Report 2: Routy

Mallu Mallu

September 22,2016

1.Introduction

I have implemented a router for routing messages between different nodes using link state protocol. In this router messages can be passed through one-way link within the connected routers. Also, a routing table is maintained containing all the information of reaching the nodes through their respective gateways. Dijkstra's algorithm has been used in creating the routing table.

2. Main Problems & Solutions

a. **Creating Map:** For implementing the message routing the first step we needed is to create a map which should help in finding the nodes which are connected to each other directly. Also, when a node adds another new node in its network this map should get updated with the new information, this functionality has been implemented creating a "map.erl" program in erlang in which I have implemented few functions for updating the information of a particular node and its link as well to find all the reachable nodes from a given specific node. Also, map can be used to view the full list of nodes present in the routing network. This has been implemented using some of the erlang **lists** library functions such as keydelete (updates a list with new value and deletes the previous entry), keysearch (searches for a given value in the list until as match is found) and usort (returns a sorted list).

b. **Routing Table:** Routing table is the most important part of this homework and I found this the difficult one. For constructing the table Dijkstra's algorithm has been used. In the table a sorted list of entries for all the nodes in network has been maintained showing the gateways available to reach that respective node. Initially, I implemented the entry, update, iterate, table, replace and route functions in "Dijkstra.erl" erlang code. These functions made it easy to update the shortest path which can be used to reach as node. The principal behind this algorithm is shortest path first. So if we find a node reachable in say 1 hop which was previously reachable in 2 hops, the entry containing the shortest path will be saved in the table and other will be deleted (i.e. replaced). Again I

have used lists library functions keymerge (merges two given lists), map (applies a given function on the given list and returns a result list) and foldl (applies a function starting from a given element of the given list) in the code. The member function from the list library returns true if the given element is present in the list thus it has been used to construct table of nodes reachable through gateways.

c. **Interfaces Record and History:** Every time we start a router we need to keep record of the references and process id that has been implemented in this home work by using interfaces, naming a new router(process) with several non-similar names. Also, in this program I have implemented a function broadcast which can be used to broadcast the given message to all the interfaces. Other problem that is faced in this link state protocol routing is resending of messages that are already sent. So I have created a program “hist.erl” in which I have used **dict** library functions append and erase which will keep track off the messages that are already sent as old messages and if the message is new it will update the history list thus keeping track of the messages sent to respective nodes.

d. **Implementing Routy (router):** We can start a command prompt shell and name it according to our choice and can start many routing nodes in it and add them in their direct link and thus update the map. After the update we can send messages from one router to another whenever we start a new router we create a new reference and this can be used to communicate between the different router nodes. Each of the router will maintain its own set of history, interfaces, map and a routing table.

3. Evaluations

I have tested the router by opening four separate command shells and giving them different names as Stockholm, UK, and Germany. After that I have started router process in each one of them for example I started *Stockholm* in *Sweden* shell, *London* and *UK* shell, *Oslo in Norway* shell and *berlin in Germany*.

After connecting Stockholm, Oslo and London triangle I have tried sending messages between them and it worked smoothly. Then I added Berlin in connection with Stockholm and Thus a network of nodes is created. Also, to finish the bonus task I killed Stockholm node and as a result the connected all the nodes received the “exit received from Stockholm” message showing that the respected node goes down. And this down notification was received in micro seconds just after closing the Stockholm node. Also communication established is one-way link communication. More challenging features can be implemented

in this router in which delivery receipt message from the receiver will be sent out in acknowledgement.

4. Conclusion

In this assignment I have learnt link state routing protocol and Dijkstra algorithm in depth. Picking shortest path to route message to the destination is the main scenario here. Also, the records for the messages and running processes can be maintained. Erlang list library is used in implementing the core functions of this router.