MODIFIED BULLY ELECTION ALGORITHM IN DISTRIBUTED SYSTEMS

M. S. KORDAFLSHARI, M. GHOLOPIOUR, M. MOSAKHANI, A. T. HAGHIGHAT, M. DEHGHAN

Department of Electrical, Computer & IT, Islamic Azad University, Qazvin Branch, Qazvin, Iran

Abstract: - Leader election is a fundamental problem in distributed computing, and regards a wide number of applications. In order to solve this problem, it is possible and convenient to exploit the topological properties of the specific distributed systems, so to reduce time and message complexity. In this paper we explain Garcia Molina’s bully algorithm and indicate the problem of this algorithm and modify the bully algorithm. We show that modify algorithm is more efficient than the Garcia Molina’s one in the time, message complexity and in the fewer stages.

Key-word: - Bully algorithm, modified Bully algorithm, election, distributed systems, Message complexity and coordinator.

1 Introduction
A leader is a member or $n$ nodes that all other nodes acknowledge as being distinguished to perform some special task. The leader election problem is the problem of choosing a leader from given set of coordinator. Each node has a unique ID and the set of IDs is a totally ordered set with respect to the usual comparison operation. A node’s ID can be expressed in log $n$ bits. The problem of leader election is fundamental in the operation of distributed systems. A distributed system is a collection of Autonomous computing nodes, which can communicate with each other and which cooperate on a common goal or task. A leader performs a centralized coordination after being selected. This may be necessary in some problems where a completely distributed solution is either not available or offers less attractive performance. Whenever some failure occurs it is necessary for the nodes to adapt to the new conditions so that they can continue working. This requires some kind of reorganization. It is precisely these elections we describe in this paper from an algorithmic point of view.

Election algorithms can be used to start up a system initially or remove nodes from the systems. Leader election has been studied extensively in different models and networks [2,3,4,5,6]. Bully algorithm is one of the most applicable elections Algorithms that was presented by Garcia Molina in 1982[1]. In this paper, we discuss the drawback of synchronous Garcia Molina’s Bully algorithm and modify it with an optimal message algorithm. We show that our algorithm is more
efficient than Garcia Molina’s Bully algorithm, because of fewer message passing and fewer stages.

In future work we will implement our algorithm with asynchronous model in order to decrease number of message passing in asynchronous bully algorithm. The rest of paper is organized as follows: In section 2 improved method for solving Bully algorithm drawbacks is presented. In section 3 Garcia Molina’s bully algorithm and our modified algorithm are compared. Section 4 presents the conclusion of the paper. Finally in the last section we explain future work.

2 Modified Bully Algorithm

As has been mentioned in Bully algorithm[1] number of messages that should be exchanged between processes is high. Therefore this method imposes heavy traffic in network. For solving this drawback we will present optimized method by modifying the Bully algorithm, that intensively decreases the number of messages that should be exchanged between processes. Furthermore the number of stages is decreased from at most five stages to at most four stages.

Our algorithm has the following steps: (figure 1)

**Step1**- When process P notices that the coordinator has 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 to

![Diagram of modified Bully election algorithm](image)

**Fig.1:** The modified Bully election algorithm (a) process 3 and 2 find out the crashed coordinator simultaneously and therefore each of which send ELECTION messages separately, (b) other processes that receive more than one ELECTION message should only send their own priority number to process with lowest id number (in this case 2), (c,d) process 3 stop the algorithm because receiving the ELECTION message from process 2, process 2 continue the algorithm.
process P.

**Step4-** If no process responds to process P, it will broadcast one COORDINATOR message to all processes, declaring itself as a coordinator. If some process response to process P by comparing the priority numbers, the process P will select the process with the highest priority number as coordinator and then sends to it the GRANT message.

**Step5-** At this stage the coordinator process will broadcast a message to all other processes and informs itself as a coordinator.

**Step6-** Immediately after the process with higher number compare to coordinator is up, our algorithm is run.

New algorithm not only has all advantages of Bully algorithm also it doesn’t have the drawback of Bully algorithm (high number of message passing). Furthermore maximum number of stages is decreased from five stages to four stages.

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 receive the COORDINATOR message, it repeats the algorithm again.

Therefore we can use this algorithm as an efficient and safe method to selecting the coordinator.

### 2.1 A novel solution for a drawback of Bully algorithm

In Bully algorithm when more than one process or all processes find out the coordinator has crashed simultaneously, all of them run in parallel Bully algorithm, therefore heavy traffic imposed to the network.

For solving this problem in modified Bully algorithm we act as follow (figure2).

![Fig. 2: An example of use Stop message in MBA](image-url)
In point of number of stages Bully algorithm always is executed in five stages, while our algorithm find out the coordinator after four stages.

### 3.1 Analytical comparison of two algorithms if only one process detects the crashed coordinator

If only one process detects crashed coordinator

- $n$: The number of processes
- $r$: The priority number of processes that find out the crashed coordinator

In bully modified algorithm the number of messages passing between processes for performing election is obtained from the following formula:

$$N_{(r)} = 2(n-r) + n$$  \hspace{1cm} (1)

Which has Order $o(n)$. In the worst case that is $r = 1$ (process with lowest priority number finds out the crashed coordinator):

$$N_{(1)} = 2(n-1) + n = 3n - 1$$   \hspace{1cm} (2)

Whereas the number of message passing between processes in the Bully algorithm for performing election is obtained from the following formula:

$$N_{(r)} = (n-r+1)(n-r) + n - 1$$  \hspace{1cm} (3)

In the worst case that is $r = 1$ (process with lowest priority number detects crashed coordinator):

$$N_{(1)} = n^2 - 1$$ \hspace{1cm} (4)

Which has Order $O(n^2)$

Number of messages in modified bully algorithm will be equal to $3n - 1$ that obviously means this modified algorithm is better than bully algorithm with fewer messages passing and the fewer stages.

Figure 3 clearly shows the comparison between bully algorithm and modified bully algorithm.
when one process finds out the crashed coordinator). Horizontal axis indicates the priority number of processes that find out crashed coordinator, and vertical axis indicates the number of message passing. For example if the number of processes is 100 and 2th process finds out the crashed coordinator, in bully algorithm the number of message passing is equal 9801 but the number of message passing in modified bully algorithm is equal 296.

### 3.2 Analytical comparison of two algorithms if set of $S = \{r_1, r_2, ..., r_m\}$ run the algorithm simultaneously.

Now assume that the set of processes in $S = \{r_1, r_2, ..., r_m\}$ from processes find out the crashed coordinator concurrently ($r_i$ is lowest process):

In bully algorithm the number of message passing between processes for performing election is obtained from the following formula:

$$T = (n - r_i + 1)(n - r_i) + n - 1$$  \hspace{1cm} (5)

In our modified algorithm the number of message passing between processes for performing election is obtained from the following formula:

$$T = (n - r_i) + \sum_{\{r_j|r_j \in S\}} (n - r_j) + n$$  \hspace{1cm} (6)

In bully algorithm the number of message passing is based on the process with lowest priority number. That means there isn’t any difference between state that only process $r_1$ detects the crashed coordinator and state that in witch the set of $S = \{r_1, r_2, ..., r_m\}$ detects crashed coordinator.

But in modified algorithm set of $S = \{r_1, r_2, ..., r_m\}$ is also important. If the priority numbers of the processes that detects the crashed coordinator is higher, the number of message passing will be decreased considerably.

### 4 Conclusion

In this paper, we discussed the drawbacks of Garcia Molina’s Bully algorithm and then we presented an optimized method for the Bully algorithm called modified bully algorithm. Our analytical simulation shows that our algorithm is more efficient rather than the Bully algorithm, in both number of message passing and the number of stages, and when only one process run the algorithm message passing complexity decreased from $O(n^2)$ to $O(n)$ (formula 1,3). In this analysis we consider the worst case in modified algorithm. Result of this analysis clearly shows that modified algorithm is better than bully algorithm with fewer message passing and the fewer stages.

### 5 Future work

In future work we will implement our algorithm with asynchronous model in order to decrease number of message passing in asynchronous bully algorithm.
Procedure INITIATE (P);
{ if (p priority number > coordinator priority number) FIND OUT (P); } //-----------------------------------------

main (void)
{ 
  While (TRUE)
  { 
    Case of event:
      "BOOT" : INITIATE (P);
      "FIND OUT " : FIND OUT (P);
      "DEFAULT " : ACTION ;
  }
  //-----------------------------------------

Fig.4: pseudo code

6 References