Report 2: Routy - A Small Routing Network

Ziheng Zhang

September 21st, 2021

1 Introduction

In this program, I implemented a small routing network by using various functions such as Map, Dijkstra, Interfaces, Router and History.

The Map function is about finding nodes directly connected to a given node. The Dijkstra focus on how to compute a routing table and find the shortest way to distribute messages. The Interfaces relate to keeping track route of messages. The History involves functions to avoid cyclic paths, and finally, the Router refers to routing messages through a network.

2 Main problems and solutions

2.1 Dijkstra

Implementing Dijkstra Algorithm to calculate a routing table is the most difficult part, especially the `iterate` function. The `iterate` function should be returned when the following two conditions are met:

1. An empty list.
2. The first node in a sorted list has an `inf` distance.

Otherwise, we can iterate over the adjacency of the node and update the information of it.

3 Evaluation

I evaluated this small routing network by using flowing steps:

1. Create 3 nodes.
2. Add node's information.
3. Broadcast.
4. Update.
5. Send messages.

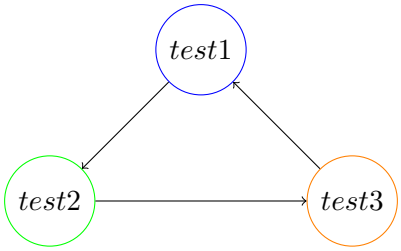


Figure 1: Structure of network

The distribution of nodes is shown in the figure 1.

Test results are shown in the Figure 2.

```
PS D:\OneDrive-Ziheng Zhang\OneDrive\KTH\ID2201 HT211 Distributed Systems, Basic Course\HomeWo
rk\Routy-a small  routing network> erl -name ziheng@zhang -setcookie routy -connect_all false
Eshell V12.0  (abort with ^G)
(ziheng@zhang)1> routy:start(r1, test1).
true
(ziheng@zhang)2> routy:start(r2, test2).
true
(ziheng@zhang)3> routy:start(r3, test3).
true
(ziheng@zhang)4> r1 ! {add, test2, {r2, 'ziheng@zhang'}}.
{add,test2,{r2,ziheng@zhang}}
(ziheng@zhang)5> r2 ! {add, test3, {r3, 'ziheng@zhang'}}.
{add,test3,{r3,ziheng@zhang}}
(ziheng@zhang)6> r3 ! {add, test1, {r1, 'ziheng@zhang'}}.
{add,test1,{r1,ziheng@zhang}}
(ziheng@zhang)7> r1 ! broadcast.
broadcast
(ziheng@zhang)8> r2 ! broadcast.
broadcast
(ziheng@zhang)9> r3 ! broadcast.
broadcast
(ziheng@zhang)10> r1 ! update.
update
(ziheng@zhang)11> r2 ! update.
update
(ziheng@zhang)12> r2 ! update.
update
(ziheng@zhang)13> r1 ! {send, test2, "Hi"}.
test1: routing message (Hi) from test1 to test2
test1: forward to test2
{send,test2,"Hi"}
test2: received message (Hi) from test1
(ziheng@zhang)14> r1 ! {send, test3, "Hello"}.
test1: routing message (Hello) from test1 to test3
test1: forward to test2
{send,test3,"Hello"}
test2: routing message (Hello) from test1 to test3
test2: forward to test3
(ziheng@zhang)15> test3: received message (Hello) from test1
(ziheng@zhang)15>
```

Figure 2: Test results

In the process of testing, I found that there are various small bugs in the document `routy.pdf`.

In function `router` in the file `routy.erl`, `routy` document said we should implement `{remove, Node}` like the following code:

```
{remove, Node} ->
    {ok, Ref} = intf:ref(Node, Intf),
    erlang:demonitor(Ref),
    Intf1 = intf:remove(Node, Intf),
    router(Name, N, Hist, Intf1, Table, Map);
```

However, if we try to remove a node which does not exist, error will occur. The test result is shown in the Figrue 3. And after this error, every operations will have no effect.

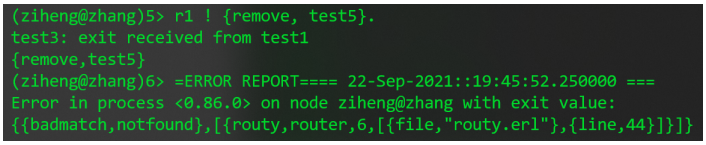


Figure 3: Error in the *router* function

Therefore, in order to fix this error, I implement `{remove, Node}` like the following code:

```
{remove, Node} ->
    case interfaces:ref(Node, Intf) of
        {ok, Ref} ->
            erlang:demonitor(Ref),
            Intf1 = interfaces:remove(Node, Intf),
            router(Name, N, Hist, Intf1, Table, Map);
        notfound ->
            io:format("Error! You are trying to
                       remove an unexisted node!")
    end,
    router(Name, N, Hist, Intf, Table, Map);
```

After handling errors in function `router`, user will receive a text message to remind that this operation is illegal, and user can continue to send messages from one node to others. Test result is shown in the Figure4.

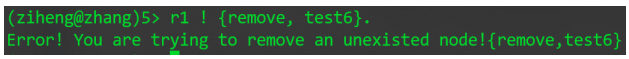
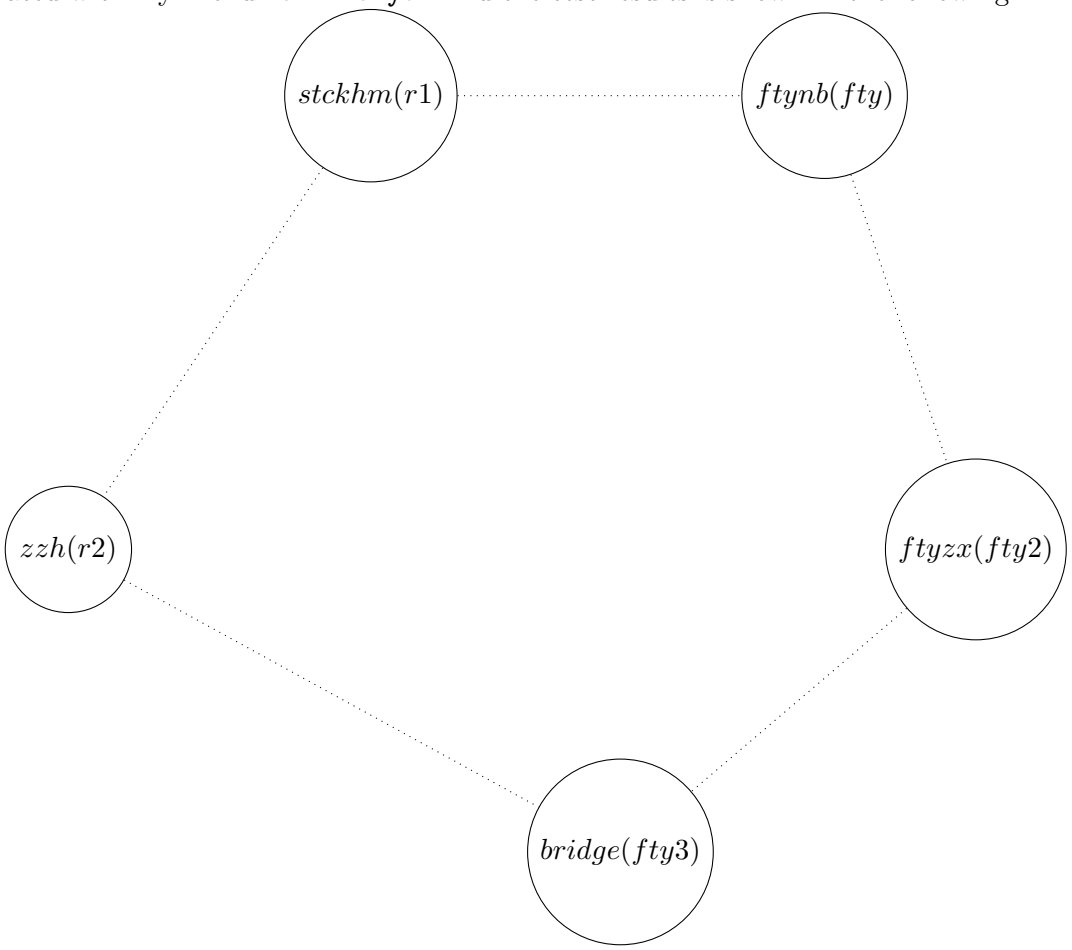


Figure 4: Result of handling errors

4 Bonus

I cooperated with my friend `Fan Tianyu`. And the test results is shown in the following



figures.

5 Conclusions

Through this project, I learned how to use and build a small routing network, and have a better understanding of how routing works. In the process of implementing functions, I also have a deeper knowledge of `Erlang` programming.