CSE 322 Term Project Report

Implementation of ASRAN Algorithm in NS3

Submitted by

Mashiat Mustaq

1705005

Level 3 - Term 2

February 22, 2022

1 Task A

Assigned Tasks -> 1705005%6 = 3

- Wired
- Wireless Low Rate Mobile

1.1 Wired Topology

1.1.1 Network Topology Under Simulation

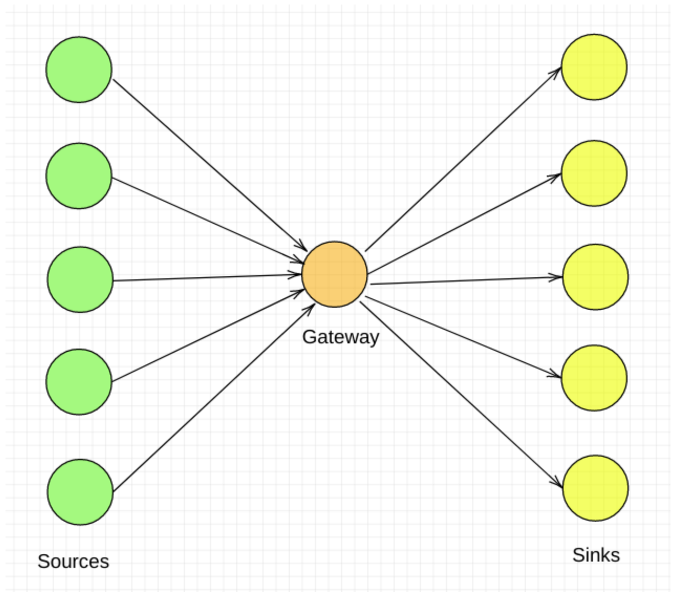


Figure 1: Wired Topology

The topology that I used for wired network has a gateway node between all source and sink nodes. To increase the flow numbers in the network, multiple applications were created in the sinks changing the port numbers. A graphical presentation of my topology is provided in 1. A brief description of the values of the parameters used in my topology is presented below in Table 1

1.1.2 Parameters Under Variation

In order to determine the behaviour of a wired network under different conditions, I varied the node numbers, packets sent per second, flows in my simulation topology and compared their throughputs, end to end delay, packet delivery and loss ratio. The varied values of different parameters are provided in Table 2

1.1.3 Results with Graphs

The graphs of different metrics by varying different parameters are provided in Figures 3, 5 & 7.

Parameters	Values
Duration(sec)	50
Packet size(byte)	512
Bandwidth(Gateway->Sink)	10Mbps
Bandwidth(Source->Gateway)	2Mbps
Channel Delay(Gateway->Sink)	45ms
Channel Delay (Source->Gateway)	0.01ms
Error Rate	0.0001

Table 1: Wired Topology - Simulation Parameters

Parameter	1st value	2nd value	3rd value	4th value	5th value
Nodes	20	40	60	80	100
Flows	10	20	30	40	50
Packets per sec	100	200	300	400	500

Table 2: Values of different parameters

1.1.4 Findings of Wired Topology

Network Throughput :

Keeeping the node numbers fixed, if number of flows are increased, then at first the network throughput increases drastically, later it increases very slowly which signifies that after a certain value, increase of flows doesn't effect much in the overall network throughput. This is due to the channels of the network reaching its maximum capacity.

Keeping the flows fixed, if the nodes are increased, it doesn't effect the network throughput that much. Initially it changes a bit, later it almost remains constant. Since the new nodes are not transmitting any packet, they do not bring much change overall.

Changing the packets sent per second causes the throughput to be increased initially. After some time, increase rate of the network throughput slows down due to congestion in the gateway node limiting the number of packets to be successfully sent.

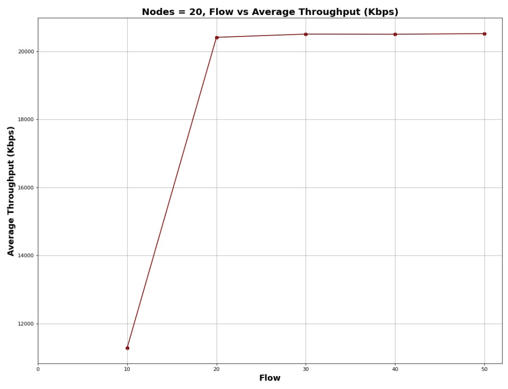
End to End Delay :

If flows and packets sent per second are increased, the delay gets larger due to heavy traffic on the gateway node. Changing the node number doesn't change the end to end delay that much and it almost keeps constant.

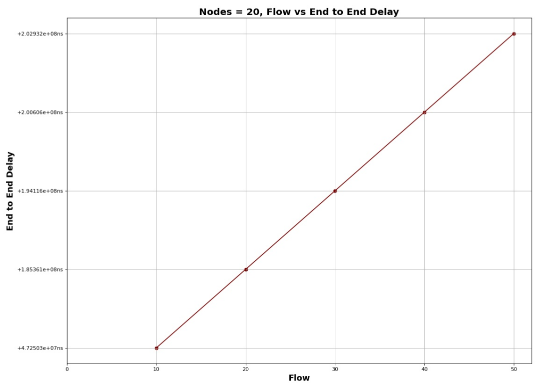
Packet Delivery Ratio & Packet Loss Ratio :

In all cases, even if the parameters varied, it stays in the range above 99% in my topology which is a very high

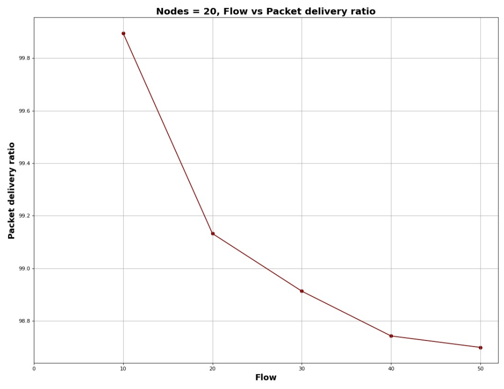
Figure 2: **Varying Flows**



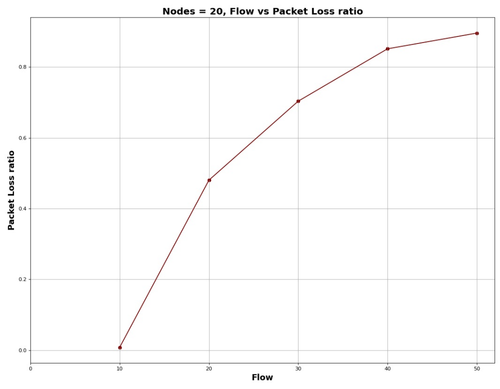
(a) Flow vs Network Throughput



(b) Flow vs End to End Delay



(c) Flow vs Packet Delivery Ratio

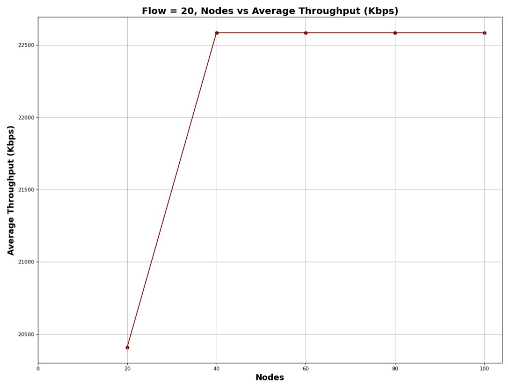


(d) Flow vs Packet Loss Ratio

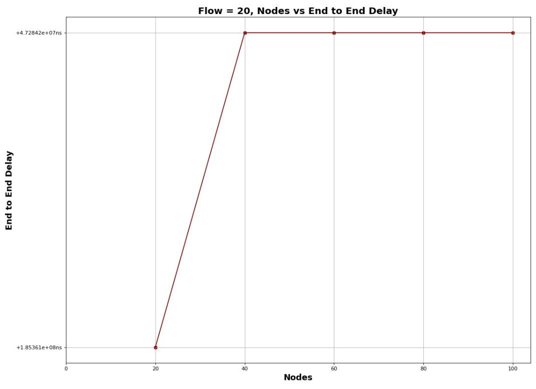
Figure 3: Varying flows and calculating different metrics in wired network

ratio. If flows are increased, delivery ratio decreases at a very small amount due to heavy traffic on the gateway. In case of changing the node number, the delivery ratio keeps almost constant. If more packets are sent, delivery ratio decreases a bit since heavy traffic in the gateway node drop some packets. But still the delivery ratio is high and the loss ratio is comparatively low than other type of network.

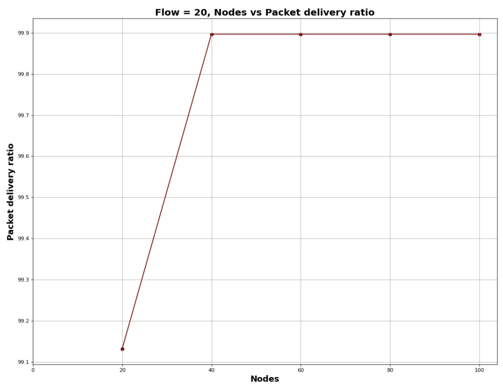
Figure 4: **Varying Nodes**



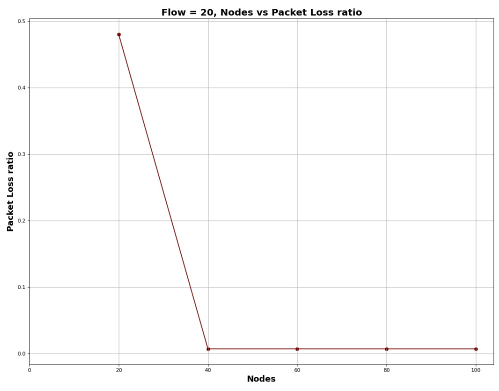
(a) Nodes vs Network Throughput



(b) Nodes vs End to End Delay



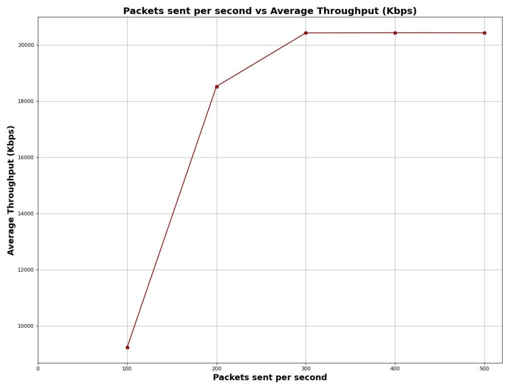
(c) Nodes vs Packet Delivery Ratio



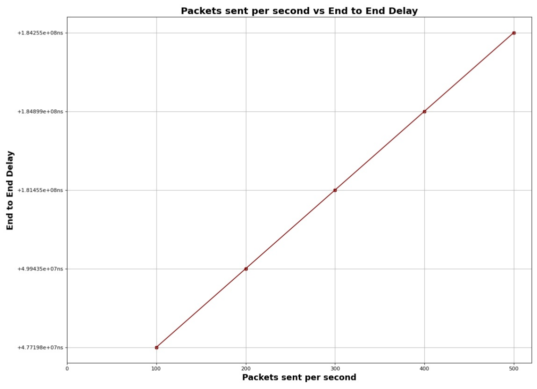
(d) Nodes vs Packet Loss Ratio

Figure 5: Varying nodes and calculating different metrics in wired network

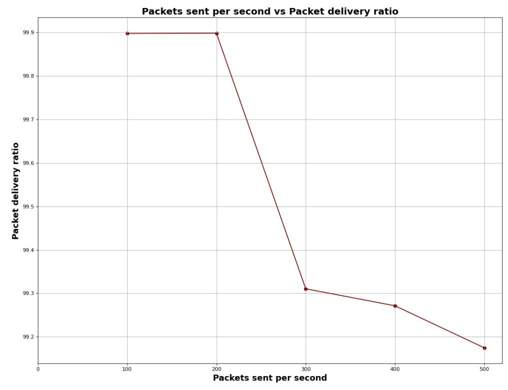
Figure 6: **Varying Packets sent per second**



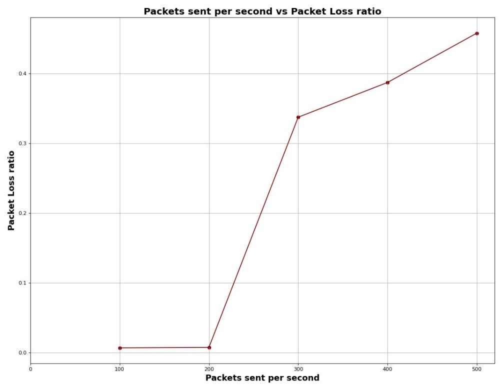
(a) Packets sent per second vs Network Throughput



(b) Packets sent per second vs End to End Delay



(c) Packets sent per second vs Packet Delivery Ratio



(d) Packets sent per second vs Packet Loss Ratio

Figure 7: Varying packets sent per second and calculating different metrics in wired network

1.2 Wireless low rate Topology (Mobile)

1.2.1 Network Topology Under Simulation

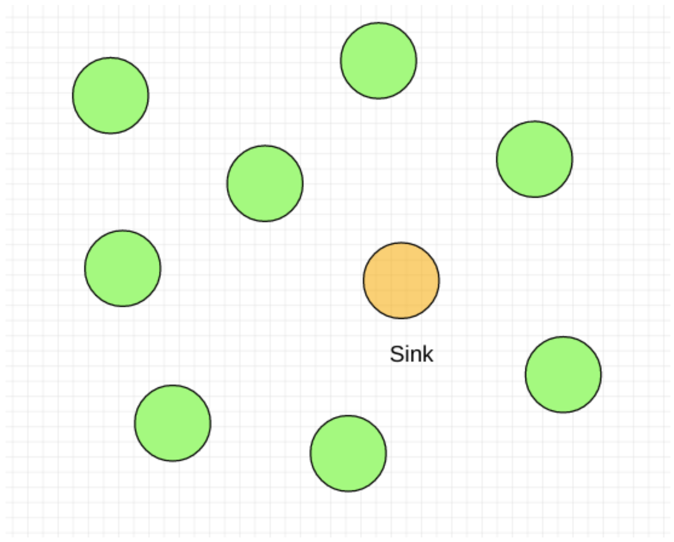


Figure 8: Wireless low rate mobile Topology

The topology that I used for low rate wireless PAN has a single receiver where all other nodes transmit packets. To increase flow numbers, different port numbers are used for creating multiple application on the sink (receiver). A graphical presentation of the topology is provided in Figure 8. A brief description of the values of the parameters used in my topology is presented below in Table 3

Parameters	Values
Duration(sec)	100
Packet size	100
Mobility Model	GaussMarkovMobilityModel
Gridsize	25*25*25

Table 3: Low rate Wireless PAN - Simulation Parameters

1.2.2 Parameters Under Variation

In order to determine the behaviour of a low rate wireless mobile network under different conditions, I varied the node numbers, packets sent per second, flows and speed of nodes in my simulation topology and compared their throughputs, end to end delay, packet delivery and loss ratio. The varied values of different parameters are provided in Table 4

1.2.3 Results with Graphs

The graphs of different metrics by varying different parameters are provided in Figures 10, 12, 14 & 16.

Parameter	1st value	2nd value	3rd value	4th value	5th value
Nodes	20	40	60	80	100
Flows	10	20	30	40	50
Packets per sec	100	200	300	400	500
Speed	5	10	15	20	25

Table 4: Values of different parameters

1.2.4 Findings of Wireless Low Rate Mobile Network

Network Throughput :

Throughput increases if the number of flows are increased in low rate wireless mobile network since more packets are transmitted overall. If nodes are increased keeping the flow fixed, network throughput decreases. Analyzing the other metrics(Total sent, received & lost packets, it was observed that the total number of packets sent decreased with the increase of nodes. Also the packet loss ratio also increased. It can be deduced that the increase of nodes somehow interrupts other nodes' flows which results in a decreased throughput.

Changing packets sent per second doesn't change the network throughput that much and varies under a same range (here around 0.6). However since it doesn't increase the throughput , that means packet loss ratio gets greater with the increase of packets. This is due to congestion in the network which drops packets from the network. Varying speed doesn't have a increasing / decreasing effect in the network. In a mobile network, positions of the nodes are undeterministic which results in unpredictable performance when nodes move in a certain speed. There is only one sink and it depends on the relative distance between sink and a particular node at a time for the sender to successfully transmit packets.

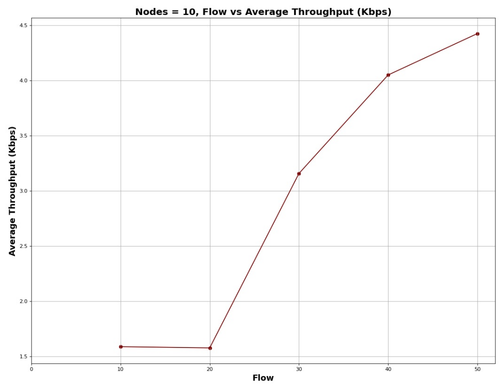
End to End Delay :

In a wireless low rate mobile network, if flows, nodes, speed or packets sent per second are increased, the delay increases linearly. Increasing flows or packets cause more traffic, which leads to slower delivery time. Increasing speed make the position of the nodes unpredictable, and the delay also increases as an effect of it. If nodes are increased, it is assumed that other nodes interrupt the performance of the flows which result in longer end to end delay.

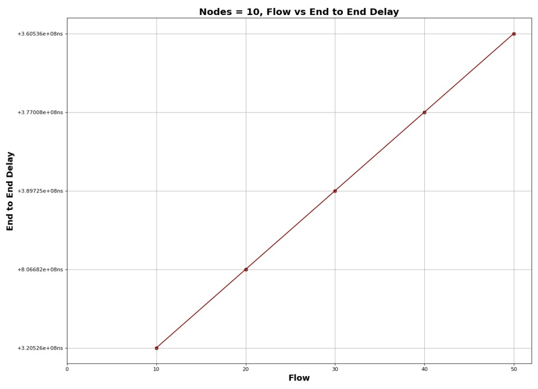
Packet Delivery Ratio & Packet Loss Ratio :

Packet Delivery Ratio decreases if nodes or flows are increased in wireless low rate mobile network. From the previous discussion, we observed that increasing nodes cause a lower throughput in the network which also indicates that the delivery ratio drops with the increase of nodes. Though increasing flows increases network throughput but gives a lower delivery ratio. It can be assumed that wireless low rate network has a capacity and if flows are

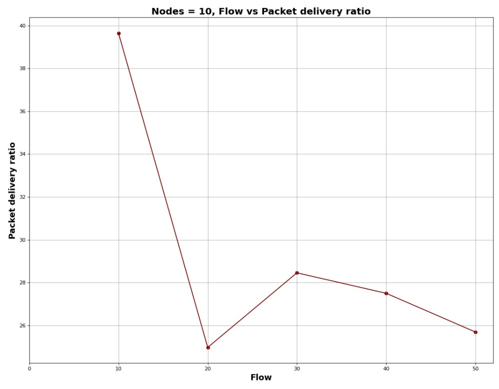
Figure 9: **Varying Flows**



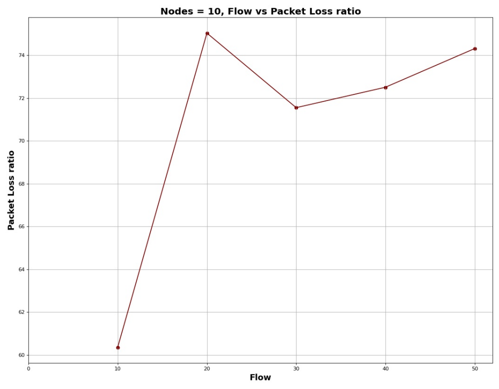
(a) Flow vs Network Throughput



(b) Flow vs End to End Delay



(c) Flow vs Packet Delivery Ratio



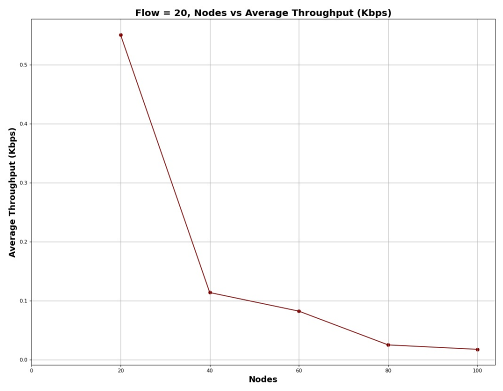
(d) Flow vs Packet Loss Ratio

Figure 10: Varying flows and calculating different metrics in wireless low rate mobile network

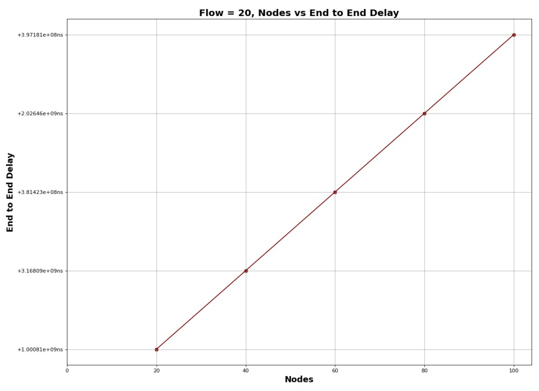
increased, flows individually get to send less data successfully as network traffic may occur which cause more packets to be dropped but the throughput increases as an overall effect of summing all the flows' transmitted packets.

As a network reaches its capacity, extra packets are dropped. So increasing packets sent per second doesn't change packet delivery or loss ratio that much and it fluctuates around a small range of value. Also increasing speed doesn't have a linear relation with packet delivery ratio and mostly depends on the performance of throughput on that speed. If the throughput is higher on that speed, the delivery ratio increases proportionally.

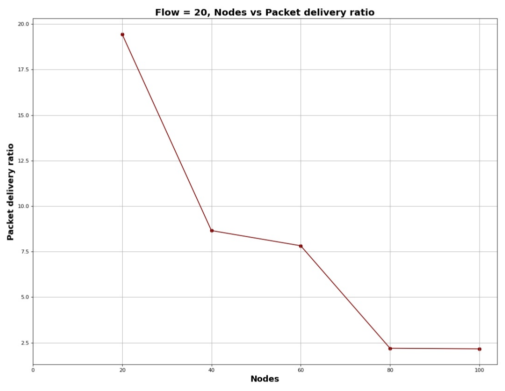
Figure 11: **Varying Nodes**



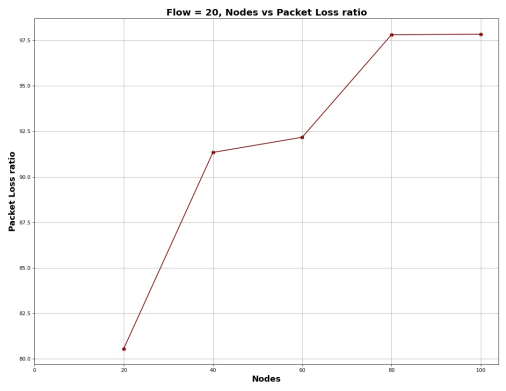
(a) Nodes vs Network Throughput



(b) Nodes vs End to End Delay



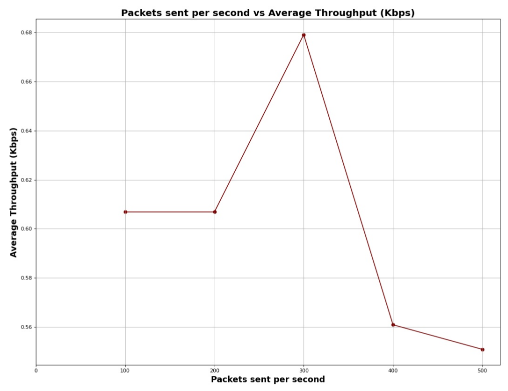
(c) Nodes vs Packet Delivery Ratio



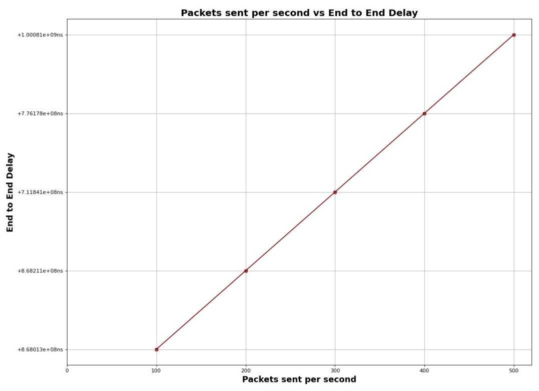
(d) Nodes vs Packet Loss Ratio

Figure 12: Varying nodes and calculating different metrics in Wireless low rate mobile network

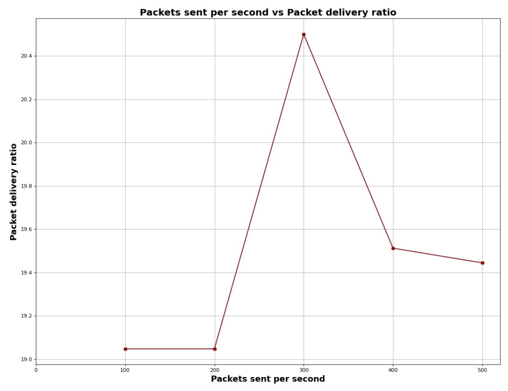
Figure 13: **Varying Packets sent per second**



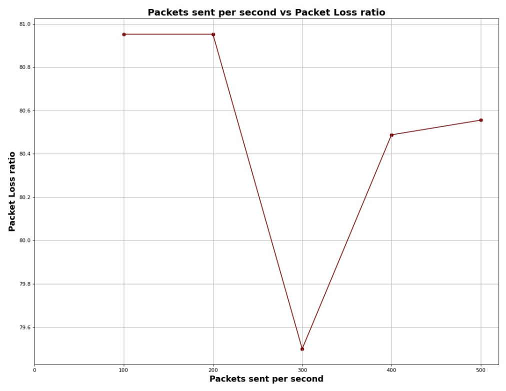
(a) Packets sent per second vs Network Throughput



(b) Packets sent per second vs End to End Delay



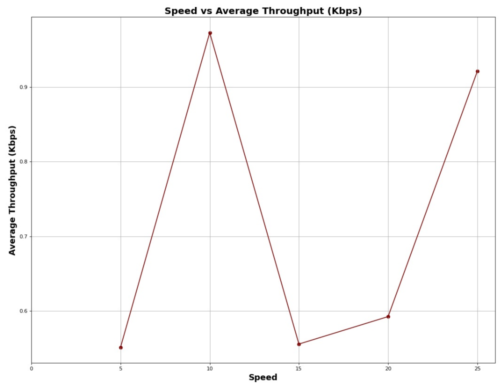
(c) Packets sent per second vs Packet Delivery Ratio



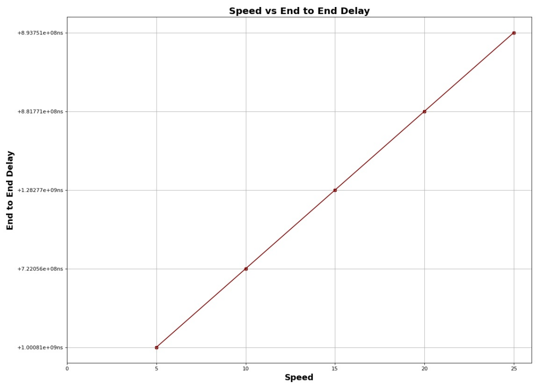
(d) Packets sent per second vs Packet Loss Ratio

Figure 14: Varying packets sent per second and calculating different metrics in Wireless low rate mobile network

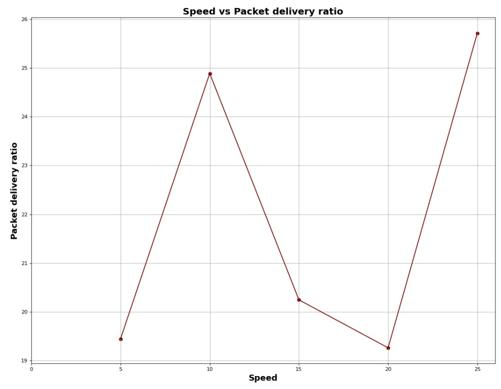
Figure 15: **Varying speed**



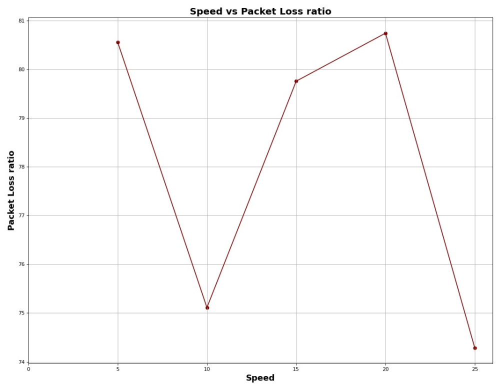
(a) Speed vs Network Throughput



(b) Speed vs End to End Delay



(c) Speed vs Packet Delivery Ratio



(d) Speed vs Packet Loss Ratio

Figure 16: Varying speed and calculating different metrics in Wireless low rate mobile network

2 Task B

My proposed modification is to implement ASRAN (Adaptive Ssthresh Reviser for Flying Adhoc Network) algorithm in NS3 in reference to [1]. This algorithm is designed for multihop UAV network where transient link instability occurs more often than congestion. Normally, whenever a segment loss occurs, it is assumed that the reason is congestion and congestion avoidance phase is activated by most congestion control algorithms. cwnd size increases linearly in congestion avoidance phase. In UAV Network, segment loss occurs mostly due to transient link instability and if congestion avoidance phase is activated each time , the average throughput is too low. This algorithm tries to identify the cause behind segment loss and activates slow start if the cause is transient link instability instead of heavy traffic. So the overall throughput is better than other congestion control algorithms.

2.1 Network Topologies Under Simulation

I designed two topologies. One is to create an environment with the case of transient link instability similar to the one mentioned in my paper. Another is for calculating Jain's fairness.

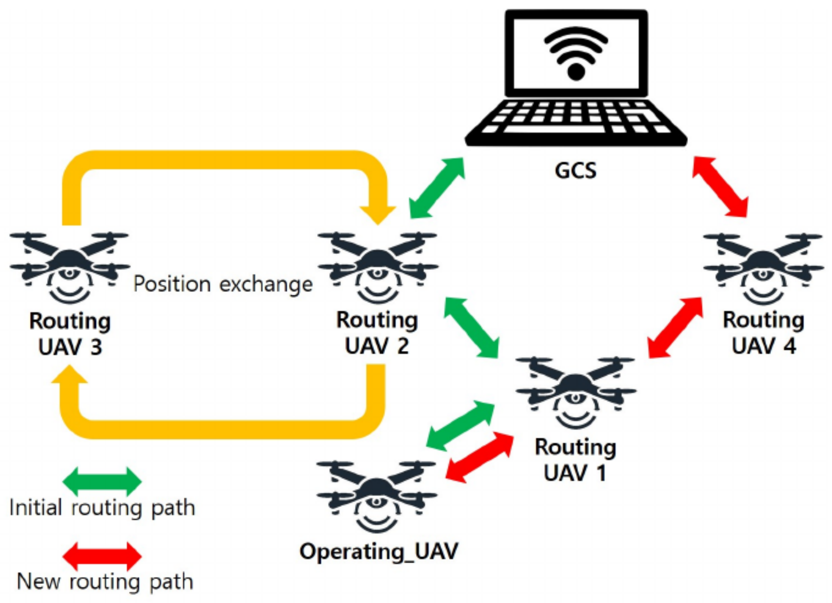


Figure 17: Simulation Topology of paper

A simulation experiment was conducted in my paper similar to 17 where GCS and Operating UAV exchanged data intially through the routing path GCS -> Routing UAV 2 -> Routing UAV 1 -> Operating UAV. After a few seconds, Routing UAV 2 and 3 exchanged their positions, so routing tables of GCS & Operating UAV get updated. During this change of transmission path, packets dropped and finally a new route is chosen for transmission of data , GCS -> Routing UAV 4 -> Routing UAV 1 -> Operating UAV. I tried to design the same topology with a few changes.

In my topology as showed in Figure 18, Node 0 (10.0.0.1) is the sender and Node 5 (10.0.0.6) is the receiver. To design a multihop network, I used RangePropagationLossModel and set the Max Range to 60. Using DSDV Routing,

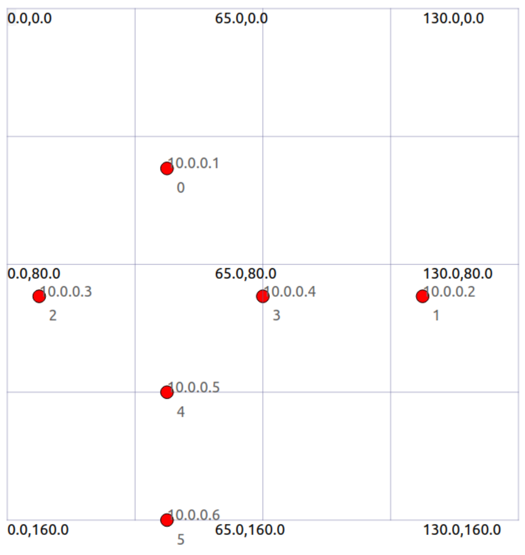


Figure 18: Initial Topology

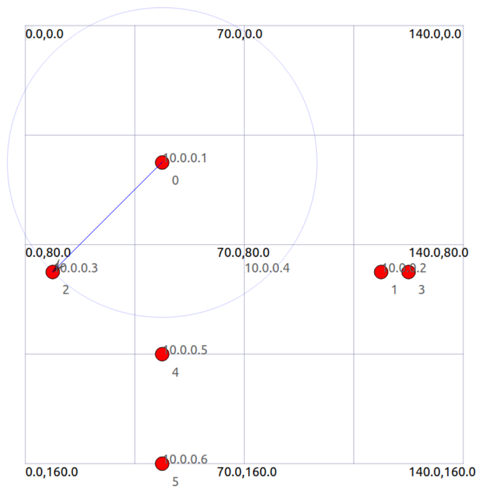


Figure 19: Topology after 9 seconds

intially packets are sent via the route 10.0.0.1 -> 10.0.0.4 -> 10.0.0.5 -> 10.0.0.6. After 9 seconds, I changed the position of Node 3 (10.0.0.4) to out of range from Node 0 (10.0.0.1) 19. So transmission path gets updated and finally packets are sent via the route 10.0.0.1 -> 10.0.0.3 -> 10.0.0.5 -> 10.0.0.6. I vailidated the change of transmission route by printing the routing tables before and after the position change of Node 3. All nodes are under a single network. Throughput was calculated after each 100 ms and Node 3 changed its position at 9.1 seconds.

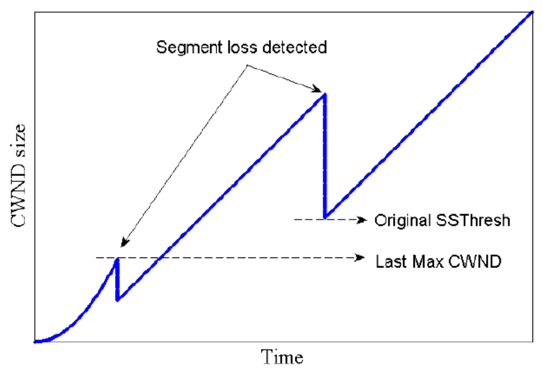
The topology mentioned was implemented in NS3.The brief value of the parameters used inside my simulator is described in Table 5.

Parameters	Values / Description
Packet size	1472
Data Rate	100 Mbps
Simulation Time	30
Range of each node	60.0
Wifi Standard	802.11 (5 GHz)
Wifi Mac	Adhoc
Mobility Model	ConstantPositionMobilityModel
Routing Algorithm	DSDV

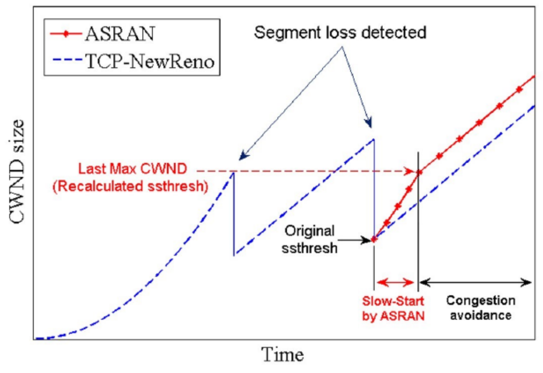
Table 5: Parameter values in simulation topology

2.2 Overview of proposed algorithm

ASRAN algorithm is an improvement on TcpNewReno. Each time a segment loss occurs, New Reno updates its ssthresh(slow start threshold) and cwnd size to half of current cwnd size. Now cwnd is equal to ssthresh, so congestion avoidance phase is activated. For ASRAN, it always stores the last maximum cwnd size prior to previous segment loss. It assumes this last_max_cwnd to be the available capacity of the network. It updates the ssthresh value by taking the maximum between last_max_cwnd and half of current cwnd size. If the last_max_cwnd is greater than half of the current cwnd size, it is set as the new ssthresh. Now after having the cwnd size, it becomes less than the ssthresh value and slow start phase is activated. As a result, cwnd size continues increasing exponentially which results in recovering lost throughput faster than TCPNewReno. Graphs for better understanding is given in 20.



(**b**) CWND fluctuation in buffer overflow scenario with *ASRAN*



(**a**) CWND fluctuation in *ASRAN* activated scenario

Figure 20: CWND fluctuation of ASRAN in transient link instability and buffer overflow scenerios

2.3 Parameters Under variation

The performance of ASRAN is compared with TCP Cubic and TCP NewReno in the above simulation topology and their throughputs were compared. Another network topology was built to compare the Jain's Fairness Index of these three algorithms.

2.4 Modifications made in the simulator

In order to implement ASRAN in NS3, a .cc & a .h files were added in src/internet/models named tcp-asran.cc and tcp-asran.h. To use this as a Congestion Control algorithm, necessary changes were made in the wscript file in the respective directory. The implementation of ASRAN is very similar to NewReno except for one function - getSSThresh(). This function is called everytime a segment is lost and the ssthresh value needs to be updated. Here each time, I took the maximum of the NewReno's ssthresh and last_max_cwnd and return it as the new threshold. The last_max_cwnd value is updated each time at the end of this function with the value of the current cwnd size. I did a slight modification from the paper and updated the last max cwnd value only when retransmission for duplicate acks occurs. While experimenting, I found that some packets are sent even after the connection gets disconnected due to transient link instability which is detected by retransmission timeout. These packets overall reduce the last_max_cwnd size as they always get lost. After this modification, my algorithm started performing better whenever a transient link drop occurs.

2.5 Results with graphs

The plots were generated following the mentioned simulation environment. Comparison between throughputs of New Reno, Cubic and ASRAN is given below.

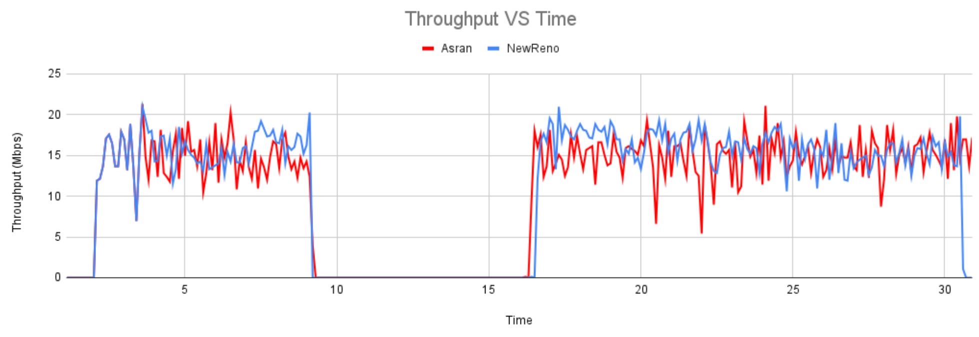


Figure 21: Throughput of TCP New Reno & ASRAN

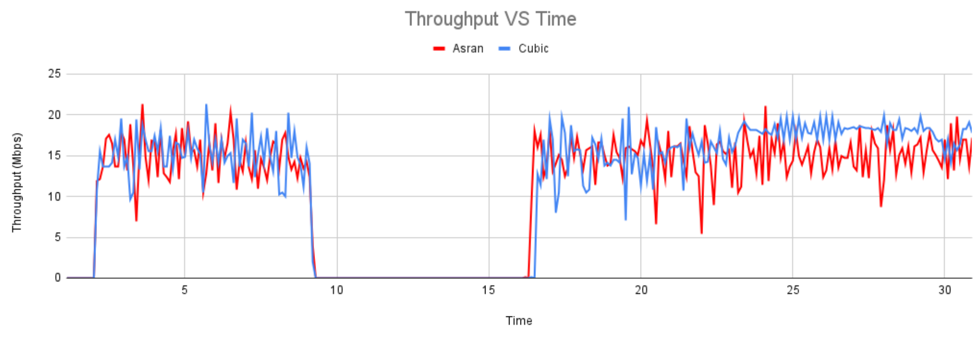


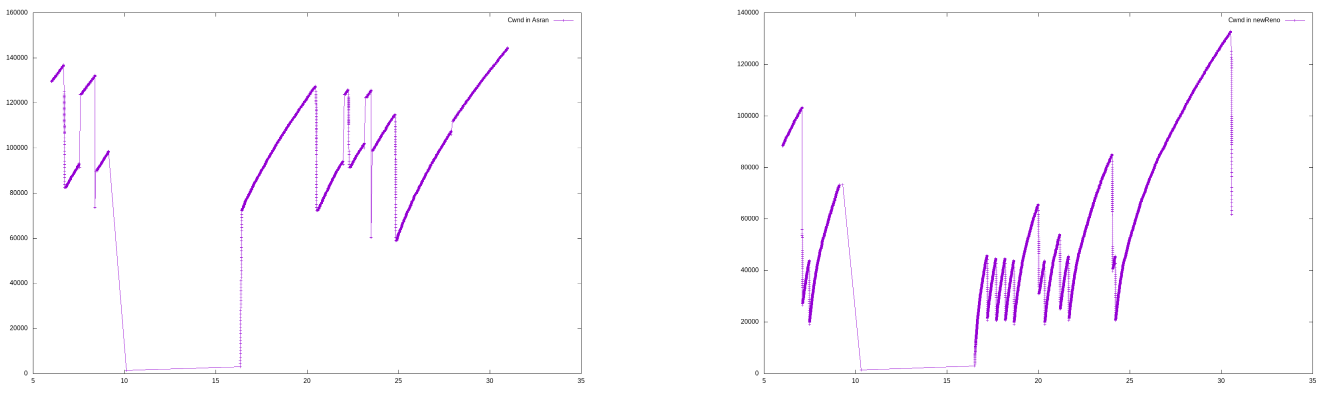
Figure 22: Throughput of TCP Cubic & ASRAN

2.6 Findings

From the graphs in 21 and 22, it is observed that the throughput is zero during the first two seconds and between 9 to 16 seconds. Initially it takes time for the routing tables to be updated. After 9 seconds, Node 3 changes its position so routing table gets updated again and it takes around 5 seconds for the routing tables to get fully updated.

From the graphs, we can see that ASRAN performed better in recovering lost throughput just after the node changed its position. It quickly got back to its previous CWND size. However, ASRAN fluctuates a lot compared to the other two algorithms. Due to its exponential rise most of the time, the buffer or the channel capacity gets full and packets are dropped more in general environment.

Though the paper mentioned that ASRAN provided better throughput than the other algorithms, my simulation couldn't provide a similar outcome. Since ASRAN increases exponentially most of the time, the buffer gets full and packets are dropped more often. The total sent packets are also much lesser than the other two algorithms mentioned. A summary of the performance of the 3 algorithms in my simulation environment is presented in Table 6. Metrics such as throughput and Jain's Fairness index was calculated for performance comparison.



(a) CWND Vs Time for ASRAN (b) CWND Vs Time for New Reno

Figure 23: CWND VS Time for both ASRAN & New Reno

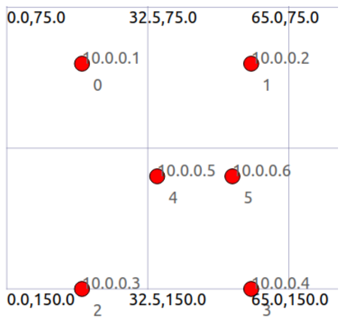


Figure 24: Simulation Topology for Fairness Calculation

There were two assumptions that has been made in the paper.

- Transient Link Instability : last_max_cwnd > current_cwnd / 2
- Buffer overflow : last_max_cwnd < current_cwnd / 2

By analyzing the results of the ssthreshold, cwnd size of my simulation, I understood that this conditions were not always met in my simulation. When transient link instability occured in 9 seconds, ASRAN performed better than the other two algorithms. But in general cases, in my simulation when packets were dropped randomly, ASRAN assumed more cases of transient link instability and activated slow start phase too often since many times last_max_cwnd were greater than current_cwnd / 2. As a result, in cases of buffer full, exponential rise of cwnd size caused more packets to be dropped. Maybe in real life scenerio those two assumptions are met. Also in cases with only and frequent transient link instabilities , ASRAN might perform better than the other algorithms.

Jain's Fairness index was measured between two congestion algorithms at a time. The topology was used similar to 24. Here Node 0, 2 send packets to Node 5 & Node 1, 3 send packets to Node 4. Node 0, 2 & 5 use one congestion control algorithm while Node 1, 3 & 4 use a different congestion control algorithm. While comparing fairness of TCP Cubic & ASRAN, fairness index of Cubic was 0.258838 & of ASRAN was 0.516844. While comparing between TCP New Reno & ASRAN, fairness index of ASRAN was & of New Reno was . This shows that ASRAN is more fair than the other two algorithms in a simulation environment.

Parameters	NewReno	ASRAN	Cubic
Average Throughput(Kbps)	6086.4	5889.16	6285.06
Total Sent Packets	43556	43135	45841
Total Received Packets	43480	42966	45717
Packet Delivery Ratio(%)	99.8255	99.6082	99.7295
End to End delay(ns)	1.88+07	2.90607+07	2.183+07

Table 6: Comparison of different Metrics

References

[1] Lee, Joon Yeop and Lee, Woonghee and Kim, Hyunsoon and Kim, Hwangnam. *"Adaptive TCP Transmission Adjustment for UAV Network Infrastructure"*