Leader Election Approach: A Comparison and Survey

By

Kshama Tiwari[1], Brajesh Kumar Umrao[2]
M. Tech. Scholar, Asst. Professor
Dept. of Computer Science and Engineering
UIT, Allahabad, India
tiwari.kshama@gmail.com, umraoniec@gmail.com

ABSTRACT

In distributed system, the coordinator is needed to manage the use of the resources in the shared environment. Many algorithms have been proposed for the same. They have various positive and negative parts. Here we will discuss those issues which ensure the efficiency of the algorithm for election leader. Here a comparison will be provided to show the advantages and disadvantages of different election algorithms. The comparison would be based on the number of messages passing and the order of time complexity.

KEY WORDS

Coordinator, successor, electioneer, ordinary set, candidate set.

1. INTRODUCTION

Various distributed algorithms require that there be a unique coordinator process in the entire system that is responsible to perform various kinds of functions. Many persons have proposed a wide range of election algorithms. The Bully algorithm is one of them, and it is one of the oldest election algorithms. It reduces redundant elections, minimizes message passing and network traffic. Here a comparative analysis of various election algorithms is proposed, which is based on different concepts like priority number, election commission, k coordinator process, successor, heap tree, max heap, Fibonacci heap, etc.

2. ELECTION ALGORITHM

Election algorithm is used to find the coordinator in the network in distributed environments, as the coordinator is responsible for managing the utilization of resources. Various algorithms require peer processes to elect a leader or a coordinator. It is required to select a new leader if the existing one fails to respond. Each process has a unique identification number that helps to specify the priority of a process in the network. So election algorithm gives the approach to find the new leader.

3. LITERATURE REVIEW

Bully algorithm proposed by Garcia-Molina for the coordinator election, works on the assumption that the system uses time as a parameter to detect process failure (the coordinator). All processes have a unique number in the system and they know the process number of all other processes. Once an election is held, a process with the highest process number is elected as a coordinator, which is agreed by other nodes (process). The Bully Algorithm has three types of messages: first is election message sent to announce an election, second is an ok message sent as the response to an election message, and third is victory messages sent to announce the new coordinator among all other live processes. When any node finds that the leader node has been crashed, then it immediately announces itself as a new leader if no other higher process id node exists. Otherwise it starts election process as bully algorithm had proposed. There are some drawbacks of the bully algorithm as whenever a crashed node becomes alive it starts election process that gives an extra overhead to system resources. Next drawback is the heavier number of repetitive election process, especially in the worst case when the election is started by the lowest id process. Then at least n-2 redundant election processes take place, where n stands for the total no of live nodes. Another drawback is that more than one process can start election parallel (no solution goes with one), so many unwanted election processes take place. So the main drawback of bully algorithm is the large number of messages passing the order O (n2) [2].

So Mamun proposed a modified bully algorithm. The modified bully algorithm has been proved better than the bully algorithm as in the worst scenario when the lowest id node detects the failure of coordinator node, the bully algorithm performs n-1 number of elections to find the new coordinator, which takes $O(n^2)$ messages to be passed. Whereas in modified bully algorithm only O(n) messages are required to be sent. Also, when the detector node finds itself as the next highest node to the failure node then it directly declares itself as a coordinator by sending n-2 coordinator messages, without passing any election message. So in the worst case the bully algorithm requires $O(n^2)$,whereas modified bully algorithm requires 2(n-1) messages [3]. By the above comparison it is justified that modified bully algorithm is better than the bully algorithm, but still it has some drawbacks as it doesn't provide any scope to cope with simultaneous detection of failure and

election of new leader node. This number of messages sent can be minimized further by using some latest data structures for nodes arrangement.

So the improved method over modified bully election algorithm is the modified election algorithm that reduces the number of messages to be transmitted and reduces the traffic on the distributed environment. Thus, the latter has proposed the concept of K coordinator group that prevents the global election of coordinator redundantly, as the coordinators after crashed nodes are already decided. The modified election algorithm defines the coordinator group like-- {Coordinator1, Alternative1, Alternative2 …Alternative k}. These alternative processes from 1 to k are the processes with the next priority numbers to the coordinator1.

So by using modified election algorithm a process with greatest number is selected as coordinator, and then k-1 processes of coordination group are selected. Now the process id of each coordinator in the group is sent to all the processes, so when any process finds that the coordinator is crashed, then it sends a message to alternative1 to inform that the coordinator has been crashed. Then one of the following two situations may arise: i) if alternative 1 is alive, ii) If alternative 1 is not alive. The main idea is that it uses candidate nodes. The complexity of proposed algorithm is O (n), whereas in bully algorithm it takes O (n^2) time. That shows the modified algorithm is better than the previous one (bully algorithm). There are several disadvantages associated with the modified election algorithm—(i) it doesn't guarantee to stop multiple elections parallel, which imposes a heavy load on the network, ii) this algorithm does not provide any solution when an electioneer node fails during leader election [4].

Now there is another algorithm known as Enhanced Bully algorithm for the leader election that has been proposed by Murshed. This algorithm has proposed some modifications to Traditional Bully algorithm and the Modified Bully Algorithm. Here the concept of set division has been proposed, according to which the complete node set is divided into two equal half sets where one is *Candidate* set, and another is *Ordinary* set. Candidate set consists of N/2 Nodes. The other N/2 nodes are ordinary nodes, as the node in the candidate set has higher id number than the ordinary set nodes. Each node has the information about other node sets and ids. Here the election message consists of the id of the failed node and the failure detector node. The answer message consists of the node ids of the leader and the complete candidate set. Here another concept has also been proposed with the name tiebreaker time δ that stops redundancy and unnecessary elections and helps to reduce the number of messages required in the election of new node.

The election procedure is performed on the basis of the following situations:

i. Ideal case (IC): This case arises when a node i in the Ordinary set detects failure of the leader node then it sends an election message to the set of candidate node set and waits until it gets Ok message. On receiving Ok, it identifies the highest id node and, generates a Coordinator message to be updated about the newer coordinator node, and sends to all the nodes of both sets. If the node detects itself as the next highest node just after the crashed node, it declares itself as a leader node.

ii. Candidate failure Case (CFC): If the failure detector doesn't get any reply from any candidate within the given time duration then it is assumed that all candidate nodes have been crashed, and starts election of the ordinary node set. But this case is rare as it is not possible that in the network 50% nodes get crashed.

iii. Electioneer Failure Case (EFC): This case arises when the electioneer node gets failed just after sending election messages to other nodes, so the candidate node waits till (T_{ok}, i) time. If the Potential candidate node doesn't get coordinator message within (T_{ok}, i) time it will announce itself as a coordinator.

iv. Simultaneous election Case (SEC): Here more than one electioneer node starts election; simultaneously potential node that gets election message sends an ok message only to the highest id sender node. So the highest id detector node declares the new leader node by broadcasting the coordinator message.

v. Node Revival Case (NRC): This case arises when a crashed node recovers from failure. If the recovered node is an ordinary node then it sends a query message to the nodes of candidate set and if it is a candidate node then it sends the query message to the just higher id node of its own set. Then the nodes which receive the query message send the answer message that contains the id of the current leader node and the ids of the members of the candidate node.

This algorithm requires 2N-1 messages in the worst case if at least one candidate node exists. When the case NRC arises, it takes N-1 messages. This algorithm also gives better solution towards simultaneous election of the leader node, where number of messages are minimized by the use of tiebreaker time and set division process. This algorithm ensures aliveness and safety requirements of the leader election algorithm. The algorithm claims to be fast, correct and efficient, and also requires fewer numbers of messages to elect a new leader. As it is said that the total nodes will be divided into two sets, the sets are required to be arranged in ascending order or descending order. Whenever any crashed node recovers, it is required to arrange the group every time. It gives a complex structure due to various technical complexities in the calculation of tiebreaker time δ other mathematical terms [5].

The other approach for election algorithm is the use of exact two successors with main coordinator node. It is also called Vice Coordinator Election algorithm. Here the author has provided the concept of defining the two successor nodes before starting any process. So, whenever the main coordinator will be crashed, the 1st successor will be decided as the next main leader, which will minimize the number of messages passing for the election of the next main coordinator. It rarely happens that both successor and main coordinator get crashed simultaneously. So, in this the process, the highest id node is elected as the main coordinator and then this main coordinator sends

two messages to the next two highest id processors to define the main successor and sub successor.

Two types of messages are used here-- one is coordinator message, and the other is election message. The coordinator message is the message that is sent by the coordinator to the two main successors to inform them that they are the successors. Also the nodes check that the successors are in the active state. The successor nodes send an acknowledgment message to ensure their availability. After this confirmation, the coordinator node sends n-3 messages to the rest of the nodes to inform the successor ids. When the main coordinator gets crashed, then the election message comes in the role, as the election message is sent by the node which detects the failure of the coordinator node to the main successor. Then the main successor checks whether the main coordinator is alive or actually dead. If it is dead, then it sends n-2 new coordinator messages to all other nodes. Here there are n-2 messages because coordinator information has to be sent to all except the dead coordinator and the successor itself. After that, the main successor selects its next two successors by an additional message, as for this the first has already been defined. So, the total time required here is n-2+1. According to the process and required number of messages, the time complexity is derived as O (n). According to algorithm whenever the coordinator fails, it will elect the new successor that gives an extra overhead of election of successors, and in the worst case if both successors fail, then it will move to the traditional election algorithm approach. Also, it doesn't provide any solution to the simultaneous election of coordinator failure detector [6].

Another approach is the concept of Heap Tree. This puts great effort in reducing the number of messages passing for the election of the coordinator. Here the concept of the election of coordinator is based on heap tree, as the set of nodes are structured as a heap tree of nodes. As the principle of election algorithm says the highest id node is selected as the leader, so in heap tree the MAX HEAP is used, because the max heap tree contains the highest id node at the root of the tree that gives an easier way to get the ids in a hierarchy with fewer comparisons. Here MAX-HEAPIFY () procedure is used to generate the max heap, which run in the complexity of O (log (n)). In max heap, the father node is always greater than their children, but children don't have any fixed sequence as the left sibling can be greater than the right and vice-versa. So there is an improvement proposed to maintain the sequence, where algorithm requires only one comparison, and we can decide which child node has the larger value. Now whenever any node gets the information that the leader has crashed (the root has been deleted), and the node sends the election message to its father, the message is traversed until it reaches to the child of root node. When the children of the root receive the message they don't need to do more comparisons as it is pre-decided that the left child has greater value than the right child. So the left child is immediately designated as the new leader. By this, the number of messages in the election of the coordinator can be optimized, as there is no need to send the election message to all the nodes in the tree. In the above approach, the leader can be decided in less than O (log n) time. But here each node has to maintain the information of its father, the left child, the right child and its sibling, which requires the same memory space as 4n, as much as the heap tree needs. When each node sends the maximum number of messages to its parents, then also n-1 messages will be sent at the most. So in the max heap method, maximum number of messages can be passed in the order of (n-1). Here the main issue is that whenever the root is deleted, each time the MAX-HEAPIFY () procedure is required to build the heap [7].

Here, the other approach over the max heap is the use of Fibonacci heap used to maintain the structure of nodes to get the new leader fast. The concept of Fibonacci heap says that it is better than max heap in the process based on time complexity and number of comparisons. In Fibonacci heap the procedures for creation of heap, insertion of node, finding minimum, decreasing root, union of heaps take O(1) time and deletion of min key takes O(log n) amortized time. The maximum degree achieved using Fibonacci heap is O (log n) and the maximum number of messages required to pass is in the order of O (log n) [8].

4. COMPARISON AND DISCUSSION

By the comparison of above various election algorithms, we got the information about the different concepts used in each and every algorithm. The comparison also defines the series of algorithms with the better approach. The efficiency of the algorithm is justified by the comparison of the number of messages that are required in the selection of the new leader node. It also gets confirmed by the comparison of time complexity of above-mentioned algorithms. By the comparison of above five to six algorithms, the first bully algorithm proposed by Garcia-Molina is proved as the simplest algorithm but not efficient as it requires heavy number of messages passing with the time complexity of O (n^2). As there are so many parameters to prove an algorithm efficient, the election algorithm with Fibonacci heap structure gives the time complexity of O(log n) and takes smaller memory space in comparison to other election algorithms.

The comparison of above defined algorithm is given in the table form that shows the comparison of the order of message passing.

Table1: Performance of Algorithm of Leader Election

Algorithm	Memory Required	Order	Minimum Message
Fibonacci Heap	4n	log(n)	log (n)
Modified max heap	4n	log(n)	log(n)
Bully with Two Successor	n^2	n	n-2
Enhanced Bully	n^2	n	n-1
Modified Election	n^2	n	(n-2)+1
Modified Bully	n^2	n	n-1
Bully	n^2	n^2	2n-2

5. CONCLUSION AND FUTURE WORK

In this paper, we presented a comparison of various election algorithms to get the idea about one of all the efficient algorithms mentioned here. It is the election algorithm with Fibonacci heap structure with the time complexity of O(log n). In future, we will try to implement the election algorithm on the binomial heap. This would be better than the Fibonacci heap in terms of processing and time complexity.

REFERENCES

[1] Sinha, P.K. Distributed Operating Systems Concepts and Design; Prentice-Hall: Upper SaddleRiver, NJ, USA, 2002; pp. 332–334.
[2] Garcia-Molina, H. Elections in a distributed computing system. IEEE Trans. Comput. 1982, C-13, 48-59.
[3] Mamun, Q.E.K.; Masum, S.M.; Mustafa, M.A.R. Modified Bully Algorithm for Electing Coordinator in Distributed Systems. In Proceedings of the 3rd WSEAS International Conference on Software Engineering, Parallel and Distributed Systems, Salzburg, Austria, 13–15 February 2004.
[4] Gholipour, M.; Kordafshari, M.; Jahanshahi, M.; ahmani, A. A New Approach for Election Algorithm in Distributed Systems. In Proceedings of the Second International Conference on Communication Theory, Reliability and Quality of Service, Colmar, France, 20–25 July 2009; pp. 70–74.
[5] Md. Golam Murshed, Alastair R. Allen*; Enhanced Bully Algorithm for Leader Node Election in Synchronous Distributed System. Computers2012,1,3 23; doi:10.3390/ computers 1010003
[6] B. alhadidi, L. H. baniata, M. H. baniata, M. Al-sharaiah; Reducing Message passing and Time Complexity in Bully Election Algorithms using Two Successors, IJEEE,2013
[7] Yadav, D.K.;Lamba, C.S.;Shukla, S.; A New Approach of leader Election in Distributed System, Sixth International Conference on Software Engineering(CONSEG),2012,Indore, India, p.p. 1-7.
[8] Jain, A.K.; Sharma, R.; Leader Election Algorithm in Wireless Environments Using Fibonacci Heap Structure, IJCTA, 2102.
[9] Tanenbaum, A.S.; Steen, M.V. Distributed Systems Principles and Paradigms; Prentice-Hall.