Routy - Lab report

A simple message routing system

Distributed systems ID2201

KTH

Hannes Rabo <hrabo@kth.se> - 2017

1 Chapter 1

Introduction

In this lab a simple message routing system is to be constructed. The said system should be implemented in erlang using the built in message system. As this is not a production system and does not have strict performance requirements we are not trying to save data in our messages. The protocol has been specified in instructions for the assignment to ensure that clients are compatible with each other.

The nodes in the network should use a link state routing algorithm where each node is broadcasting the directly connected nodes to the rest of the network which passes on this information. After all nodes have information of the connections in the network, Dijkstra's algorithm can be used to calculate the shortest paths through the network and create a routing table later used for incoming messages.

This assignments builds upon code provided for the course and extends some parts to complete the functionality.

2 Chapter 2

Difficulties and improvements

One of the major difficulties of this lab was to properly understand what behaviour that was expected from the implementation. Most files/modules had a very specific interface and all functions were named and had a short description. It was not on the other hand not very specific about what the behaviour was in a whole and specifics about different functions. Personally I think it would have been easier to implement Dijkstra's algorithm if it had'nt been split in to multiple different functions without a very clear connection between them. This can on the other hand be a good guidance when you don't know where to start.

The second difficulty I had during this project was when trying to run the entire program the first time and connect routers. As most of the code for this was alredy provided, I did not have great insight in how the routers were intended to operate on connection. I later realized that every single aspect of the network had to be manually operated and configured which largely defeats a purpose of a network like this. This feature on the other hand is relatively easy to add as an extension to the project and would increase the usability of this router network a lot.

There are a number of things that could be altered in this network message service depending on the use case as this clearly is not anywhere near a usable product in the current state. The first thing to add to this project would be to add more autonomous operation, as already stated above. The network detects disconnecting routers but it really needs to be easier to connect new router as well as some features that automatically recalculates routes and reconnects to other nodes. One feature that would greatly improve the stability of a message network based on this is the ability to automatically restart nodes that dies. The must also connect to the rest of the network in this case which is not a feature that exists in the current state.

Another problem that probably requires more changes to fix has to do with data and network structure. In this network every node needs to know exactly where each node is located in the entire network which can be quite many for real use cases. One solution to this problem is to divide the network in hierarchical zones where we only route messages to a unit closer to the target that hopefully knows more than

we do about the final destination. This is not a problem for a very small network of nodes as the data structures are more than capable to handle use cases connected school or academic research.

At the end of the work (during the phase where I tried to connect to multiple other students) I experienced the most complicated problems. It seemed like the code from the other students had multiple problems and was not able to perform the required tasks correctly which slowed down the development process for the network we were trying to build.