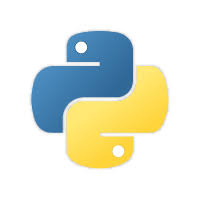
Department of (ECE) & (CS), Florida Institute of Technology, Melbourne, Florida



Project 2

Performance Evaluation of MAC Protocols

(Using Python)



ECE 5534 – Computer Networks

7th December 2018

By: Ravi Pandhi Prof: Dr. Chul-Ho Lee

ID: 9035381742

By: Adolf A D’costa

ID: 9035384040

Index

1. Project Goal……………………………………………………..3
2. Results Archived……………..………………………………....3
3. Pre-requisites to run the code…………………………………...4
4. Code Description………………………………………………..4
5. Calculations Performed…………………………………………5
6. GUI……………………………………………………………...6
7. Simulation 1.……………………………………………………6
8. Simulation 2…………………………………………………….7
9. Simulation 3…………………………………………………….9
10. Reference…………………………………………………….12

Goal –

Objective of this project was to reinforce your understanding on medium access control (MAC) protocols and let you have more visible experience through the design and implementation of a custom-made network simulator with focus on the performance evaluation of MAC protocols. The MAC protocol of interest in this project is 1-persistent CSMA/CD with Binary Exponential Back off. Project not based on the off-the-shelf network simulators such as ns-2, ns-3, OPNET, and OMNeT++. While you can still refer to the existing simulators, ultimately you must implement your own discrete-event network simulator from scratch.

Results Archived –

1. Frame generation
2. Frame loss
3. Frame awaiting
4. Successful frame transmission
5. Throughput
6. Channel utilization
7. Channel waste
8. Retransmission overhead
9. Delay
10. Graphical User Interface
11. User-friendly interface or graphical user interface in real-time
12. Addition Statistics (Total Number of Collisions).

Pre-Requisites—

Python IDE.

“Driver File.py” is the file to run in python

Supporting python files needed in the same folder for execution of the code is

(Gui.py, Node.py, Packet.py, networking.png)

Description of Code –

This Code is based on Real-Time execution. The concept of the code is there is a main outer “For Loop” this loop is based on the time slots and multiple computations and processes are carried out in parallel, in that time domain. For example, there is packet generation if the node wins the probability, packets are sent to the buffer and dropped if the buffer is full a count is kept on the amount of packets that are dropped, One or Multiple Node could wins the probability to transmit over the channel, if there is a successful transmission then the counter keep a track of “Successful transmission”, if there is a collision that occurs, the packets is regenerated at the node and the “Collision” counter is used to keep a track of it. The node now waits for a random amount of time slots to re-transmit. All other parameters are computed depending on the above iterations performed.

“Rand” function is used in python to generate a random number and its packet generation probability to win is checked depending on the user input probability value. Random function “rand” is also used to provide a random wait time for the node that is collided, before it can re-transmit again.

“tkinter” library was used for coding the GUI for this program. It gives a user-friendly window where the user can provide an input and the output is given on the same window on the click of “Simulate” button.

“float” and “double” functions are used in python to provide the precision needed for computation of number that have decimals.

Various classes were made in the system “Node” & “Packet” these classes were made so that the code could be systematic so that it could be understood well as well as helps in the smooth execution and transmission of the program.

“Deque” data structure was used in the program as a buffer to store the Packet Class, it works on the principle FIFO (First in First Out). When ever the node won the probability to transmit the pop left function will be used to pop the Data ie packet from the buffer and transmit it over the channel.

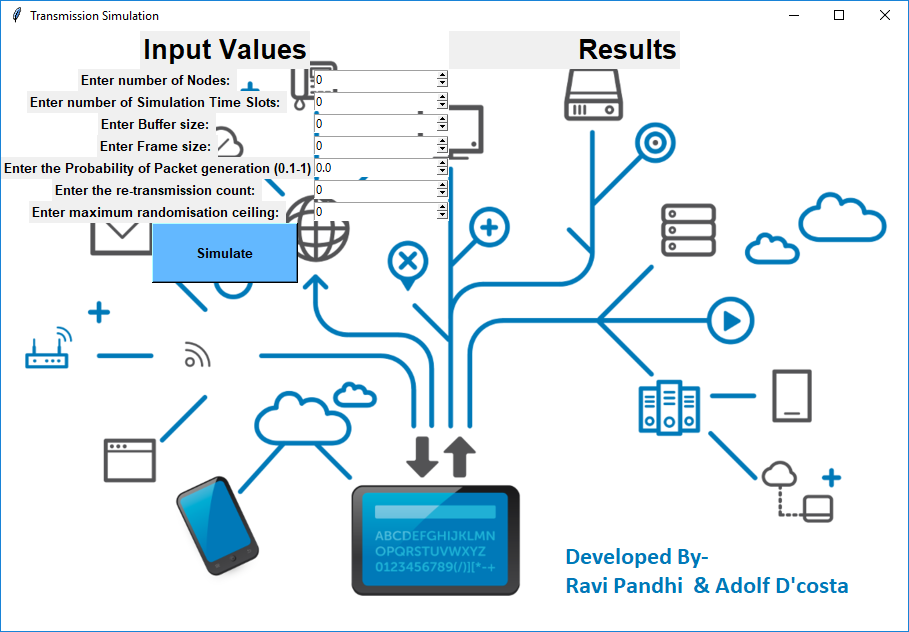
(Detailed explanation of the Code is given as Comments in the Python File)

Calculations Performed –

1. System Throughput =
2. Channel Utilization =
3. Channel Waste =
4. Re-transmission Overhead =
5. Average Delay =

GUI—

Background image denotes networking components [1]



Simulation 1 —

Basic network setup output as mentioned in the project description

Nodes = 6

Frame generated Probability = 0.1

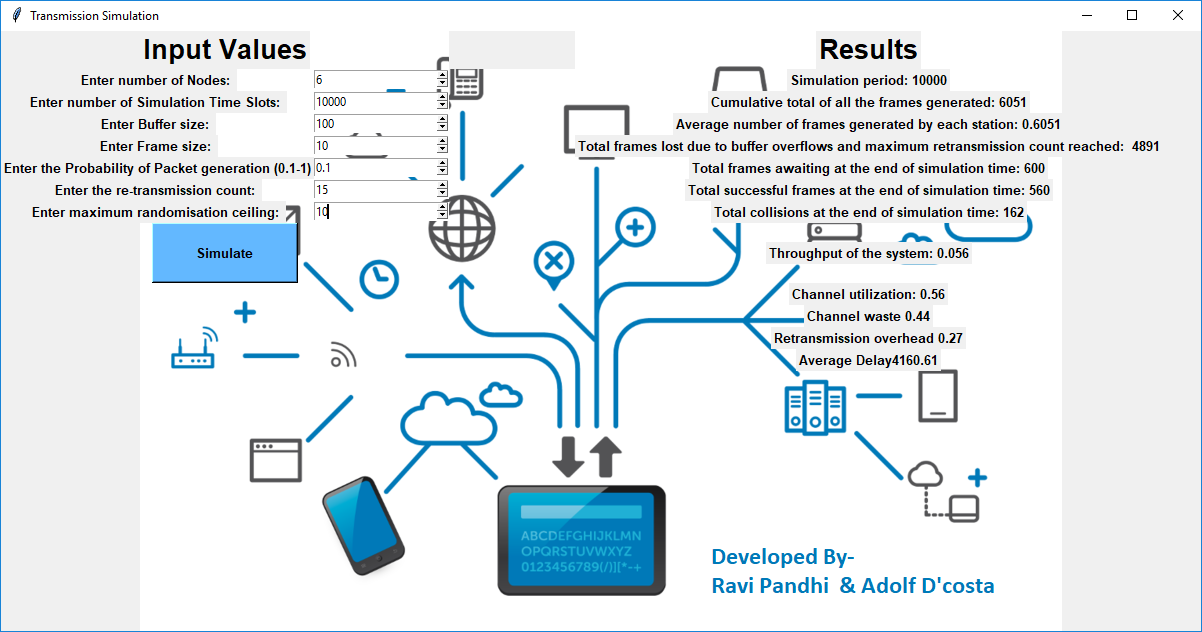
Frame size = 10

Re-Transmission count = 15

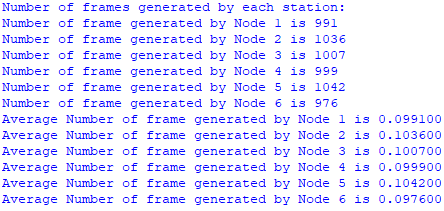
Randomization ceiling = 10

Simulation time slots = 10000

Buffer size = 100



In Terminal –



Simulation 2 —

Basic network setup output as mentioned in the project description

Nodes = 10

Frame generated Probability = 0.20

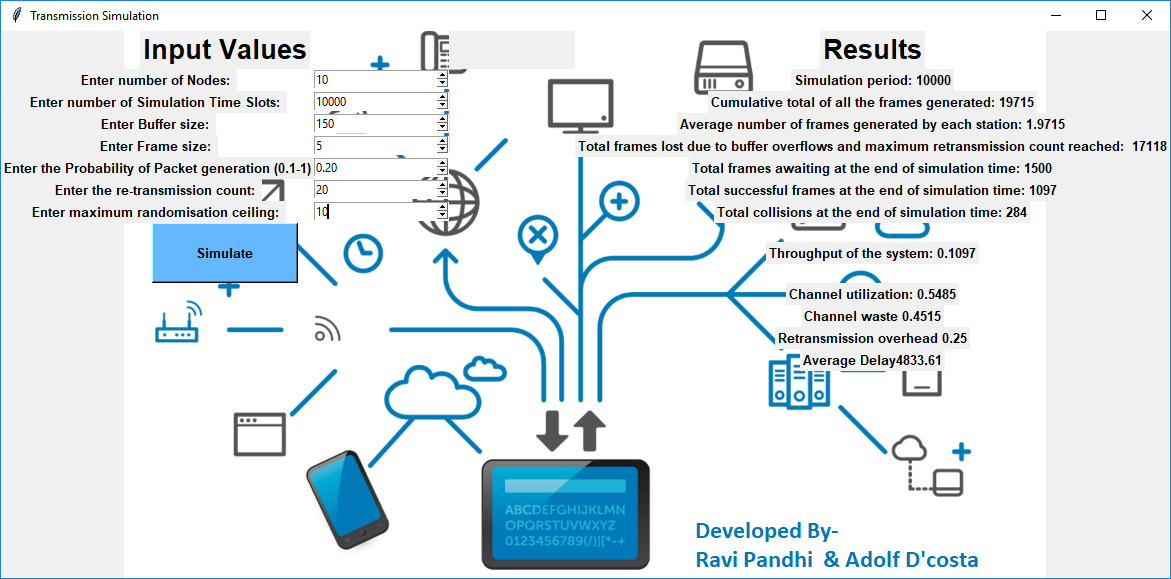
Frame size = 5

Re-Transmission count = 20

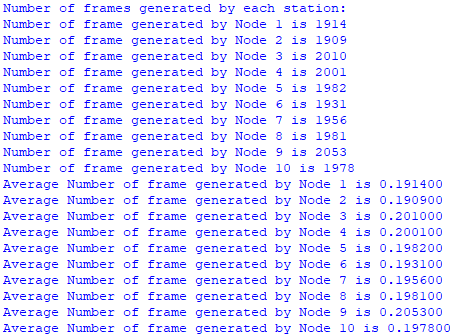
Randomization ceiling = 10

Simulation time slots = 10000

Buffer size = 150



In Terminal –



Simulation 3 —

Nodes = 50

Frame generated Probability = 0.3

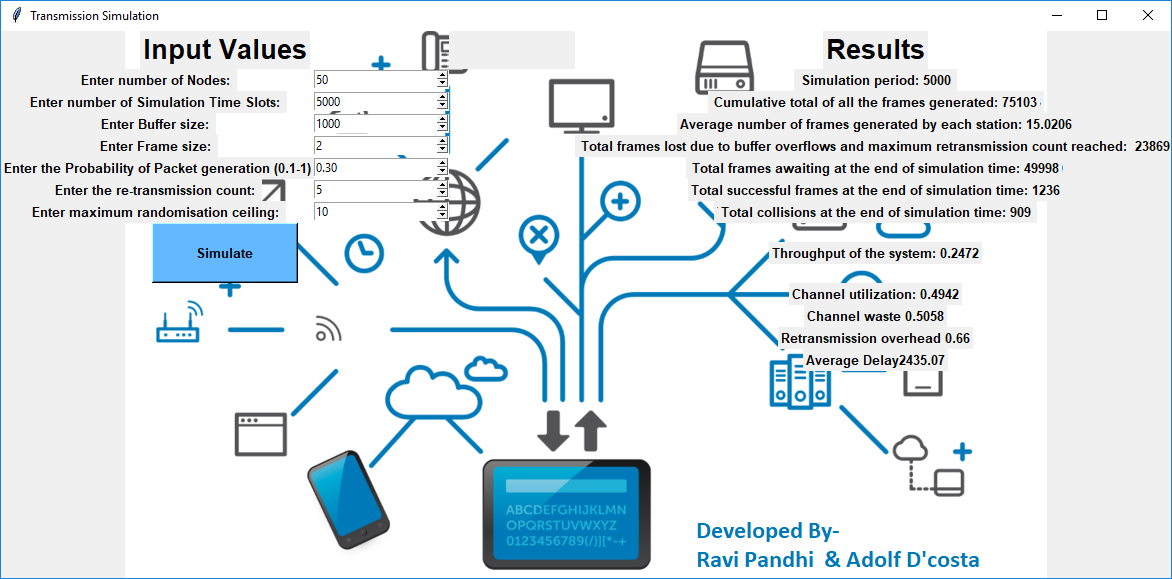
Frame size = 2

Re-Transmission count = 5

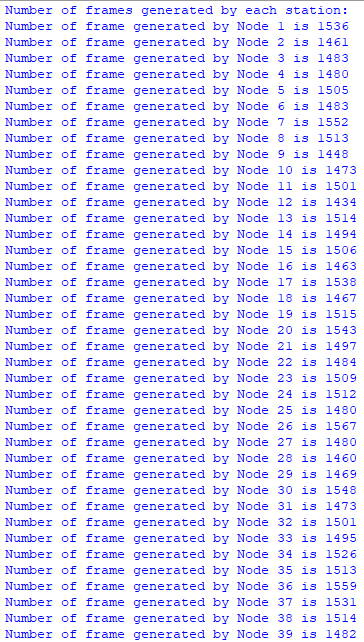
Randomization ceiling = 10

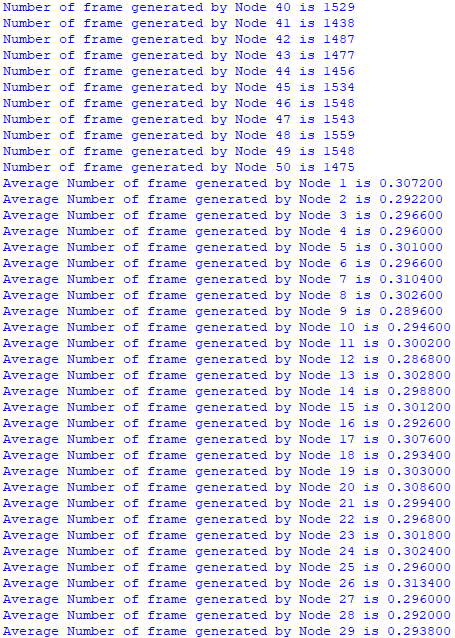
Simulation time slots = 5000

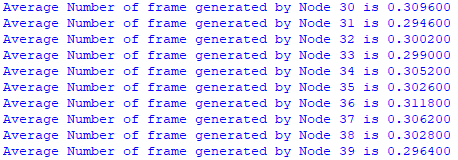
Buffer size = 100

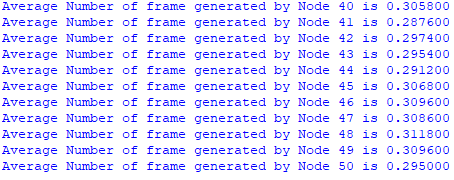


In Terminal –









References:

“WeisserAdler.” WeisserAdler|NetworkInfrastructure,weisseradlerng.co m/infra.php.[1]