Bangladesh University of Engineering and Technology

Department of Computer Science and Engineering

CSE 322 - Computer Networks Sessional

Report Of NS2-Project

TCP-FIT: An improved TCP congestion control algorithm and its performance

Submitted by:
Abdur Rafi
1805008

Supervisor:
Dr. Md. Shohrab Hossain
Professor
Department of Computer Science and Engineering, BUET.

Date of Submission
27 February, 2023

Table of contents

Chosen Paper

Paper link

[TCP-FIT: An improved TCP congestion control algorithm and its performance]

Authors

[Jingyuan Wang]
[Jiangtao Wen]
[Jun Zhang]
[Yuxing Han]

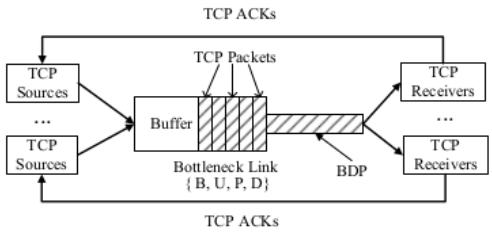
Intuition behind the TCP-Fit algorithm

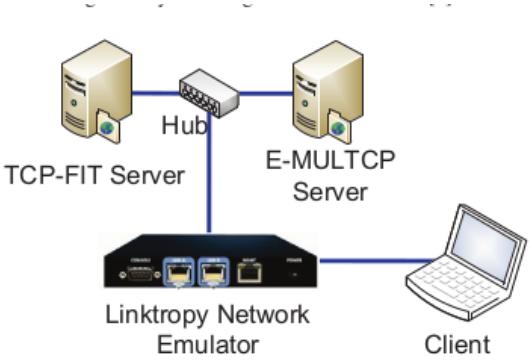
Tcp algorithms in general fail to maintain performance in wireless high BDP networks. In such networks there are packet drops and high delays. TCP algorithms may think that the network is congested and send less packets. As a result, bandwidth utilization decreases.

The TCP-Fit algorithm takes a different approach to solving this issue. It simulates N parallel connections through the congestion window. N is dynamically changed based on network condition. The idea is similar to parallelism in computer hardware. When we failed to increase the performance of a single cpu by a good margin, we employed parallel cores to increase performance. I understood similar kind of intuition behind the TCP-Fit algorithm

Network Topologies Under Simulation

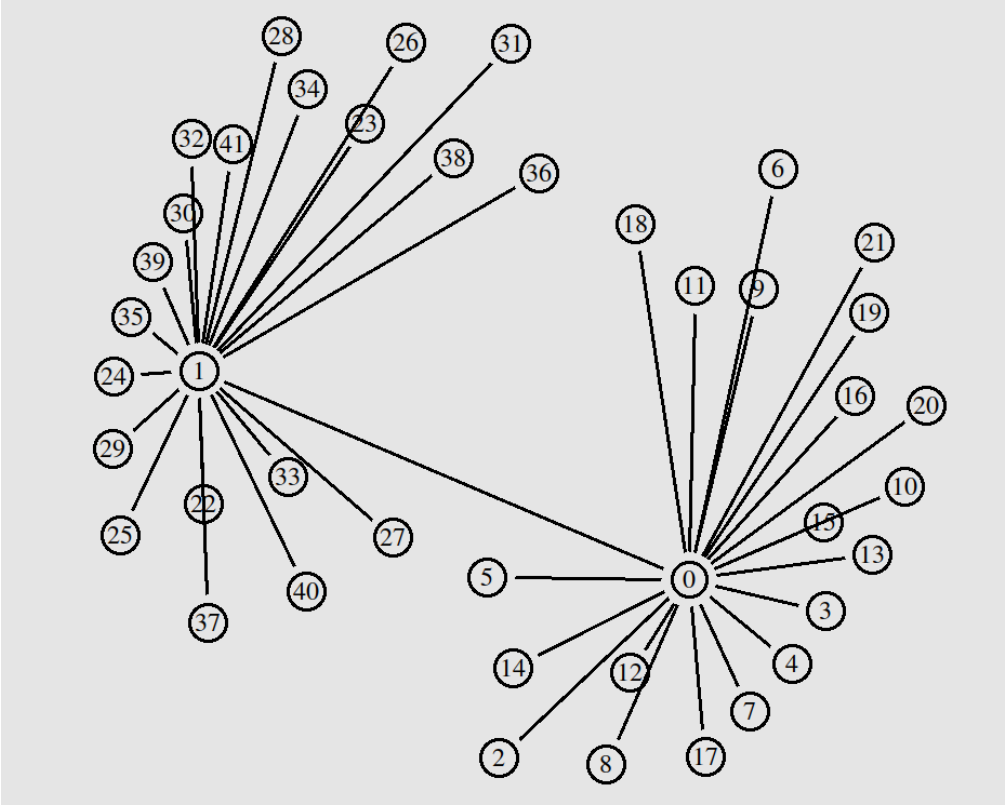
Topology used in paper





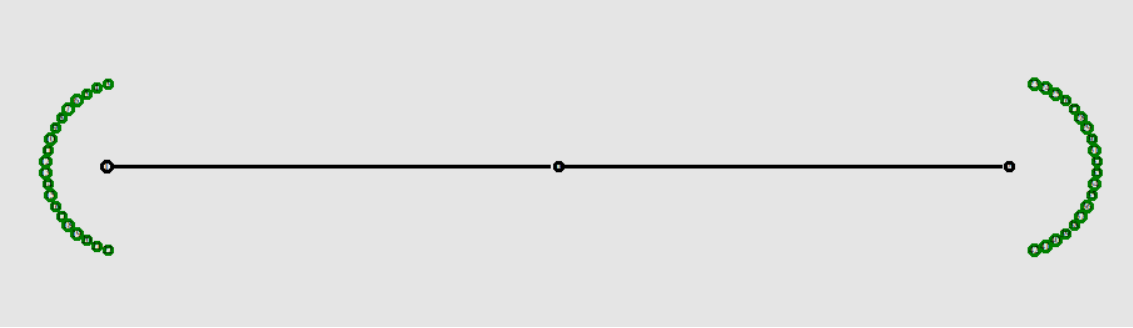
Topology simulated

Wired:



Single High BDP Bottleneck Link

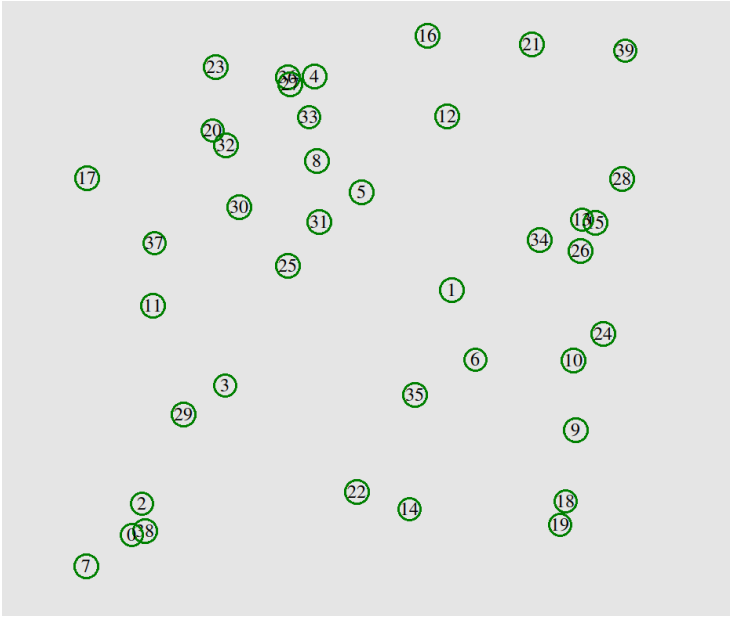
Wired and Wireless:



Single High BDP Bottleneck Link

Wireless:

High BDP Link Simulated in Wireless Network



Wireless Random Nodes

Wireless Grid Network

Parameters Under Variation

1. Number of Nodes (20, 40, 60, 80, 100)
2. Number of flows (10, 20, 30, 40, 50)
3. Number of packets sent (100, 200, 300, 400, 500)
4. Speed (for wireless nodes) (5 m/s, 10 m/s, 15 m/s, 20 m/s, 25 m/s)

Modifications Made In The Simulator

Equations implemented

$$Each\ \ RTT:\ w \leftarrow w + N,$$
$$Each\ \ Loss:\ w \leftarrow w - \frac{2}{3N+1}w.$$

Where w is the congestion window
N is a dynamic parameter which is updated after each packet loss

$$N_i = \begin{cases} N_{i-1} + 1 & ,\text{if}\ \ Q < \alpha \cdot \frac{\bar{w}}{N_{i-1}}, \\ N_{i-1} & ,\text{if}\ \ Q = \alpha \cdot \frac{\bar{w}}{N_{i-1}}, \\ \max\{1, N_{i-1} - 1\} & ,\text{if}\ \ Q > \alpha \cdot \frac{\bar{w}}{N_{i-1}}, \end{cases}$$

Here Q is an estimate of the number of in-flight session that are queued in the network buffer

$$Q = (\overline{rtt} - rtt_{min})\frac{\bar{w}}{\overline{rtt}},$$

Change in code

A new agent for tcp fit added in tcp.h
A new file tcpMod.cc added

```
class TcpAgentMod : public virtual TcpAgent{
    public:
    TcpAgentMod();
    virtual void opencwnd();
    virtual void slowdown(int how);
    virtual void rtt_update(double tao);
    virtual void timeout(int tno);
    virtual void recv(Packet *pkt, Handler*);
    protected:
    int N;
    double alpha, windowSum, rttSum;
    int windowUpdateCount, rttUpdateCount;
    TracedInt t_rtt_min;
};
```

Corresponding TCL class

```
static class TcpAgentFitTclClass : public TclClass {
public:
    TcpAgentFitTclClass() : TclClass("Agent/TCP/FitW") {}
    TclObject* create(int, const char*const*) {
        return (new TcpAgentMod());
    }
} class_agent_fit;
```

In **opencwnd** method

```
case 1:
    /* This is the standard algorithm. */
    increment = (increase_num_ * N) / cwnd_;
    if ((last_cwnd_action_ == 0 ||
      last_cwnd_action_ == CWND_ACTION_TIMEOUT)
      && max_ssthresh_ > 0) {
        increment = limited_slow_start(cwnd_,
          max_ssthresh_, increment);
    }
    cwnd_ += increment;
    _ = cwnd_;
    // printf("cwnd : %lf\n", _);
    break;
```

Updating congestion window at each RTT

In **slowdown** method

```
if (first_decrease_ == 1 || slowstart || decrease_num_ == 0.5) {
    // printf("changed stuff \n");
    cwnd_ = (halfwin * 2) * (3. * N - 1) / (3. * N + 1);
} else cwnd_ = decreasewin;
```

Decreasing congestion window at packet loss

In **rtt_update** Method

```
if (t_rtt_min > t_rtt_)
    t_rtt_min = t_rtt_;

rttSum += t_rtt_.getValue();
rttUpdateCount += 1;
return;
```

Keeping track of minimum rtt value and average rtt

In **timeout** method

```
if (windowUpdateCount == 0 || rttUpdateCount == 0 || rttSum == 0)
    return;
double avgwnd = windowSum * 1. / windowUpdateCount;
double avgRtt = rttSum * 1. / rttUpdateCount;
double q = ((avgRtt - t_rtt_min) * avgwnd) / avgRtt;
double r = (alpha * avgwnd) / N;
if(q < r){
    N += 1;
}
else if(q > r){
    N = N > 2 ? N - 1 : 1;
}
```

Updating N

In recv method

```
windowSum += cwnd_;
windowUpdateCount++;
```

Keeping track of average window size

Default values of parameters

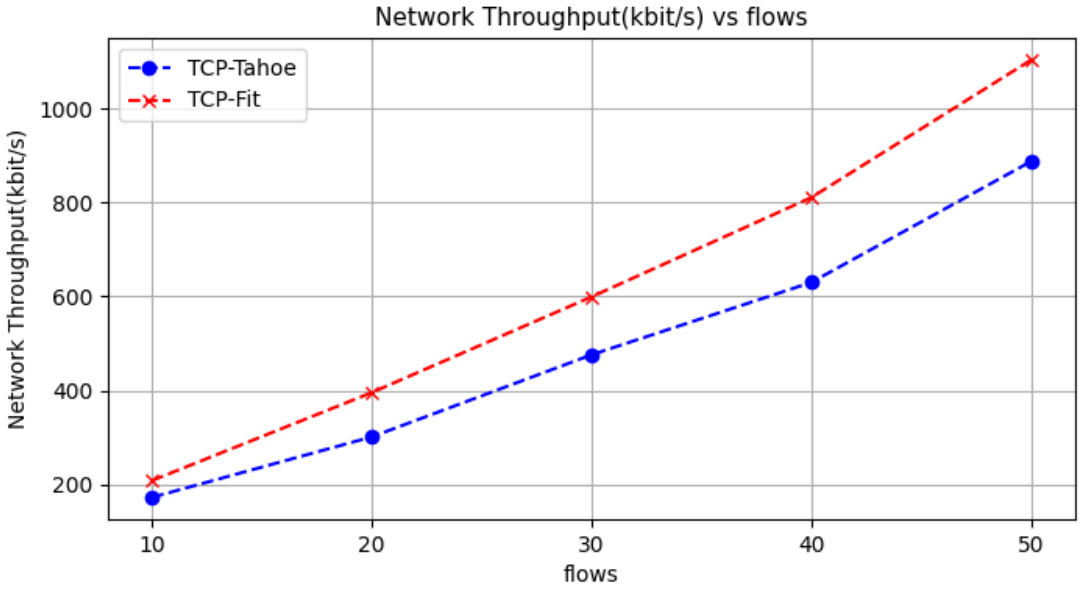
Number of nodes = 40
Number of flows = 40
Area = 500
Speed = 15 m/s
Routing protocol for wireless networks = DSR
Queue size = 50
Queue Type = CMUPriQueue
Traffic = Exponential
Packet Rate of traffic source = 300
Packet size = 200 bytes

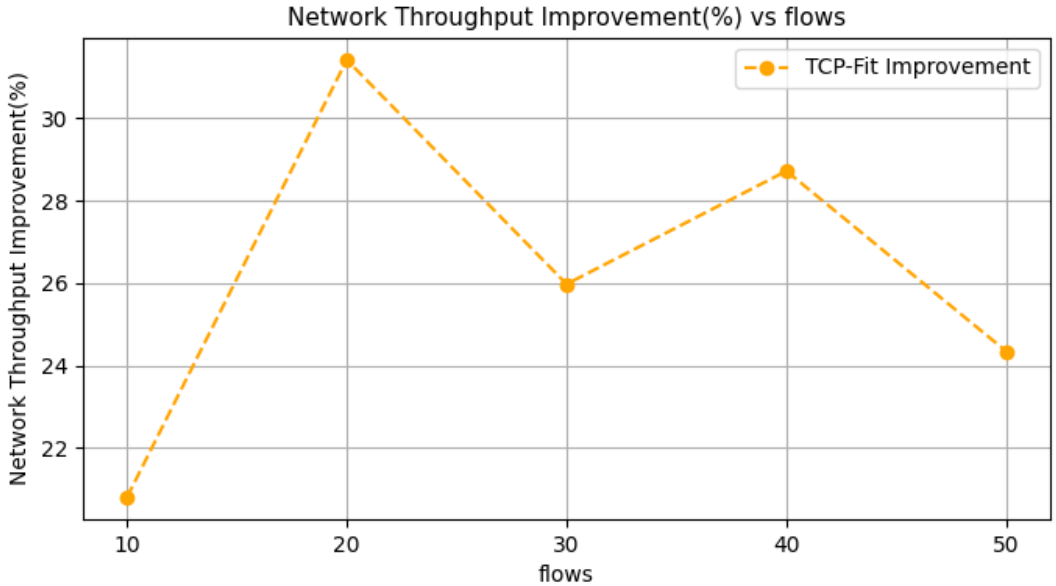
Results with graph

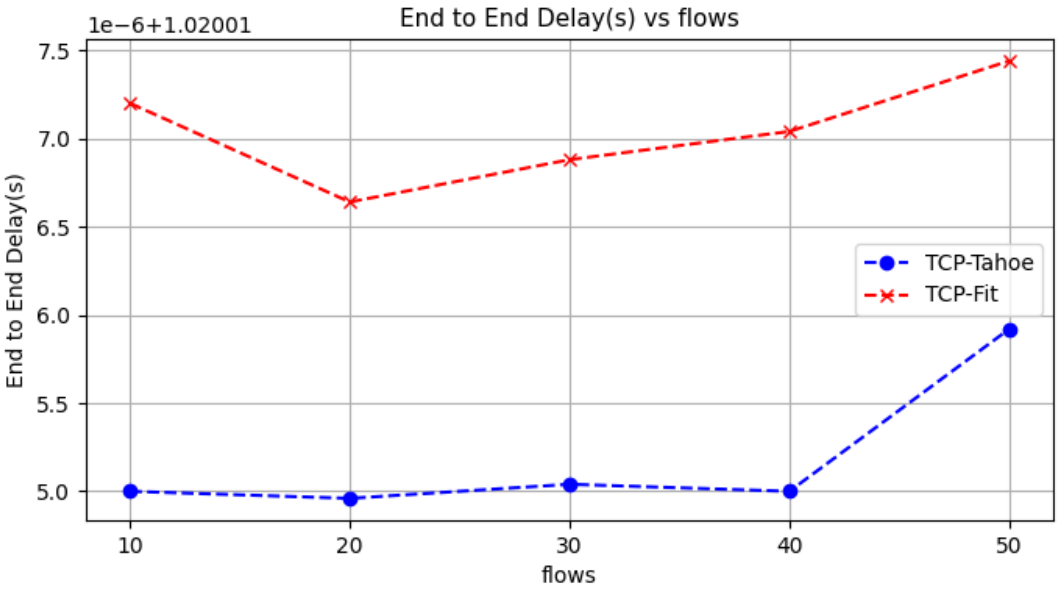
Wired Network

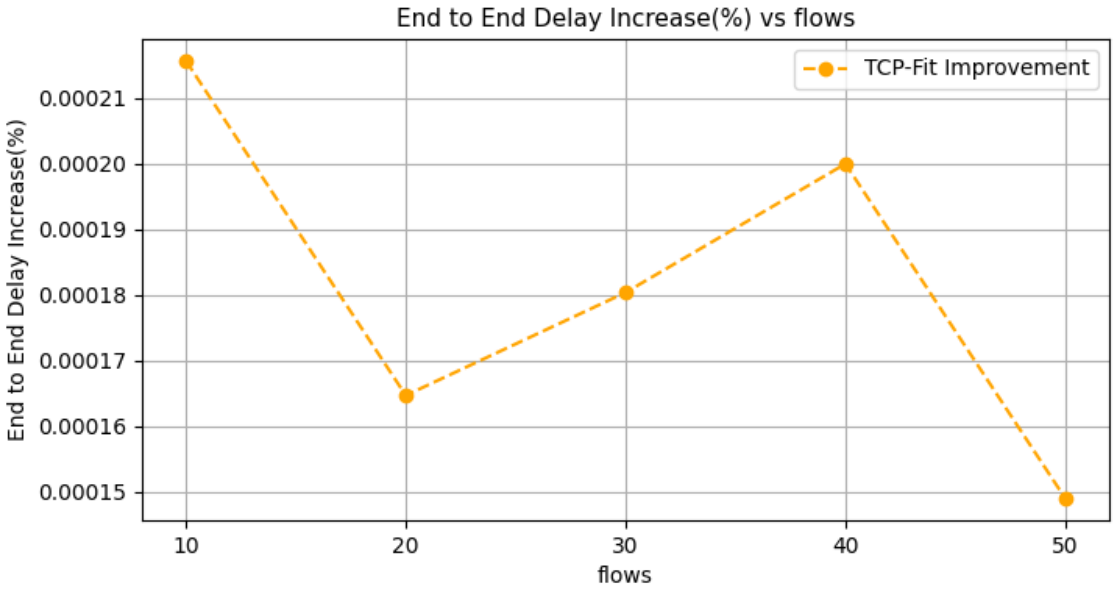
Wired network with a single high BDP link. The link introduces 5% random uniform packet drops.

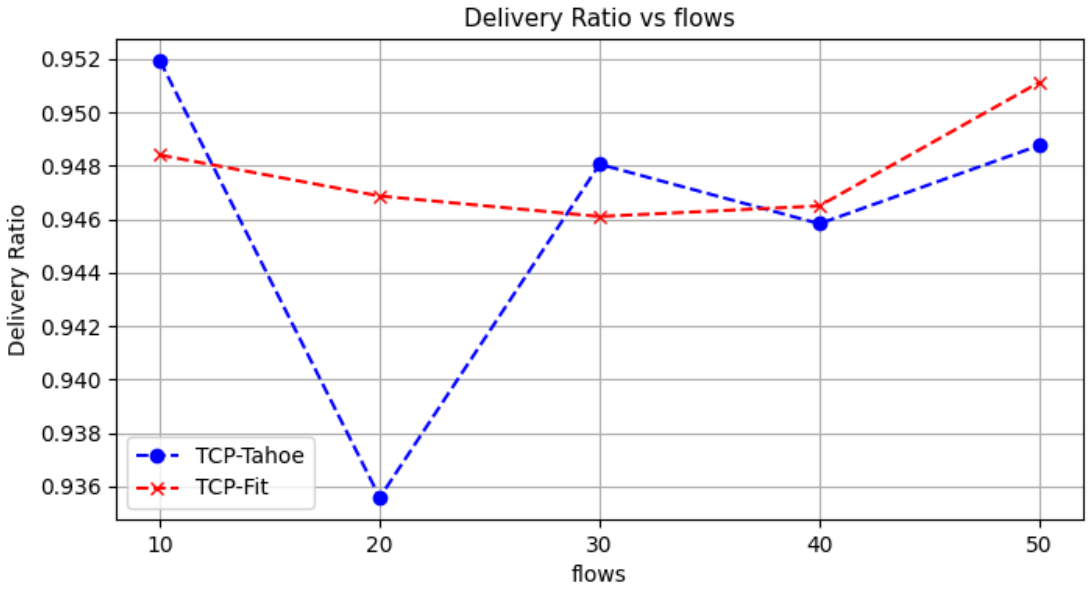
Varying flows

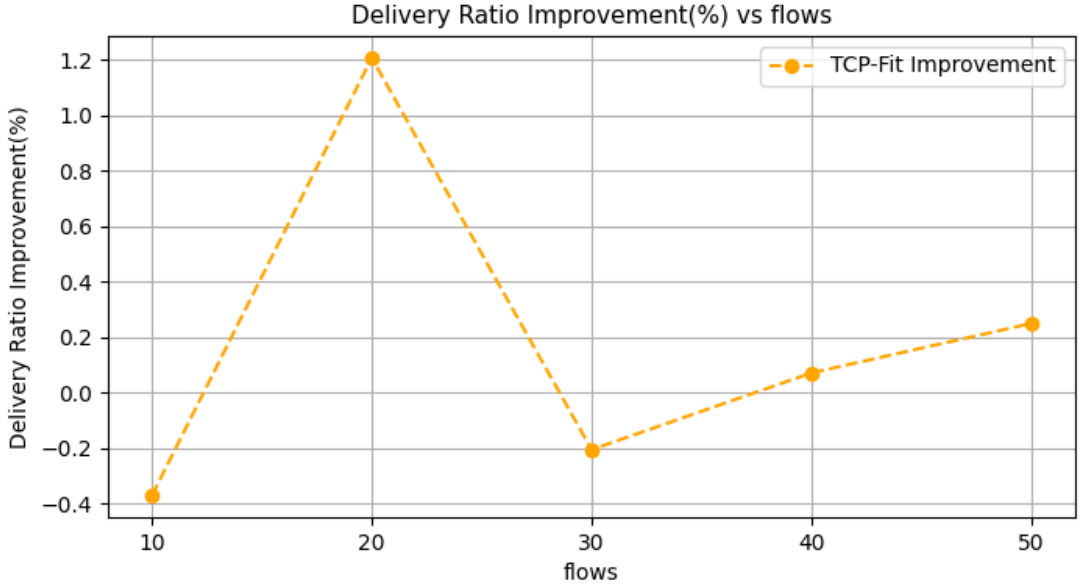


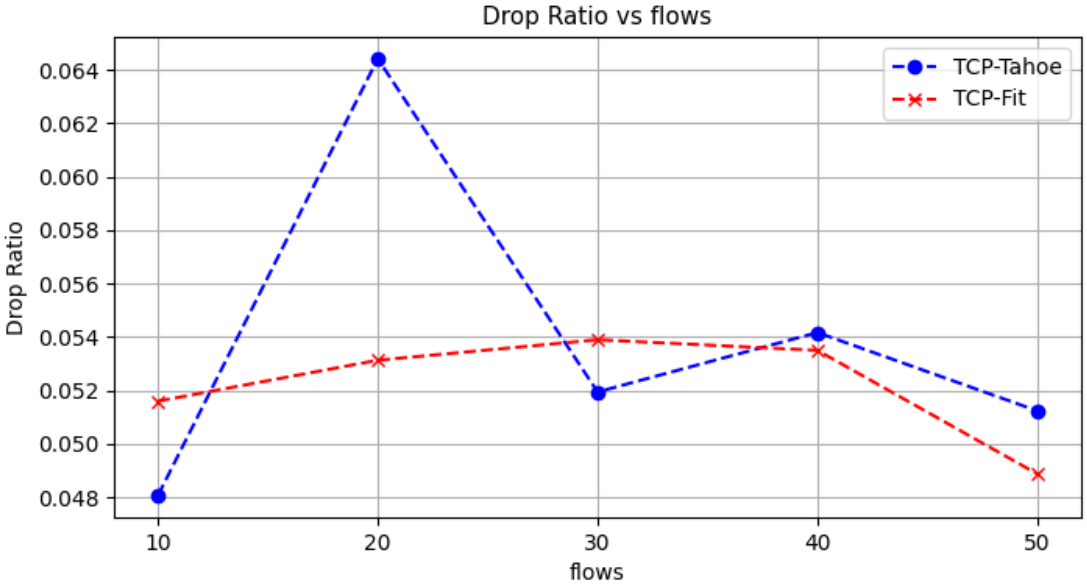


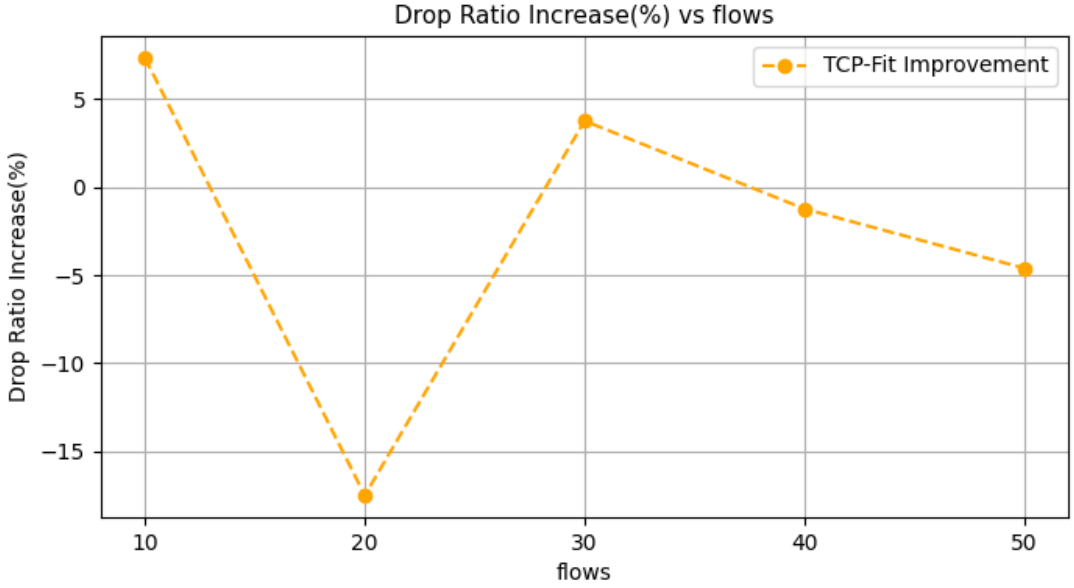

End to End Delay(s) vs flows
1e−6+1.02001
End to End Delay(s)
7.5
7.0
6.5
6.0
5.5
5.0
10
20
30
40
50
flows
TCP-Tahoe
TCP-Fit


End to End Delay Increase(%) vs flows
End to End Delay Increase(%)
0.00021
0.00020
0.00019
0.00018
0.00017
0.00016
0.00015
10
20
30
40
50
flows
TCP-Fit Improvement


Delivery Ratio vs flows
Delivery Ratio
0.952
0.950
0.948
0.946
0.944
0.942
0.940
0.938
0.936
TCP-Tahoe
TCP-Fit
10
20
30
40
50
flows


Delivery Ratio Improvement(%) vs flows
Delivery Ratio Improvement(%)
1.2
1.0
0.8
0.6
0.4
0.2
0.0
−0.2
−0.4
TCP-Fit Improvement
10
20
30
40
50
flows

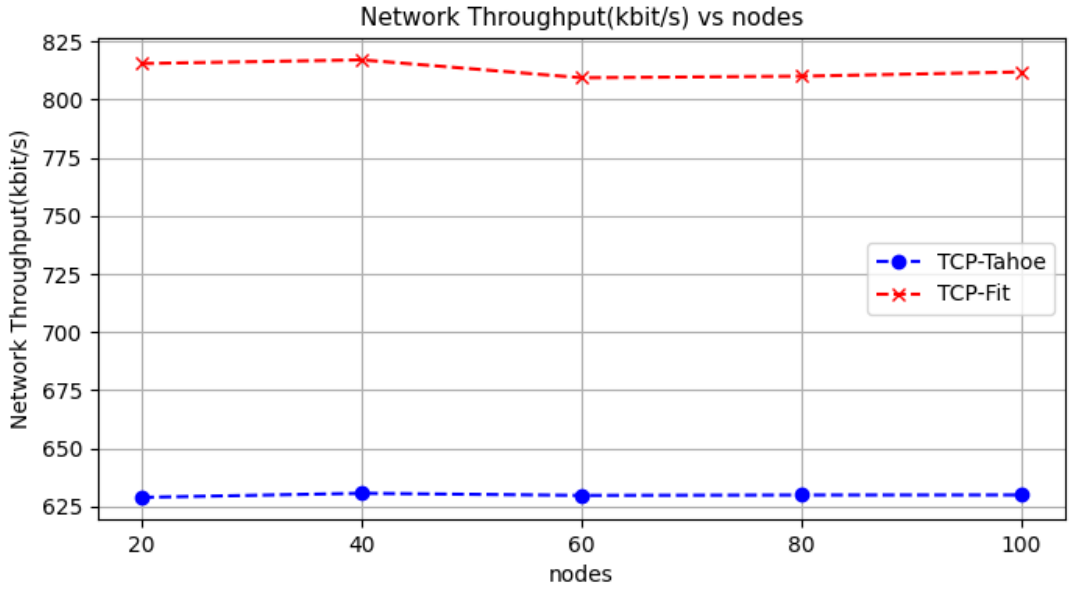

Drop Ratio vs flows
TCP-Tahoe
TCP-Fit
Drop Ratio
0.064
0.062
0.060
0.058
0.056
0.054
0.052
0.050
0.048
10
20
30
40
50
flows

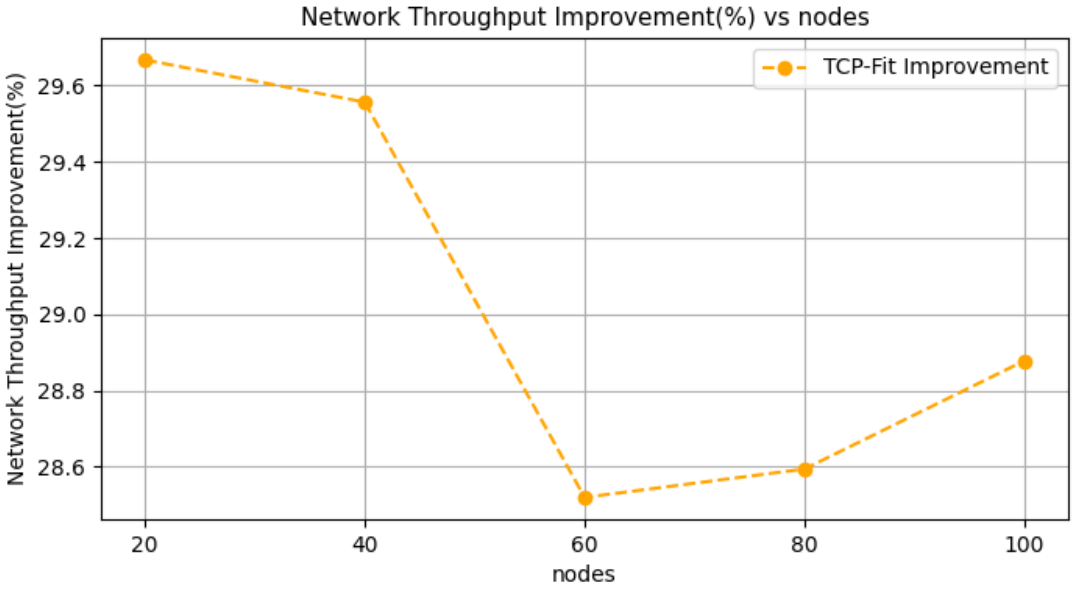

Drop Ratio Increase(%) vs flows
TCP-Fit Improvement
Drop Ratio Increase(%)
5
0
−5
−10
−15
10
20
30
40
50
flows

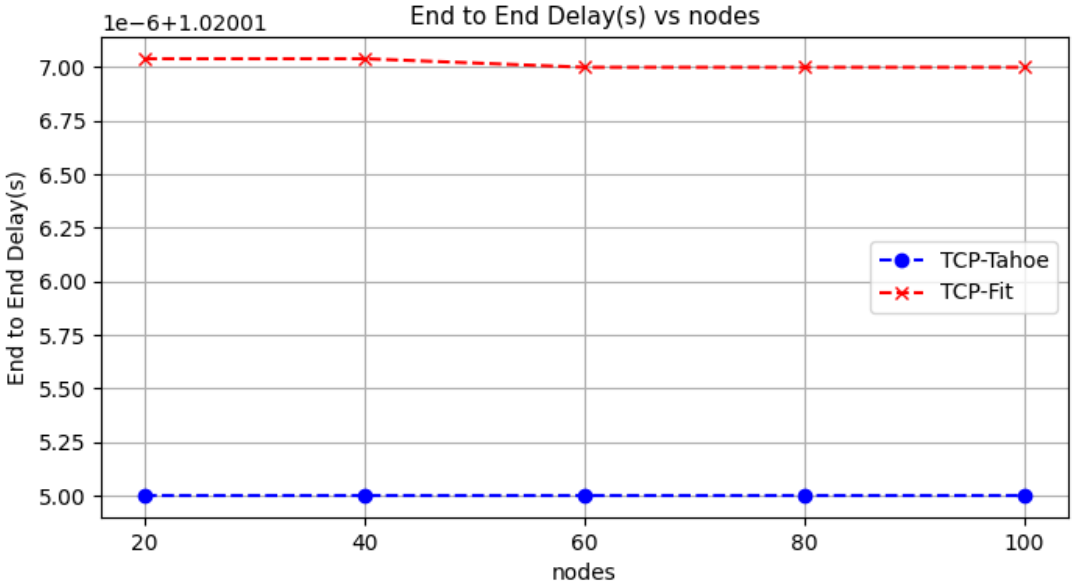
Observations

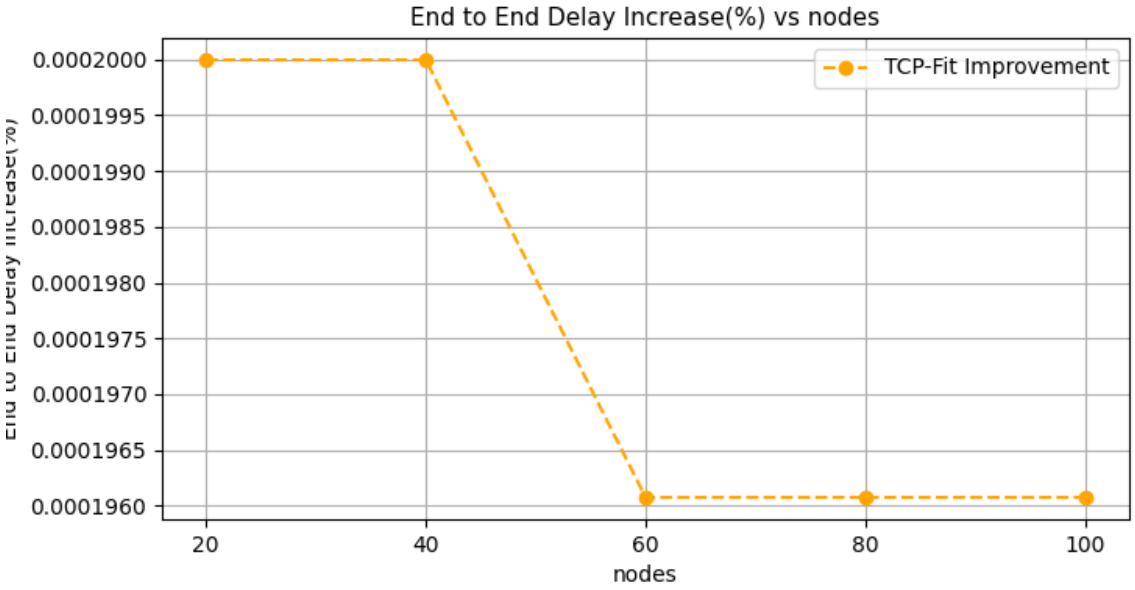
1. Throughput is seen to increase with increasing number of flows. This aligns with intuition. More flows means more packets sent at a unit time and so more throughput. Tcp fit shows significant improvement in network throughput compared to TCP Tahoe
2. End to end delay pretty much stays the same for varying flows, which is somewhat expected for wired networks. End to end delay is very slightly higher for TCP Fit.
3. 5% random packet drop was added to the bottleneck link to simulate random packet drops in a real life network. The delivery ratio and drop ratio are in accordance with that random packet drop.

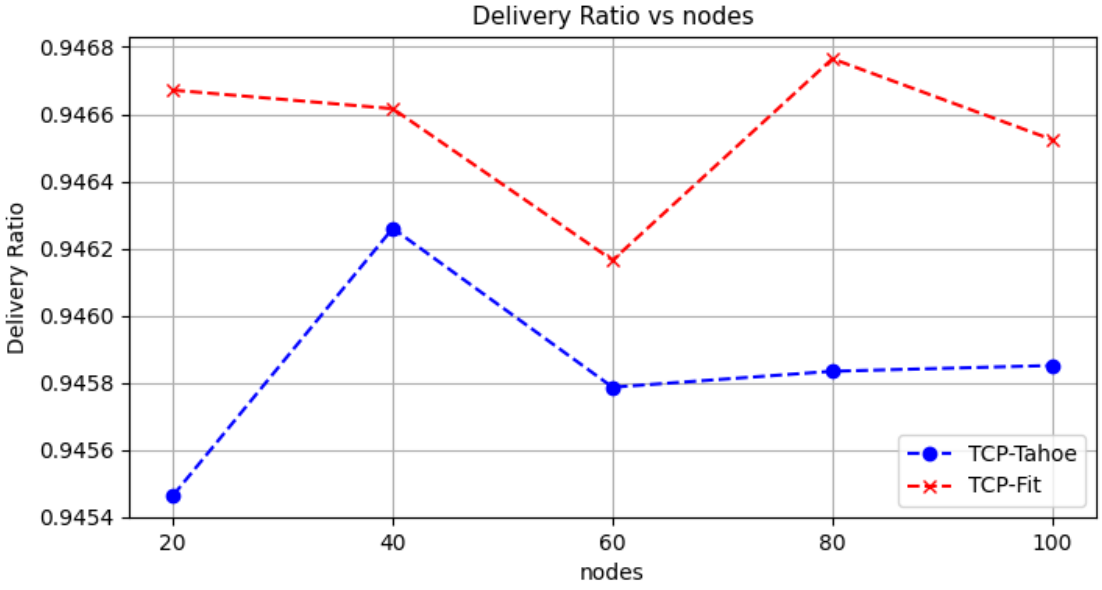
Varying nodes

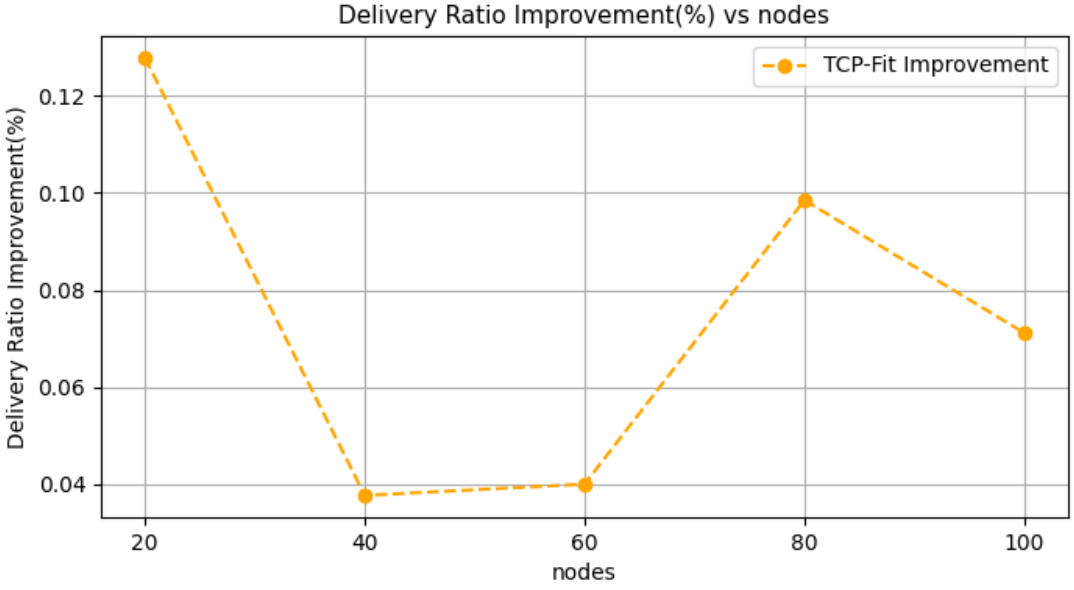


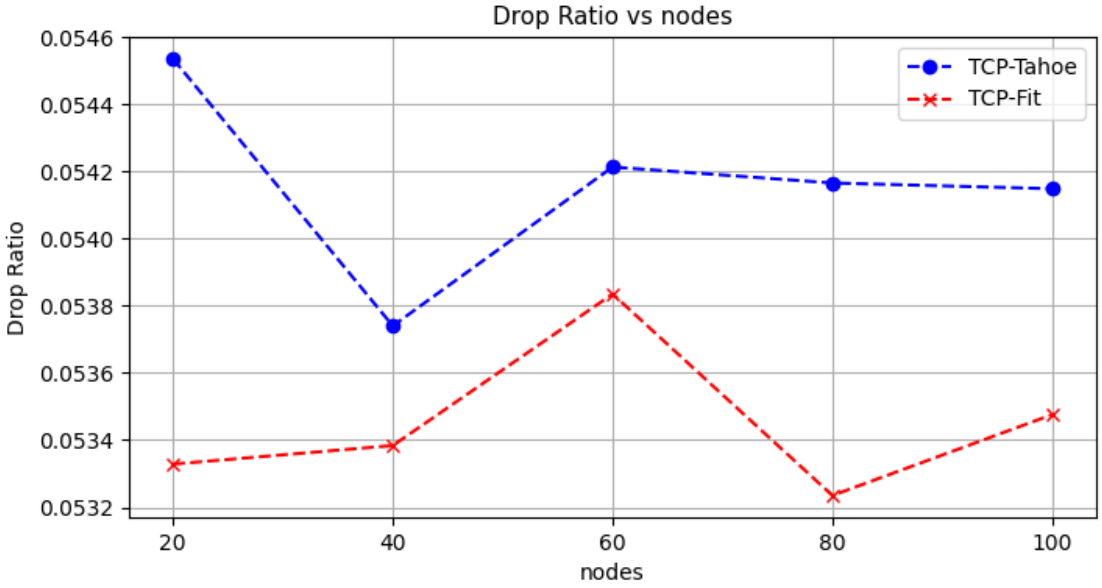


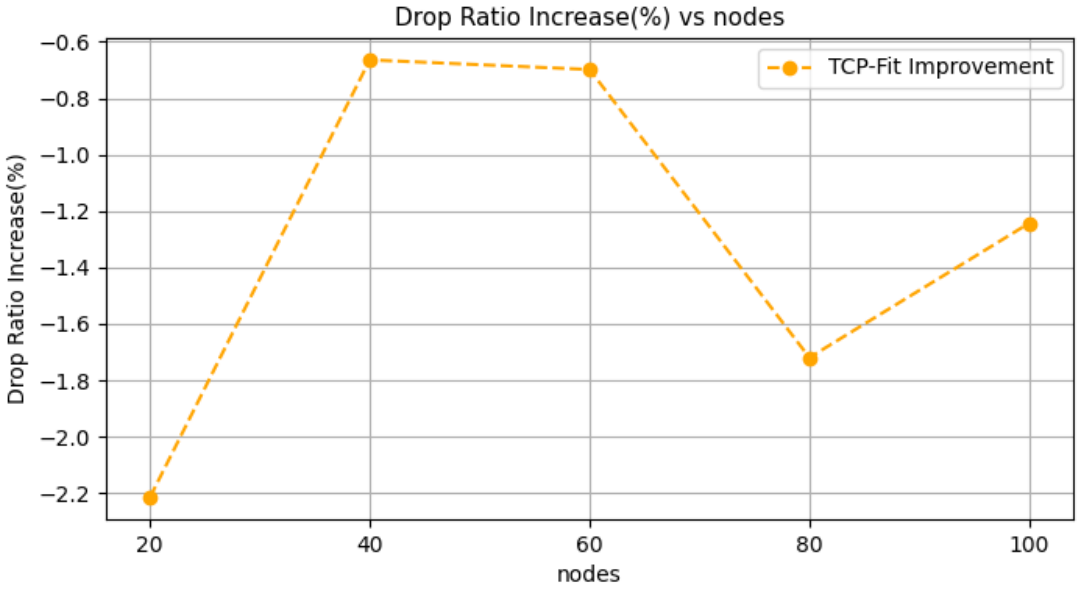

End to End Delay(s) vs nodes
1e−6+1.02001
7.00
6.75
6.50
6.25
6.00
5.75
5.50
5.25
5.00
End to End Delay(s)
20
40
60
80
100
nodes
TCP-Tahoe
TCP-Fit


End to End Delay Increase(%) vs nodes
0.0002000
0.0001995
0.0001990
0.0001985
0.0001980
0.0001975
0.0001970
0.0001965
0.0001960
20
40
60
80
100
nodes
TCP-Fit Improvement


Delivery Ratio vs nodes
Delivery Ratio
nodes
0.9468
0.9466
0.9464
0.9462
0.9460
0.9458
0.9456
0.9454
20
40
60
80
100
TCP-Tahoe
TCP-Fit


Delivery Ratio Improvement(%) vs nodes
Delivery Ratio Improvement(%)
nodes
0.12
0.10
0.08
0.06
0.04
20
40
60
80
100
TCP-Fit Improvement

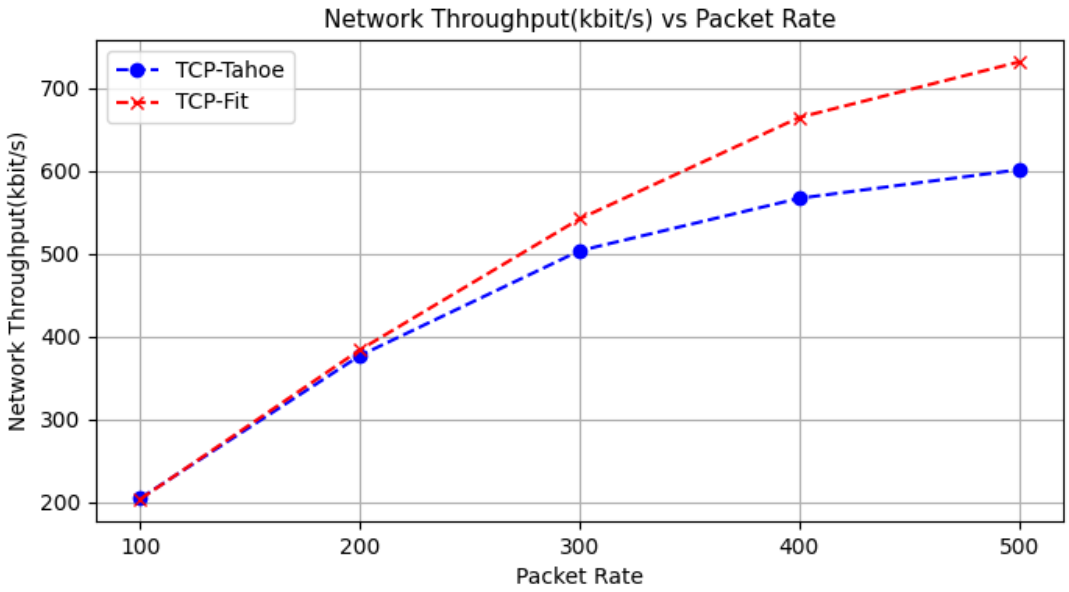


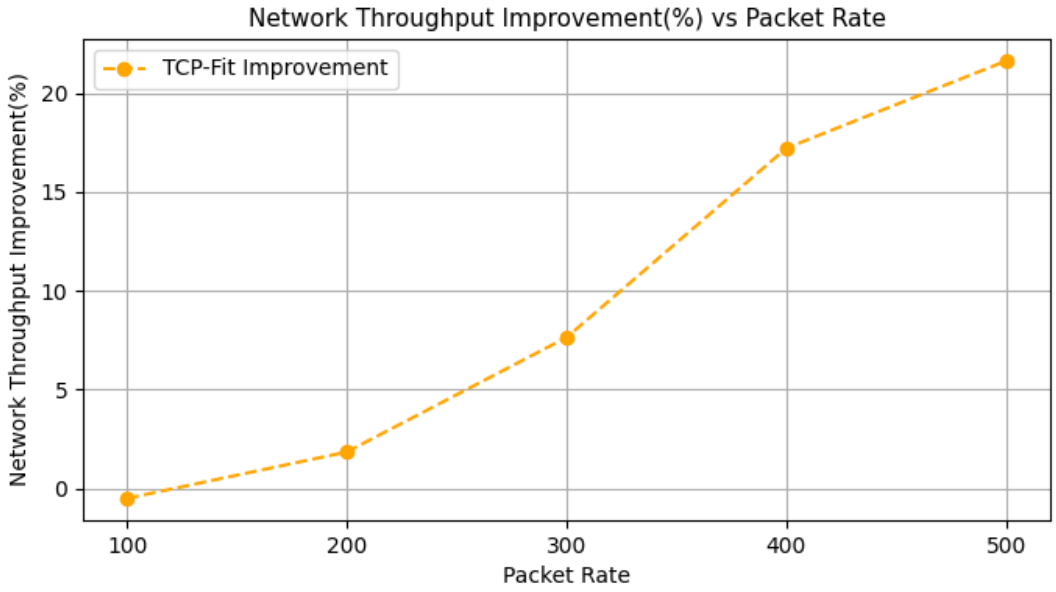


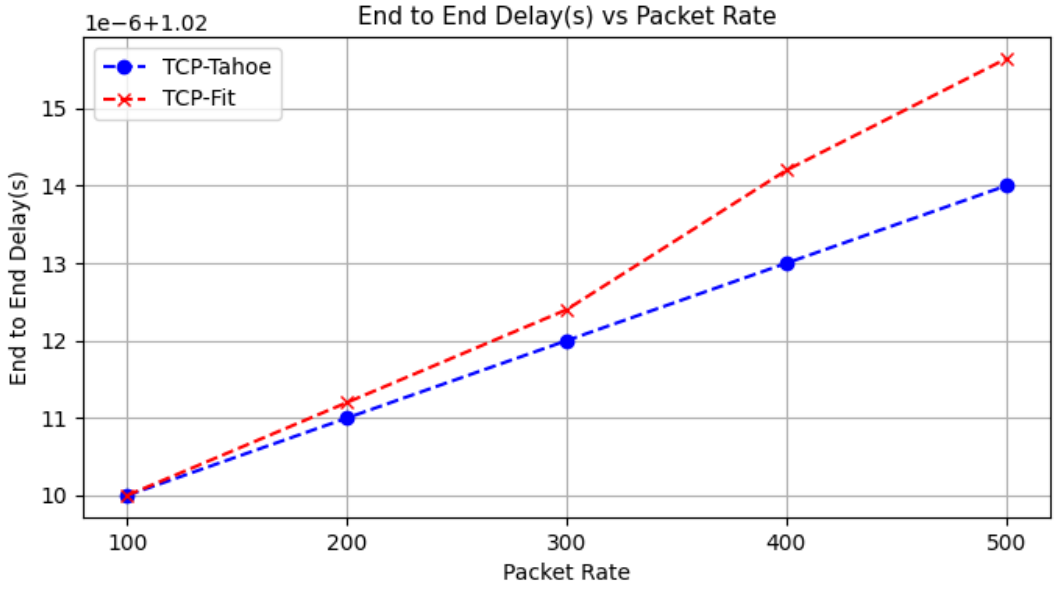
Observations

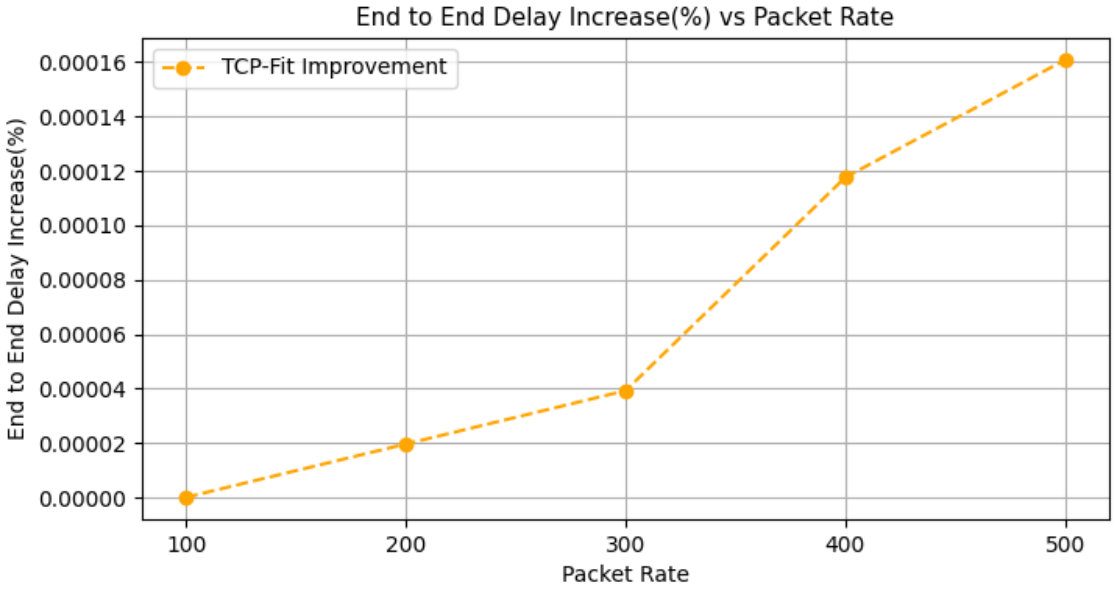
Varying number of nodes in wired networks with the considered topology does not really have any considerable effect on any of the metrics as they do not affect the path of the packets.

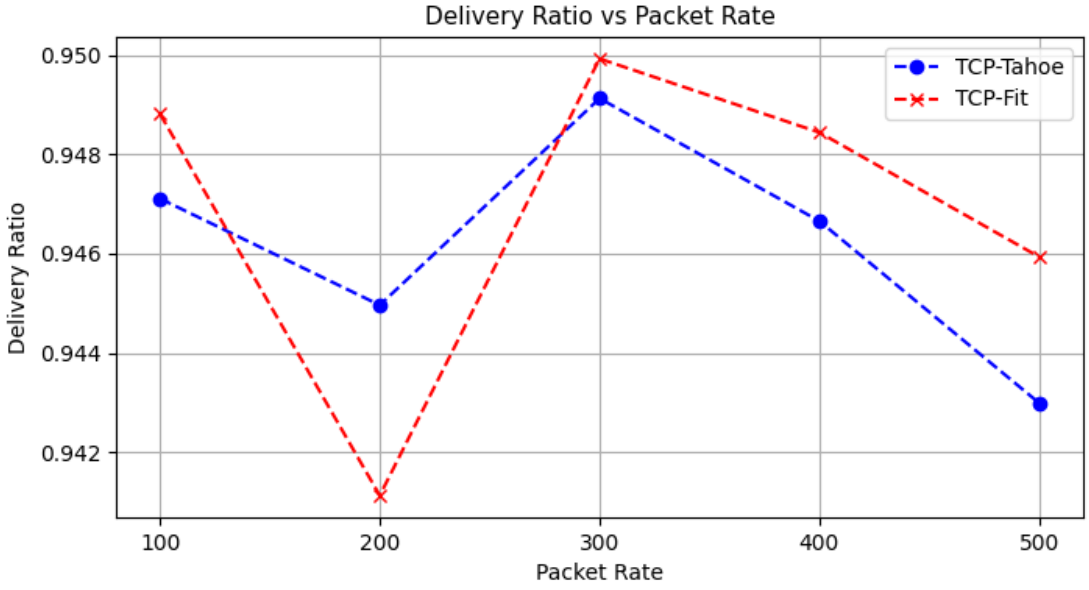
Varying packet rate

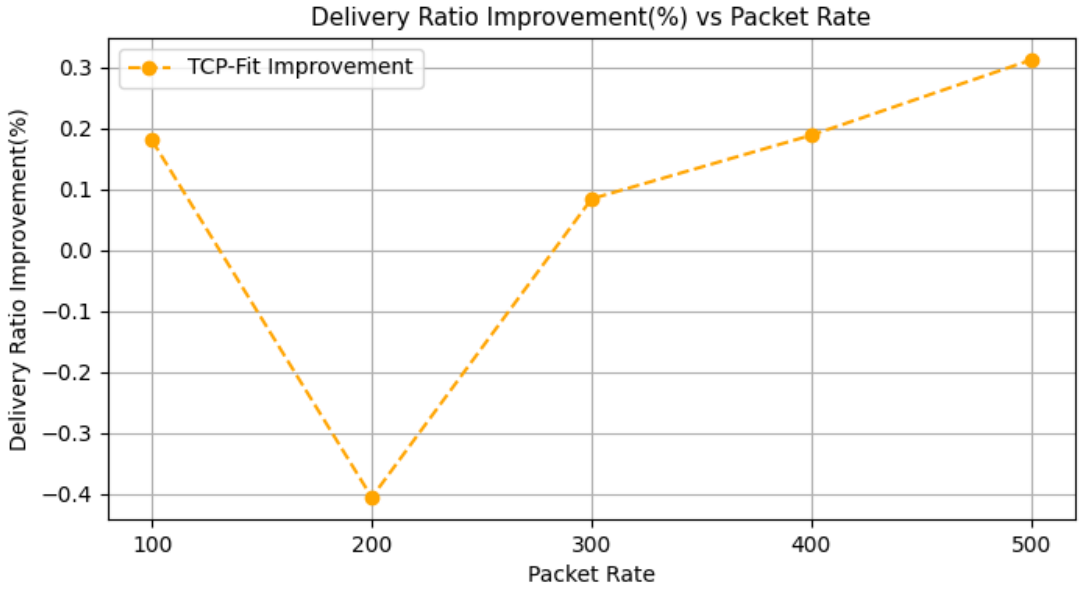


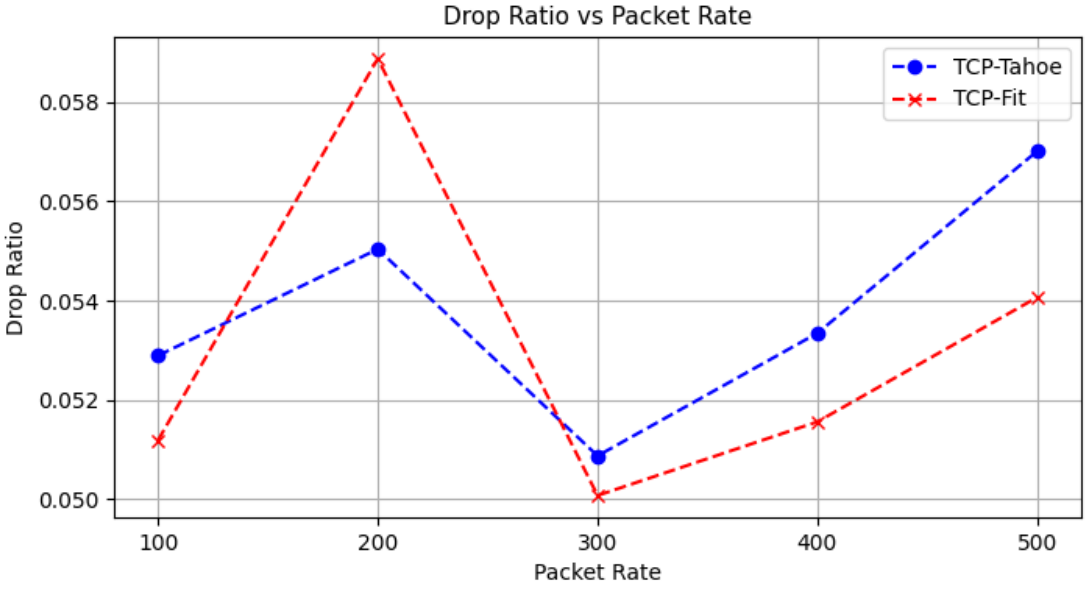


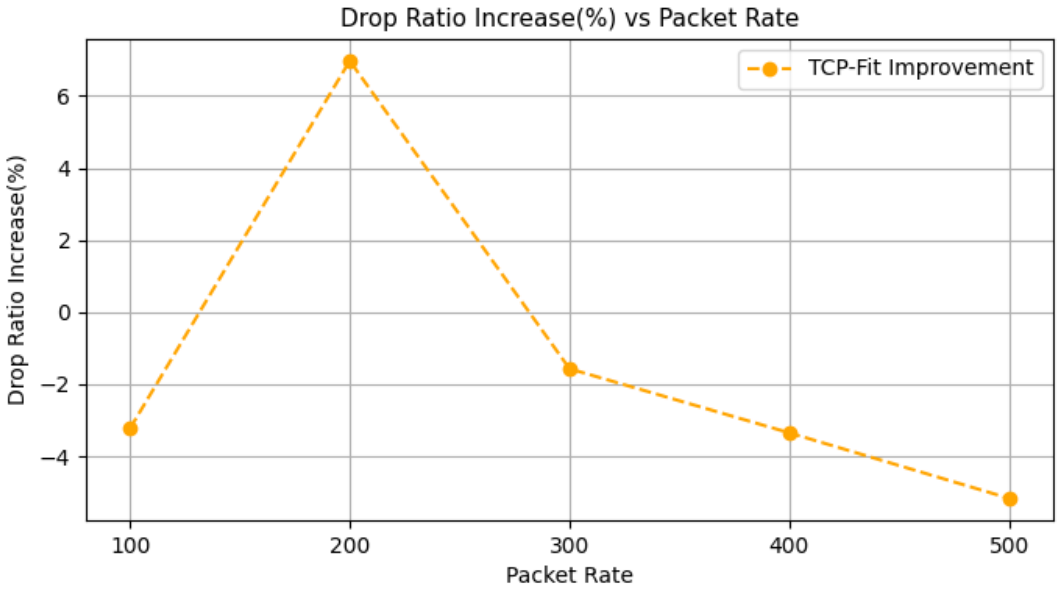

End to End Delay(s) vs Packet Rate
1e−6+1.02
TCP-Tahoe
TCP-Fit
End to End Delay(s)
Packet Rate
10
11
12
13
14
15
100
200
300
400
500


End to End Delay Increase(%) vs Packet Rate
TCP-Fit Improvement
End to End Delay Increase(%)
Packet Rate
0.00000
0.00002
0.00004
0.00006
0.00008
0.00010
0.00012
0.00014
0.00016
100
200
300
400
500


Delivery Ratio vs Packet Rate
TCP-Tahoe
TCP-Fit
Delivery Ratio
0.950
0.948
0.946
0.944
0.942
100
200
300
400
500
Packet Rate


Delivery Ratio Improvement(%) vs Packet Rate
TCP-Fit Improvement
Delivery Ratio Improvement(%)
0.3
0.2
0.1
0.0
−0.1
−0.2
−0.3
−0.4
100
200
300
400
500
Packet Rate


Drop Ratio vs Packet Rate
TCP-Tahoe
TCP-Fit
Drop Ratio
0.058
0.056
0.054
0.052
0.050
100
200
300
400
500
Packet Rate


Drop Ratio Increase(%) vs Packet Rate
TCP-Fit Improvement
Drop Ratio Increase(%)
6
4
2
0
−2
−4
100
200
300
400
500
Packet Rate

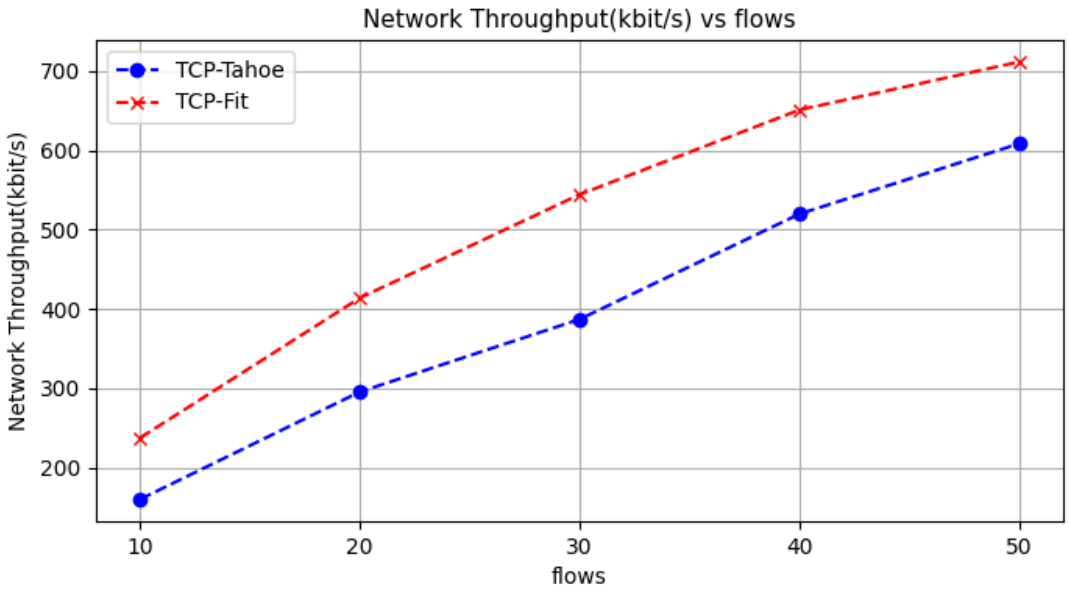
Observations

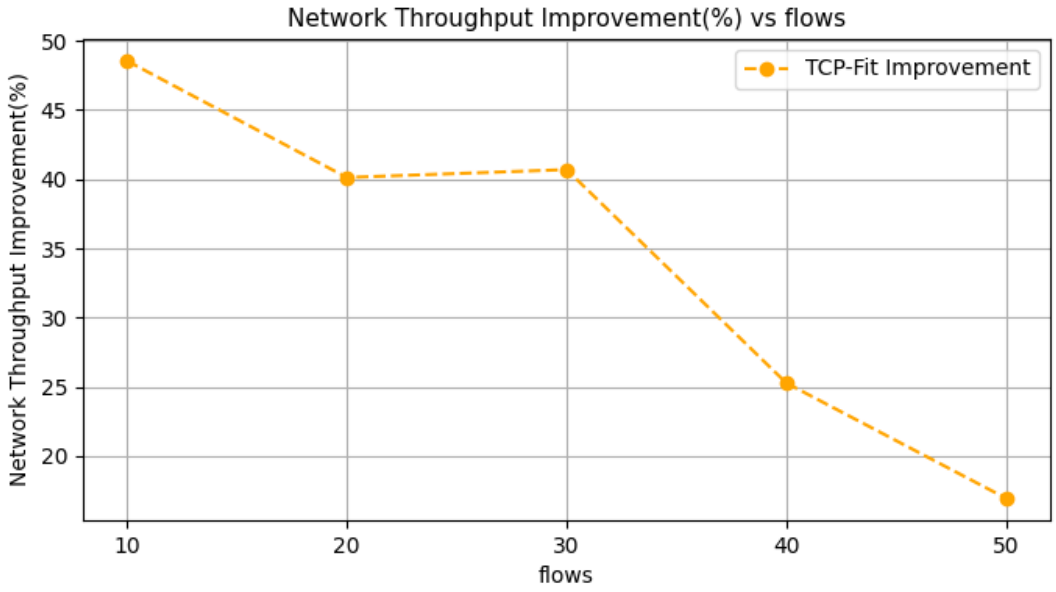
1. Throughput increases with increasing packet rate. The reasoning is similar to increasing the number of flows. As packet rate increases, TCP Fit seems to perform increasingly better compared to TCP Tahoe.
2. End to end delay is seen to increase very slightly with increasing packet rate. This is intuitive as higher packet rate should mean higher queuing delay. However, this delay is evidently very small as seen from the step size of the y axis. Again, end to end delay is very slightly higher for TCP-Fit

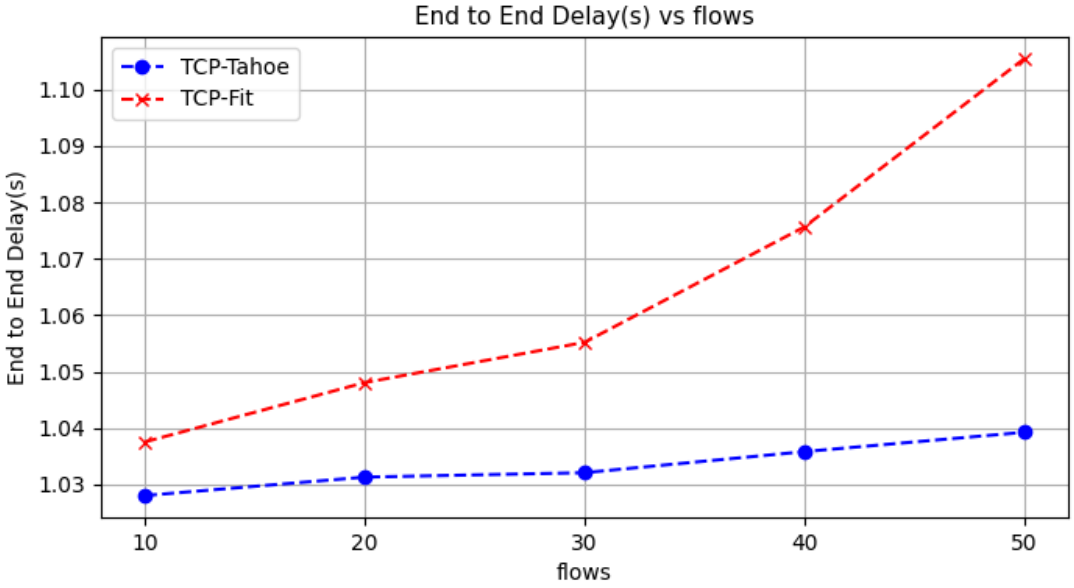
Wired-Wireless

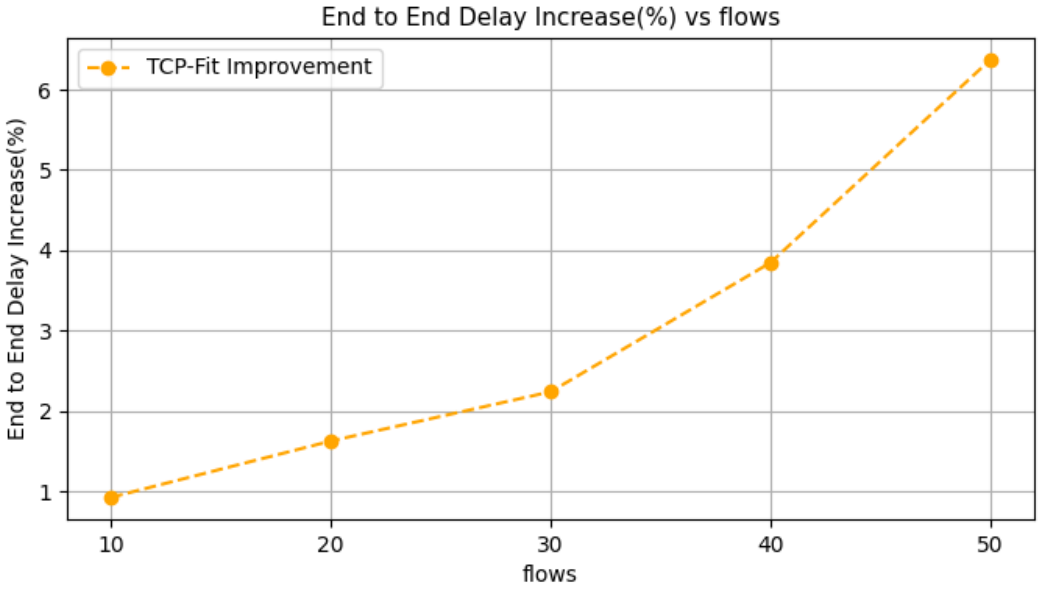
The bottleneck link was given 7% random packet drop and 1s delay

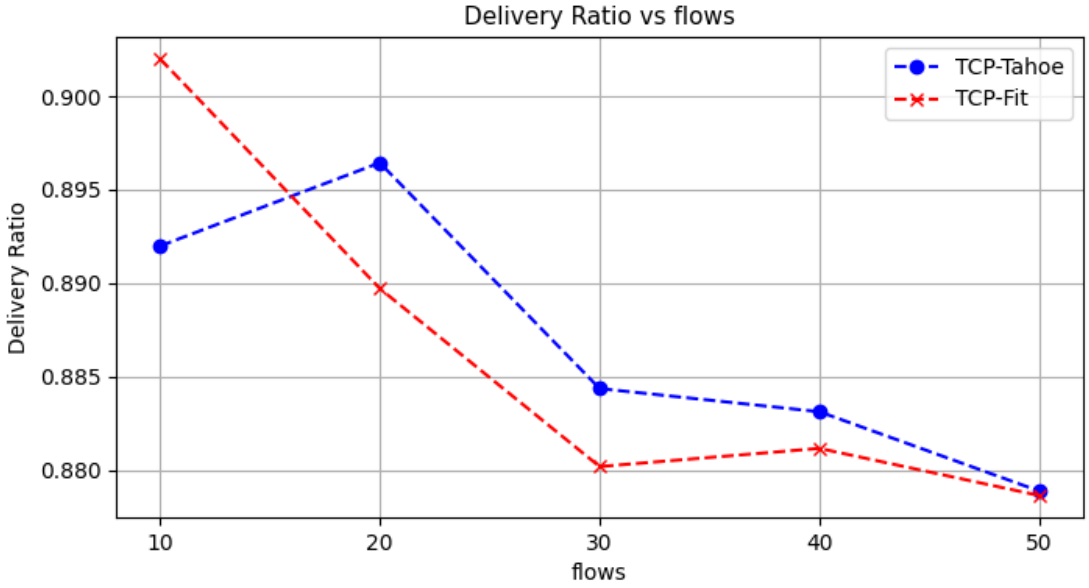
Varying flows



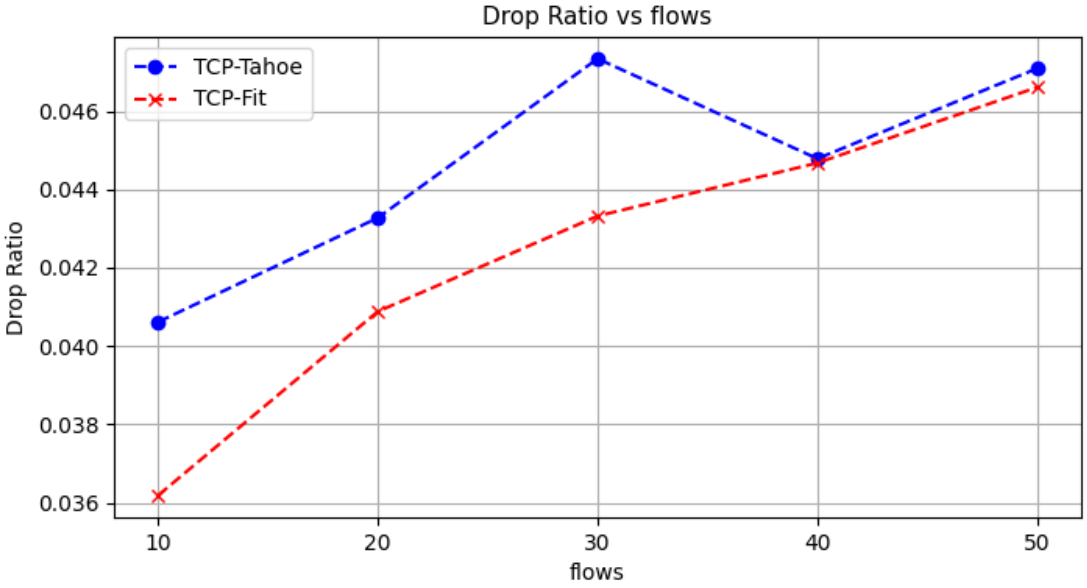


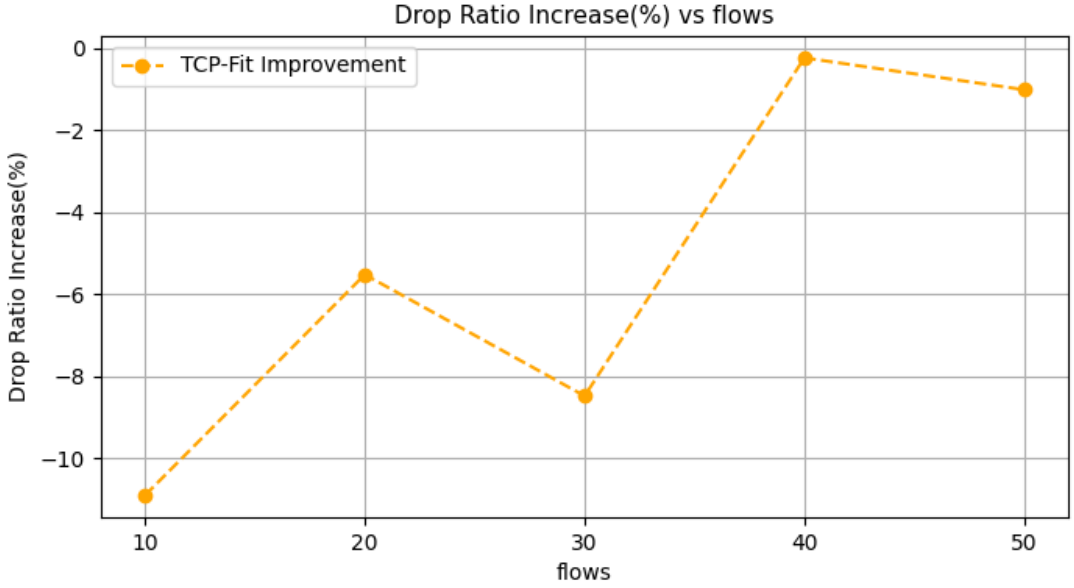

End to End Delay(s) vs flows
TCP-Tahoe
TCP-Fit
End to End Delay(s)
1.10
1.09
1.08
1.07
1.06
1.05
1.04
1.03
10
20
30
40
50
flows


End to End Delay Increase(%) vs flows
TCP-Fit Improvement
End to End Delay Increase(%)
6
5
4
3
2
1
10
20
30
40
50
flows


Delivery Ratio vs flows
TCP-Tahoe
TCP-Fit
Delivery Ratio
0.900
0.895
0.890
0.885
0.880
10
20
30
40
50
flows


Delivery Ratio Improvement(%) vs flows
TCP-Fit Improvement
Delivery Ratio Improvement(%)
1.00
0.75
0.50
0.25
0.00
−0.25
−0.50
−0.75
10
20
30
40
50
flows

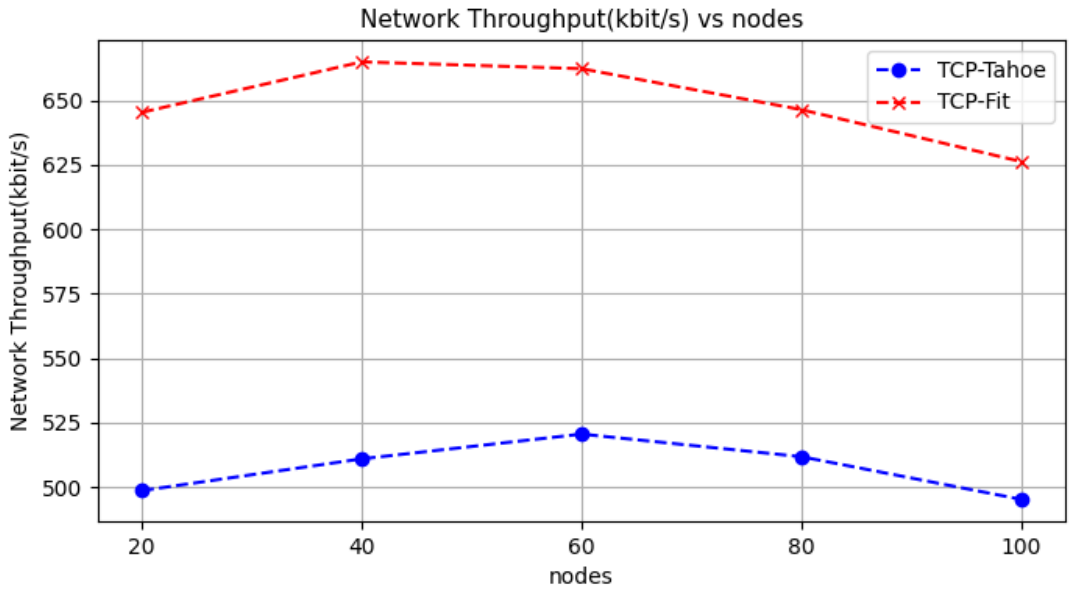

Drop Ratio vs flows
TCP-Tahoe
TCP-Fit
Drop Ratio
0.046
0.044
0.042
0.040
0.038
0.036
10
20
30
40
50
flows

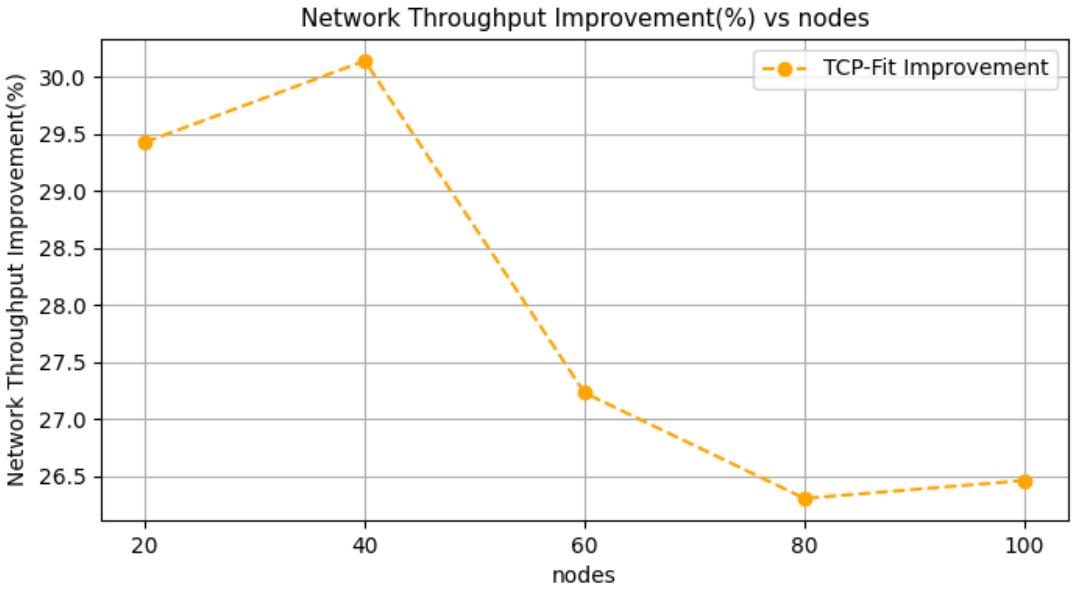

Drop Ratio Increase(%) vs flows
TCP-Fit Improvement
Drop Ratio Increase(%)
0
−2
−4
−6
−8
−10
10
20
30
40
50
flows

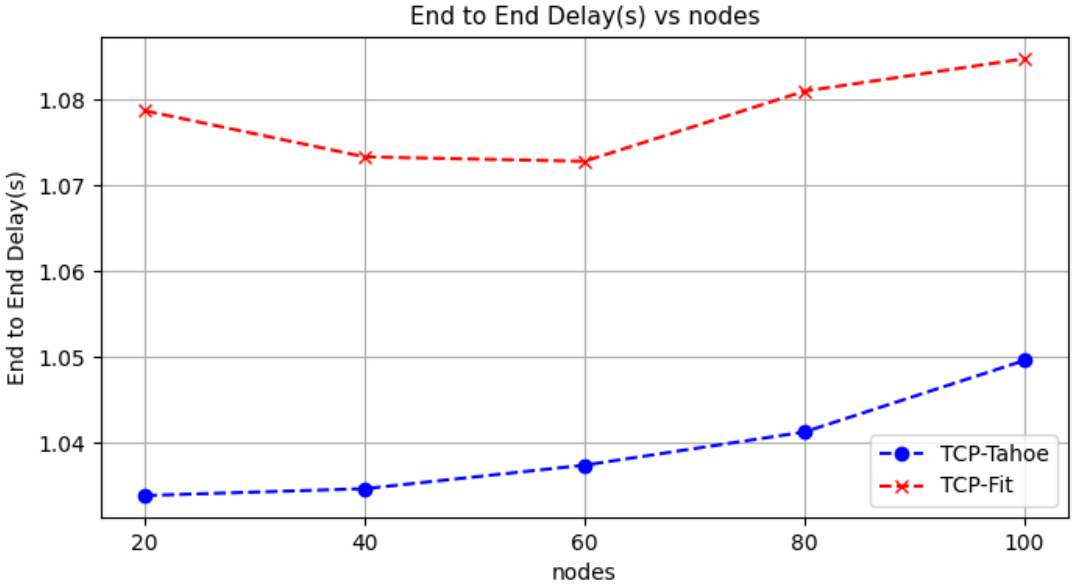
Observation

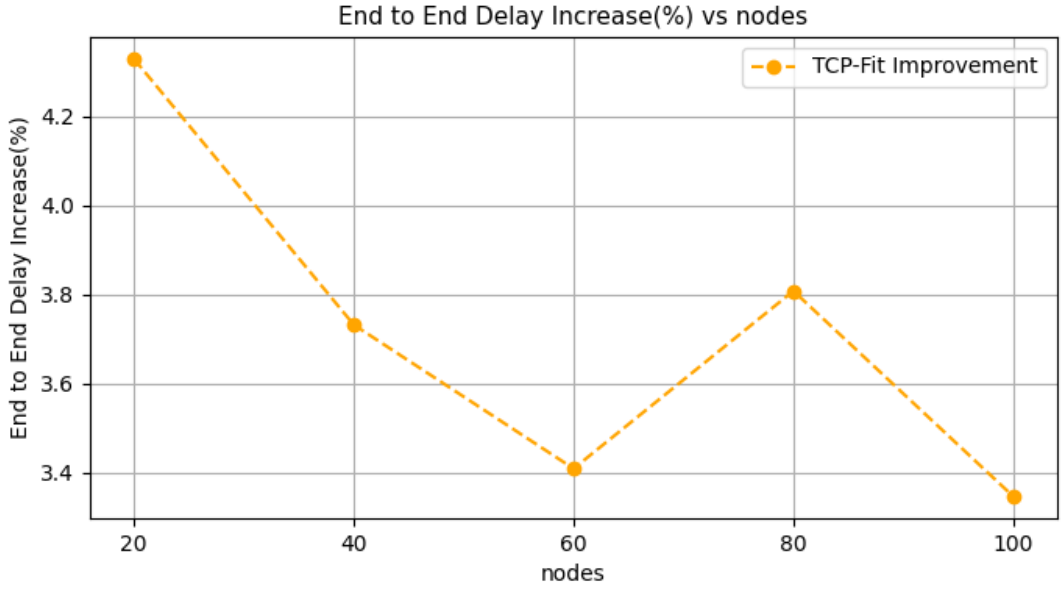
1. Throughput increases with the number of flows due to similar reasoning as wired networks. TCP-Fit gives significantly higher throughput than TCP-Tahoe
2. End to end delay is seen to increase with increasing flows. Notably end to end delay in TCP-Fit is higher and the difference seem to increase with more flows
3. It is seen that the delivery ratio drops and so the drop ratio increases slightly with increasing number of flows. This is most likely because of the packet drops due to limited buffer size. TCP-Fit does not seem to have any advantage or disadvantage against TCP-Tahoe in this regard.

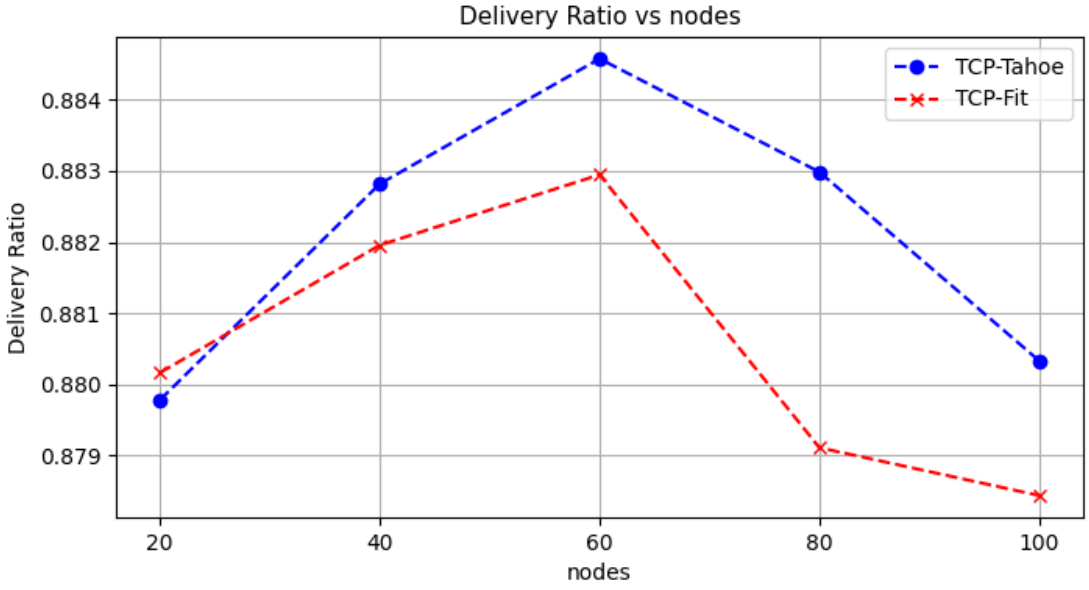
Varying nodes



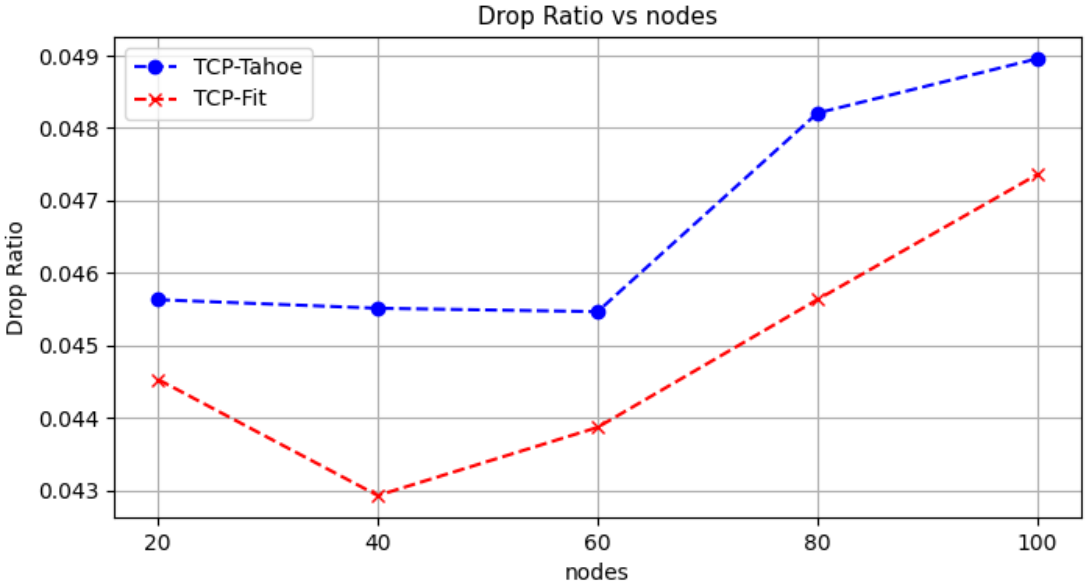


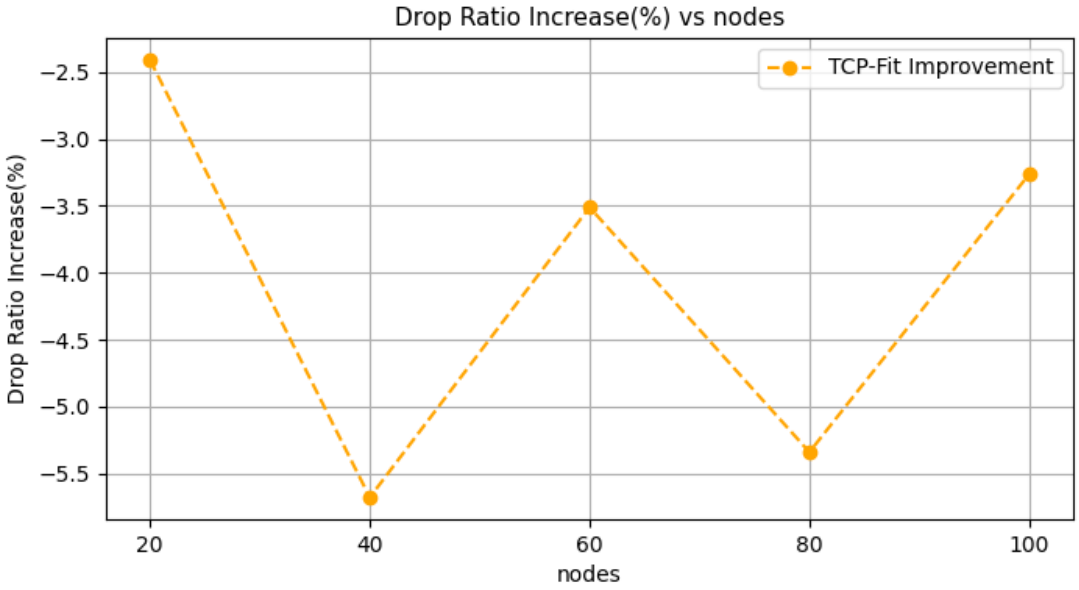

End to End Delay(s) vs nodes
End to End Delay(s)
1.08
1.07
1.06
1.05
1.04
20
40
60
80
100
nodes
TCP-Tahoe
TCP-Fit


End to End Delay Increase(%) vs nodes
End to End Delay Increase(%)
4.2
4.0
3.8
3.6
3.4
20
40
60
80
100
nodes
TCP-Fit Improvement


Delivery Ratio vs nodes
TCP-Tahoe
TCP-Fit
Delivery Ratio
0.884
0.883
0.882
0.881
0.880
0.879
20
40
60
80
100
nodes


Delivery Ratio Improvement(%) vs nodes
TCP-Fit Improvement
Delivery Ratio Improvement(%)
0.0
−0.1
−0.2
−0.3
−0.4
20
40
60
80
100
nodes

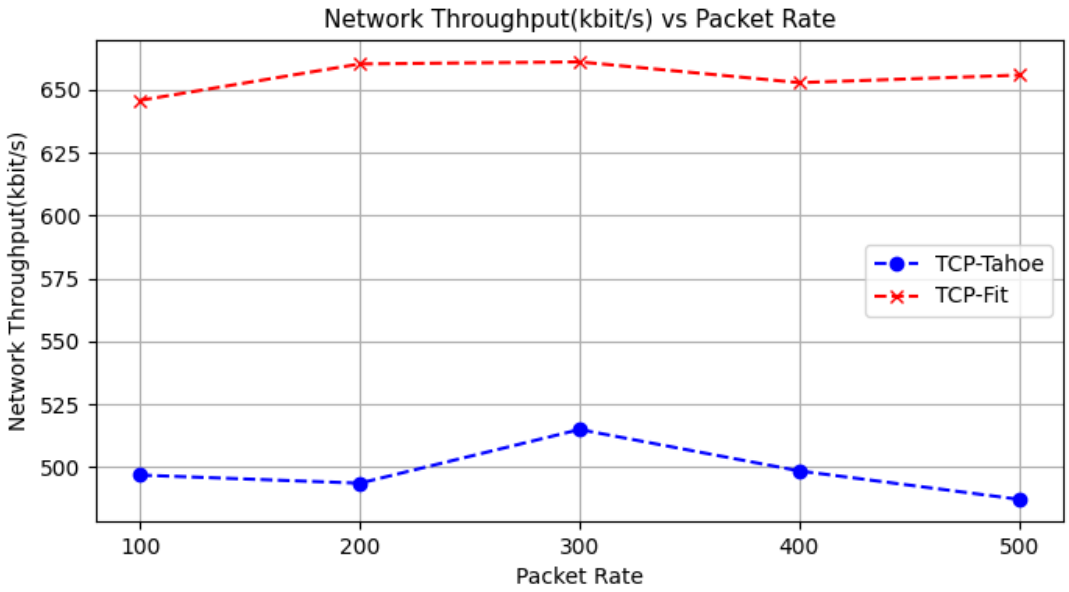

Drop Ratio vs nodes
TCP-Tahoe
TCP-Fit
Drop Ratio
0.049
0.048
0.047
0.046
0.045
0.044
0.043
20
40
60
80
100
nodes

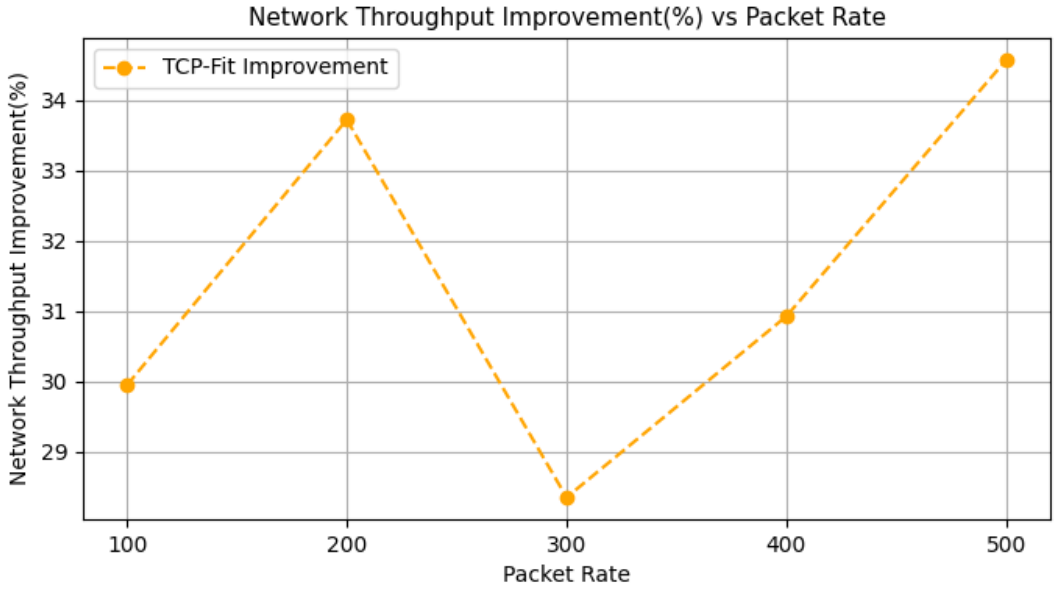

Drop Ratio Increase(%) vs nodes
TCP-Fit Improvement
Drop Ratio Increase(%)
−2.5
−3.0
−3.5
−4.0
−4.5
−5.0
−5.5
20
40
60
80
100
nodes

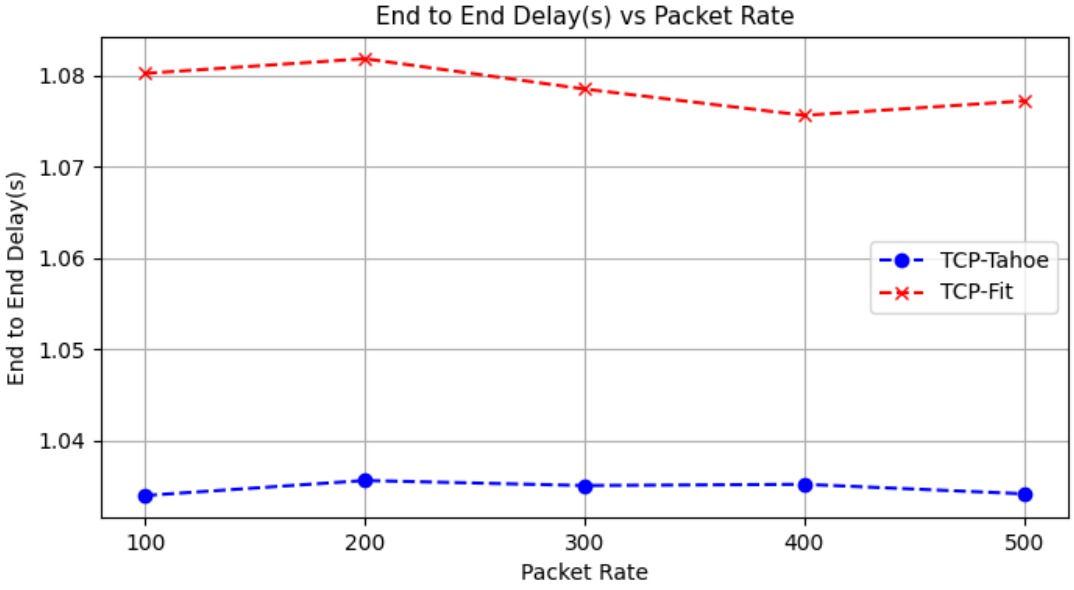
Observations

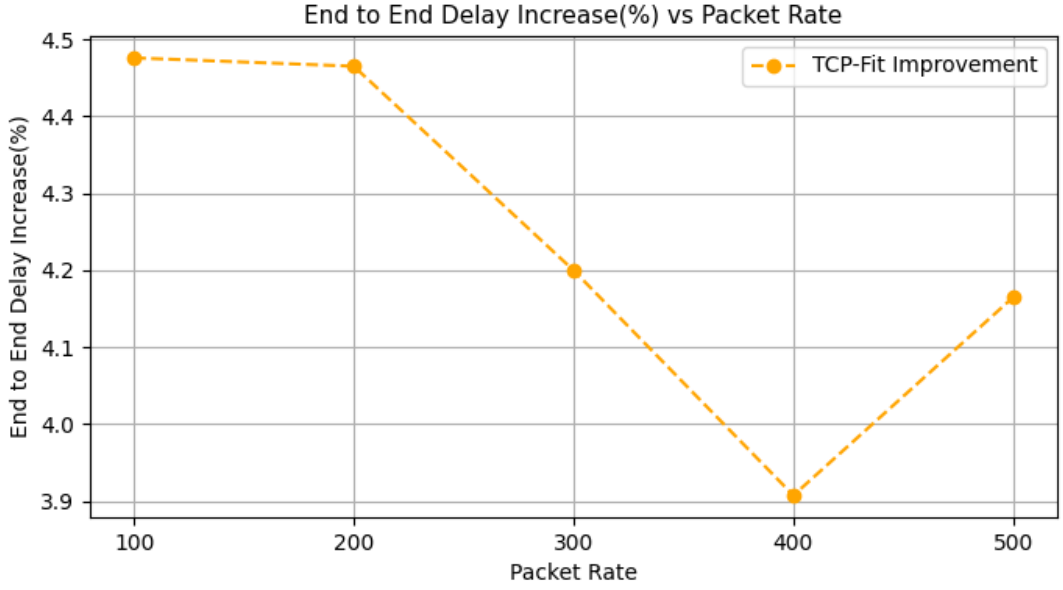
1. Throughput seems to increase with increasing number of nodes up to some point, then decrease. As the number of flows remains the same, increasing node count at first most likely distributes the flows. As a result throughput increases. Then at some point ,increase of interference takes over, and causes the throughput to decrease.
2. End to end delay is slightly affected. With the current topology, end to end delay is not much affected since most packets directly goes to the base station, only some packets take multiple hops before reaching base station or before reaching destination
3. No notable trend in drop ratio or delivery ratio is noticed
4. TCP-Fit continues to give large improvement in network throughput with small changes in other metrics

Varying packet rate

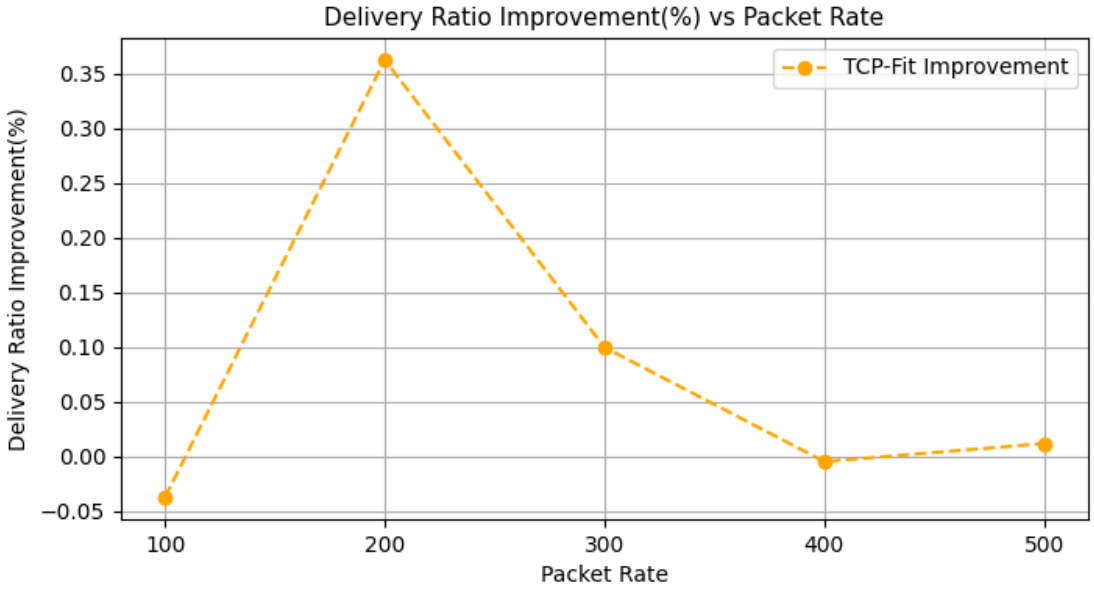


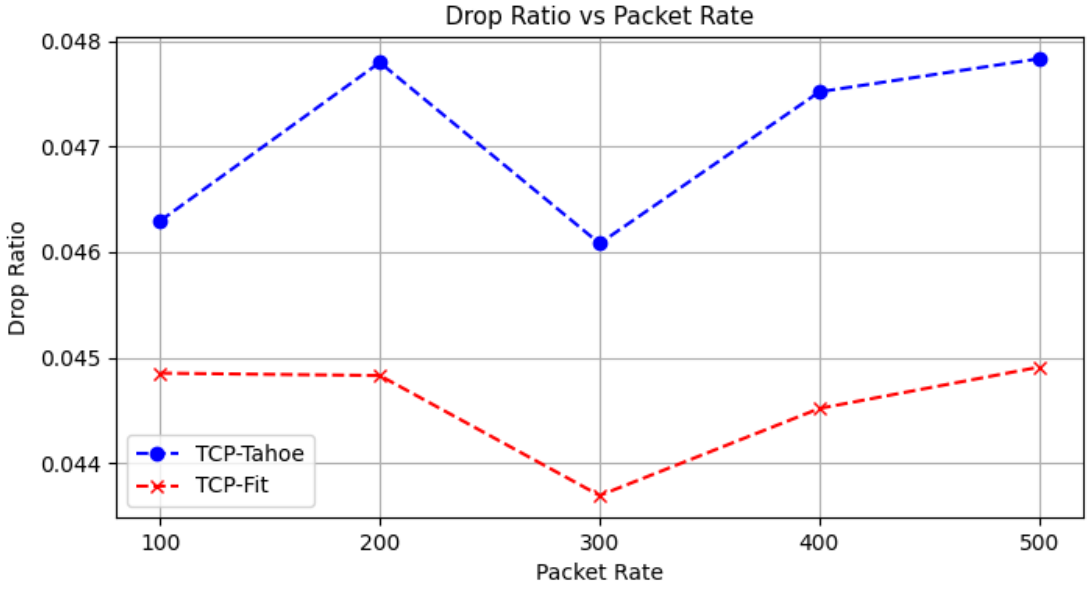


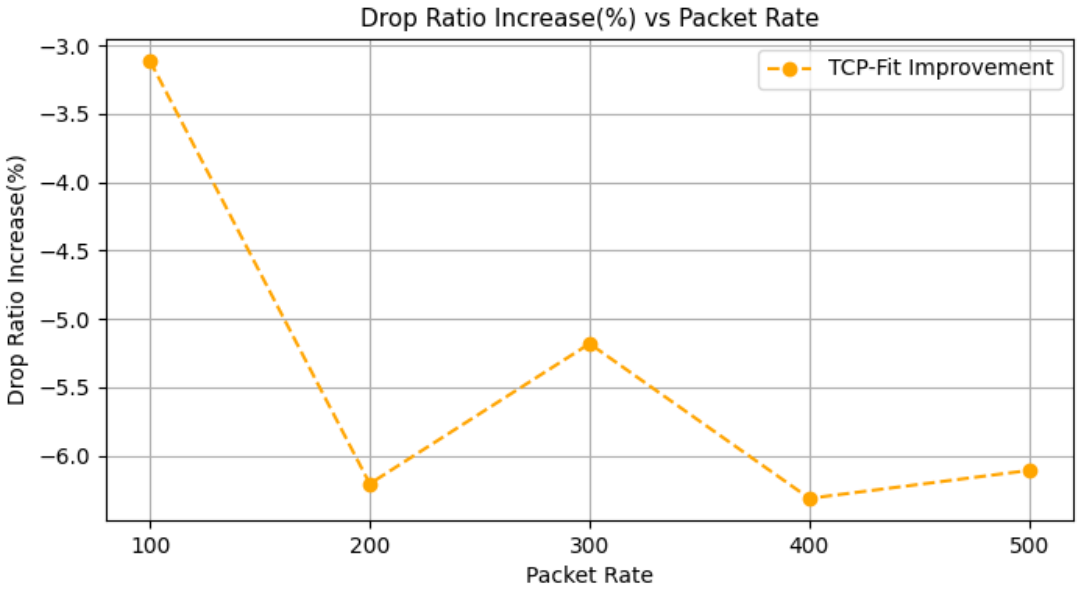

End to End Delay(s) vs Packet Rate
End to End Delay(s)
1.08
1.07
1.06
1.05
1.04
100
200
300
400
500
Packet Rate
TCP-Tahoe
TCP-Fit


End to End Delay Increase(%) vs Packet Rate
End to End Delay Increase(%)
4.5
4.4
4.3
4.2
4.1
4.0
3.9
100
200
300
400
500
Packet Rate
TCP-Fit Improvement


Delivery Ratio vs Packet Rate
TCP-Tahoe
TCP-Fit
Delivery Ratio
0.8825
0.8820
0.8815
0.8810
0.8805
0.8800
0.8795
0.8790
100
200
300
400
500
Packet Rate


Delivery Ratio Improvement(%) vs Packet Rate
TCP-Fit Improvement
Delivery Ratio Improvement(%)
0.35
0.30
0.25
0.20
0.15
0.10
0.05
0.00
−0.05
100
200
300
400
500
Packet Rate

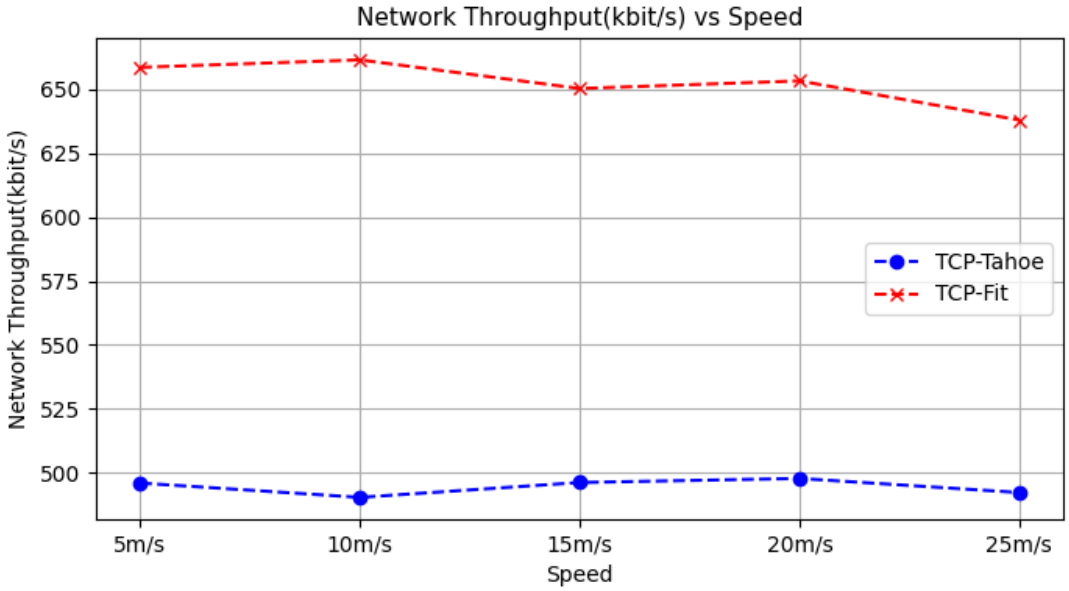

Drop Ratio vs Packet Rate
0.048
0.047
0.046
0.045
0.044
Drop Ratio
TCP-Tahoe
TCP-Fit
100
200
300
400
500
Packet Rate

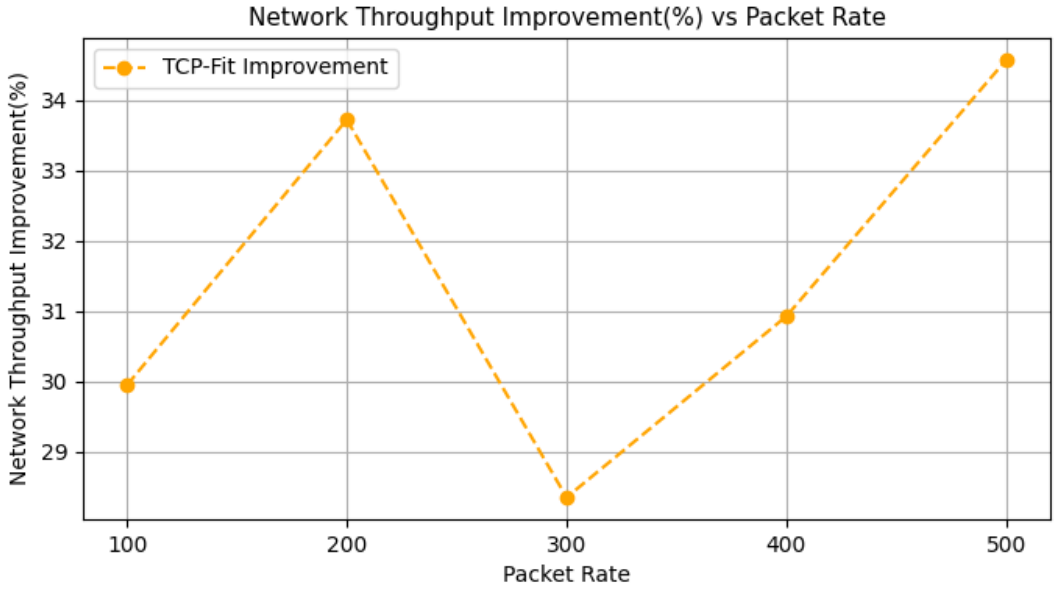

Drop Ratio Increase(%) vs Packet Rate
−3.0
−3.5
−4.0
−4.5
−5.0
−5.5
−6.0
Drop Ratio Increase(%)
TCP-Fit Improvement
100
200
300
400
500
Packet Rate

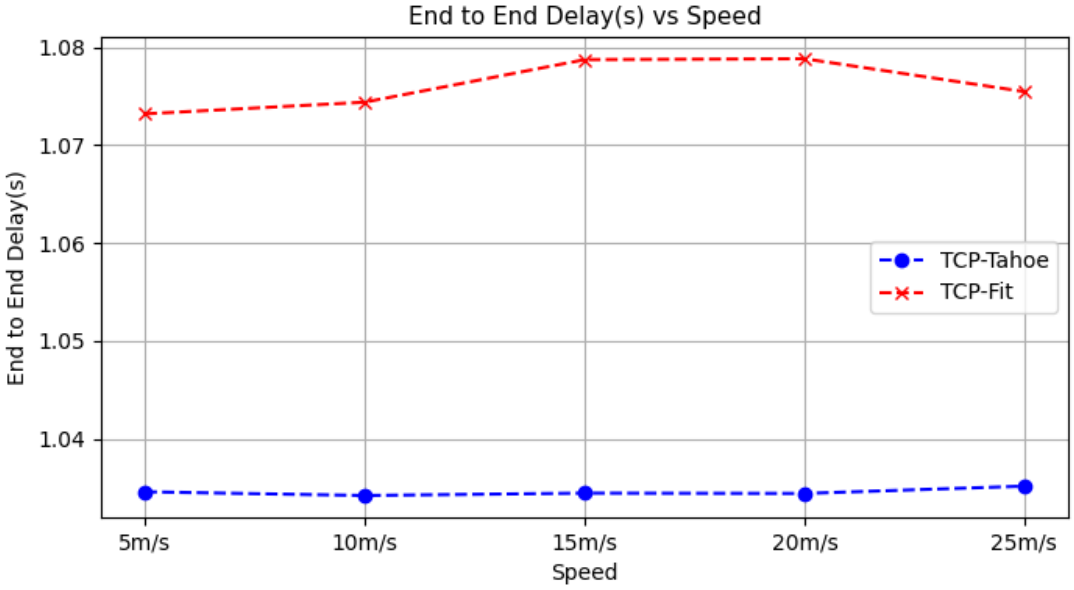
Observations

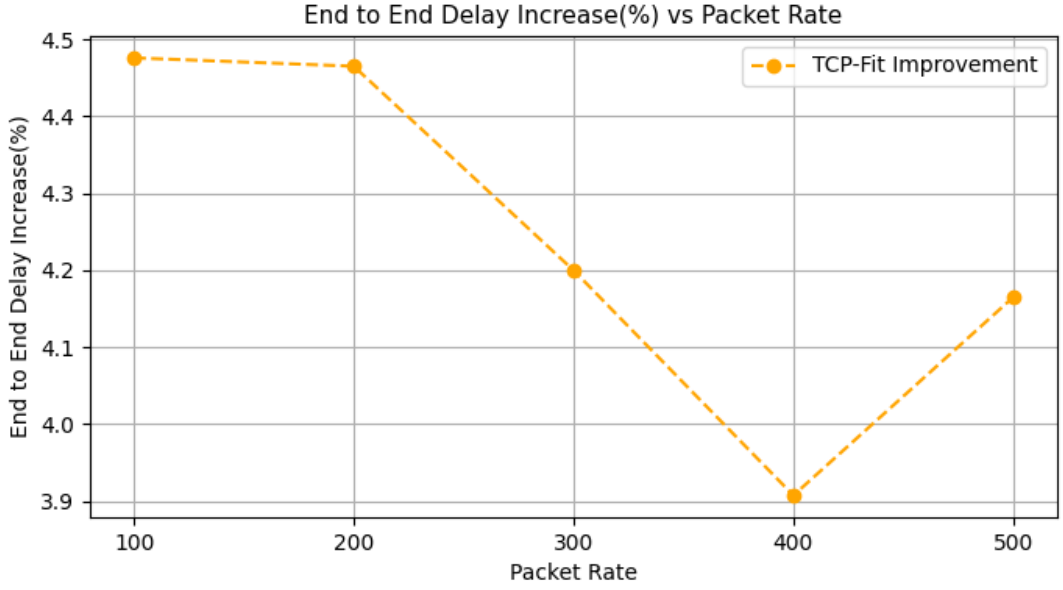
1. The metrics does not seem much affected by the packet rate indicating that bandwidth utilization for both algorithms were were at their maximum capacity for all the packet rate
2. TCP Fit continues to give large improvement in network throughput
3. TCP Fit seems to have slightly higher end to end delay and slightly better delivery ratio and so slightly smaller drop ratio

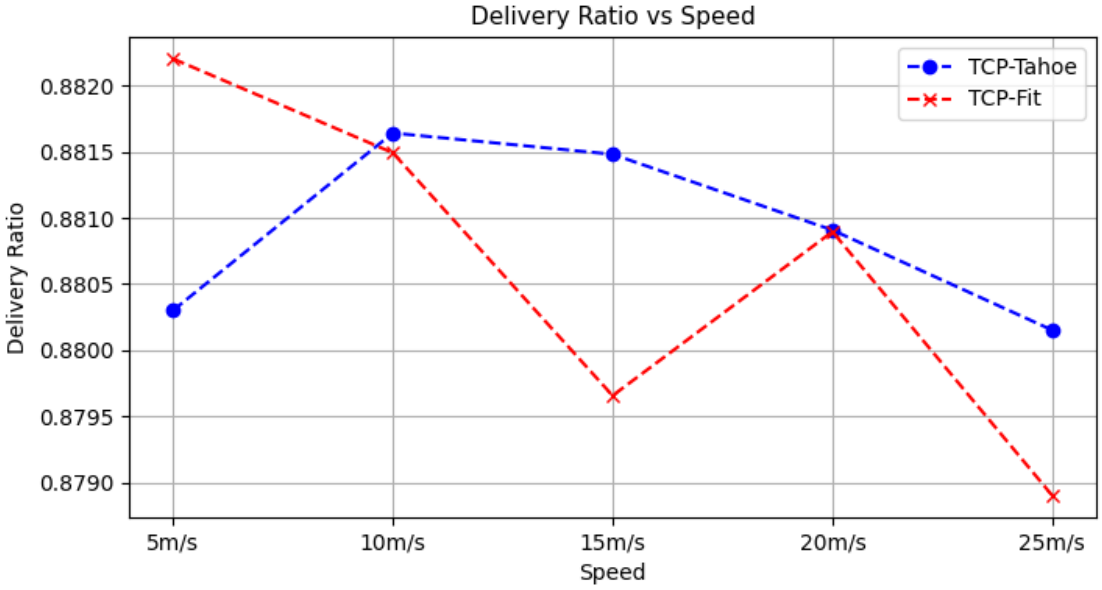
Varying speed

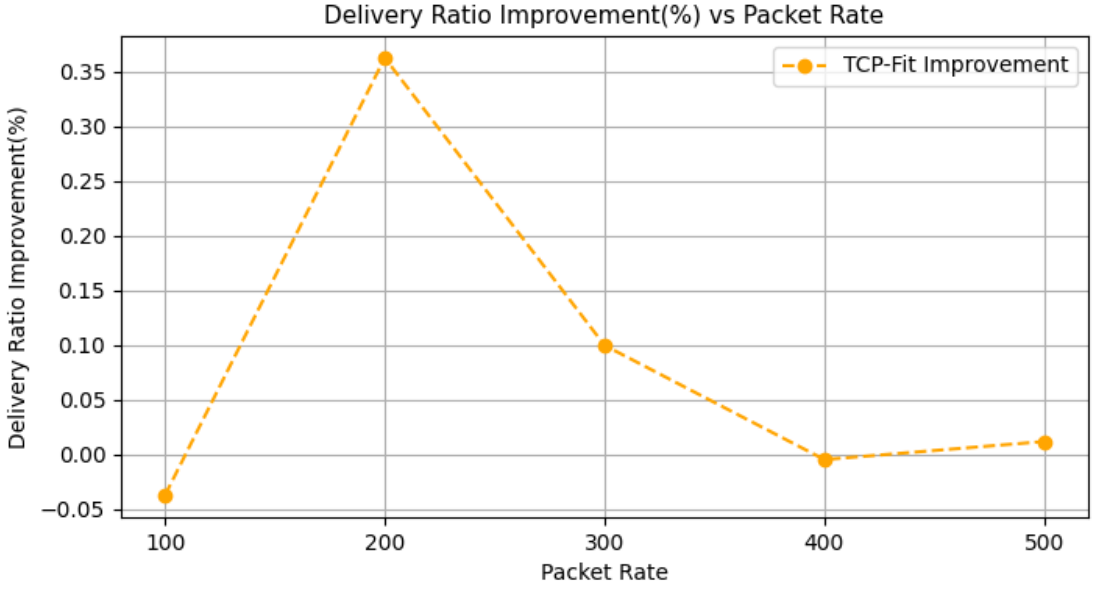


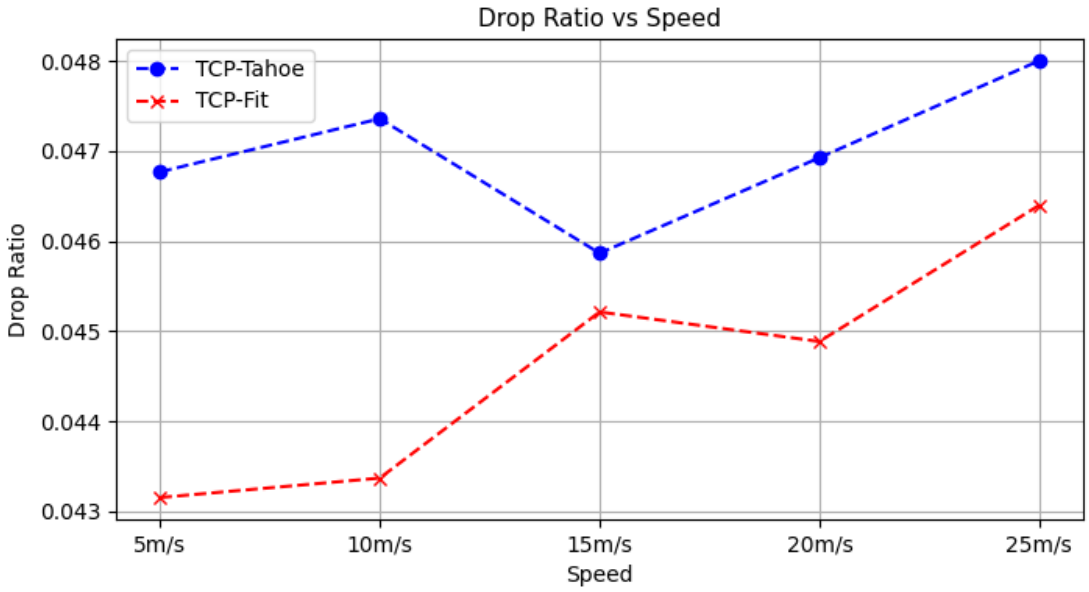


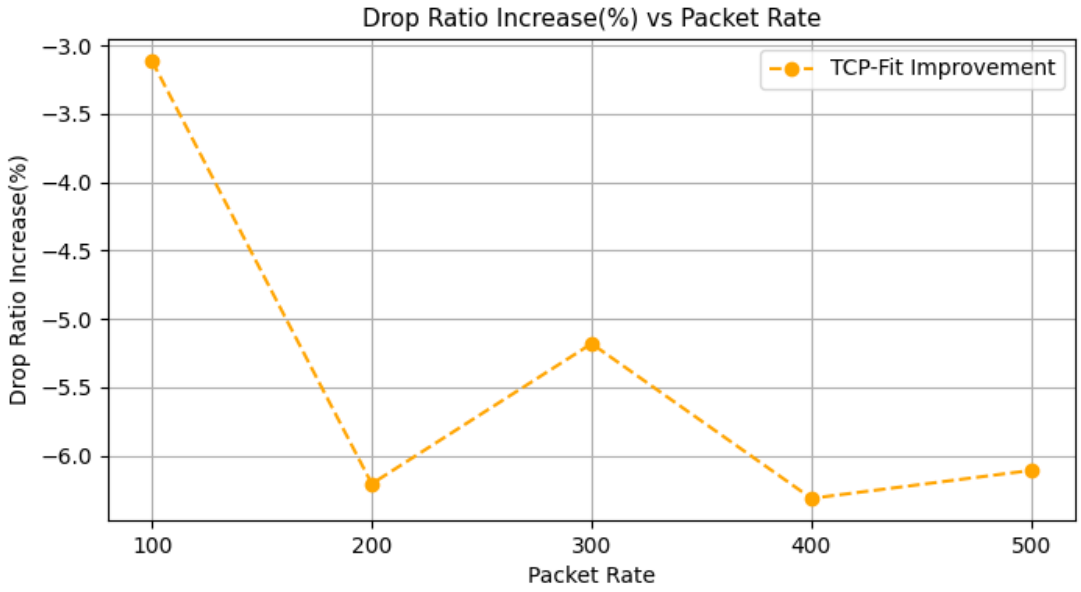

End to End Delay(s) vs Speed
End to End Delay(s)
1.08
1.07
1.06
1.05
1.04
TCP-Tahoe
TCP-Fit
5m/s
10m/s
15m/s
20m/s
25m/s
Speed


End to End Delay Increase(%) vs Packet Rate
End to End Delay Increase(%)
4.5
4.4
4.3
4.2
4.1
4.0
3.9
TCP-Fit Improvement
100
200
300
400
500
Packet Rate


Delivery Ratio vs Speed
TCP-Tahoe
TCP-Fit
Delivery Ratio
0.8820
0.8815
0.8810
0.8805
0.8800
0.8795
0.8790
5m/s
10m/s
15m/s
20m/s
25m/s
Speed


Delivery Ratio Improvement(%) vs Packet Rate
TCP-Fit Improvement
Delivery Ratio Improvement(%)
0.35
0.30
0.25
0.20
0.15
0.10
0.05
0.00
−0.05
100
200
300
400
500
Packet Rate





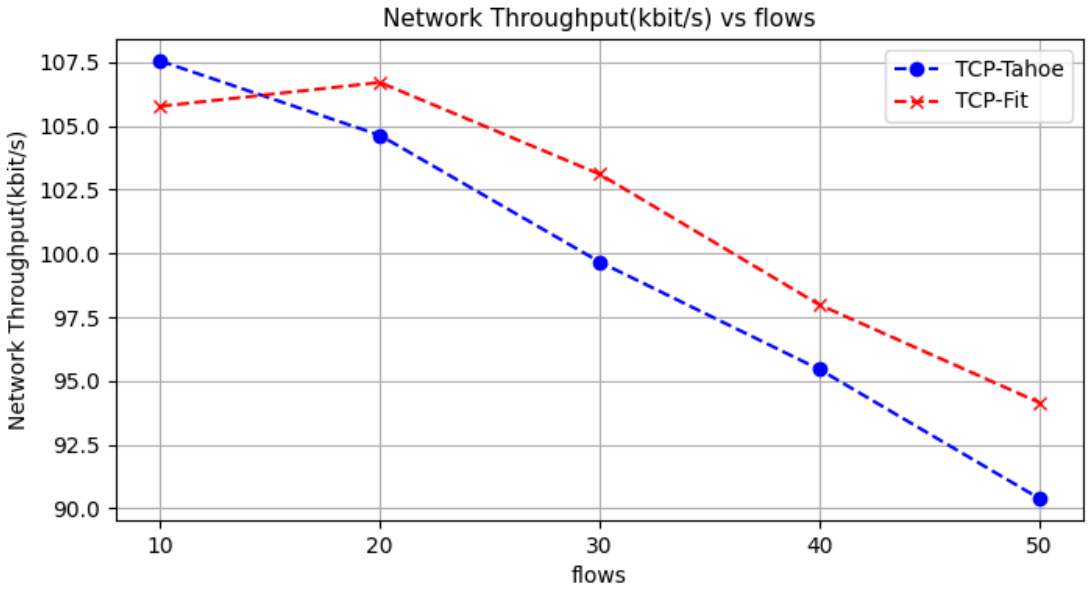
Observations

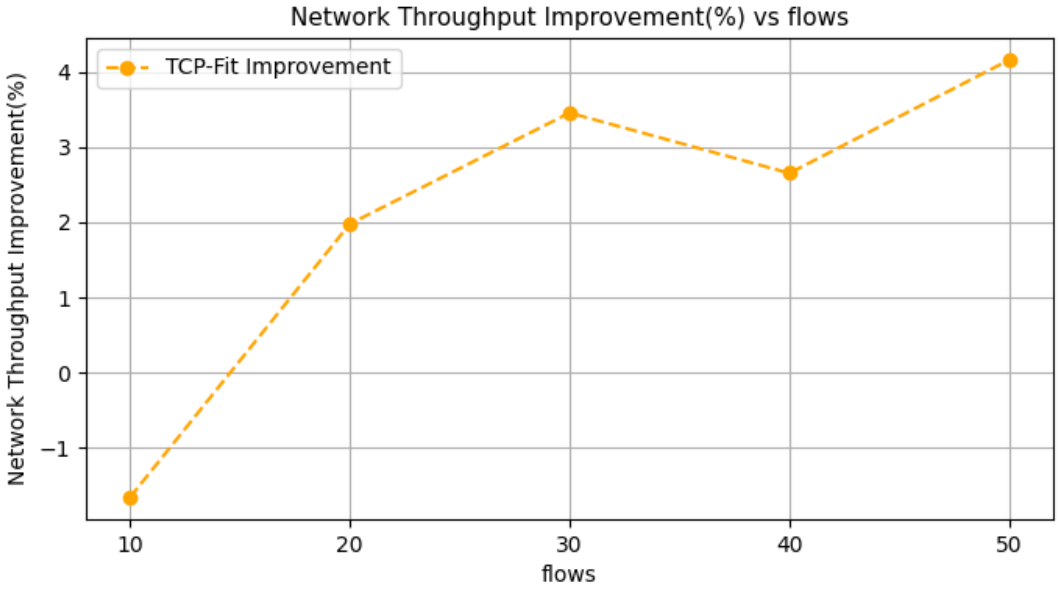
1. Varying speed does not seem to have a notable effect. This could be due to the nature of the motion. The motion was random. The nodes mostly moved about their initial position and so, there probably was not much change in routing paths throughout the simulation due to node movement.

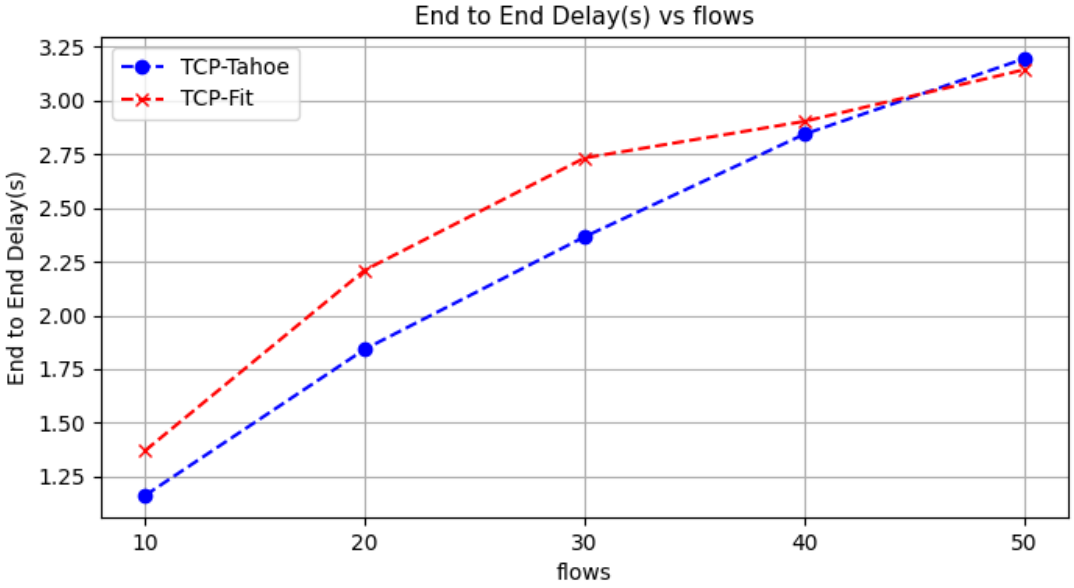
2. TCP Fit continues to give large improvement in network bandwidth with marginal difference in other metrics

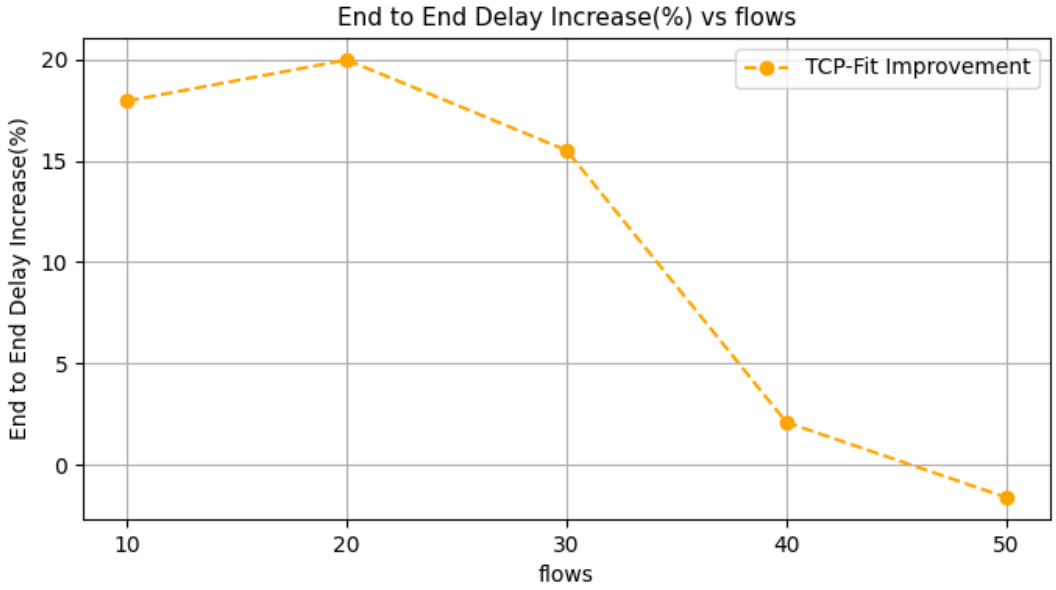
Wireless-Bottleneck

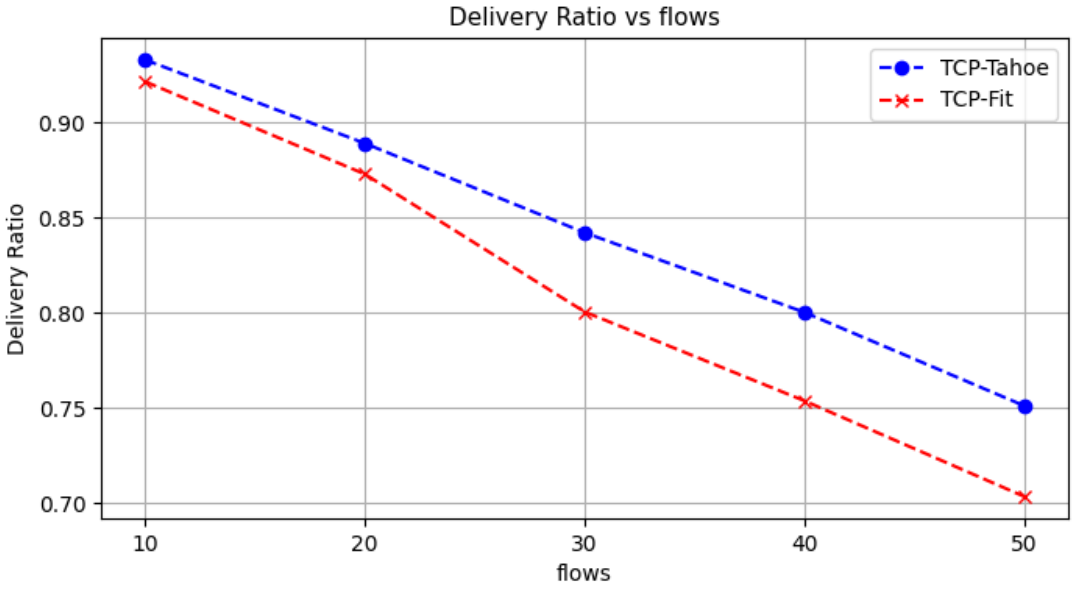
Varying flows

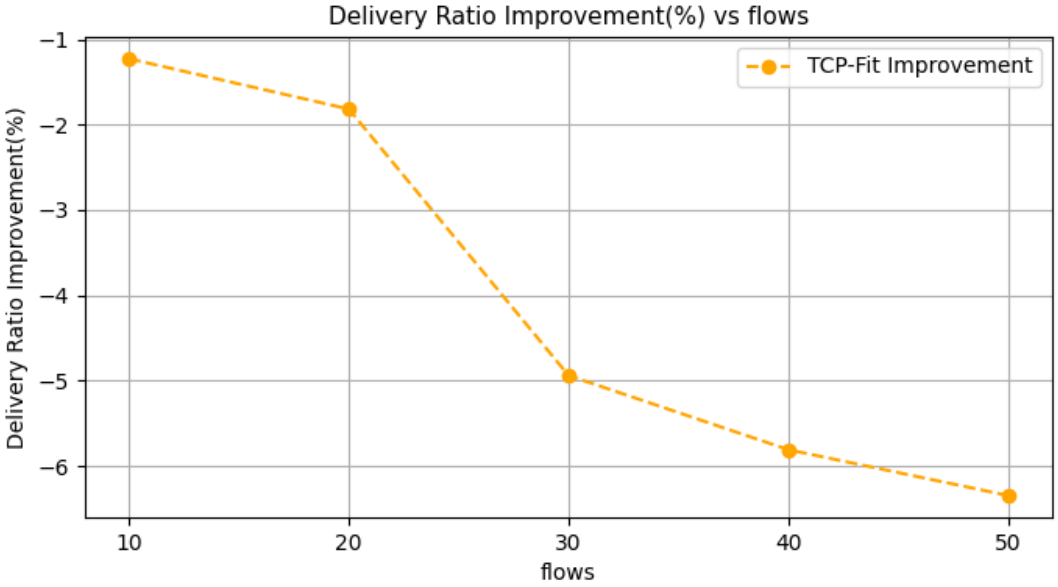


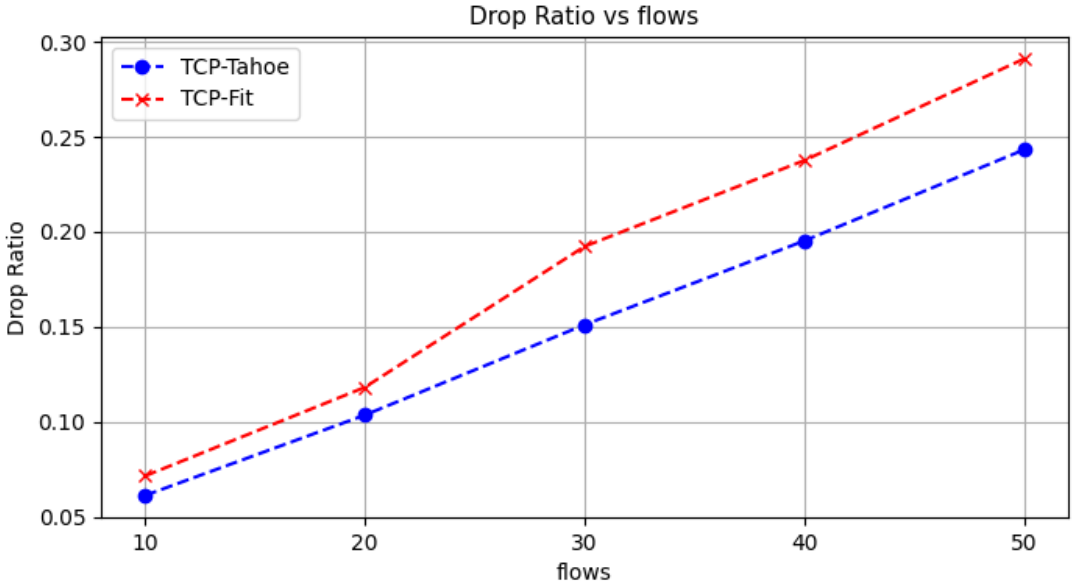


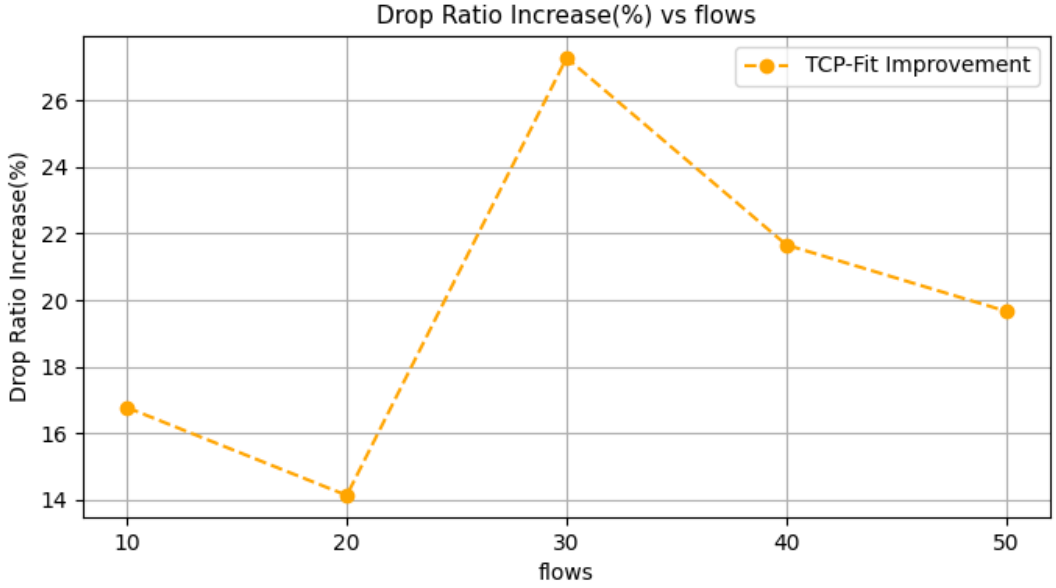

End to End Delay(s) vs flows
TCP-Tahoe
TCP-Fit
End to End Delay(s)
3.25
3.00
2.75
2.50
2.25
2.00
1.75
1.50
1.25
10
20
30
40
50
flows

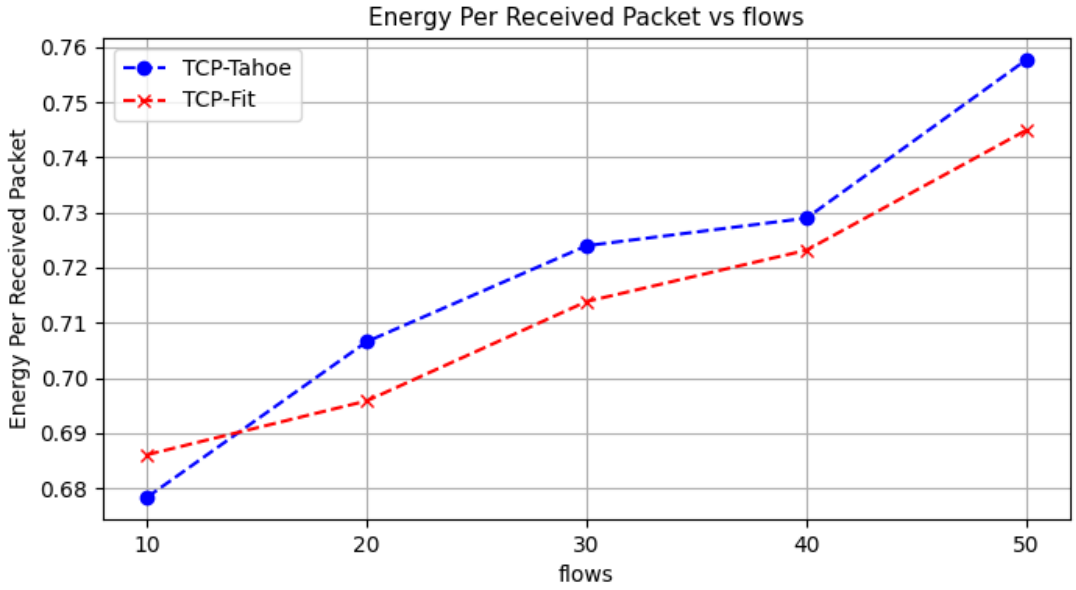

End to End Delay Increase(%) vs flows
TCP-Fit Improvement
End to End Delay Increase(%)
20
15
10
5
0
10
20
30
40
50
flows

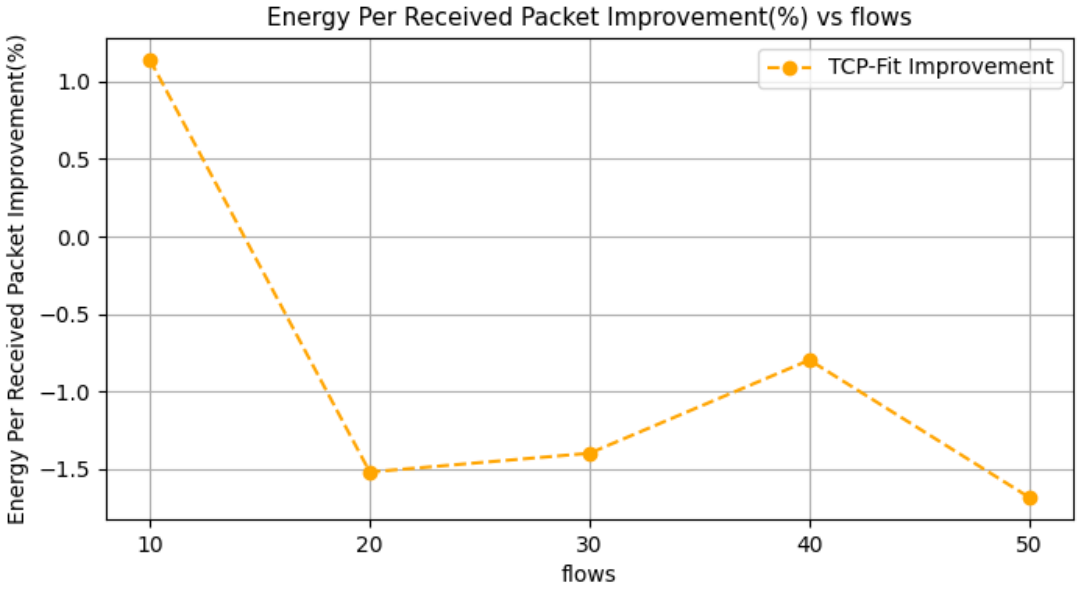

Delivery Ratio vs flows
TCP-Tahoe
TCP-Fit
Delivery Ratio
0.90
0.85
0.80
0.75
0.70
10
20
30
40
50
flows


Delivery Ratio Improvement(%) vs flows
TCP-Fit Improvement
Delivery Ratio Improvement(%)
−1
−2
−3
−4
−5
−6
10
20
30
40
50
flows


Drop Ratio vs flows
TCP-Tahoe
TCP-Fit
Drop Ratio
flows
0.30
0.25
0.20
0.15
0.10
0.05
10
20
30
40
50


Drop Ratio Increase(%) vs flows
TCP-Fit Improvement
Drop Ratio Increase(%)
flows
26
24
22
20
18
16
14
10
20
30
40
50



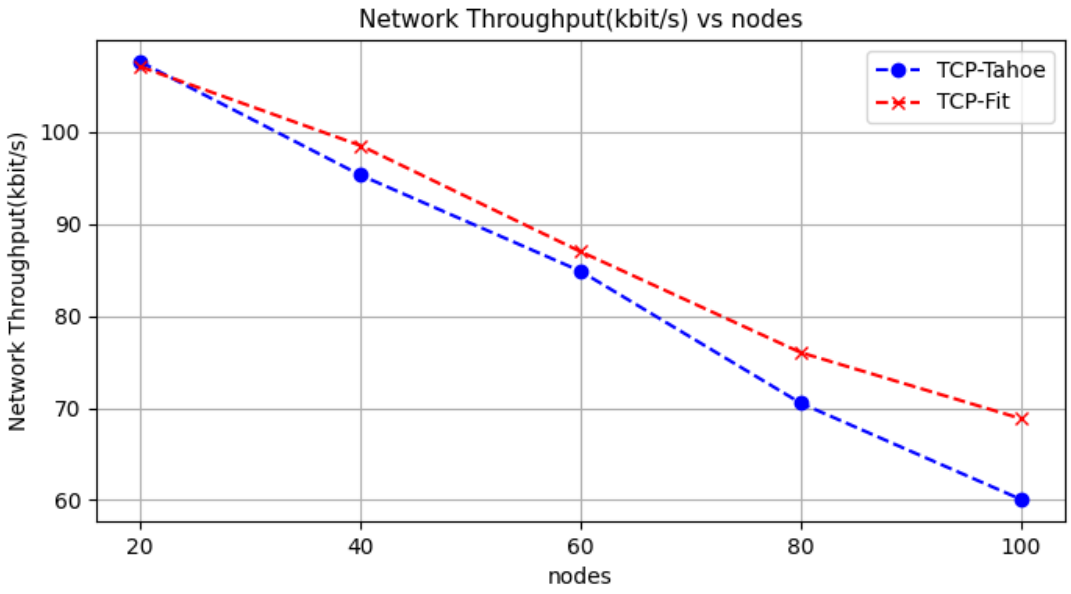


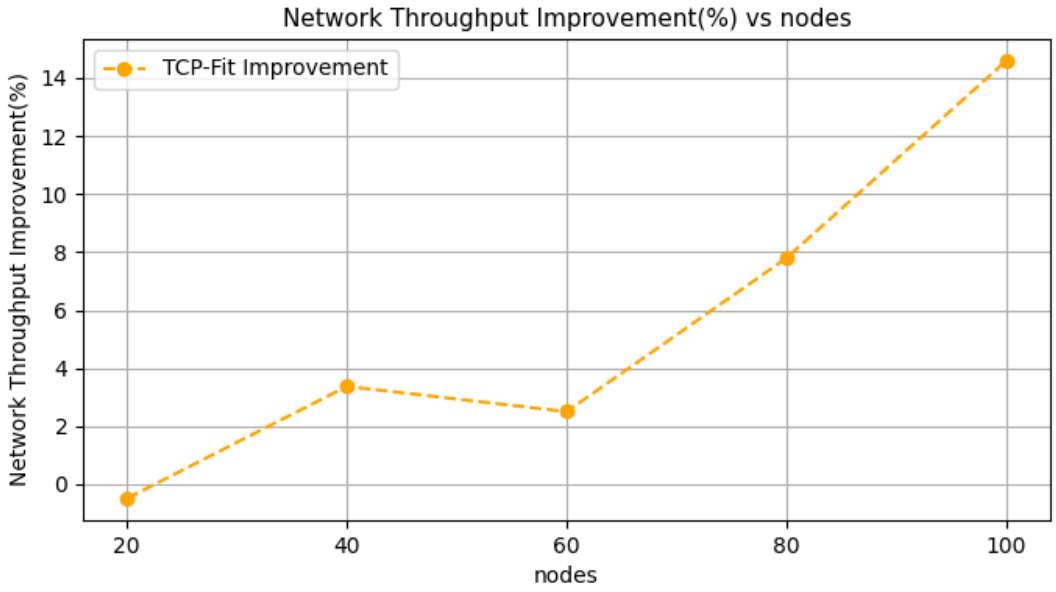
Observations

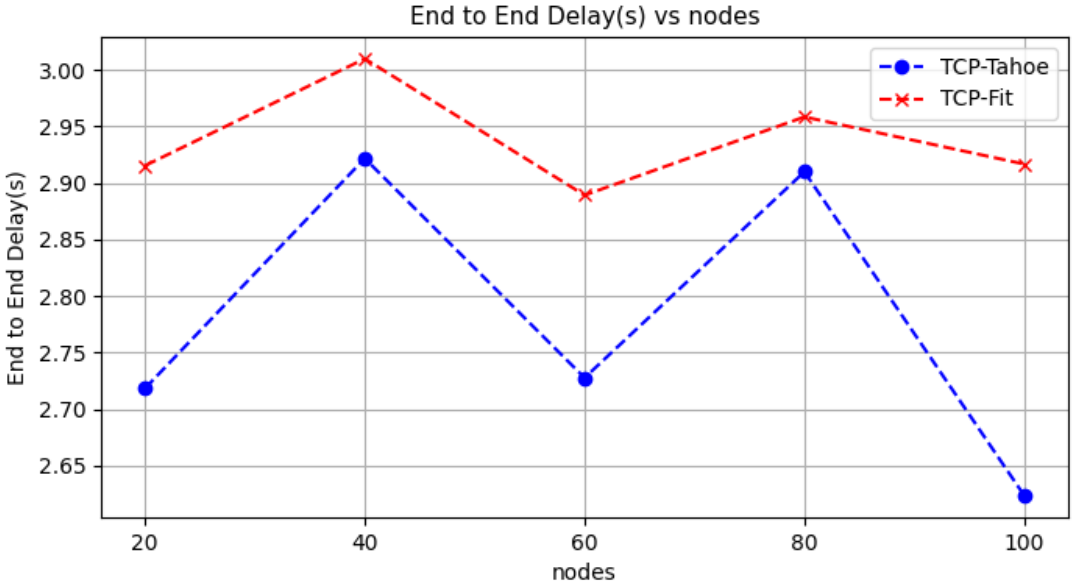
1. Throughput is seen to decrease with the number of flows, indicating that the network is congested. TCP-Fit only performs slightly better than TCP-Tahoe
2. End to end delay increases with more congestion. TCP Fit shows higher end to end delay

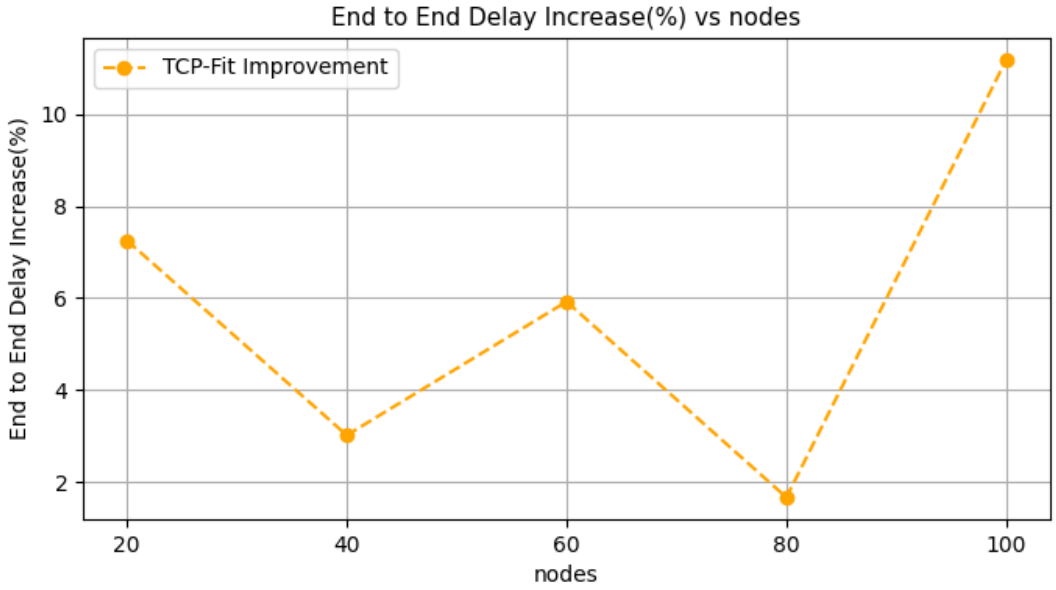
3. Delivery ratio drops and drop ratio increases with increasing flow, as expected from the throughput graph.
4. Higher amount of energy required for each received packet due to the higher packet loss. TCP-Fit shows lower energy requirements. This is due to the increased throughput.
5. TCP-Fit shows quite poor performance on the congested network. The throughput improvement was lower than 5%, however end to end delay increased as much as 20%, delivery ratio decreased by as much as 6% and drop ratio increased by as much as 26%

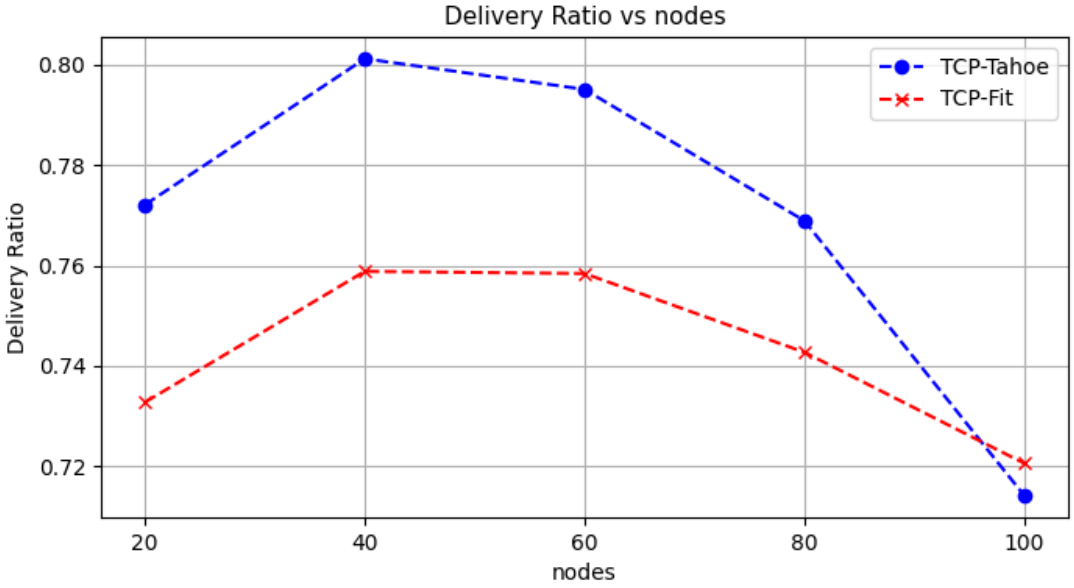
Varying nodes

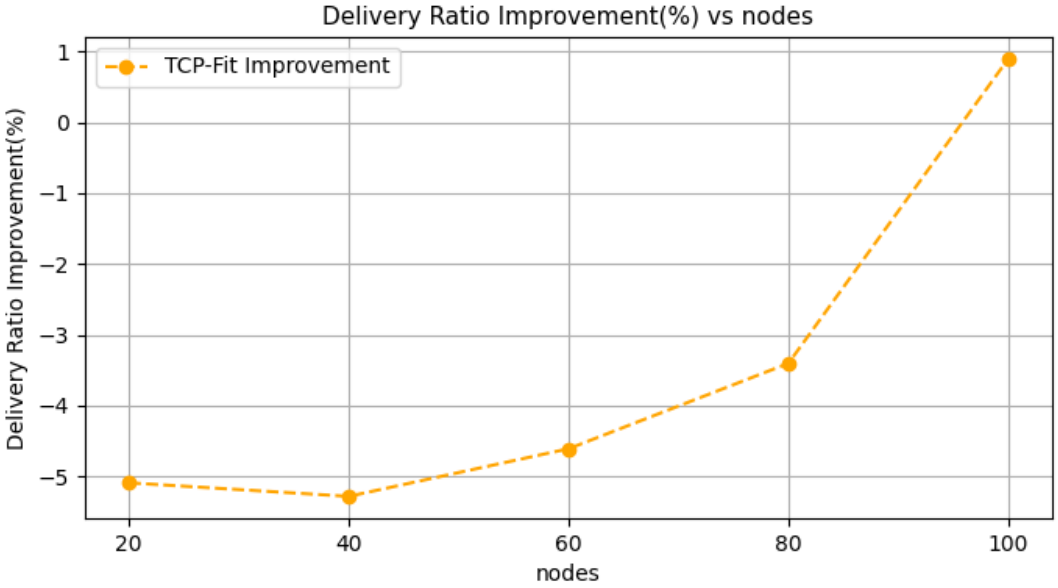


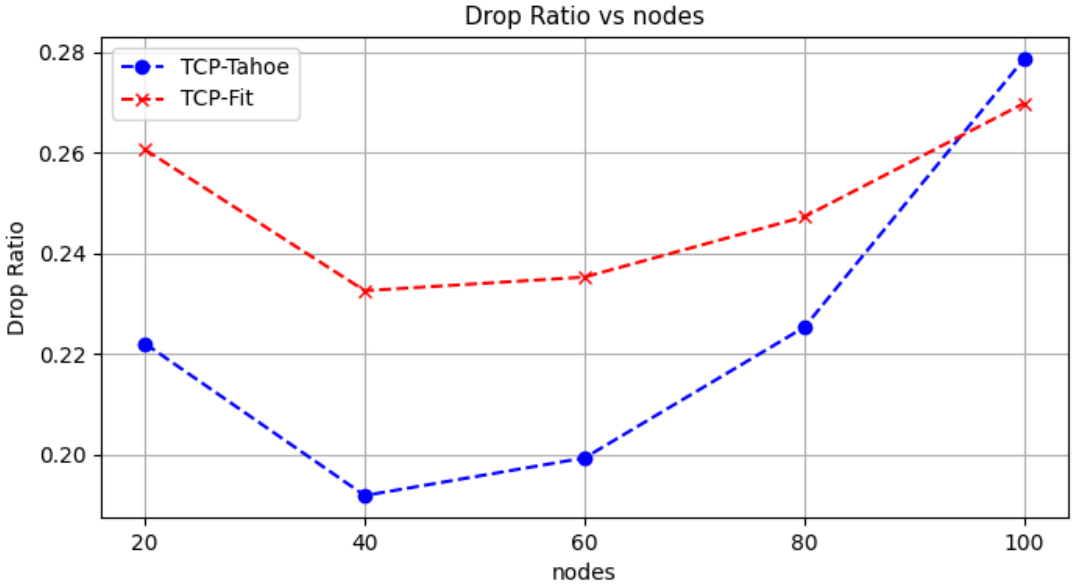


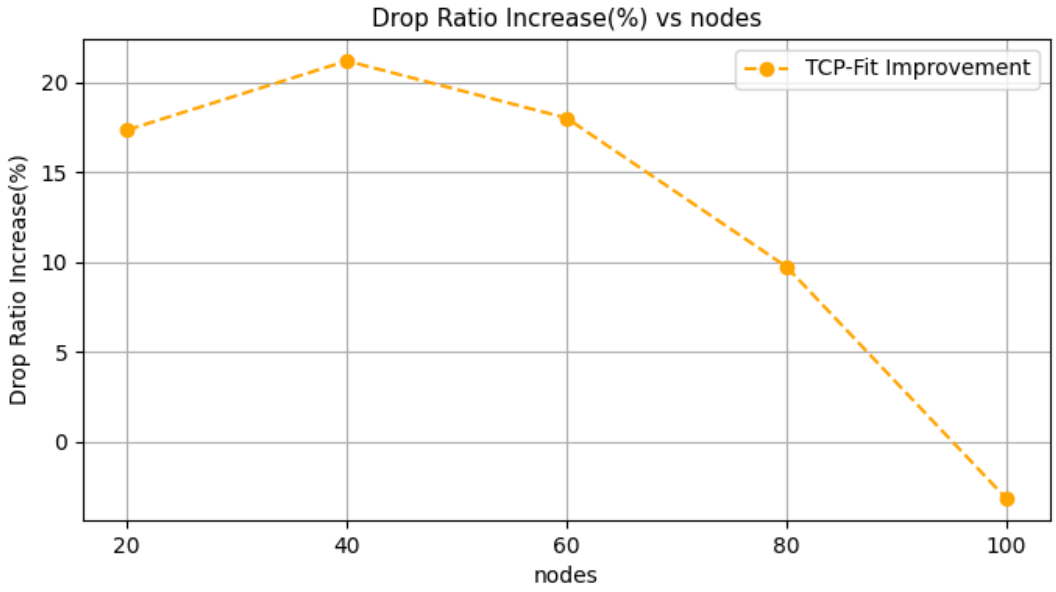

End to End Delay(s) vs nodes
TCP-Tahoe
TCP-Fit
End to End Delay(s)
3.00
2.95
2.90
2.85
2.80
2.75
2.70
2.65
20
40
60
80
100
nodes

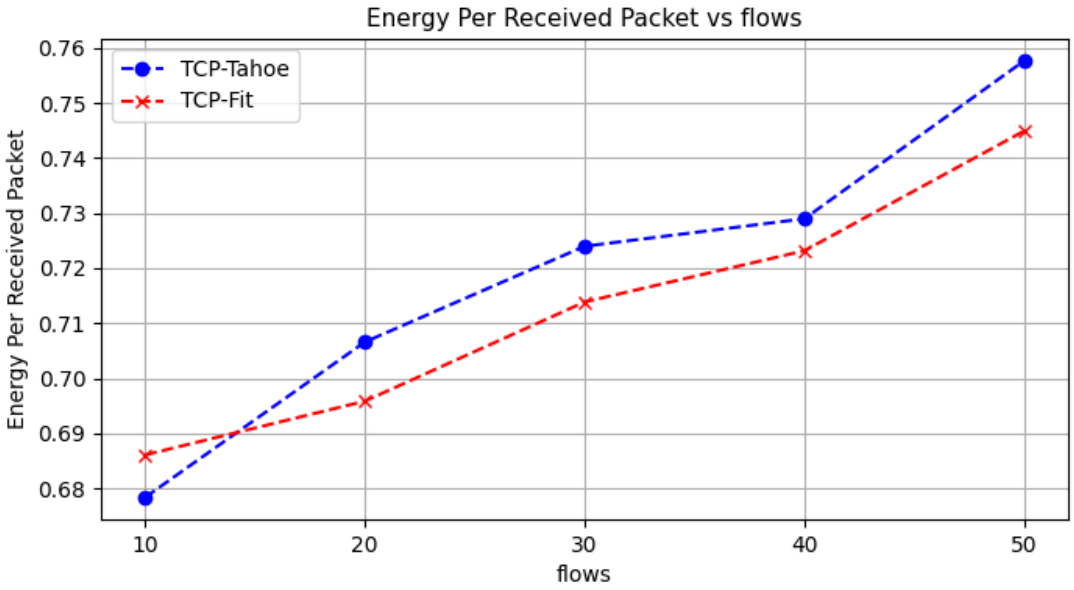

End to End Delay Increase(%) vs nodes
TCP-Fit Improvement
End to End Delay Increase(%)
10
8
6
4
2
20
40
60
80
100
nodes

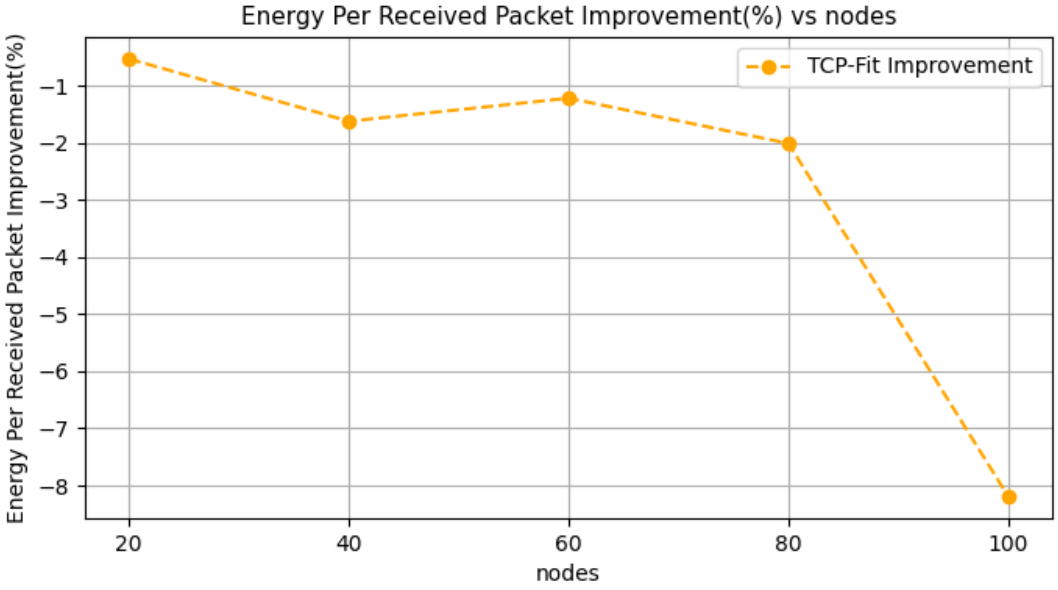

Delivery Ratio vs nodes
TCP-Tahoe
TCP-Fit
Delivery Ratio
0.80
0.78
0.76
0.74
0.72
20
40
60
80
100
nodes


Delivery Ratio Improvement(%) vs nodes
TCP-Fit Improvement
Delivery Ratio Improvement(%)
1
0
−1
−2
−3
−4
−5
20
40
60
80
100
nodes


Drop Ratio vs nodes
TCP-Tahoe
TCP-Fit
Drop Ratio
0.28
0.26
0.24
0.22
0.20
20
40
60
80
100
nodes


Drop Ratio Increase(%) vs nodes
TCP-Fit Improvement
Drop Ratio Increase(%)
20
15
10
5
0
20
40
60
80
100
nodes



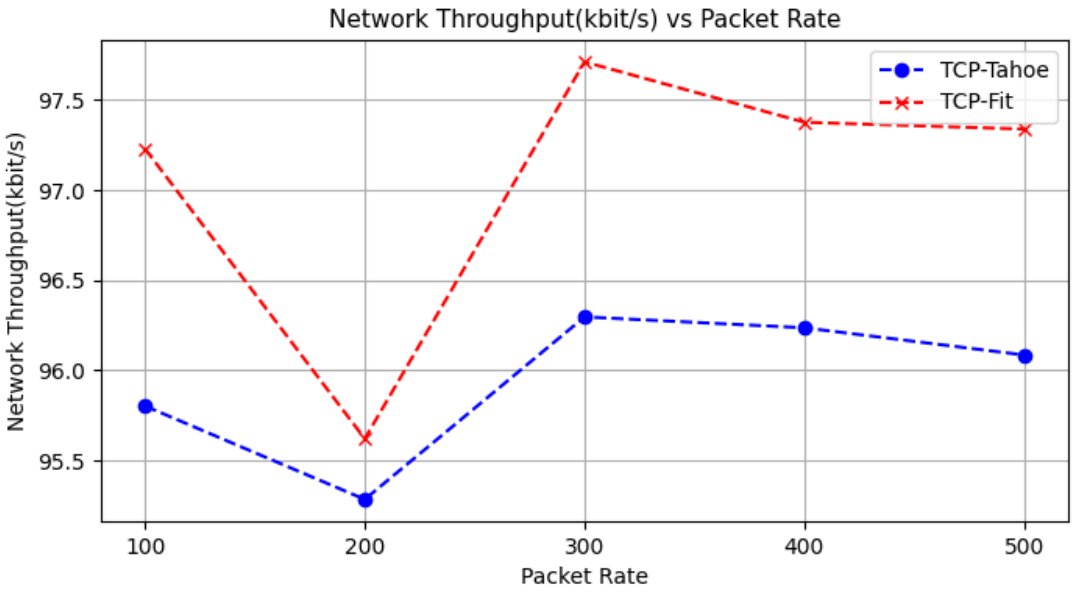


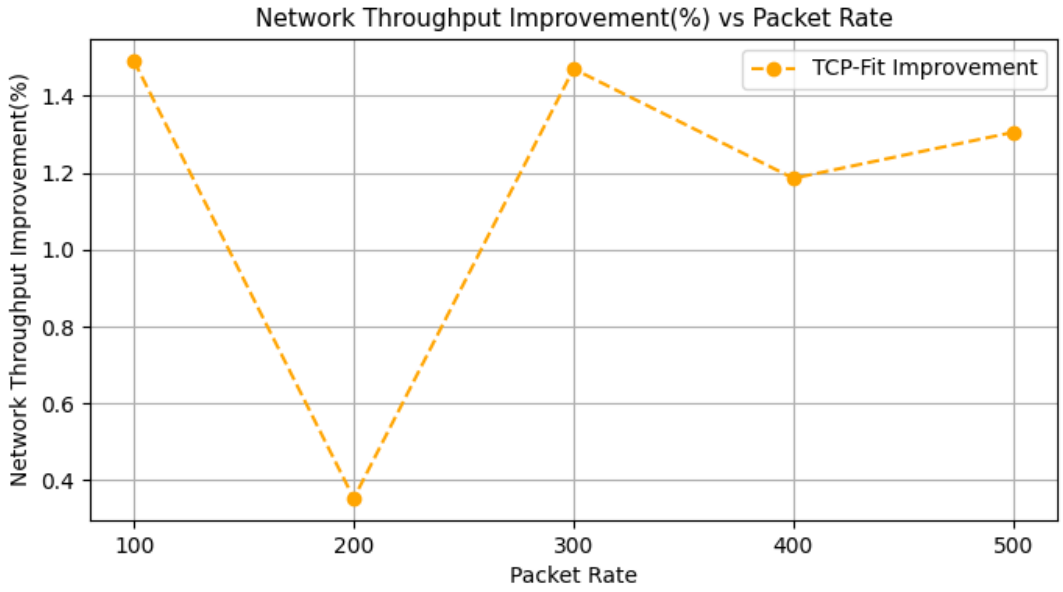
Observations

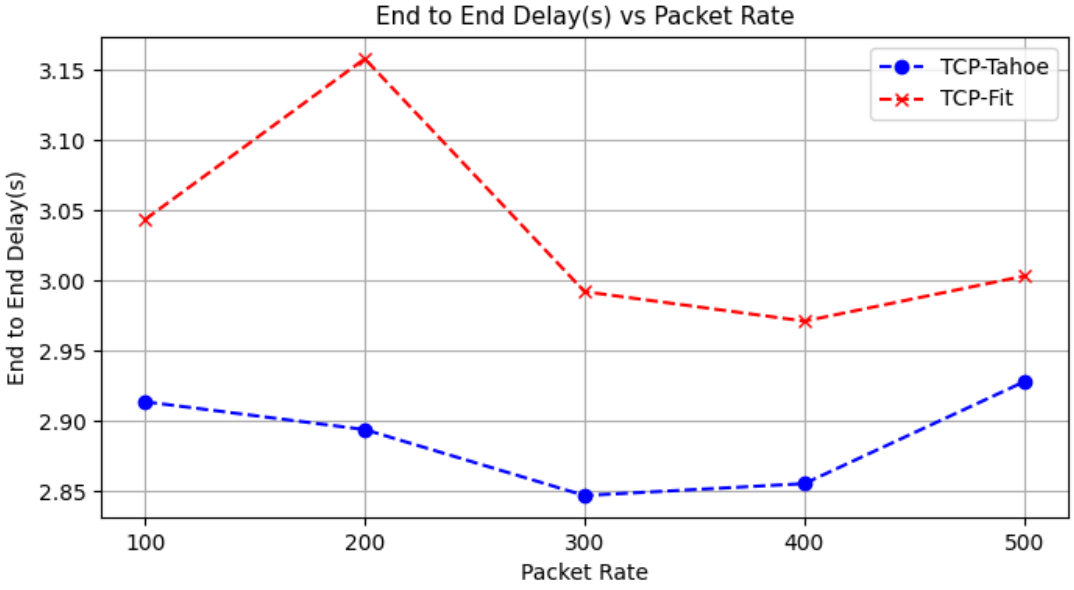
1. Throughput decreases rapidly with increasing node count. However the end to end delay does not show such a trend. This indicates that the increased interference due to higher node count is most likely the cause of decreased throughput
2. End to end delay is increased with TCP-Fit algorithm
3. Energy per received packet also increased

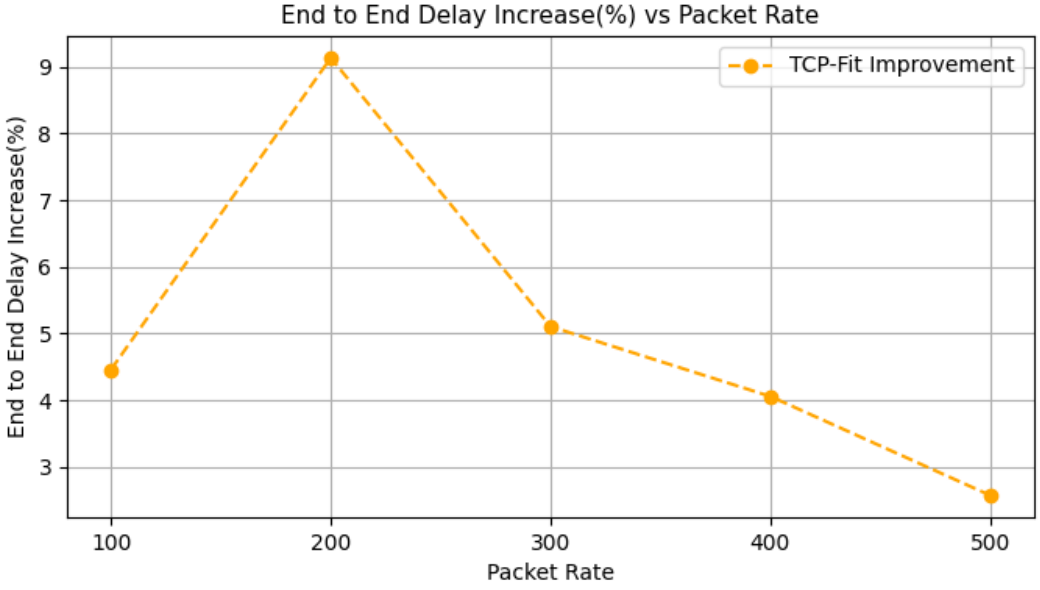
4. Throughput improvements are not as much as previous topologies

Varying packet rate

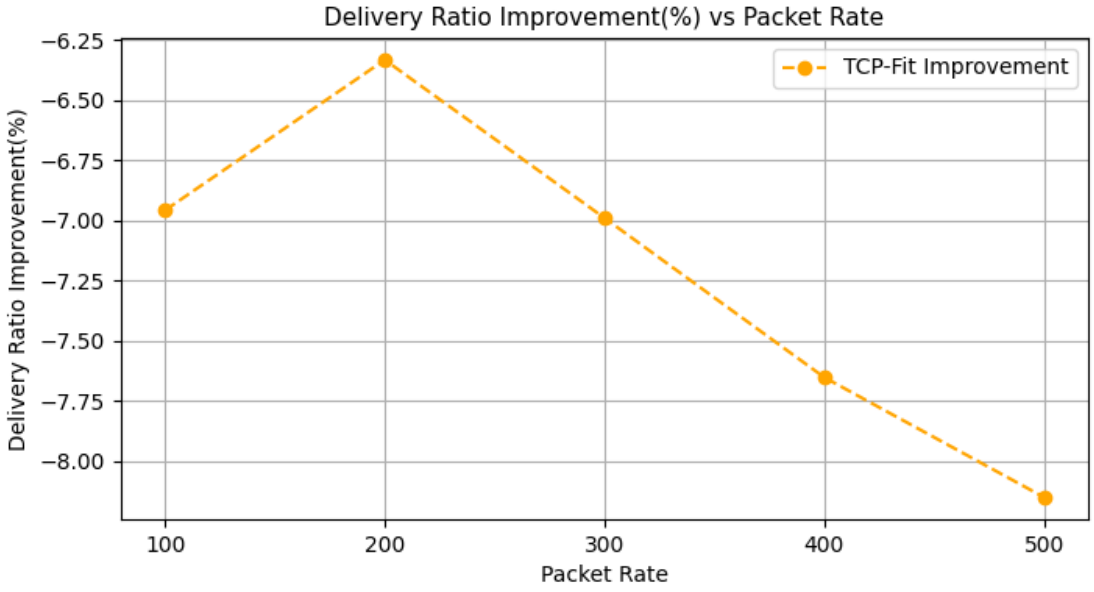


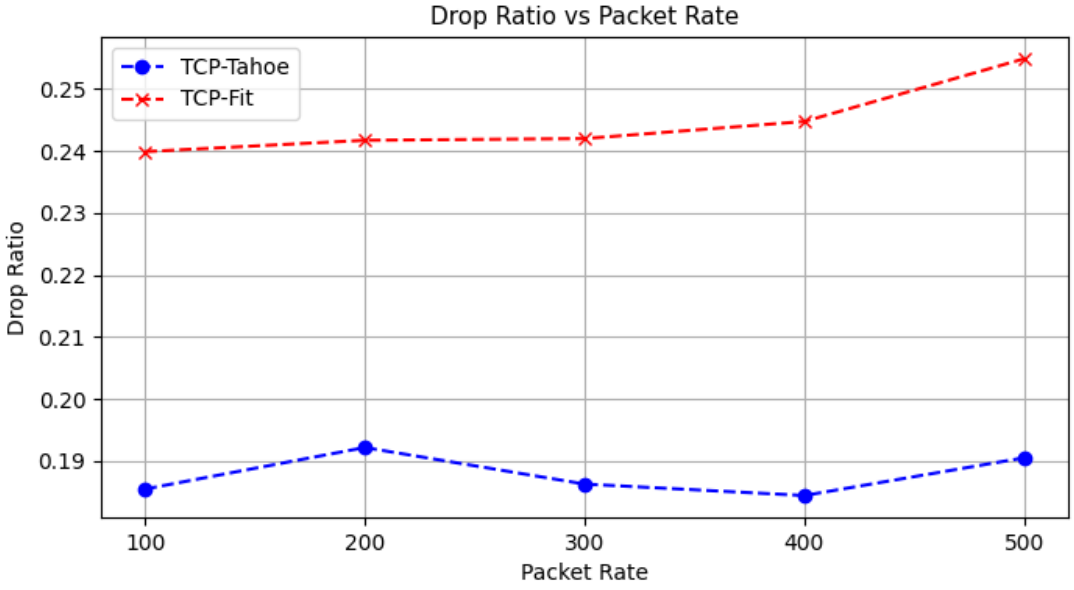


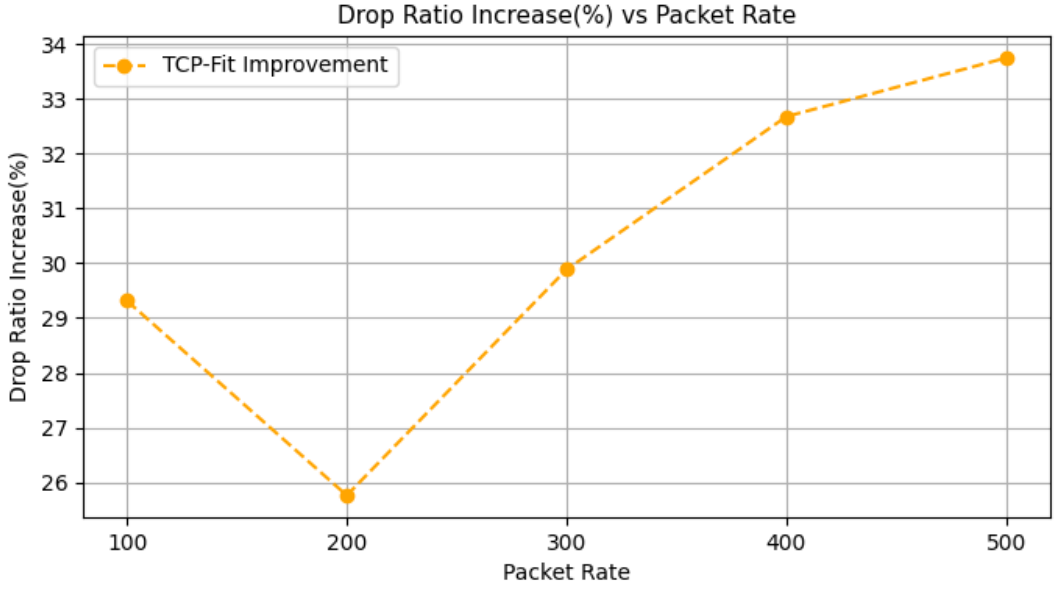

End to End Delay(s) vs Packet Rate
TCP-Tahoe
TCP-Fit
End to End Delay(s)
3.15
3.10
3.05
3.00
2.95
2.90
2.85
100
200
300
400
500
Packet Rate

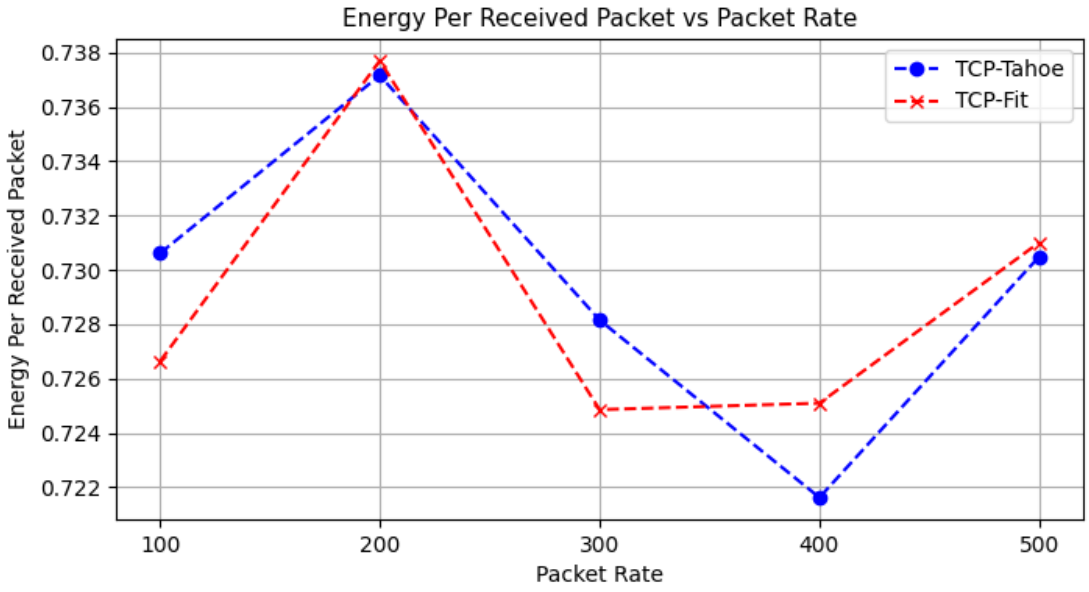

End to End Delay Increase(%) vs Packet Rate
TCP-Fit Improvement
End to End Delay Increase(%)
9
8
7
6
5
4
3
100
200
300
400
500
Packet Rate

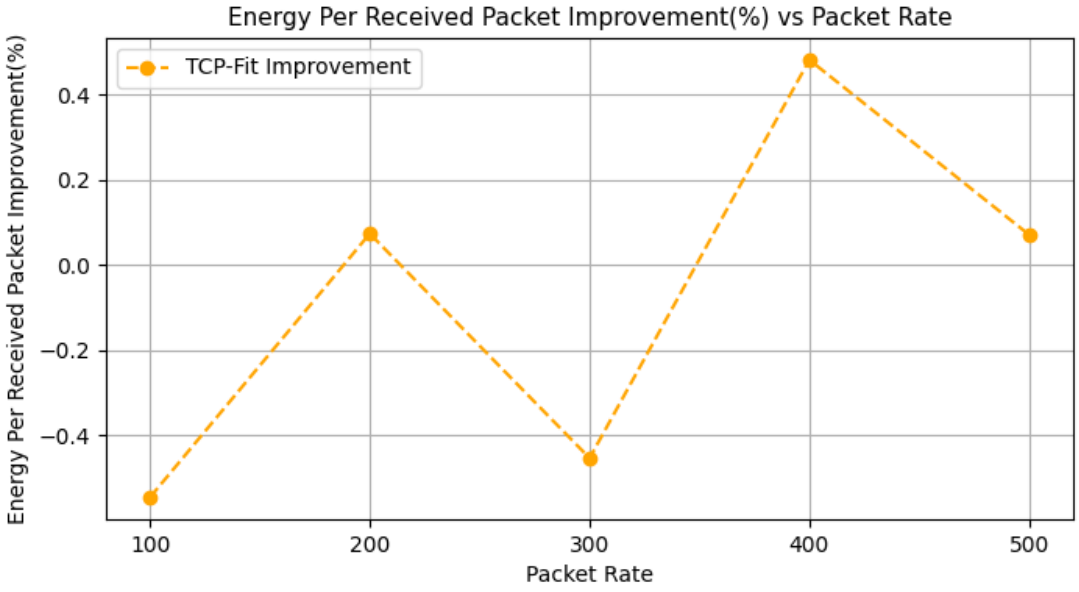

Delivery Ratio vs Packet Rate
Delivery Ratio
0.81
0.80
0.79
0.78
0.77
0.76
0.75
0.74
TCP-Tahoe
TCP-Fit
100
200
300
400
500
Packet Rate


Delivery Ratio Improvement(%) vs Packet Rate
Delivery Ratio Improvement(%)
−6.25
−6.50
−6.75
−7.00
−7.25
−7.50
−7.75
−8.00
TCP-Fit Improvement
100
200
300
400
500
Packet Rate


Drop Ratio vs Packet Rate
TCP-Tahoe
TCP-Fit
Drop Ratio
0.25
0.24
0.23
0.22
0.21
0.20
0.19
100
200
300
400
500
Packet Rate


Drop Ratio Increase(%) vs Packet Rate
TCP-Fit Improvement
Drop Ratio Increase(%)
34
33
32
31
30
29
28
27
26
100
200
300
400
500
Packet Rate

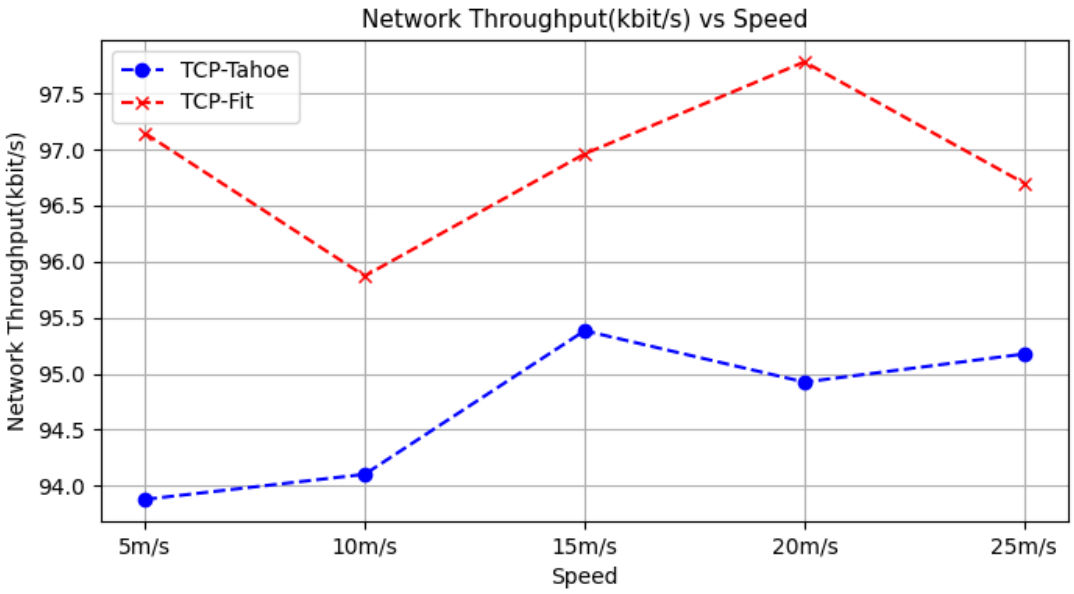


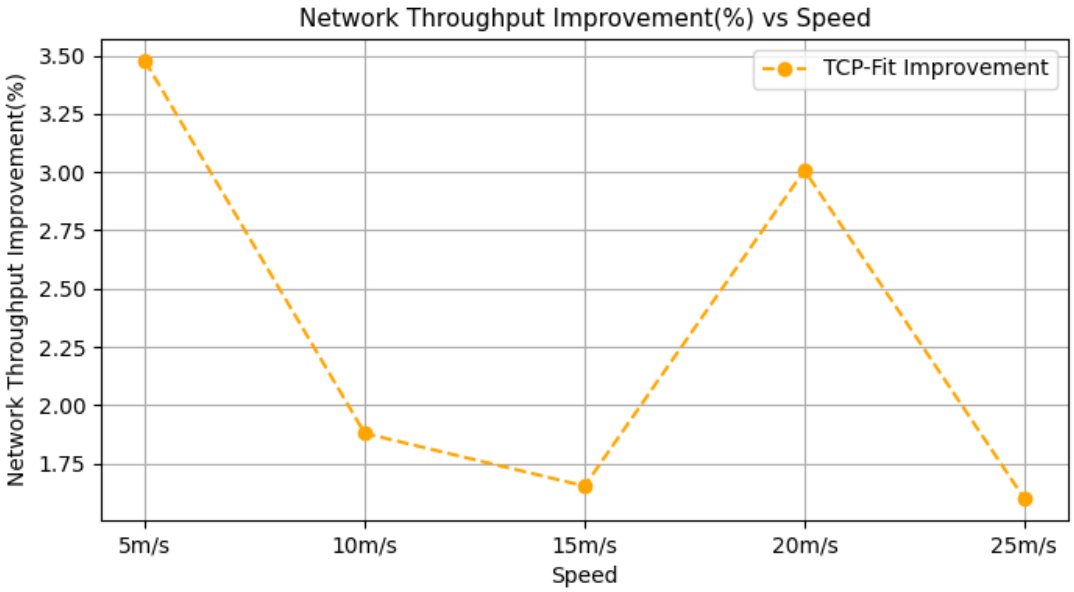


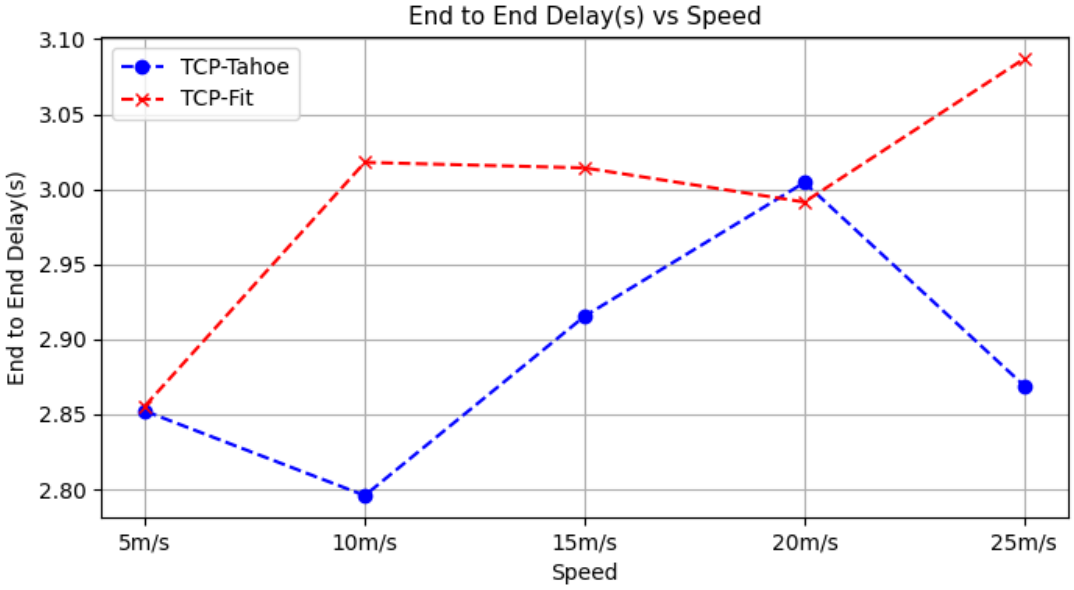
Observations

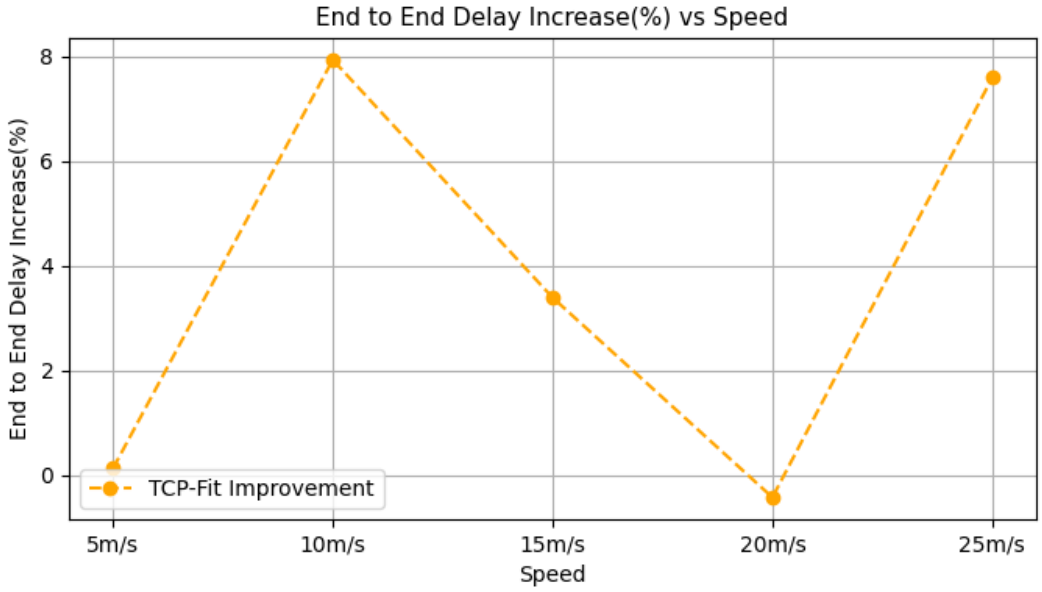
1. The metrics does not seem much affected by the packet rate indicating that bandwidth utilization for both algorithms were at their maximum capacity for all the packet rate
2. TCP-Fit performs very poorly. Its 1% increases in throughput comes at a cost of higher end to end delay, higher drop rate, lower delivery ratio

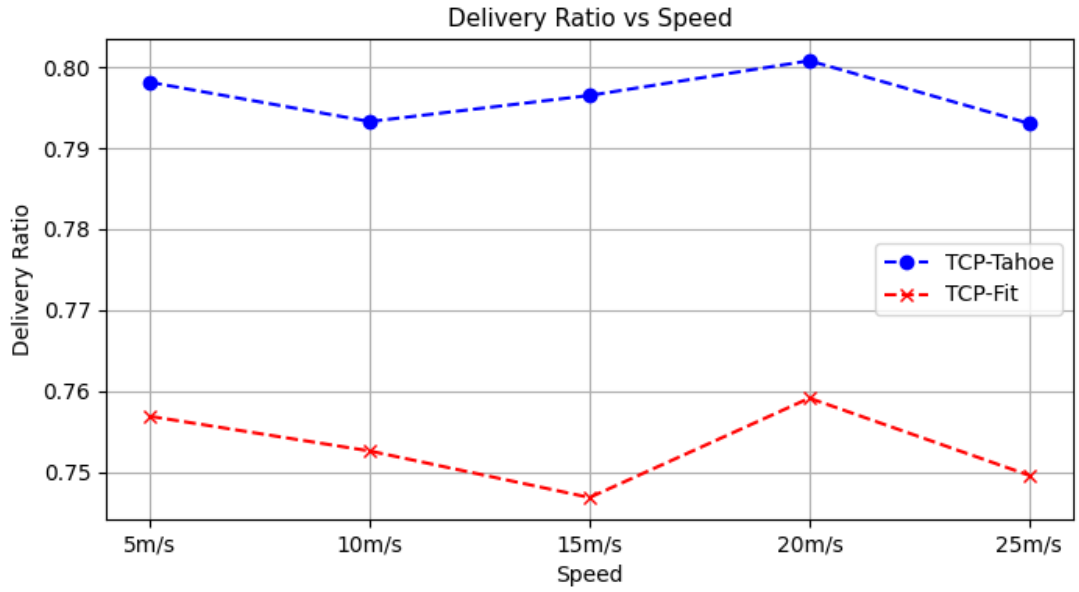
Varying speed

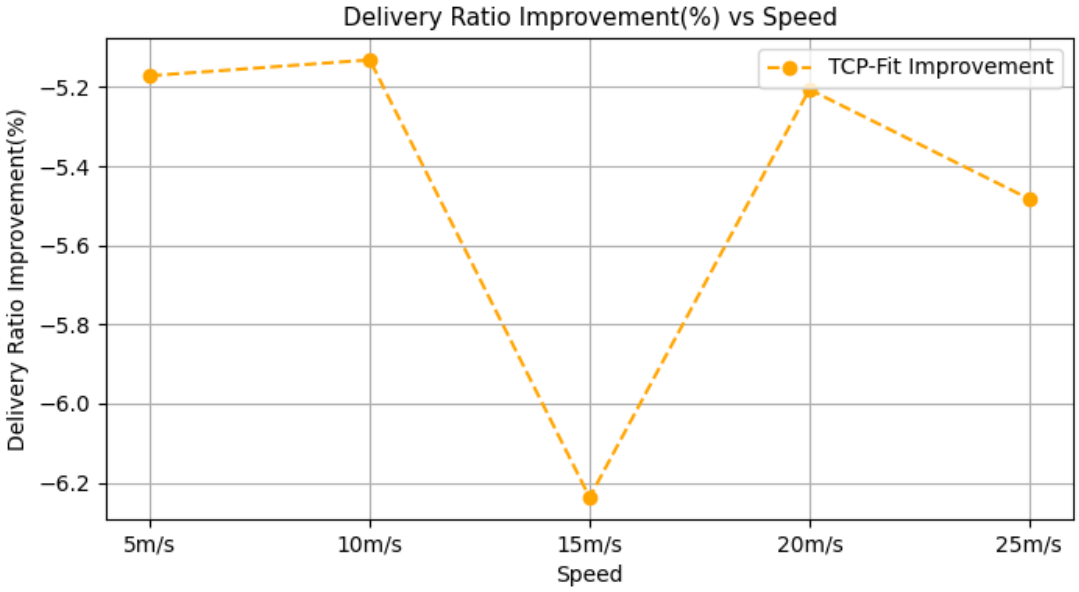


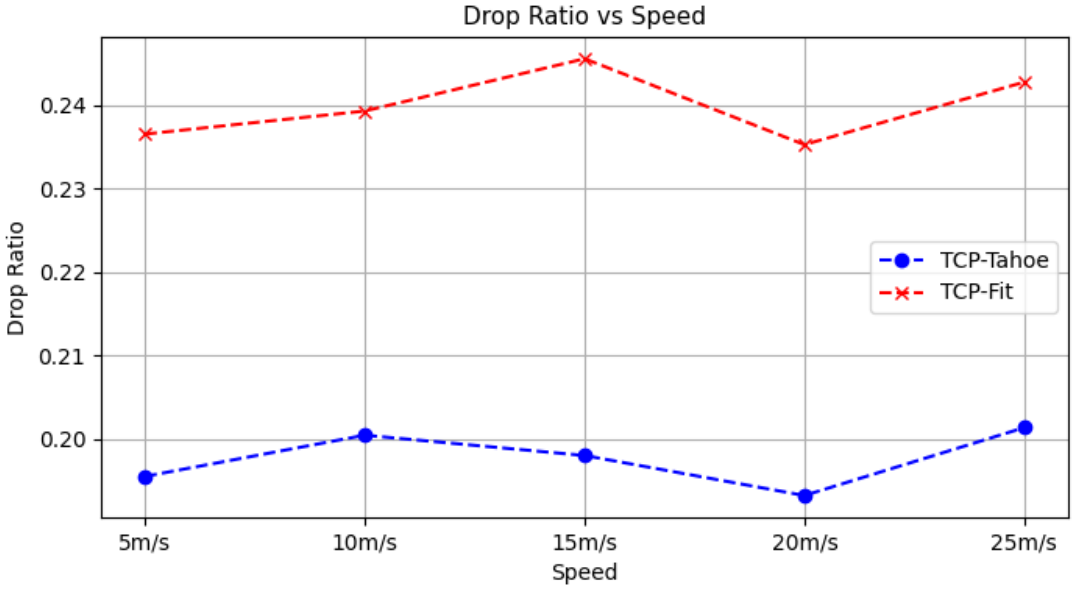


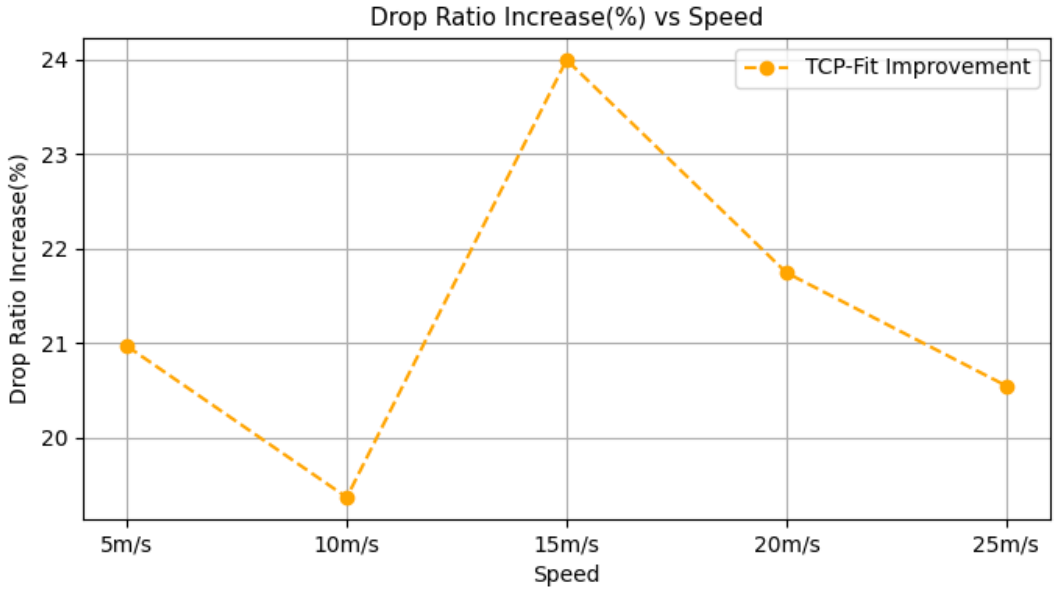

End to End Delay(s) vs Speed
TCP-Tahoe
TCP-Fit
End to End Delay(s)
3.10
3.05
3.00
2.95
2.90
2.85
2.80
5m/s
10m/s
15m/s
20m/s
25m/s
Speed

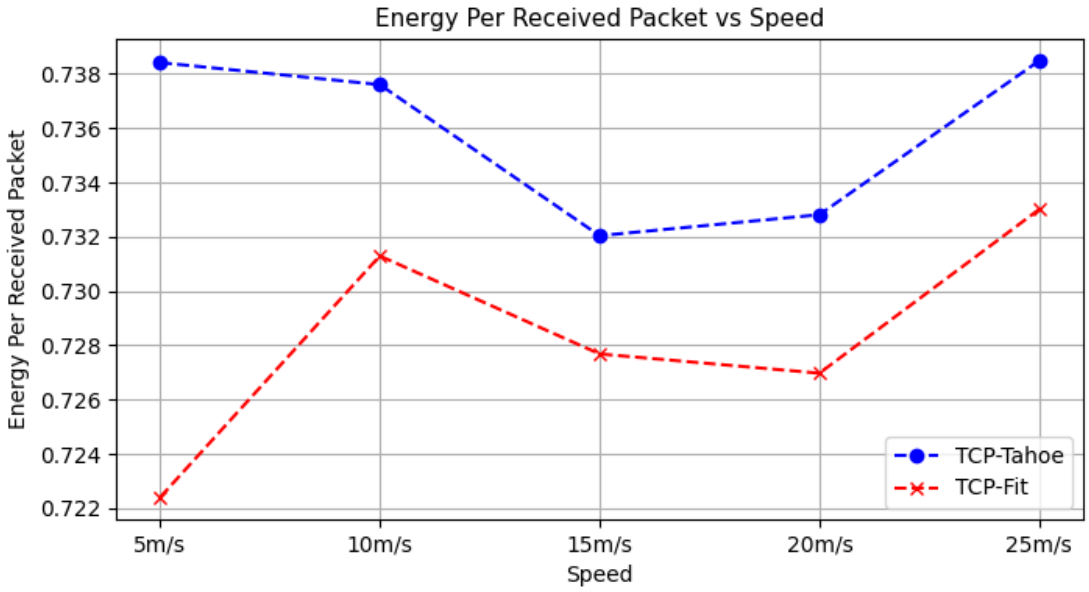

End to End Delay Increase(%) vs Speed
TCP-Fit Improvement
End to End Delay Increase(%)
8
6
4
2
0
5m/s
10m/s
15m/s
20m/s
25m/s
Speed

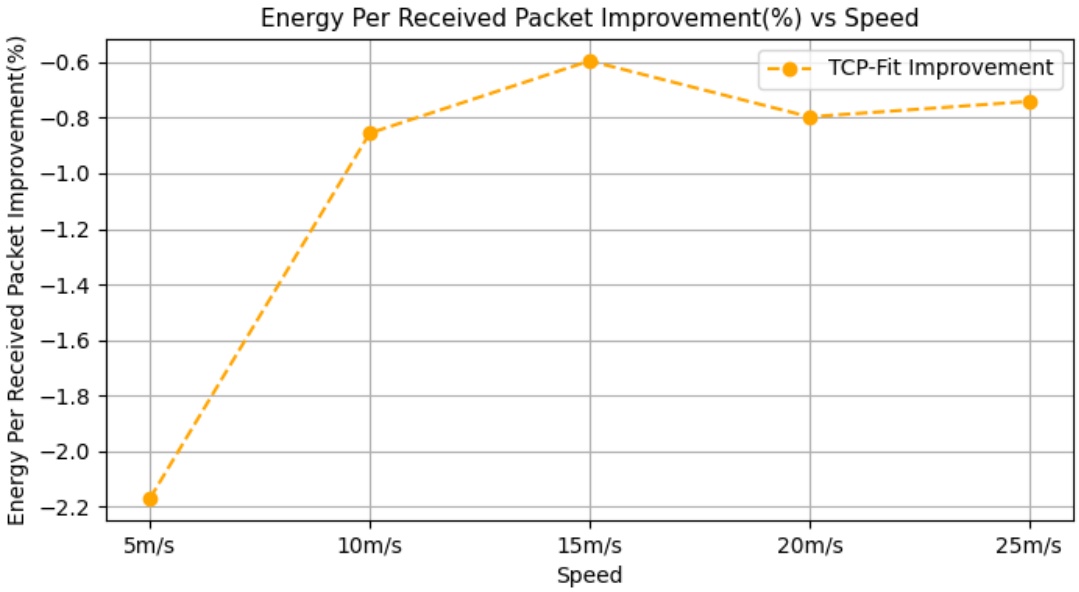

Delivery Ratio vs Speed
Delivery Ratio
0.80
0.79
0.78
0.77
0.76
0.75
TCP-Tahoe
TCP-Fit
5m/s
10m/s
15m/s
20m/s
25m/s
Speed


Delivery Ratio Improvement(%) vs Speed
Delivery Ratio Improvement(%)
−5.2
−5.4
−5.6
−5.8
−6.0
−6.2
TCP-Fit Improvement
5m/s
10m/s
15m/s
20m/s
25m/s
Speed


Drop Ratio vs Speed
Drop Ratio
0.24
0.23
0.22
0.21
0.20
TCP-Tahoe
TCP-Fit
5m/s
10m/s
15m/s
20m/s
25m/s
Speed


Drop Ratio Increase(%) vs Speed
Drop Ratio Increase(%)
24
23
22
21
20
TCP-Fit Improvement
5m/s
10m/s
15m/s
20m/s
25m/s
Speed



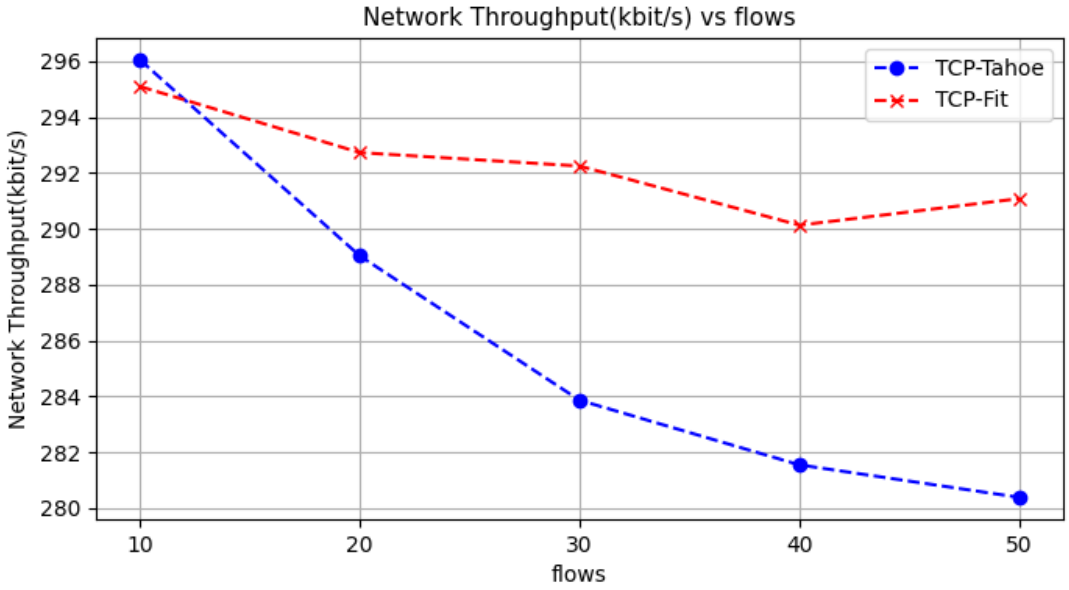


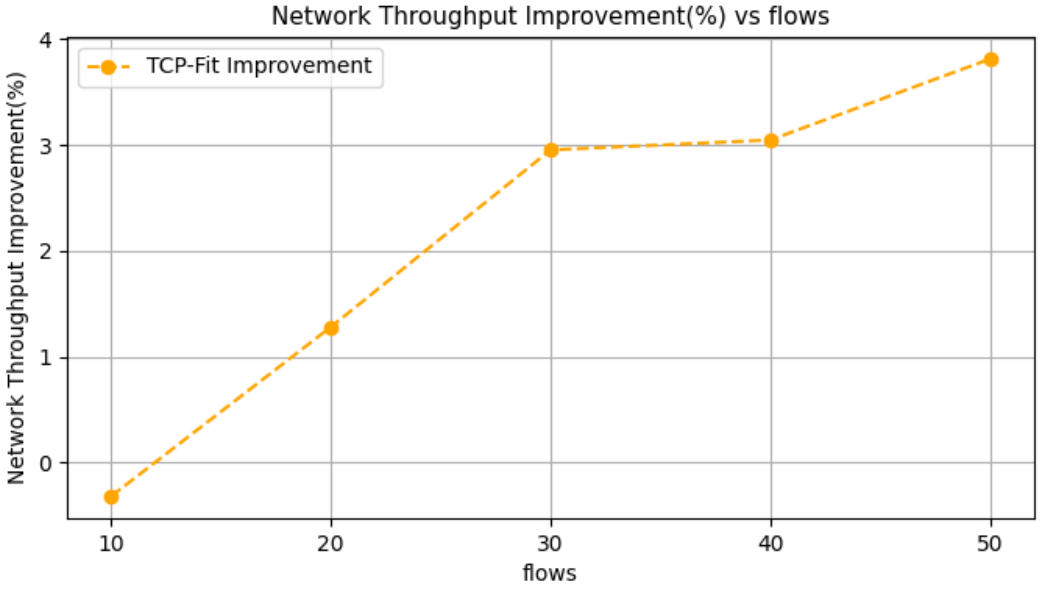
Observations

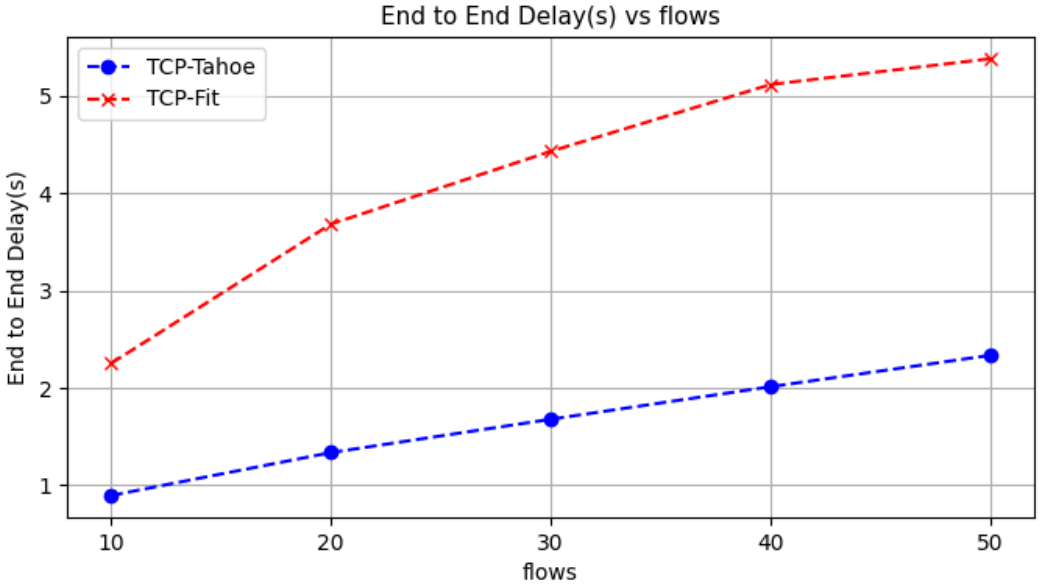
1. Again varying speed does seem to have very marginal difference on the metrics
2. TCP-Fit performs is poor overall

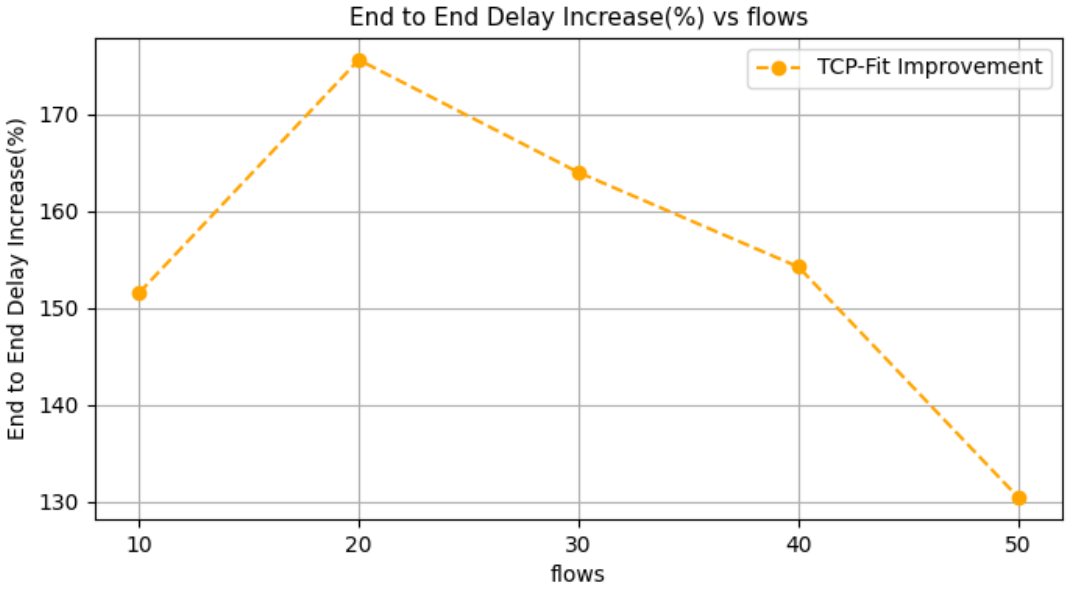
Wireless Random

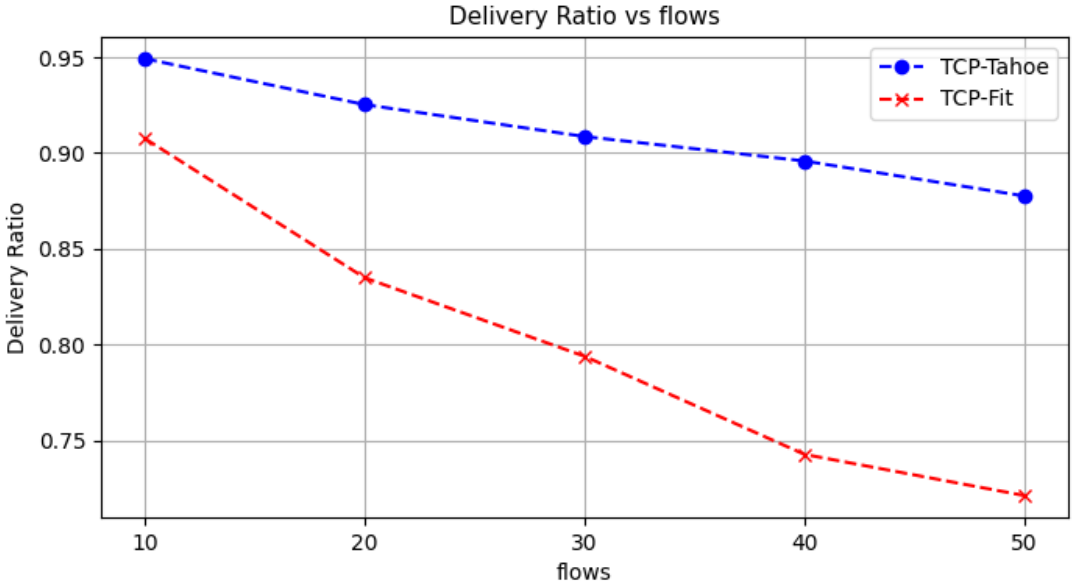
Varying flows

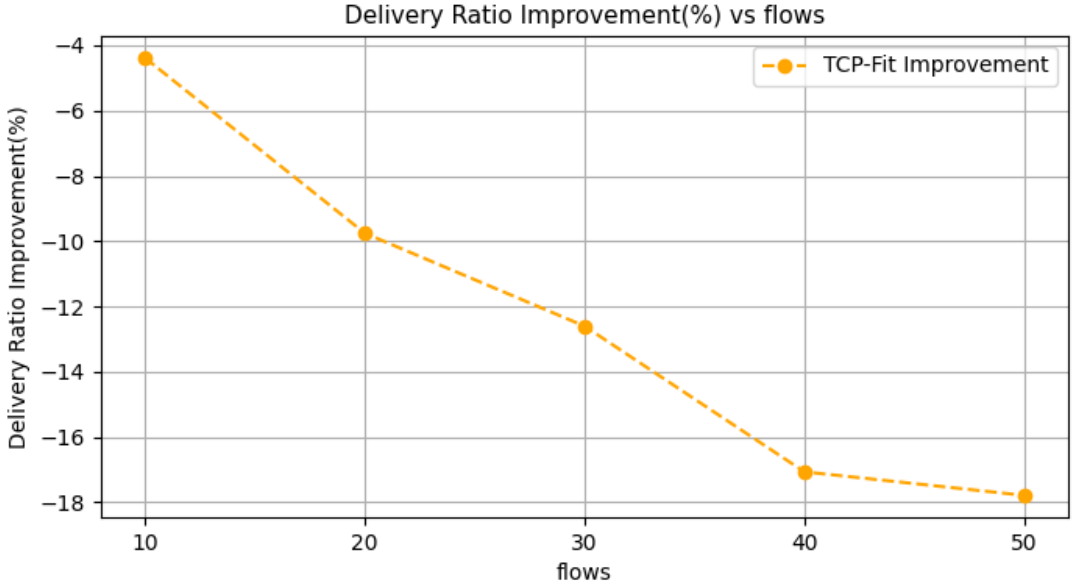


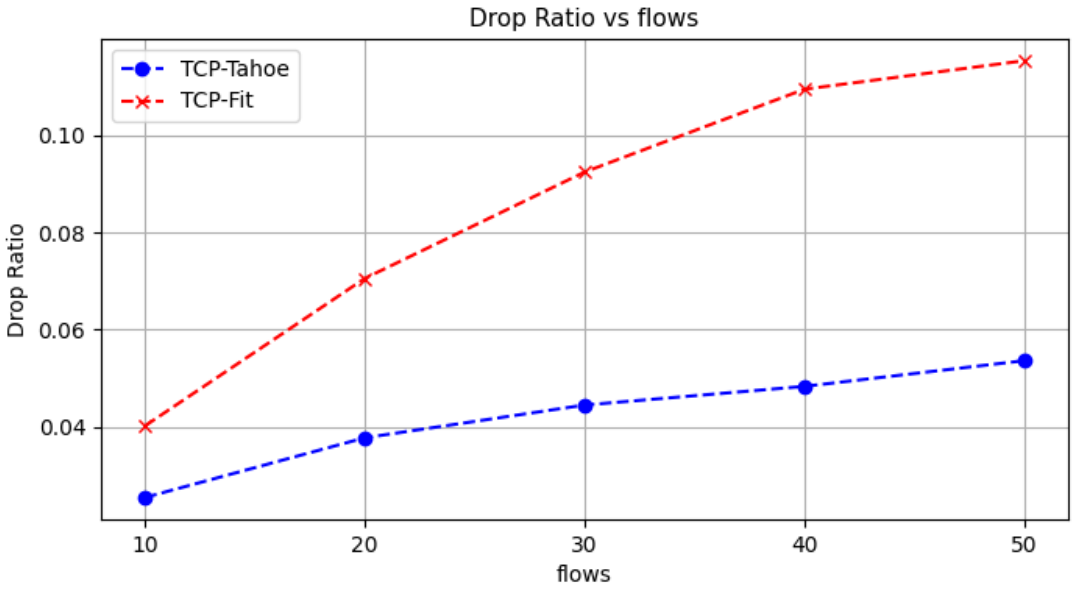


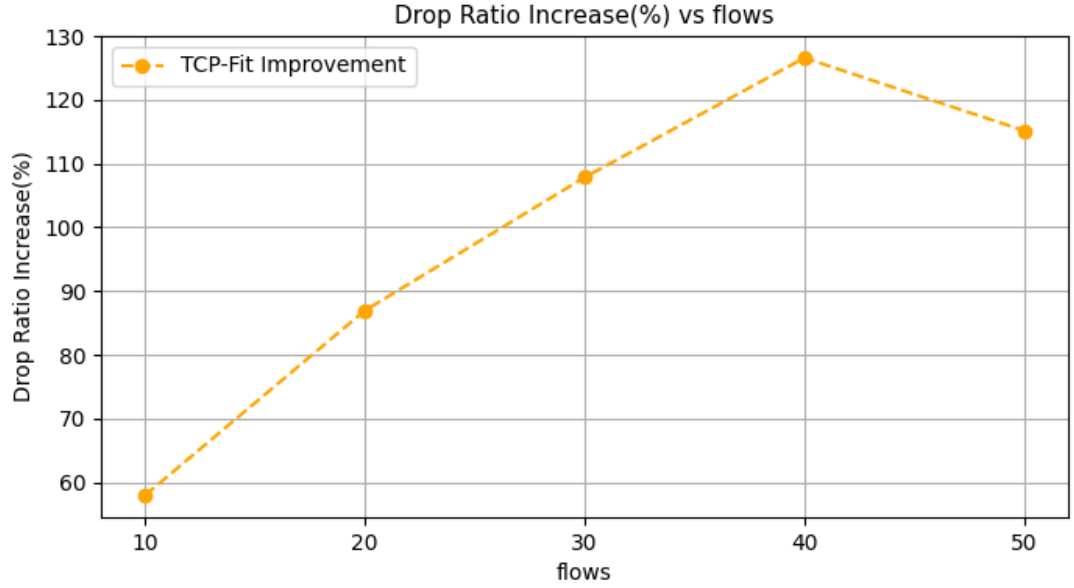

End to End Delay(s) vs flows
TCP-Tahoe
TCP-Fit
End to End Delay(s)
1
2
3
4
5
10
20
30
40
50
flows


End to End Delay Increase(%) vs flows
TCP-Fit Improvement
End to End Delay Increase(%)
130
140
150
160
170
10
20
30
40
50
flows


Delivery Ratio vs flows
TCP-Tahoe
TCP-Fit
Delivery Ratio
flows
0.95
0.90
0.85
0.80
0.75
10
20
30
40
50


Delivery Ratio Improvement(%) vs flows
TCP-Fit Improvement
Delivery Ratio Improvement(%)
flows
−4
−6
−8
−10
−12
−14
−16
−18
10
20
30
40
50

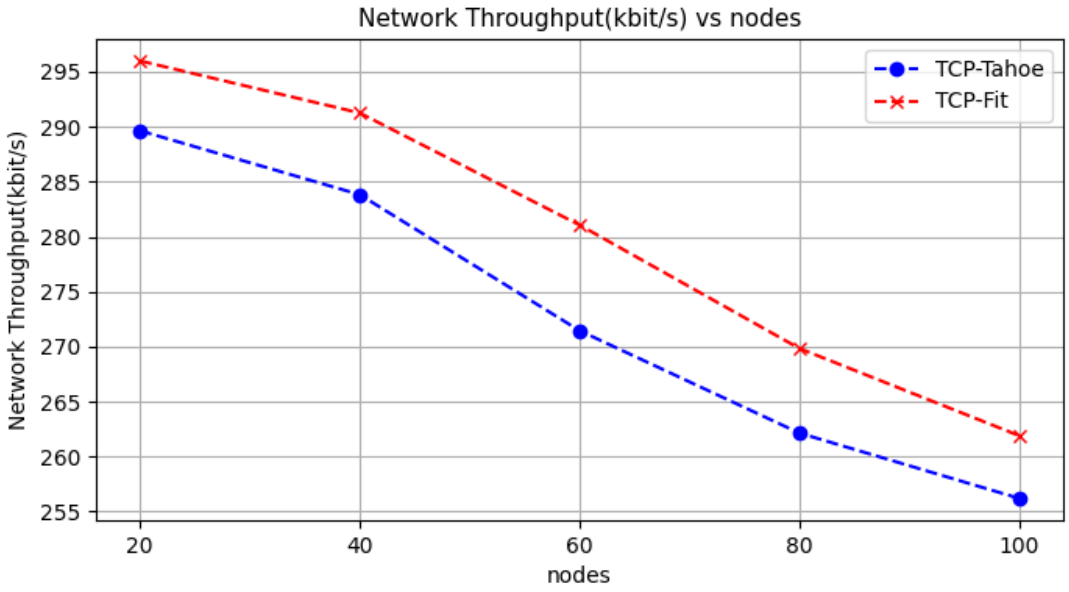


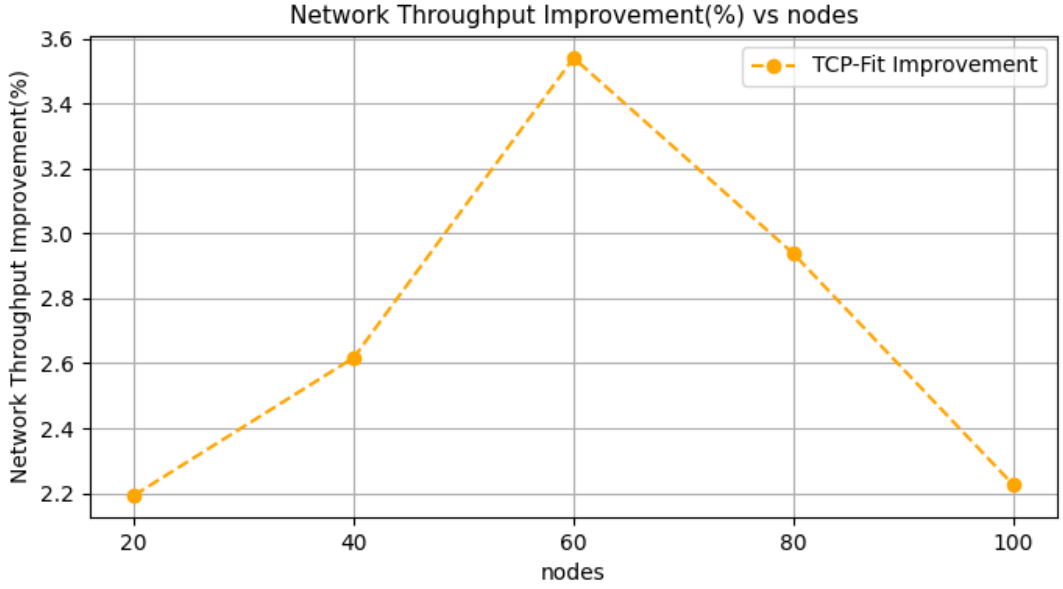


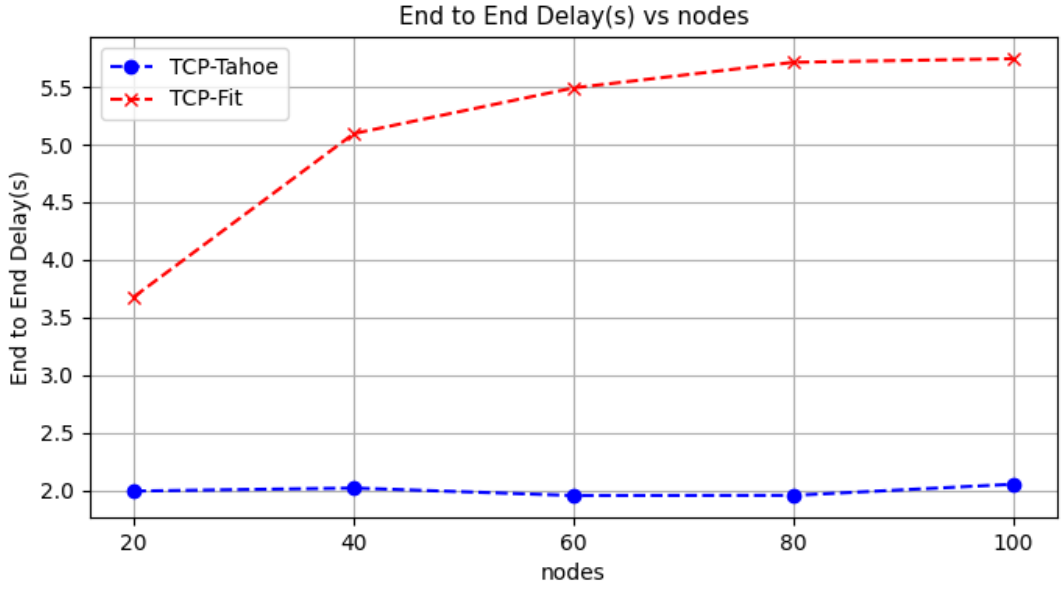
Observations

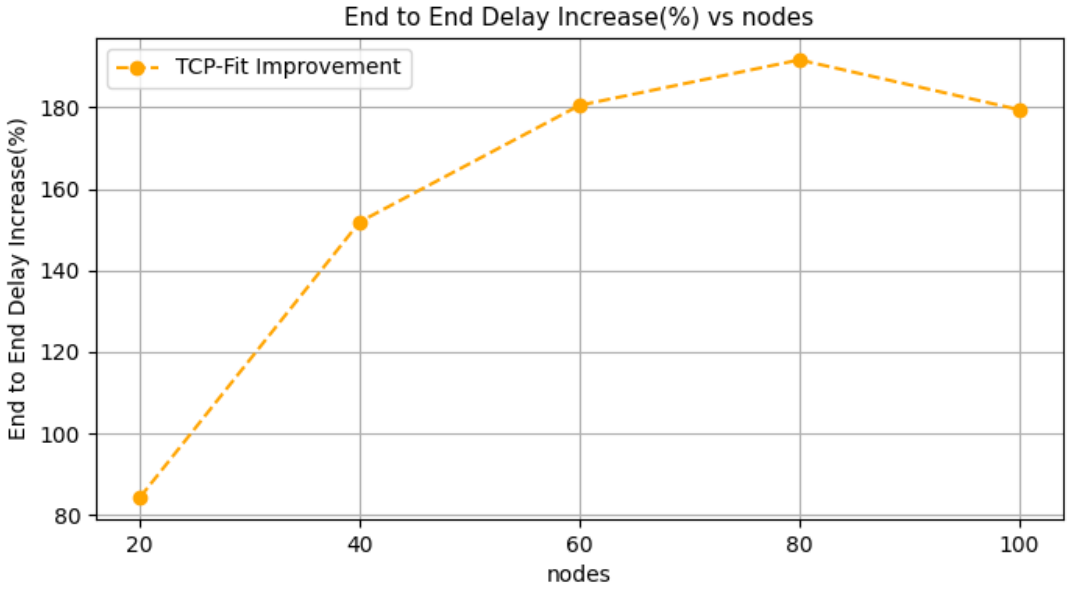
1. Throughput decreases with increasing flows indicating congested network. Slight improvement of throughput by TCP-Fit
2. Massive increase in end to end delay in TCP-Fit
3. Massive decrease of delivery ratio with TCP-Fit
4. Massive increase in drop rate as well with TCP-Fit

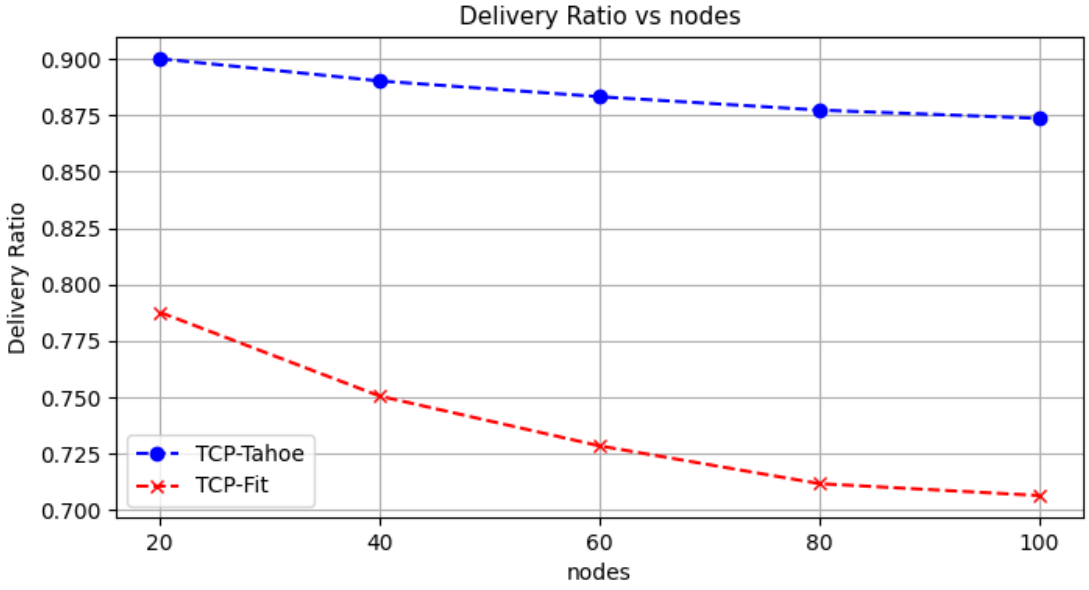
Varying nodes

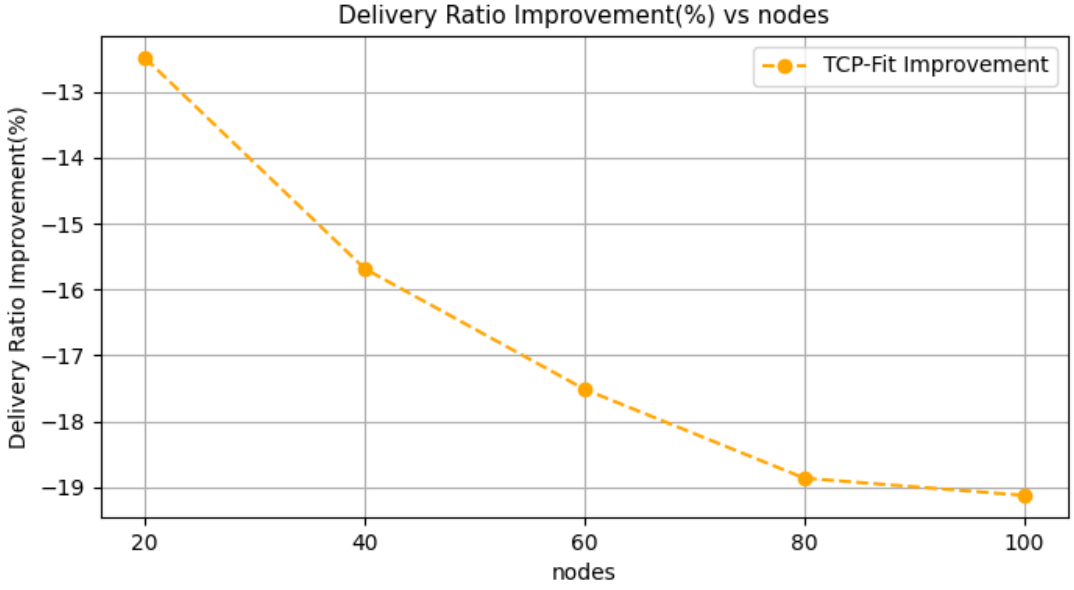


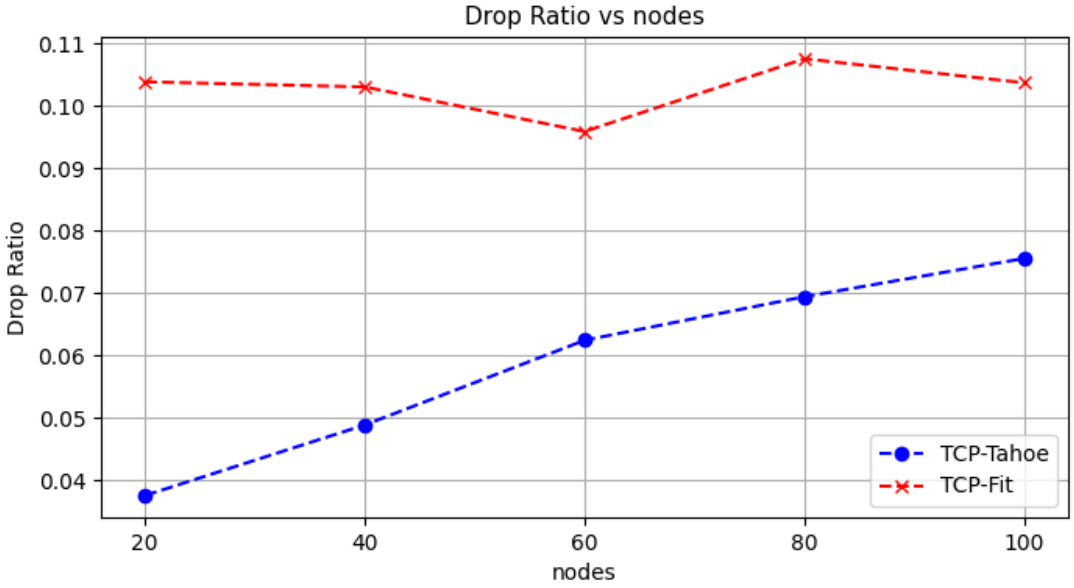


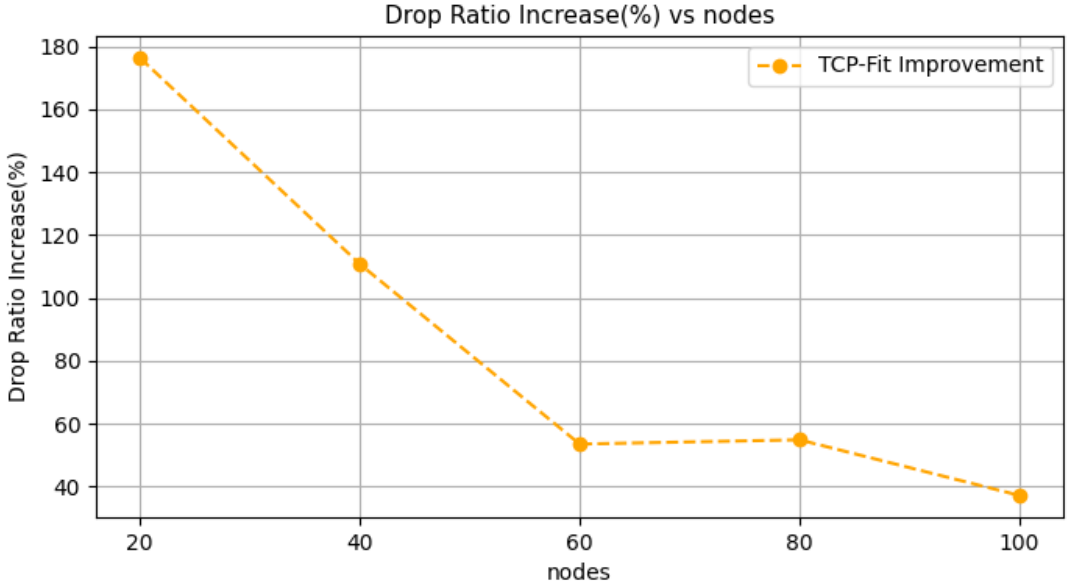

End to End Delay(s) vs nodes
TCP-Tahoe
TCP-Fit
End to End Delay(s)
5.5
5.0
4.5
4.0
3.5
3.0
2.5
2.0
20
40
60
80
100
nodes

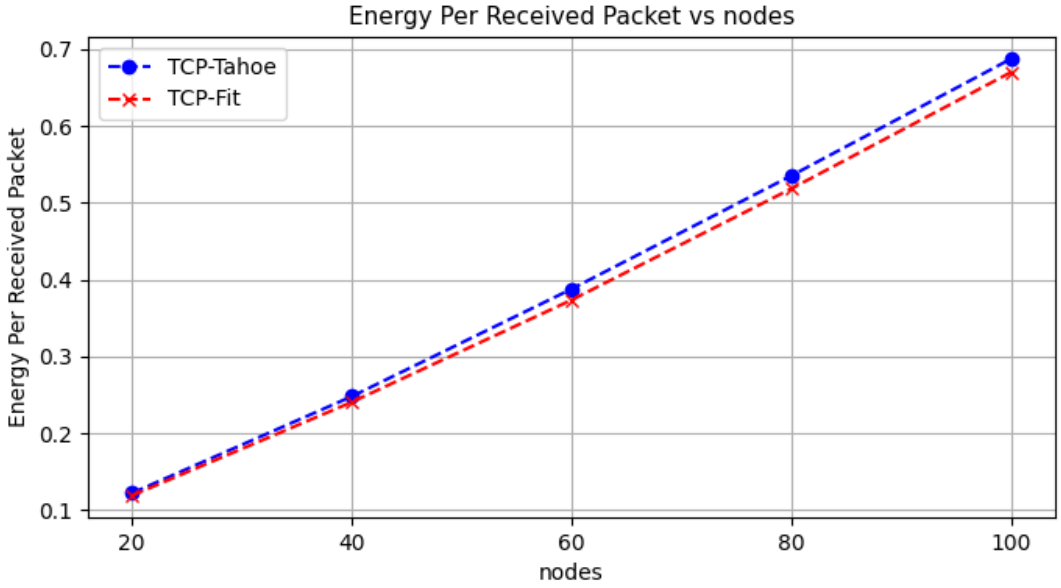

End to End Delay Increase(%) vs nodes
TCP-Fit Improvement
End to End Delay Increase(%)
180
160
140
120
100
80
20
40
60
80
100
nodes

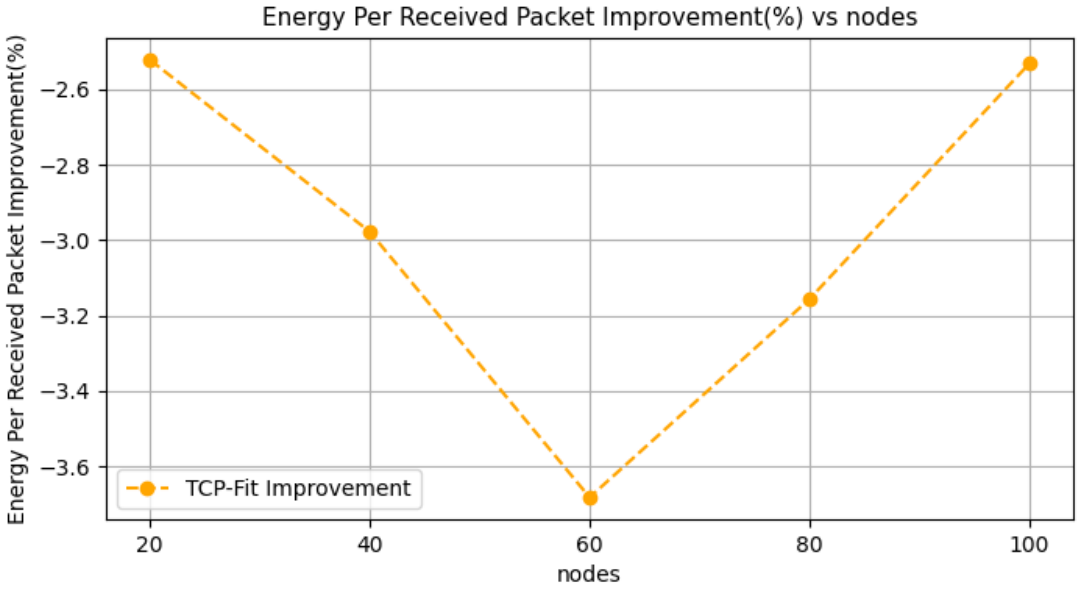

Delivery Ratio vs nodes
Delivery Ratio
0.900
0.875
0.850
0.825
0.800
0.775
0.750
0.725
0.700
TCP-Tahoe
TCP-Fit
20
40
60
80
100
nodes


Delivery Ratio Improvement(%) vs nodes
Delivery Ratio Improvement(%)
−13
−14
−15
−16
−17
−18
−19
TCP-Fit Improvement
20
40
60
80
100
nodes


Drop Ratio vs nodes
Drop Ratio
nodes
0.11
0.10
0.09
0.08
0.07
0.06
0.05
0.04
20
40
60
80
100
TCP-Tahoe
TCP-Fit


Drop Ratio Increase(%) vs nodes
Drop Ratio Increase(%)
nodes
180
160
140
120
100
80
60
40
20
40
60
80
100
TCP-Fit Improvement

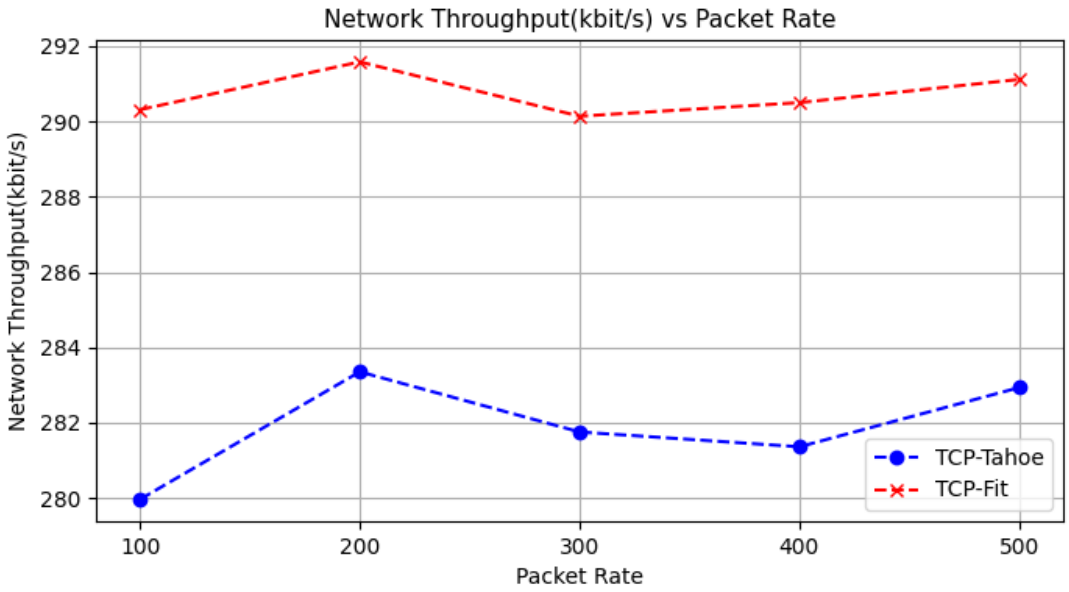


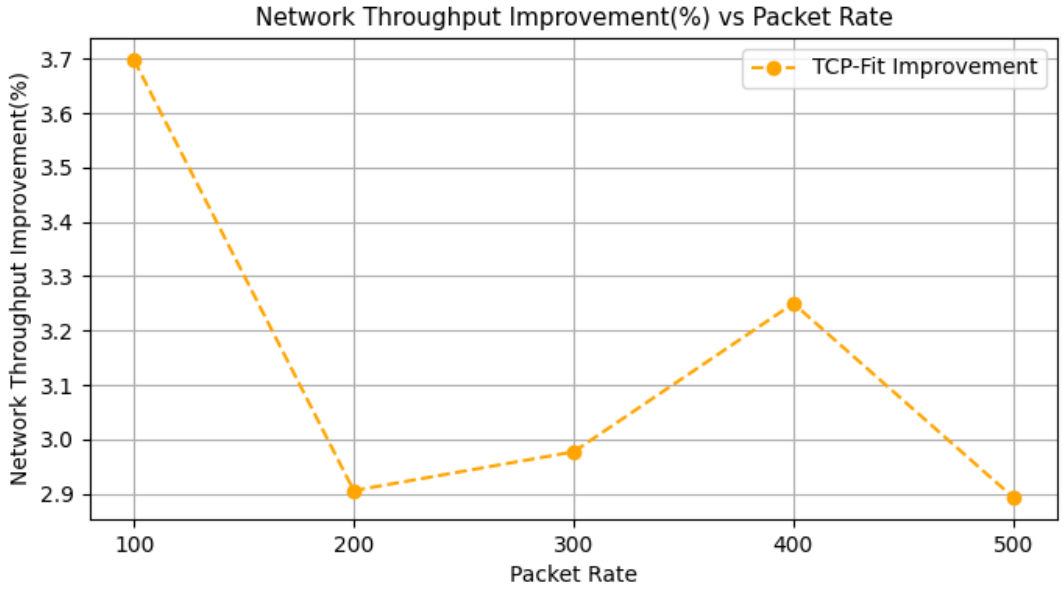


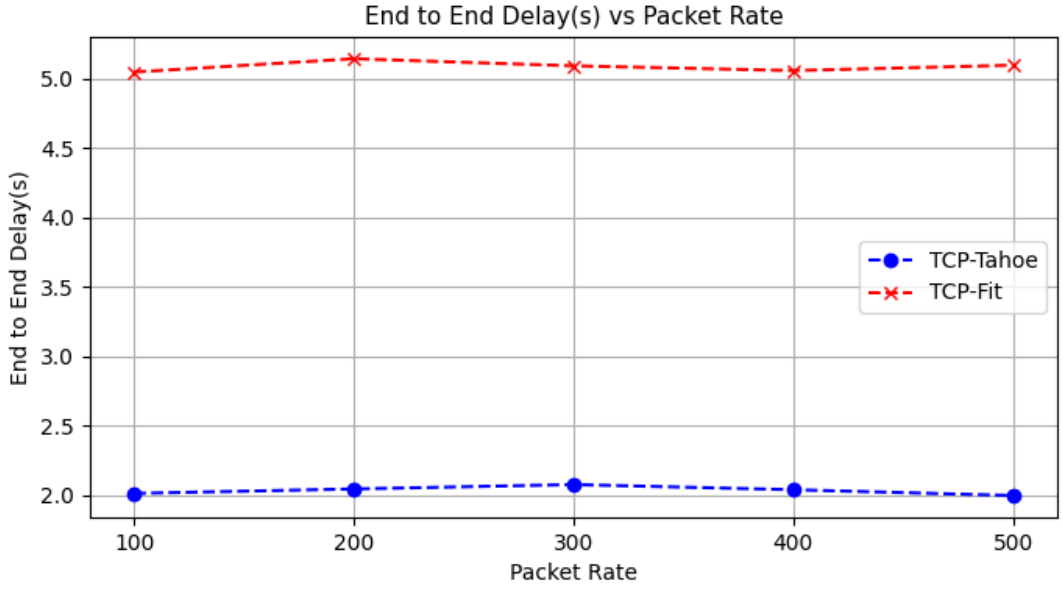
Observations

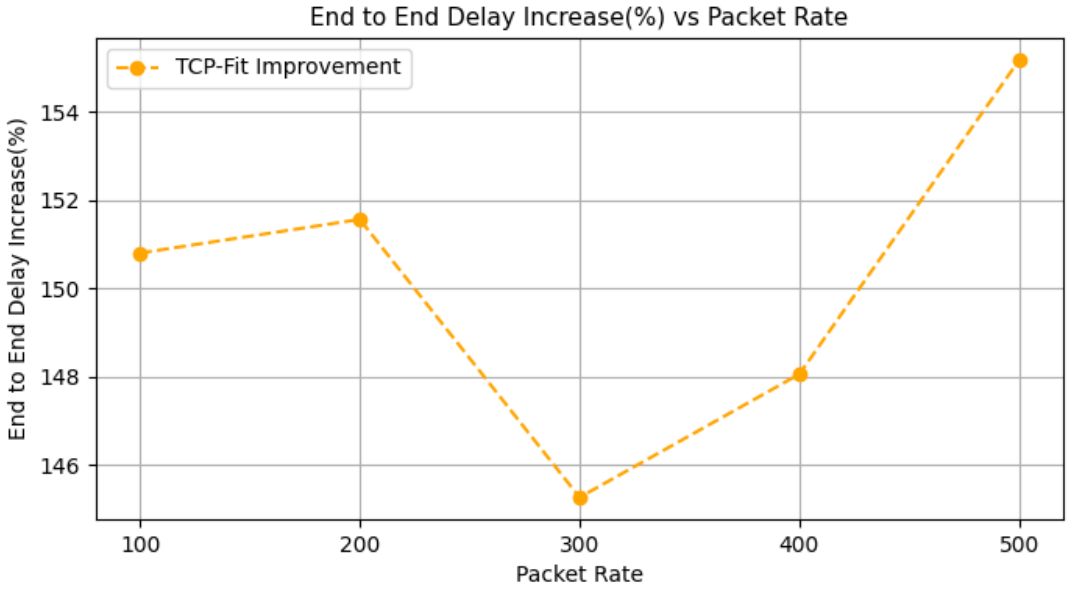
1. Throughput drops with increasing node numbers. The reasoning could be similar to WIreless-bottleneck topology
2. Massive increase in end to end delay for TCP-Fit algorithm
3. Delivery ratio worsens, and drop ratio increases for TCP-Fit algorithm

Varying packet rate

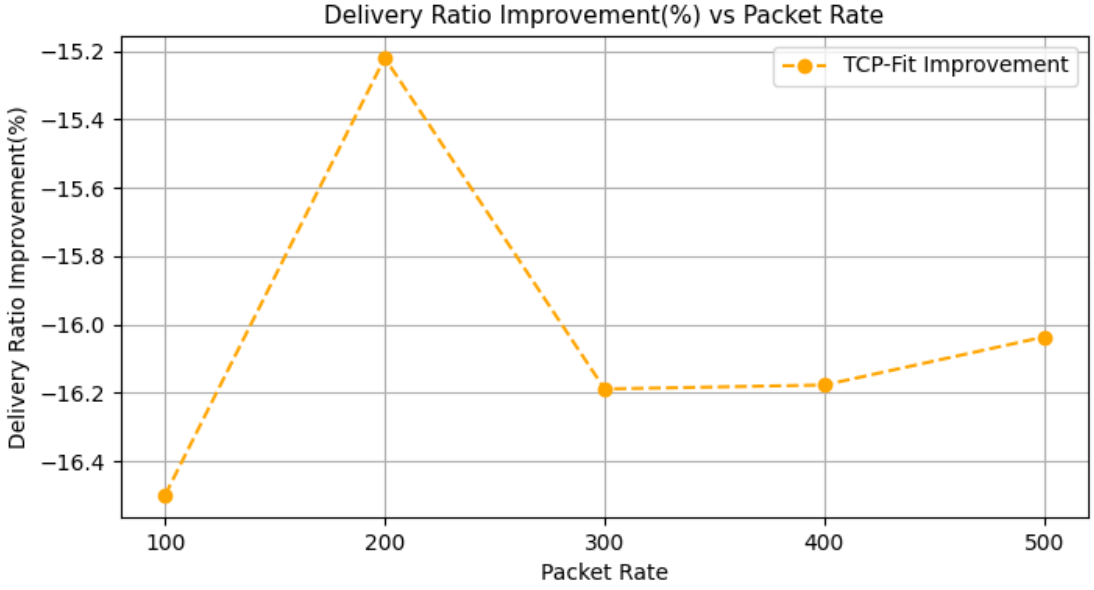


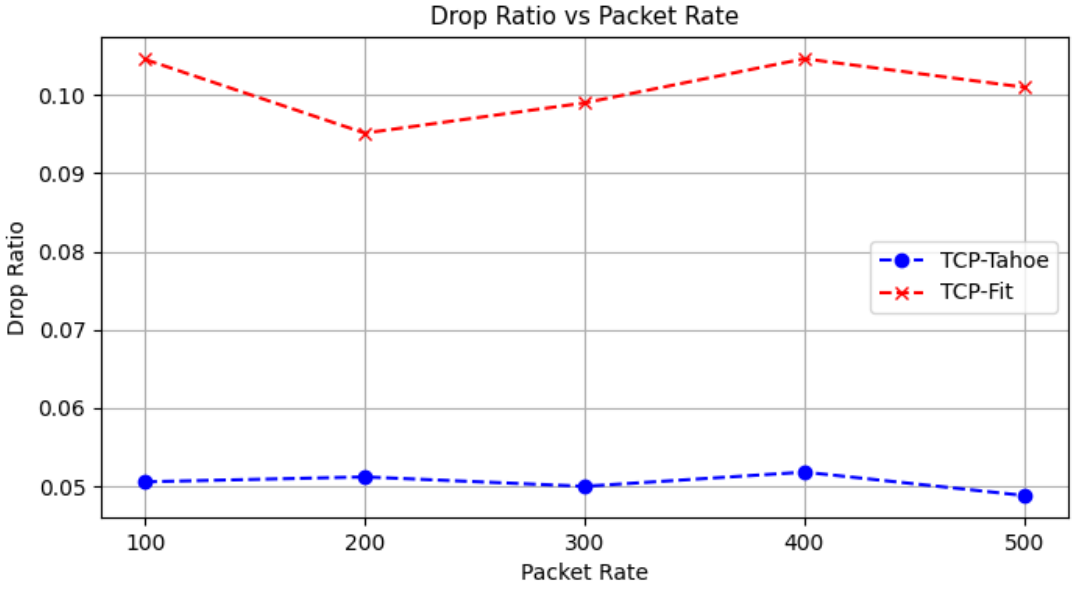


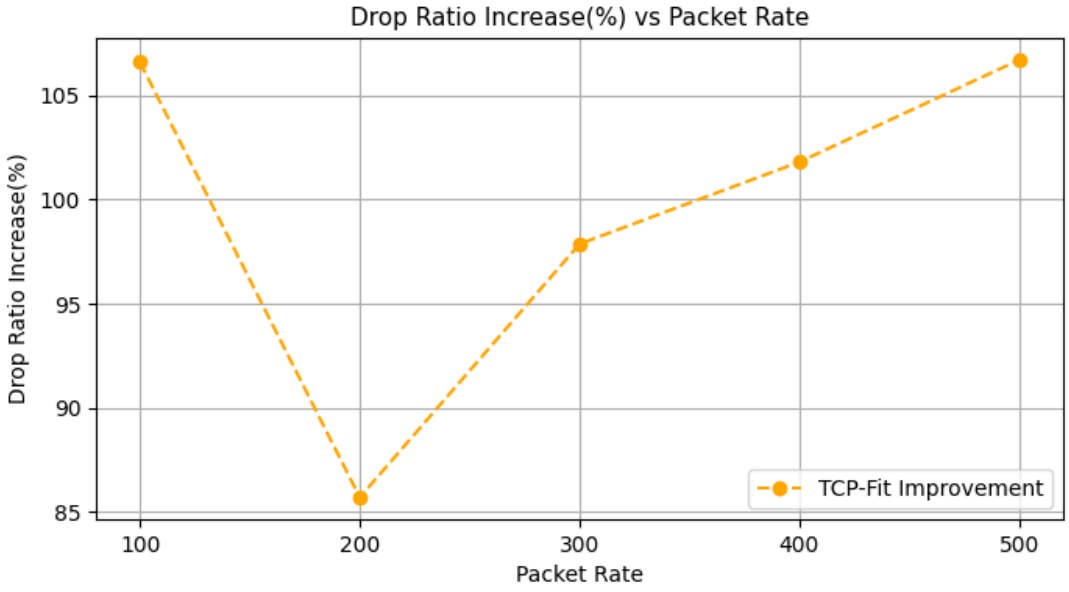

End to End Delay(s) vs Packet Rate
End to End Delay(s)
5.0
4.5
4.0
3.5
3.0
2.5
2.0
100
200
300
400
500
Packet Rate
TCP-Tahoe
TCP-Fit

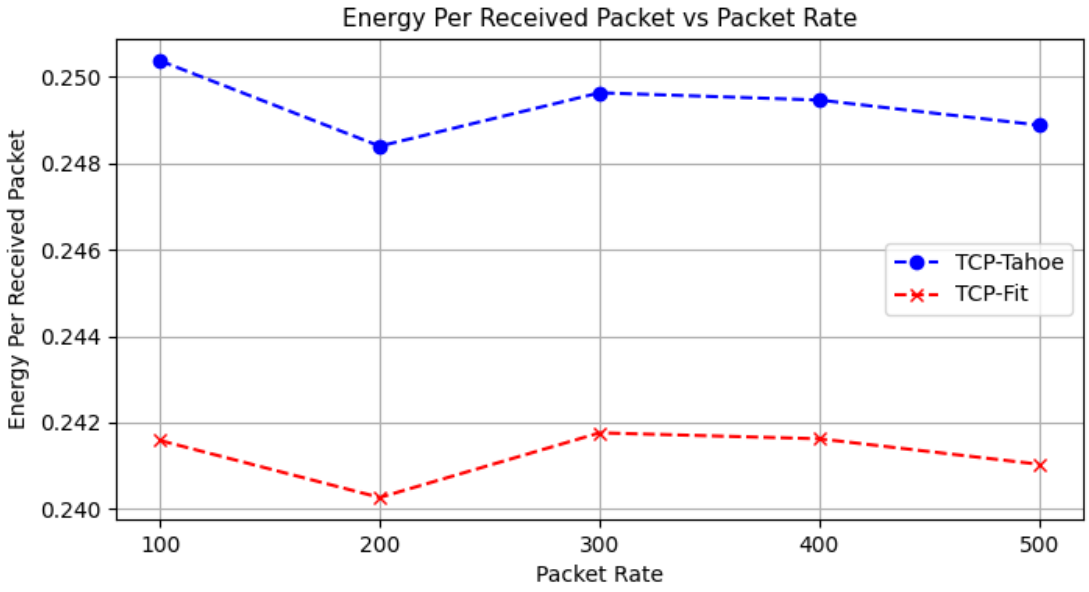

End to End Delay Increase(%) vs Packet Rate
End to End Delay Increase(%)
154
152
150
148
146
100
200
300
400
500
Packet Rate
TCP-Fit Improvement

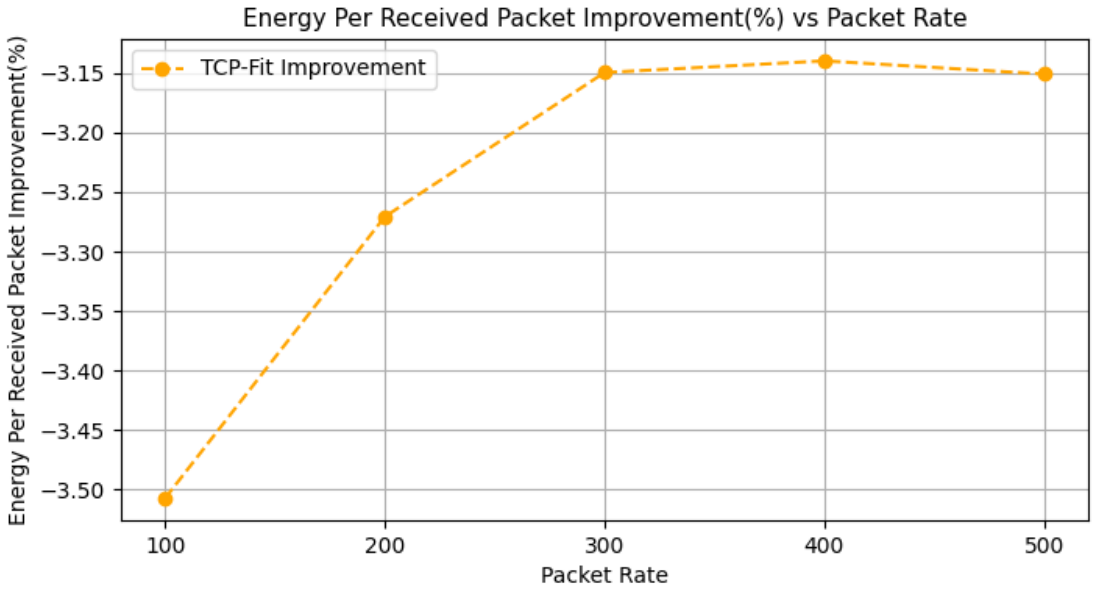

Delivery Ratio vs Packet Rate
Delivery Ratio
Packet Rate
0.90
0.88
0.86
0.84
0.82
0.80
0.78
0.76
0.74
100
200
300
400
500
TCP-Tahoe
TCP-Fit


Delivery Ratio Improvement(%) vs Packet Rate
Delivery Ratio Improvement(%)
Packet Rate
−15.2
−15.4
−15.6
−15.8
−16.0
−16.2
−16.4
100
200
300
400
500
TCP-Fit Improvement


Drop Ratio vs Packet Rate
Drop Ratio
0.10
0.09
0.08
0.07
0.06
0.05
100
200
300
400
500
Packet Rate
TCP-Tahoe
TCP-Fit


Drop Ratio Increase(%) vs Packet Rate
Drop Ratio Increase(%)
105
100
95
90
85
100
200
300
400
500
Packet Rate
TCP-Fit Improvement

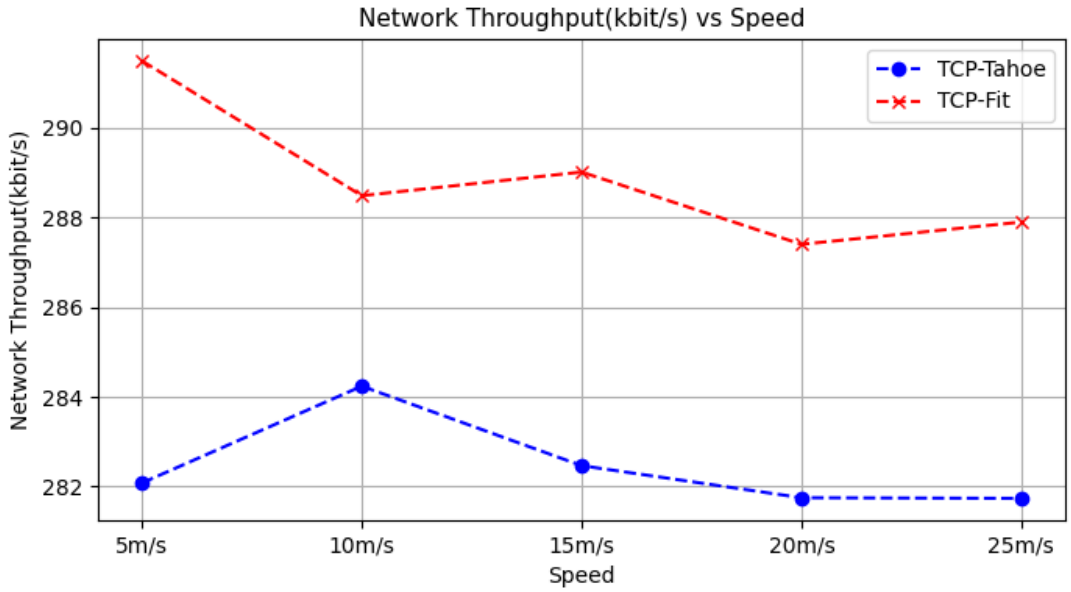


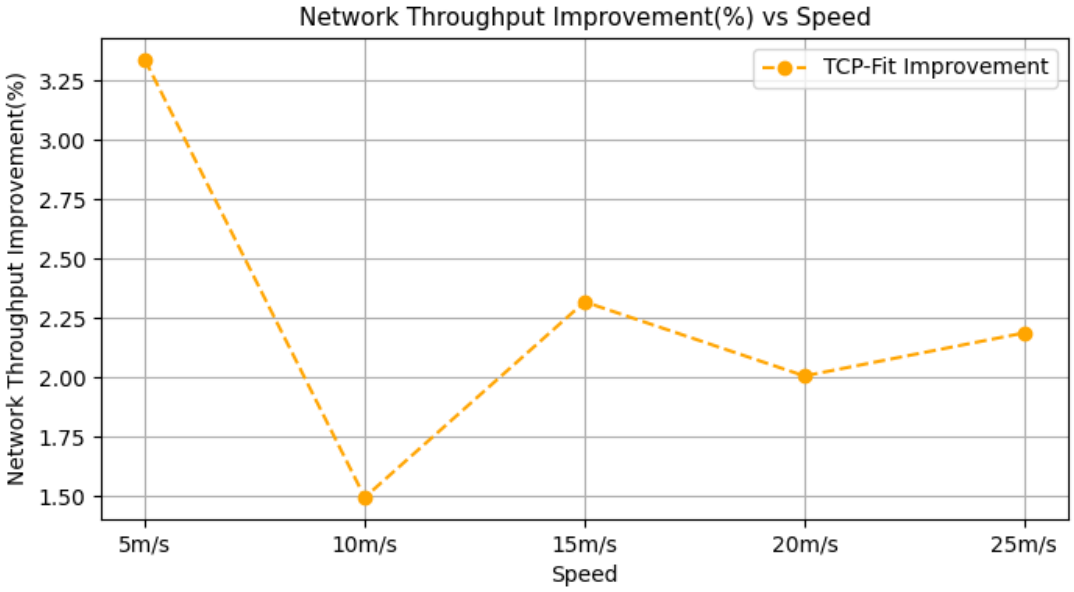


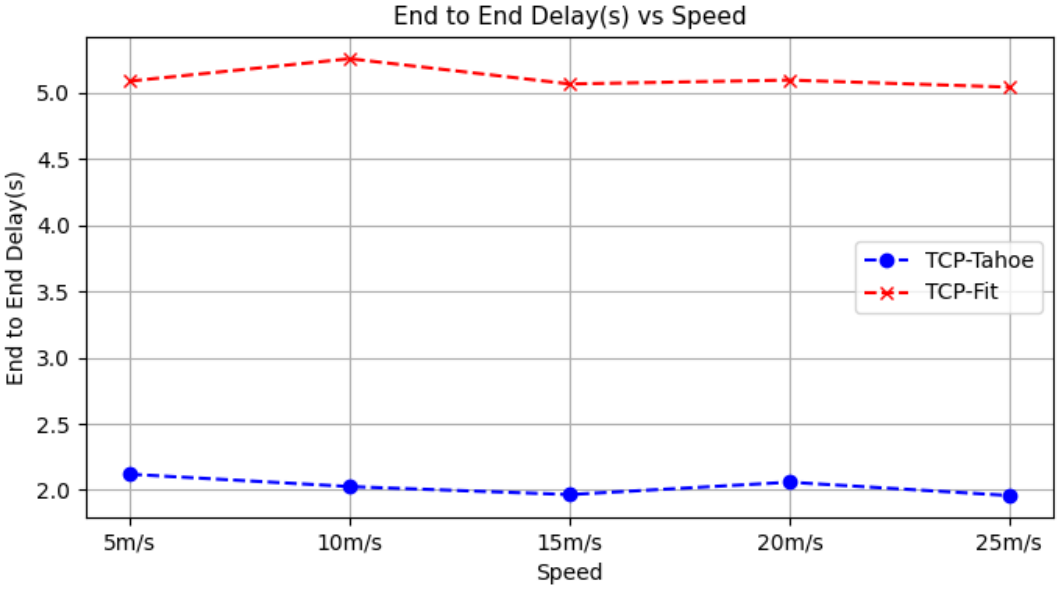
Observations

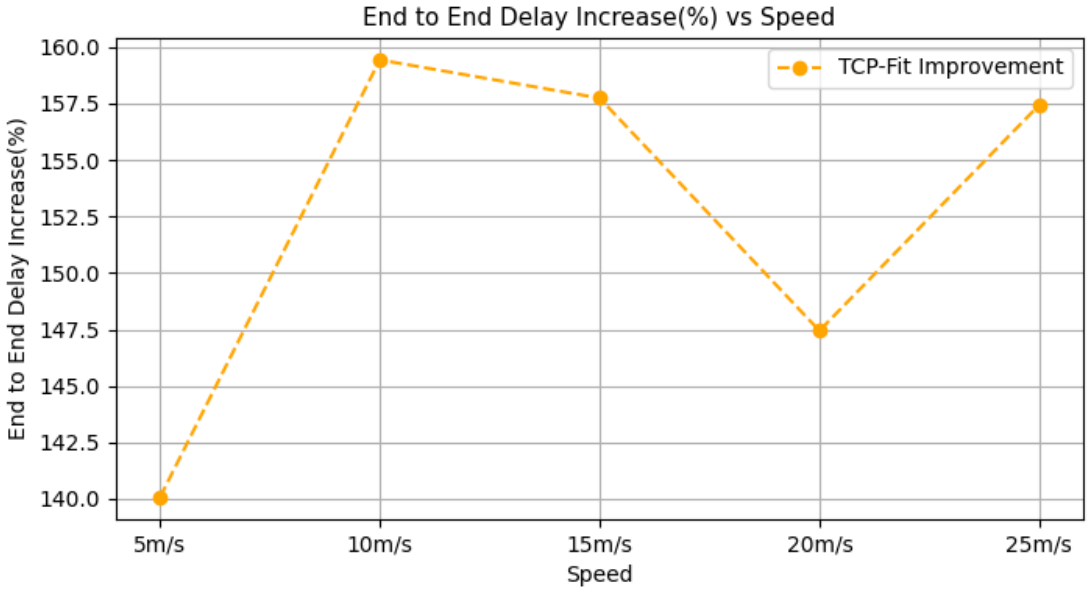
1. Again, metrics do not show any notable trend with change in packet rate of the chosen values
2. Again, quite worse performance of TCP-Fit in case of end to end delay, delivery ratio and drop ratio
3. Energy per packet is lower for TCP-Fit due to increased throughput

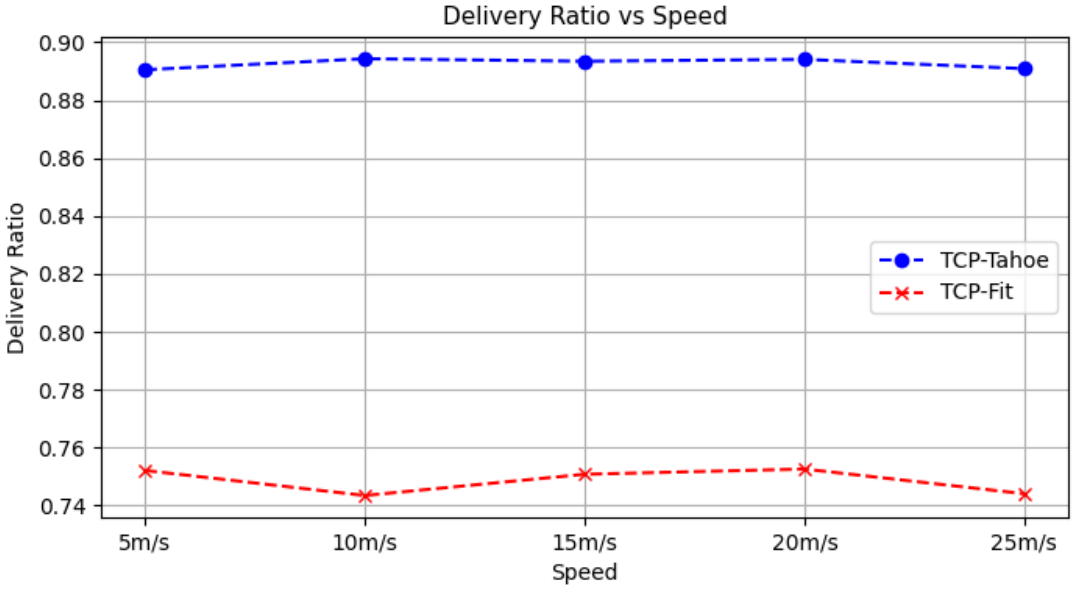
Varying speed

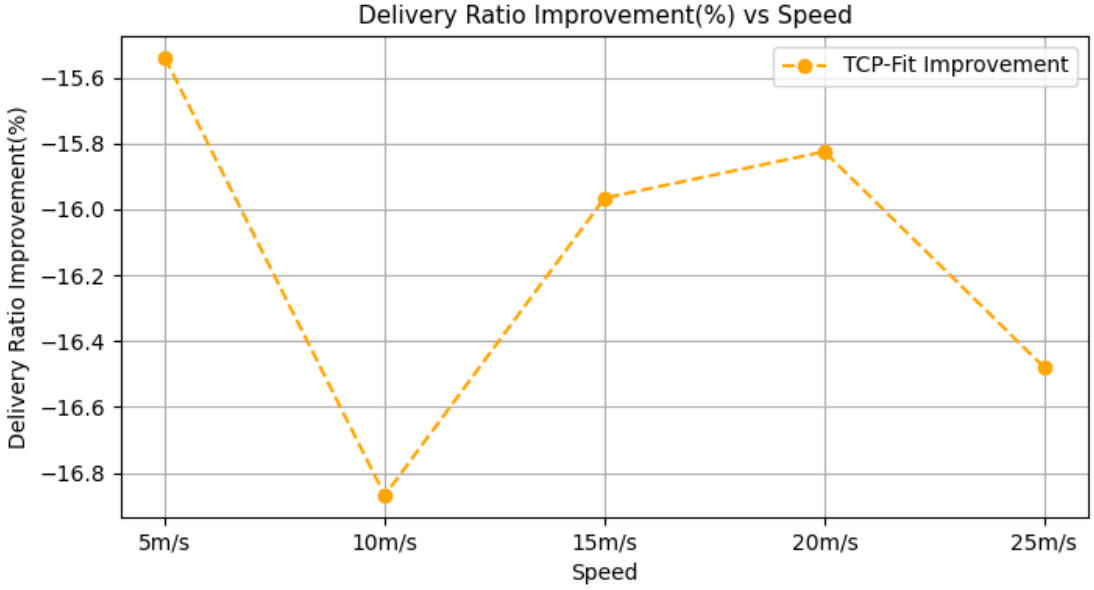


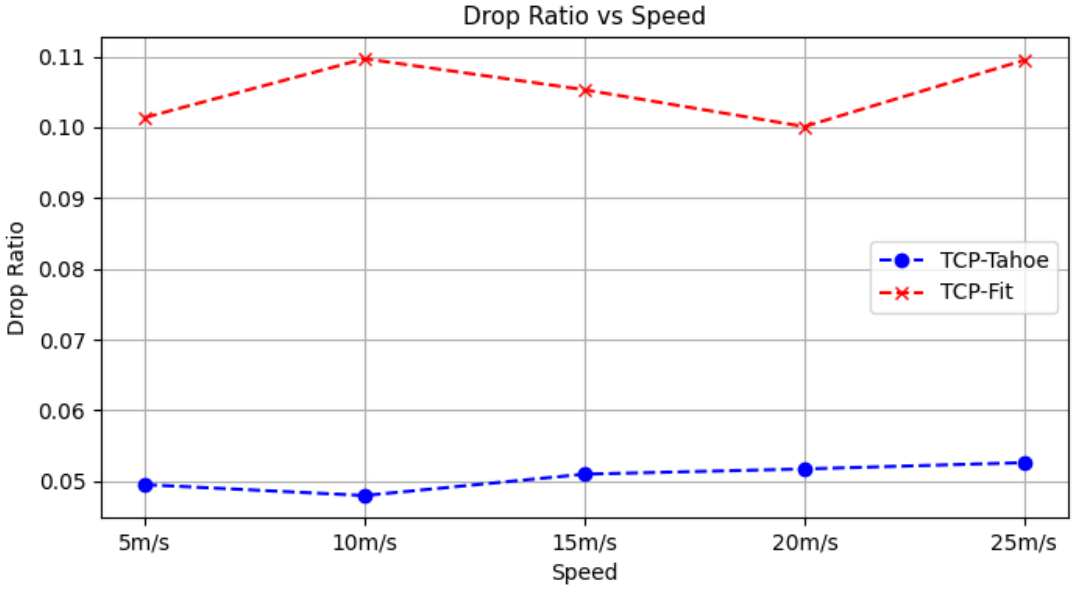


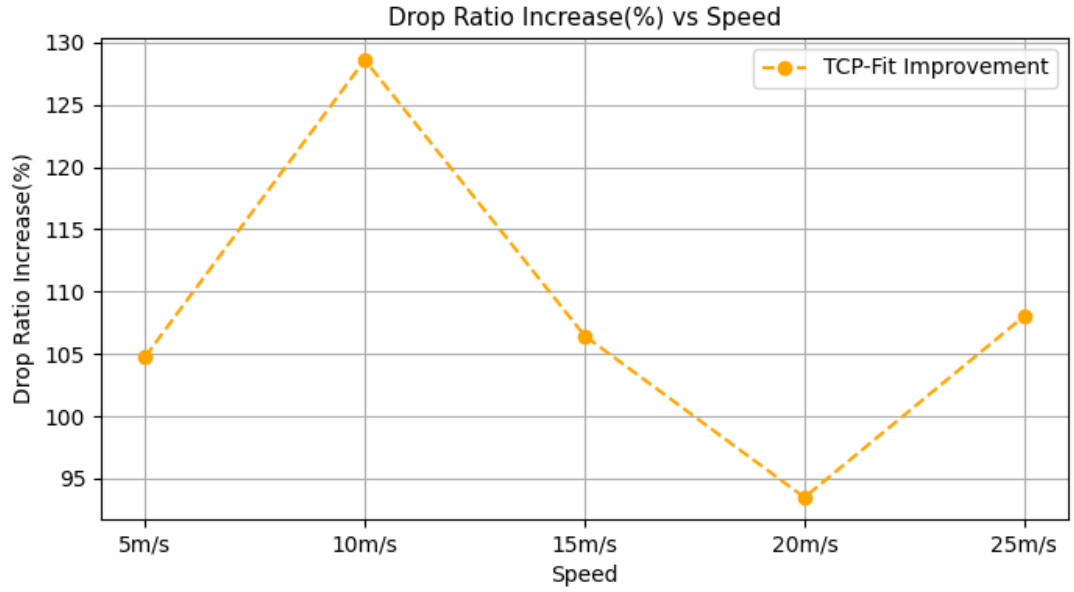

End to End Delay(s) vs Speed
End to End Delay(s)
5.0
4.5
4.0
3.5
3.0
2.5
2.0
TCP-Tahoe
TCP-Fit
5m/s
10m/s
15m/s
20m/s
25m/s
Speed

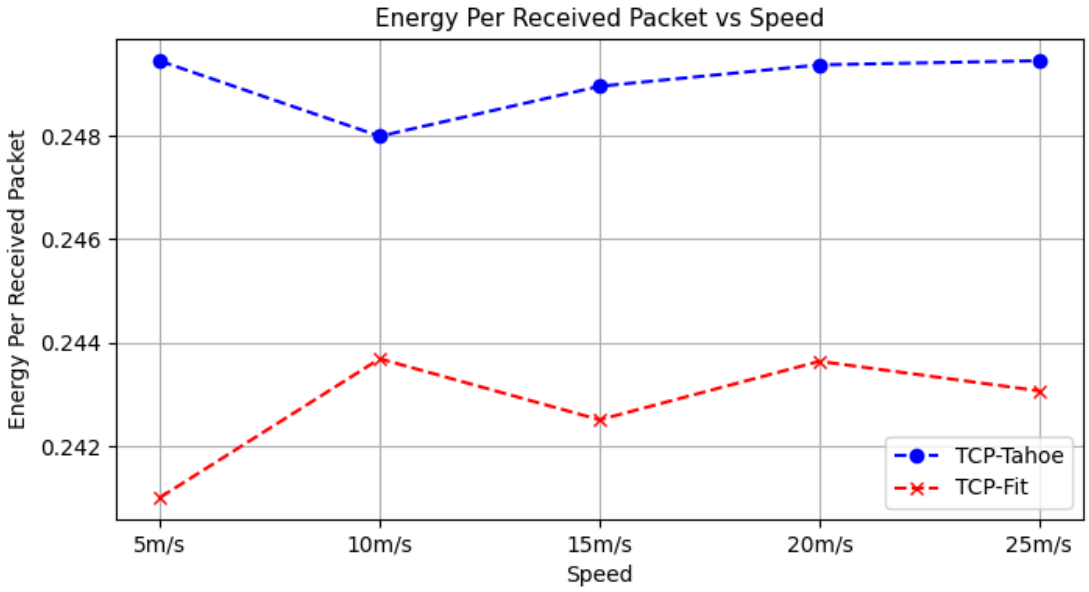

End to End Delay Increase(%) vs Speed
End to End Delay Increase(%)
160.0
157.5
155.0
152.5
150.0
147.5
145.0
142.5
140.0
TCP-Fit Improvement
5m/s
10m/s
15m/s
20m/s
25m/s
Speed

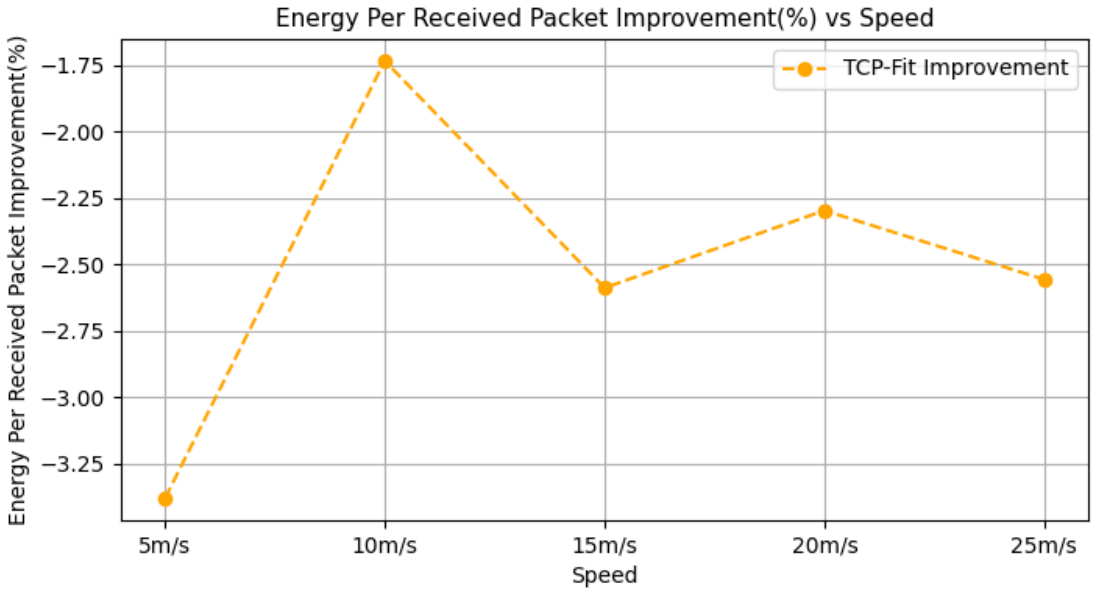

Delivery Ratio vs Speed
Delivery Ratio
0.90
0.88
0.86
0.84
0.82
0.80
0.78
0.76
0.74
TCP-Tahoe
TCP-Fit
5m/s
10m/s
15m/s
20m/s
25m/s
Speed


Delivery Ratio Improvement(%) vs Speed
Delivery Ratio Improvement(%)
−15.6
−15.8
−16.0
−16.2
−16.4
−16.6
−16.8
TCP-Fit Improvement
5m/s
10m/s
15m/s
20m/s
25m/s
Speed


Drop Ratio vs Speed
Drop Ratio
0.11
0.10
0.09
0.08
0.07
0.06
0.05
TCP-Tahoe
TCP-Fit
5m/s
10m/s
15m/s
20m/s
25m/s
Speed


Drop Ratio Increase(%) vs Speed
Drop Ratio Increase(%)
130
125
120
115
110
105
100
95
TCP-Fit Improvement
5m/s
10m/s
15m/s
20m/s
25m/s
Speed



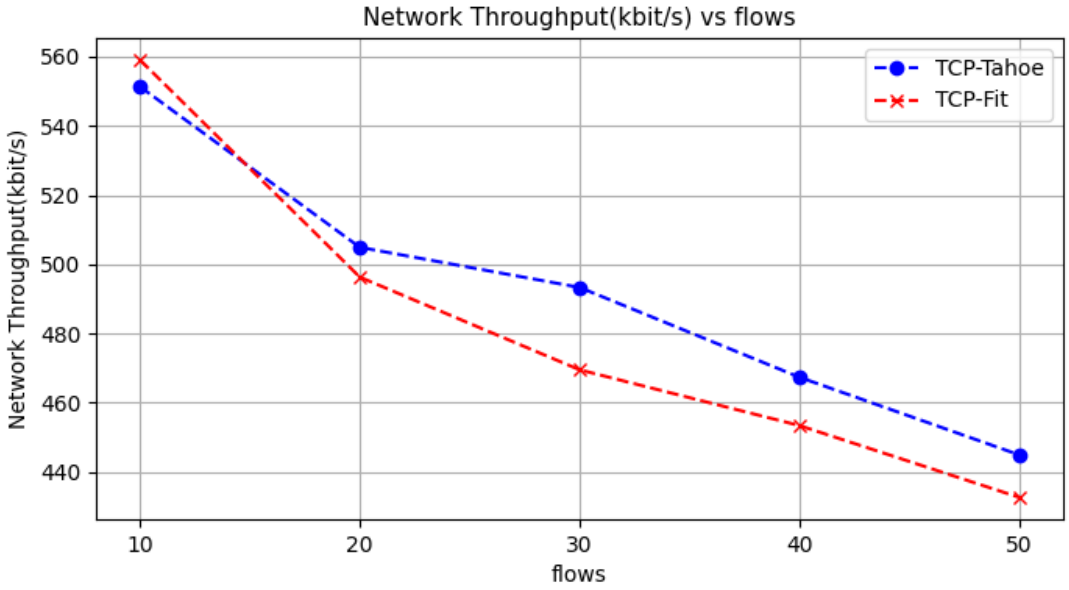


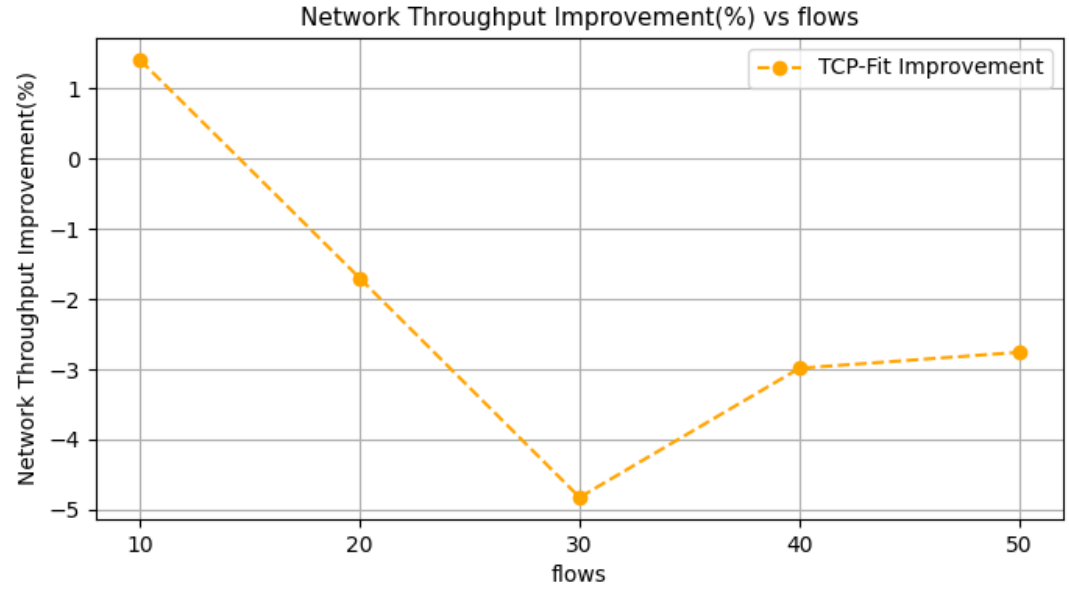
Observations

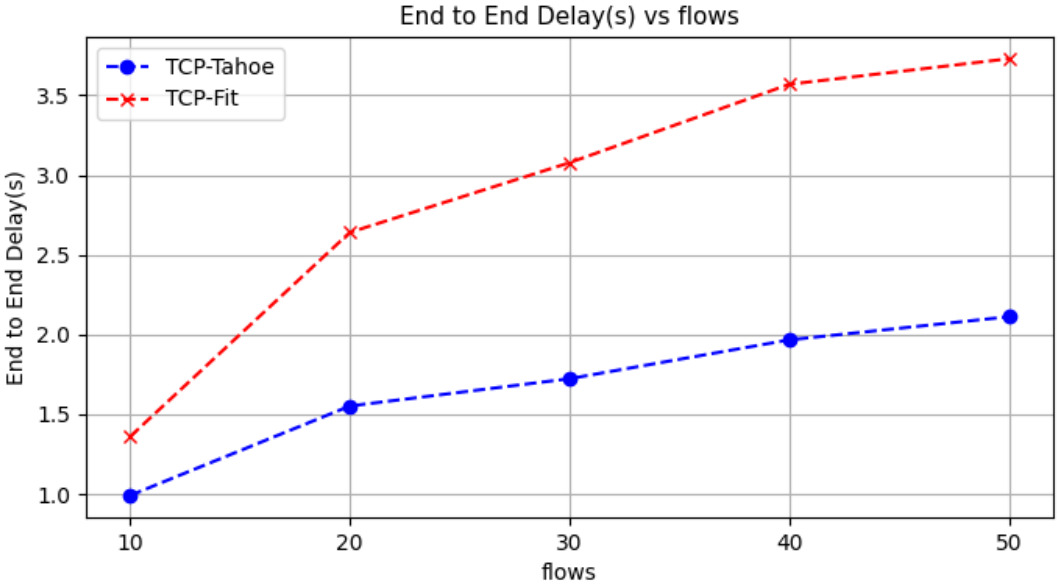
1. SImilar observations as previous topology

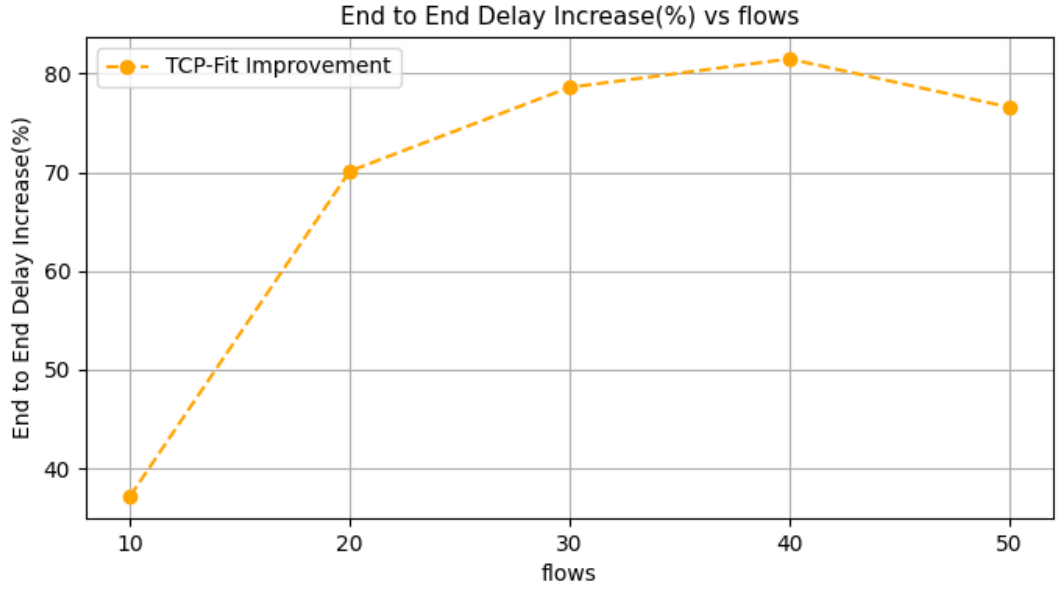
Wireless-Grid

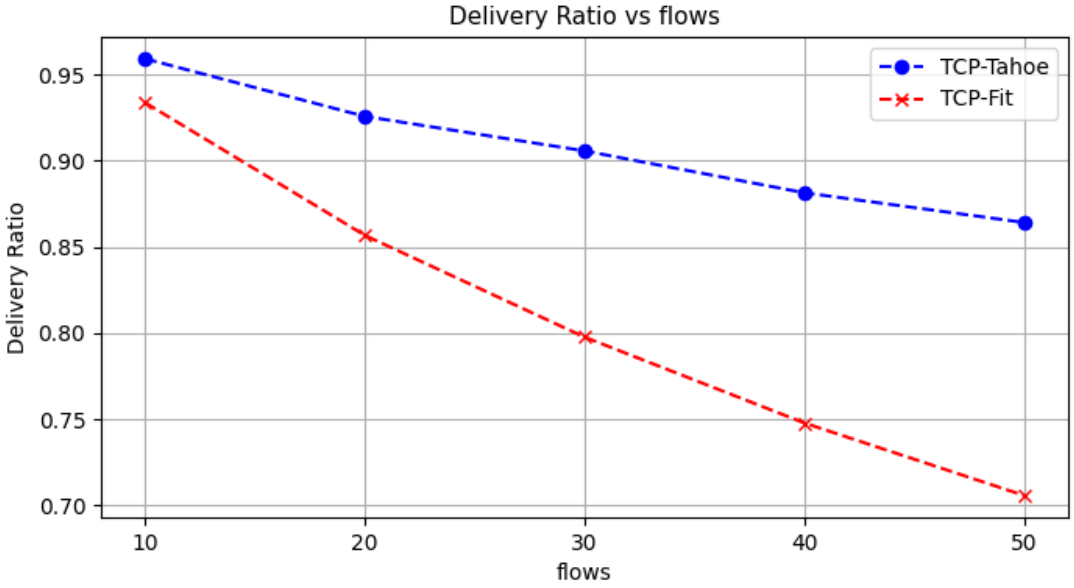
Varying flows



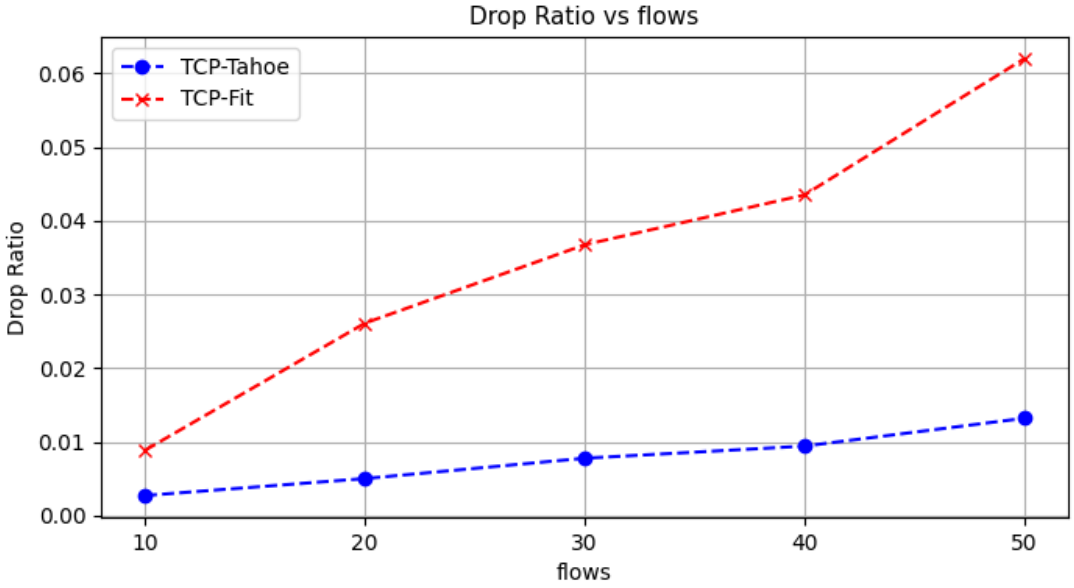


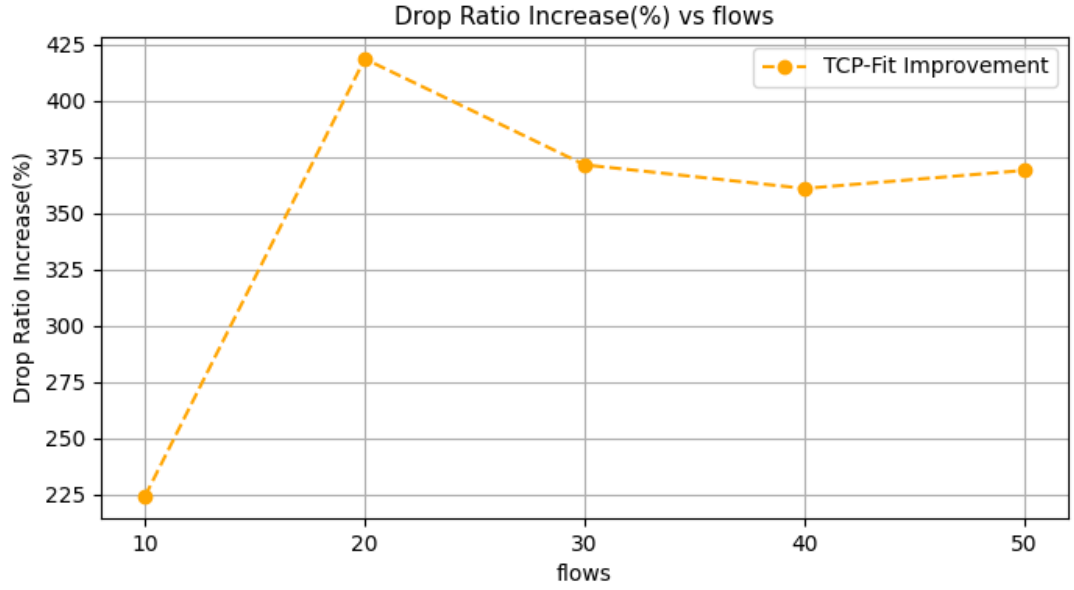

End to End Delay(s) vs flows
TCP-Tahoe
TCP-Fit
End to End Delay(s)
1.0
1.5
2.0
2.5
3.0
3.5
10
20
30
40
50
flows

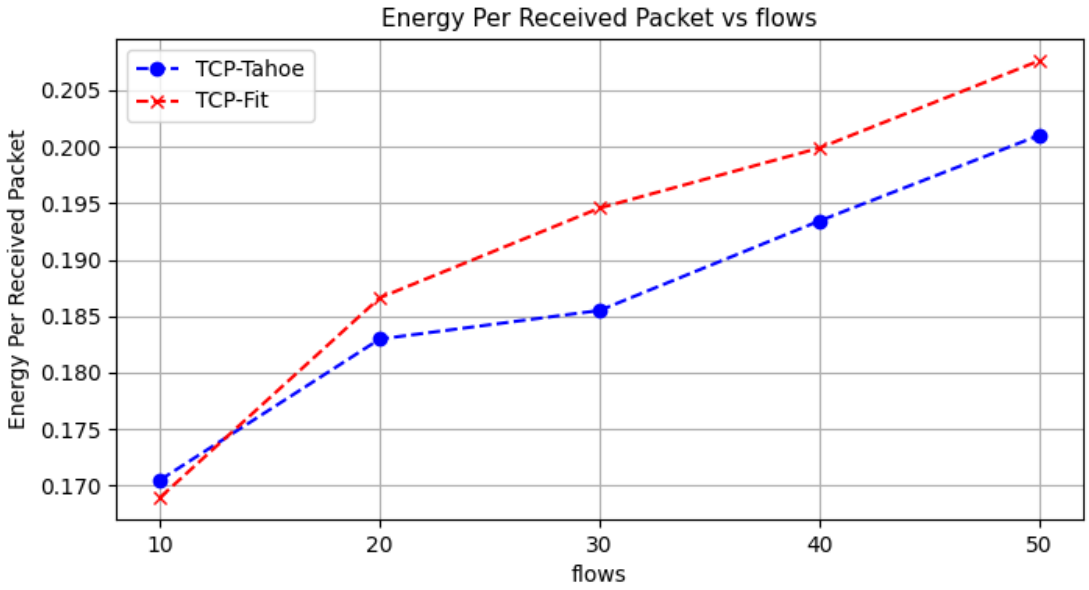

End to End Delay Increase(%) vs flows
TCP-Fit Improvement
End to End Delay Increase(%)
40
50
60
70
80
10
20
30
40
50
flows

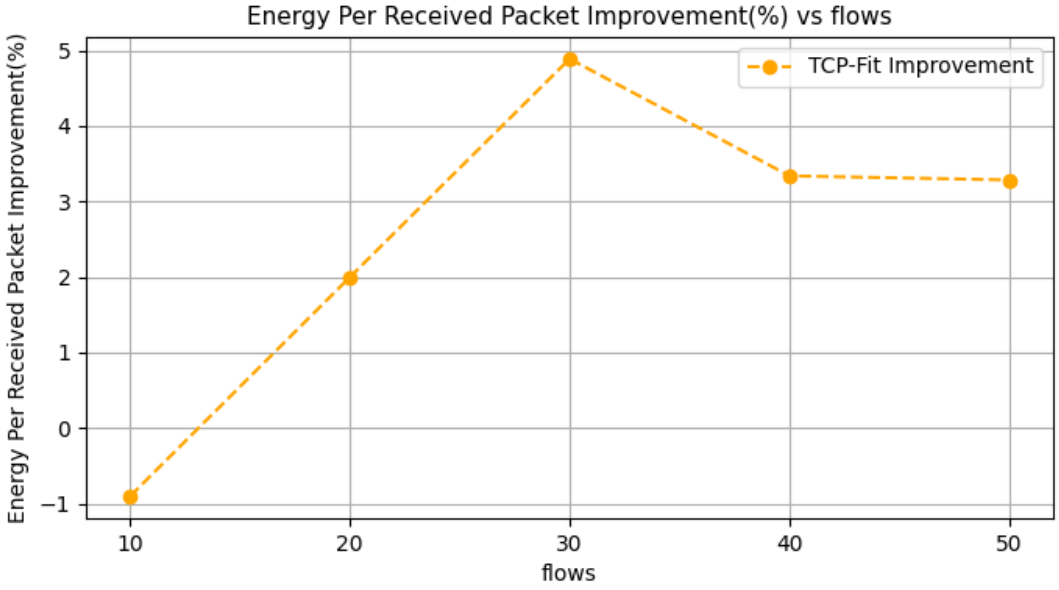

Delivery Ratio vs flows
TCP-Tahoe
TCP-Fit
Delivery Ratio
0.95
0.90
0.85
0.80
0.75
0.70
10
20
30
40
50
flows


Delivery Ratio Improvement(%) vs flows
TCP-Fit Improvement
Delivery Ratio Improvement(%)
−2
−4
−6
−8
−10
−12
−14
−16
−18
10
20
30
40
50
flows


Drop Ratio vs flows
TCP-Tahoe
TCP-Fit
Drop Ratio
0.00
0.01
0.02
0.03
0.04
0.05
0.06
10
20
30
40
50
flows


Drop Ratio Increase(%) vs flows
TCP-Fit Improvement
Drop Ratio Increase(%)
225
250
275
300
325
350
375
400
425
10
20
30
40
50
flows

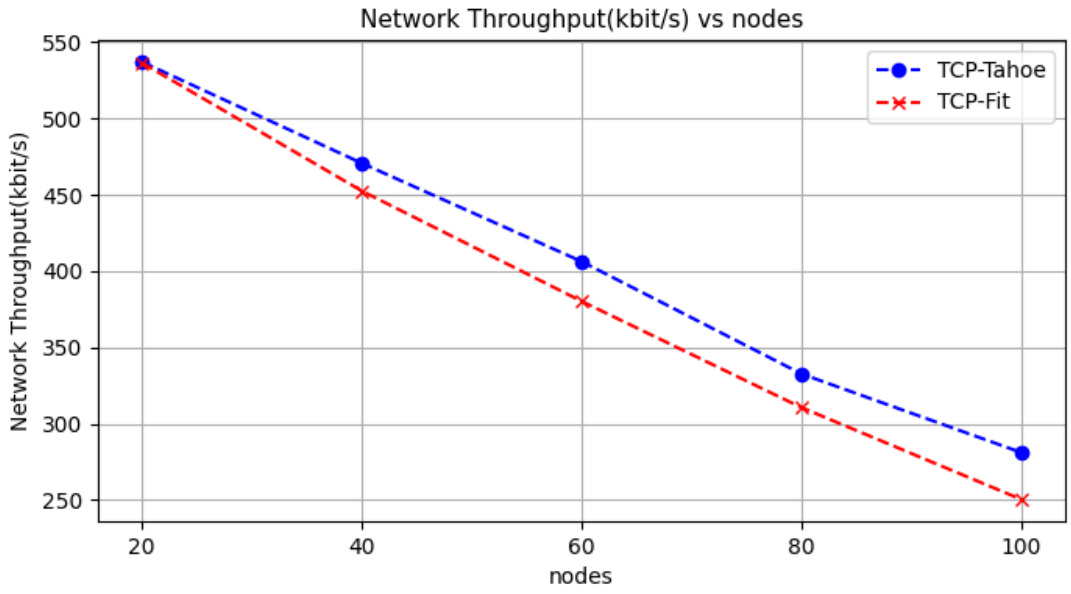


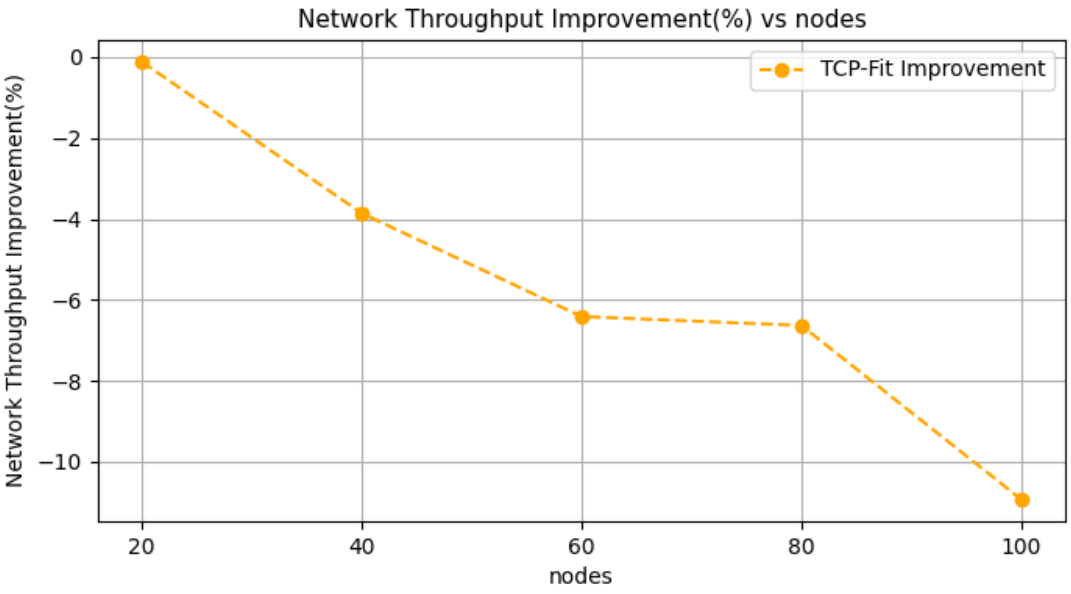


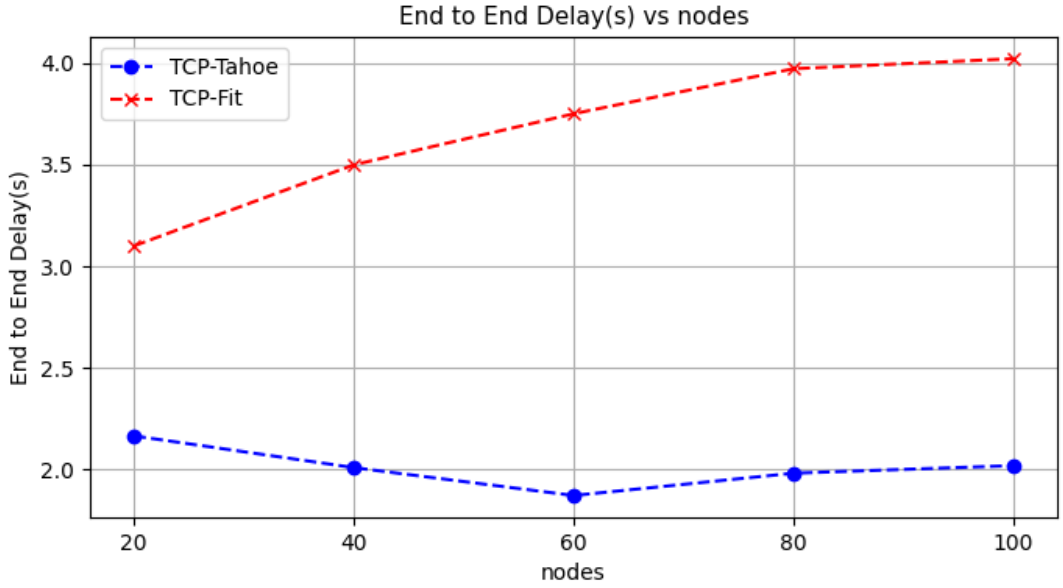
Observations

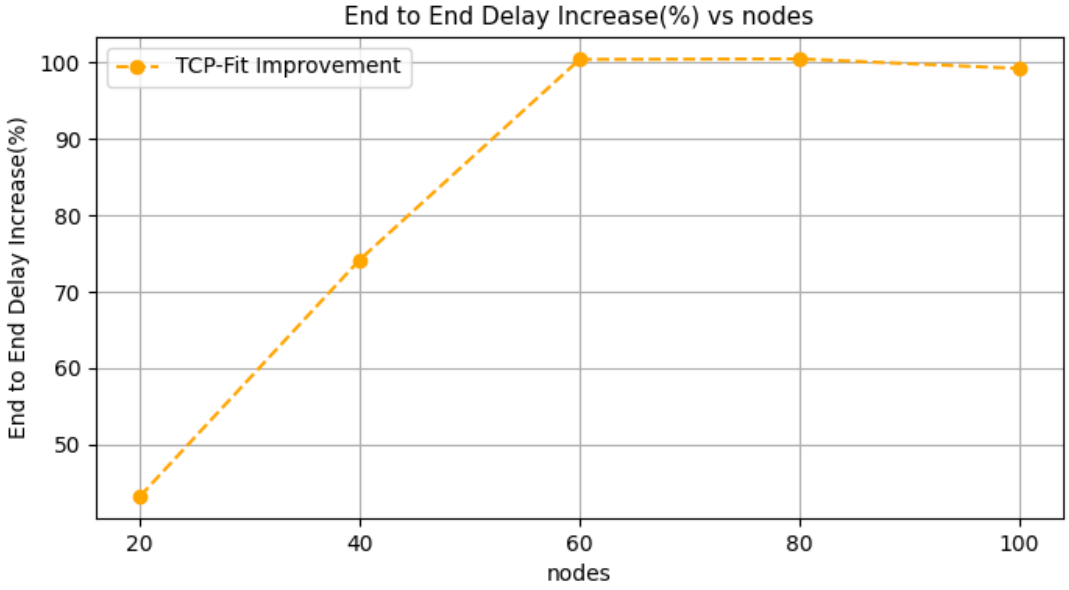
1. Worst performance of TCP-Fit seen so far. No improvement at all in any metric. Rather very poor performance in end to end delay, delivery ratio and drop ratio

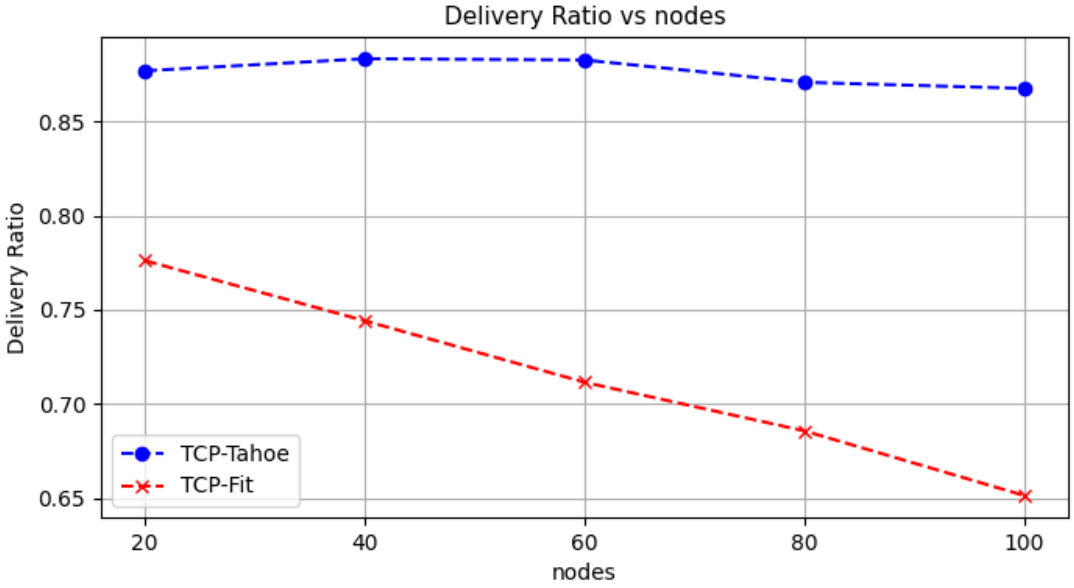
Varying nodes

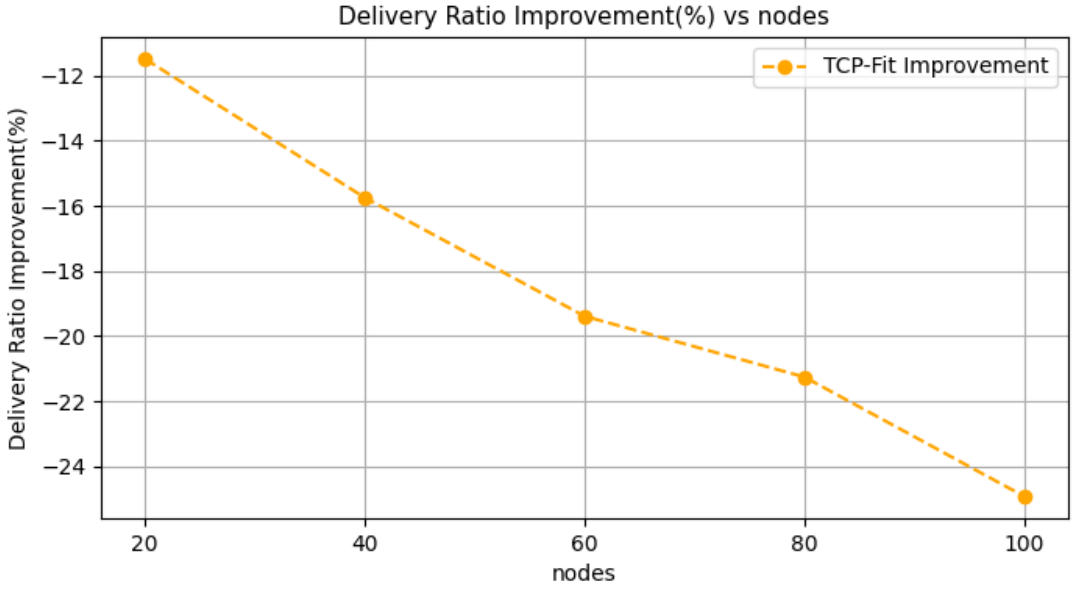


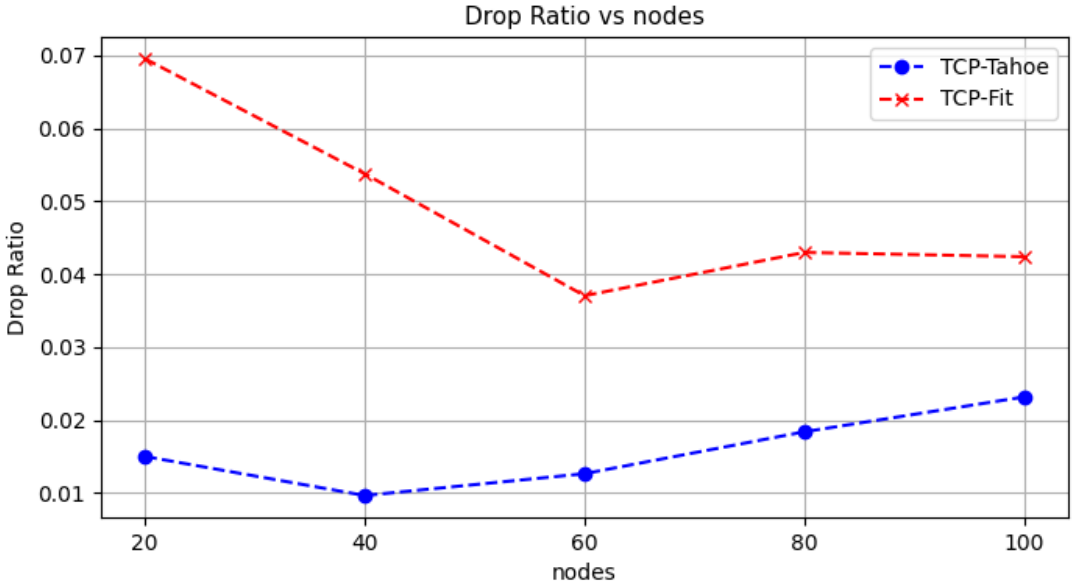


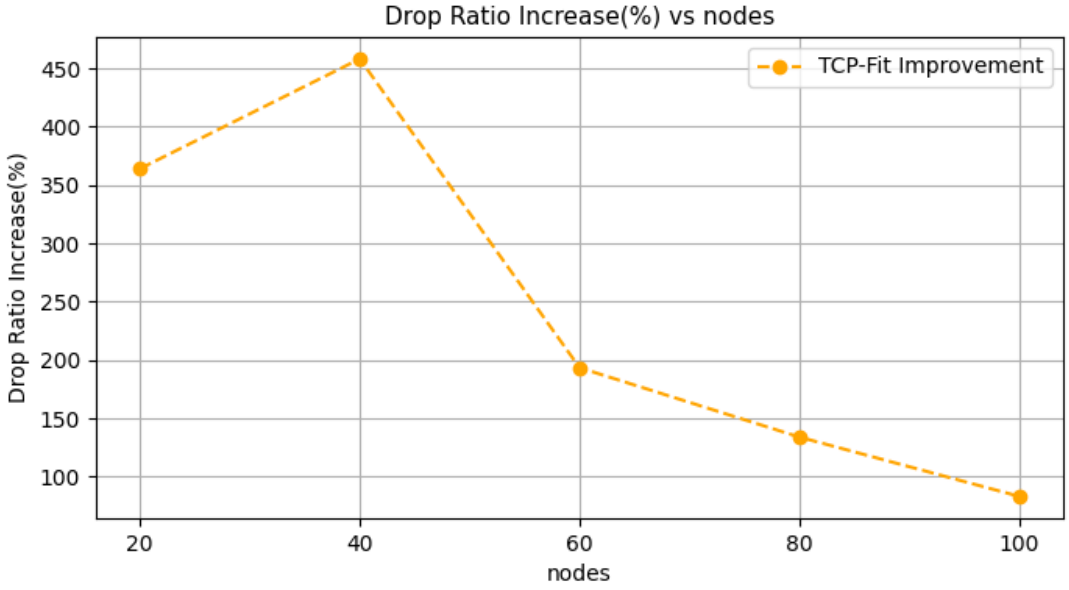

End to End Delay(s) vs nodes
TCP-Tahoe
TCP-Fit
End to End Delay(s)
4.0
3.5
3.0
2.5
2.0
20
40
60
80
100
nodes

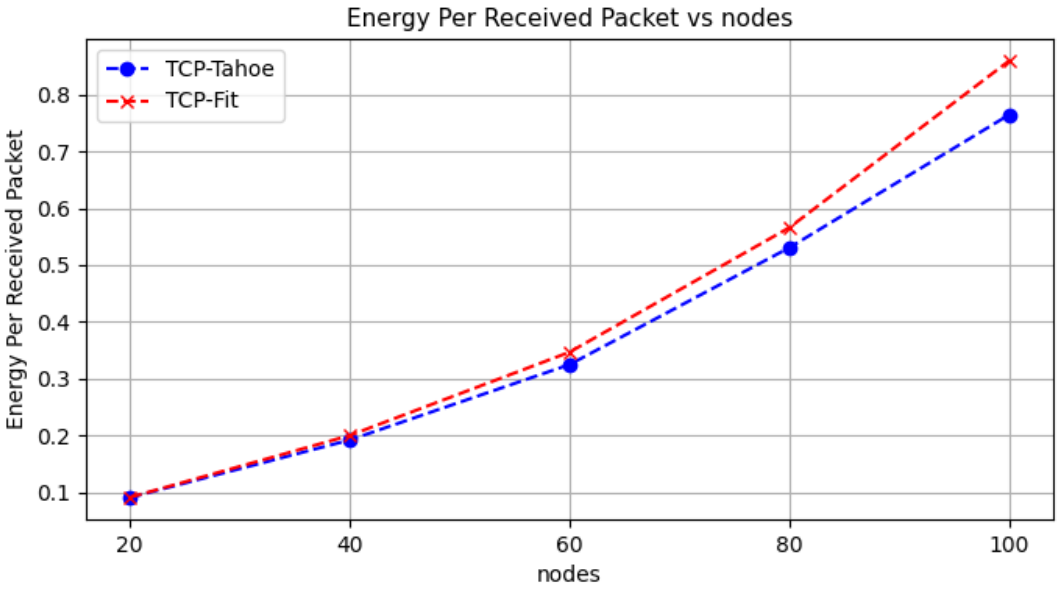

End to End Delay Increase(%) vs nodes
TCP-Fit Improvement
End to End Delay Increase(%)
100
90
80
70
60
50
20
40
60
80
100
nodes

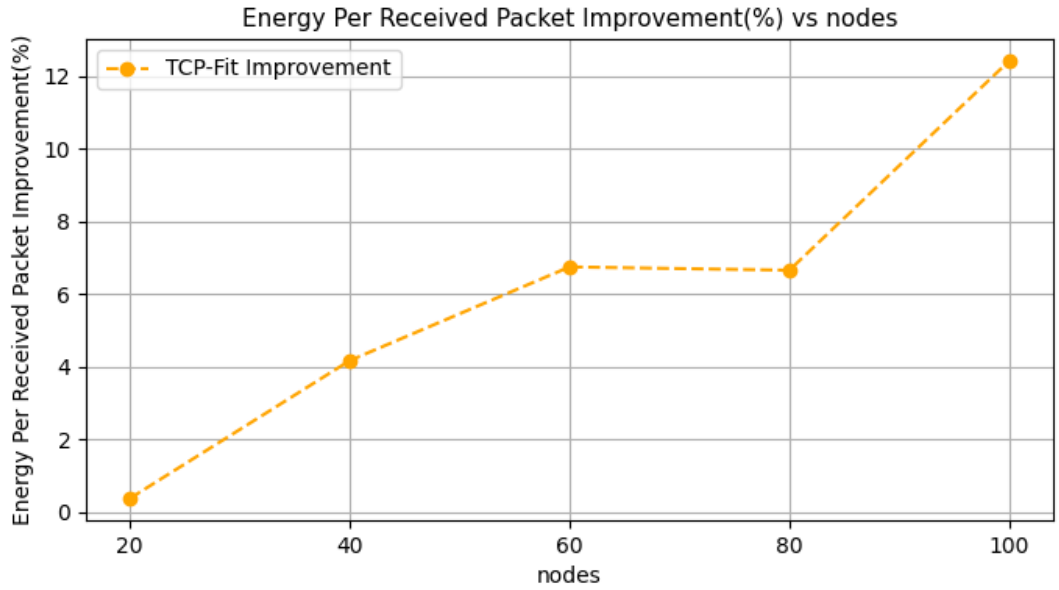

Delivery Ratio vs nodes
Delivery Ratio
0.85
0.80
0.75
0.70
0.65
TCP-Tahoe
TCP-Fit
20
40
60
80
100
nodes


Delivery Ratio Improvement(%) vs nodes
Delivery Ratio Improvement(%)
−12
−14
−16
−18
−20
−22
−24
TCP-Fit Improvement
20
40
60
80
100
nodes


Drop Ratio vs nodes
TCP-Tahoe
TCP-Fit
Drop Ratio
0.07
0.06
0.05
0.04
0.03
0.02
0.01
20
40
60
80
100
nodes


Drop Ratio Increase(%) vs nodes
TCP-Fit Improvement
Drop Ratio Increase(%)
450
400
350
300
250
200
150
100
20
40
60
80
100
nodes

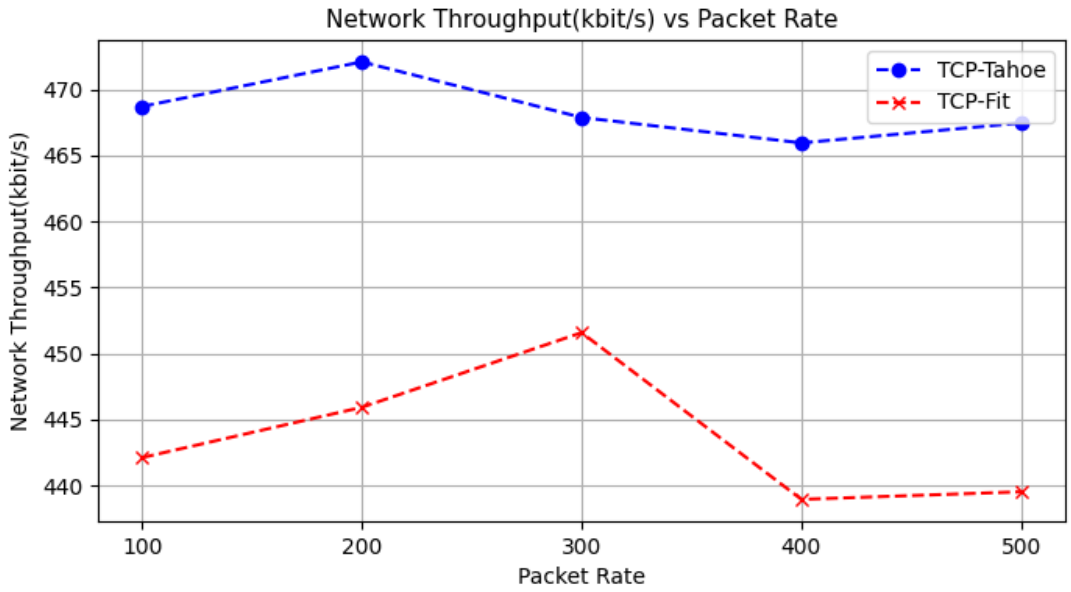


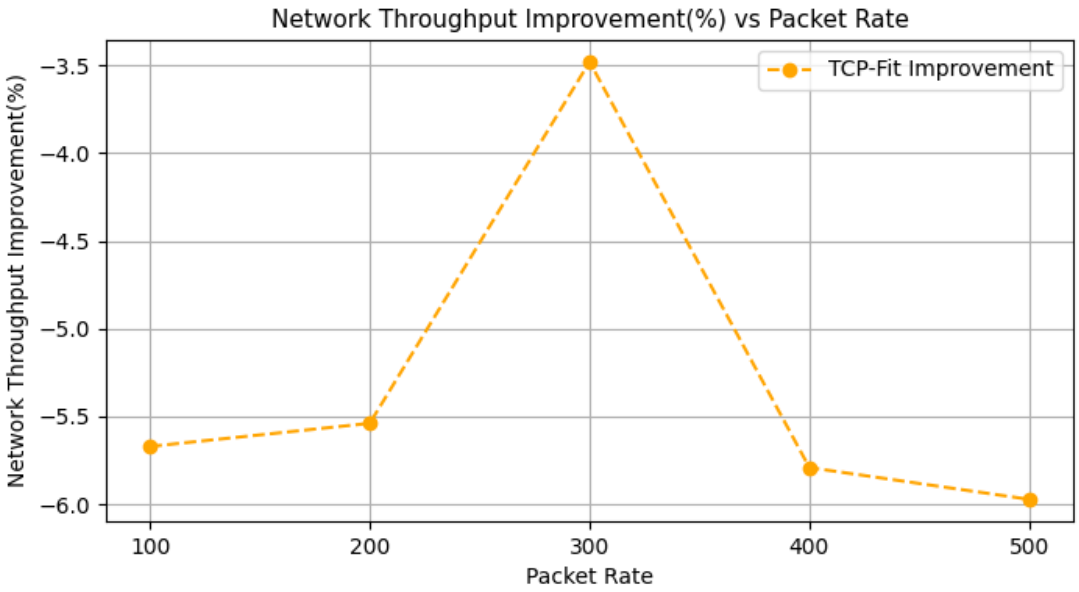


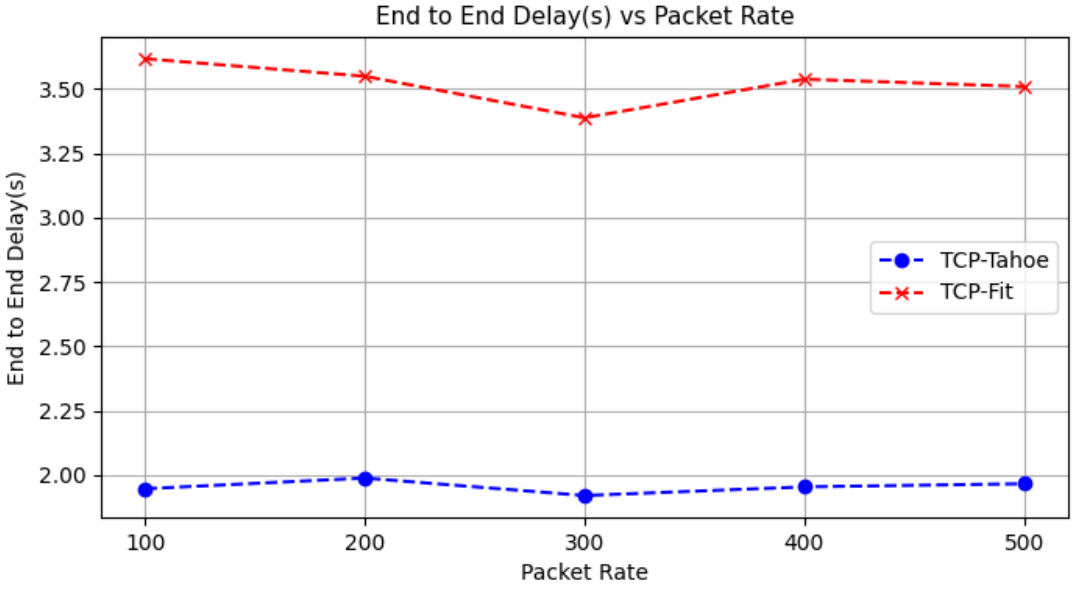
Observations

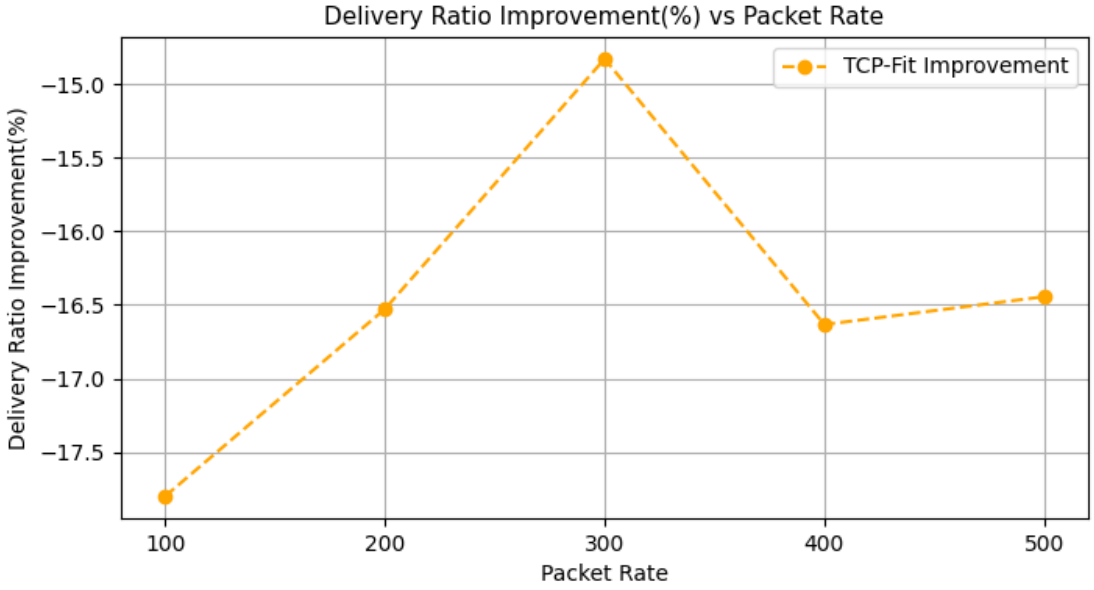
1. Similar observations as the previous topology
2. Very poor performance of TCP Fit algorithm

Varying packet rate

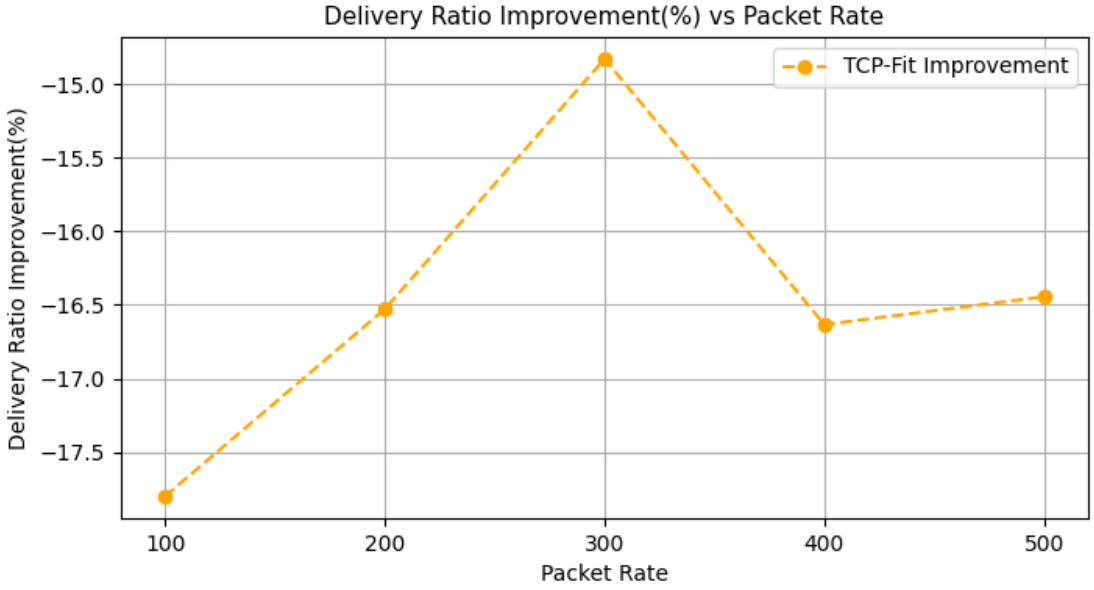


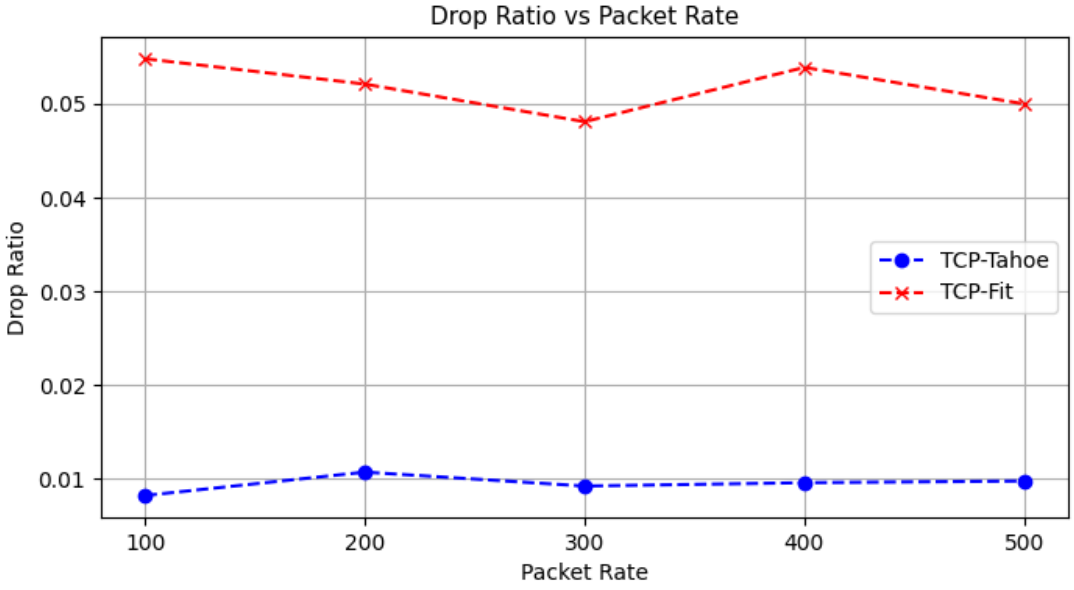


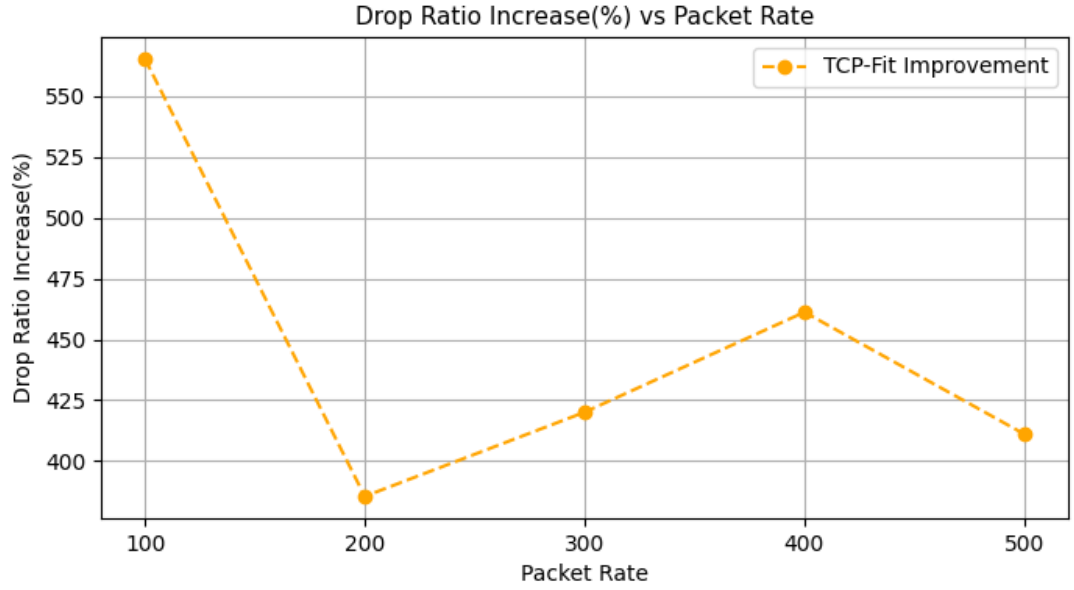

End to End Delay(s) vs Packet Rate
End to End Delay(s)
3.50
3.25
3.00
2.75
2.50
2.25
2.00
100
200
300
400
500
Packet Rate
TCP-Tahoe
TCP-Fit

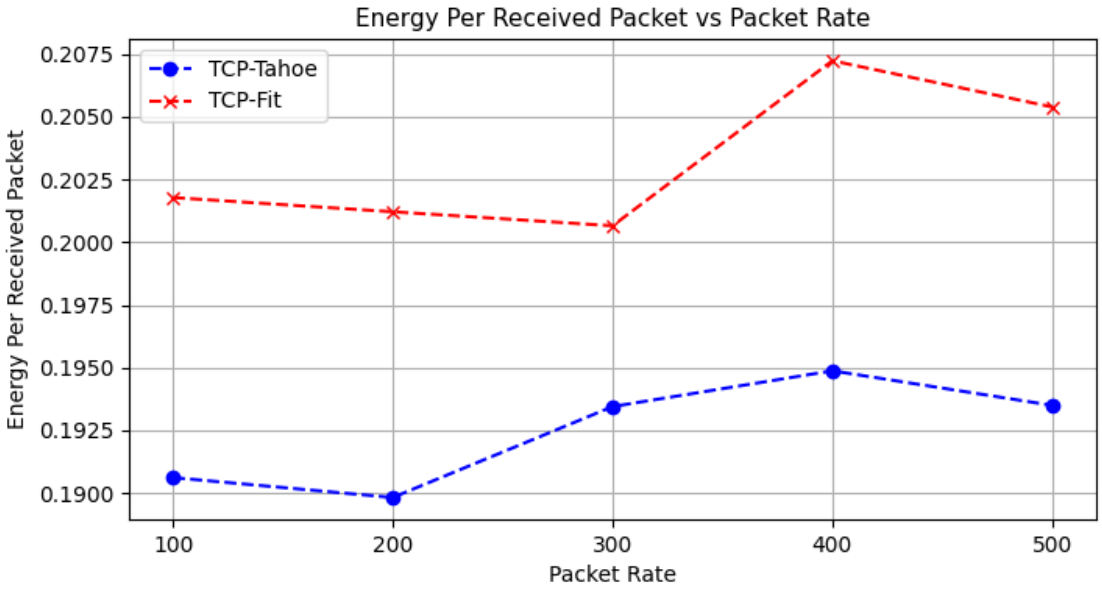

Delivery Ratio Improvement(%) vs Packet Rate
Delivery Ratio Improvement(%)
−15.0
−15.5
−16.0
−16.5
−17.0
−17.5
100
200
300
400
500
Packet Rate
TCP-Fit Improvement

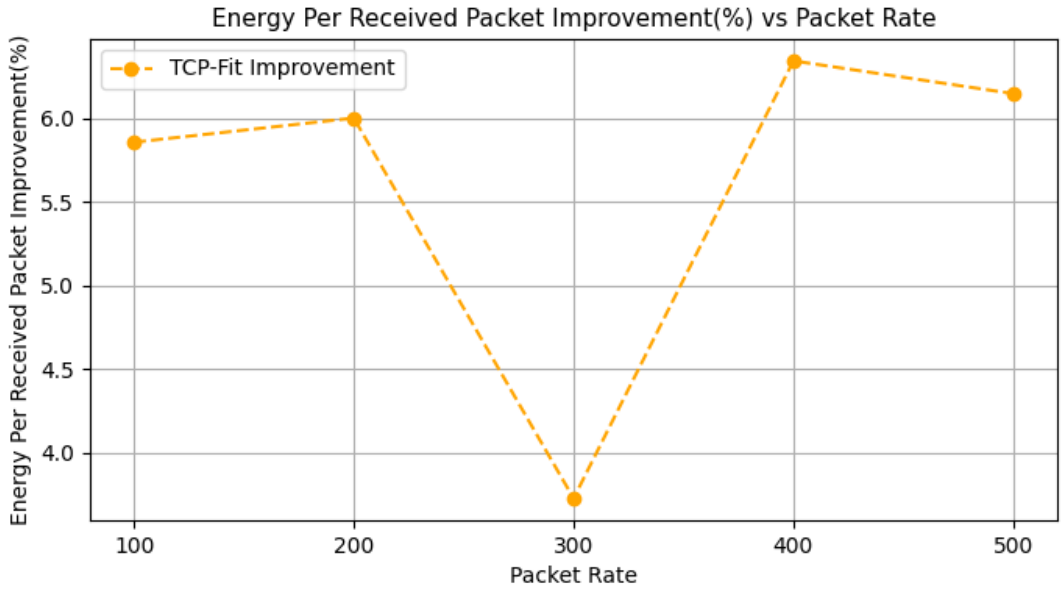

Delivery Ratio vs Packet Rate
Delivery Ratio
0.88
0.86
0.84
0.82
0.80
0.78
0.76
0.74
0.72
100
200
300
400
500
Packet Rate
TCP-Tahoe
TCP-Fit


Delivery Ratio Improvement(%) vs Packet Rate
Delivery Ratio Improvement(%)
−15.0
−15.5
−16.0
−16.5
−17.0
−17.5
100
200
300
400
500
Packet Rate
TCP-Fit Improvement


Drop Ratio vs Packet Rate
Drop Ratio
0.05
0.04
0.03
0.02
0.01
100
200
300
400
500
Packet Rate
TCP-Tahoe
TCP-Fit


Drop Ratio Increase(%) vs Packet Rate
Drop Ratio Increase(%)
550
525
500
475
450
425
400
100
200
300
400
500
Packet Rate
TCP-Fit Improvement

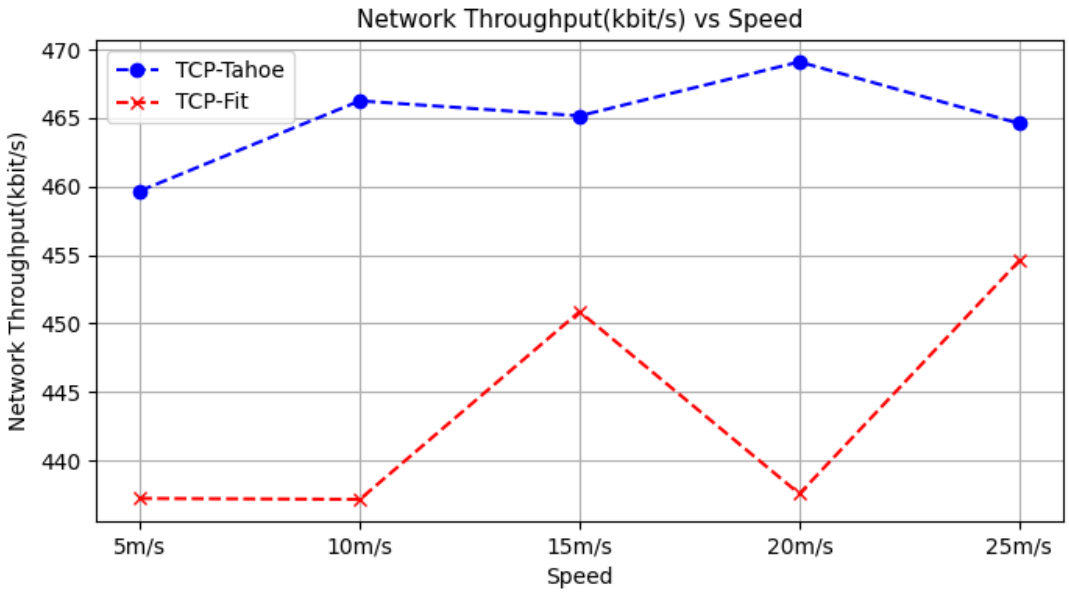


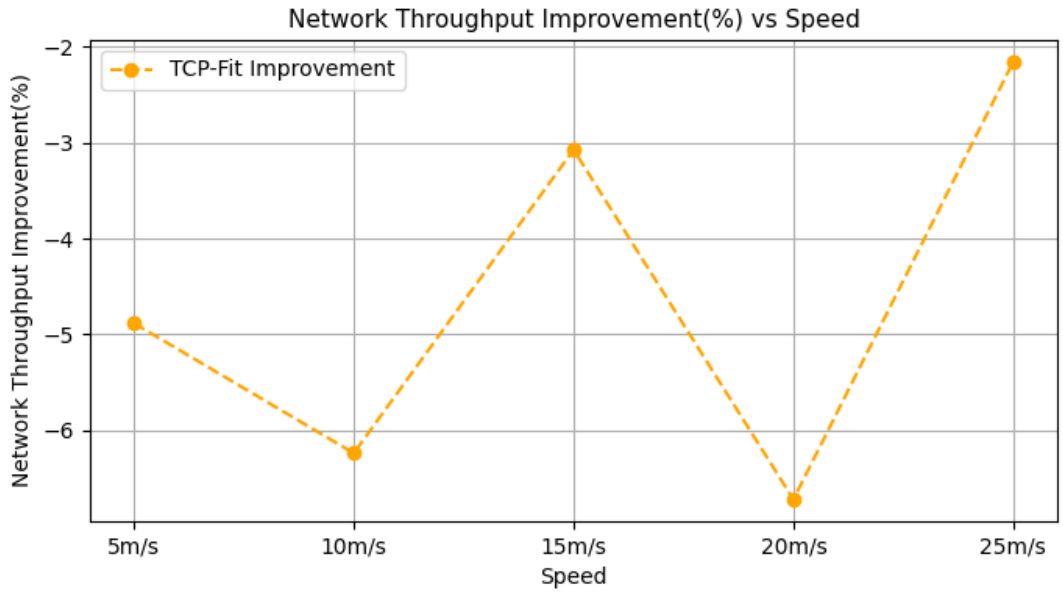


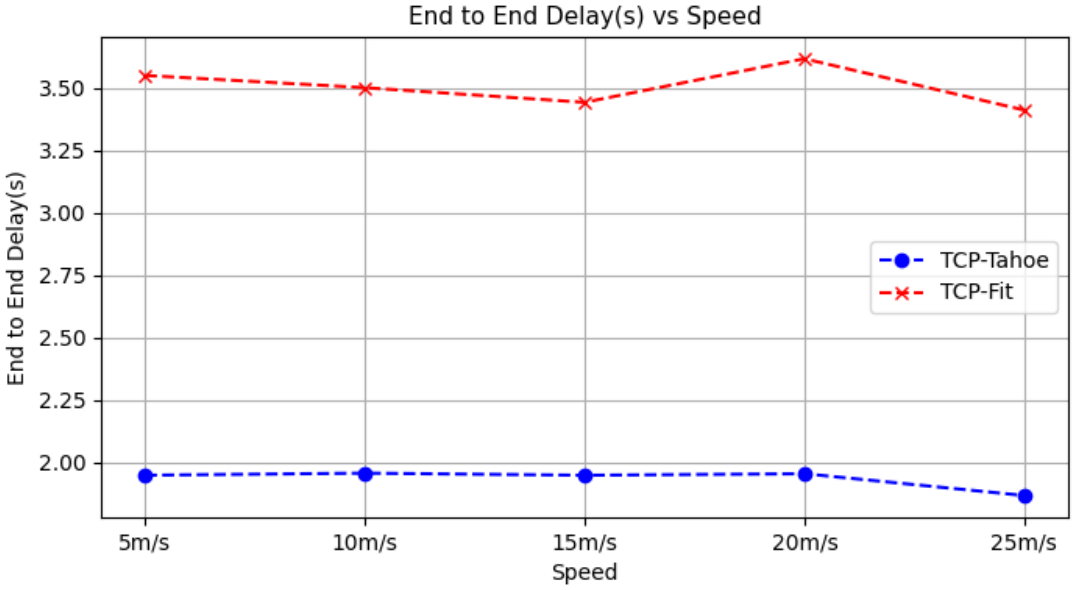
Observations

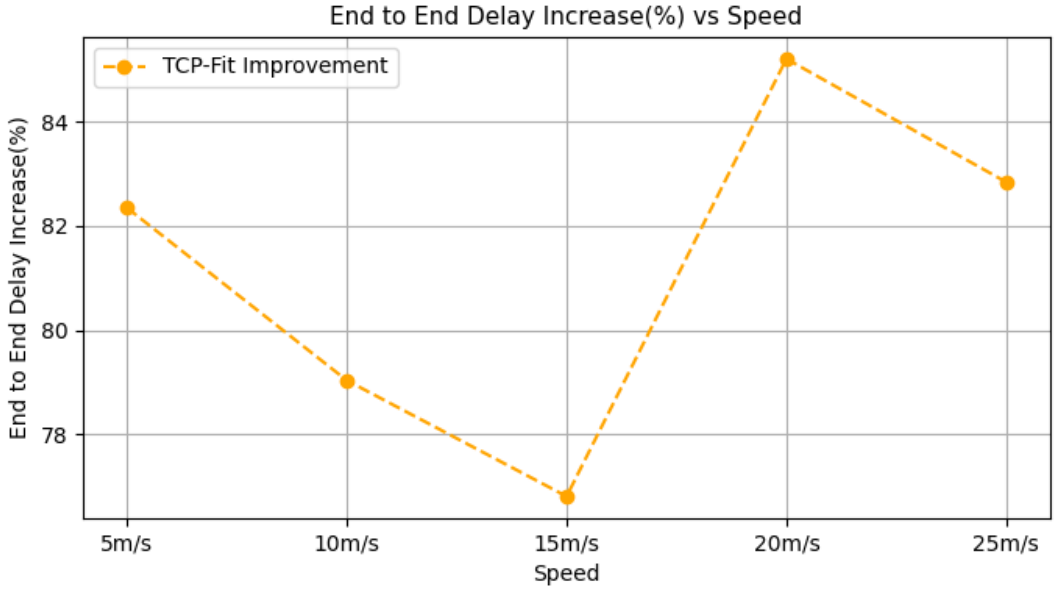
1. Similar observations as previous topology
2. TCP-Fit continues to show poor performance

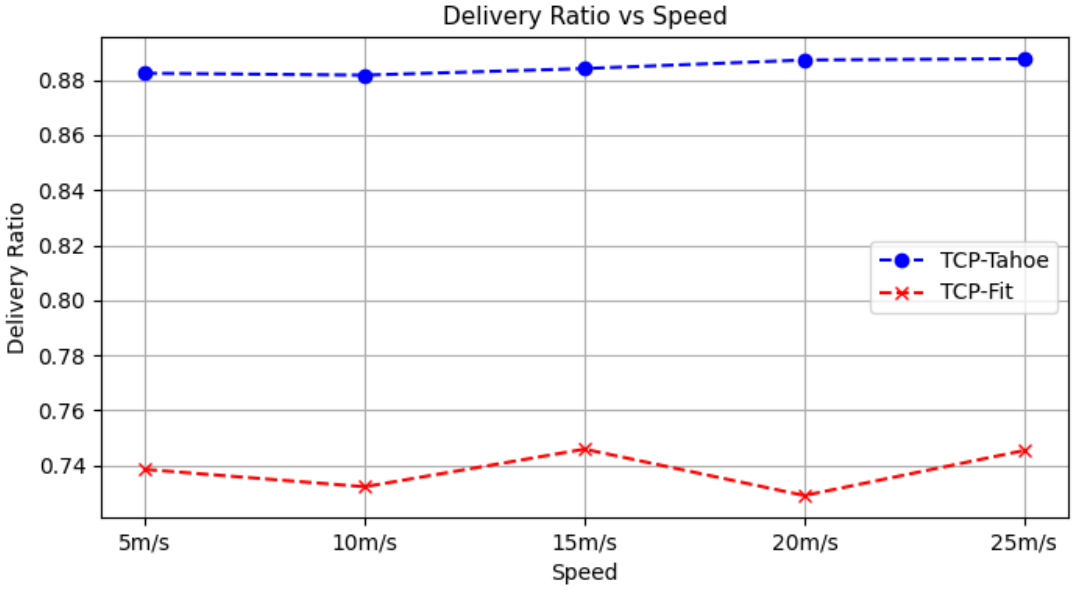
Varying speed

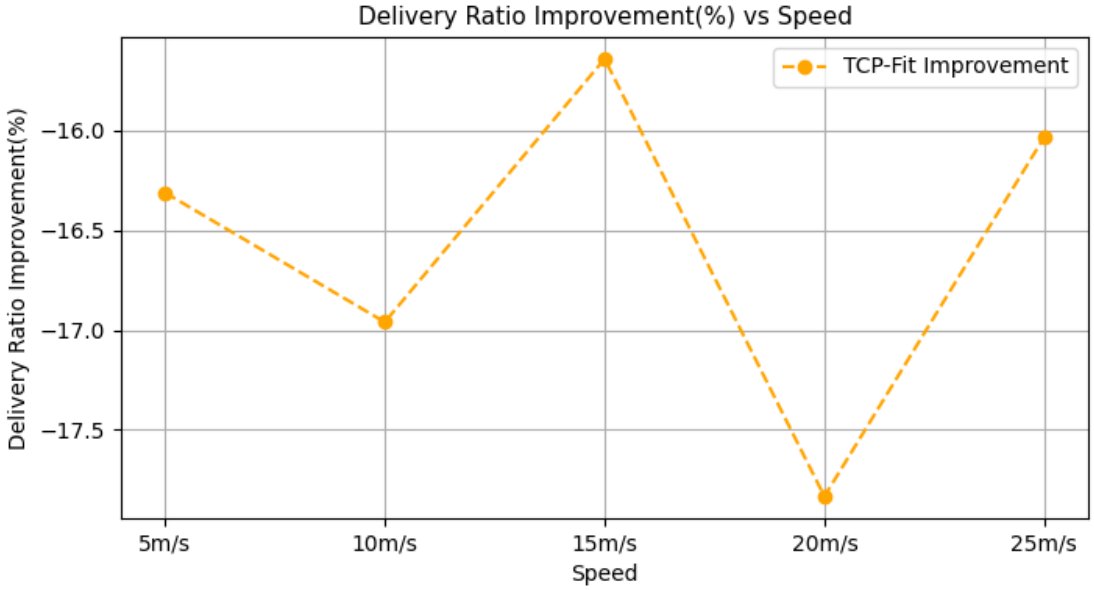


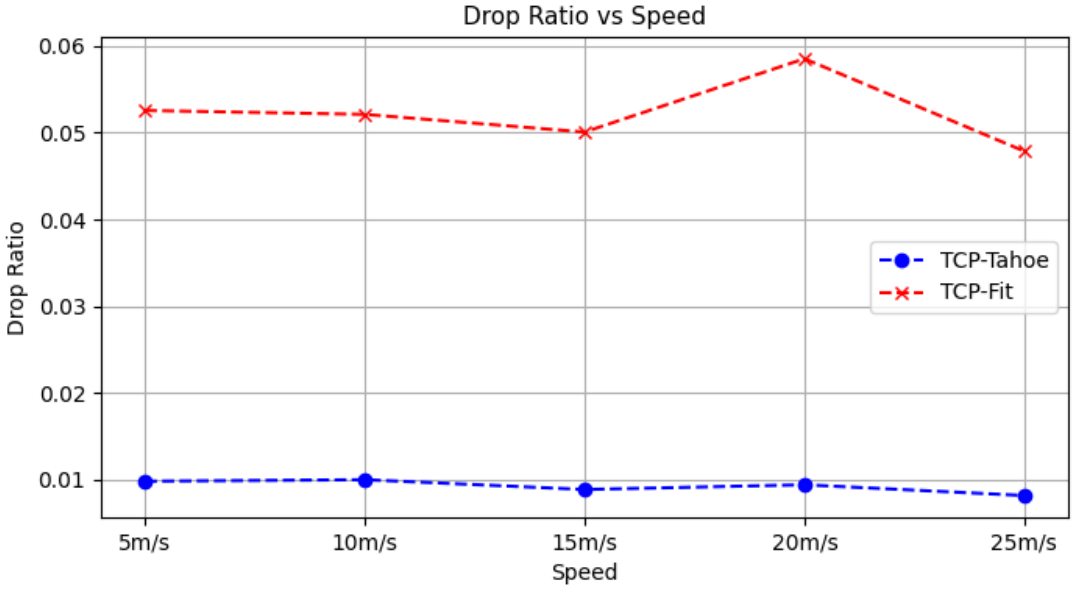


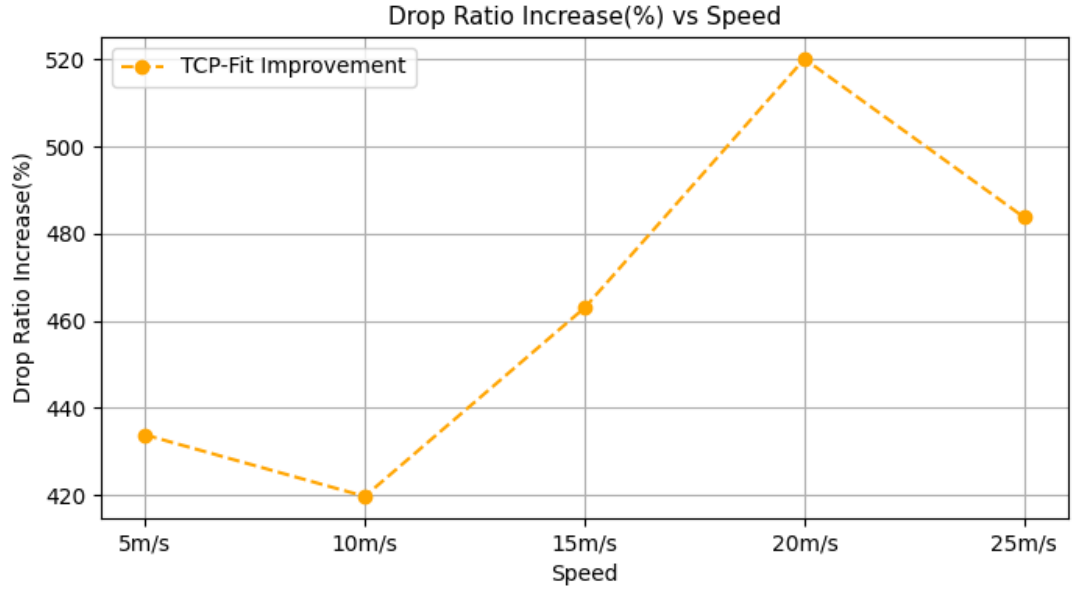

End to End Delay(s) vs Speed
End to End Delay(s)
3.50
3.25
3.00
2.75
2.50
2.25
2.00
TCP-Tahoe
TCP-Fit
5m/s
10m/s
15m/s
20m/s
25m/s
Speed

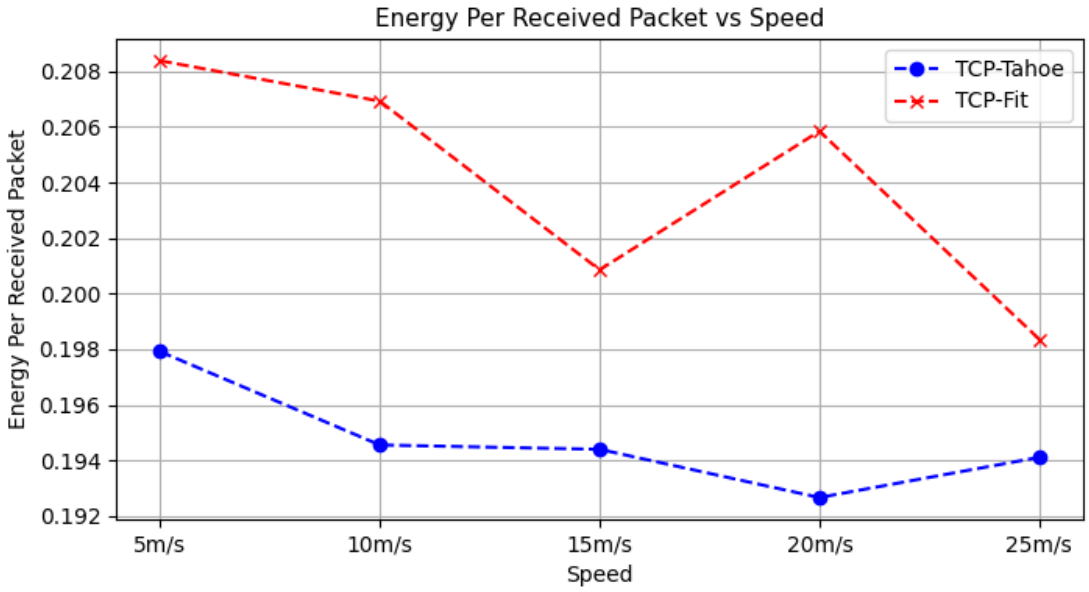

End to End Delay Increase(%) vs Speed
TCP-Fit Improvement
End to End Delay Increase(%)
84
82
80
78
5m/s
10m/s
15m/s
20m/s
25m/s
Speed

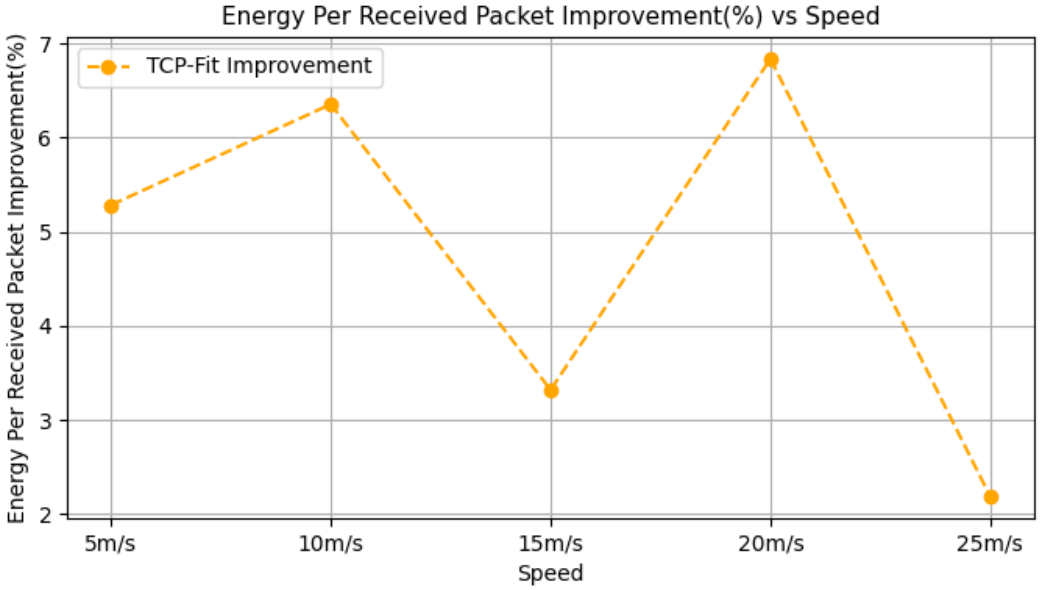

Delivery Ratio vs Speed
Delivery Ratio
0.88
0.86
0.84
0.82
0.80
0.78
0.76
0.74
TCP-Tahoe
TCP-Fit
5m/s
10m/s
15m/s
20m/s
25m/s
Speed


Delivery Ratio Improvement(%) vs Speed
Delivery Ratio Improvement(%)
−16.0
−16.5
−17.0
−17.5
TCP-Fit Improvement
5m/s
10m/s
15m/s
20m/s
25m/s
Speed


Drop Ratio vs Speed
Drop Ratio
0.06
0.05
0.04
0.03
0.02
0.01
TCP-Tahoe
TCP-Fit
5m/s
10m/s
15m/s
20m/s
25m/s
Speed


Drop Ratio Increase(%) vs Speed
Drop Ratio Increase(%)
520
500
480
460
440
420
TCP-Fit Improvement
5m/s
10m/s
15m/s
20m/s
25m/s
Speed





Observations

1. Similar observations as previous topology
2. Tcp Fit shows poor performance

Summary Findings

The TCP-Fit algorithm seems to provide great improvement of throughput in networks where there are considerable amounts of packet drops that are not due to congestion. The wired and wired-wireless topology simulates these by adding random packet drop on the bottleneck link. We find a 30% to 50% increase in throughput with a very small increase in end to end delay.

However, the performance does not hold up in congested networks as seen in the Wireless networks. The throughput improvement is not as much (5% - 10%), sometimes even negative. However the end to end delay can become very high (20%-190%), the delivery ratio can drop quite low and drop ratio can become very high compared to TCP-Tahoe.

The algorithm seems to fail to detect that a network is congested. It tries to send more packets in a congested network and so end to end delay, delivery ratio, drop ratio all worsens. The more the network is congested, the worse TCP-Fit seems to perform.

An important constant was α in updating the value of N. The paper only says that it lies between 0 and 1, but does not mention the used value in the experiments. From my experiment I have found that TCP-Fit can be made to perform similar to TCP-Tahoe or better by tuning the value for each network configuration. However, this is not practical at all since the value needs to be tuned for any change in network configuration.

In summary

1. The algorithm works well in topology considered in the paper, scenarios with packet drops that are not caused by congestion but rather due to the medium,
2. The algorithm does not perform that well in congested networks, it can barely increase the throughput, but increases end to end delay, drop ratio and worsens delivery ratio
3. The algorithms' performance seems topology dependent. It fails to perform well in other wireless topology such as random or grid topology. So the algorithm does not generalize well.