# A DYNAMIC ELECTION STRATEGY IN DISTRIBUTED SYSTEM

Heta Jasmin Jhaveri
Computer Science and Engineering
Government Engineering college
Gandhinagar, India

Prof. Sanjay Shah
Computer Science and Engineering
Government Engineering college
Gandhinagar, India

*Abstract*— In distributed system, electing a leader for the various coordination activities is an important challenge. The coordinating activities can be a directory search, balancing the load of the distributed system,etc.The major coordinating activity is to manage the use of a shared resource in an optimal manner.
The goal of a leader election in distributed system of autonomous processes is to select one of the currently alive processes as a leader so as to manage the coordination activities of the other processes in the system. Such an election is done at the start of the distributed system and it is required every time when a current coordinator fails or crashes.
Among many algorithms reported in the literature, the Bully and Ring algorithms have gained the utmost popularity. This paper proposes a dynamic way of electing a coordinator process based on the machine having the set of available resources.

**Keywords- Election, Coordinator, Resource characteristics.**

## I. INTRODUCTION

A distributed system is a collection of processors interconnected by a communication network in which each processor has its own local resources like memory, etc. and the communication between them is held by message passing over the communication network [1].

Distributed algorithms require that there be a leader process in the entire system that performs some type of coordination activity needed for the smooth running of other processes in the system. As the processes in the system need to interact with the leader process, they all must know who the current leader is. The important case to be dealt is when the leader fails, so there is a requirement for a temporary new leader to take the job of coordination. Hence an election is done to find out the next leader in the system.

Leader election is a fundamental problem in the network of distributed systems. The election process is initiated when one or more processes discover that the leader has failed, and it terminates when the remaining processes know who the new leader after the result of election declared is.

Election algorithms are based on the following assumptions:

1. Each process in the distributed system has a unique priority number.
2. Whenever an election is supposed to be held, the process which has the highest priority number among the currently active process is elected as the new coordinator.
3. On recovery, a failed process can later take appropriate actions in order to rejoin the set of active processes.

## II. EXISTING ELECTION ALGORITHMS

Much research has been carried out in the election area of distributed systems. Among the prominent algorithms in the literature, below are the two algorithms as listed :
(1) Hector Garcia-Molina,(1982; also known as Bully Algorithm).
(2) Silberschatz and Galvin (1994; also called Ring algorithm).

Both the above algorithms assume that all the processes have a unique priority and all the processes know the priority of every other processes in the system. Also that the process with the highest priority among the currently alive processes will be the candidate for the new leader in case when the leader has failed due to some reason.

## III. OVERVIEW OF DYNAMIC ELECTION STRATEGY

This paper presents an election strategy that is based on the resources currently available on the machine where the processes are running.

The strategy is to find out a process which is running on a machine with the richest set of resource characteristics available at the time of the election. The resource characteristics of a machine may include the level of security provided by the machine, number of processors in a multiprocessor system, speed of each processor, available RAM on the machine, etc. Thus in addition to reducing the messaging overhead among the processes, the resource characteristics is an important criteria in electing a process as a leader.

The main goals of the proposed algorithm strategy are:

(1) To achieve a greater overall improvement in system performance at a reasonable cost.
(2) To utilize the available resources most efficiently.
(3) To have the ability to modify the system itself in accordance with any changes.
(4) To find an optimum leader whose machine is enriched with the best resource characteristics.

*Architecture of the distributed system.*
Consider a distributed system of m resources and n processes. Every resource is equipped with its resource characteristics such as :

- Number of processors on each machine
- Speed of each process
- RAM
- Security level provided
- OS
- Architecture

The distributed system is organized as follows : One or more processing elements (PE) (i.e.CPU) are created with different speeds measured in Million Instructions per Sec (MIPS) rating. Then one or more PEs are grouped together to create a machine. Similarly one or more machines are grouped together to create a Grid Resource. Thus, a Grid Resource can be a single processor, shared memory multiprocessors or a distributed memory cluster of computers. Figure 4.1 shows the organization of a distributed network with a grid resource.

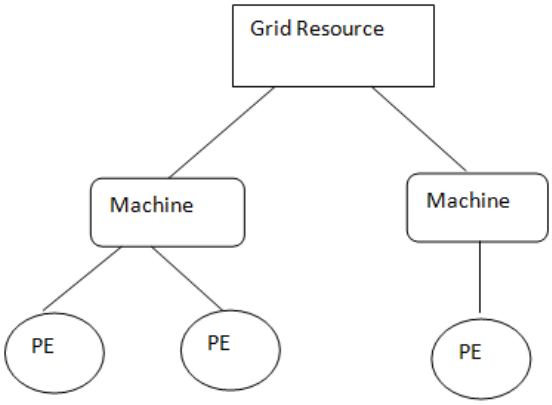


Figure 1. Organization of a Distributed network with a grid resource

**Resource Factor of a machine**
Let α0, α1,α2,…….αn be some weighted values given to the resource characteristics corresponding to the resources of a machine and let r0,r1,r2,……..rn be the values of the available resources on that machine such that,
**α0 > α1 > α2 > …….. αn …..……(1.1)**
ranked according to the priority of resource characteristics of each machine and each αi, 0≤i≤n, for n characteristics, in the range [0-1] and
**α0 + α1 + α2 + …….+ αn ……(1.2)**

Then resource factor of a machine is calculated as :

**Resource factor (RF)**
**= r0 * α0 + r1*α1 + …+rn* αn ….. …….(1.3)**

Thus for every such j machine out of m machines of the system, the resource factor RF for each machine j is :

**for j=0 to m, i =0 to n,**

$$RF_j = \sum_i [\![(ri*\alpha i]\!] \text{ ……..……(1.4)}$$

The machine with the highest resource factor is a machine with the highest resources. We assign a weighted value for each resource characteristic as follows :

Security : 0.4
No. of PEs : 0.3
MIPS of PEs : 0.2
Free RAM : 0.1

The resource characteristic with the highest weighted value is considered to be of higest priority and then with the decreasing weighted values, decreases the priorities.
Note that the values sum up to 1 and all the values are in the range [0,1].

1. The security of each machine is rated in 5 categorized as follows :

- Unsecured : 0.0

- Fairly Secured : 20.0
- Good Secured: 40.0
- Quite Good Secured : 60.0
- Highly Secured : 80.0

2. No. of PEs indicate the total number of processing elements that a machine has.
3. MIPS rating indicates the capability of each PE to execute the number of million instruction of a job in 1 second.
4. RAM indicates the available free RAM within a machine.

Every process's length is defined in terms of MIs(Million Instructions) and each processors speed by MIPS (Million Instructions per second). Thus,

Actual CPU Time Requirement by a Process Pi

$$= \frac{\text{MIs of I}}{\text{MIPS on whi}}$$

**……..(1.5)**

## IV. DYNAMIC ELECTION STRATEGY

All the resources along with their resource characteristics are registered with a GIS, which keeps track of all the machines. As and when the processes are created, they are sent for execution according to the time shared system under consideration.

Now, among the currently running processes, one process is designated as a coordinator for a special role to coordinate the processes of the system. The selection of this coordinator process is important as it has an important role of the working system.

Consider that a process Pi is the coordinator at time t1, $0 \leq i \leq k$, out of total k processes. At a later time t2, a process Pj $0 \leq j \leq k$ sends a REQUEST message to the coordinator process Pi. But Pj does not receive a reply from coordinator Pi till Pj's timeout period.

$$\mathbf{T_{timeout} > MaxT_{send} + MaxT_{reply}}$$

Where,

$MaxT_{send}$ : Maximum time taken to send a message from source to destination

$MaxT_{reply}$ : Maximum time taken to receive a message from destination to source

This condition should be satisfied in order to avoid unnecessary timeouts before the reply reaches from source to destination.

At this point, process Pj discovers that the coordinator Pi has failed due to some reason and it is the time to elect a new coordinator. Thus, the proposed algorithm is executed which runs as follows :

The resource characteristics of each and every resources in the system are fetched from the GIS along with the current status of every process running on the which PE. The resource factor of all the machines is calculated (using equation 1.4). The machine with the highest resource factor say $m_k$ is chosen. If the only process running on the machine $m_k$ is process Pi which has just crashed, then the machine with 2nd highest resource factor is chosen, other wise one of the processes except process Pi is chosen on machine mk.

For multiple processes running on the chosen machine, the process's CPU time requirement is calculated(using equation 1.5) and the one with the least CPU is elected as the new coordinator. This message is broadcasted to all the other processes by process Pj about the new status. Thus the process running on the richest set of resource characteristics and the least CPU time requirement is chosen as the new leader.

Consider a system with 5 grid resources, each with a machine and each machine may have one or more PEs.also consider that currently 8 processes are in execution. Let the current coordinator be process with id 6 and the status of the other processes be as shown in table 2 .

Table 1. Processes in the dynamic election strategy

| Process ID | Status |
|---|---|
| 1 | NORMAL |
| 2 | NORMAL |
| 3 | NORMAL |

| | |
|---|---|
| 4 | NORMAL |
| 5 | NORMAL |
| 6 | COODINATOR |
| 7 | NORMAL |
| 8 | NORMAL |

The status of all the processes except process 6 is NORMAL, which means they are alive in the system. Process 6 is the current coordinator. Now suppose at some point in time, a process 2 sends a REQUEST message to process 6, but does not receive a reply till its time out. Thus process 2 discovers that the coordinator with process id 6 has crashed and so it is the time for an election. So the simulation results when the algorithm is run with the given architecture of the distributed system calculates the resource factor of all 5 grid resources.

Resource 1:
Resource ID : 5
Resource Factor : 152.9
Process ID : 1
Process ID : 6
---
Resource 2:
Resource ID : 9
Resource Factor : 302.0
Process ID : 2
Process ID : 7
---
Resource 3:
Resource ID : 13
Resource Factor : 91.6
Process ID : 3
Process ID : 8
---
Resource 4:
Resource ID : 17
Resource Factor : 212.9
Process ID : 4
---
Resource 5:
Resource ID : 21
Resource Factor : 61.7
Process ID : 5

Thus, the resource 2 with resource id 9 has the highest resource factor RF = 302.0., and the processes running on that resource are PID 2 and PID 7. Thus now with more than one processes on the same resource, their CPU time requirement is calculated using equation 1.5. let their CPU time requirements be :
Process ID 2 :
CPU time requirement : 5.46 sec
Process ID 7 :
CPU time requirement : 4.42 sec

Thus the process with process ID 7 with the minimum CPU time requirement is chosen as the new coordinator. A new message is broadcasted to all the processes informing about the new coordinator as shown in table 2 :

Table 2 Broadcast message about the new coordinator

| Process ID | Status |
|---|---|
| 1 | NORMAL |
| 2 | NORMAL |
| 3 | NORMAL |
| 4 | NORMAL |
| 5 | NORMAL |
| 6 | FAILED |
| 7 | COODINATOR |
| 8 | NORMAL |

All the processes receive this message and then coordinates with the new coordinator there on.
Pseudo code of the algorithm is given below :

**DynamicElection**

```
Begin
    Do
        Send(Pj, REQUEST,C)
        setTimer(Timeout_period)
        msg ← receive (C, REPLY)
        if(msg!=null)
            continue
        else if (msg = null AND Pj timeout)
            ;coordinator crashed
            ;Pj contacts GIS
            send (Pj, RES,GIS)
            data ← receive(GIS,ResChar)
            maxRF ← 0
            for i = 1 to n
                RF_i ← 0
                for j = 1 to r
                RF_i ← RF_i+r_ij*w_j
                end-for
            If maxRF < RF_i
                maxRF ← RF_i
                res ← i
            end-if
            end-for
            minCPU ← ∞
            for i = 1 to k processes on machine res
            CPU_i ← length(Pi) / Speed_res
            If minCPU > CPU_i
                minCPU ← CPU_i
                Leaderprocess ← i
            End-if
            End-for
        sendBROADCAST("New Coordinator"& Leaderproc)
        end-if
until forever
End.
```

## V. CONCLUSION

This paper presents a dynamic election strategy in order to elect a leader when a leader of a coordinator fails. The main goal of this strategy is to utilize the resources of the distributed system in its most efficient way. Thus the process running on a machine that has the best resources available at the time of election will be designated as a new leader. This process can then coordinate and manage the activities in the network with its rich set of resources.

## REFERENCES

[1] Sinha P.K, Distributed Operating Systems Concepts and Design, Prentice-Hall of India private Limited, 2008.

[2] Tanenbaum A.S Distributed Operating System, Pearson Education, 2007.

[3] [Garcia – Molina 1982] "Elections in a distributed computing system", IEEE transactions on computers, vol C-31, No 1,pp 48-59.

[4] "A Leader election algorithm for Clustered Groups" – 2007 IEEE

[5] S.Park, Y.Kim and J.S.Hwang, "An Efficient Algorithm for LeaderElection in Synchronous Distributed Systems," 1999 IEEE TENCON.

Article in a journal:

[6] "An Efficient Approach of Election Algorithm in Distributed Systems" - Sandipan Basu / Indian Journal of Computer Science and Engineering (IJCSE), Vol 20, No 1, 2010

[7] [Fredrickson and Lynch 1987] Fredrickson, and Lynch, "Electing a Leader in a synchronous Ring", journal of the ACM, Vol 34, pp 98-115.

[8] GridSim: a toolkit for the modeling and simulation of distributed resource management and scheduling for Grid computing - Rajkumar Buyya and Manzur Murshed