Report 5: Chordy

Chen, Zidi

October 8, 2020

1 Introduction

In this assignment, a simple distributed hash table which is based on Chord scheme is implemented. The important steps are to construct a growing ring, a local storage and to be able to handle the new nodes and died nodes.

2 Main problems and solutions

There are slightly difference between the assignment and the chord scheme. In this section, I tried to clarify some of the differences.

For key module, the between/3 function allows Xkey equal to Ykey. I think it does not matter here. This might because we are not using Hash functions but a random number generator here to generate keys. In this way, there might be the same keys, which we actually do not want to have. But since this possibility is not high, I think that it does not matter.

For the stabilizing time interval, if we stabilize more frequently, then the ring will be more precise. But the program cost more time on stabilizing. If we do not stabilize often, then the newly added node and dead node will not be handled in time.

For the notify function, we do not need to inform the new node about the predecessor. The new node knows who is the successor. The predecessor of the new node will be clear in the next time of stabilizing.

3 Evaluation

Here I described how I tested the node modules.

3.1 Ring

Here is the result of testing if the ring which contains three nodes is connected.

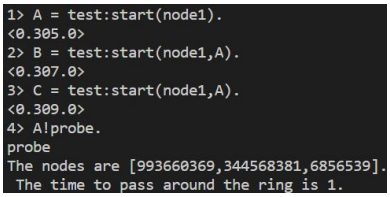

Figure 1: result of ring implementation

3.2 Storage

Here I tested the add and lookup function for the local storage. The key value pair works fine in the program.

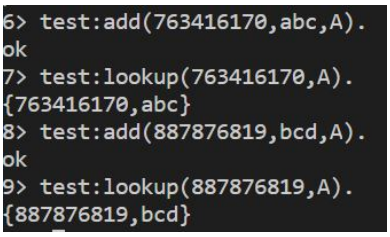

Figure 2: result of storage implementation

Now I test a more complicated situation. After node A, B, C, D generated a ring, I added values to a node. Then, I added a new node E to the ring. E can also find the exist value. (line 9 in the figure 3 is a typing error. I cut it from the figure.)

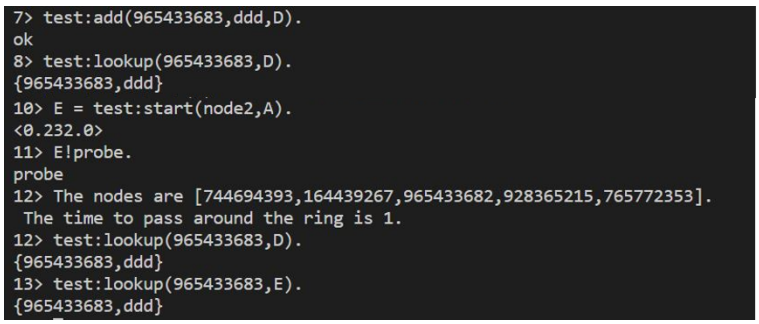

Figure 3: result of storage implementation

3.3 Handling failure

Here is the bonus task 1, to detect the failure nodes. In the test, I created a ring with three nodes. Then I stopped a node by hand. The ring still works with out this node. It is worth to mention that there is no "" in erlang. Remind myself that do not use "DOWN" anymore!!

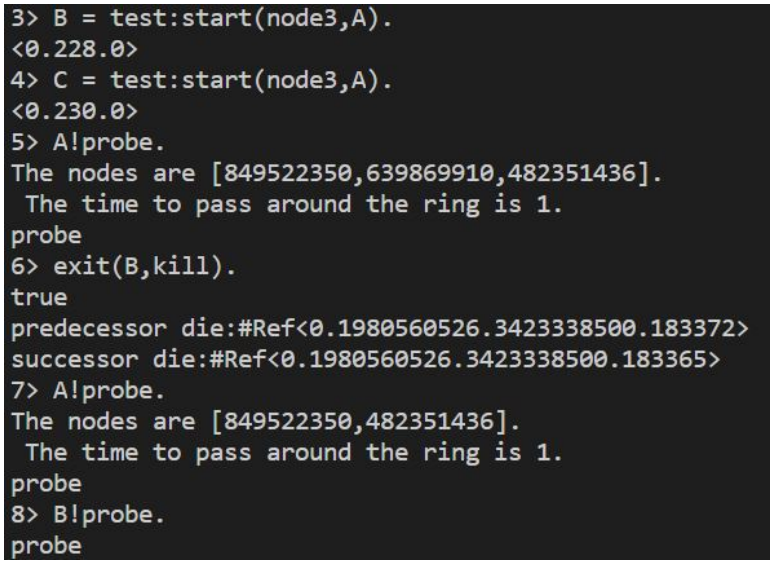

Figure 4: result of failure detection

4 Conclusion

In conclusion, this assignment is based on chord but much more simpler than chord. The core of the assignment is to dealing the new node and dead node. Each node is responsible for one part of the ring. It is difficult to understand at first, because there are a lot of messages between peers. If time allows, I hope that I can finish the replica part and go through the chord algorithm again.