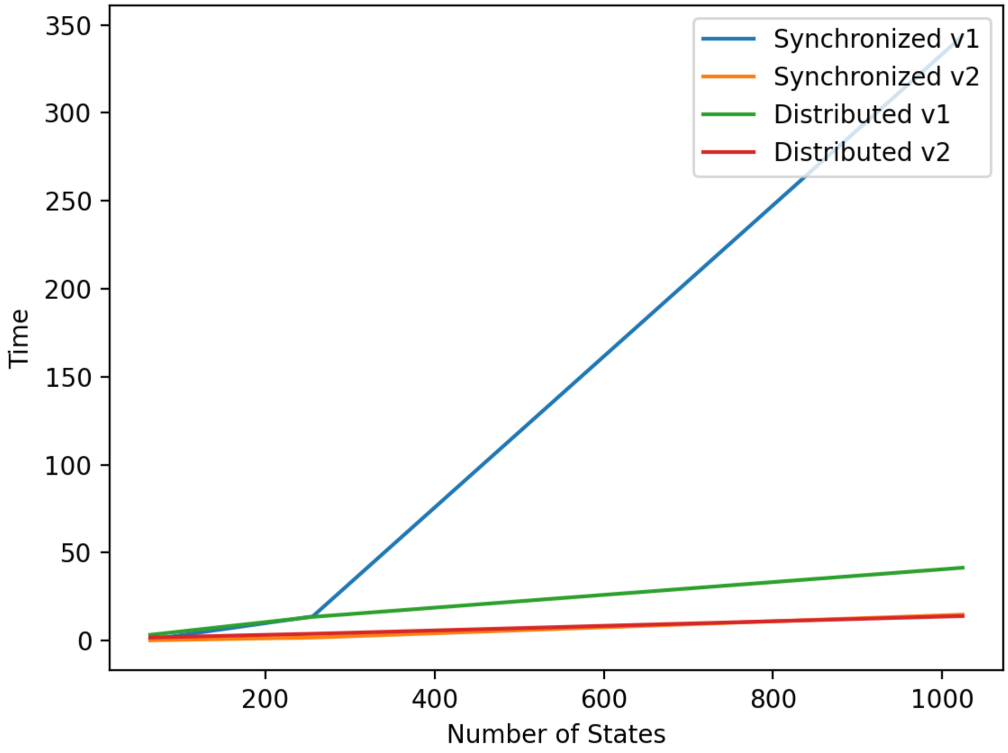
Assignment 2

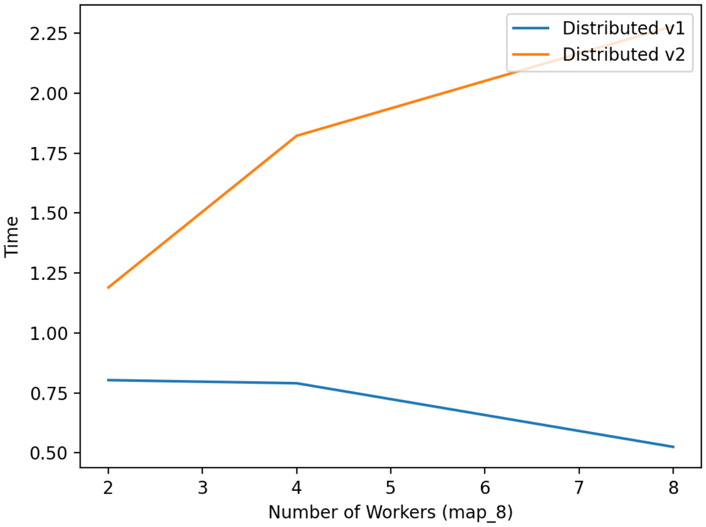
The code to produce these graphs are available on hw2.ipynb (it contains code for all the graphs at the end), graph_timevsmaps.py (code for the performance of dist v1 and v2) and graph_timevsworkers.py (code for the performance with different number of workers.

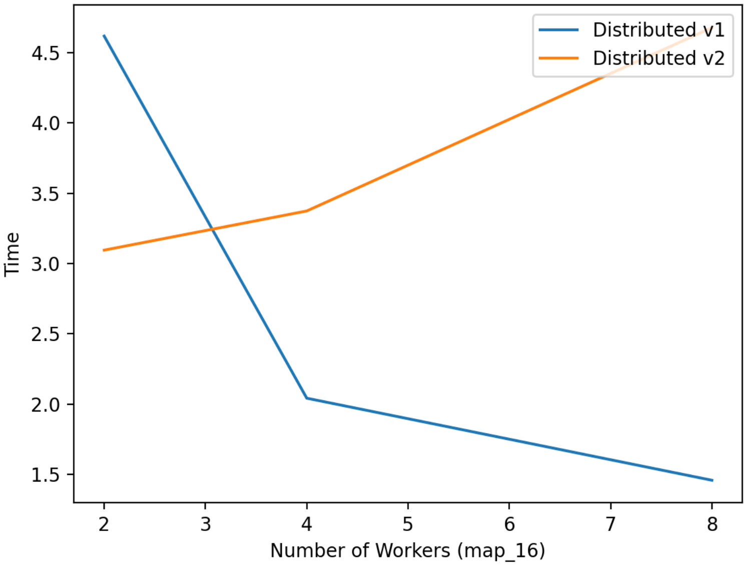
1)

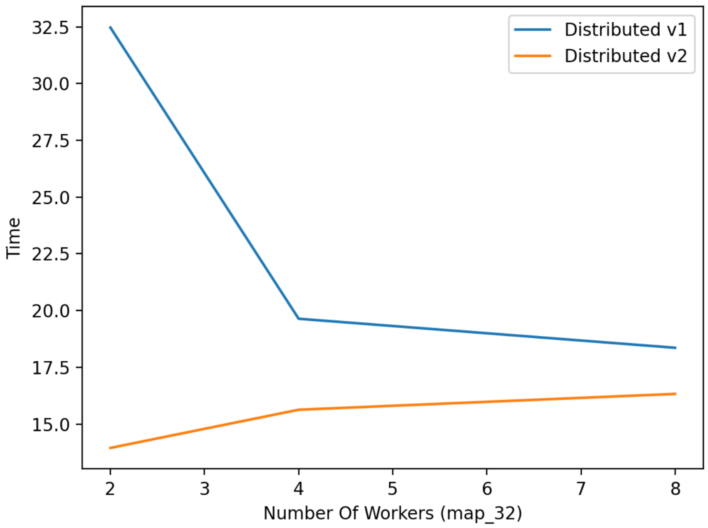


(The orange line is present but is not visible because it is being overlapped by the red line)

2)


Distributed v1
Distributed v2
Time
2.25
2.00
1.75
1.50
1.25
1.00
0.75
0.50
2
3
4
5
6
7
8
Number of Workers (map_8)


Distributed v1
Distributed v2
Time
4.5
4.0
3.5
3.0
2.5
2.0
1.5
2
3
4
5
6
7
8
Number of Workers (map_16)



3)

Briefly explain why the second distributed method is faster than the first one?

- In the first, each state must be loaded into a worker after one state is finished processing. In the second, it is loaded in batches and thus, each worker has more states to process before a new batch is passed in. Maybe the second is faster because there is less data being transferred from the value server to each worker.

4)

Compare the best distributed method with the best non-distributed approach. Which one is better? Briefly explain why.

- The best non-distributed is the synchronized v2 and the best distributed is the distributed v2. Distributed v2 is significantly faster the synchronized v2 for all map sizes. This is because of the parallel processing that is taking place when using ray for distributed v2. Batches of states are being concurrently calculated in distributed v2 whereas in synchronized v2, each state is iteratively (one of the other) calculated. However, it is important to note that distributed v2 performance degrades as the number of workers increase. From the first graph on this report, we can also see that synchronized v1 time to process 32x32 maps significantly **increases** compared to the smaller maps whereas distributed v2 performance gradually **improves** in terms of time taken.