Project Report: 02

Distributed System
CSE 5306; Section: 004
Date: 04/09/2022

We have neither given nor received unauthorized assistance on this work. However, some of the codes were adopted and modified to generate ideas of how to work based on the provided instructions (proper citations were included). Furthermore, the 1st and the 2nd part of the project were done by Nasir Rakib and Venkatesh Vinnakota respectively. However, both of them are available to answer any questions that arise in the future.

Mohammad Abu Nasir Rakib
ID: 1001870163
Venkatesh Vinnakota
ID: 1001876739

Part 1:

Introduction:

According to the given instructions, this project's logical clock in each machine will represent the number of messages sent and received by the machine also known as the counter. For clock synchronization, Berkeley's Algorithm is a popular technique used in distributed systems to achieve synchronization. Based on this algorithm, in the network, each of the machine nodes doesn't contain a UTC server or accurate time source.

Problem Encountered and Learning:

In the beginning, there were several challenges observed in the first part, including what functions were necessary, how to achieve synchronized time, and maintain correct latency in the different nodes. Different sources were helpful in learning how to implement and learn the above issues there several internet sources helped, which were cited accordingly. On top of that, the whole project provided an idea of adopting Berkeley algorithm and how to improve Christine's algorithm.

Implementation:

In order to implement, it has two parts i.e., the server.py and client.py which will need to run one after another respectively. The master clock and client clock will be running on the same machine. The introduced time delay will be very minimal to a nanosecond. An individual node has been chosen as a master node. Due to the use of "dateutil", it offers a generic date/time string parser that will generate common formats to represent a time and time. Running multiple clientclock will be connected and a common time/date will be associated with the client from the server.

References:

1. https://dateutil.readthedocs.io/en/stable/parser.html
2. Berkeley's Algorithm
3. Berkeley's Algorithm concept

Part 2:

Introduction:

The system consists of three nodes(i.e 3 servers) that communicate with each other by passing messages. Each server consists of a Counter and a 3d vector which represents the logical clock of that particular server. Each coordinate of the 3d vector represents the position for each of the server(node) in the system. Whenever an event(message passing from one server to another) takes place, the counter values and the 3d vector(logical clock) of both the servers get updated and the timestamps will be printed. Initially, the client communicates with the Servers and synchronizes the counter values.

Implementation:

- Consists of four parts, client, server1, Server2 and Server3.
- Each server contains a counter(logical clock) that will be initialized to a random integer value when the servers are started.
- The Client communicates with the servers and synchronizes the counters of the servers as per berkeley's algorithm.
- The Servers communicate with each other by passing messages within themselves.
- Each server consists of two threads, the main thread which receives the incoming messages from other servers and prints the received message and another thread which asks for the message and the server number of the server the message has to be sent.
- Whenever a server sends a message to another server, the timestamps before and after sending the message are printed as per the logical clock
- Similarly, the server that receives the message prints the timestamps before and after receiving the message.
- This part is implemented using java. The servers(nodes) communicate with each other by RPC(java RMI).

Things learned:

- Got a clear picture of how logical clocks work, their necessity and applications in real world.
- Learned about the advantages of vector clocks over lamport clocks.

References:

1. https://youtu.be/x-D8iFU1d-o