ECE 478

University of Arizona

Project 1:

Simulating the Distributed Coordination Function (DCF) of 802.11

Kai Ward

10/24/23

Introduction:

The purpose of this report is to present the findings of a simulation study aimed at understanding the performance of multiple access protocols in a wireless setting through event-driven simulation. The main focus of this project was to evaluate the Distributed Coordination Function (DCF) of 802.11, keeping in mind 2 different network topologies, single collision domain and hidden terminals. Through this study I aimed to uncover the behaviors and efficiency of these two protocols in different conditions with special attention to throughput, collision occurrences, and fairness between transmitting stations.

I undertook this project independently in all parts of the simulation - designing and developing the simulation model in python, analyzing the results, drawing conclusions, and depicting the data. This report details the methodology used for creating the simulations, presenting the results through various graphs, and provides an interpretation of these results.

Simulations:

The simulation was built using python and implemented a way to step through each slot and display the time stamps when they occurred. The parameters for the simulation are listed below:

3 Simulation parameters

Parameter	Value	Parameter	Value
Data frame size	1,500 bytes	ACK, RTS, CTS size	3 slots
Slot duration	10 μs	DIFS duration	4 slots
SIFS duration	1 slot	Bandwidth	10 Mbps
CW_0	8 slots	$CW_{\max}$	512 slots
$\lambda_A = \lambda_C$	$\{100, 200, 300, 500, 800, 1000\}$ frames/sec	Simulation time	10 sec

The simulation was modeled to follow these two topologies:

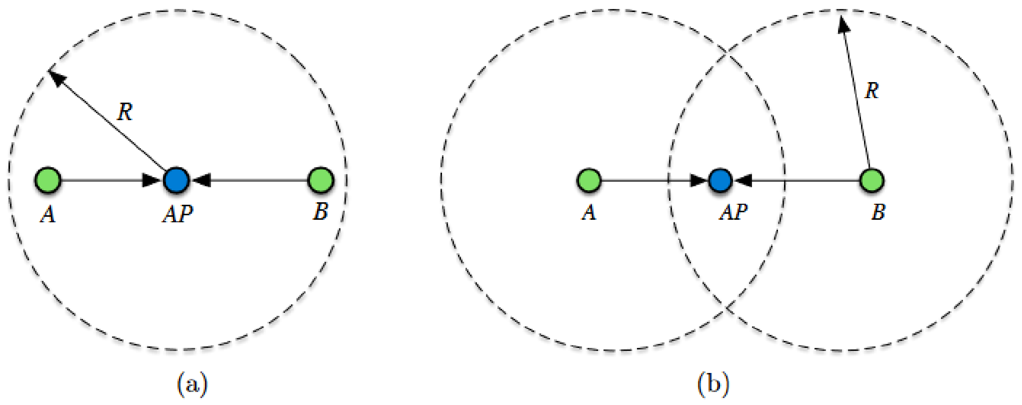


Figure 1: (a) Topology for parallel transmissions within the same collision domain, (b) topology for parallel transmissions when A and B are hidden terminals.

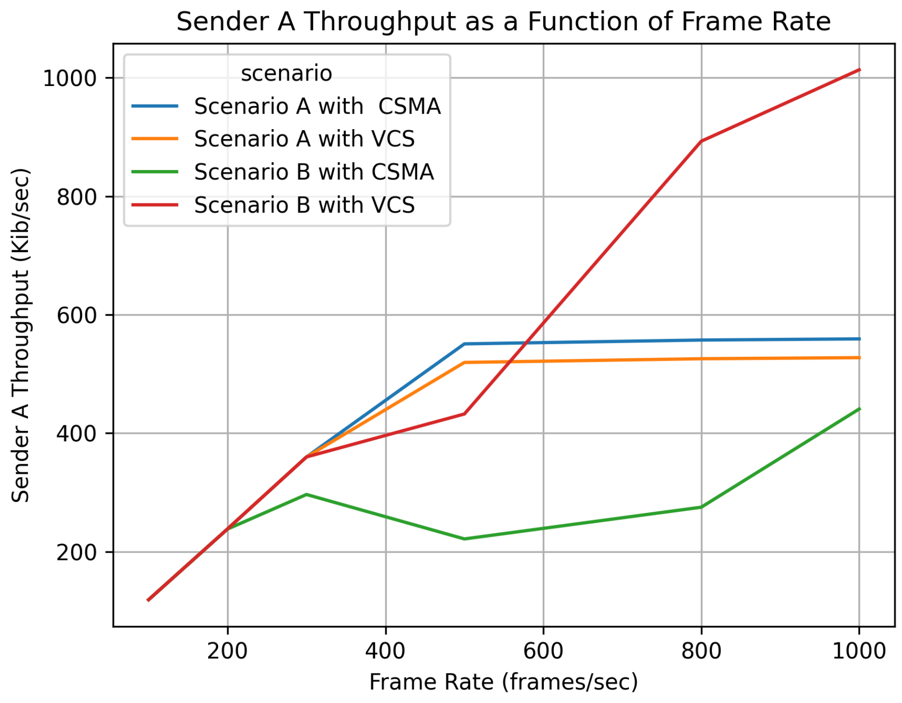
The implementation of a queuing system was used to schedule transmissions with the use of this equation:

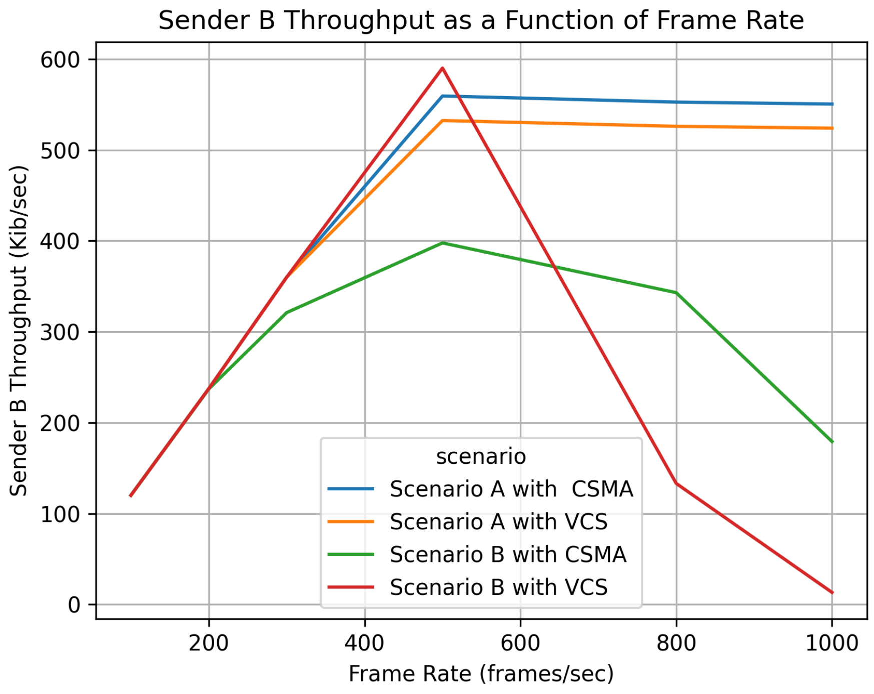
$$X = -\frac{1}{\lambda} \ln(1 - U).$$

X here represents the slot in which a transmission is added to the queue, Lambda represents the frame rate, and U represents a series of uniformly distributed numbers between 0 and 1.

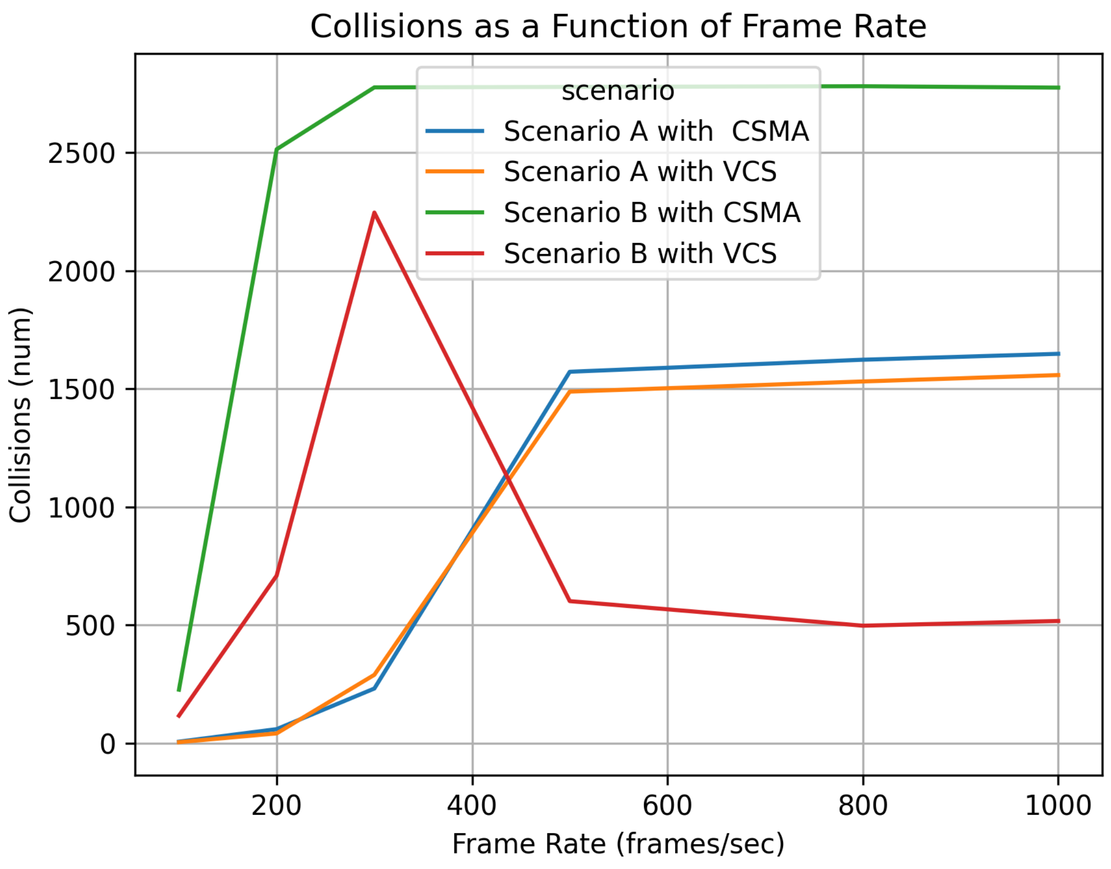
The repository can be found at https://github.com/ward1771/project_1_478.

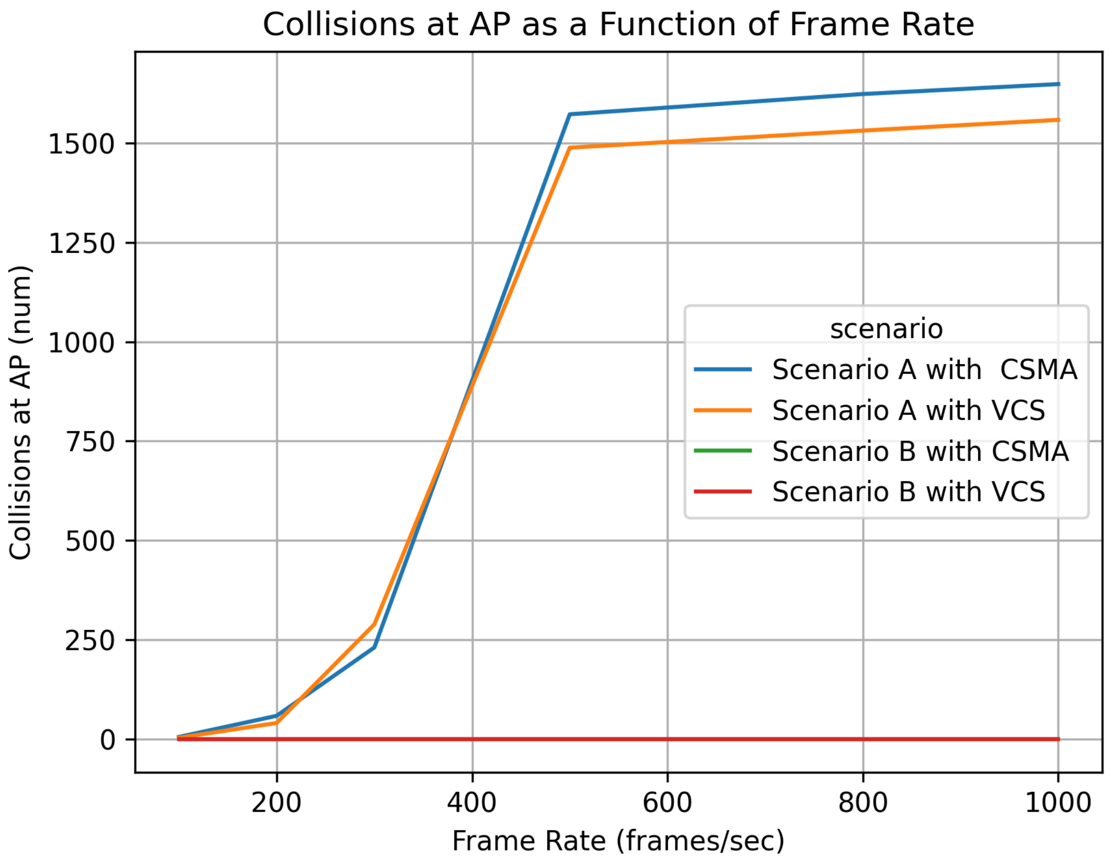
Graphs:


Sender A Throughput as a Function of Frame Rate
scenario
Scenario A with CSMA
Scenario A with VCS
Scenario B with CSMA
Scenario B with VCS
Sender A Throughput (Kib/sec)
Frame Rate (frames/sec)
200
400
600
800
1000

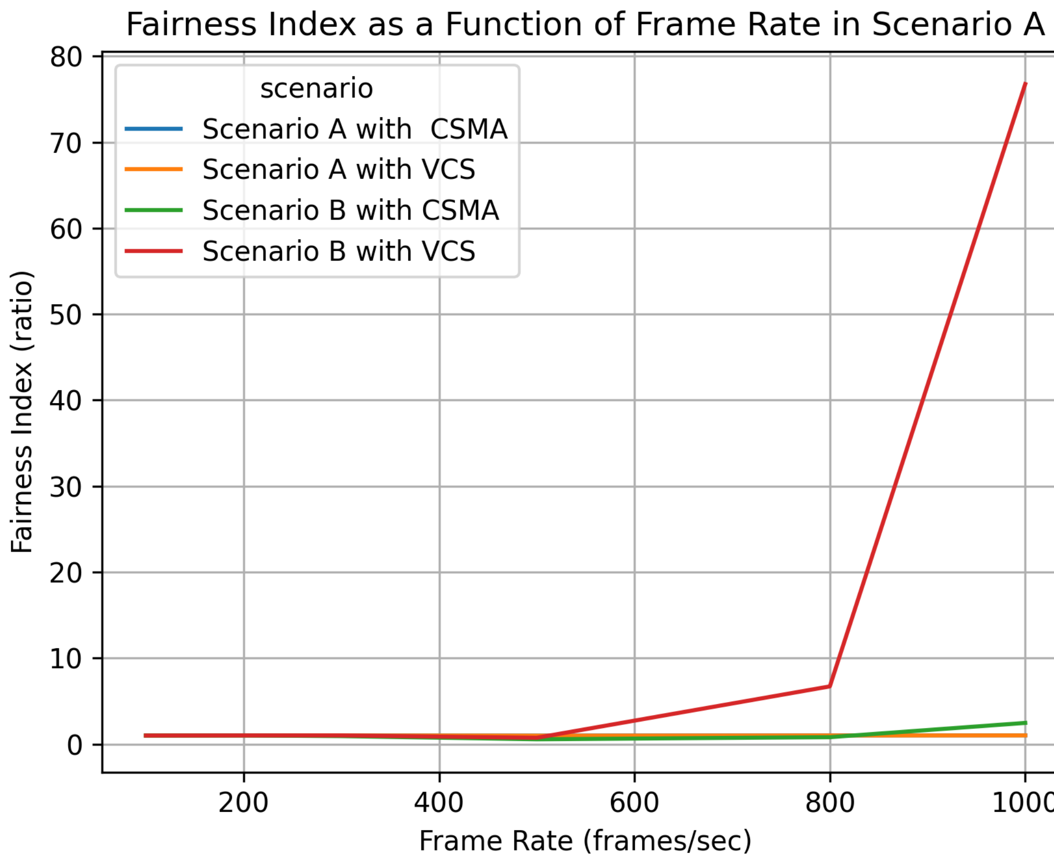

Sender B Throughput as a Function of Frame Rate
scenario
Scenario A with CSMA
Scenario A with VCS
Scenario B with CSMA
Scenario B with VCS
Sender B Throughput (Kib/sec)
Frame Rate (frames/sec)
0
100
200
300
400
500
600

In the figures above we can see that sender A generally has a higher throughput than sender B. Interestingly in both sender A and B Scenario A was more stable after the frame rate reached around 500. In graph 2 Senario B's collisions decreased immediately and decreased by a lot while Scenario B increased in graph 1. It makes sense for graph 1 for scenario B VCS to have such a high throughput because the VCS would help allow for more data transmissions to pass through uninterrupted. However in the second graph we see that scenario B throughput is very low. This might be due to an increased number of collisions at AP since we can see that graph 1 sender A is much more dominant in getting through than sender B. This could be an error in the code to why there is such a noticeable difference in throughput.





Graph 1 collisions shows that in scenario B VCS is very effective at reducing collisions. Scenario A is more prone to having collisions. This can be attributed to the collision domain of Scenario A. This is made even more clear with the second graph showing us that the number of collisions at AP are very high. The red line at the bottom is a visual mistake in the graphing of the data. It is worth noting that the collisions at around 500 again stabilize the trends to nearly the same degree. With VCS enabled it made minimal difference to the number of collisions found at AP. This is interesting because we should expect to see less collisions than what is shown.



Scenario B shows a very big difference between CSMA and VCS. Since B with VCS has a fairness index of more than 1, it shows us that A should have been the more favored. This might be due to A having more collisions and thus not getting the messages through to AP. This way A could still be sending some more but the message is not able to be received. Another possibility is that the simulation has gotten the fairness index wrong or even backwards by taking the inverse of the actual result. It is interesting to note this only happened with the VCS so there could be an underlying problem with the VCS algorithm used for the simulation.