Name : Aniket Inamdar
Roll no. : PE 25 Batch : E1 Panel : E PRN: 1032201692

AIM : To study and implement Bully Election Algorithm

OBJECTIVES :

1. To understand the working of election algorithms
2. To simulate bully algorithm

THEORY :

Elections –

Elections are a crucial step in the Bully Algorithm's procedure for choosing a coordinator or leader for a distributed system. Elections are held in order to guarantee that a coordinator is always in place to oversee system functioning.

Bully Algorithm –

In distributed systems, the Bully Algorithm is a mechanism for leader elections in which several processes vie to choose a coordinator. A process starts an event when it notices that there isn't a coordinator or that it wants to take over as the coordinator. Election through contesting higher-priority processes (lower process IDs). When a timeout passes and no higher-priority process responds, the initiator takes over as coordinator. If not, it gives way to the procedure with a higher priority. This guarantees that a coordinator oversees the procedures in the event of a breakdown as well.

Types of messages -
There are basically 3 types of messages :

1. An election message to initiate the election
2. A reply/response message given in response to the election message
3. A coordinator message sent to inform other processes, the id of the coordinator process.

INPUT : Process IDS

OUTPUT : Selected Coordinator Process

PLATFORM: UNIX

PROGRAMMING LANGUAGE : C Language

CONCLUSION: Thus, bully algorithm is successfully implemented.

Name : Aniket Inamdar
Roll no. : PE 25 Batch : E1 Panel : E PRN: 1032201692

FAQS

1. What is the time complexity (best,avg,worst)of bully algorithm?

Ans.

- Best Case: O(1) in which there is no election required and the coordinator has the highest process ID at first.
- Average Case: In random process distribution across the network, this is usually O(n).
- Worst Case: O(n^2), when an election is started by the lowest ID and all processes exchange messages in full.

2. Why do we have to elect the coordinator process?

Ans. In distributed systems, we choose a coordinator process to make sure that nodes coordinate and communicate effectively. By centralising decision-making and minimising conflict, the coordinator improves system reliability and fault tolerance by facilitating tasks like resource allocation, distributed computation, and data consistency.

3. How did the name of "Bully" approach come up?

Ans. The "bully" approach in the distributed computing context derives its name from the behaviour exhibited by nodes in the system. When a node detects the failure of the current leader, it "bullies" its way to assume leadership by challenging other nodes through message exchanges to assert its dominance and become the new leader.

Code :

```
#include <iostream>
#include <bits/stdc++.h>
using namespace std;

class node
{
public:
    int id;
    int coordinator;
    bool isAlive;
    bool electionStarted;
    bool message_back;
    bool visited;
    node *next = NULL;

    node()
    {
        this->isAlive = true;
        this->electionStarted = false;
```

Name : Aniket Inamdar
Roll no. : PE 25 Batch : E1 Panel : E PRN: 1032201692

```
    }

    node(int i, bool c)
    {
        this->id = i;
        this->coordinator = c ? i : -1;
        this->isAlive = true;
        this->electionStarted = false;
        this->message_back = false;
        this->visited = false;
    }

    node *startProcess(int total_nodes)
    {
        int i_d;
        bool isC;
        node *p = NULL;
        node *start = NULL;

        for (int i = 1; i <= total_nodes; i++)
        {
            cout << "Enter id for process " << i << ": ";
            cin >> i_d;
            cout << "Is process " << i_d << " a coordinator (1/0): ";
            cin >> isC;
            node *curr = new node(i_d, isC);

            if (p == NULL)
            {
                start = curr;
            }
            else
            {
                p->next = curr;
            }
            p = curr;
        }
        return start;
    }

    void display_processes(node *n)
    {
        node *head = n;
        while (head != NULL)
        {
            cout << "Process id : " << head->id << endl;
            cout << "Is coordinator : " << (head->coordinator != -1 ? "Yes" :
"No") << endl;
```

Name : Aniket Inamdar
Roll no. : PE 25 Batch : E1 Panel : E PRN: 1032201692

```
            cout << "Election Started : " << (head->electionStarted ? "Yes" :
"No") << endl;
            cout << "Is Alive : " << (head->isAlive ? "Yes" : "No") << endl
                 << endl;
            head = head->next;
        }
    }

    void kill_node(int i_d, node *n)
    {
        node *head = n;
        while (head != NULL)
        {
            if (head->id == i_d)
            {
                head->isAlive = false;
            }
            head = head->next;
        }
    }

    void startElection_bully(int total_nodes, node *start_node, node *coppy)
    {

        if (start_node == NULL)
        {
            return;
        }
        node *s = start_node;
        if (!s->electionStarted)
        {
            s->electionStarted = true;
            for (int i = s->id; i < total_nodes; i++)
            {
                cout << start_node->id << " sent message to " << i + 1 <<
endl;
            }

            s = s->next;
            bool flag = false;

            while (s != NULL)
            {
                if (s->isAlive)
                {
                    cout << s->id << " sent ok back to " << start_node->id <<
"\n";
                    flag = true;
```

Name : Aniket Inamdar
Roll no. : PE 25 Batch : E1 Panel : E PRN: 1032201692

```
            }
            s = s->next;
        }
        if (flag)
        {
            start_node->message_back = true;
        }
        if (!start_node->message_back)
        {
            cout << "New coordinator is " << start_node->id << endl;
            while (coppy != NULL)
            {
                coppy->coordinator = start_node->id;
                coppy = coppy->next;
            }
        }
        if (start_node->next == NULL)
        {
            return;
        }
        s = start_node;
        while (s->next != NULL)
        {
            if (s->next->isAlive)
            {
                startElection_bully(total_nodes, s->next, coppy);
            }
            s = s->next;
        }
    }
}
void startElection_ring(int total_nodes, node *start_node)
{
    int ans = INT_MIN;
    int num_visited = 0;
    node *current_node = start_node;
    while (current_node != nullptr && !current_node->visited)
    {
        current_node->visited = true;
        num_visited++;

        ans = max(ans, current_node->id);

        node *next_node = current_node->next;
        while (next_node != nullptr && !next_node->isAlive)
        {
            next_node = next_node->next;
        }
```

Name : Aniket Inamdar
Roll no. : PE 25 Batch : E1 Panel : E PRN: 1032201692

```
            if (next_node != nullptr && !next_node->visited)
            {
                cout << current_node->id << " Sent election e" <<
current_node->id << " to " << next_node->id << endl;
            }
            current_node = next_node;
        }

        node *reset_node = start_node;
        while (reset_node != nullptr)
        {
            reset_node->visited = false;
            reset_node = reset_node->next;
        }

        cout << "New coordinator is " << ans << endl;
    }

};

int main()
{
    cout << "Election Algorithms (Bully and Ring) : " << endl;
    node x;
    node *start_node = NULL;
    int choice = 0;
    int num_nodes = 0;
    do
    {
        cout << "Menu :" << endl;
        cout << "1. Add nodes" << endl;
        cout << "2. Disable a node" << endl;
        cout << "3. Display nodes" << endl;
        cout << "4. Start Bully algorithm election" << endl;
        cout << "5. Start Ring algorithm election" << endl;
        cout << "6. Exit" << endl;
        cin >> choice;

        switch (choice)
        {
        case 1:
            cout << "Number of nodes to be added : ";
            cin >> num_nodes;
            start_node = x.startProcess(num_nodes);
            break;

        case 2:
            if (start_node != NULL)
```

Name : Aniket Inamdar
Roll no. : PE 25 Batch : E1 Panel : E PRN: 1032201692

```
            {
                int node_idx_to_kill;
                cout << "Enter node to disable: ";
                cin >> node_idx_to_kill;
                x.kill_node(node_idx_to_kill, start_node);
            }
            else
            {
                cout << "No nodes added yet " << endl;
            }
            break;

        case 3:
            if (start_node != NULL)
            {
                x.display_processes(start_node);
            }
            else
            {
                cout << "No nodes added yet!" << endl;
            }
            break;

        case 4:
            int node_id;
            cout << "Enter node id that starts the election\n";
            cin >> node_id;
            for (int i = 0; i < node_id - 1; i++)
            {
                start_node = start_node->next;
            }
            x.startElection_bully(num_nodes, start_node, start_node);
            break;

        case 5:
            x.startElection_ring(num_nodes, start_node);
            break;

        default:
            cout << "Enter valid choice" << endl;
            break;
        }
    } while (choice != 6);

    return 0;
}
```

Name : Aniket Inamdar
Roll no. : PE 25 Batch : E1 Panel : E PRN: 1032201692

Output :

- **Bully Algorithm**

```
PS D:\MIT-WPU\Final-Year\Sem8\DC\LCA2> cd "d:\MIT-WPU\Final-Year\Sem8\DC\LCA2\" ; if ($?) { g++ main.cpp -o main } ; if ($?) { .\main }
Election Algorithms (Bully and Ring) :
Menu :
1. Add nodes
2. Disable a node
3. Display nodes
4. Start Bully algorithm election
5. Start Ring algorithm election
6. Exit
1
Number of nodes to be added : 10
Enter id for process 1: 1
Is process 1 a coordinator (1/0): 0
Enter id for process 2: 2
Is process 2 a coordinator (1/0): 0
Enter id for process 3: 3
Is process 3 a coordinator (1/0): 0
Enter id for process 4: 4
Is process 4 a coordinator (1/0): 0
Enter id for process 5: 5
Is process 5 a coordinator (1/0): 0
Enter id for process 6: 6
Is process 6 a coordinator (1/0): 0
Enter id for process 7: 7
Is process 7 a coordinator (1/0): 0
Enter id for process 8: 8
Is process 8 a coordinator (1/0): 0
Enter id for process 9: 9
Is process 9 a coordinator (1/0): 0
Enter id for process 10: 10
Is process 10 a coordinator (1/0): 1
Menu :
1. Add nodes
2. Disable a node
3. Display nodes
4. Start Bully algorithm election
5. Start Ring algorithm election
6. Exit
2
Enter node to disable: 7
Menu :
1. Add nodes
2. Disable a node
3. Display nodes
4. Start Bully algorithm election
```

Name : Aniket Inamdar

Roll no. : PE 25 Batch : E1 Panel : E PRN: 1032201692

```
5. Start Ring algorithm election
6. Exit
2
Enter node to disable: 10
Menu :
1. Add nodes
2. Disable a node
3. Display nodes
4. Start Bully algorithm election
5. Start Ring algorithm election
6. Exit
3
Process id : 1
Is coordinator : No
Election Started : No
Is Alive : Yes

Process id : 2
Is coordinator : No
Election Started : No
Is Alive : Yes

Process id : 3
Is coordinator : No
Election Started : No
Is Alive : Yes

Process id : 4
Is coordinator : No
Election Started : No
Is Alive : Yes

Process id : 5
Is coordinator : No
Election Started : No
Is Alive : Yes

Process id : 6
Is coordinator : No
Election Started : No
Is Alive : Yes

Process id : 7
Is coordinator : No
Election Started : No
```

Name : Aniket Inamdar

Roll no. : PE 25 Batch : E1 Panel : E PRN: 1032201692

```
Process id : 8
Is coordinator : No
Election Started : No
Is Alive : Yes

Process id : 9
Is coordinator : No
Election Started : No
Is Alive : Yes

Process id : 10
Is coordinator : Yes
Election Started : No
Is Alive : No

Menu :
1. Add nodes
2. Disable a node
3. Display nodes
4. Start Bully algorithm election
5. Start Ring algorithm election
6. Exit
4
Enter node id that starts the election
4
4 sent message to 5
4 sent message to 6
4 sent message to 7
4 sent message to 8
4 sent message to 9
4 sent message to 10
5 sent ok back to 4
6 sent ok back to 4
8 sent ok back to 4
9 sent ok back to 4
5 sent message to 6
5 sent message to 7
5 sent message to 8
5 sent message to 9
5 sent message to 10
6 sent ok back to 5
8 sent ok back to 5
9 sent ok back to 5
6 sent message to 7
6 sent message to 8
```

Name : Aniket Inamdar

Roll no. : PE 25 Batch : E1 Panel : E PRN: 1032201692

```
6 sent message to 9
6 sent message to 10
8 sent ok back to 6
9 sent ok back to 6
8 sent message to 9
8 sent message to 10
9 sent ok back to 8
9 sent message to 10
New coordinator is 9
Menu :
1. Add nodes
2. Disable a node
3. Display nodes
4. Start Bully algorithm election
5. Start Ring algorithm election
6. Exit
6
Enter valid choice
PS D:\MIT-WPU\Final-Year\Sem8\DC\LCA2>
```

- **Ring Algorithm**

```
PS D:\MIT-WPU\Final-Year\Sem8\DC\LCA2> cd "d:\MIT-WPU\Final-Year\Sem8\DC\LCA2\" ; if ($?) { g++ main.cpp -o main } ; if ($?) { .\main }
Election Algorithms (Bully and Ring) :
Menu :
1. Add nodes
2. Disable a node
3. Display nodes
4. Start Bully algorithm election
5. Start Ring algorithm election
6. Exit
1
Number of nodes to be added : 5
Enter id for process 1: 1
Is process 1 a coordinator (1/0): 0
Enter id for process 2: 2
Is process 2 a coordinator (1/0): 0
Enter id for process 3: 3
Is process 3 a coordinator (1/0): 0
Enter id for process 4: 4
Is process 4 a coordinator (1/0): 0
Enter id for process 5: 5
Is process 5 a coordinator (1/0): 1
Menu :
1. Add nodes
2. Disable a node
3. Display nodes
4. Start Bully algorithm election
5. Start Ring algorithm election
6. Exit
2
Enter node to disable: 5
Menu :
1. Add nodes
2. Disable a node
3. Display nodes
4. Start Bully algorithm election
5. Start Ring algorithm election
6. Exit
5
1 Sent election e1 to 2
2 Sent election e2 to 3
3 Sent election e3 to 4
New coordinator is 4
Menu :
1. Add nodes
2. Disable a node
```