Computer Networks and Applications (COMP3331)

Programming Assignment:

Routing Performance Analysis

Nicholas Ho z3422527

Steven Ru z3418434

# Data Structures

We have chosen to represent the topology with an undirected graph, which is made up of 3 Java classes:

### Node

This represents each of the routers in the network.

Contains 3 fields:

* **String** name: The name of the node (A or B or C etc)
* **Double** minDistance: Keeps track of the minimum distance this particular node is from the start Node. Required for Dijkstra’s Algorithm.
* **Node** prev: Keeps track of the previous Node object that was visited before the current Node was visited.

### Edge

This represents each of the links in the network. Since the graph is an undirected graph, we represent each undirected Edge with 2 directed Edges. Any operations on such an undirected Edge will affect both Edges running in either direction.

Contains 5 fields:

* **Node** from: Contains one of the two Nodes at either end of the Edge.
* **Node** to: Contains one of the two Nodes at either end of the Edge.
* **Int** propagationDelay: Propagation delay is an argument taken in from topology.txt.
* **Double** numSimulCircuits: numSimulCircuits is an argument taken in from topology.txt.
* **ArrayList<VirtualCircuit>** circuits: This is the list of Virtual Circuits that are currently using this particular Edge as a Link.

### UndirectedGraph

This represents the entire network.

Contains 2 fields:

* **ArrayList<Node>** nodes: Contains the list of all nodes or routers that are involved in this network.
* **ArrayList<Edge>** edges: Contains the list of all edges or links that are involved in this network.

# Comparison of Performance Metrics (VCN)

Temporary values. Still need to fix bug regarding bidirectional edges.

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | Total VC | Total  pkt | Succ. Routed  pkt | % routed  pkt | Num blocked  pkt | % blocked  pkt | Avg hops | Cum. Prop delay |
| SHP | 8377 | 259106 | 231891 | 89.50 | 27215 | 10.50 | 2.37 | 167.48 |
| SDP | 8377 | 259106 | 236516 | 91.28 | 22590 | 8.72 | 3.04 | 140.49 |
| LLP | 8377 | 259106 | 246764 | 95.24 | 12342 | 4.76 | 3.82 | 239.21 |

# Analysis of Results

As can be seen, Total VC and Total packets are not affected by the type of algorithm used, and remain constant at 8377 and 259106. However, the number and percentage of successfully routed packets is slightly higher for SDP (236516, 91.28%) than SHP (231891, 89.50%). Correspondingly, the number and percentage of blocked packets for SDP (22590, 8.72%) is lower than that of SHP (27215, 10.50%)

The reason for the lower amount of blocked packets for SDP could be the higher average number of hops (3.04) than SHP (2.37). Because a greater number of links are involved in SDP, there are (3.04 – 2.37 = 0.67) more links to take some of the load, as compared to SHP, where the load is shared between only 2.37 links. This means that there is a lower likelihood that the number of virtual circuits involved in the links in SDP will exceed the numSimulCircuits field in each Edge than in SHP.

Additionally, the cumulative propagation delay for SDP is 142.25, smaller than that of SHP, 173.53. This smaller propagation delay is directly beneficial for SDP, given that the SDP algorithm uses propagation delays of the Links as the length of the path taken. This means that packets take a shorter time to travel the distance from source node to destination, and its resources can be freed for use by other packets more quickly.

Together, the greater average hops yet lower propagation delay for SDP allows it to route more packets successfully, resulting in fewer blocked packets.

# Evaluation of Virtual Packet Network(VPN)

Table

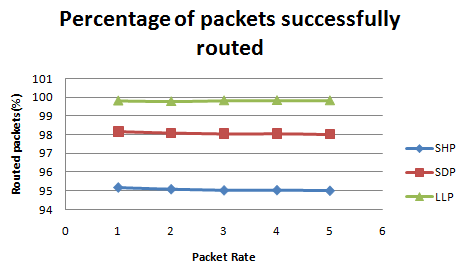
|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| SHP |  |  |  |  |  |  |  |  |
| Packet Rate | Total VC | Total | Succ. Routed | % routed | Num blocked | % blocked | Avg hops | Cum. Prop delay |
| pkt | pkt | pkt | pkt | pkt |
| 1 | 259106 | 259106 | 246606 | 95.18 | 12500 | 4.82 | 2.67 | 170.74 |
| 2 | 514032 | 514032 | 488778 | 95.09 | 25254 | 4.91 | 2.67 | 170.64 |
| 3 | 768919 | 768919 | 730715 | 95.03 | 38204 | 4.97 | 2.67 | 170.65 |
| 4 | 1023878 | 1023878 | 973100 | 95.04 | 50778 | 4.96 | 2.67 | 170.63 |
| 5 | 1278849 | 1278849 | 1215091 | 95.01 | 63758 | 4.99 | 2.67 | 170.64 |

Table

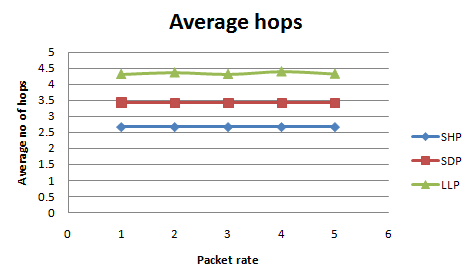
|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| SDP |  |  |  |  |  |  |  |  |
| Packet Rate | Total VC | Total | Succ. Routed | % routed | Num blocked | % blocked | Avg hops | Cum. Prop delay |
| pkt | pkt | pkt | pkt | pkt |
| 1 | 259106 | 259106 | 254344 | 98.16 | 4762 | 1.84 | 3.43 | 141.15 |
| 2 | 514032 | 514032 | 504220 | 98.09 | 9812 | 1.91 | 3.42 | 141.16 |
| 3 | 768919 | 768919 | 753963 | 98.05 | 14956 | 1.95 | 3.42 | 141.16 |
| 4 | 1023878 | 1023878 | 1004003 | 98.06 | 19875 | 1.94 | 3.42 | 141.14 |
| 5 | 1278849 | 1278849 | 1253488 | 98.02 | 25361 | 1.98 | 3.42 | 141.17 |

Table

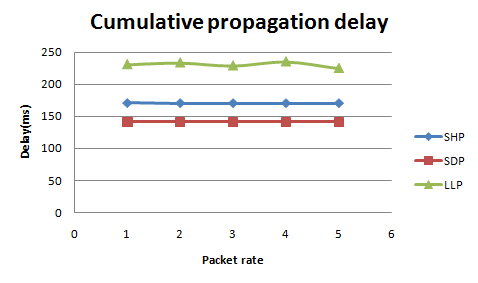
|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| LLP |  |  |  |  |  |  |  |  |
| Packet Rate | Total VC | Total | Succ. Routed | % routed | Num blocked | % blocked | Avg hops | Cum. Prop delay |
| pkt | pkt | pkt | pkt | pkt |
| 1 | 259106 | 259106 | 258629 | 99.82 | 477 | 0.18 | 4.32 | 230.87 |
| 2 | 514032 | 514032 | 512877 | 99.78 | 1155 | 0.22 | 4.37 | 233.48 |
| 3 | 768919 | 768919 | 767610 | 99.83 | 1309 | 0.17 | 4.32 | 228.94 |
| 4 | 1023878 | 1023878 | 1022222 | 99.84 | 1656 | 0.16 | 4.4 | 235.27 |
| 5 | 1278849 | 1278849 | 1276702 | 99.83 | 2147 | 0.17 | 4.33 | 224.84 |



Figure



Figure



Figure

# Analysis of Results

From the tabulated results from tables 1, 2 and 3, it is evident that there is a new virtual circuit created for each and every packet. For example in table 1 the total VC is equal to the total number of packets. This is to be expected since the packet network algorithm is supposed to generate a new virtual circuit from each individual packet. It is also clear from tables 1, 2 and 3 that as the packet rate increases, the number total number of packets increases. This is also expected since increasing the packet rate will allow for more packets to flow through the topology.

Next, the performance of the routing algorithms is analysed. From figures 1,2 and 3 it is evident that different routing algorithms have their own benefits when it comes to the amount of successfully routed packets, average hop length and cumulative propagation delay. In the case where the most successful packets are routed, it appears that the LLP routing algorithm results in the least blocked packets. By varying the packet rate, the largest percentage of blocked packets using LLP was only 0.22%, a minor very minor figure. This is expected since LLP takes the least loaded path which theoretically results in the smallest amount of blocked packets.

Next, we analyse the average hops that each algorithm creates for virtual circuits. From figure 2, it is evident that for packets rates of 1-5, SHP has the lowest average hops with around 2.6 hops. Since the algorithm is specifically designed to find the shortest hop path to the destination, this result is to be expected.

Then, we analyse the cumulative propagation delay. From figure 3 it is evident that SDP has the lowest cumulative propagation delay with a maximum of only ~150ms. Similarly, this is expected since the algorithm was designed to find the lowest cumulative delay.

Therefore, all three of the algorithms have their advantages and disadvantages for factors such successful packets, hops and propagation delay. There is no one correct algorithm to use all of the time and it must be chosen to meet the users specific needs.