TRIBHUVAN UNIVERSITY
INSTITUTE OF ENGINEERING
PURWANCHAL CAMPUS

A
LAB REPORT
ON

"Simulation for Clock Synchronization in Distributed System using Vector Clock"

Submitted By:

SOHEL AKHTAR (075 BCT 081)

Submitted To:

PUKAR KARKI

DEPARTMENT OF
ELECTRONICS AND COMPUTER ENGINEERING
DHARAN, NEPAL

Lab 2: Simulation for Clock Synchronization in Distributed System using Vector Clock.

Vector clocks: Mattern and Fidge developed vector clocks to overcome the shortcoming of Lamport's clocks: the fact that from L(e) < L(e') we cannot conclude that e → e'. A vector clock for a system of N processes is an array of N integers. Each process keeps its own vector clock Vi, which it uses to timestamp local events. Like Lamport timestamps, processes piggyback vector timestamps on the messages they send to one another, and there are simple rules for updating the clocks:

VC1: Initially, $\mathbf{V_i[j]}$ = 0, for i, j = 1,2… N.

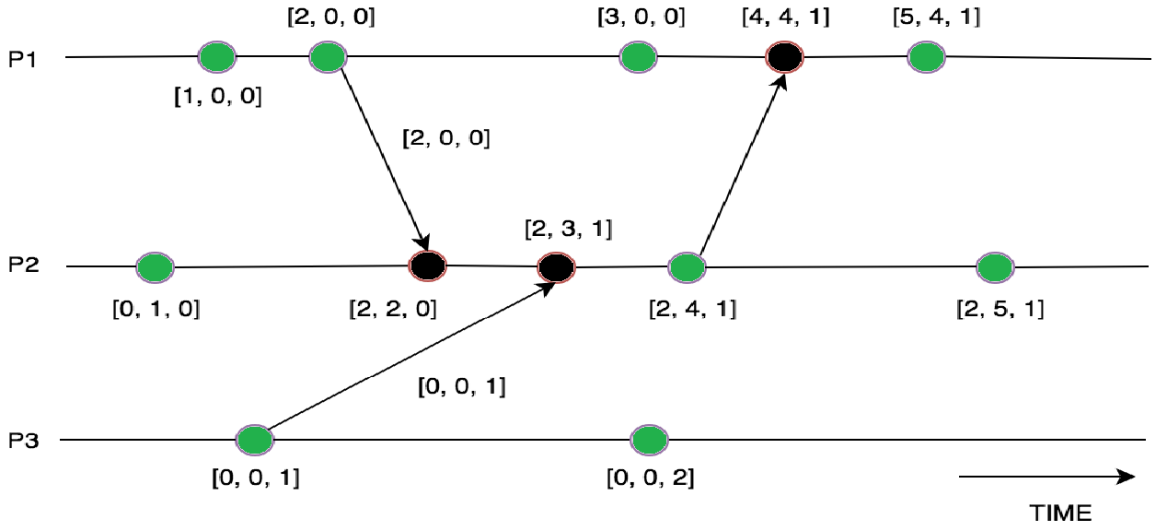
VC2: Just before $\mathbf{p_i}$ timestamps an event, it sets $\mathbf{V_i[i] := V_i[i] + 1.}$

VC3: $\mathbf{p_i}$ includes the value t = Vi in every message it sends.

VC4: When $\mathbf{p_i}$ receives a timestamp t in a message, it sets $\mathbf{V_i[j] := max(V_i[j], t[j])}$, for j = 1,2...N.

Taking the component-wise maximum of two vector timestamps in this way is known as a merge operation.

Vector Timestamps for the events



We may compare vector timestamps as follows:

V = V' iff V[j] = V'[j] for j = 1, 2… ,N

V ≤ V' iff V[j] ≤ V'[j] for j = 1, 2… ,N

V < V' iff V ≤ V' ∧ V ≠ V'

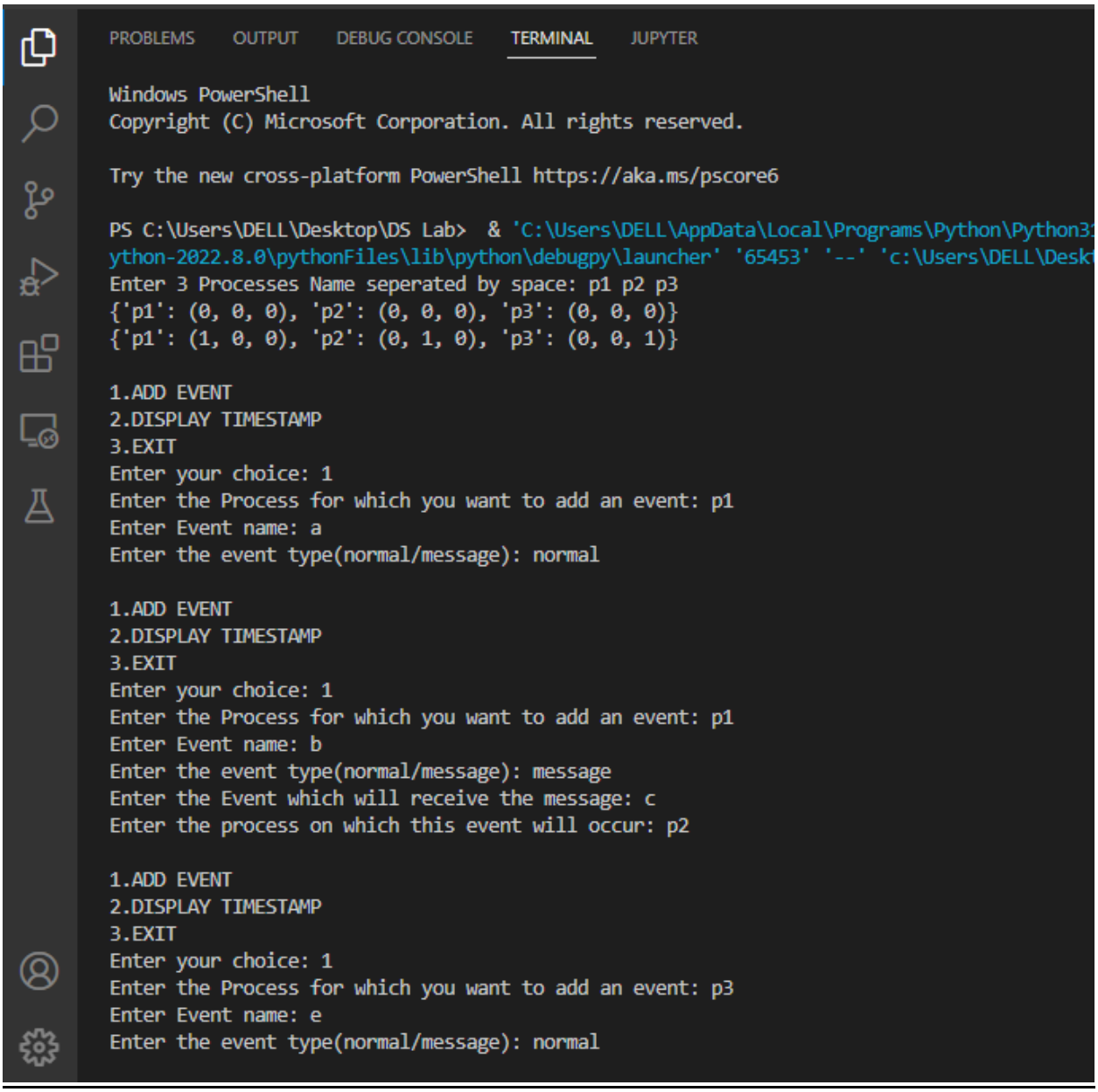
Vector timestamps have the disadvantage, compared with Lamport timestamps, of taking up an amount of storage and message payload that is proportional to N, the number of processes.

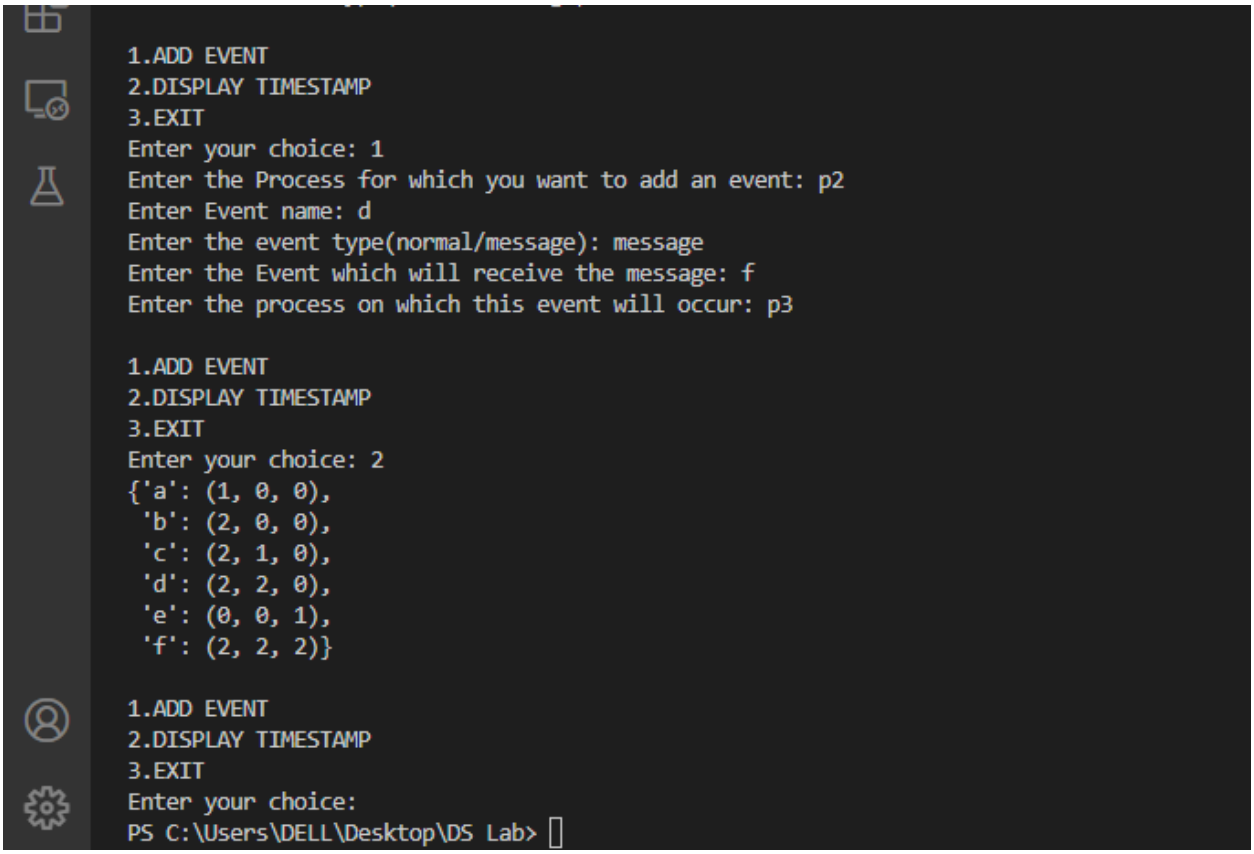
Source Code

```
import pprint

processList = []
logicalClock = {}
timeStamp = {}
processIndexAdd = {}


def addProcess():
    pName = input("Enter 3 Processes Name seperated by space: ")
    processList = pName.split()
    for process in processList:
        logicalClock[process] = (0, 0, 0)

    for i in range(0, len(processList)):
        dummy_list = [0, 0, 0]
        dummy_list[i] = 1
        processIndexAdd[processList[i]] = tuple(dummy_list)

    print(logicalClock, processIndexAdd, sep="\n")

# tuple(map(sum, zip(a, b)))


def addEvent():
    pName = input("Enter the Process for which you want to add an event: ")
    eName = input('Enter Event name: ')
    etype = input("Enter the event type(normal/message): ")
    if etype == "normal":
        logicalClock[pName] = tuple(
            map(sum, zip(logicalClock[pName], processIndexAdd[pName])))
        timeStamp[eName] = logicalClock[pName]
    if etype == "message":
        logicalClock[pName] = tuple(
            map(sum, zip(logicalClock[pName], processIndexAdd[pName])))
        timeStamp[eName] = logicalClock[pName]
        sendmessage(timeStamp[eName])


def sendmessage(t):
    eName = input("Enter the Event which will receive the message: ")
    pName = input("Enter the process on which this event will occur: ")
```

```
    dummyList = list(logicalClock[pName])
    for i in range(3):
        if t[i] > logicalClock[pName][i]:
            dummyList[i] = t[i]
    logicalClock[pName] = tuple(
        map(sum, zip(dummyList, processIndexAdd[pName])))
    timeStamp[eName] = logicalClock[pName]


def display():
    pprint.pprint(timeStamp)


if __name__ == "__main__":
    addProcess()
    while(1):
        print("\n1.ADD EVENT\n2.DISPLAY TIMESTAMP\n3.EXIT")
        n = int(input("Enter your choice: "))
        if n == 1:
            addEvent()
        elif n == 2:
            display()
        else:
            break
```

Output


PROBLEMS
OUTPUT
DEBUG CONSOLE
TERMINAL
JUPYTER
Windows PowerShell
Copyright (C) Microsoft Corporation. All rights reserved.
Try the new cross-platform PowerShell https://aka.ms/pscore6
PS C:\Users\DELL\Desktop\DS Lab> & 'C:\Users\DELL\AppData\Local\Programs\Python\Python3
ython-2022.8.0\pythonFiles\lib\python\debugpy\launcher' '65453' '--' 'c:\Users\DELL\Deskt
Enter 3 Processes Name seperated by space: p1 p2 p3
{'p1': (0, 0, 0), 'p2': (0, 0, 0), 'p3': (0, 0, 0)}
{'p1': (1, 0, 0), 'p2': (0, 1, 0), 'p3': (0, 0, 1)}
1.ADD EVENT
2.DISPLAY TIMESTAMP
3.EXIT
Enter your choice: 1
Enter the Process for which you want to add an event: p1
Enter Event name: a
Enter the event type(normal/message): normal
1.ADD EVENT
2.DISPLAY TIMESTAMP
3.EXIT
Enter your choice: 1
Enter the Process for which you want to add an event: p1
Enter Event name: b
Enter the event type(normal/message): message
Enter the Event which will receive the message: c
Enter the process on which this event will occur: p2
1.ADD EVENT
2.DISPLAY TIMESTAMP
3.EXIT
Enter your choice: 1
Enter the Process for which you want to add an event: p3
Enter Event name: e
Enter the event type(normal/message): normal


1.ADD EVENT
2.DISPLAY TIMESTAMP
3.EXIT
Enter your choice: 1
Enter the Process for which you want to add an event: p2
Enter Event name: d
Enter the event type(normal/message): message
Enter the Event which will receive the message: f
Enter the process on which this event will occur: p3
1.ADD EVENT
2.DISPLAY TIMESTAMP
3.EXIT
Enter your choice: 2
{'a': (1, 0, 0),
'b': (2, 0, 0),
'c': (2, 1, 0),
'd': (2, 2, 0),
'e': (0, 0, 1),
'f': (2, 2, 2)}
1.ADD EVENT
2.DISPLAY TIMESTAMP
3.EXIT
Enter your choice:
PS C:\Users\DELL\Desktop\DS Lab>