Andrew Camps

Derek Paris

October 10, 2017

ECE 478 Project 1

Loukas Lazos

**The Distributed Coordination Function of 802.11**

As a team Derek Paris and Andrew Camps studied the performance of multiple access protocols in a wireless setting. The project studied two different topological orderings of nodes, and two different setups of Carrier Sensing Multiple Access (CSMA), all at different frame rate distributions. To complete this project as a team we decided to meet up on multiple occasions and work on creating a simulation for each test. We worked together to list out the many possible edge cases, then design four separate simulations which would be able to calculate all of the different metrics required by this study. After working out much of the needed logic, Andrew began to code the simulations for the first few scenarios. After the framework for the simulation program was developed Derek and Andrew peer programmed many of the more logically intense edge cases. Derek spend most of the time debugging while Andrew was coding the main simulations. Once all simulation code was complete and tested, Andrew produced the needed data and created the figures in the report while Derek styled and wrote descriptions for each figure.

In order to test all of the different scenarios in the project, our team decided that we would develop this simulation using C/C++. The program was developed to output data that could be used to accurately compare the difference in performance metrics for both topologies, multiple access protocols and rate sets. To do this, a standardized testing method was used. The simulation consisted of a single main.cpp file which, when compiled and ran, outputted average calculated performance metrics for a given rate set of all test scenarios. Two main data outputs were used meaning the program was run twice, getting the data for both Node A and Node C having the same rate set and Node A having a doubled rate set. Given these 1:1 or 2:1 ratio rate sets for each node, the program was run and all four simulation scenarios were conducted 500 times with different generated Poisson-distributed traffic upon each loop. In each of the 500 loops, Node A generated a Poisson-distribution and Node C generated a separate distribution given one of the four frame rates from the rate set. With these individual distributions for each node, all four simulation scenarios were ran with the same distributions. This ensured that all scenarios could be compared with consistent frame distributions for each node. Scenario functions themselves were passed the individual node data distributions where a main clock was used to drive a scenario simulation for 10 seconds. Scenarios took into account all possible cases where packets were ready to transmit and collisions could potentially happen. After all four scenarios ran, the performance metrics were stored in a running average, and the next iteration test occurred with a new set of packet distributions generated. Average data was then output at the end of the program. This data for all scenarios included: average throughputs, collisions and fairness indexes. See below for the graphical representation of each simulation.

**Figure 1:** Average Throughput of Node A (λA = λC)

Figure 1 shows average throughput of node A, while λA = λC, throughout the 500 simulation tests. What we found was that in Scenario A, all nodes in the same collision domain, the type of CSMA (VCS or non-VCS) did not substantially affect the throughput of Node A. This is shown in the solid gray and dashed yellow line in Figure 1. In Scenario B, however, for only rates greater than 100 frames/sec, CMSA with VCS far outperformed CSMA without VCS. At a transmission rate of 200 frames/sec, CSMA-VCS outperforms CSMA by 33.7%. At 300 frames/sec, that number is 45.7%. These can be seen in the solid blue and dashed orange line in Figure 1. At transmission rates at or less than 100 frames/sec, there is no significant difference in throughput for any scenario with any type of CSMA. What this suggests to us is that at lower transmission rates, efficient throughput is more easily achieved and requires less optimization.

**Figure 2:** Average Throughput of Node A (2λA = λC)

Figure 2 shows average throughput of node A, while 2λA = λC, throughout the 500 simulation tests. What we found was that in Scenario A, all nodes in the same collision domain, the type of CSMA (VCS or non-VCS) did not substantially affect the throughput of Node A, though the difference between the two did show more than in the λA = λC case. This is shown in the solid blue and dashed orange line in Figure 2. In Scenario B, however, for only rates greater than 100 frame/sec, CMSA with VCS far outperformed CSMA without VCS. At a transmission rate of 200 frame/sec, CSMA-VCS outperforms CSMA by 51.7%. At 300 frame/sec, that number is 87.1%. These can be seen in the solid gray and dashed yellow line in Figure 2. It is interesting to note that even though the transmission rate of node A was doubled, we do not see a divergence in throughput values until the after the 2nd data point, 200 frame/sec. In Figure 1, we also did not see the divergence until after the 2nd data point, however it was only 100 frame/sec. Further research may be required to discover why this is so. Also note that in throughput in Scenario B with CSMA-VCS far outperformed every other situation. This will be discussed further after Figure 4, where throughput of node C is also analyzed.

**Figure 3:** Average Throughput of Node C (λA = λC)

Figure 3 shows average throughput of node C, while λA = λC, throughout the 500 simulation tests. What we found was that in Scenario A, all nodes in the same collision domain, the type of CSMA (VCS or non-VCS) did not substantially affect the throughput of Node C. That is, until we surpassed a transmission rate of 200 frame/sec. Then there is a slight divergence between CSMA and CSMA-VCS. This is shown in the solid blue and dashed orange line in Figure 3. In Scenario B, however, for only rates greater than 100 frame/sec, CMSA with VCS far outperformed CSMA without VCS. At a transmission rate of 200 frame/sec, CSMA-VCS outperforms CSMA by 30.9%. At 300 frame/sec, that number is 41.3%. These can be seen in the solid gray and dashed yellow line in Figure 3.

**Figure 4:** Average Throughput of Node C (2λA = λC)

Figure 4 shows average throughput of node c, while 2λA = λC, throughout the 500 simulation tests. What we found was that in Scenario A, all nodes in the same collision domain, the type of CSMA (VCS or non-VCS) did not substantially affect the throughput of Node C. This is shown in the solid blue and dashed orange line in Figure 2. In Scenario B, however, for only rates greater than 100 frame/sec but also less than 250 frame/sec, CSMA with VCS outperforms CSMA without VCS. At transmission rates greater than 250 frame/sec, throughput with CSMA-VCS drops below CSMA without VCS. This can be seen in the solid gray and dashed yellow line in Figure 2. This drop in throughput in Node C is likely connected to the spike in node A’s throughput that we observed in Figure 2. What probably happened, is that at higher transmission values, node A “blocks out” node C. This occurs when there are so many collisions and each node’s backoff time grows so large, that as soon as one node transmits successfully and resets its backoff time, it can rapidly transmit packets while the other node is stuck backing off.

**Figure 5:** Average Collisions of Node A (λA = λC)

Figure 5 shows the average number of collision frames for node A, while λA = λC, over the 500 simulation tests. In Scenario A, we find that the average number of collisions does not vary more than 20%, between CSMA and CSMA-VCS. You can see this through the solid blue and dashed orange line in Figure 5. This is likely due to the fact that both nodes are always within the same collision domain, able to sense when the other is sending, which results in far less collisions than in Scenario B. Scenario B shows a vast increase in the number of collisions at node A. CSMA also consistently has a higher number of collisions, no matter what the transmission rate, than any other situation. This is likely due to the slow response time that CSMA deploys after a collision. It cannot detect the failed transmission until after fully transmitting its entire data packet. It takes time to resend the entire packet, which simply opens it up to more collision potential. CSMA-VCS shortens the time needed for error recognition but checking for a CTS in the first SIFS period. Because of this, CSMA-VCS had a lower average number of collisions. Note that at 200 Frame/sec both CSMA techniques in Scenario B have very similar numbers of collisions, differing by ~7%. The reason for this is currently unknown, but can be explored through peer consultation and further experimentation. You can see these lines as the solid gray and dashed yellow lines in Figure 5.

**Figure 6:** Average Collisions of Node A (2λA = λC)

Figure 6 shows the average number of collision frames for node A, while λA = 2λC, over the 1000 simulation tests. In Scenario A we find, once again, that both CSMA techniques have very similar numbers of collisions to each other. You can see these lines as the solid blue and dashed orange lines in Figure 5. Scenario B gets interesting though, as the doubled transmission rate seems to have very little effect on the CSMA trials, with the exception of the 100 frame/sec rate, where collisions are reduced by over 40%. The doubled transmission rate also had the effect of almost halving the number of collisions of CSMA-VCS at its peak of 200 frame/sec. The reason for this is likely due to the “blocking out” of node C that we observed in Figures 2 and 4. The less node C can send, the less chance there is of collision. You can see both of these lines as the solid gray and dashed yellow lines in Figure 5.

**Figure 7:** Average Collisions of Node C (λA = λC)

Figure 7 shows the average number of collision frames for node A, while λA = λC, over the 1000 simulation tests. What we can immediately observe is that this figure is identical to Figure 5. What this, and common sense, tells us is that anytime there is a collision involving node A, it also colliding with and involving node C. One node cannot have a collision without the other node involved. Therefore each node’s count of collisions is identical.

**Figure 8:** Average Collisions of Node C (2λA = λC)

Figure 8 shows the average number of collision frames for node A, while 2λA = λC, over the 500 simulation tests. We can immediately notice that this figure is identical to Figure 6. The reasoning as to why that is can be found in the description of Figure 7.

**Figure 9:** Fairness Index (λA = λC)

**Figure 10:** Fairness Index (2λA = λC)