Distributed Systems Assignment 4

a1781040 - Haorui xu

a1751699 - Zhao Ming Soh

a1753701 - Jin Hong Xiong

Introduction

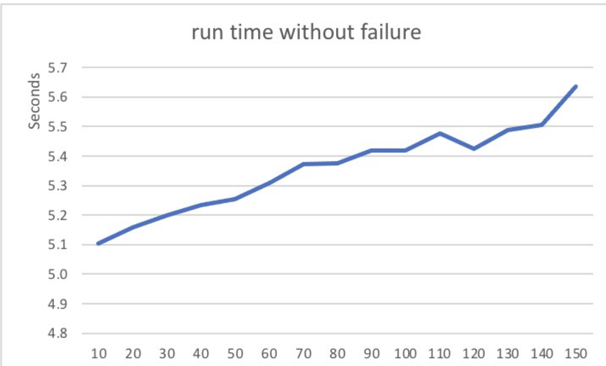
Paxos algorithm is one of the most commonly used consensus algorithms in distributed systems. Its purpose is to ensure that multiple nodes or servers that communicate in an asynchronous distributed network can achieve a consistent state, by that we meant the data that lives in all the nodes can converge to the same value at any point in time. Paxos algorithm is also fault-tolerant by nature as it only ever needs a majority of the nodes or servers to agree on a value that is proposed by any one of the nodes or servers. The implementation that we chose to use is that of assignment 3 which is a basic Paxos Algorithm that deals with an arbitrary number of members where the first 3 is always the proposer. One of the key implementations that may affect our Paxos runtime would be the resending of new prepare messages to all members excluding itself with a higher proposal ID. This is triggered if any one of the proposers fails to receive a majority of promise messages in the promise phase (Phase 1b) or accepted messages in the accepted phase (Phase 2b). This behaviour will most definitely create overhead in the number of messages communicated between proposers and acceptors, thus increasing the overall time it takes for the algorithm to reach a consensus.

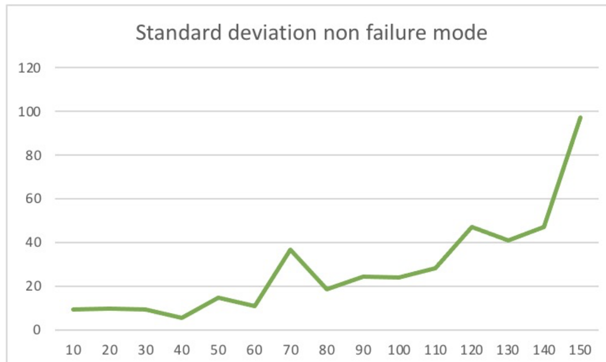
Experimental Setup

2 variables were used in our experiment where one is the number of members(nodes) and the other one is the time it takes for the Paxos Algorithm to reach consensus. The collection of the relevant data for each of the metrics were done by running our Paxos Algorithm 15 times, starting with 10 members and in each run, the members were incremented by 10 until 150 members, at the same time, the run time of the Paxos Algorithm was recorded as well. This is done for both the no failure mode and the failure mode of the Paxos Algorithm. The failure mode is defined by the disconnection of proposers 2 and 3 after having sent an accept message in Phase 1a. Whereas, the non-failure mode is defined by not having any disconnections. All of the data collected were imported into Microsoft Excel, as such 4 different graphs were rendered. 2 of the graphs illustrate the relationship between the run time of Paxos and the number of

members, one for the failure mode and the other one for the non-failure mode. Another 2 illustrate the relationship between the standard deviation of the run times of Paxos with the different number of members, one for the failure mode and the other one for the non-failure mode.

No Failure Mode





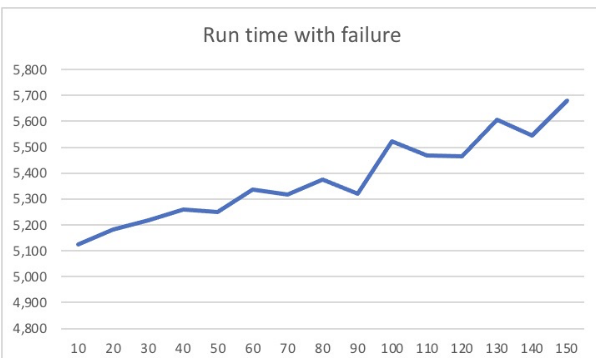
- In the Paxos algorithm, as the number of members increases, the Paxos system needs to send more prepare and propose messages and process more responses received from members. This would increase the run time of the system because it would take more time for sending and processing more messages. And in our assumption, the relationship between the run time of the Paxos system and the number of members would be positive correlations.

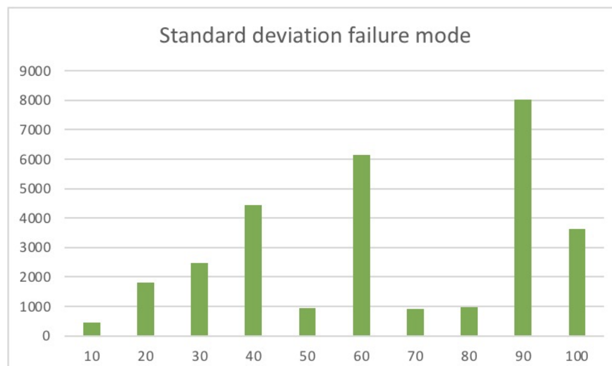
The graph above indicates the relationship between the runtime of the Paxos system and the number of members. From the graph, we could see that when the value of members increases and runtime would also increase. Therefore these two variables are positive correlations. Also, the graph is almost a straight line and there are errors in data in practice experiments. So we could find the relationship between the run time of Paxos system and the number of members that are linearly proportional. Since the relationship is linear like y=kx+b.K is constant, therefore the performance of the Paxos system would not change as members as the number of members increasing or decreasing

For this experiment, we could see that the relationship between the run time and number of members is linearly proportional, since in the Paxos algorithm the more members, the more messages the system needs to send and receive which increase the run time. And the performance of the Paxos system would not change when the number of members changed. SInce from the graph, we would see that the time for the Paxos system to send prepare and propose messages and process responses received for a single member is constant. So the performance of the system would not be affected by the number of members.

With Failure Mode

In our failure mode experiment, we tested the performance of the Paxos system for when proposers 2 and 3 would lose their connection after having sent a PROPOSE message in Phase 1a. For example, if proposer 2 managed to get its PROPOSE message accepted by the majority of the members excluding itself, meaning all the members have accepted the value proposed by proposer 2, because proposer 2 is no longer alive, other proposer with bigger proposal ID, in our case, proposer 1 will piggyback the previously accepted value from proposer 2 to reach the final consensus phase. The same goes for proposer 3.





From the graph, the run time of the Paxos system is also positively correlated to the number of members. The runtime of the system does not have huge differences from the model without failure and the overall trend of change is very similar to the model without failure. This is due to the same reason we discussed in the no failure model. However, in the failure model graph, the line indicates the relationship between run time and the number of members has more ups and downs. This shows the performance of the Paxos system under the failure model is not stable. Since the Paxos system needs to handle the error that the proposer loses connection, and the proposal with a lower ID would loop to retry with a higher ID when time out to make acceptor to acceptor promise. This would make the runtime of Paxos system increase, and affect the performance of the system which causes the graph has more ups and downs.

Conclusion

In these two experiments, we tend to investigate the relationship between the performance of the system and the number of members in the system. We analyze the performance of the Paxos System by measuring the run time of the system. In the first experiment with no failure mode, we found out that the relationship between the run time and number of members is linearly proportional, since in the Paxos algorithm the more members, the more messages the system needs to send and receive which increase the run time.

Under the failure mode, the run time of the Paxos system is also positively correlated to the number of members. But performance is not stable, because in the Paxos system when the proposer loses connection and can not go to the next phase, the proposal with a lower ID would loop to retry with a higher ID when time is out to make acceptor to acceptor promise to go to phase 2. This would make the runtime of the Paxos system increase, and affect the performance of the system which causes the graph to have more ups and downs.