# NS3 Project Report

**Nahian Salsabil**
Student ID: 1705091

Department of Computer Science and Engineering

February 24, 2022

# Contents

# 1 Task A: Varying Parameters Without Modification

## 1.1 Assigned Network

Student ID: **1705091**

$$1705091 \bmod 6 = 5$$

**Network:**

- Wireless low-rate (e.g. 802.15.4) (Static)
- Wired

## 1.2 Wireless low-rate (802.15.4) (Static

### 1.2.1 Topology Used

Topology used: **Mesh Topology**

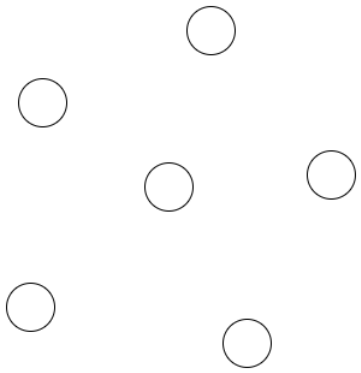


Figure 1: Wireless Low-rate Topology

### 1.2.2 Configuration

Simulation Time: 10 Seconds
Packet Size: 1024 byte

### 1.2.3 Parameters Varied

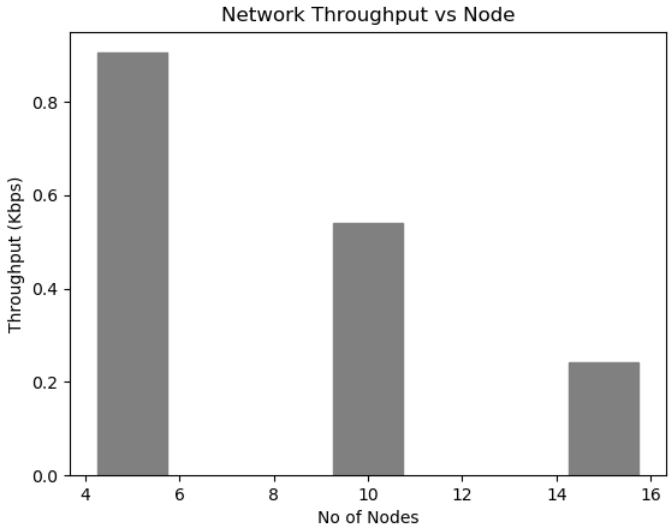
- Number of Nodes
- Number of Flows
- Number of Packets Per Second
- Coverage Area

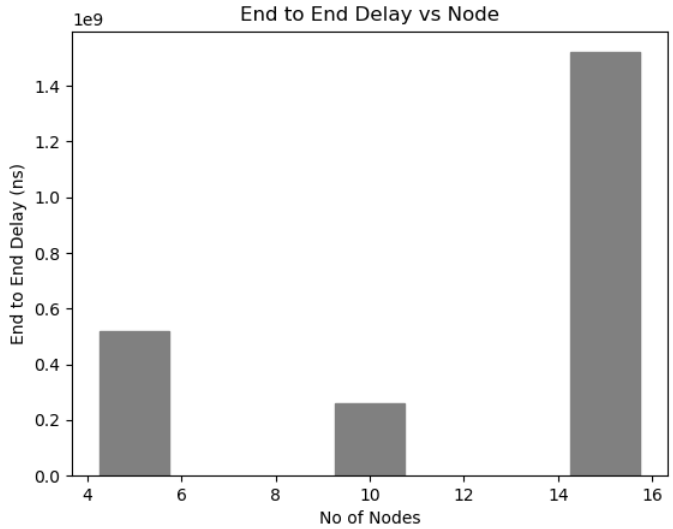
### 1.2.4 Metrics Calculated

- Network Throughput
- End to End Delay
- Packet Delivery Ratio
- Packet Drop Ratio

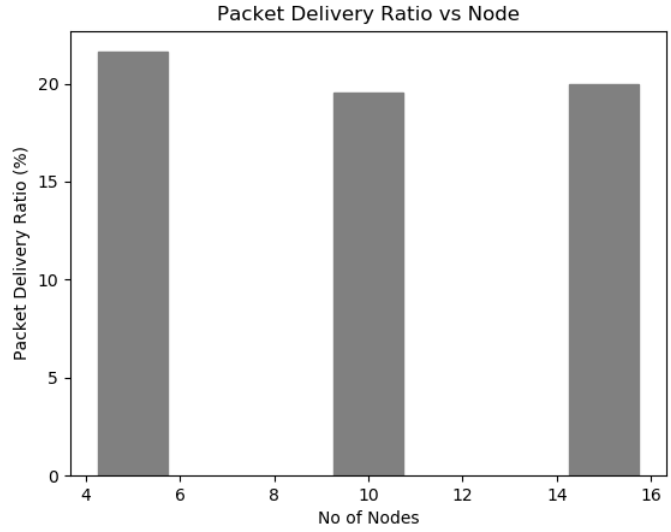
### 1.2.5 Graphs with Explanation

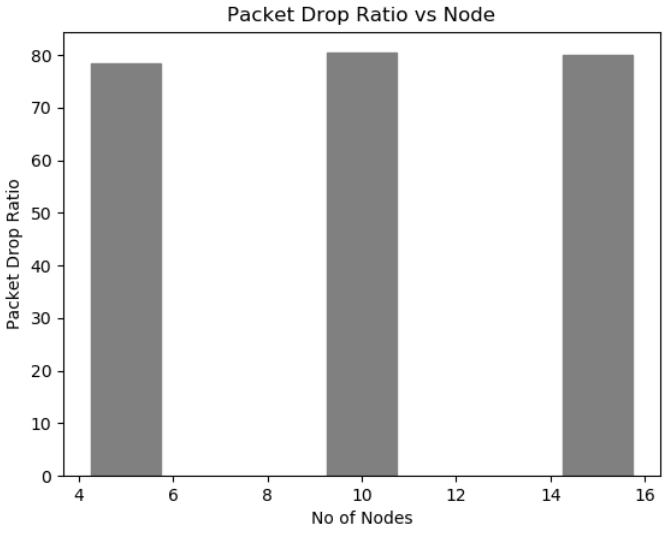
- **Metrics Varying Number of Nodes:**



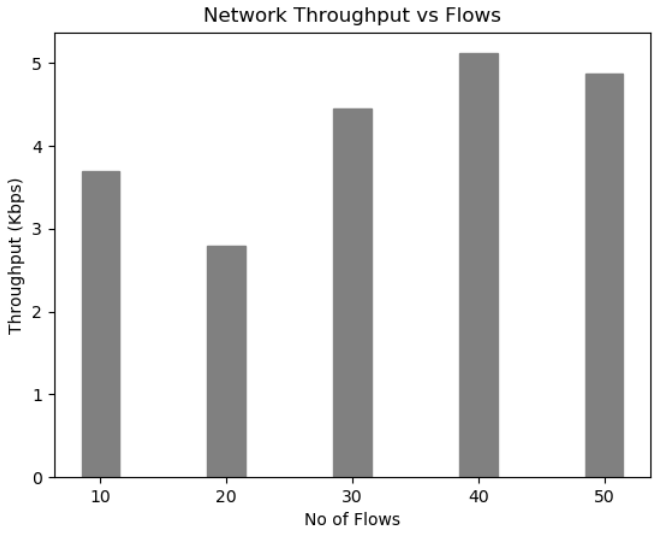



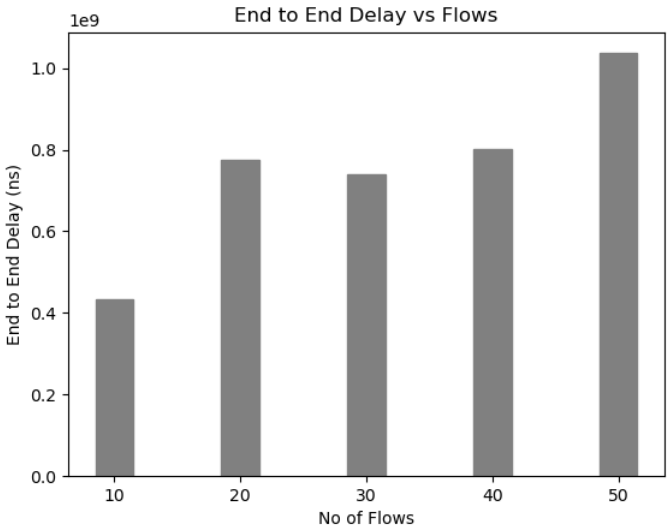

Network throughput is decreasing gradually because with the increase of the nodes traffic is also increased in the network. If we increase node farther, the network throughput becomes zero because the nodes go out of coverage area and no packet is sent.


Packet Delivery Ratio vs Node
Packet Delivery Ratio (%)
0
5
10
15
20
4
6
8
10
12
14
16
No of Nodes

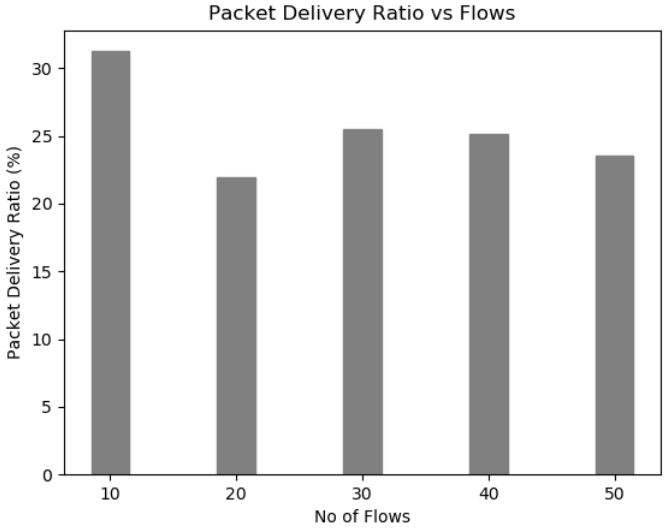

Packet Drop Ratio vs Node
Packet Drop Ratio
0
10
20
30
40
50
60
70
80
4
6
8
10
12
14
16
No of Nodes

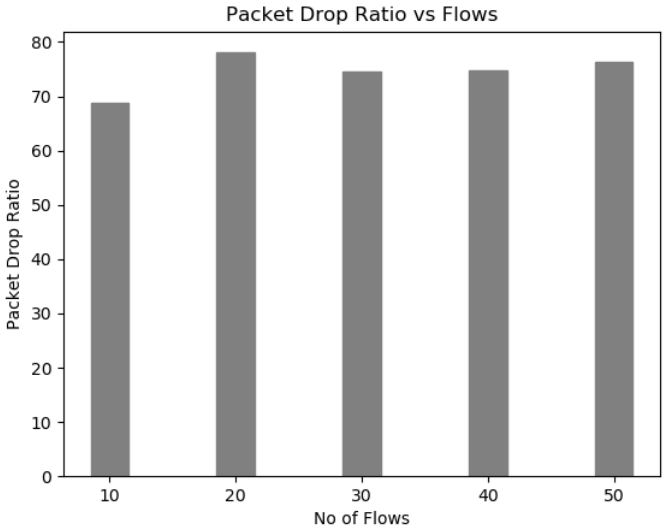
- **Metrics Varying Number of Flows:**



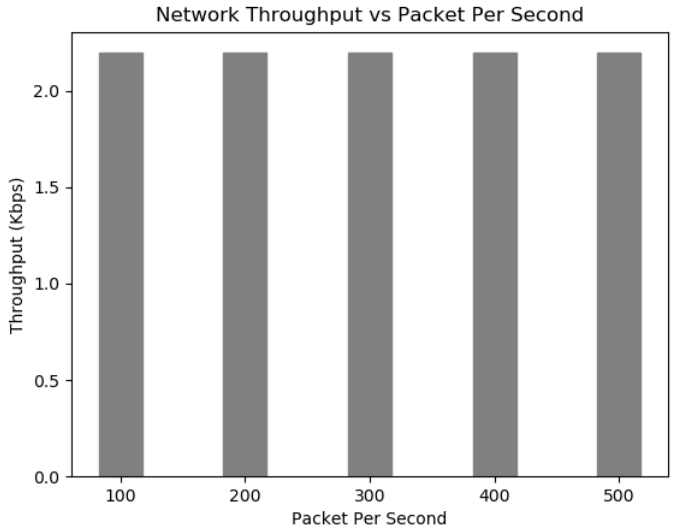

The throughput and end to end delay is being increased with the increase of the number of flow.

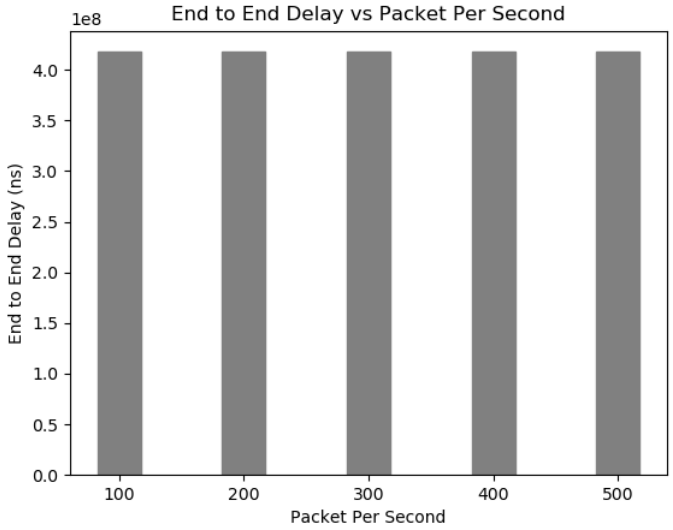


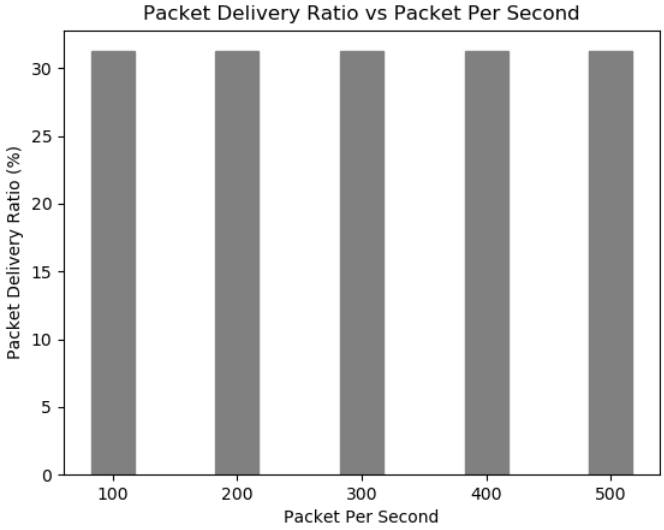

The packet drop ratio is also increased. As there is more traffic in the path there is more drop in packets.

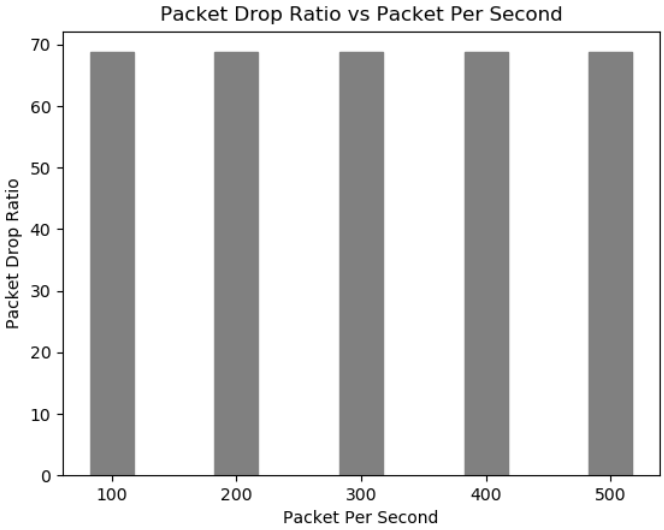
- **Metrics Varying Number of Packet Per Second:**



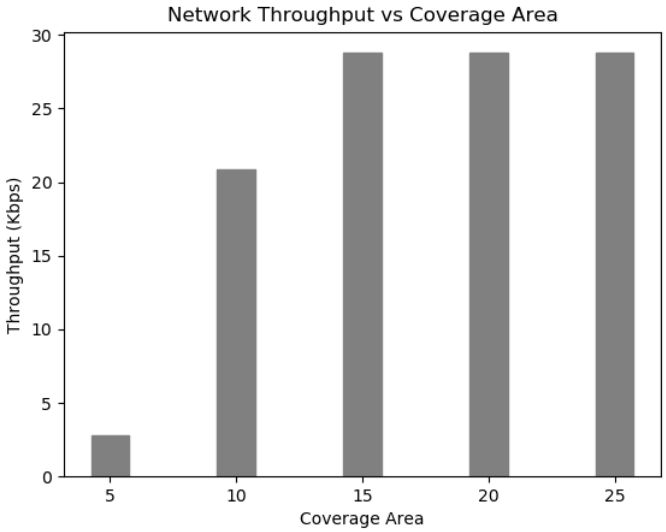

Nothing is changed with the variation of packet per second.

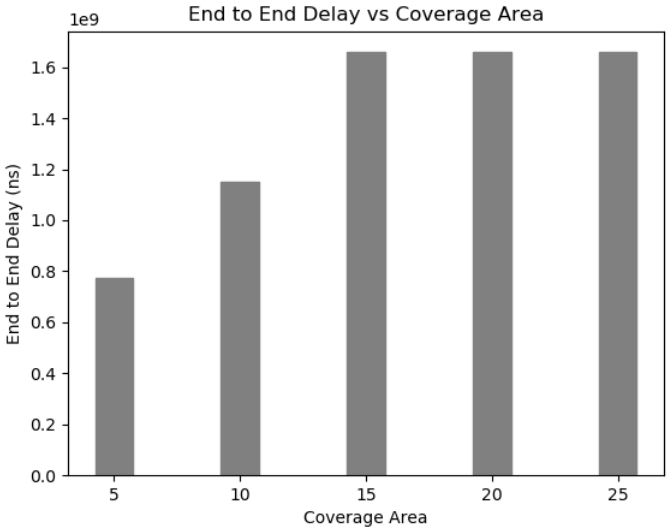


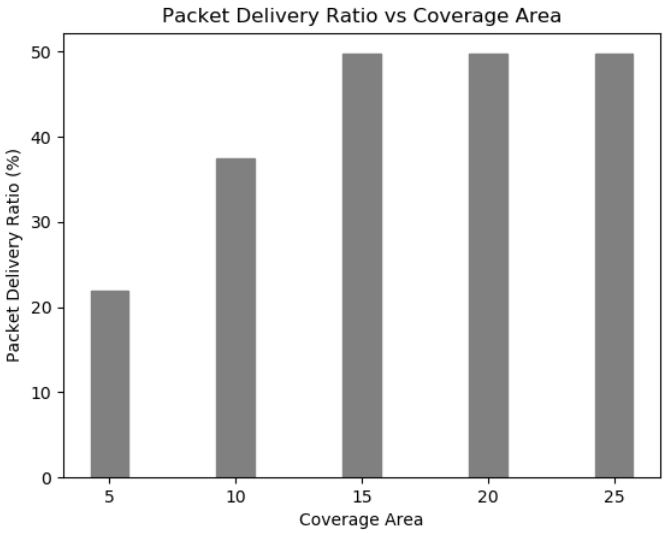

Nothing is changed with the variation of packet per second.

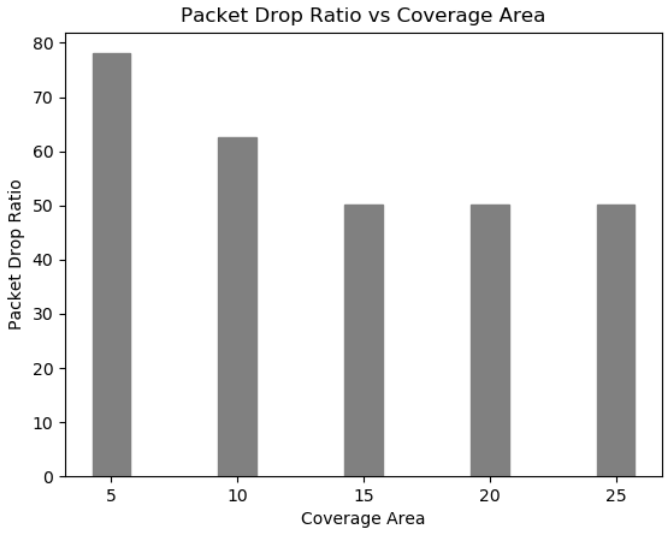
- **Metrics Varying Coverage Area:**




With the increase of the coverage area, network throughput is getting increased first. After that it doesn't change. Same thing happened with end to end delay




With the increase of the coverage area, delivery ratio is getting increased first. After that it doesn't change. Same thing happened with drop ratio.

## 1.3 Wired

### 1.3.1 Topology Used

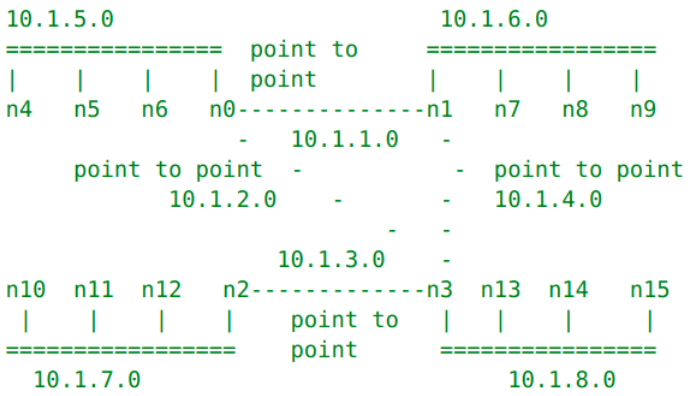



Figure 2: Wired Topology

### 1.3.2 Configuration

Simulation Time: 10 Seconds
Packet Size: 1024 byte

### 1.3.3 Parameters Varied

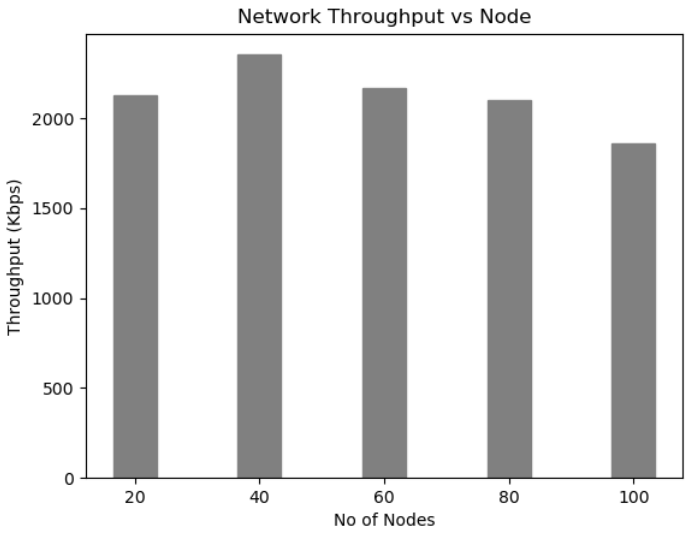
- Number of Nodes
- Number of Flows
- Number of Packets Per Second

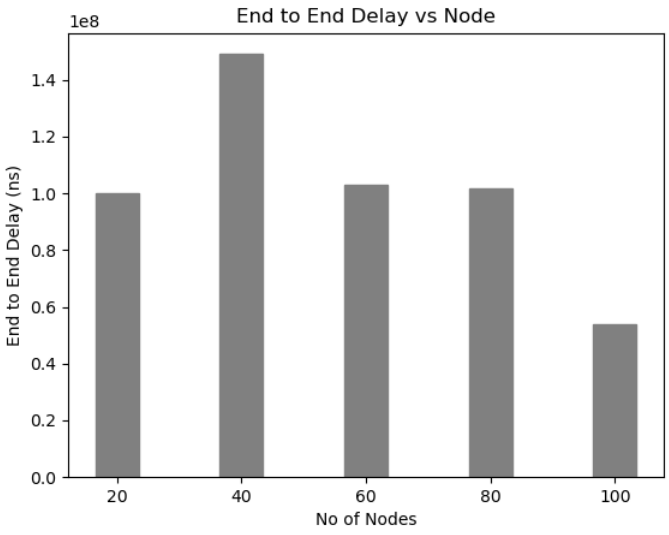
### 1.3.4 Metrics Calculated

- Network Throughput
- End to End Delay
- Packet Delivery Ratio
- Packet Drop Ratio

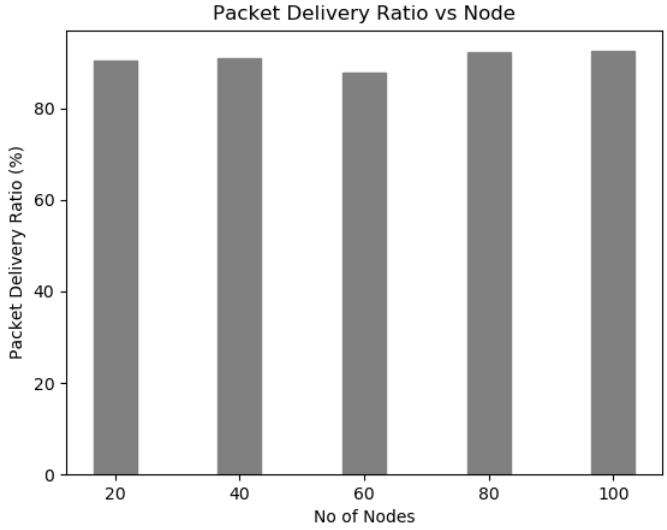
### 1.3.5 Graphs with Explanation

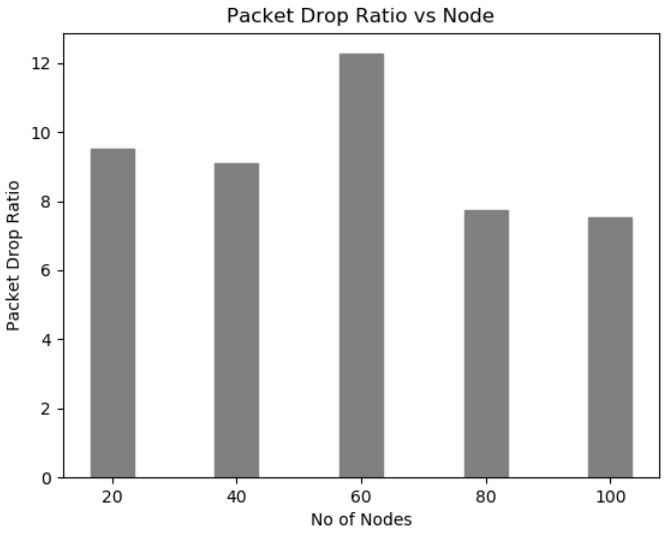
- **Metrics Varying Number of Nodes:**



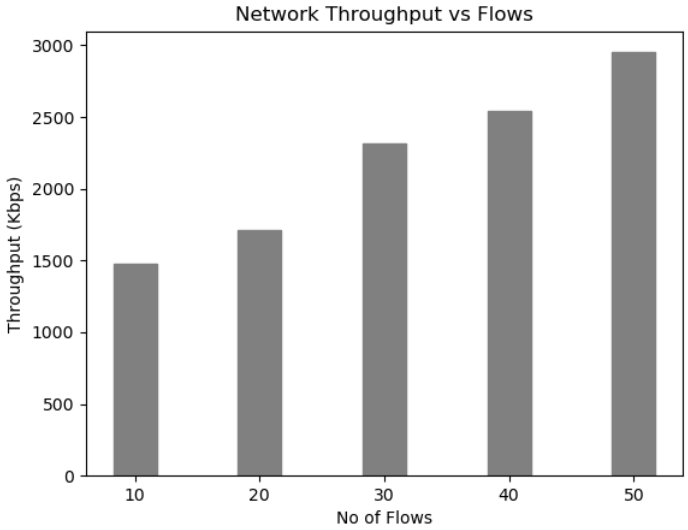

As the distance between the source node and the sink node increases due to the increase of the number of nodes, the throughput decreases.

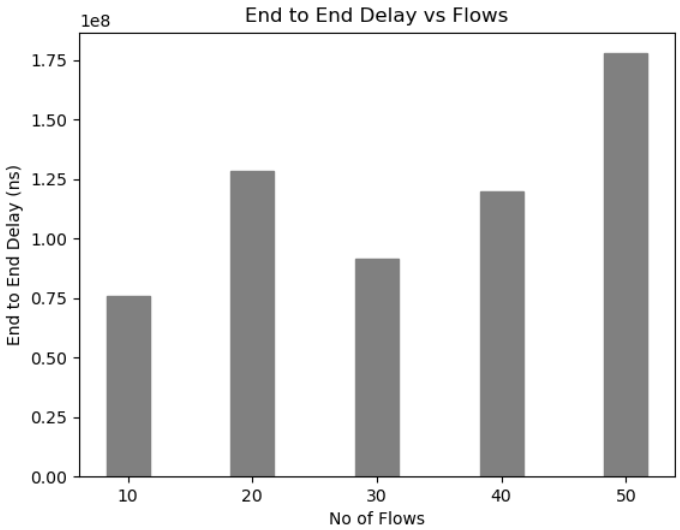


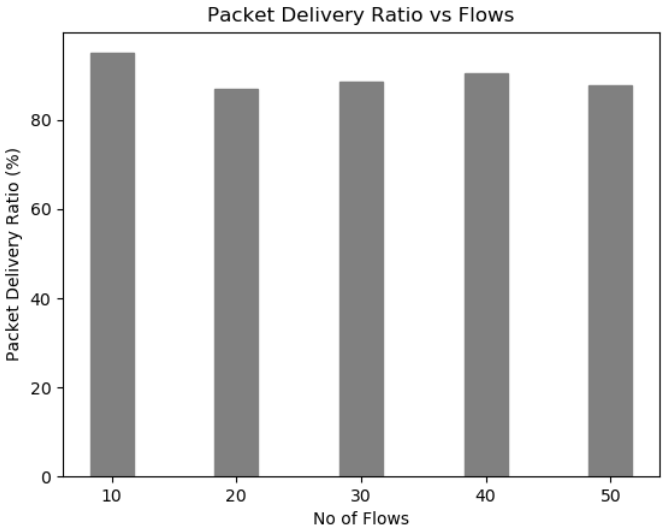

The packet delivery ratio was changed a little throughout the variation of the number of nodes.Same goes for the packet drop ratio.

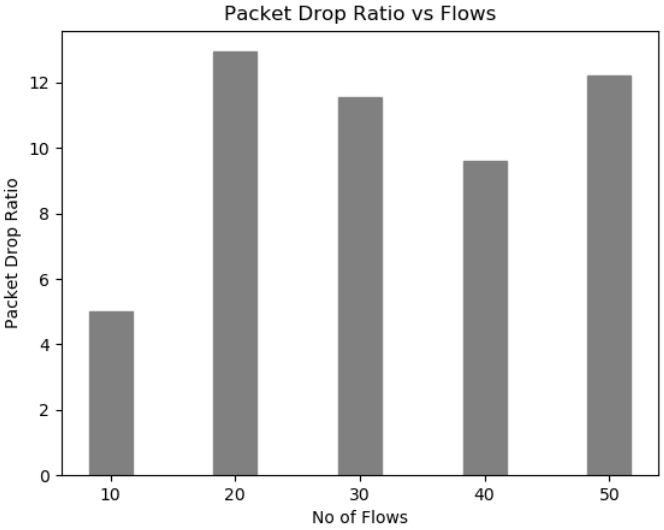
- **Metrics Varying Number of Flows:**



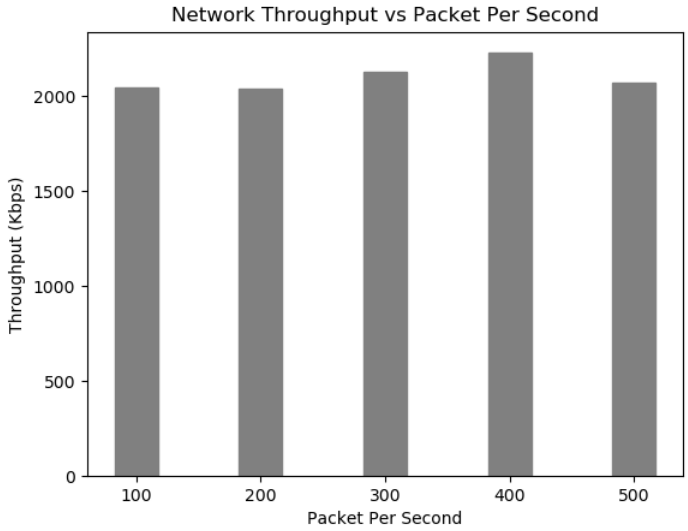

The throughput and end to end delay is being increased with the increase of the number of flow.

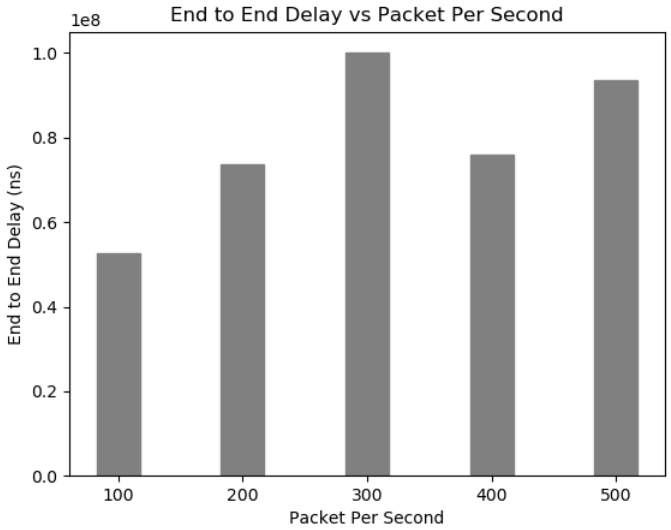


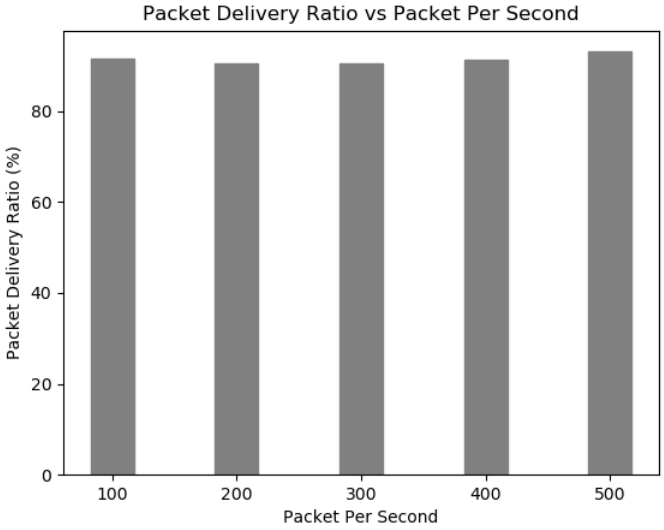

The packet delivery ratio was changed a little throughout the variation of the number of nodes.Same goes for the packet drop ratio.

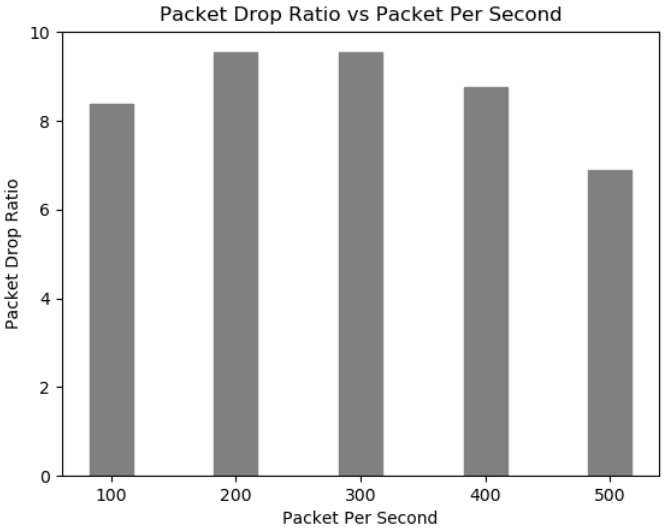
- **Metrics Varying Number of Packet Per Second:**




Throughput is barely changed by varying the number of packets per second.




Delivery Ratio is barely changed by varying the number of packets per second.

# 2 Task B: Simulation with Proposed Modification

**Paper Name:** *CUBIC-FIT: A High Performance and TCP CUBIC Friendly Congestion Control Algorithm*

## 2.1 Overview of the Proposed Algorithm

Proposed Algorithm : *CUBIC-FIT*

This is a modified congestion control algorithm of TCP CUBIC (default congestion control algorithm in NS3). Cubic-fit is a delay-based TCP to extend the CUBIC algorithm framework. It simulates N no of flow in a single TCP connection to fully utilize network capacity. Improvement:

- Performance over large range of network
- Throughput performance over wireless network
- Decrease end-to-end delay
- Maintain graceful friendliness with plain CUBIC networks

## 2.2 Implemented Network

Wireless High-rate (e.g 802.11) (Static)

## 2.3 Topology Used

Topology: Dumbbell

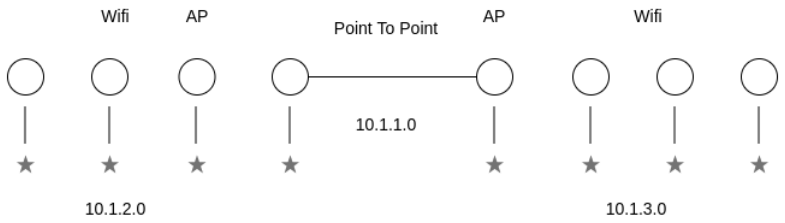



Figure 3: Wireless High-rate Topology

## 2.4 Modification made in the simulator

**In Equation:**

In CUBIC:

$$\omega_{cubic} = C(t - I)^3 + \omega_{max} \tag{1}$$

$$I = \sqrt[3]{\omega_{max} b / C} \tag{2}$$

In CUBIC-FIT:

$$\omega_{cubic}^{fit} = 0.4(Nt - I)^3 + \omega_{max} \tag{3}$$

$$I = \sqrt[3]{10\omega_{max}/19N + 1} \tag{4}$$

where

$$\frac{C(4-b)}{4b} = 1.9N^4 \tag{5}$$

$$N_{t+1} = \max\{1, N_t + 1 - \frac{RTT_t - RTT_{min}}{\alpha \cdot RTT_t} N_t\} \tag{6}$$

$$\alpha = \min\{\frac{1}{10}, \frac{RTT_{max} - RTT_{min}}{2RTT_{max}}\} \tag{7}$$

**In Simulator:**

```
else
  {
    // Cubic Fit Modification
    m_bicK = std::pow ((10*m_lastMaxCwnd) / (19*N_t + 1), 1 / 3.);
    m_bicOriginPoint = m_lastMaxCwnd;
    NS_LOG_DEBUG ("lastMaxCwnd > m_cWnd. K=" << m_bicK <<
                  " and origin=" << m_lastMaxCwnd);
  }
```

Figure 4: Modification 1

```
//Cubic Fit Modification
m_alpha = std::min(1/10.0, (m_maxRtt.GetSeconds() - m_minRtt.GetSeconds())/(2*m_maxRtt.GetSeconds()));
N_t = std::max(1.0, (N_t + 1 - ((m_currRtt.GetSeconds() - m_minRtt.GetSeconds())/(m_alpha*m_currRtt.GetSeconds())*N_t)));


if (t.GetSeconds () < m_bicK)       /* t - I */
  {
    offs = m_bicK - (N_t * t.GetSeconds ());     // Cubic Fit Modification   (I-Nt)
    NS_LOG_DEBUG ("t=" << t.GetSeconds () << " <k: offs=" << offs);
  }
else
  {
    offs = (N_t * t.GetSeconds ()) - m_bicK;    //  // Cubic Fit Modification  (Nt-I)
    NS_LOG_DEBUG ("t=" << t.GetSeconds () << " >= k: offs=" << offs);
  }
```

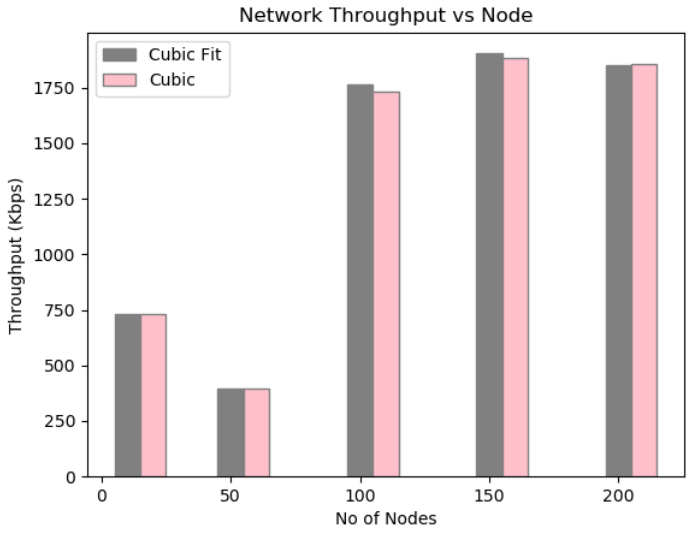
Figure 5: Modification 2

## 2.5 Performance Graph

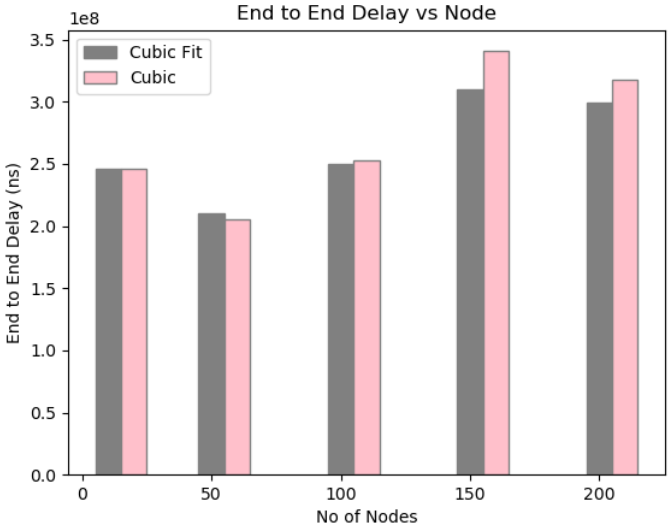
**Parameter Varied:** *Number of Nodes*
**Metrics Calculated:**

- Network Throughput
- End to End Delay

**Graph:**




The network throughput has been increased in Cubic-Fit for larger network than Cubic.




The end to end delay has been decreased significantly in Cubic-Fit.

## 2.6 Fairness Measurement

Fairness Metric: *Jain's Fairness Index*

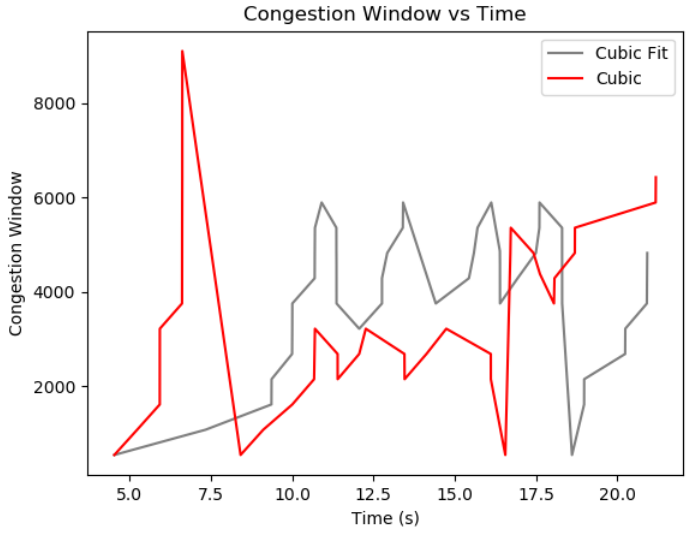
| Propagation Delay | Cubic fit | Cubic |
|---|---|---|
| 10 | 0.495 | 0.48 |
| 50 | 0.537 | 0.497 |
| 100 | 0.569 | 0.4577 |
| 150 | 0.559 | 0.5037 |
| 200 | 0.557 | 0.577 |

Figure 6: Jain's Index of Different TCP

From the table, the Jain's fairness index of Cubic-Fit has been improved than Cubic. So, it can be said that TCP CUBIC-FIT is quite fair with TCP CUBIC.

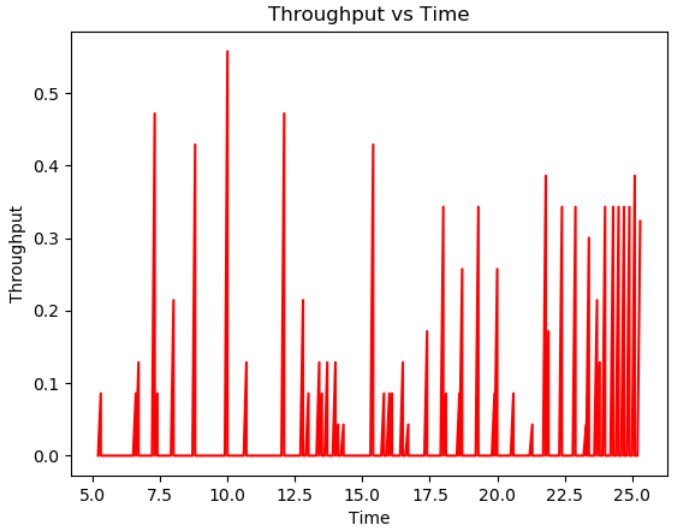
**Additional Metric:**

- Congestion Window vs Time




The Congestion window size has been increased in Cubic-Fit.

- Throughput vs Time for CUBIC




- Throughput vs Time for CUBIC-FIT