“Backoff Protocol”
Simulation Report

Simulation Setup:

The following assumptions were made to setup the simulation: Each slot takes a discretized unit time slot. So, if a window has 3 size then that means it has three slots and if there is no collision then three unique users can be accommodated inside it at one instance. All the users are equidistant from each other, that also means that there is same transmission delay for all the users. That transmission delay is also not considered based on the distance between the users. The packet size is uniform for all users. Also the inner details of how a packet is formed and then transmitted and received are not included in this simulation (as it was mentioned in the project sheet). Each user transmits same sized data with same transmission time hence taking one available slot to communicate by sending the data and then going to sleep mode.
Initially a random time value is given to all the users. That value is selected from a small range (which is same for all iterations and protocols) so as to observe what would happened during collision. If during an iteration two or more users have the same time for transmission then a collision is detected (which is displayed during the execution of the simulation as well). Those users which have non unique transmission times are marked. Then a random backoff time is allocated to each of those marked users. The random backoff time is taken randomly from the following range: 0 to (2^n-1). Here “n” is the number of collision entities competing for a particular slot at a particular instance. Also if there is no user can successfully compete in the slot selected due to collision then that slot is reset and then the protocol starts from the subsequent window. This is done so that a wait is provided to compensate for the random backoff time. During their random backoff waiting other users are selected if they have unique transmission times and if another collision is detected then the marked users are again given random backoff time based on the range mentioned above.
Also within each protocol the first window starts with 2 time slots (as given).
A slot is selected by a user if there is no collision and that user has a unique transmission time (i.e in that time there is no other user is transmitting). If a user has the smallest unique value from the available unmarked users then that user is selected. After selection it is meant that it has successfully transmitted in that slot then that user would be not be tracked further during the rest of the simulation.

Results:

The simulation results were obtained by iterating each number of users (N) 10 times each. The 10 iterations per each user group were then averaged out to eliminate the outliers (generated due to randomization). Total 6000 users (N=6000) were used in the simulation in a step wise manner, i.e 60 main iterations were done starting from N=100 and going towards N=6000 with a 100 increment. The resulting latency values are stored in the three text files for each protocol. Moreover, an array is used to keep track of the wasted slots that were affected due to collision between two or more users using the same

transmission time. If the last user has used slot while the window still has more free slots then those free slots are not included in the latency calculations as they are not used by any user because there is no user remaining to transmit a data. Since the windows has slots and this simulation is done in a discretized time manner so the size of the windows or the total number of the windows used by all the users equate to the time is takes for the complete number of users to transmit their data in the medium.

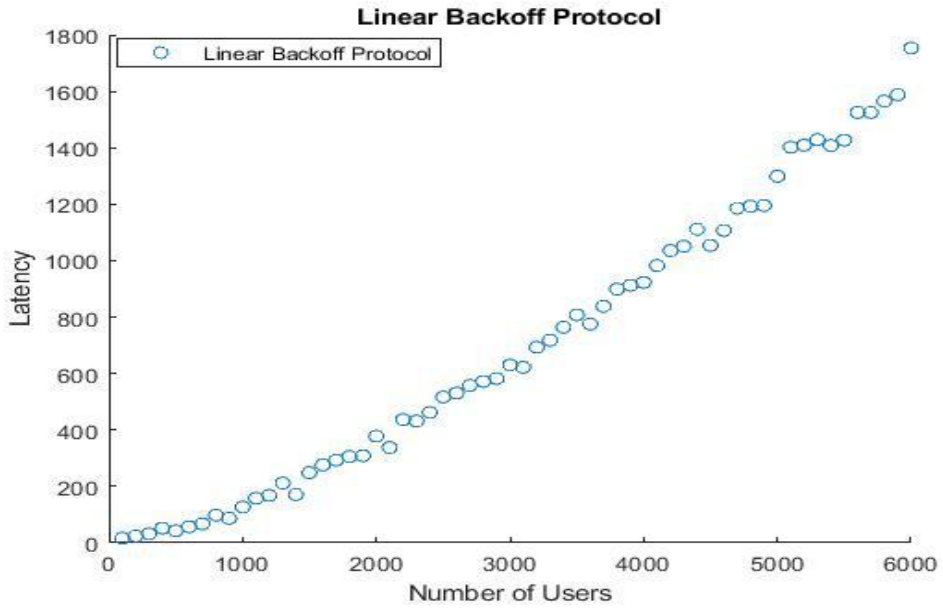

Figure 1: Latency Graph for Liner Backoff Protocol

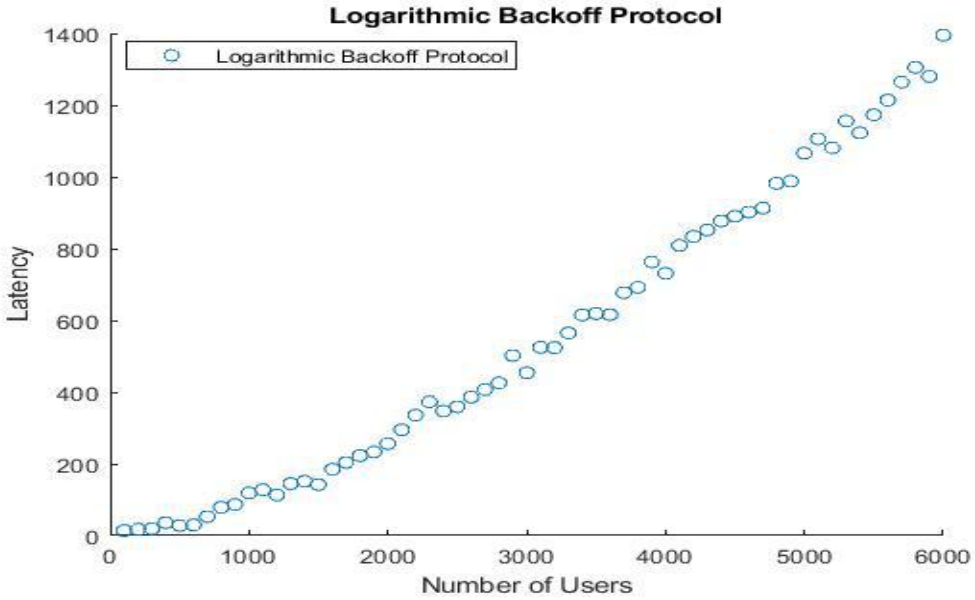

Figure 2: Latency Graph for Logarithmic Backoff Protocol

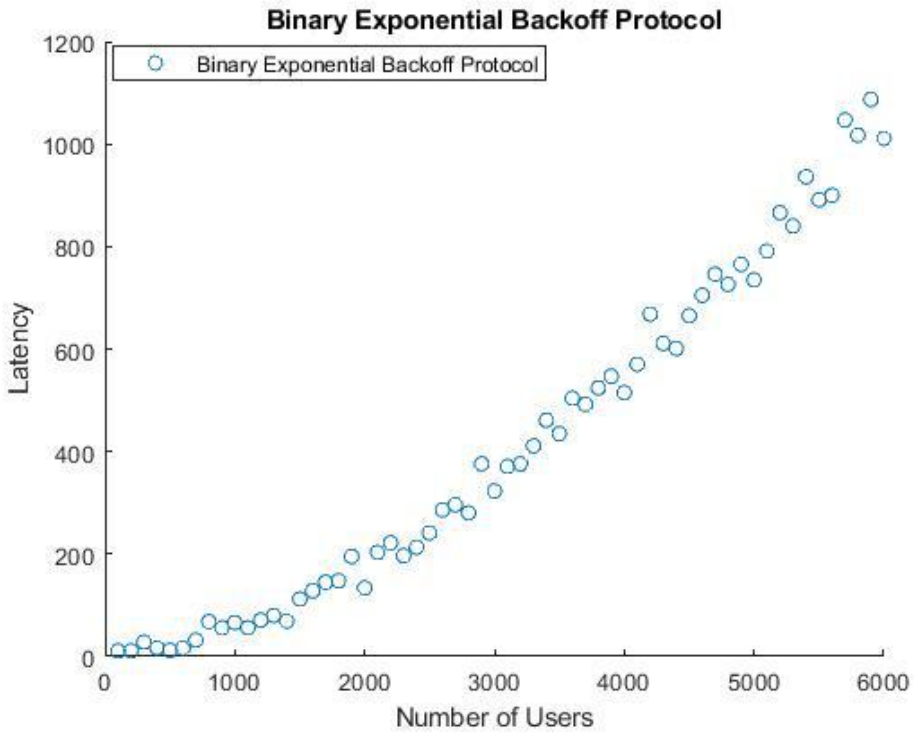

Figure 3: Latency Graph for Binary Exponential Backoff Protocol

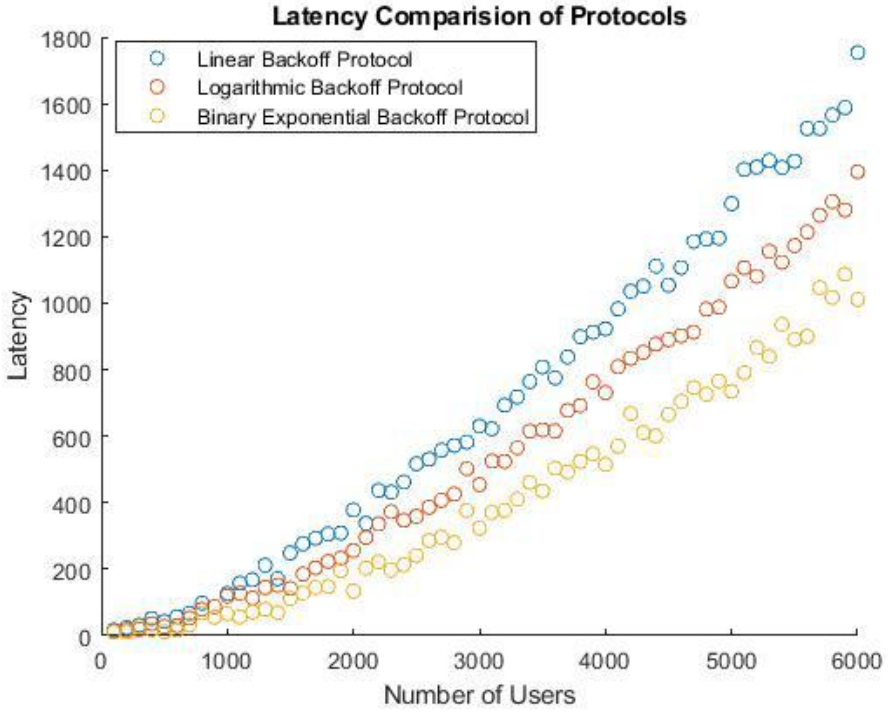

Figure 4: Latency Graph Comparisons for Liner, Log, and Exponential Backoff Protocols

Analysis:

(A) Which backoff protocol appears to give the best (lowest) average latency value as the number of devices tends to infinity?

According to the graphs 3 and 4 it can be seen that the best (lowest) average latency is that of the Binary Exponential backoff protocol. This protocol is very fast to give access to the users because at a particular instance a large number of users can compete for the data transmission after their collision has been resolved due to random backoff time. The window size in this protocol increases exponentially, so the number of users catered by each subsequent window increases exponentially as well.

(B) Which backoff protocol appears to give the second best (lowest) average latency value as the number of devices tends to infinity?

According to graphs 2 and 4 it is shown that the second best latency is that of Logarithm backoff protocol. This is faster than Linear backoff protocol but slower than Binary Exponential backof protocol. For smaller number of users the difference between the latency of Logarithmic backoff protocol and Linear backoff protocol is quite similar. Though they change significantly for larger number of users. Even though the window size increases in an almost exponential manner but still the number of users accommodated per window, by this protocol is still lower then that of the Binary Exponential backoff protocol.

(C) Which backoff protocol appears to give the third best (lowest) average latency value as the number of devices tends to infinity?

According to graphs 1 and 4 it can observed that the third best average latency is that of the Linear backoff protocol. This is due to the fact the increment in the window size is very small. It linearly increases with only an addition of one slot per window. So for large number of users there would be a large number of latency value because the windows do not have enough size to accommodate them easily in a fast time. They would accommodate the large number of users but it would be more time consuming.

"Binary Exponential Backoff Protocol is Best": Justification:

Consider the following range for window sizes with respect to how many unique users can be accommodated in the slots.

Table 1: Window Size Comparison

	W_1	W_2	W_3	W_4	W_5	W_6	W_7	W_8	W_9
Linear	2	3	4	5	6	7	8	9	10
Log	2	4	6	8	10	13	16	20	24
Exp	2	4	6	16	32	64	128	256	512

Now consider that there are 40 users (N=40) that wish to communicate via these slots. Considering that all have unique time slots and there is no collision between them then it would take W_8 to complete the communications for all the users using Linear backoff

protocol. Whereas it would take W_5 to complete the same communication for the all the users using Binary Exponential backoff protocol. Also, it would take W_6 to complete the same communication for the all the users using Logarithmic backoff protocol. Here it can be seen that Binary Exponential backoff protocol can accommodate more users per window as compared to the other two backoff protocols. This is due to the fact that the window size increases in Binary Exponential backoff protocol increases exponentially, which is by default higher than window size increase methods in the other two backoff protocols.

Also a noteworthy point here is that with increasing window size the difference between logarithmic backoff protocol and linear protocol is even more significant. Whereas, the difference between Binary Exponential backoff protocol is different than the other two backoff protocols even from the start.