Homework 2 Report: Routy

Justas Dautaras

September 17, 2019

1 Introduction

In the second Distributed Systems task a link-state routing protocol in Erlang is implemented. It is the most used routing protocol for Internet routers. Doing this task allows to understand the link-state routing protocol, how a consistent view is maintained and the problems related to network failures.

This task is divided into four parts:

· **Map** It is a directional map which shows nodes directly connected to a given node. It as a list of entries where each entry consist of a city with a list of directly connected cities.

· **Djikstra algorithm** This algorithm computes a routing table which represents a list with one entry per node where the entry describes which gateway, should be used to reach the node.

· **Interfaces** This entity is used to keep track of interconnected routers - add, remove, lookup, broadcast.

· **History** This entity is used to keep track if a link-state message is not in a loop. Message number is tracked and checked whether is is not in a loop.

· **The router** This entity implements all other parts to have a full router functionality.

After all of this is implemented, a router with full functionality is created. It can connect with other routers and send messages through the shortest known path.

2 Main problems and solutions

During this task some one of the problems (specifics) encountered include that when a node broadcasts it's map, all the nodes are updated, but itself.

Meaning that if it is a linearly connected routers structure, the first node, not having any nodes connected to it, cannot send messages, since it does not know the route.

```
{links, Node, R, Links} ->
      case hist:update(Node, R, Hist) of
        {new, Hist1} ->
          intf:broadcast({links, Node, R, Links}, Intf),
          Map1 = map:update(Node, Links, Map),
          router(Name, N, Hist1, Intf, Table, Map1);
        old ->
          router(Name, N, Hist, Intf, Table, Map)
      end;
```

It is this way, because a node only sees to forward connected nodes and someone has to tell it the path further down the route. Thus, in a way, the routers must have a cyclic structure and requires maps to be broadcasted after each new route, as well as updating other nodes.

3 Evaluation

The problem described above is just a characteristic of a lightweight link-state routing protocol and is a perfectly functional base. It has some requirments to be fully functional and some of it can be solved with additional coding, such as auto re-broadcasting within a certain time interval and same with updating. There may be many more solutions, but for testing purposes this works perfectly.

4 Conclusions

Solving this task has taught me to program on a more intermediate level of functional programming and helped to understand underlying network routing characteristics and the fundamentals of distributed systems. As well as the link-state routing protocol itself, how a consistent view is maintained and problems related to network failures.

5 Testing

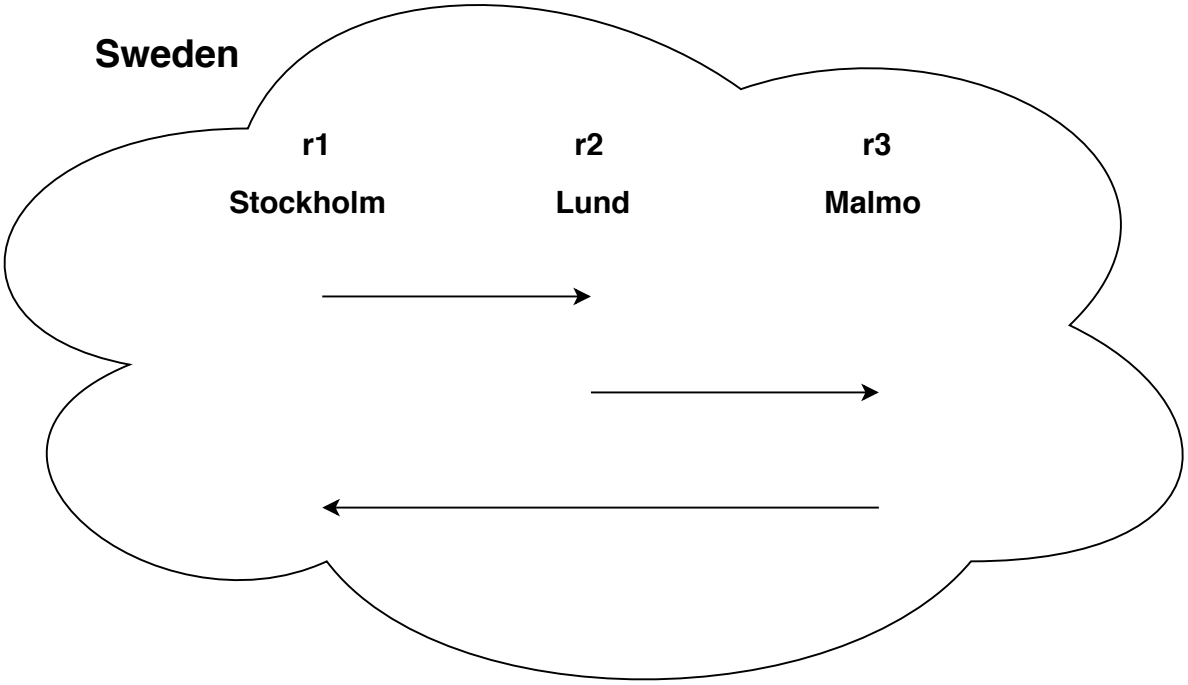

Figure 1: Test case 1

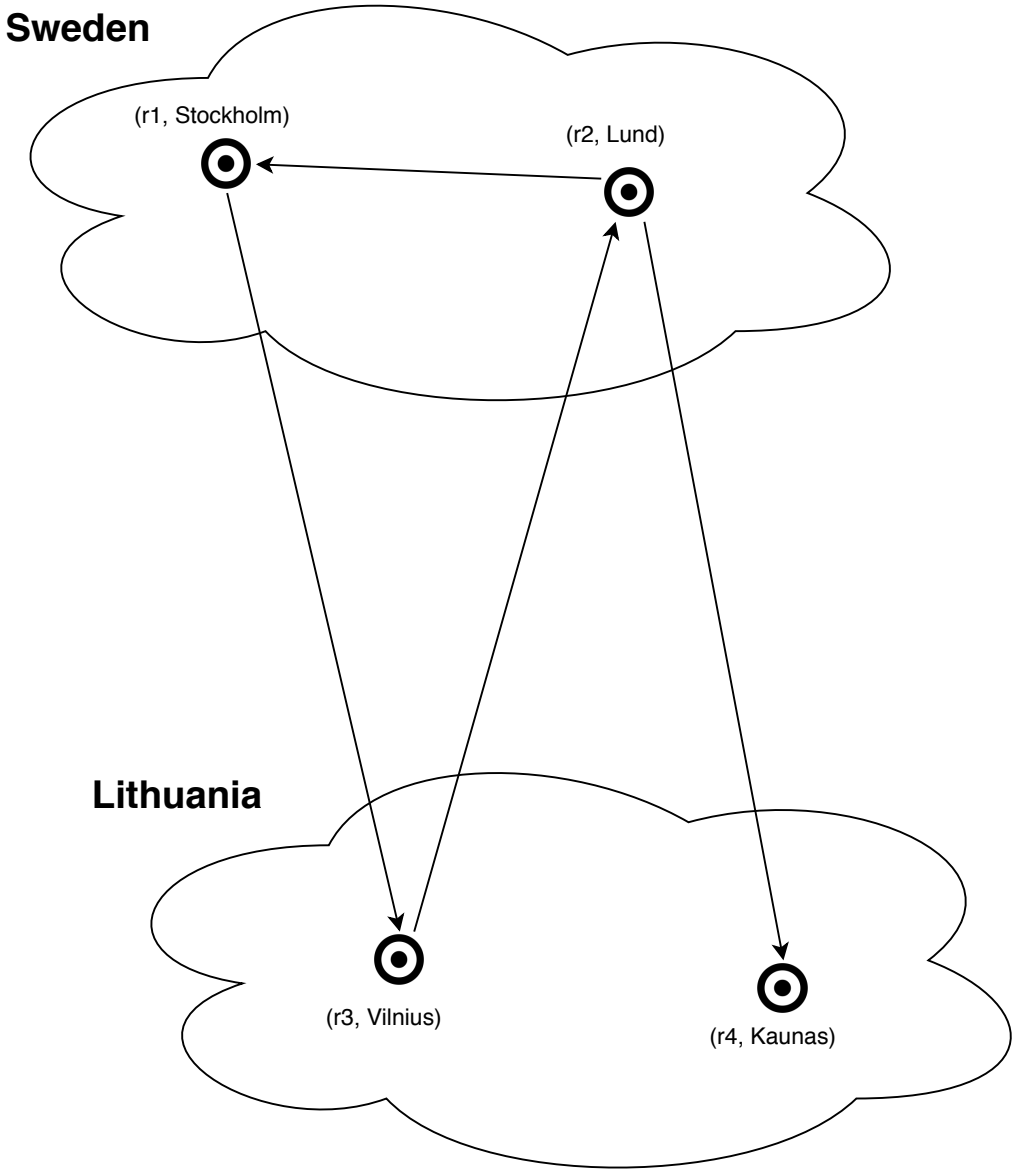

Figure 2: Test case 2