Link Costs

RTT of each link (*ex: r1 ⇔ s*) is calculated by taking the average of discovery messages rtt.

On each host simultaneously starts to send 1000 discovery messages(*according to defined scenario*) to each neighbour host and each host that received a message is responds that message.

Each discovery message contains 27 to 31 characters.
Each response message(ACK) to discovery message contains 30 to 35 characters.

Example: Let's say Host r1 is sending its first discovery message to host d.

1. node r1 gets current time and saves as **`st`**
2. node r1 sends Msg1 to node d, and waits for a response from node d.
3. When node d gets Msg1 then it sends Msg2 to node r1.
4. If node r1 receives ACK from noded,
   4.1. noder1 gets current time as **`rt`**
   4.2. **`rtt = rt - st`**
   4.3. **`Total_rtt_r1 += rtt_1`**
5. …
6. If all discovery message got respond
   6.1. **`rtt_r1 = total_rtt_r1/discovery_message_num`**

Msg1*:* `"START_R1*current_time*NA*discovery_msg_id"`
Msg2: `"ACK_D*START_R1*current_time*NA*discovery_msg_id"`

Following tables are RTT's of r1, r2 and r3 to each neighbours.

r1⇔ neighbour host	**rtt(ms)**
r1 ⇔ s	262.828
r1 ⇔ r2	170.607
r1 ⇔ d	96.542

Table1: rtt of r1 with its each neighbours

r2⇔ neighbour host	**rtt(ms)**
r2 ⇔ s	92.509
r2 ⇔ r1	99.073
r2 ⇔ r3	90.555
r2 ⇔ d	170.627

Table2: rtt of r2 with its each neighbours

r3⇔ neighbour host	**rtt(ms)**
r3 ⇔ s	1.989
r3 ⇔ r2	90.467
r3 ⇔ d	2.841

Table3: rtt of r3 with its each neighbours

The Shortest Path using the Dijkstra Algorithm

Topology have 5 nodes(including source and destination nodes), which means there is 9 unique path from source to destination. And RTT of each path is calculated using table1, table2, table3.

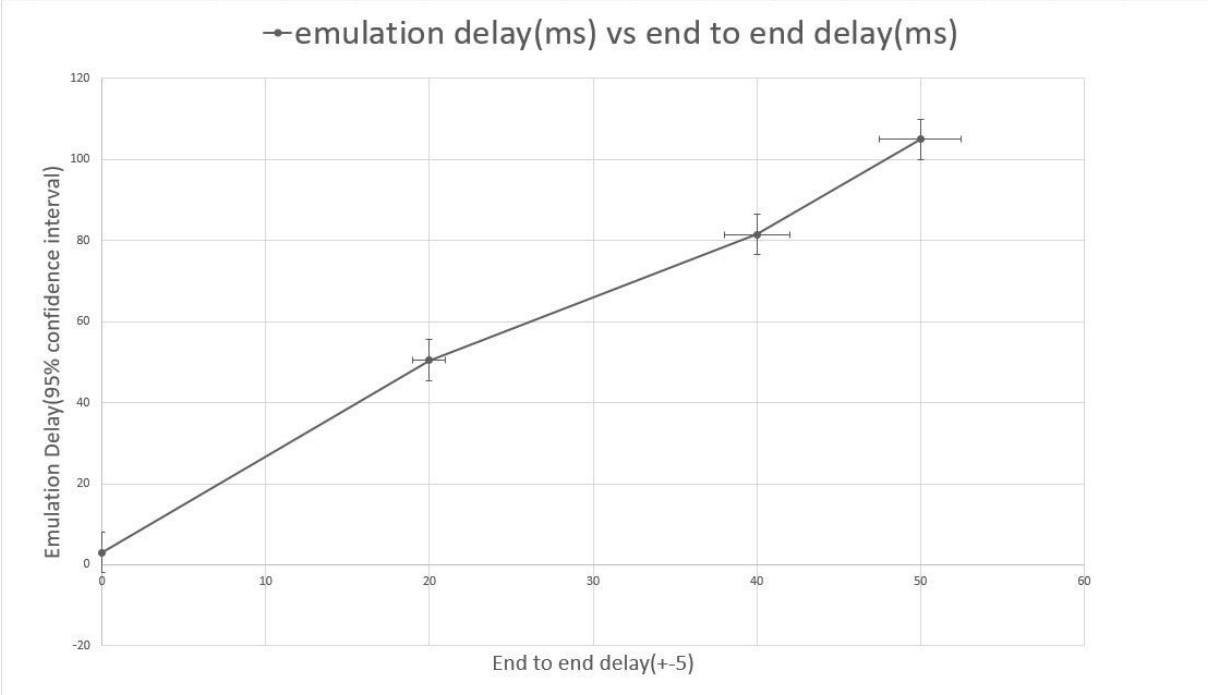
Following table shows all possible path from source node **s** to destination node **d** with link cost in milliseconds.

possible path s to d	rtt(ms)
s→r3→r2→r1→d	288.071
s→r3→r2→d	263.083
s→r3→d	4.83
s→r2→r1→d	288.124
s→r2→r3→d	185.905
s→r2→d	263.136
s→r1→r2→r3→d	526.831
s→r1→r2→d	604.062
s→r1→d	359.37

Table4: all possible paths from source node s to destination node d using Djikstra algorithm
The shortest path from source to destination is in the 3rd row(blue row)

Source code of dijkstra algorithm is present on my GitHub account:
https://github.com/nursultan-a/protocol-to-transmit-a-large-file/blob/master/all_possible_path.py

Experiment



The network emulation delay and end to end delay graph is plotted by taking the average rtt of 10 discovery message sent from source node s to destination node d using shortest path(table-4). The path from source to destination is s→r3→d and path from destination to source is d→r3→s

Discussion/Opinion

When I tried 4 times with 1000 discovery message for the first experiment I got smaller rtt than I tried with 10 discovery messages. Then I realized that, as we increase the discovery message time accordingly each experiment(delay time) at the end-end-to end delay time of each experiment will become equal to rtt of normal(without any emulation delay).