LAB 7

Aim: To implement Election Algorithms

Lab Outcome:
Implement techniques for Election Algorithms.

Theory:
In distributed systems, election algorithms are used to elect a leader or coordinator node among a group of nodes. The leader node is responsible for coordinating the activities of other nodes, managing resources, and making decisions on behalf of the group.

There are various election algorithms used in distributed systems, and the choice of algorithm depends on the requirements of the system, such as fault-tolerance, scalability, and availability.

One commonly used election algorithm is the Bully algorithm, which is a centralized algorithm. In this algorithm, the node with the highest priority becomes the leader, and if the leader fails, the next highest priority node takes over.

Another algorithm is the Ring algorithm, which is a decentralized algorithm. In this algorithm, nodes are arranged in a ring, and each node sends a message to its neighbor indicating its willingness to become the leader. The node with the highest priority becomes the leader.

A third algorithm is the Paxos algorithm, which is fault-tolerant and can handle network failures. In this algorithm, nodes propose a value to be chosen as the leader, and if the majority of nodes accept the proposal, the value is chosen as the leader.

These election algorithms ensure that a leader is elected in a fair and efficient manner, and they provide fault-tolerance and availability in distributed systems.

Bully Algorithm
The bully Algorithm proposed by Garcia-Molina follows the following algorithm

- When a process notices that the coordinator is no longer responding to requests, it initiates an election.
- A process, P, holds an election as follows:
  1. P sends an ELECTION message to all processes with higher numbers
  2. If no one responds, P wins the election and becomes the coordinator
  3. If one of the higher-up's answers, it takes over. P's job is done
- If a process can get an ELECTION message from one of its lower-numbered colleagues.

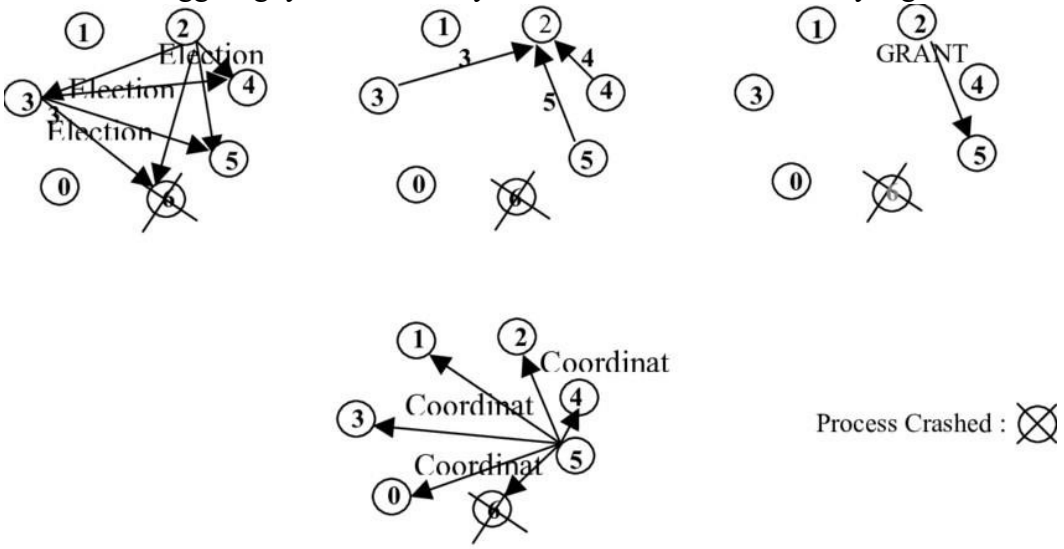
  The message arrives => the receiver sends an OK message back to the sender to indicate that he is alive and will take over.

  The receiver then holds an election, unless it is already holding one.
- Eventually, all processes give up but one does not give up and that one is the new coordinator.

It announces its victory by sending all processes a message telling them that starting immediately it is the new coordinator.

- If a process that previously went down came back up, it holds an election. If it happens to be the highest-numbered process currently running, it will win the election and take over the coordinator's job.

· Thus, the biggest guy in town always wins, hence the name "bully algorithm."



(Code & Output:)

```
import java.io.*;
import java.util.Scanner;

class Main {
    static int n;
    static int pro[] = new int[100];
    static int sta[] = new int[100];
    static int co;

    @SuppressWarnings("resource")
    public static void main(String args[]) throws IOException {
        System.out.println("Enter the number of process");
        Scanner in = new Scanner(System.in);
        n = in.nextInt();
        for (int i = 0; i < n; i++) {
            System.out.println("For process " + (i + 1) + ":");
            System.out.println("Status:");
            sta[i] = in.nextInt();
            System.out.println("Priority");
            pro[i] = in.nextInt();
        }

        System.out.println("Which process will initiate the election?");
        int ele = in.nextInt();
        elect(ele);
        System.out.println("Final coordinator is " + co);
    }

    static void elect(int ele) {
        ele = ele - 1;
```

```
        co = ele + 1;
        for (int i = 0; i < n; i++) {
            if (pro[ele] < pro[i]) {
                System.out.println("Election message is sent from " + (ele + 1) + " to "
+ (i + 1));
                if (sta[i] == 1)
                    elect(i + 1);
            }
        }
    }
}
```

OUTPUT:

```
dt_ws\.codes_2b1a4ee1\bin' 'Main'
Enter the number of process
7
For process 1:
Status:
1
Priority
1
For process 2:
Status:
1
Priority
2
For process 3:
Status:
1
Priority
3
For process 4:
Status:
1
Priority
4
For process 5:
Status:
1
Priority
5
For process 6:
Status:
1
Priority
6
For process 7:
Status:
0
Priority
7
```

```
Which process will initiate the election?
4
Election message is sent from 4 to 5
Election message is sent from 5 to 6
Election message is sent from 6 to 7
Election message is sent from 5 to 7
Election message is sent from 4 to 6
Election message is sent from 6 to 7
Election message is sent from 4 to 7
Final coordinator is 6
PS C:\Users\krisc\Documents\.Notes\.codes>
```

Conclusions :

1. Understood and learnt the concept of Election Algorithms and significance of a coordinator in a system.
2. Implemented the Bully Election Algorithm using Java.
3. The coordinator is elected using the Bully algorithm by considering the highest process id in the network.

Postlab Questions:

1. What is the role of a coordinator in Distributed systems?
2. Compare Bully and Ring algorithms.