DistributedSystemsPSets50.041

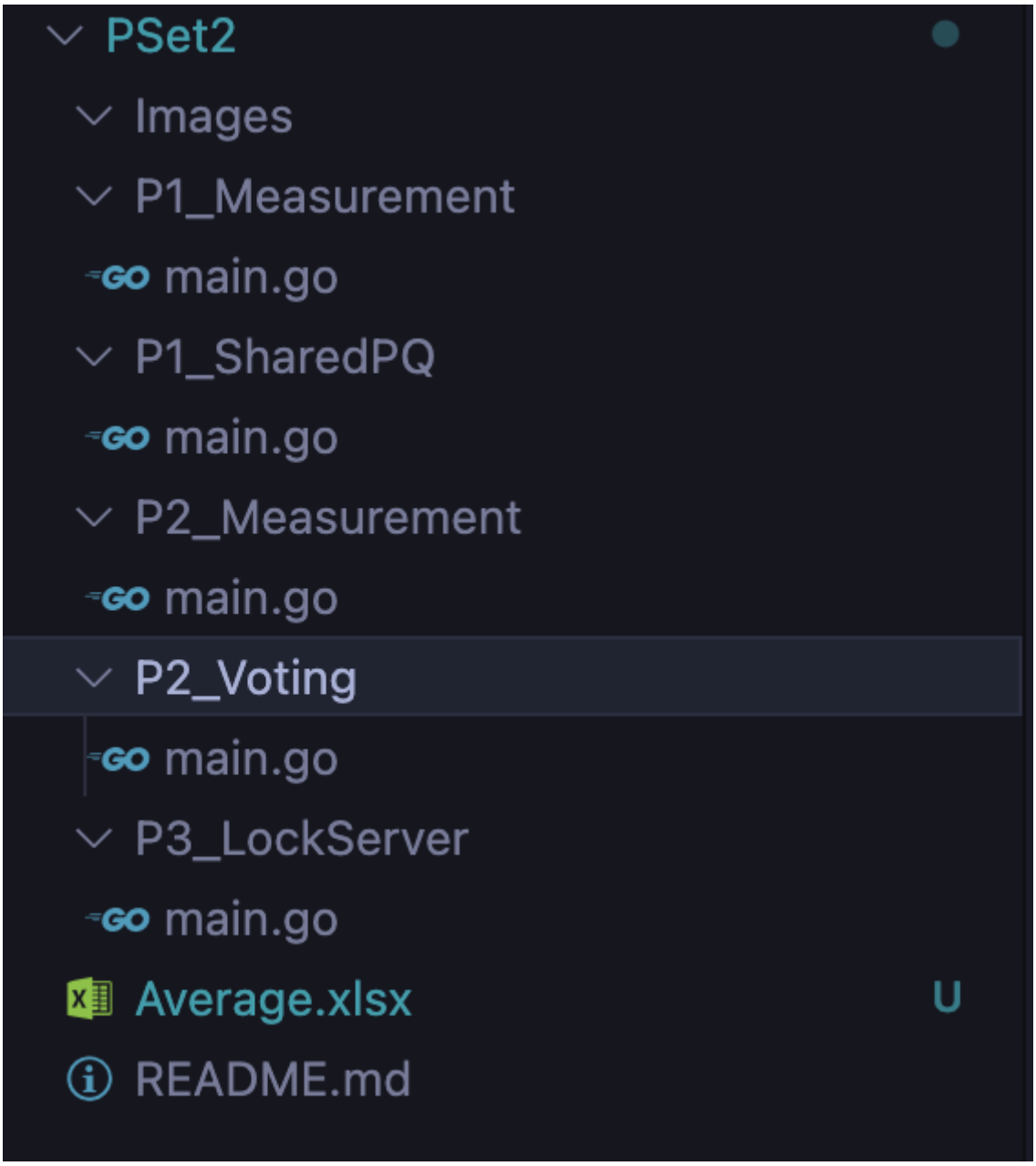
Assignment 2

Toh Kai Feng

1004581

Introduction

This is the following file structure used:



1. To run questions P1 to P3, replace "--Question--" with either "P1_SharedPQ","P2_Voting","P3_LockServer" in the following command. (Note: p1_Measurement and p2_Measurement are additional scripts to measure the time of protocols):

```
go run -race --Question--/main.go
```

Part 1

This would be the output when part 1 is ran:

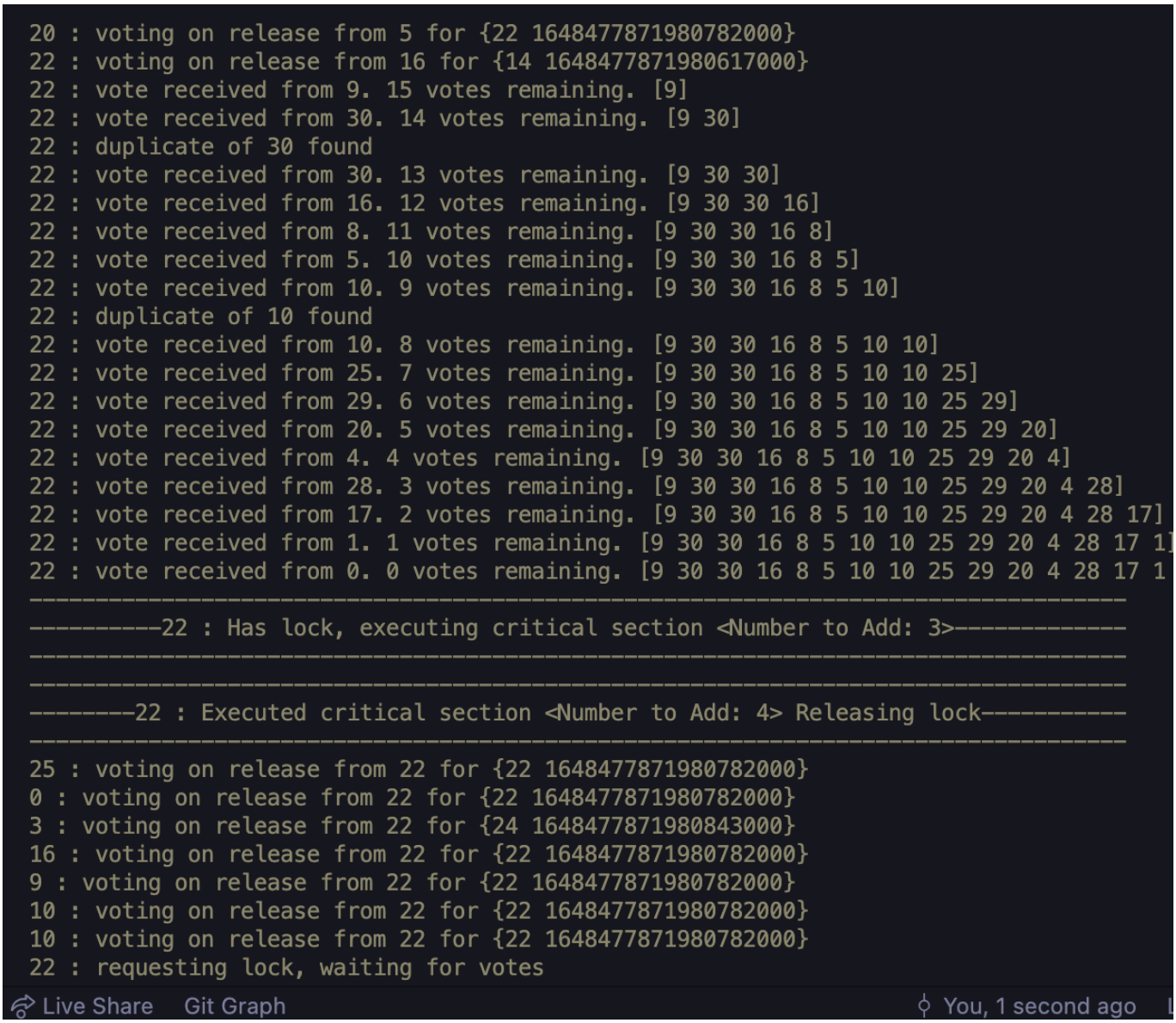
```
0 : Reply received from 9. 1 replies remaining. [0 4 7 5 3 10 6 8 1 9]
------------------------------------------------------------------------------------------
--------2 : Executed critical section <Number to Add: 208> Releasing lock----------
------------------------------------------------------------------------------------------
2 : requesting lock, waiting for replies
3 : Reply received from 2. 4 replies remaining. [3 10 6 8 1 9 2]
7 : Reply received from 2. 2 replies remaining. [7 5 3 10 6 8 1 9 2]
Here, false
4 : Reply received from 9. 2 replies remaining. [7 5 3 4 10 6 8 1 9]
2 : Reply received from 2. 10 replies remaining. [2]
4 : Reply received from 2. 1 replies remaining. [7 5 3 4 10 6 8 1 9 2]
9 : Reply received from 2. 9 replies remaining. [9 2]
8 : Reply received from 2. 6 replies remaining. [8 1 6 9 2]
1 : Reply received from 2. 7 replies remaining. [6 1 9 2]

10 : Reply received from 2. 5 replies remaining. [10 6 8 1 9 2]
0 : Reply received from 2. 0 replies remaining. [0 4 7 5 3 10 6 8 1 9 2]
------------------------------------------------------------------------------------------
----------0 : Has lock, executing critical section <Number to Add: 208>------------
------------------------------------------------------------------------------------------
------------------------------------------------------------------------------------------
--------0 : Executed critical section <Number to Add: 209> Releasing lock----------
------------------------------------------------------------------------------------------
0 : requesting lock, waiting for replies
2 : Reply received from 0. 9 replies remaining. [2 0]
8 : Reply received from 0. 5 replies remaining. [8 1 6 9 2 0]
4 : Reply received from 0. 0 replies remaining. [7 5 3 4 10 6 8 1 9 2 0]
------------------------------------------------------------------------------------------
----------4 : Has lock, executing critical section <Number to Add: 209>------------
------------------------------------------------------------------------------------------
------------------------------------------------------------------------------------------
--------4 : Executed critical section <Number to Add: 210> Releasing lock----------
------------------------------------------------------------------------------------------
4 : requesting lock, waiting for replies
```

what we are looking for is that the programme does not enter into the race condition. This can be seen with the print statements where each node is trying to add to a number with a single address. each node is trying to increment the value of the same number but must first acquire the lock with permission from all other nodes.

Part 2

This would be the output when part 2 is ran:



Similarly, what we are looking for is that the programme does not enter into the race condition. This can be seen with the print statements where each node is trying to add to a number with a single address. each node is trying to increment the value of the same number but must first acquire the lock with permission from all other nodes. It is also interesting to note that voting protocol is much slower than lamport shared priority queue, presumably because more messages has to be exchanged before each node gets to enter the critical section. (second node has wait for first node to to release all votees and allow other nodes to vote before second node can enter critical section)

Part 3

This would be the output when part 3 is ran:

```
❯ go run -race PSet2/P3_LockServer/main.go
1 out of 11 concurrent requests done
Number of Nodes: 1,    Time taken: 80426
2 out of 11 concurrent requests done
Number of Nodes: 2,    Time taken: 68844
3 out of 11 concurrent requests done
Number of Nodes: 3,    Time taken: 73085
4 out of 11 concurrent requests done
Number of Nodes: 4,    Time taken: 72104
5 out of 11 concurrent requests done
Number of Nodes: 5,    Time taken: 74559
6 out of 11 concurrent requests done
Number of Nodes: 6,    Time taken: 94310
7 out of 11 concurrent requests done
Number of Nodes: 7,    Time taken: 98041
8 out of 11 concurrent requests done
Number of Nodes: 8,    Time taken: 99520
9 out of 11 concurrent requests done
Number of Nodes: 9,    Time taken: 47065
10 out of 11 concurrent requests done
Number of Nodes: 10,    Time taken: 123765
11 out of 11 concurrent requests done
Number of Nodes: 11,    Time taken: 95430
```

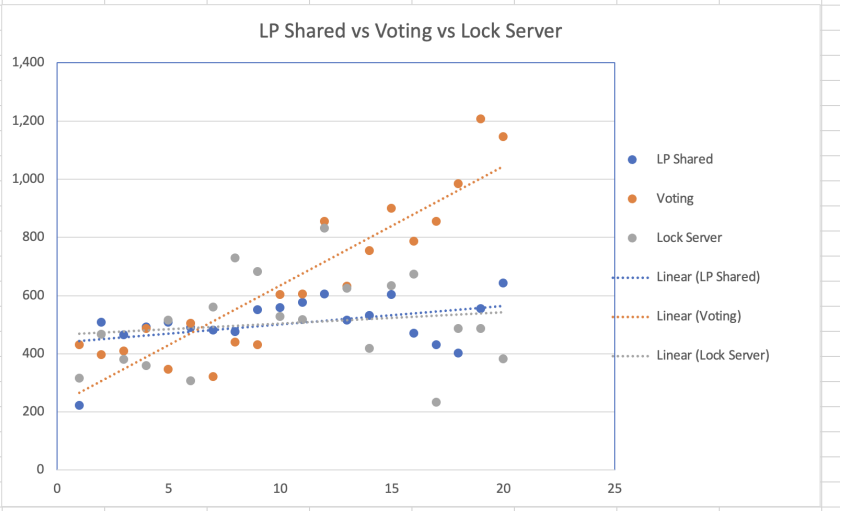
What is different from the output for lockserver is that I am not printing any text about critical section because the lock server protocol is quite straight forward in its safety. (the lock server would queue all requests and allow the next node to enter only when earlier reqeusting nodes have completed)

Performance Comparison

This is a graph plot comparing:

1. Lamport Shared Priority Queue (LP Shared)
2. Voting
3. Central Lock Server

num concurrent	lp shared (ms)	voting (ms)	LockServer (ms)
1	221	430	316
2	507	396	466
3	465	409	379
4	491	485	359
5	508	346	515
6	489	503	307
7	480	320	560
8	475	440	728
9	550	431	681
10	557	602	528
11	576	604	517
12	604	855	830
13	514	631	624
14	530	754	417
15	603	899	633
16	469	786	672
17	431	854	232
18	401	984	485
19	555	1,207	485
20	642	1,145	381



A best fit line is plotted to measure the rate of growth in time taken when the number of concurrent requests are made.

As mentioned, I believe voting protocol is much slower than lamport shared priority queue, presumably because more messages has to be exchanged before each node gets to enter the critical section. (second node has wait for first node to to release all votees and allow other nodes to vote before second node can enter critical section)