Available online at http://www.journalcra.com

International Journal of Current Research
Vol. 3, Issue, 11, pp.131-133, October, 2011

INTERNATIONAL JOURNAL OF CURRENT RESEARCH

ISSN: 0975-833X

RESEARCH ARTICLE

ELECTION ALGORITHM WHEN CRASH LEADER RECOVERS IN DISTRIBUTED SYSTEMS

[1]Renu Nekkanti and [2]G.L. Aruna Kumari

[1]Department of CST, VITAM College of Engineering, Visakhapatnam, India
[2]Department of Computer Science, Gitam University, Visakhapatnam, India

ARTICLE INFO

Article History:
Received 15th July, 2011
Received in revised form
19th August, 2011
Accepted 23rd September, 2011
Published online 15th October, 2011

Key words:
Coordinator; Election.

ABSTRACT

An important challenge in distributed systems is the adoption of suitable and efficient algorithms for coordinator election. The main role of an elected coordinator is to manage the use of a shared resource in an optimal manner. Among all the algorithms the Bully and Ring algorithms have gained more popularity for coordinator election. Leader election is an important problem in distributed computing systems. In this problem, when the leader is crashed, other nodes must elect another leader. Garcia-Molina's Bully Algorithm is a classic solution to cope with this problem. If the leader crash, the alternative takes, care of the leader's responsibilities. Our results revealed that often, after a leader crash, leader alternative elect as a leader and continue to work. In already existing Election algorithm (also known as Bully Algorithm) proposed by Hector Garcia-Monila (1982). In my paper I proposed about what happens if the crashed leader recovers .

INTRODUCTION

Selecting Coordinator in Bully algorithm is difficult in distibuted computing systems. Depending on a network topology, many algorithms have been presented for electing leader in distributed systems [1, 2, 3, 4,5] . The highest-ranking auditor selects 'audit Coordinator' which responsible for detection of failures. A disadvantage of this approach is that, if multiple node failures (or partitioning), it may take many sequential executions of the promotion protocol before a candidate is successful in reaching coordinator rank [5]. Bully algorithm is one of the most applicable elections algorithms that was presented by Garcia Molina in 1982[1]. In this paper, we initially explain bully algorithm and discuss the drawbacks of it and present algorithm considering about what happens if the crashed leader recovers. When a node (e. g. node N) realizes that the leader is crashed, it sends an ELECTION message to all nodes with higher numbers. If no one responds, node N wins the election and becomes leader. If one of the higher processes answers, it takes over. Node N's job is done. In case of receiving a message, the receiver sends an OK message back to the sender to indicate that he is alive and will take over. The receiver then holds an election, unless it is already holding one. Eventually, all nodes give up but one and that one is the new leader [1]. The effectiveness of an algorithm depends on the validity of the assumptions that are made. In distributed mutual exclusion environment certain assumptions must be considered to make the work successful. In my paper I proposed about what happens if already crash leader recovers .

***Corresponding author:** maddipatirenu@gmail.com
chandran.aruna@gmail.com

Modified Bully algorithm when crash leader recovers

As it has been mentioned in Section 1 in Bully algorithm number of messages that should be exchanged between processes is high. Therefore this method imposes heavy traffic on the network. To cope with this drawback we will present an optimized method by modifying it such that our method intensively decreases the number of messages that should be exchanged between processes when crash leader recovers. In existing algorithm, we use orderd nodes composing coordination group {coordinator, alternative1, alternative2,… alternativek}. These nodes are used for preventing global election between all nodes. Initially using the modified election algorithm described in Section b-2, a process with greatest number is selected as coordinator. Similarly other (k-1) processes of coordination group which have the next priority numbers are selected asalternative1, alternative2,.. alternative K. Then their process id(s) are sent to all processes.When process P notices that the coordinator is crashed, sends crash- leader message to alternative1 informing coordinator is crashed.

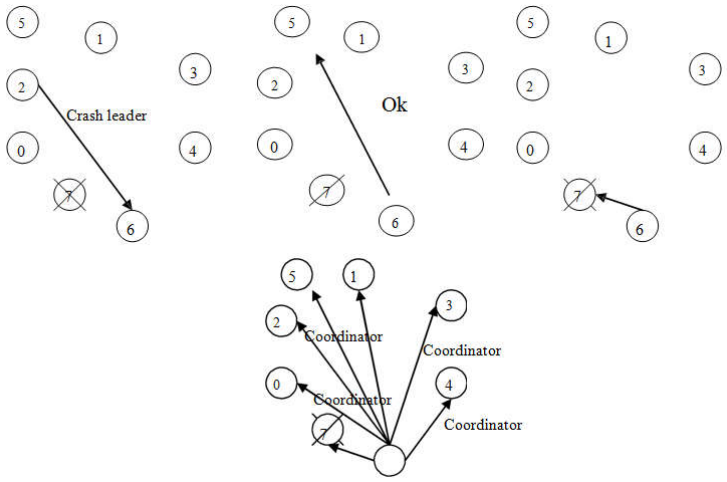

Fig. 1: coordinator 7 is crashed and the other node in coordinator group 6 takes the role

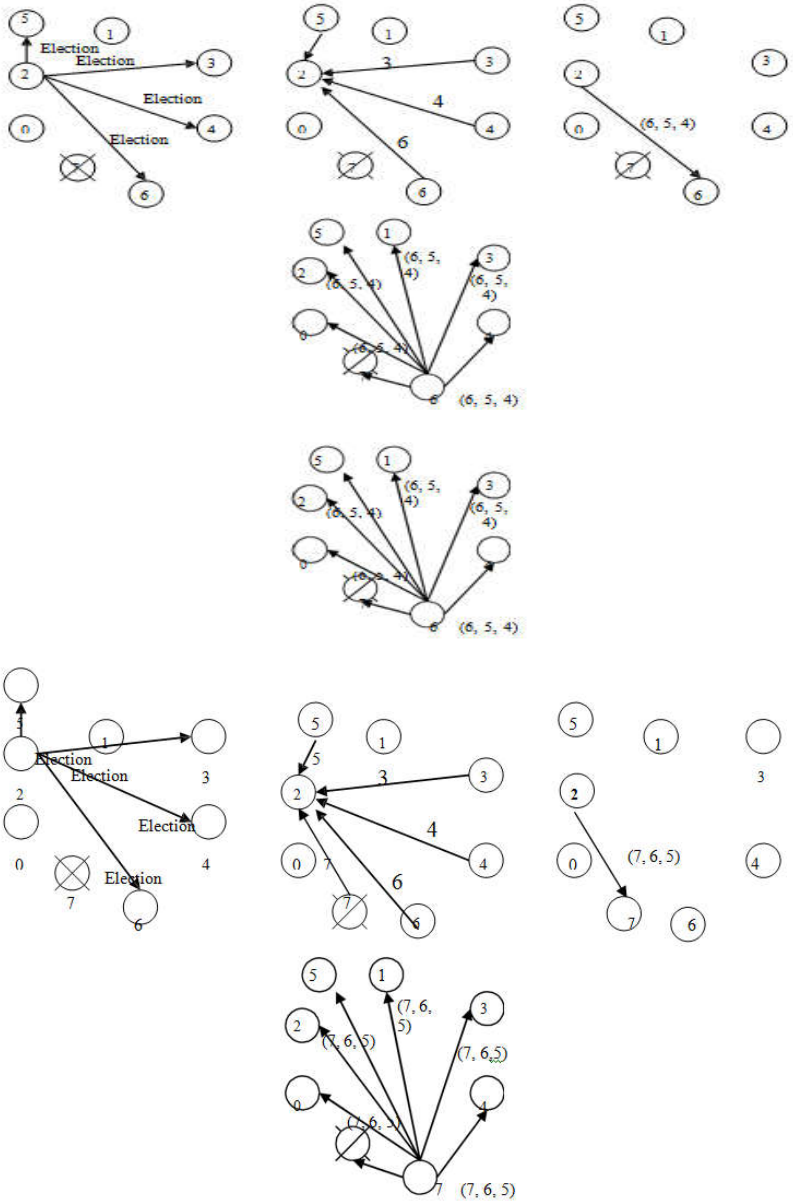

Figure 2: The modified election algorithm 1)process 4and 2 find out the crashed coordinator simultaneously and therefore each of which send ELECTION messages separately.(2)other processes that receive more than one ELECTION message should only send their own

A) ***If alternative1 is alive****:* Alternative1 sends ok message to process p informing it from receiving the message. Then it sends a message to coordinator in order to be sure from its crashing. In case of receiving ok from coordinator, it obtains that the process p made a mistake and the algorithm is finished. Otherwise alternative1 is selected as coordinator and broadcasts leader message and in this message introduce the process p as itssubstitution.

B) ***If alternative1 is not alive****:*Process p waits for t time step for receiving ok from alternative1 (t=2d; d is average ofpropagation delay). If p does not receive any message, sends crash- leader to other alternatives in order. If all alternatives are crashed, then process p runs the modify election algorithm as follows: Afetr some time t if crash coordinator recovers then the node sents cocordinator message to coordination group

Step1- When process P notices that all members of the coordinator group have crashed, it initiates an election algorithm.

Step2- When the process P finds out the coordinator is crashed, sends ELECTION message to all other processes with higher priority number.

Step3- Each process that receives ELECTION messages (with higher process than P) sends OK message with its unique priority number toprocess P.

Step4*-a)* If no process responses to process P,In this case p is selected as coordinator. Then it sends a message to processes with lower number in order to select other members of the coordination group and waits 2d time steps. After receiving all process ids, it selects k-1 higher process as members of coordination group and it will broadcast one Coordinator message to all processes, declaring coordinator group member.

Step4*-b)* If some process response to process P:the process P will select the coordinator group*i.e.* process with the highest priority number as coordinator and k higher processes as altarnative1 and alternative 2,…alternative k, then it sends to the new coordinator the GRANTmessage

Step5- at this stage the coordinator process will broadcast a message to all other processes that contains its priority number and alternative1 and alternative 2,…alternative k numbers.(coordinator group numbers) .

Step 6:If mean while crashed coordinator recovers then the node sents cocordinator message to coordination group

It is clear that if process P crashes after sending ELECTION message to higher processes, or crashes after receiving the priority numbers from process with higher priority number, higher process wait at most 3D time for coordinator broadcast. (D is average propagation delay), If it will not receive, this process runs the modified algorithm. If a process with higher priority number crashes after sending its priority number to P, process P sends GRANT message to it meaning that it is the highest process and P waits for broadcasting coordinator message. If after D time, process P doesn't receives the COORDINATOR message, it repeats the algorithm again. Therefore we can use this algorithm as an efficient and safe method to selecting the coordinator.*If multiple processes simultaneously realize thatthe coordinator is crashed*When more than one process or all processesfind out the coordinator has crashed simultaneously, there is no problem if at least one candidate is alive. But if coordination groupis crashed all of them run in parallel election algorithm, therefore heavy traffic imposed to the network. For solving this problem in modified election algorithm we act as follow.

Step1-When process P realizes that the all members of the coordinator group have crashed, it initiates modified election algorithm

Step2-When process P ' (P ' may be P) receives the ELECTION message from process or processes with lower priority number compare to itself, it waits a short time that can be specified perfectly and then answers to the process with lowest priority number only. In this situation if P=P' (This process initiates the algorithm and also received the ELECTION message from other processes), then stops the algorithm.

Step3-After process P' answered to P, if P 'receives an ELECTION message from process R(R<P<P'), P' answers to process R by sendingits priority number and sends STOP message to process P.(Figure 4)

Step4*- when a* process receives the STOP message stops the algorithm immediately.

Step5- if process p neither receives any response from other process(es), nor does it receive any ELECTION message form processes with lower Priority number, then in this case it can inform other processes containing it(P) as COORDINATOR.

Conclusion

Many algorithms related to the leader election are proposed We proposed a new algorithm for leader election in adhoc networks. The main idea is to overcome the difficulty is when crash leader recovers

REFERENCES

[1] A new approach for election algorithm in distributed systemc, 2009. Second International Conference on Communication Theory, Reliability, and Quality of Service

[2] N. Fredrickson and N. Lynch, 1987. Electing a Leader in a Synchronous Ring." *J.ACM*, vol.34, no.1, pp.98-115.

[3] Le Lann, G., "Distributed Systems – Towards a Formal Approach", in Information Processing 77, B. Gilchrist, Ed. Amsterdam, The Netherlands: North-Holland, pp. 155-160, 1977.

[4] S. Park, Y. Kim and J.S. Hwang, 1999. An Efficient Algorithm for Leader-Election in Synchronous Distributed Systems, IEEE TENCON.

5] Kim, J. L. and Belford, Geneva G., 1988. A robust, distributed election protocol", Proc. of seventh IEEE Computer Soc. Symp. Reliable Distributed Systems, pp. 54-60, Columbus, Ohio.

[6] Kim, W., "AUDITOR: A Framework for high availability of DB/DC systems", IBM Res. Rep. RJ3512.