Crimson Publishers
Wings to the Research

Review Article

Modified Bully Algorithm Incorporating the Concept of Election Commission

Gajendra Sharma*[1] and Anmol Shakya[1]

[1]Department of Computer Science & Engineering, school of Engineering, Department of Computer Science and Engineering, Kathmandu University, Nepal

ISSN: 2694-4391

For HTML Version scan this QR code:

***Corresponding author:** Gajendra Sharma, Department of Computer Science & Engineering, school of Engineering, Department of Computer Science and Engineering, Kathmandu University, Nepal

Submission: April 29, 2018
Published: March 11, 2019

Volume 4 - Issue 5

How to cite this article: Gajendra S, Anmol S . Modified Bully Algorithm Incorporating the Concept of Election Commission. Int J Conf Proc.1(1). ICP.000507.2019.
DOI: 10.31031/ICP.2019.01.000507

Abstract

Distributed computing system comprises of several nodes performing multiple task or same task in parallel. It is usually desirable to have a single node which governs the activity of all these nodes known as coordinator. The purpose of this study is to propose a suitable algorithm for selection of the coordinator in efficient manner. In this work we have developed a new algorithm based on the concept of special type of node, Election Commission (EC), which conducts the activity of selecting the coordinator through election procedure. We have compared out work with the most popular election algorithm, Bully Algorithm, by Garcia Molina and the Improved Bully Algorithm. It was found that our study outperformed other two in context of memory efficiency and message generation in the system. In this work we assumed the EC as an ideal one so for future work we can upgrade the study by assuming the realistic behavior of the EC.

Introduction

Distributed computing system is a very complex system of interconnected processors which need to run different processes as per the requirement. The process that needs to be processed might require different kind of resources which may not be in the local host so the process needs to either migrate the process or request for the resources from the remote location. However, the task of both migrating the process and requesting the resources from the remote location is not an easy task. Many things have to be considered to perform such task correctly and consistently. There need to be cooperation between the different participating processes, if not the process might get corrupted, the integrity of the resource might get compromised, the system might not be consistent, and many other difficulties may arise. Centralized control in distributed system helps to achieve mutual exclusion, time scheduling, load balancing, and synchronization in comfortable way then in ordinary ad-hoc system. Therefore, to perform this task there need to be good coordination between the processes which is maintained by the special process known as coordinator (leader) process.

The coordinator process helps to manage the tasks and maintain the coordination in the distributed system so that the system does not have to suffer from problems like dead-lock, thrashing, loss of consistency, etc. However, for coordinator to perform these tasks, first the coordinator has to be selected. The main role of an elected coordinator is to manage the use of a shared resource in an optimal manner which in turn maintains the coherency of the system even during partial failures. There are different algorithms to select the coordinator process in the distributed system. The most popular among those election algorithms is bully algorithm by Garcia [1]. The general task performed by and election algorithm is selecting one of the live process as the coordinator process which then performs the coordination task in the distributed system.

Distributed system has been a field of interest for many researchers in present time because it has unlocked many possibilities in the field of computing. Cloud computing and grid computing are the good examples of distributed system which has changed the way how computing used to be in past. Although, distributed system has unlocked possibilities, it also has arisen problems like increasing the data traffic in the network, difficult in maintaining consistency, difficulty in data security and integrity, etc. This type of problem is more dangerous in ad-hoc network, since there is no any control on how the task should

be performed so the idea of single coordinator seems better and easy way to manage mess in the network. Selection of coordinator is generally performed by issuing different kind of messages like election message, acknowledgement message, coordinator message etc. [1-5]. In bully algorithm itself, the election process generates messages like election initiation message, OK message and coordinator message [1]. But issuing many messages means more network traffic which degrades the efficiency of the entire distributed system, so it is necessary to minimize the amount of messages as low as possible. Designing a new algorithm for selection of coordinator which issues lesser messages as well as decreases the amount of redundancy message in the network can solve this problem and may raise the overall performance of the system.

Coordinator selection process generally has two basic steps initiating an election process and selecting the coordinator process, however in between these two processes many messages are generated and broadcasted, multi-casted or unicasted. Too much message means more time, more data traffic in the network and more process overhead, which is not desirable. So, the main objective of this paper is to propose such election algorithm which minimizes these problems.

Statement of problem

The bully election algorithm is widely used election algorithm in distributed system however it has certain shortcomings. Although the bully algorithm is easy to implement but its communication complexity is high. In this algorithm there are three types of messages that is issued between electioneer and rest of the participating nodes: election message, OK message and coordinator message. In best case this algorithm has to issue 1 election message and N-1 coordinator message s which is denotes as o(n) where as in worse case this election has to issue $\sum_{i=1}^{N-1} N-i$ election message N-1 OK message and N-1 coordinator message denoted by $o(N^2)$ [3]. It is preferable to achieve the linear communication complexity of the algorithm for better performance of the overall system because less inter-process communication means less traffic and better performance.

Another important problem that many election algorithm faces is when two or more processes detects the coordinator failure simultaneously. In this case most of the algorithm has to restart and this rises redundant election messages, so it is preferable to avoid the problem of simultaneous failure detection. This problem has been somehow minimized in [3] but still there is some more room for improvement which we will discuss later in this paper.

Literature review

Different election mechanism has been proposed to improve the overall performance of the distributed system. Bully election algorithm by Garcia Molina is one of the simplest and widely used election algorithms in the distributed computing system. Many researchers has studied this algorithm and have proposed different version of modification to the original bully algorithm. Since the main pitfall of the original bully algorithm is its runtime complexity $o(N^2)$, most of the modified algorithm has worked upon to resolve this problem, but they have traded off some other performance metrics in return. We will look into some of the modified version of the bully algorithm in this section.

Garcia [1], proposed Bully algorithm for electing the coordinator node in the distributed computing system. This algorithm is one of the most promising and widely used election algorithms even today. In this algorithm the author made assumption that:

a) Each process have a unique priority number

b) Each process have knowledge of the priority number of all other processes in the system

c) The winner of the election is always the alive process with largest priority number

d) A failed process can rejoin the system after its recover

e) And this model is time bounded

This algorithm has following steps:

a) A failure of coordinator node is detected by process P

b) The process p initiates the election mechanism by sending the election message to all the nodes having greater priority number than p

c) If the sending node does not receive the ok message for certain threshold time, the process p elects itself as the coordinator and sends coordinator message to all the nodes in the system.

d) When the receiving node receives the election message from the node with lower priority it send back the ok message to that node and initiates its own election by sending election message to the nodes having higher priority than itself

e) Finally, the election process ceases at some point and only one node will be remaining which will be the coordinator node

f) The coordinator node broadcast the coordinator message to all the nodes in the system and announces its victory

Although this algorithm is simple and easy to implement in distributed system, the main drawbacks of this election algorithm is that it produces lots of message during the election process with time complexity of $o(N^2)$. Each node has to keep the record of all other node in the system which is a data over head if the system has large number of nodes in it. There is also redundancy in election when the older coordinator is recovered; the election has to be initiated to re-instate the older coordinator as new one. Another disadvantage is that if two or more nodes detect the failure simultaneously the whole election process has to restart adding more burden in the system.

Kumar et al. [2], presented a new approach of bully algorithm which minimizes the redundancy in electing the coordinator, reduce the recovery problem of a crashed process and maximize the effectiveness of traditional bully algorithm. The assumptions made in this algorithm by the authors are same as in Bully algorithm. Unlike Bully algorithm this one has 5 different types of messages: election message, OK message, Query message, answer message

and coordinator message. The election algorithm proposed by the authors is as follow

A. Like Bully algorithm, the process detecting the failure sends the election message to all the nodes with higher priority than oneself

B. If no OK message is receiving process P select itself as the coordinator and broadcast the coordinator message to all the nodes

C. If it receives the OK messages from the nodes with higher priorities, it selects the node X with the highest priority number and issues a coordinator message notifying that the coordinator is node X

the query message is issued by any node that has recovered. as soon as the node recovers it issues the query message to all the nodes having higher priority number than itself. if the answer message is received than it will know who the current coordinator is. if it does not receive the answer message than it will know it has the highest priority number, so it issues the coordinator message to rest of the node and elect itself as the new coordinator.

This algorithm is improvement over the Bully algorithm in case of minimizing the redundancy election. However, this algorithm also fails to address the problem when two or more nodes detect the failure node simultaneously. This problem is same for the recovered node too if two or more nodes is recovered at the same time there might be conflict in the result as well. The time complexity of the algorithm is still $o(N^2)$ same as that of Bully algorithm and it still has to maintain the database of the nodes in the system to know the priority of each processes.

Gupta et al. [5], proposed an algorithm that claims to outperform the original Bully algorithm [1] and the modified Bully algorithm [2] in case of minimum memory uses, since in this algorithm the nodes do not have to know the priority information about other nodes in the system. Author also claims that this algorithm is less complex, and it maintain a smaller number of messages passing during the communication. The assumptions made in this algorithm are: Each process has a distinct priority number, no need to know the priority number of other processes, no time bound and the process with highest priority is elected as coordinator. The overview of the proposed algorithm is as follow:

A. When the process P detect the coordinator failure it broadcasts the leader crash message along with its own priority number

B. All the process with higher priority number than that of P should respond with the OK message

C. If the process P does not receive any response from other nodes it elects itself as an coordinator and broadcast the coordinator message

D. When the processes with higher priority level respond to the leader crash message by process P the process P selects the one with the highest priority level as the coordinator and broadcast this to all the nodes in the system

This algorithm is better than previous algorithms in the context of message passing during election process. However, like previous algorithm it has not specified the simultaneous failure detection case. Another problem identified in this algorithm is that, as this algorithm claims to be time unbounded, there is always a possibility that the coordinator is falsely detected as the failure when coordinator is alive but busy to respond. This algorithm has not specified the case with in what circumstances the node is declared failed.

Allen et al. [4], proposed Enhanced Bully algorithm for leader node election in synchronous distributed system which claims to enhance bully algorithm by decreasing the time complexity and minimizing the message passing during election process. Authors have also included the scheme of tie breaker time as the solution to the simultaneous elections initiated by different node, which was not addressed properly in previous works.

In this work authors have proposed a new concept of dividing the entire system in to two sets: Candidate set and Ordinary set. The Ordinary set comprises of nodes having low priority level where as the Candidate set comprises of node having higher priority level. Here all nodes have to have the knowledge of the priority level of all the nodes in the system. In this work the election procedure for different cases like Ideal Case, Candidate Failure Case, Electioneer Failure Case, Simultaneous Election Case and Node Revival Case has been addressed.

Although this algorithm have lots of improvement over the Bully algorithm and other modified versions of Bully algorithm there are still some pitfalls. Here the author have introduced the wait times like, election wait time and ok wait time. This metric might degrade the overall system performance as the system might have to wait even when the job is completed. The dynamic measure to calculate those wait time could have solve this problem.

Soundarabai et al. [6], proposed Improved Bully election algorithm for distributed systems. In this algorithm they have made all the assumption same as that of bully algorithm and additional assumptions that all processes hold an election flag and if this flag is true election cannot be initiated by any process and another assumption that all processes have a variable to store coordinator information. This algorithm claims to solve the problem of simultaneous election as the election flag prevents any node to commence the election when it is true. This algorithm is similar to other modified bully algorithms, the overview of algorithm is states as follow:

A. When the process P detects the failure coordinator it sets it election flag true and broadcast the election message.

B. The receiving nodes sets its election flag to true so that no further election could be commenced

C. All the nodes with higher priority number receiving the election message replies to the process P with OK message

D. The process P finds the highest priority among the received OK message and replies back to that particular node with highest priority informing that it is the new coordinator

E. The new coordinator then check if any node having higher priority is still alive or not

F. If it receives the OK message the new coordinator is that node

G. If it does not receive the OK message than this node broadcast the COORDINATOR message to rest of the nodes and declare itself the coordinator.

Although, this algorithm has improved the Bully algorithm and also has solved the problem of the simultaneous election, it still has increased data burden. All the nodes have to have the knowledge of the priority of other nodes which demand data space in each node resulting in redundant data storage. There is also no any specified time or event that tells the nodes to wait or move on with the procedure. So, it should also have been mention for better reliability.

Theoretical foundation

Coordinator election is very important activity in distributed system. We have already seen some of the coordinator election algorithm in distributed system and there are other many more election algorithms. But when it comes to election in distributed system the Bully algorithm has been a foundation to many election algorithms in distributed system. In this work also we are using the Bully algorithm as the foundation of our algorithm. All the assumption made for the Bully algorithm by Garcia [1] is true except for the condition that each node in the system has to know the priority number of the all the nodes in the system. In this algorithm of our, only the special node called the election commission node has to know the priority level of all the nodes in the network there for this algorithm alleviates the burden of storing the priority level of all the nodes in the system by each node. Another change that is made in our algorithm is that the election process is conducted only by the election commission, which means rest of the nodes in the system need not worry about any kind of the election related task like initiating the election message or broadcasting the coordinator message.

Bully algorithm involves all the nodes of the system in the election procedure which might arise the difficulty in managing the election task when the number of the nodes per system increases. The election is also conducted in several stages by the nodes of the system which increases the election time and productive work has to wait until the election is completed. In best case the election is conducted in single stage when the node having priority just less than the failed coordinator node detects the failure. But when the node with the lowest priority detect the failure the election has to be conducted N-1 times (where N is total number of nodes in the system), which increases the time complexity as the number of nodes grows. But in our algorithm, the node detecting failure with any priority level will only commence the election once saving the valuable time of the system.

We have already mentioned the assumptions made and overview of algorithm in Bully algorithm by Garcia Molina in our previous section; now let's look at an example of how this algorithm is implemented. Let us consider there are 7 nodes in the system with seven priority levels and the coordinator node for time being is node 7. But this node has crashed and the node 2 has detected the failure. The scenario is depicted in the Figure 1.

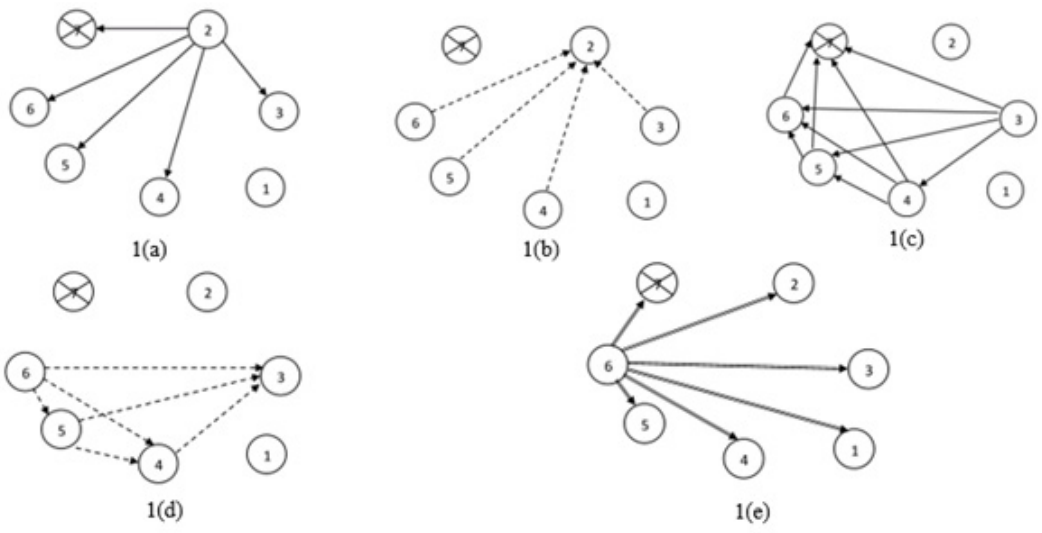

Figure 1:

(a): Node 2 detects the failure and issues ELECTION message to nodes having greater priority level than itself.

(b): Nodes with greater priority level reply to Node 2 with OK message stating their liveliness.

(c): Each node which replied with OK message initiates its own election by sending ELECTION message to nodes with greater priority than oneself.

(d): The alive node reply with OK message.

(e): The only remaining node which did not receive the OK message from higher priority node is new coordinator (Node 6), and finally broadcast the COORDINATOR message to declare its win.

Election commission based modified bully algorithm

In this work we have proposed a new election algorithm based on the Bully algorithm: Election Commission Based Modified Bully (ECBMB) Algorithm. This algorithm implements the same semantics as in Bully algorithm for election process. However, we have made modifications that will enhance the performance of the original Bully algorithm by Garcia Molina.

Assumption for ECBMB algorithm:

A. All the assumption made in original Bully stand true except that each node in the system have knowledge of the priority level of all other nodes

B. Election Commission (EC) node is a special node which conduct the election in the system

C. Only EC node need to know the priority level of all the nodes in the system

D. The alive node having highest priority level will be selected as the coordinator

E. EC node only response to the first node which notifies about the failure for that round

F. EC is assumed to be ideal node and is not susceptible to failure

Here we use 8 types of messages failure detected message, test message, ack message, election message, ok message, coordinator message, query message and answer message Figure 2.

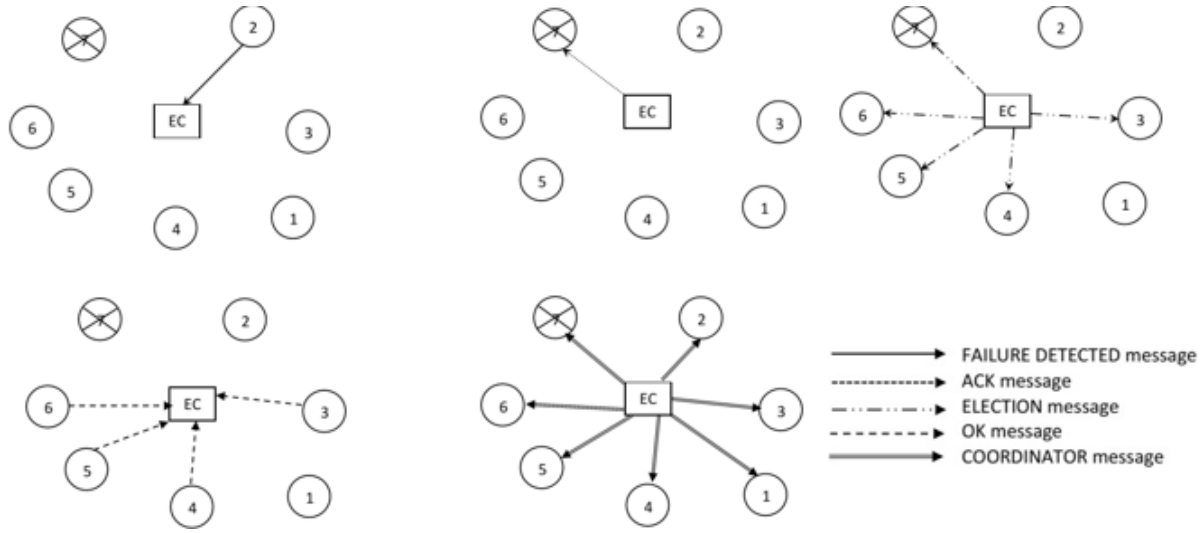

Figure 2: Steps of the algorithm when operated in normal scenario.

Failure detected message: this message is issued by the node which detect the failure of the coordinator. This message is issued to the EC node.

Test message: upon receiving the failure detected message the EC node issues test message to ping if the coordinator has really failed or not.

Ack message: ack message is issued by the EC node to inform that the coordinator has failed or not to the node which issued failure detected message earlier.

Election message: upon detection of failure of the coordinator node and acknowledged the failure detecting node, the EC issues the election message to all the nodes having priority level greater than the node which detected failure.

Ok message: this message is send by all the nodes which received the election message to inform the EC commission that they are alive

Coordinator message: after receiving the ok message the EC finds the node with the highest priority level and broadcast coordinator message to inform the entire node about the new coordinator.

Query message: this message is issued by the node which has recovered to the EC node to know who the current coordinator is.

Answer message: this is issued by the EC node in response to the query message to answer who the current coordinator is. We have designed this algorithm to address 3 basic scenarios: normal election scenario, simultaneous detection scenario and node recover scenario.

Normal election scenario

The steps in normal election scenario are as follow:

A. Start

B. The process P detects the coordinator failure

C. Process P add its priority level and issues FAILURE DETECTED message to EC

D. EC on receiving FAILURE DETECTION message issues the TEST message to confirm the coordinator has really failed

E. If coordinator is still alive:

a) The EC issues ACK message to process P that the coordinator is alive

b) END

F. If coordinator has failed:

a) The EC issues the ELECTION message to all the nodes in the system having higher priority level than process P

G. If EC receives no OK message as reply form any of the nodes

a) Process P is selected as new coordinator

b) EC broadcasts the COORDINATOR message to all the nodes in the system informing P as the new coordinator

c) END

H. IF EC receives OK messages from the nodes

a) The EC computes the highest priority level among the OK message senders

b) EC selects the node with the highest priority as the coordinator

c) EC broadcast the COORDINATOR message to all the nodes in the system about new coordinator

d) END

Simultaneous detection scenario

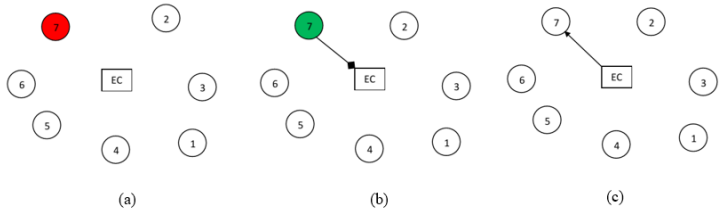

Figure:3

(a): Node 7 is dead.

(b): Node 7 recovered and issued QUERY message.

(c): EC replied with ANSWER message to notify about new coordinator.

When two or more processes detects the failure of the coordinator at the same time, the processes issues the failure detected message to the EC. the EC only respond to any one of the process and ignores the request of rest of the processes. after responding to one of the failure detected message therestoftheprocedure is same as that of normal scenario.

Node recover scenario

When the previously failed node recovers from its failure state, the node has no knowledge about the current coordinator, so it issues the QUERY message to the EC which is responded by the ANSWER message from the EC informing the querying node about the current coordinator of the system. This algorithm has no provision of selecting the recovered node as the coordinator even if the recovered coordinator has the highest priority among all other node. This node has to wait until the next election. This saves redundant election and the also minimizes the generation of unnecessary messages in the system. The example is shown in Figure 3.

Methodology

In this algorithm of ours we have a special node, Election Commission node, which conducts all the election operation. The time taken to perform single transmission is denoted as T_{tx}. The default value of T_{tx} is considered as 200 micro second as per the experiment conducted in [4]. This time is use to test the performance of the each step of the algorithm, for example when the failure is detected by the process P, that process sends the failure detected message to EC and EC performs TEST and then acknowledge the process P which take in total of 4 message so the average time for this operation can be calculated as 200*4=800micro second (Figure 4).

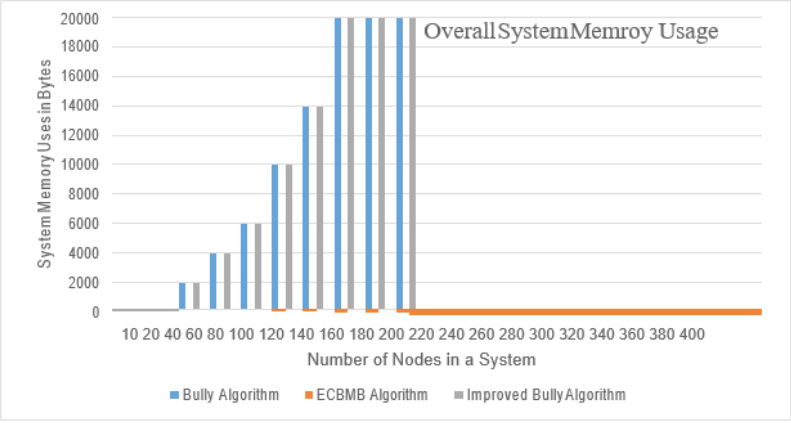

Figure 4: Comparison of memory space occupied by each algorithm.

In order to perform the comparison of our work with other works we first evaluated the basic working equation for each scenarios (Normal, Simultaneous, Node Recovery) and input the variable to calculate the performance metrics like latency and the number of message generated. This type of mathematical operation is performed by writing a simple C-programming code which inputs the variables like the total number of nodes (N), Priority level of current node (R) and default latency of the system. This program outputs the number of message generated in each election and the total time taken to perform the complete selection of the coordinator node.

Once the program is ready, the collection of data for different scenario is relatively easy. After the program is coded the only requirement will be to input the variables which will output the required data that will be used for comparison of our algorithm with the rival one. The main competitor selected for the comparison with our algorithm is the Bully algorithm by Garcia [1] and Improved Bully Election Algorithm [6].

Results

We are going to compare our algorithm with two other algorithms: Bully Algorithm and Improved Bully Election algorithm. The metric that we will be evaluating are Memory/Space utilization, Message generation in network and the latency of the process. Memory space is also one of the important components in any computing system though it is considered less important than time factor. However, the performance of the system is improved with higher free memory so it is always good to utilize the available memory efficiently. It is found that our algorithm outperforms the other two algorithms in using the less memory. This result is depicted in Table 1. It is assumed that the information of each node requires 1byte of memory (Figure 5).

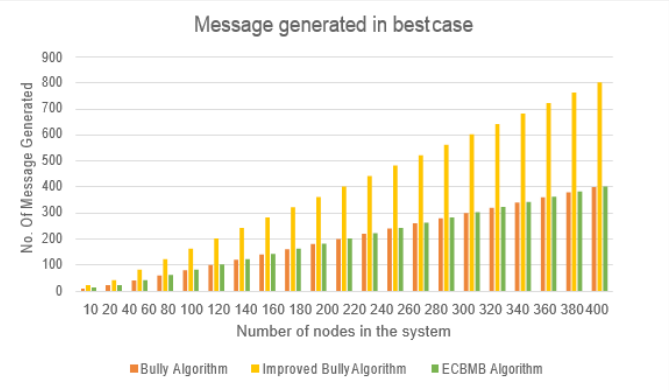

Figure 5: Comparison of message generated by each algorithm in Best case Table.

Table 1: Overall System Memory Usage by 3 Schemes.

	Bully Algorithm	Improved Bully Algorithm	ECBMB Algorithm
Number of nodes in System	Memory Uses in Bytes	Memory Uses in Bytes	Memory Uses in Bytes
10	90	90	10
20	380	380	20
40	1560	1560	40
60	3540	3540	60
80	9860	9860	80
100	9900	9900	100

The number of messages issued for original Bully election during the election in normal scenario is explained by M=(N-R+1) (N-R) + (N-1) [5]. Similarly, the number of message issued foe Improved Bully Algorithm in normal scenario is explained by M=3N-R+2 [6] and for ECBMB is explained by M=3N-2R+1. Where M is number of messages generated, N is the number of the nodes in the system and R is the priority level of the node which detected the failure of the coordinator. The comparative study of message generated in the system by each of the algorithm is depicted in Table 2 for best case and Table 3 for worse case. Case is considered best when R=N-1 and case is considered worse case when R=1 (Figure 6).

Table 2: Message generated in best case by all algorithms.

	Bully Algorithm	Improved Bully Algorithm	ECBMB Algorithm
No. of Nodes in the System	No. of Message Generated	No. of Message Generated	No. of Message Generated
10	11	23	13
20	21	43	23
40	41	83	43
60	61	123	63
80	81	163	83
100	101	203	103

Table 3: Message generated in worse case.

	Bully Algorithm	Improved Bully Algorithm	ECBMB Algorithm
No. of Nodes in the System	No. of Message Generated	No. of Message Generated	No. of Message Generated
10	99	31	29
20	399	61	59
40	1599	121	119
60	3599	181	179
80	6399	241	239
100	9999	301	299

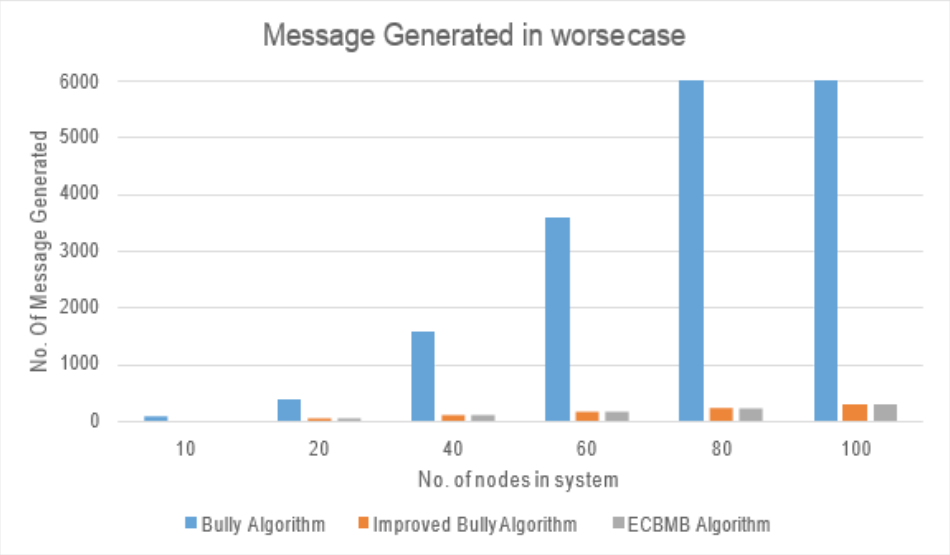

Figure 6: Comparison of message generation by each algorithm in worse case.

We see that ECBMB algorithm generates less message both in best and worst case. It is also seen that Bully algorithm generates highest message in worse case and Improved bully algorithm generates highest message in best case scenario. However, it is seen that improved bully is better than Bully algorithm and it is tough competitor for ECBMB algorithm. In Figure 7 we have made comparison between ECBMB algorithm and the Improved bully algorithm when the number of nodes in the system is N=1000 and the failure detection node are varied.

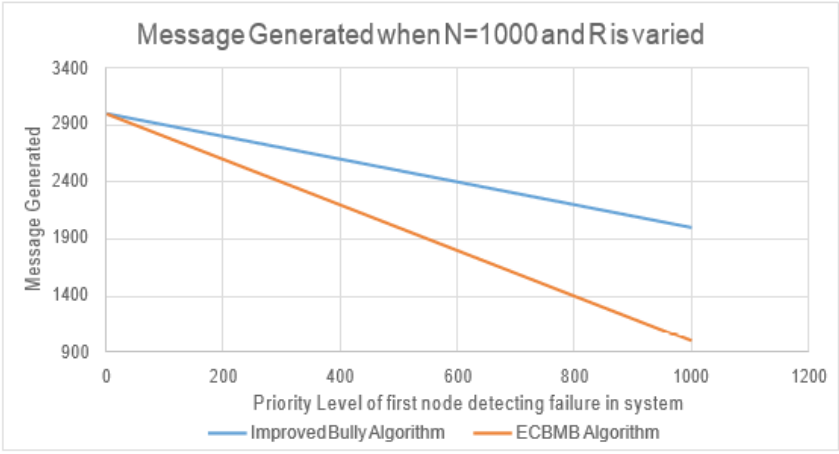

Figure 7: The graph showing comparison between Improved Bully and ECBMB when node is fixed and failure detector is varied.

Comparison between the proposed algorithm and modified bully algorithm using election commission

A Electric leader is required to make synchronization between different processes in distributed network. And different election algorithms are used to elect a coordinator among the available processes in the system such a way that there will be only one coordinator at any time. Bully election algorithm is one of the classical and well-known approaches in coordinator election process. Thus, this paper presented a modified version of bully election algorithm using a new concept called election commission. This approach will not only reduce redundant elections but also minimize total number of elections and hence it will minimize message passing, network traffic, and complexity of the existing system. The new algorithm is more efficient than bully algorithm and modified bully algorithm with respect of message passing, redundant election and network traffic

Discussion

Among the three algorithms compared in previous section it was observed that our proposed algorithm outperforms both other two algorithm in the comparison of the memory usage and the message generated during normal election process. It was found that this algorithm is highly memory efficient algorithm than other two algorithms. Where the memory demand of other algorithms increases exponentially with increase in number of nodes in the system, our algorithm only requires exact amount of memory as the number of nodes in the system. Even while considering the message generated during the election process, our algorithm generates minimum message in both the best case and worse case compared to other two mechanisms.

We know that number of message generated is directly proportional to the latency of the procedure. Since the time complexity of the Bully algorithm is $o(N^2)$ the latency of the system increases exponentially with the increase in the number of nodes in the system. Whereas the time complexity of both the Improved Bully algorithm and ECBMB algorithm are O(N), but still our proposed algorithm outperforms the Improved Bully algorithm when it comes to message generation and it can be clearly seen in Figure 7. The proposed algorithm, ECBMB Algorithm, can be implemented easily in the distributed system for coordination election as it is more memory efficient, faster and generates less data traffic. Other hidden advantage of this algorithm is that it avoid redundant election unlike Bully election, it provide better provision for recovered node and this algorithm can efficiently handle the simultaneous failure detection of the coordinator.

Although, EBMB algorithm is better than original Bully and Improved Bully, it is not the optimum mechanism. In this algorithm we have assumed that EC is ideal one, so further work could be carried out in different issues when EC is not ideal. Here we have not addressed the problem for what happens when the failed node is replaced with the node of same priority and later the failed node recovers. The time bound is another problem that we have not addressed in this work. All these limitation are good areas for the further elaboration of this work.

Conclusion

A. In this paper we came up with the new election algorithm which uses the concept of the Election Commission (EC) which carries out all the election procedure. This algorithm is memory efficient as well as time efficient compared to other two competitor algorithm of original Bully algorithm and Improved Bully algorithm. This algorithm can be implemented with ease in the distributed computing system where coordinator election has to be carried out. This has also improved the time complexity of the original Bully algorithm from $o(N^2)$ to O(N). As the future work, we can focus on operation of the non-ideal EC and the time bound limitation of the algorithm.

References

1. Kohn L, Corrigan J, Donaldson M (1999) To err is human: Building a safer health system. National Academy Press, Institute of Medicine, Washington DC, USA.
2. American Association of Nurse Anesthetists (AANA) (2016) AANA Membership Statistics.
3. Metzner J, Posner K, Lam M, Domino K (2011) Closed claims' analysis. Best Practice & Research Clinical Anesthesiology 25(2): 263-276.
4. Jordan L, Quaraishi J (2015) The AANA Foundation Malpractice Closed Claims Study: A descriptive analysis. AANA Journal 83(5): 318-323.
5. Morris R, MacNeela P, Scott A, Treacy P, Hyde A (2007) Reconsidering the conceptualization of nursing workload: Literature Review. Journal of Advanced Nursing 57(5): 463-471.
6. Weinger M, Reddy S, Slagle J (2004) Multiple measures of anesthesia workload during teaching and nonteaching cases. Anesthesia & Analgesia 98(5): 1419-1425.
7. Hoonakker P, Carayon P, Gurses A, Brown R, McGuire K, et al. (2012) Measuring workload of ICU nurses with a questionnaire survey: The NASA Task Load Index. IIE Transactions on Healthcare Systems Engineering 1(2): 131-143.
8. Leedal J, Smith A (2005) Methodological approaches to anesthetists' workload in the operating theatre. British Journal of Anaesthesia 94(6): 702-709.
9. Young G, Zavelina L, Hooper V (2008) Assessment of workload using NASA-Task Load Index in perianesthesia nursing. Journal of Peri Anesthesia Nursing 23(2): 102-110.
10. Swiger P, Vance D, Patrician P (2016) Nursing workload in the acute-care setting: A concept analysis of nursing workload. Nursing Outlook 64(3): 244-254.
11. Aiken L, Cimiotti J, Sloane D, Smith H, Flynn L, et al. (2011) Effects of nurse staffing and nurse education on patient deaths in hospitals with different nurse work environments. Medical Care 49(12) 1047-1053.
12. Aiken L, Clarke S, Sloane D, Lake E, Cheney T (2008) Effects of hospital care environments on patient mortality and nurse outcomes. Journal of Nursing Administration 38(5): 223-229.
13. Thomas HC, Flynn L (2015) Patient safety culture and nurse-reported adverse patient events in outpatient hemodialysis unit. Research and Theory for Nursing Practice 29(1): 53-65.
14. Aiken L, Sermeus W, Van den Heede K, Sloane DM, Busse R, et al. (2012) Patient safety, satisfaction, and quality of hospital care: Cross sectional surveys of nurses and patients in 12 countries in Europe and the United States. BMJ 344(e1717): 1-14.
15. McMullan SP, Thomas HC, Shirey MR (2017) Certified registered nurse anesthetist perceptions of factors impacting patient safety. Nursing Administration Quarterly 41(1): 56-69.

16. Demaria S, Neustein S (2010) Production pressure, medical errors, and the pre-anesthesia checkout. Middle East Journal of Anesthesiology: 20(5): 631-638.

17. Gaba D, Howard S, Jump B (1994) Production pressure in the work environment: California anesthesiologists' attitudes and experiences. Anesthesiology 81(2): 488-500.

18. Creswell JW, Plano Clark VL (2007) Designing and conducting mixed methods research (3rd edn), Thousand Oaks, Sage, Canada, pp. 2-16.

19. Wisdom J, Creswell JW (2013) Mixed methods: Integrating quantitative and qualitative data collection and analysis while studying patient-centered medical home models. Agency for Healthcare Research and Quality. AHRQ Publication No. 130028-EF. Rockville, Maryland.

20. Hart SG (2006) NASA-Task Load Index (NASA-TLX): 20 years later. Proceedings of the Human Factors and Ergonomics Society 50th Annual Meeting, Santa Monica, Canada, pp. 904-908.

21. Cox KS, Teasley SL, Zeller RA, Lacey SR, Parsons L, et al. (2006) Know staff's "Intent-to-Stay". Nursing Management 37(1): 13-15.

22. Flynn L, Thomas HC, Clarke S (2009) Organizational traits, care processes, and burnout among chronic hemodialysis nurses. Western Journal of Nursing Research 31(5): 569-582.

23. Lacey S, Cox K, Lorfing K, Teasley S, Carroll C, et al. (2007) Nursing support, workload, and intent to stay at magnet, magnet-aspiring, and non-magnet hospitals. Journal of Nursing Administration 37(4): 199-205.

24. Shi J, Mo X, Sun Z (2012) Content validity index in scale development. Journal of Central South University Medical Sciences 37(2): 152-155.

25. Polit D, Beck C, Owen S (2007) Is the CVI an acceptable indicator of content validity? Appraisal and recommendations. Research in Nursing & Health 30(4): 459-467.

26. Lynn M (1986) Determination and quantification of content validity. Nursing Research 35(6): 382-386.

27. Krippendorff K (2018) Content analysis: An introduction to its methodology (4th edn), Thousand Oaks, Sage, Canada.

28. Jones TL, Hamilton P, Murray N (2015) Unfinished care, missed care, and implicitly rationed care: State of the science review. International Journal of Nursing Studies 52(6): 1121-1137.

29. Kalisch BJ, Williams RA (2009) Development and psychometric testing of a tool to measure missed nursing care. Journal of Nursing Administration 39(5): 211-219.

30. Clark J, Lang N (1992) Nursing's next advance: An internal (International) classification for nursing practice. International Nursing Review 39(4): 109-111.