Computer Networks Laboratory

Assignment 3

Name: Anirban Das Class: BCSE-III Roll: 001910501077 Group: A3

Problem Statement:

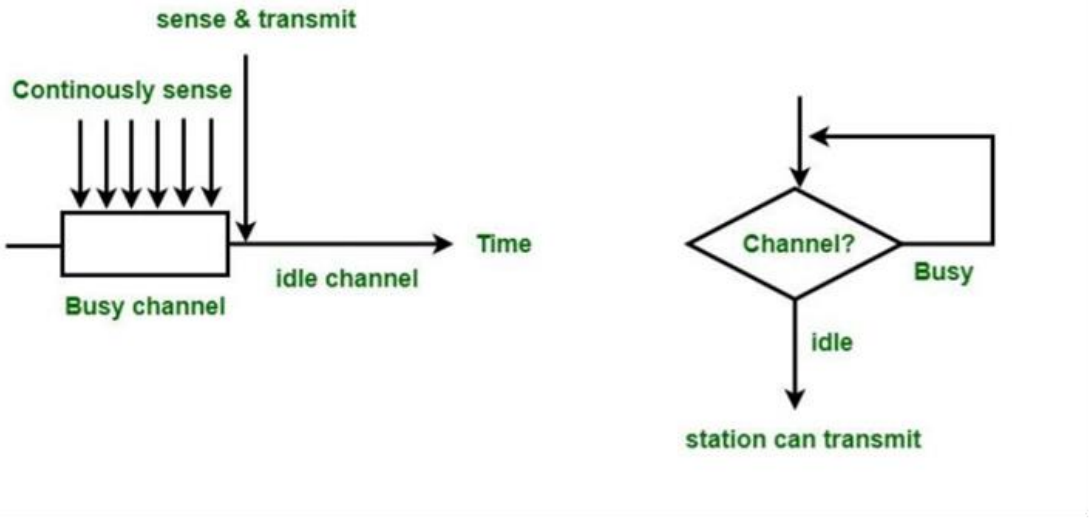
Implement 1-persistent, non-persistent and p-persistent CSMA techniques.

In this assignment, you have to implement 1-persistent, non-persistent and p-persistent CSMA techniques. Measure the performance parameters like throughput (i.e., average amount of data bits successfully transmitted per unit time) and forwarding delay (i.e., average end-to-end delay, including the queuing delay and the transmission delay) experienced by the CSMA frames (IEEE 802.3). Plot the comparison graphs for throughput and forwarding delay by varying p. State your observations on the impact of performance of different CSMA techniques.

Design:

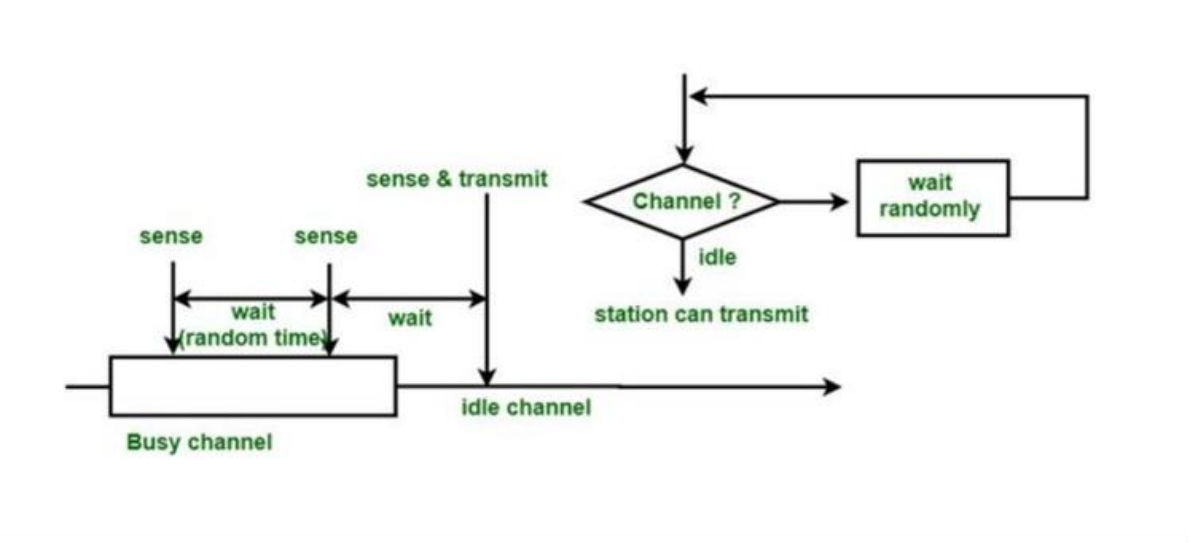
One-persistent CSMA:

In 1-persistent CSMA, the station continuously senses the channel to check its state i.e. idle or busy so that it can transfer data or not. In case when the channel is busy, the station will wait for the channel to become idle. When a station finds an idle channel, it transmits the frame to the channel without any delay. It transmits the frame with probability 1. Due to probability 1, it is called 1-persistent CSMA. The problem with this method is that there are a large number of chances for the collision because there is a chance when two or more stations found a channel in an idle state and the transmit frames at the same time. At the time when a collision occurs, the station has to wait for the random time for the channel to be idle and to start all again.



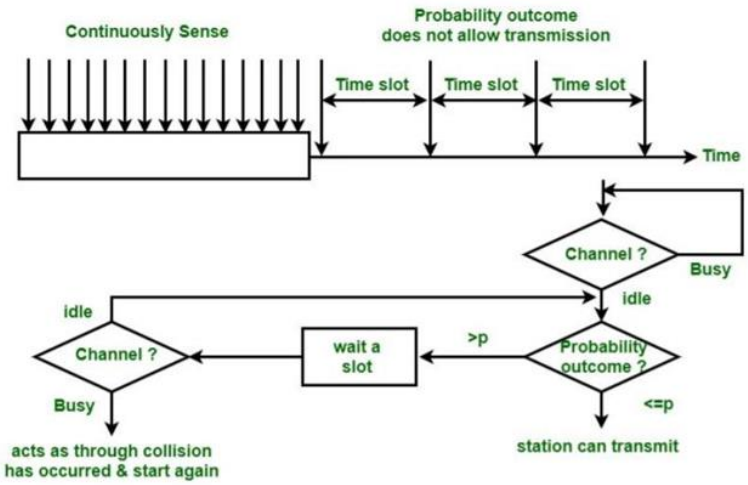
Non-persistent CSMA:

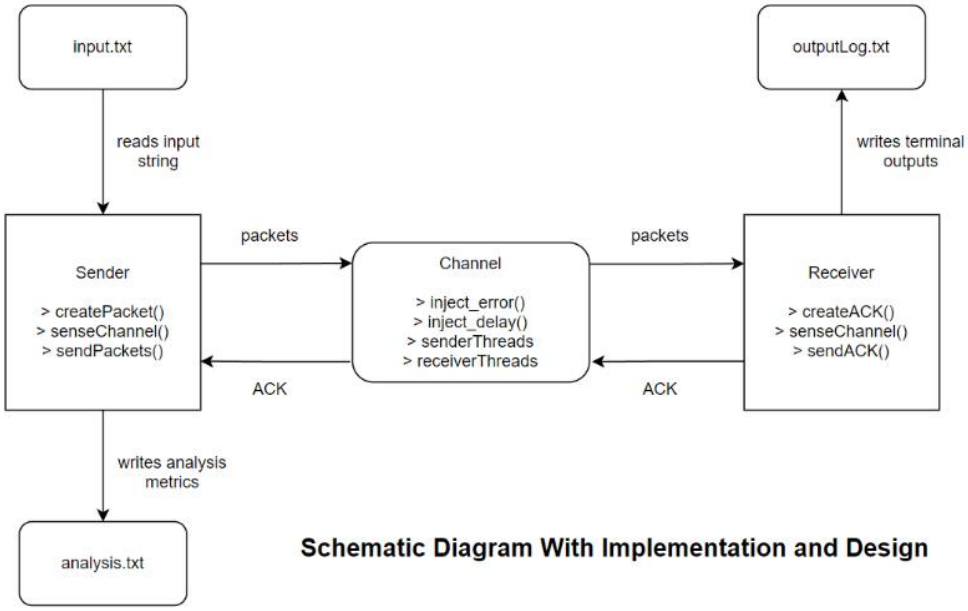
In this method, the station which has frames to send, only that station senses the channel. In case of an idle channel, it will send a frame immediately to that channel. In case when the channel is found busy, it will wait for the random time and again sense the state of the station whether idle or busy. In this method, the station does not immediately sense the channel for only the purpose of capturing it when it detects the end of the previous transmission. The main advantage of using this method is that it reduces the chances of collision. The problem with this is that it reduces the efficiency of the network.



P (1/n) -persistent CSMA:

This is the method that is used when the channel has time-slots and that time-slot duration is equal to or greater than the maximum propagation delay time. When the station is ready to send the frames, it will sense the channel. If the channel is found to be busy, the channel will wait for the next slot. If the channel is found to be idle, it transmits the frame with probability p, thus for the left probability i.e. q which is equal to 1-p the station will wait for the beginning of the next time slot. In case, when the next slot is also found idle it will transmit or wait again with the probabilities p and q. This process is repeated until either the frame gets transmitted or another station has started transmitting.

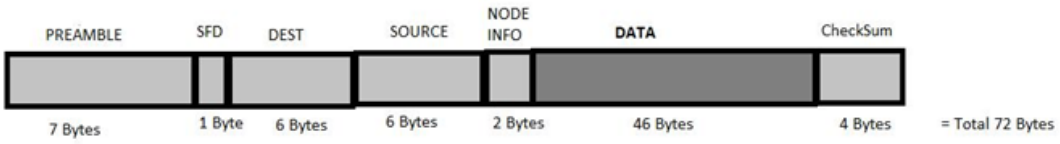




Schematic Diagram With Implementation and Design

Implementation:

We have used the IEEE 802.3 Ethernet Frame Format for our packets here. Size of a frame is 72 bytes.



Code Structure

- const.py :

  All the constants (number of threads, time-out, packet-size, etc) are defined here.

- checker.py :

  Contains functions for CheckSum generation and ErrorChecking.

- gen_packet.py :

  Packet class containing methods for generating packets maintaining IEEE 802.3 format, extracting the original data from the generated packets, decode the source and destination mac addresses, checking errors by CheckSum algorithm, etc.

- sender.py :

  Sender Class containing methods for selecting a receiver, reading data from the input file(s), selecting the sending method (one/non/p-persistent), adding the packets into stream/channel after sensing whether the channel is busy or not, etc.

- channel.py :

  Channel Class for channelizing packets from the sender, channelizing response from the receiver, starting the sender and receiving threads one by one in a queued manner.

- receiver.py :

  Receiver Class for extracting the data from the receiver packet, writing the confirmation into the output file, etc.

- main.py :

  Contains support for adding the sender and receiver connections to separate lists one by one, adding the sender and receiver threads to separate lists one by one, and starting the sending of packets based on the chosen option and generating a performance report.

- /textfiles :

  Folder containing input and output files, and the report file (written number of packets sent, received and total delay and calculated throughput).

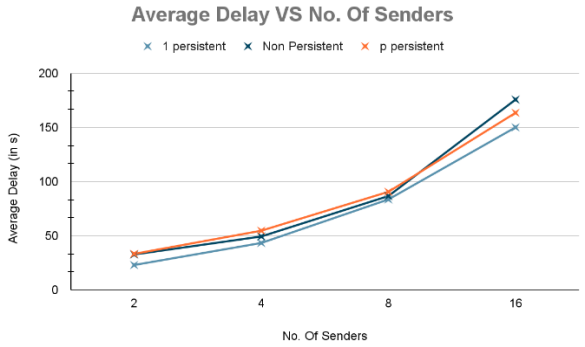
Results:

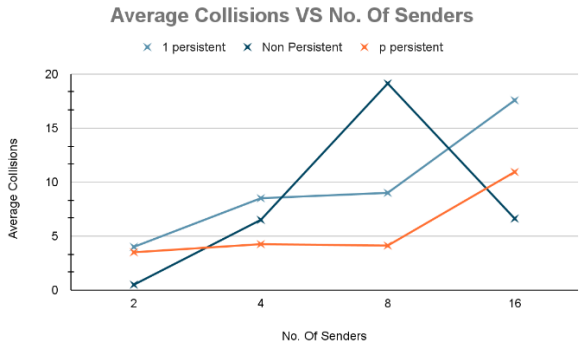
Setting the Channel to transmit data with a frame size of 72 bytes. It came out to be approximately 500 kbps.

- **Avg packets send from each sender = 13**
- **Throughput = (successfulPacketsTransmitted / totalDelay)**

TABLE 1

	Average Delay (in s)			Average Collisions			Average Throughput (in bps)		
CSMA Technique	**1 persistent**	**Non Persistent**	**p persistent**	**1 persistent**	**Non Persistent**	**p persistent**	**1 persistent**	**Non Persistent**	**p persistent**
No. Of Senders									
2	22.98	32.70	33.26	4	0.5	3.5	0.59	0.41	0.39
4	43.32	49.32	54.64	8.5	6.5	4.25	0.31	0.25	0.28
8	83.50	86.62	90.54	9	19.12	4.12	0.19	0.15	0.25
16	150	175.63	163.43	17.56	6.62	10.93	0.09	0.07	0.12





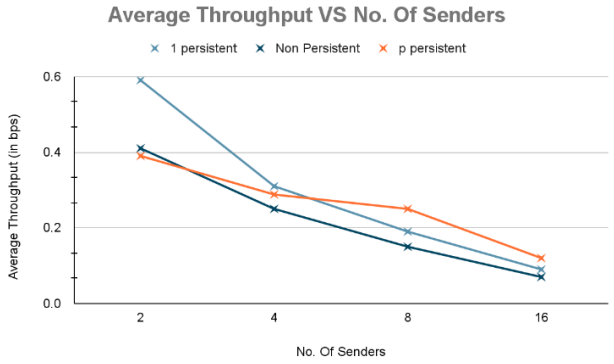


TABLE 2

CSMA Technique	Throughput Range (in bps)	Collisions Range
1 – Persistent	0.50	13.56
Non – Persistent	0.34	18.62
p - Persistent	**0.27**	**7.43**

Analysis:

When comparing Average Collision and Average Throughput, **non-persistent CSMA** performs better than both of the rest two methods for small number of senders. As the number of senders increase, **p-persistent CSMA** shows better results compared to the other two.

TABLE 2 shows the Range of values for the above metrics. It is clear that **p-persistent** is a reliable CSMA technique for flow control because of its less distributed stats for different number of senders.

However, in the case of Average Delay comparison, non-persistent and p-persistent CSMA techniques show huge delay compared to **1-persistent CSMA**.

All the above conclusions are depicted in the charts given earlier.

Improvements:

In a relatively small network, the number of nodes falls somewhere between 10-30. Considering more data points at a higher number of nodes would give better analysis. Regarding **non-persistent** CSMA varying the range of random sleep would have been helpful.