Network assignment 2 Report z3492998

This assignment is run on Python 2.7

Program design

Outline

The assignment is run in a multi thread environment. There are 4 thread which handle link-state broadcasts, heartbeat, receive packet and finding shortest path. Link-state broadcast happen every 1s. Heartbeat wakes every 0.3s. And the shortest path get updated every 30s as the spec required. The rest of time will be in thread receive.

Data structure

- Every node store their neighbors in list with 4 attribute: name, cost, port and a Boolean to indicate if it is dead or life.
- Node also has a graph list to store the existing path that it has received.
- For each neighbor in node, there is a history list to record path sent to its neighbor.
- A 2d list to store the cost for Dijkstra algorithm.

Restricts excessive link-state broadcasts

There is a history list for each neighbor node. Every time when the link-state is broadcast, it would only send the path it has in graph list that is not in that history list.

Deals with node failures

There is a Boolean for each neighbor that get set to false every time after broadcast. When the receive thread get the packet from that neighbor node, it will reset that Boolean to true. If it does not receive any packet from a neighbor node in any one second, the node would be consider as dead. Then, the dead message will be send to its neighbor saying that the specific neighbor is dead. After that the other node that is not the neighbor will receive the message as well. It will resend it if it is the first time it hear that specific node is dead.

Design tradeoffs and improvements

In my code, I mainly use list to store the path as most of the list is embedded to another list. It is easier for me to write to program. However, dictionary is faster in performance. Therefore, to improve the code, I can change most of my list to dictionary.