Assignment Report - z5119666

Run the Lsr.py in **Python 2.7**

A brief discussion of how you have implemented the LSR protocol. Provide a list of features that you have successfully implemented. In case you have not been able to get certain features of LSR working, you should also mention that in your report.

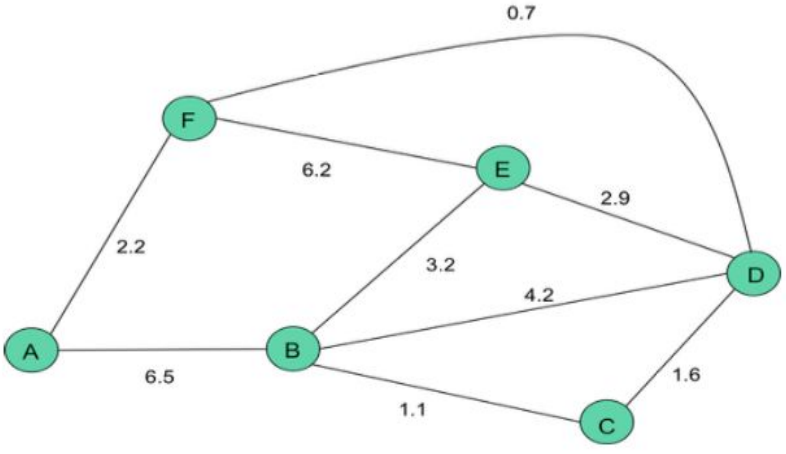
The LSR protocol includes sending data packet to neighbours, checking failures, and calculating the optimal path between the host and any other routers. Every router broadcasts packet to its neighbours every second and calculate optimal path via dijkstra's algorithm every 30 seconds. The router failure is tested via heartbeat, if there's no heartbeat for 5 second, then we consider it as 'failure'. The feature that I didn't implement was after confirming failure, the program supposed to wait for 2 route update interval which is 60 seconds. The program consists of 3 threads, one for broadcasting, one for failure checking, and one for optimal path calculation.

Describe the data structure used to represent the network topology and the link-state packet format. Comment on how your program deals with node failures and restricts excessive link-state broadcasts.

The network topology structure was simple. Based on the knowledge i leant from COMP2511, the program includes two structure, Router and Link. Link consists of destination router and the cost of this link. Router consists of router id, router port, and a list of links connected to it. The link-state packet format is as the combination of sender id, space, sender port in the first line, and the rest are all neighbours data which is same as config?,txt data format and starts from the second line.

For example (from assignment specification), the packet sent from router A is as the format below:

A 5000
2
B 6.5 5001
F 2.2 5005



For failure checking, whenever a router sends a link-state packet, the routers which receives it increment one to the heartbeat for the router that form that packet. Every 5 seconds, the program check the heartbeat for each router, if any router's heartbeat is zero, then we mark it as failure. Further, I used a 'activeRouters' list to keep the routers that works fine. For unnecessary broadcast, I used 2 lists called 'broadcasted' which includes the router that formed this packet and 'sentRouters' which contains the router that formed this packet and the routers that have connections with the

current router (i.e. the routers that received the packet the same time as current router) to keep the routers that have seen this packet and don't send to them again.

Discuss any design trade-offs considered and made. List what you consider is special about your implementation. Describe possible improvements and extensions to your program and indicate how you could realise them.

Nothing special. Possible improvements would be instead of storing routers id in different list, it may better to keep the router struct in a list. And also if the router is completely isolated from the main network topology then it cannot be detected at all. For example, in the topology shown above, if router B and F failed first, then fail A, the failure of Router A cannot be acknowledged, although Router A will no work.