DISTRIBUTED SYSTEMS

PROJECT-II

TEAM MEMBERS:

Ajay Venkatesha (1001861936)

Sayali Dilip Deshmukh(1001966628)

DECLARATION:

I have neither given nor received unauthorized assistance on this work.
Ajay Venkatesha: Worked on Part 1 & Part 2
Sayali Dilip Deshmukh: Worked on Part 1 & Part 2
Worked together on Readme file and Report

Part 1:

Berkeley's Clock Synchronization algorithm

1) Berkeley's Algorithm is a distributed systems clock synchronization method. The technique makes the assumption that neither each machine node in the network has a reliable time source nor has a UTC server.

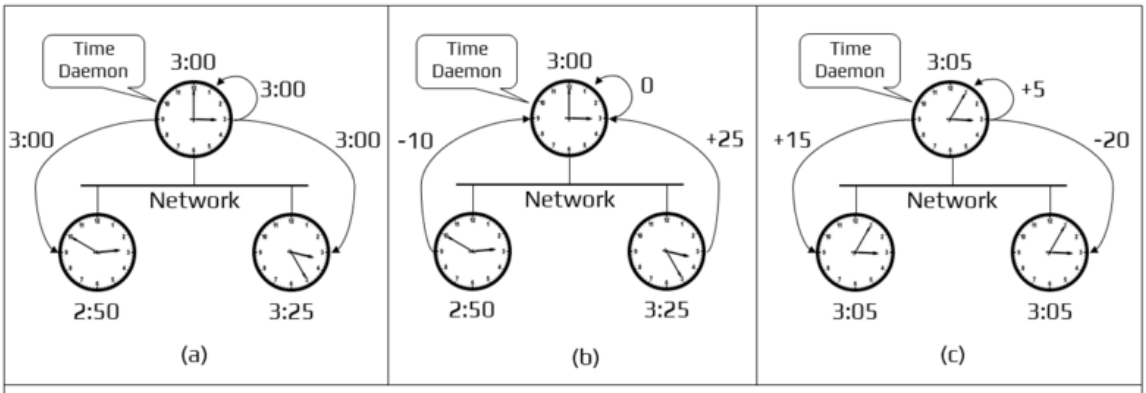
Algorithm

2) A certain node is selected as the network's master node from a pool of nodes. The other nodes in the network serve as slaves to this main node, which serves as a master. An election procedure or leader election algorithm is used to select the master node.

3) The master node uses Cristian's technique to ping subordinate nodes periodically and retrieves their clock times.

4)The average time difference between all the clock times received and the clock time provided by the master's system clock is calculated by the master node. The current time at the master's system clock is increased by this typical time difference before being transmitted over the network

5)Below diagram shows the scientific working of this algorithm:



(a) The time daemon asks all the other machines for their clock values.
(b) The machines answer.
(c) The time daemon tells everyone how to adjust their clock.

SNAPSHOTS OF HOW WE HAVE IMPLEMENTED IT:

```
~/Desktop/ProjectDS — -bash                    ~/Desktop/PROJECTDS — -bash ...
Last login: Sat Jul 16 20:53:12 on ttys000

The default interactive shell is now zsh.
To update your account to use zsh, please run `chsh -s /bin/zsh`.
For more details, please visit https://support.apple.com/kb/HT208050.
Sayalis-MacBook-Air:~ sayalideshmukh$ cd Desktop
Sayalis-MacBook-Air:Desktop sayalideshmukh$ cd ProjectDS
Sayalis-MacBook-Air:ProjectDS sayalideshmukh$ g++ server.cpp -o server -std=c++11
Sayalis-MacBook-Air:ProjectDS sayalideshmukh$ g++ client.cpp -o client -std=c++11
Sayalis-MacBook-Air:ProjectDS sayalideshmukh$ ./server
Hey!Sever has been started...
Server Local CLock -  1.000000

Server is listening...

Start the client processes.

Number of clients connected -  1
New client accepted!Updated number of connected clients - 1
To add more clients press 0
To continue with process, press 1
0
Continue creating new clients


Number of clients connected -  2
New client accepted!Updated number of connected clients - 2
To add more clients press 0
To continue with process, press 1
1

Clients creation finished
Total number of connected clients - 2

Requesting all the clients for their local clock value:
Receiving local clock of client  -  3.000000
Receiving local clock of client  -  2.000000


After applying Berkeley's algorithm
The updated local clock of this server is -  2.000000


Sayalis-MacBook-Air:ProjectDS sayalideshmukh$
```

```
Client process started.
Local clock of this client is -  3.000000

Client is connected to the server!


Message from SERVER - Requesting your local clock value


Sent message to the server with local clock value - 3.000000

Message from SERVER - The offset of your clock is -   minus 1.000000

Change the value of this client's clock by offset:  minus 1.000000
Local Clock of this client is - 2.000000

Sayalis-MacBook-Air:ProjectDS sayalideshmukh$
```

```
Sayalis-MacBook-Air:PROJECTDS sayalideshmukh$ ./client
Client process started.
Local clock of this client is -  2.000000

Client is connected to the server!


Message from SERVER - Requesting your local clock value


Sent message to the server with local clock value - 2.000000

Message from SERVER - The offset of your clock is -   minus 0.000000

Change the value of this client's clock by offset:  minus 0.000000
Local Clock of this client is - 2.000000

Sayalis-MacBook-Air:PROJECTDS sayalideshmukh$
```

---

2)Part 2:

A distributed system can construct a partial ordering of events and identify causality violations using the algorithm known as the vector clock. These clocks extend Scalar time and determine if a contributing event has contributed to another event in the distributed system in order to enable a causally coherent representation of the distributed system. In essence, it encompasses all the causal connections. This approach enables us to mark each process in the system with a vector (a list of numbers) containing an integer for each local clock. So there will be a vector or array of size N for each of the N processes specified.

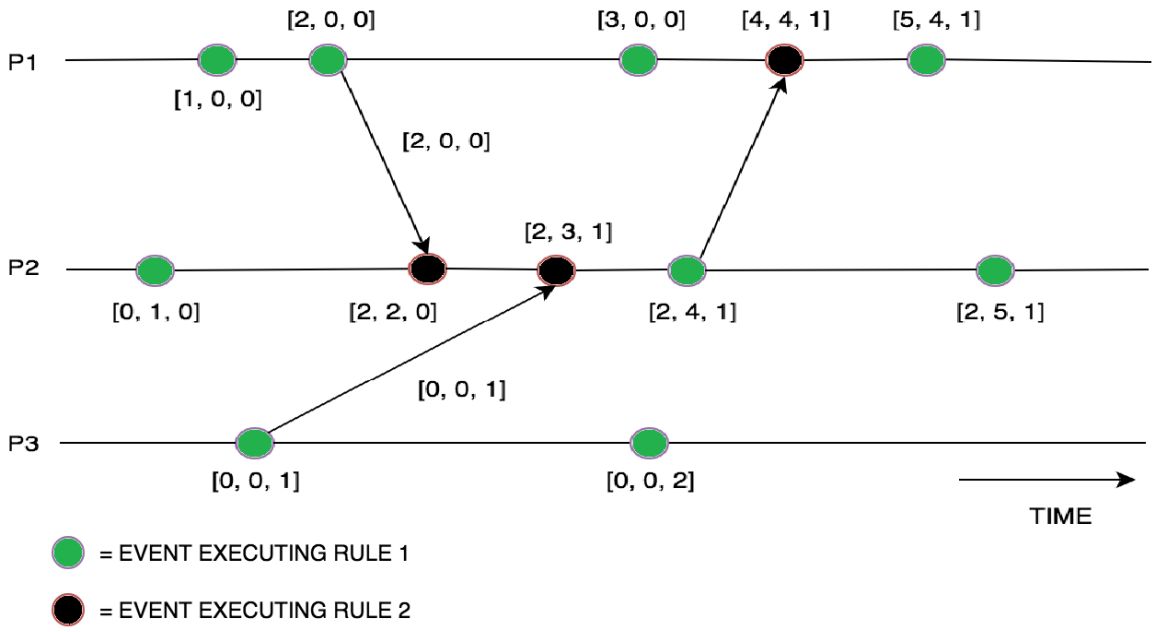
What is the vector clock algorithm's operation?

1) All the clocks are initially set to zero.

2) The value of a process's logical clock in the vector is increased by one whenever an internal event happens in the process.

3) Each time a process sends a message, its logical clock's value in the vector is also increased by one.

Every time a process receives a message, its logical clock's value in the vector is increased by one. In addition, each element is updated by summing the values in the received message's vector and the process's own vector clock (for every element).



SNAPSHOTS:

```
Last login: Sat Jul 16 21:00:39 on ttys004

The default interactive shell is now zsh.
To update your account to use zsh, please run `chsh -s /bin/zsh`.
For more details, please visit https://support.apple.com/kb/HT208050.
Sayalis-MacBook-Air:~ sayalideshmukh$ cd Desktop/ProjectDS
Sayalis-MacBook-Air:ProjectDS sayalideshmukh$ g++ CausalOrdered.cpp -o CausalOrdered -lpthread
Sayalis-MacBook-Air:ProjectDS sayalideshmukh$ ./CausalOrdered 1
Successfully binded socket & port!
Broadcasting messsages to every processes simultaneously -  : [1,0,0]
Message received from the process: 2, client clock received : [1,1,0], Before Sending Message: : [1,0,0], After Sending Message: [1,1,0]
Broadcasting messsages to every processes simultaneously -  : [2,1,0]
Message received from the process: 3, client clock received : [1,1,1], Before Sending Message: : [2,1,0], After Sending Message: [2,1,1]
Broadcasting messsages to every processes simultaneously -  : [3,1,1]
Message received from the process: 2, client clock received : [1,2,0], Before Sending Message: : [3,1,1], After Sending Message: [3,2,1]
Message received from the process: 3, client clock received : [2,2,2], Before Sending Message: : [3,2,1], After Sending Message: [3,2,2]
Broadcasting messsages to every processes simultaneously -  : [4,2,2]
Message received from the process: 2, client clock received : [2,3,1], Before Sending Message: : [4,2,2], After Sending Message: [4,3,2]
Message received from the process: 3, client clock received : [3,2,3], Before Sending Message: : [4,3,2], After Sending Message: [4,3,3]
Message received from the process: 3, client clock received : [3,3,4], Before Sending Message: : [4,3,3], After Sending Message: [4,3,4]
Message received from the process: 2, client clock received : [3,4,2], Before Sending Message: : [4,3,4], After Sending Message: [4,4,4]
```

```
The default interactive shell is now zsh.
To update your account to use zsh, please run `chsh -s /bin/zsh`.
For more details, please visit https://support.apple.com/kb/HT208050.
Sayalis-MacBook-Air:~ sayalideshmukh$ cd Desktop/ProjectDS
Sayalis-MacBook-Air:ProjectDS sayalideshmukh$ ./CausalOrdered 2
Successfully binded socket & port!
Message received from the process: 1, client clock received : [1,0,0], Before Sending Message: : [0,0,0], After Sending Message: [1,0,0]
Broadcasting messsages to every processes simultaneously -  : [1,1,0]
Broadcasting messsages to every processes simultaneously -  : [1,2,0]
Message received from the process: 3, client clock received : [1,1,1], Before Sending Message: : [1,2,0], After Sending Message: [1,2,1]
Message received from the process: 1, client clock received : [2,1,0], Before Sending Message: : [1,2,1], After Sending Message: [2,2,1]
Broadcasting messsages to every processes simultaneously -  : [2,3,1]
Message received from the process: 3, client clock received : [2,2,2], Before Sending Message: : [2,3,1], After Sending Message: [2,3,2]
Message received from the process: 1, client clock received : [3,1,1], Before Sending Message: : [2,3,2], After Sending Message: [3,3,2]
Broadcasting messsages to every processes simultaneously -  : [3,4,2]
Message received from the process: 3, client clock received : [3,2,3], Before Sending Message: : [3,4,2], After Sending Message: [3,4,3]
Message received from the process: 3, client clock received : [3,3,4], Before Sending Message: : [3,4,3], After Sending Message: [3,4,4]
Message received from the process: 1, client clock received : [4,2,2], Before Sending Message: : [3,4,4], After Sending Message: [4,4,4]
```

```
Sayalis-MacBook-Air:ProjectDS sayalideshmukh$ ./CausalOrdered 3
Successfully binded socket & port!
Message received from the process: 1, client clock received : [1,0,0], Before Sending Message: : [0,0,0], After Sending Message: [1,0,0]
Message received from the process: 2, client clock received : [1,1,0], Before Sending Message: : [1,0,0], After Sending Message: [1,1,0]
Broadcasting messsages to every processes simultaneously -  : [1,1,1]
Message received from the process: 1, client clock received : [2,1,0], Before Sending Message: : [1,1,1], After Sending Message: [2,1,1]
Message received from the process: 2, client clock received : [1,2,0], Before Sending Message: : [2,1,1], After Sending Message: [2,2,1]
Broadcasting messsages to every processes simultaneously -  : [2,2,2]
Message received from the process: 1, client clock received : [3,1,1], Before Sending Message: : [2,2,2], After Sending Message: [3,2,2]
Broadcasting messsages to every processes simultaneously -  : [3,2,3]
Message received from the process: 2, client clock received : [2,3,1], Before Sending Message: : [3,2,3], After Sending Message: [3,3,3]
Broadcasting messsages to every processes simultaneously -  : [3,3,4]
Message received from the process: 1, client clock received : [4,2,2], Before Sending Message: : [3,3,4], After Sending Message: [4,3,4]
Message received from the process: 2, client clock received : [3,4,2], Before Sending Message: : [4,3,4], After Sending Message: [4,4,4]
```

ISSUES WE ENCOUNTERED:

While executing we faced issues regarding ports as well as some system errors.

WORKING AS A TEAM:

We worked as a group where we isolated the errands between us and worked together on required functionalities.

PROJECT OUTCOMES:

We developed a distributed system with n nodes that uses a vector clock.
We learned about a distributed system that timestamps messages sent and received across nodes using a logical clock, and we also find out that a distributed system would simulate employing several processes on a single machine.

REFERENCES:

https://www.geeksforgeeks.org/vector-clocks-in-distributed-systems/
https://www.geeksforgeeks.org/berkeleys-algorithm/