Report 2: Routy

Chen Sihan

September 18, 2020

1 Introduction

The goal of this project is to implement a link-state protocol by using different processes to simulate different routers over the network.

2 Main problems and solutions

Before starting the explaining, we should firstly know that in the project we are using different processes that represent different "cities" routers in "the world". That means each process (router) should have its routing table and map. When there are connection changes, the process must broadcast the message to the network and the other processes in the network should update continuously to receive the latest change.

2.1 Map

Firstly, the directional graph should be implemented. Here provides a data structure that can easily describe the graph, which is like [{beijing,[shanghai, stockholm]}, ...]. In this case, we have that beijing can forward information to other cities (beijing to shanghai and beijing to stockholm). However, shanghai and stockholm cannot forward information to beijing.

2.2 Dijkstra algorithm

This algorithm has a input of Map and a output of routing table. The routing table is like [{a, b}, {c, d}], e.g. if you want to send message to a, you should firstly send message to the gateway b. To implement the algorithm, we should firstly calculate the sorted list which has a hop count that may be compared to calculate the route with minimal distance. The hop count of an entry (with the structure Node, N, Gateway in the sorted list is set to be **inf** if the node is different from the gateway (One node only knows itself in initial time). Then an iteration function should be done continuously until any of three conditions:

1. no more entries (already end of list).
2. first has a hop count inf

Otherwise, for all the nodes directly linking to the Node, update their distance (Some of the inf will be updated to smaller number at this time). In the end the list is complete and all routing pair {Node, Gateway} of the minimal distance is stored in the routing table.

2.3 Routy

To manage the message transmission around the network, we need one process for each "city". We need a router recursively call it self to store and carry the information such as the city name, N, History (avoid cyclic), interface information, routing table and map. Also the behavior functions such as add, remove must be implemented in this file.

3 Bonus: the world

I cooperate with my partner to implement the task. I create three processes under my erlang shell named "sweden@192.168.3.12", when my partner create a erlang shell with name "china@192.168.3.2". The cities that we built are shown in the picture below:

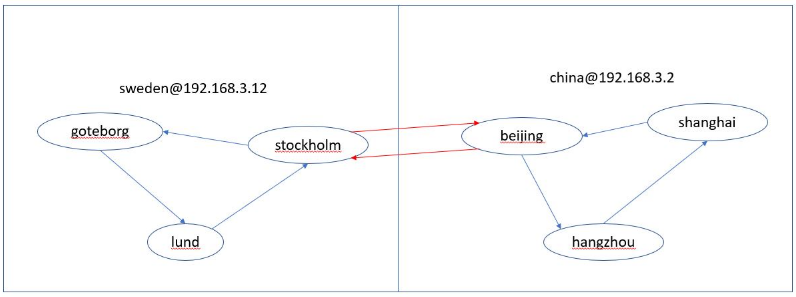

Figure 1: connect regional "routers" on different computers

The key idea is to establish the to-way connection between the two "regional network", which means we should send an {add, beijing, {r6, 'china@192.168.3.2'}}. from Stockholm to beijing to let the china network know about the sweden router information. On the other hand, we should send an {add, stockholm, {r1,'sweden@192.168.3.12'}}, from beijing to Stockholm to let the sweden knows about china router information. In that case, we can send messages between eaach two of the routers in the network.

4 Fault Handling

There are some errors that may happen when we do some unexpected operations. Here are some of the fault handling:

case 1: routers closed and restarted solution: when a router is closed and restarted, its routing table and interfaces and map information will be lost. Now, we should repeat the "add" operations for each of its outgoing links, and then broadcast and update all the routers.

case2: routers remove links solution: when a router remove links, there should be two possible errors. First is that before the router remove a link, the link should be searched in the interface lists to show whether the link exists or not. Without this step, the router may crash if the we remove an unexisted link. The second is that after a link is removed, the other routers are not aware of it. That may cause the problem that if the other routers want to send a message through the router that just remove its outgoing link, the router would crash. Therefore, after the link is removed, the broadcast and update should also be done to keep the map consistency.