ECE3141 Project – Simulation of a Medium Access Control Protocol (Slotted ALOHA)

Prepared By – Paarth Bhasin and Jason Wijaya

**INTRODUCTION**

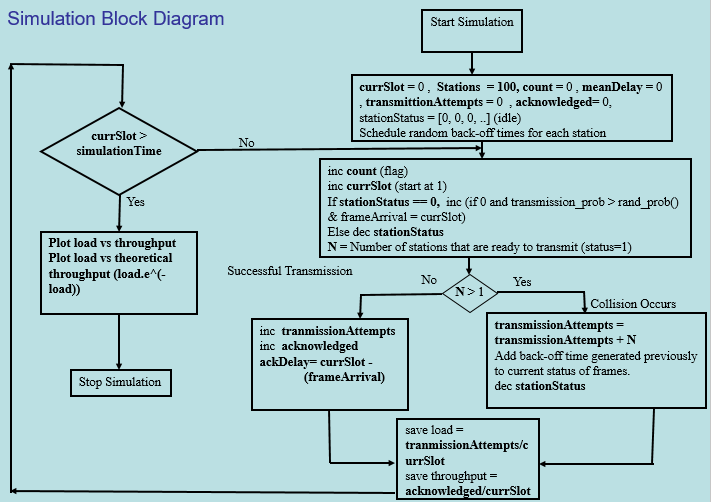
In this mini project, we tried to simulate one of the popularly used MAC protocols (Slotted ALOHA) and then compare the results we got to the what we studied in the class or what the general theoretical result is. Our results were very close to the theoretical one’s in some of the experiments we conducted, while it was not so close in the other ones.

**EXPERIMENT**

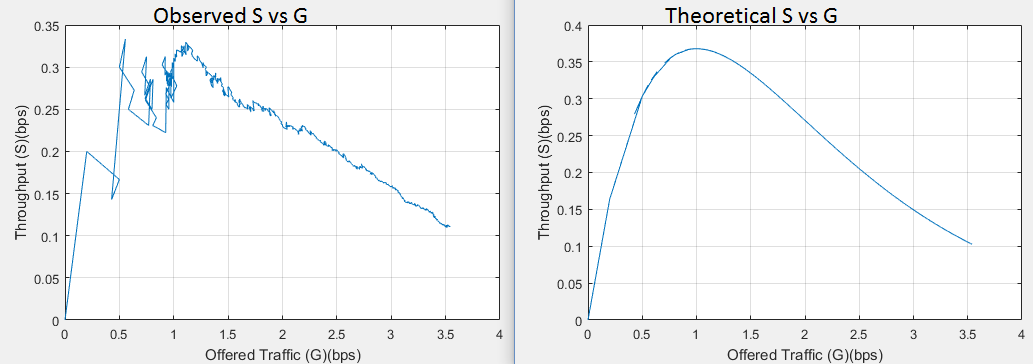
We simulated the Slotted ALOHA protocol using MATLAB for a specific simulation time. The idea behind the program is to run for a certain time interval, and during that interval we analyse the Slotted ALOHA protocol. Once the program finishes executing, we have the results of Load (G) and the corresponding Throughput (S). We calculate two different throughputs during the program, one was for what we observed and the other was for what it should have been as per the widely-known formula:

**S =** **G x e-G**

The block diagram is shown as follows:



The graphs for S vs G for both the simulation results and the expected results are shown below:



**DISCUSSION & RESULTS:**

The results we got were very close to each other but still not the same, since the theoretical results is the best possible throughput we can get at any instance. In reality, the experimental probability of transmitting a frame successfully, differs from the theoretical probability (but approaches it on a large sample). This is the reason why we can come close to resembling the ideal behaviour but never be the same.

The two graphs are very similar in their shape and values, which shows that the slotted aloha algorithm follows the theoretical results to a large extent.

The following values were used as input to reach this accurate result:

*N (number of stations) = 200, Simulation time/Number of Time Slots (5000 = 0.5s), Probability of transmission = 0.0057, Maximum Back-off time = 100 (time-slots)*

In both graphs, there is a optimum point in the graph where throughput becomes the highest (0.34 ~ 0.36 ). Increasing the traffic/load (G) after this point in the graph decreases the throughput and increases the delay further, making the efficiency worse.

**REFERENCES:**

**Papers:**

[1] M. SamiM.Yousef, H. A. Elsayed and A. Zekry, "Design and Simulation of Random Access Procedure in LTE", International Journal of Computer Applications, vol. 110, no. 16, pp. 16-22, 2015.

[2] N. Sidhu, P. Singh and S. Rani, "Improved Optimal Slotted CSMA/CA Protocol", *International Journal of Computer Applications*, vol. 79, no. 6, pp. 43-47, 2013.

**Web Links:**

[3] 2017. [Online]. Available: http://web.mit.edu/modiano/www/6.263/lec10.pdf. [Accessed: 09- May- 2017].

[4] 2017. [Online]. Available: http://protocols.netlab.uky.edu/~calvert/classes/571/lectureslides/MAC-Intro.pdf. [Accessed: 09- May- 2017].

**APPENDIX**

**MATLAB Source Code (Simulation algorithm based on: http://au.mathworks.com/matlabcentral/fileexchange/3834-home-networking-basis?focused=5049813&tab=function):**

**clc; % Clear the console window**

**stations = 200;**

**tranmissionProb = 0.0057;**

**max\_backoff = 100;**

**simulationTime = 5000;**

**stationStatus = zeros(1,stations); % Initially all stations are idle**

**% 0: Station has no frame to be transmitted (idle)**

**% 1: Station has a frame to be transmitted, either because new data must be**

**% sent or a previously collided packet has waited the backoff time**

**% >1: source is backlogged due to previous packets collision, the value of**

**% the status equals the number of slots it must wait for the next transmission attempt**

**stationBackoffTime = zeros(1,stations); % Random amount of time station must backoff to resolve collision**

**transmissionAttempts = 0; % Number of transmission attempts in total of all stations**

**acknowledgedDelay = zeros(1,simulationTime);**

**acknowledgedFrameCount = 0; % Total number of frames that have been successfully transmitted.**

**frameCollisions = 0; % Total number of collisions that have occured till now.**

**frameCreationTime = zeros(1,stations); % Time (t) when frame was generated**

**currentTimeSlot = 0; % We start at time t = 1 to t = 5000**

**count = 0; % flag to keep track of current load and throughput for each time slot**

**x1 = zeros(1, stations); % Load (G) at different timeslots**

**y1 = zeros(1, stations); % Throughput (S) at the corresponding Load.**

**% We only run for a finite/simulated time, to study Slotted Aloha protocol.**

**% In reality this is an infinite never-ending loop.**

**while currentTimeSlot < simulationTime**

**currentTimeSlot = currentTimeSlot + 1; % Increment time slot**

**for currStation = 1:length(stationStatus) % For all stations**

**if stationStatus(1,currStation) == 0 && rand(1) <= transmissionProb % new frame generated if conditions met**

**stationStatus(1,currStation) = 1; % Ready to be transmitted**

**stationBackoffTime(1,currStation) = randi(max\_backoff,1); % Backoff time in case collision occurs**

**frameCreationTime(1,currStation) = currentTimeSlot; % Storing the creation time of this frame**

**elseif stationStatus(1,currStation) == 1 % backlogged packet**

**stationBackoffTime(1,currStation) = randi(max\_backoff,1);**

**end**

**end**

**transmissionAttempts = transmissionAttempts + sum(stationStatus == 1);**

**if sum(stationStatus == 1) == 1 % Frame can successfully be transmitted without collision**

**% We transmit only if no collisions are happening, in other words no other station is**

**% transmitting in current time slot.**

**acknowledgedFrameCount = acknowledgedFrameCount + 1;**

**[~,stationId] = find(stationStatus == 1); % Extract the station which is transmitting**

**acknowledgedDelay(acknowledgedFrameCount) = currentTimeSlot - frameCreationTime(stationId);**

**elseif sum(stationStatus == 1) > 1 % More than one stations are trying to transmit at the same time**

**% Results in collision**

**frameCollisions = frameCollisions + 1; % Incrementing the collisions occured**

**stationStatus = stationStatus + stationBackoffTime;**

**end**

**stationStatus = stationStatus - 1; % Decrease the backoff time. Case 3 becomes negative**

**stationStatus(stationStatus < 0) = 0; % idle sources stay idle (only permitted status values allowed)**

**stationBackoffTime = zeros(1,stations); % Reset all backoff time values as they have been recorded**

**loadPresent = transmissionAttempts / currentTimeSlot; % Load (G)**

**if acknowledgedFrameCount == 0**

**meanDelay = simulationTime;**

**else**

**meanDelay = mean(acknowledgedDelay(1:acknowledgedFrameCount));**

**% Calculate the mean delay from the delays of each individual acknowledged frames.**

**end**

**throughput = acknowledgedFrameCount / currentTimeSlot; % Normalized thoughput**

**collisionProb = frameCollisions / currentTimeSlot; % Normalized collision probability of each frame**

**count = count + 1; % Incrementing flag counter**

**x1(1, count) = loadPresent; %Load**

**y1(1, count) = throughput; %Throughput on given load**

**end**

**loadPresent = (transmissionAttempts / currentTimeSlot); % Final normalized load**

**% Final average delay/time for being acknowledged.**

**if acknowledgedFrameCount == 0**

**meanDelay = simulationTime;**

**else**

**meanDelay = mean(acknowledgedDelay(1:acknowledgedFrameCount));**

**end**

**throughput = (acknowledgedFrameCount / currentTimeSlot); % Final Normalized throughput**

**collisionProb = (frameCollisions / currentTimeSlot); % Final normalized collision probability**

**fprintf('\nLoad/Channel Traffic (G): %.3f,\nThroughput (S): %.3f,\nMean delay (D): %.2f slots,\nCollision probability (p): %.3f.\n',loadPresent,throughput,meanDelay,collisionProb);**

**end**

**w3=x1.\*exp(-x1); % Theoretical value of throughput**

**figure(2) % First figure/graph**

**plot(x1, y1, '-')**

**title('Slotted Aloha - Observed S vs G graph ')**

**grid**

**xlabel('Offered Traffic (G)(bps)')**

**ylabel('Throughput (S)(bps)')**

**figure; % Second figure/graph**

**plot(x1, w3, '-') % Observed value of throughput**

**title('Slotted Aloha - Theoretical S vs G graph ')**

**grid**

**xlabel('Offered Traffic (G)(bps)')**

**ylabel('Throughput (S)(bps)')**