, such as directory server, token regenerator. A mobile ad hoc network (MANET) is a collection of mobile nodes that can communicate via message passing over wireless links. Communication takes place directly between nodes which are in transmission range of each other else communication is done through message relay. A MANET is different from existing distributed networks because of their concurrent and frequently changing wireless link formation and removal, network partitioning and disconnections, limited bandwidth and energy and highly variable message delay. An election algorithm elects a leader to coordinate and organize task in distributed systems that include MANET. In contrast to traditional distributed systems, nodes arrive and leave MANET more frequently. In the case of a leader node departure or failure, nodes detecting the non-availability of the leader initiate a leader election process to select a new leader. A leader election is a primary control problem in both wireless and wired systems. In wireless network, leader has a variety of applications routing coordination, sensor coordination and general control.

A leader election problem was first posed by Le Laan [1], and developed a solution. Leader election problem is a task to elect a node or a process in particular, uniquely a leader of the system. A number of leader election algorithms have been proposed for classical distributed system [2], [3], [4], [5]. However, with the advent of wireless communication technology, the concept of classical computing has been changed due to introduction of mobility in the nodes or computers. In this paper, we analyze ten leader election algorithms applied in MANET using common factors such as time complexity, message complexity, assumptions considered, timing model, basis for leader election and consideration of security factor, etc. Our analysis proposes enhancements such as maximum utilization of nodes during election to reduce load. It also proposes reduction of a costlier new election when a leader wakes up after crashing. It simply inquires to its functioning neighbors to identify its current leader. It uses better election criteria such as battery life, mobility and security. It also proposes a new leader election algorithm. The organization of paper is as follows: section II describes the literature survey, section III provides discussion, analysis and comparison, section IV provides proposed algorithm and section V gives experimental results. Finally in section VI, we present our concluding remarks and future works.

II. Related Work

A. Design and Analysis of a Leader Election Algorithm for Mobile Ad Hoc Networks [8]

This paper proposed a leader election algorithm for mobile Ad-hoc network. This algorithm is based on diffusion computation and extrema finding. It is designed to handle dynamic topological changes. As election begins, the node starts a diffusing computation to find out its new leader. When a number of nodes identify that leader got crashed or leader is exiting, they start diffusing process. At a time number of diffusion processes can be carried out. But a node can participate in only one diffusing process at a time. Diffusion computation will search for a node with highest value, which is determined by the node's Unique Identity (UID). Fig. 1, 2 shows a typical diffusion and shrinkage. This algorithm works in an asynchronous environment with channels that are not FIFO. When any node detects that there is no leader, the algorithm begins. A detector node sends an 'ELECTION' message that it spreads to its entire neighbor which spreads like spanning tree. The other nodes, upon the arrival of the 'ELECTION' message, forward it to their children and declare the sender as their parent. When the source receives an acknowledgement, it spreads a 'LEADER' message to the entire newly constructed spanning tree with the information of the current new leader. But if the number of nodes is more, the time required will be more. If we consider mobility, it increases time required for election.

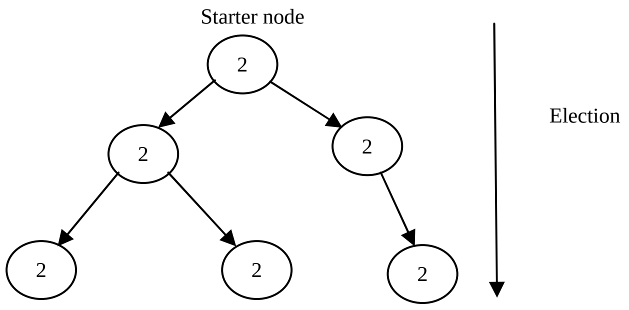


Fig.1 Diffusion of tree

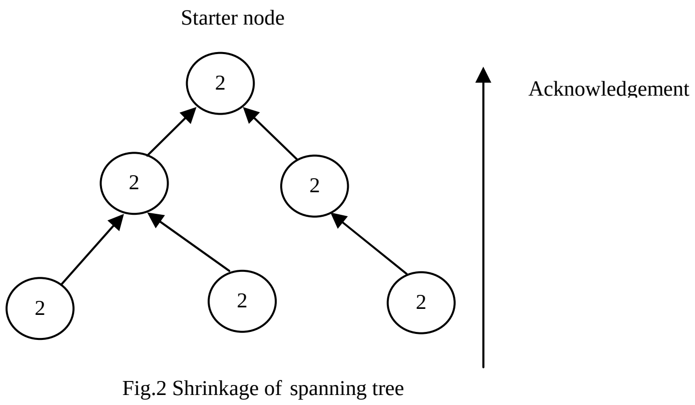


Fig.2 Shrinkage of spanning tree

B. Research of Asynchronous Leader Election Algorithm on Hierarchy Ad hoc Network [9]

Initially, the network establishes a hierarchy within election group. When a node or several nodes detect the loss of a leader, each waits for a given time-out. After the time-out, the node sends an ELECTION message. The other node, upon the receipt of the message, forwards it to its unvisited neighbors with highest priority. Nodes that have received the ELECTION message are marked. This avoids duplicate messages. When a node has no neighbors, it resends the ELECTION message back to sender. After election, next phase is declaration of a leader. It is performed in reverse order than that of in previous phase, the last unvisited node transfers the leader message, containing leader identity, to all the other nodes. Message handling is the same as in previous phase, but here the node forwards the leader message to all of its neighbors. This phase ends when all the nodes have been visited by the leader message. When a node is separated from the network during the election, it is not allowed to join the election process. If it wants to join cluster, request receiving nodes compare it with its Cluster ID and will accept or deny. This paper uses extrema finding diffusion computation and leaders are chosen considering the system's most beneficial choice. Moreover, this approach does not consider the case, in which leader node is separated during the election process. It indicates conduction of new election.

C. Leader Election Algorithm within Candidates on Ad Hoc Networks [10]

The paper is extension of [11] and [12]. The assumptions used are nodes with UIDs, asynchronous FIFO, bidirectional channels, and sufficiently large node buffers. This algorithm also deals with networks partitions and mergers that can occur during the election process. The new concept to the original algorithms [11] [12] is added that it uses candidate lists. Candidate list has the UIDs of current leader and four other main candidates. When nodes confirm that the leader is no more, the nodes initiate ELECTION messages. A number of messages are spread along tree to know highest UID. When this election message reaches the leaf nodes in the diffusion computation-spanning tree, they send acknowledgement messages to their parents. These acknowledgement messages contain the candidate lists. While returning back the candidate list is updated and sent back to initiator. Update is done to obtain the highest known UID of the nodes. Finally, the initiator combines the candidate lists and highest UID is elected as the new leader.

The paper gives an energy efficient algorithm which proves that every node has a leader in every situation and saves energy, a number of leader election processes must be less. This paper has shown that the candidate-based algorithm saves considerable energy.

D. Cooperative Leader Election Algorithm for Master/ Slave Mobile Ad Hoc Network [13]

An election algorithm for master/slave networks is proposed in [14]. The master is responsible for assigning and coordinating tasks and roles including scheduling mutual exclusive access. The slaves follow the master's order. The algorithm's assumptions are that by utilizing the algorithm iteratively, the MANET will end up with one unique leader. The system is synchronous and its channels are reliable. Nodes have unique IDs. In the election algorithm, each node maintains a tuple of two values, the ID of the current leader and the current leader's master value. Nodes that wish to be the new master, replace their tuple with each of its slaves. If it finds a node with higher criterion value, it will replace its own and broadcast it to the entire network of its slaves. In the network, there can be nodes that are Participant in Multiple Piconets (PMP)-a piconet is a network with at least two nodes and, at most one master. These nodes participate in multiple networks, but during each time slot, they obey only one master. When a new node joins the network and if its value is lower than the current leader of the network it just accepts the current leader. However, if the new node's value is greater than that of the current leader, the new node is the leader and the leader information is broadcast to the entire network. In case when the new node joins the network that has no leader or is yet to establish its leader, then if its role is a master node then it increases the counter by 1 or takes no action if it is a pure slave node.

E. An Energy Efficient Leader Election Algorithm for Mobile Ad hoc Networks [15]

It is based on the **Ω** failures detector [16]. An algorithm is highly adaptive with ad hoc networks. It can tolerate intermittent failures, such as link failures, sudden crash or recovery of mobile nodes, network partitions, and merging of connected network components associated with ad hoc networks. It also reduces the number of exchanged messages to make it energy efficient. Each process starts the execution by electing itself and sends the identifier of its leader to all processes in the system. A competition will take place between different leaders. The winner will be the process that has the smallest ID. Each process p on receipt of (ALIVE, q) tests whether q is equal or less than its leader. If it is equal to *leaderp*, p sends the same message to every process r such that q<r. If q< *leaderp,* the process p will not send the message (ALIVE, q) till it makes its leader equal to q. In both previous cases, p restarts its timer. Upon expiration of its timer, p is elected as a leader. This paper has given a leader election algorithm that is well-suited for use in mobile ad hoc networks. It is highly adaptive to arbitrary and concurrent topological changes induced by node mobility, network partitioning and merging components. This algorithm ensured that eventually each connected component of the topology graph has exactly one leader. Moreover, the algorithm is efficient in number of exchanged messages and requires processes to communicate with only a subset of their current neighbors.

F. A Consensus-Based Leader Election Algorithm for Wireless Ad Hoc Networks [17]

An algorithm for a consensus-based leader election algorithm guaranteed that a consensus be reached, reduction in the number of message passing & fault tolerance. It is based on Bully and Paxos[18] algorithm. This algorithm is also called as group leader election algorithm. An author considered maintenance of information integrity as a crucial issue. This algorithm considers residual power and node degree as election criteria. In this algorithm, when a group leader has left or crashed, the finder node will request higher priority nodes and starts the election process. Highest priority nodes confirm the existence of leader. If it does not exist or crashed they send confirmation message. A node with highest priority proposes he is going to be a leader. Every other node confirms and accepts proposal. When more than half of nodes accept proposal and declare a node with highest priority as a COORDINATOR. The meaningful leader election votes help the group to elect a leader of great ability. This algorithm not only reduces the number of message passing but also improves network consistency. But the message faking or security problems are not included.

G. An Efficient Overhead-Aware Leader Election Algorithm [19]

This paper given an efficient algorithm where leader election is performed with lower number of messages and thus reduces the overhead associated with message passing. The amount of time each node should wait in order to know about the elected leader is also decreased. This algorithm followed concept of Bully Election which states the bigger guy wins the war but it additionally maintains descending-sorted list (MAP) where the first item in list is a node with highest ID. In this algorithm, when a node detects that leader is crashed, it does not send ELECTION message to every other nodes. Instead it looks in MAP to check next lower ID of crashed leader and sends ELECTION message to it. If a node with lower ID than crashed leader is available then it send OK to sender and also sends COORDINATOR message to every other nodes. It also mentioned that if suddenly crashed leader becomes available then there is no need of conducting election again. It can send ELECTION message to other nodes with lower ID and can become COORDINATOR. This algorithm works efficiently with passing lower number of messages but it has not considered network partitioning, network merging.

H. Cooperative Clustering Based Secure Leader Election Model for Intrusion Detection in MANET [20]

This paper gives an election algorithm based on Partner Clustering algorithm (PCA). In this paper a model first prevents node from behaving selfishly, then it introduces incentives designed in the form of reputation to support nodes to truthfully join in the election scheme for this it introduces Vickrey, Clarke and Groves (VCG) model. A Partner Clustering algorithm (PCA) is used to elect suitable Cluster Head and partition the network into non overlapped 1-hop clusters. It decreases the percentage of leaders, single-node clusters and greatest cluster size and increases average cluster size. This algorithm initially estimates appropriateness for being a Leader node, after estimation a suitability distribution mechanism is used and finally the nodes with higher suitability scores than their neighbors are elected as the leader. Cluster Head suitability is calculated on parameters such as Coverage distance, transmission power and normal speed of all nodes in the network. In every cycle of routing suitability score is sent to every nodes into BEACON message. Each node stores this information and once again transmits with next BEACON. A node among the biggest score becomes a Cluster Head then it updates BEACON message and broadcasts it and then remaining neighbors join the cluster and become Cluster Members. In suitability distribution if BEACON message got lost due to collision or some other issues, a node with poorer suitability score may declare itself as a Cluster head. Thus two or more cluster head in the similar 2-hop coverage area appear and compete for cluster members. But there are no two heads for single cluster .This algorithm gives cost efficient solution for intrusion detection for a cluster head on behalf of cluster. After implementing this algorithm we can get secure and an energy efficient MANET.

I. Leader Election Algorithm in Wireless Environments using Fibonacci Heap Structure [21]

This algorithm is very simple. In this algorithm, an author assumed that there is only one process at each node of the distributed system. These remaining N-1 nodes are candidate nodes for leader election. After some period of time a node from N-1 nodes is elected as a new leader. Finally, only one leader should be present, this condition is satisfied by an author. In this paper, concept of Fibonacci heap [22] tree is used for selecting the leader. The communication model is a point-to-point communication network which is represented as a simple connected undirected graph, where the vertices represent processes and two vertices are linked by an edge if the corresponding processes have a direct communication link. Processes communicate by message passing through these edges. Such algorithm works well in environment where faults are present. It finds minimum key of heap and declare it as the leader. A *Fibonacci heap* [23] is a collection of heap-ordered trees. It helps to declare the leader in O(1) time in case of failure. A root of heap containing minimum value is called the leader. In this algorithm a node joining the network group gives information about its parent, child, left, right sibling. It satisfies the condition that, the value of node is always greater than its parent. This concept helps to maintain smallest node at the root of a tree. It proved that insertion of node in a network can be done faster in O(1) time. Thus it is useful and efficient for wireless networks where there is frequency of topology changes is high. In topology changes, lot of insertions and deletions are performed. When deletion of root occurs, it indicates the leader has been crashed. In such case its children will become root of new tree. A comparison is done between children and a child with minimum value become root of a tree who is new leader. This algorithm takes O(log n) time to do this activity. This algorithm proved that making heap, inserting a node in heap, decreasing root and making union of children values in case of failure takes O(1) time and deletion of minimum node takes O(log n) time. Since a node stores information about its parent, child, left and right sibling it consumes less memory 4n. Degree of a node indicates its number of children and mark field indicates whether a node has child or lost. Maximum degree is O(log n) and message complexity is O(log n).

J. Self-Stabilizing Leader Election Algorithm in Highly Dynamic Ad-hoc Mobile Networks [24]

This algorithm has given method to handle frequent topology changes. It uses time interval based computation model. It helps to manage topology changes in fast way. A network partitioning, topology changes, merging of network components are common properties of a mobile ad hoc network. This algorithm works in highly dynamic and asynchronous networks. It uses TORA mechanism. A node has less overhead because its knowledge is limited up to one hop. As compared to earlier algorithm it considered highly changing nature of topology. The base of given algorithm starts with number of nodes and number of edges. In mobile network there is chance that a node can leave a network or can join a network. But this algorithm keeps the track of such change by informing its neighbor and this information is spread to other nodes through neighbor node of leaving node. Later the algorithm assigns weights to each edge, and identifies the source node and destination node. Thus keeps track of neighbors to each corresponding node. Upon execution of the system we can determine the leader node among available nodes. It is based on weights of each node [25], [26]. The weight is dependent on frequency of message sending, receiving, number of neighbors and is computed by stabilizing Algorithm [27]. If the leader node leaves the network due to network problems then that information passed to its neighbor's and subsequently the stabilizing algorithm select other node as a leader node or coordinator node. In this paper, an author proposed a self-stabilizing leader election algorithm that can converge to a legitimate state even in the presence of topological changes during convergence time.

III. Analysis

In algorithm [8], with increase in mobility there is increase in time complexity. It assumes if node is failed and again wake up from crash, then it should start a new election to know his new leader. This activity is costlier and message passing will be more. In proposed enhancement, a waked up node can ask its alive neighbor about new leader instead of stating new election. This saves number of messages sent. This algorithm uses extrema finding using the nodes UIDs in which a node aborts its current diffusion and joins the higher order UID. We can make it faster if comparison is done on starter's quality. In algorithm [9], for cluster formation nodes consider two important factors viz. their locality and closeness in vicinity. Elections are conducted in tiny clusters, and then cluster's newly elected leaders broadcast their identity to other clusters. The performance is better as the initial leader election is in small groups. Un-visited nodes should not be allowed to join the election process until it is completed since this increases the overhead on election. In algorithm [10], it gives energy saving arbitrarily changing topology. We suggest that the election of leaders must use the highest remaining battery life and the lowest mobility range in the election process. Since nodes with low mobility have least chances of merging and partitioning from their current clusters. The algorithm uses the UID to handle the multiple diffusion computations to elect new leader. We suggest to use CSMA/CD algorithm to handle such multiple diffusion computations since it requires less number of messages. In algorithm [13], during election only the master can communicate with the rest of the nodes. In proposed enhancement, we recommend slave nodes be allowed to communicate during election. This reduces the election time. Here only master is conveying message. If a node wishes to start an election process, it sends a message to its master or other active masters if in PMP. If master is not currently active for the node, it broadcasts the need to its neighbors to start the election process, and nodes wishing to become leaders will make themselves candidates. This creates more chances for nodes to become leaders. Then the system will not be totally dependent on master. An algorithm [15] is well suited for use in mobile ad hoc networks. It is highly adaptive to arbitrary and concurrent topological changes induced by node mobility, network partitioning and merging components. This algorithm ensures that eventually each connected component of the topology graph has exactly one leader. Moreover, the algorithm is efficient in number of exchanged messages and requires processes to communicate with only a subset of their current neighbors. But it needs enhancement of maximum battery life or maximum computation power. In [17], the meaningful leader election votes help the group to elect a leader of great ability. This algorithm not only reduces the number of message passing but also improves network consistency. But the message faking or security problems are not included. An algorithm [19] works efficiently with passing lower number of messages but it has not considered network partitioning, network merging. An algorithm [20] gives cost efficient solution for intrusion detection for a cluster head on behalf of cluster. After implementing this algorithm we can get secure and an energy efficient MANET. An algorithm [21] is very simple and an efficient algorithm. In paper [24], an author proposed a self- stabilizing leader election algorithm that can converge to a legitimate state even in the presence of topological changes during convergence time. Table 1 below shows the comparison between different existing Leader Election algorithms. The summary of analysis indicates that all algorithms consider fault tolerance, most MANET uses asynchronous channel for communication and most of algorithms support for topology changes.

In [8], when a crash node wakes up, then by default starts an election. It also indicates the number of elections is directly proportional to the number of nodes waking up from crash.

$$\textit{Number of Elections } \alpha \textit{ Number of nodes waking up from crash}$$

But as an enhancement, a waked up node can send inquiry message to its neighbor to inquire about its current leader. A number of neighbors are dependent on node density. A node density is number of nodes attached to a given node. If a node N has M neighbors (M<N), then it can send M inquiry messages to its neighbors. Thus instead of starting new election again with M inquiry messages it can find its leader. This solution avoids full-

scale election to be conducted. Thus with this enhancement election time can be reduced and also number of messages will be less in case of failure. We can say in case of failure,

$$T(N) \alpha M$$

Where, *T(N)* – Time required by waked up nodes N for election and *M* – A node density

In [21], in case of failure waked up node can get its leader with O(1) message complexity provided it has point-to-point connection. It indicates a leader can be found with single hop.

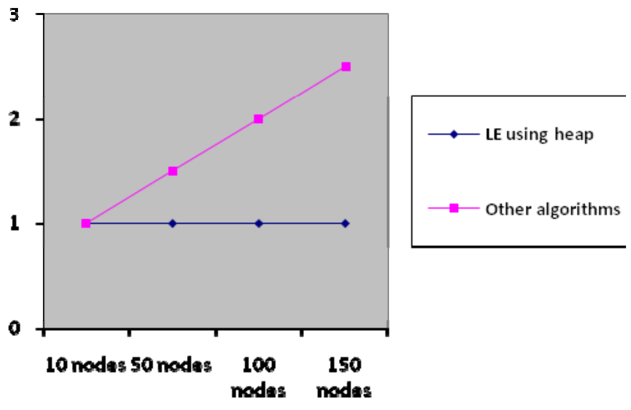


Figure 3: Message Complexity

IV. Proposed Algorithm

We have given a formal algorithm of electing a new leader in a MANET. In the process of election, the position of nodes in the network, the battery life, a time complexity, a message complexity and security factors are taken into considerations. We designed a function point by considering these five parameters of a node. Our main focus is to find a node with maximum remaining battery life, less time complexity, less message complexity, which can handle attacks very well.

We consider 4 values of battery life (B) for each node.

$$B = \begin{cases} 3, \text{ if the remaining battery power is above 80\% of total capacity} \\ 2, \text{ if the remaining battery power is between 60\%-80\% of total capacity} \\ 1, \text{ if the remaining battery power is between 40\%-60\% of total capacity} \\ 0, \text{ otherwise} \end{cases} \tag{3.1}$$

Similarly, 4 values for Security implementation (S),

$$S = \begin{cases} 3, \text{ if the security implemented is above 90\% against total attacks} \\ 2, \text{ if the security implemented is between 70\%-90\% against total attacks} \\ 1, \text{ if the security implemented is between 30\%-70\% against total attacks} \\ 0, \text{ otherwise} \end{cases} \tag{3.2}$$

Time complexity T can be by number of steps required to reach consensus on the newly elected leader. Message complexity M can be calculated using number of messages required to elect a new leader. For the proposed function point, we associated five weights, m1, m2, m3, m4 and m5 with the node's position, time complexity, message complexity, battery life and security.

$$Ob(i) = m1 \text{ X } \frac{d-di}{d} + m2X \frac{1}{T} + m3X \frac{1}{M} + m4XB + m5 \text{ X } S \tag{3.3}$$

Where *d* is median value of the network [28] and *di* is the average distance of all nodes in the network from node i. Node L is the leader of the system if ***Ob(L) = max(Ob(i))***, for all i.

Algorithm

Step 1: For all nodes find *Ob(i)* using equations 3.1, 3.2, 3.3 also find time complexity and message complexity.

Step 2: Identify node L, where *Ob(L) = max(Ob(i))*, for all i.

Step 3: Node L broadcasts a message to inform others about its new status as a leader.

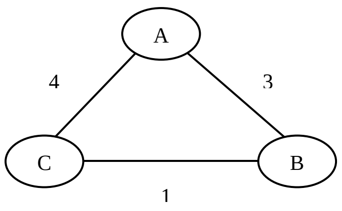
Step 4: A leader keeps a track of addition and removal of nodes from a network and it also takes care of attacks on network e.g. DOS attacks

Step 5: If the recomputation of median node is needed then a fresh election is fired by executing Step 1 to Step 3

Step 6: If someone identified that a leader does not exist, then a new election can be fired by executing Step 1 to Step 3 again.

V. EXPERIMENTAL RESULTS

We considered following example,



There are three nodes present in a network with (assume) following configuration-:

Node	Average distance (di)	Time required	Message complexity	Battery life	Security
A	3.5 meters	3 sec	2	90%	90%
A	3.5 meters	3 sec	2	90%	90%
B	6.5 meters	8 sec	2	75%	85%
C	7 meters	10 sec	2	20%	65%

Table 2: Example

Assume, T=1, M=1, m1=1 and median value = 4

Using equation 3.3,

Ob(1)= 4- 3.5 / 4 X 1 + 1 X 1/3 + 1 X 1/ 2 + 3 X 90 + 3 X 90 = 540.955

Ob(2)= 4- 6.5 /4 X 1 + 1 X 1/8+ 1 X 1/ 2 + 3 X 85 + 2 X 85 = 425

Ob(3)= 4- 7/4 X 1 + 1 X 1/10 + 1 X 1/2 + 0X 20+ 1X 65 = 64.85

Thus, *Ob(L) = max (Ob(1), Ob(2), Ob(3))* = 540.955

In analytical way we can say that **Node A** is the leader.

VI. CONCLUSION AND FUTURE WORK

We analyzed ten leader election algorithms applied in MANET using common factors such as time complexity, message complexity, assumptions considered, timing model, whether there is duplication of message, channel used, whether topology changes considered, energy efficient, mobility, battery life, basis for leader election and consideration of security factor, type of messages and who will be the leader. Our analysis proposes enhancements such as maximum utilization of nodes during election to reduce load. It also proposes reduction of a costlier new election when a leader wakes up after crashing. It simply inquires to its functioning neighbors to identify its current leader. It uses better election criteria such as battery life, mobility and security. From our comparative study we can say that *A Consensus-Based Leader Election Algorithm for Wireless Ad Hoc Networks* and *A leader election algorithm using Fibonacci Heap structure* are novel approach towards leader election. They are faster and work best in faulty environment. If we will integrate security concept in this algorithm, these will be best algorithms. We have introduced a deterministic algorithm for electing a leader in MANET. We consider a wireless ad hoc network where each node has a weight value that represents some performance related property e.g. node’s position, time complexity, message complexity, battery power, security. Similarly for changing topology we can recompute median node value. If we select best technique for finding leader by reducing number of messages and time for computation and will make the leader strong enough to handle all types of attacks, the algorithm will be best for leader election problem.

As a part of future work, we intend to include more recent election algorithms, some more additional factors for further analysis. This is analytical algorithm on the basis of properties of a node in a network such as its position, time complexity, message complexity, its battery life and security. As an enhancement a performance oriented algorithm can be implemented in real life mobile environment.

REFERENCES

[1] G.L. Lann. Distributed systems, towards a formal approach. In IFIP Congress, pages 155–160, 1977.

[2] Mina Shirali, Abolfazl HaghighatToroghi, and Mehdi Vojdani. "Leader election algorithms: History and novel schemes". Third 2008 International Conference on Convergence and Hybrid Information Technology,2008.

[3] H. Garcia-Molina, "Elections in a distributed computing system". IEEE Trans. Comput., 31(1):48–59, 1982.

[4] G.L. Lann. "Distributed systems, towards a formal approach". In IFIP Congress, pages 155–160, 1977.

[5] Ernest Chang and Rosemary Roberts. "An improved algorithm for decentralized extrema-finding in circular configurations of processes". Commun. ACM, 22(5):281–283, 1979.

[6] www.antd.nist.gov (The National Institute of Standards and Technology).

[7] P. Nicopolitidis, M. S. Obaidat, G. I. Papadimitriou, and A. S. Pomportsis. In Wireless Communication Principles and Fundamentals, page 422, July 2009.

[8] S. Vasudevan, J. Kurose, and D. Towsley, "Design and Analysis of a Leader Election Algorithm for Mobile Ad Hoc Networks," ICNP'04, 2004.

[9] Gang Zhang, Jing Chen, Yu Zhang, and Chungui Liu, "Research of Asynchronous Leader Election Algorithm on Hierarchy Ad Hoc Network," WiCom '09. 2009.

[10] S. Lee, M. Rahman, and C. Kim, "A Leader Election Algorithm Within Candidates on Ad Hoc Mobile Networks," Embedded Software and Systems, Lecture Notes in Computer Science, Vol. 4523, pp: 728-738, Springer Berlin / Heidelberg 2007.

[11] G. H. Molina, "Elections in a Distributed Computing System." IEEE Trans. Comp, vol.31, no. 1, pp.48-59, 1982.

[12] Leslie Lamport, "Time, Clocks, and the Ordering of Events in a Distributed System," Communications of ACM, vol. 21, no. 7, July 1978.

[13] R.Ali, S. Lor, R. Benouaer, M. Rio, "Cooperative Leader Election Algorithm for Master/Slave Mobile Ad Hoc Network," 2nd IFIP Wireless Days (WD), Paris, 15-17 December 2009, pp. 1-5, December 2009.

[14] G. Le Lann. "Distributed systems: Towards a formal approach", Information Processing 77,Proc. of the IFIP Congress, pp. 155-160, 1977.

[15] Leila Melit, Nadjib Badache "An Energy Efficient Leader Election Algorithm for Mobile Ad hoc Networks", IEEE, 2011

[16] T. Chandra and S. Toueg, "Unreliable failure detectors for reliable distributed systems", Journal of ACM, march 1996, pp. 225–267, 1996.

[17] Hsu-Chia Cahng, Chi-Chun Lo, "A Consensus-Based Leader Election Algorithm for Wireless Ad Hoc Networks", International Symposium on Computer, Consumer and Control, IEEE, 2012

[18] L. Lamport, "Paxos made simple," Nov. 2001. Available at http://pdos.csail.mit.edu/6.824/papers/paxos-simple.pdf(Nov.2011)

[19] Muneer Bani Yassein, Ala'a N. Alslaity, Sana'a A. Alwidian, "An Efficient Overhead-Aware Leader Election Algorithm for Distributed Systems", International Journal of Computer Applications (0975 – 8887), Volume 49– No.6, July 2012

[20] R. Jasmin Reeda, A.Ananthkumari, "Cooperative Clustering based Secure Leader Election Model for Intrusion Detection in MANET", IJMIE volume 2, Issue 7 ISSN: 22249-0558, July 2012

[21] Arihant Kumar Jain, Ramashankar Sharma,Arihant Kumar Jain, Int.J. Computer Technology and Applications, Vol 3 (3),871-873, ISSN:2229-6093, IJCTA| MAY-JUNE 2012

[22] Michael L. Fredman, Robert Endre Tarjan, Fibonacci Heaps and their uses in improved network optimization algorithms´, IEEE, 1984

[23] Cormen T.H., Leiserson C.E., Rivest R. L., Stein C., "Introduction to Algorithm", McGraw Hill Book Company, (2001)

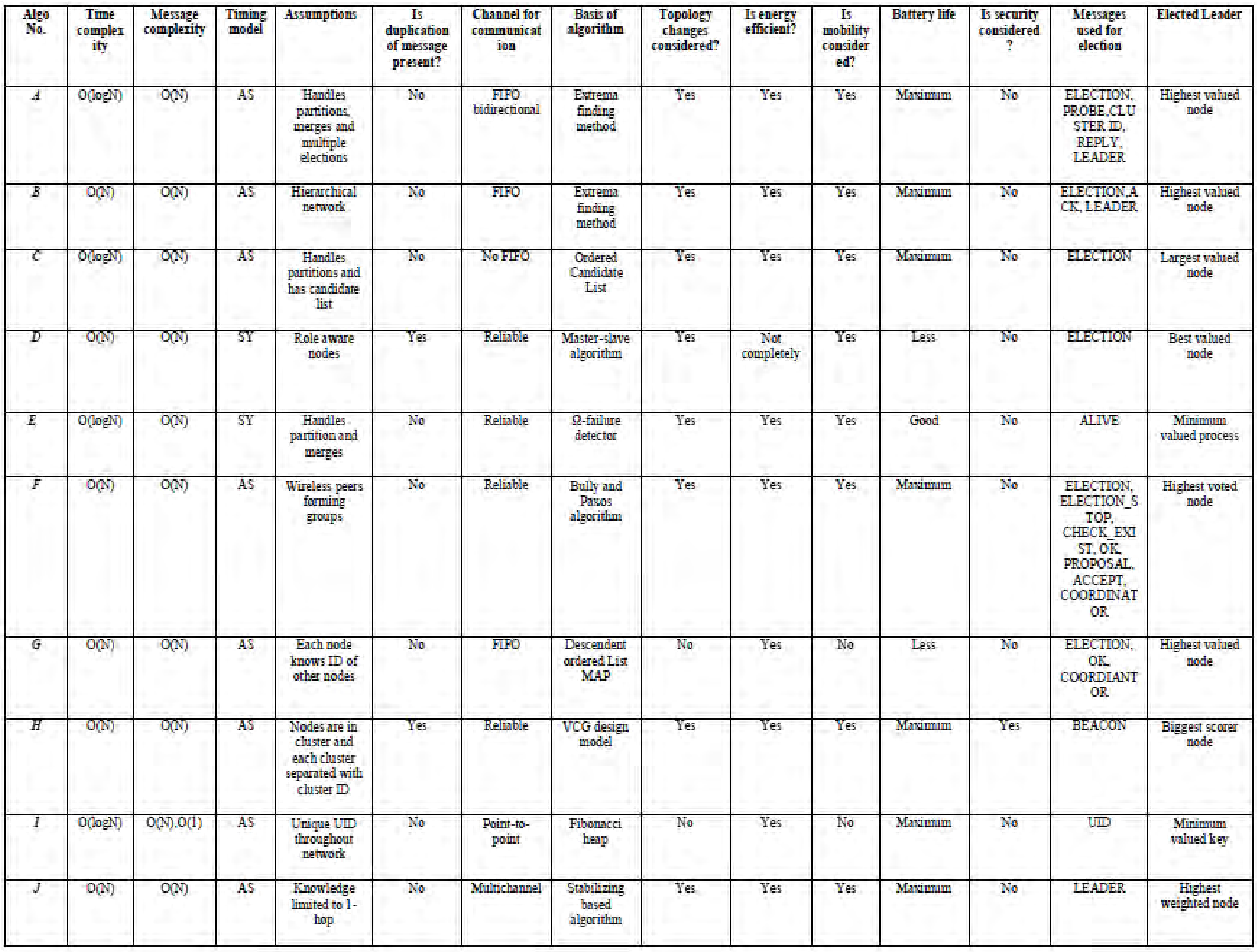
[24] K Ranganath, L.Naveen Kumar, Y.V.Sreevani, "Self-Stabilizing Leader Election Algorithm in Highly Dynamic Ad-hoc Mobile Networks", Journal of Emerging Trends in Computing and Information Sciences, Volume 2 No.4,April 2011

[25] A.B. McDonald and T.F. Znati, " A Mobility-Based Framework for Adaptive Clustering in Wireless Ad Hoc Networks," IEEE J. Selected Areas in Comm, vol.17, no.8,pp. 1466-1487, Aug.1999

[26] V. Ramasubramarian, R.chandra and D. Mosse, "Providing a Birectional Abstraction for Unidirectional Ad Hoc Networks", Proc. IEEE INFOCOM, 2002.

[27] K. Romer, "Time Synchronization in Ad Hoc Networks", Proc. ACM MobiHoc '01, pp. 173-182, 2001

[28] O. Kariv and S. L. Hakimi. "An algorithmic approach to network location problems: The p-medians". In SIAM Journal on Applied Mathematics, volume 37, No. 3, pages 539–560. Society for Industrial and Applied Mathematics, Dec 1979.



Algo No.	Time complexity	Message complexity	Timing model	Assumptions	Is duplication of message present?	Channel for communication	Basis of algorithm	Topology changes considered?	Is energy efficient?	Is mobility considered?	Battery life	Is security considered?	Messages used for election	Elected Leader
A	O(logN)	O(N)	AS	Handles partitions, merges and multiple elections	No	FIFO bidirectional	Extrema finding method	Yes	Yes	Yes	Maximum	No	ELECTION, PROBE,CLUSTER ID, REPLY, LEADER	Highest valued node
B	O(N)	O(N)	AS	Hierarchical network	No	FIFO	Extrema finding method	Yes	Yes	Yes	Maximum	No	ELECTION,ACK, LEADER	Highest valued node
C	O(logN)	O(N)	AS	Handles partitions and has candidate list	No	No FIFO	Ordered Candidate List	Yes	Yes	Yes	Maximum	No	ELECTION	Largest valued node
D	O(N)	O(N)	SY	Role aware nodes	Yes	Reliable	Master-slave algorithm	Yes	Not completely	Yes	Less	No	ELECTION	Best valued node
E	O(logN)	O(N)	SY	Handles partition and merges	No	Reliable	Ω-failure detector	Yes	Yes	Yes	Good	No	ALIVE	Minimum valued process
F	O(N)	O(N)	AS	Wireless peers forming groups	No	Reliable	Bully and Paxos algorithm	Yes	Yes	Yes	Maximum	No	ELECTION, ELECTION_STOP, CHECK_EXIST, OK, PROPOSAL, ACCEPT, COORDINATOR	Highest voted node
G	O(N)	O(N)	AS	Each node knows ID of other nodes	No	FIFO	Descendent ordered List MAP	No	Yes	No	Less	No	ELECTION, OK, COORDIANTOR	Highest valued node
H	O(N)	O(N)	AS	Nodes are in cluster and each cluster separated with cluster ID	Yes	Reliable	VCG design model	Yes	Yes	Yes	Maximum	Yes	BEACON	Biggest scorer node
I	O(logN)	O(N),O(1)	AS	Unique UID throughout network	No	Point-to-point	Fibonacci heap	No	Yes	No	Maximum	No	UID	Minimum valued key
J	O(N)	O(N)	AS	Knowledge limited to 1-hop	No	Multichannel	Stabilizing based algorithm	Yes	Yes	Yes	Maximum	No	LEADER	Highest weighted node

Table 1: Comparison of Election Algorithms