*IEEE 802.11 Saturation Throughput Project*

Charlie Andre

Ming Hsieh Department of Electrical Engineering

University of Southern California

candre@usc.edu

Samuel Bruner

Ming Hsieh Department of Electrical Engineering

University of Southern California

sbruner@usc.edu

# NS3 Implementation

The following describes our method for writing the final code:

1. **Start with third.cc** (from the tutorials)
   1. Code: <https://www.nsnam.org/doxygen/third_8cc_source.html>
   2. Clean out everything related to point-to-point and CDMA, leaving only code related to WiFi
2. **Star layout.** The cdma-star example didn’t make a lot of sense, and we couldn’t figure out how to make a star topology based only on wifi. For this reason, we hard-coded the star topology using a for loop along with sine & cosine. Prior to allocating the spokes, we allocated the position of the hub at the origin.
   1. There were several examples to show how to allocate positions from (x,y,z) values. Here is one of them: <https://www.nsnam.org/doxygen/wifi-tcp_8cc_source.html>
3. **Internet Stack.** Implement and install the internet stack on all nodes.
   1. Source: <https://www.nsnam.org/doxygen/wifi-spectrum-saturation-example_8cc_source.html>
4. **Assign IP addresses.** To make finding the IP address of the hub easy, we made sure to assign the hub’s IP first. Then we allocated IP addresses for the spokes.
5. **Create UDP Application at the hub.** Because we strategically assigned the IP address of the hub before that of the spokes, we knew that the hub would have an IP address of <subnet>.1
   1. Source: <https://www.nsnam.org/doxygen/wifi-hidden-terminal_8cc_source.html>
6. **Make the Hub Application into a Sink.** This allowed us to calculate stats, such as the total number of received packets at the hub.
   1. Source: <https://www.nsnam.org/doxygen/wifi-tcp_8cc_source.html>
7. **Create UDP Applications at the Spokes.** We did this in a loop to get all of the spokes to create a UDP application to send to the hub.
   1. Source: <https://www.nsnam.org/doxygen/wifi-spectrum-saturation-example_8cc_source.html>
8. **Start Simulation.** Self-explanatory, shown in tutorials and every other piece of ns3 code.
9. **Calculate Throughput using statistics at the hub.** This was made easy because we had the hub as a sink.
   1. Source: <https://www.nsnam.org/doxygen/wifi-tcp_8cc_source.html>
10. **Change minCw and maxCw MAC parameters using Txop**. This was particularly difficult.
    1. Source: <https://www.nsnam.org/doxygen/wifi-80211e-txop_8cc_source.html>
11. **Change some other MAC parameters using WifiMacHelper::SetType**. Here we modified parameters such as Slot, DIFS, SIFS, and ACK Timeout to match the parameters chosen in Bianchi’s paper.

# Results

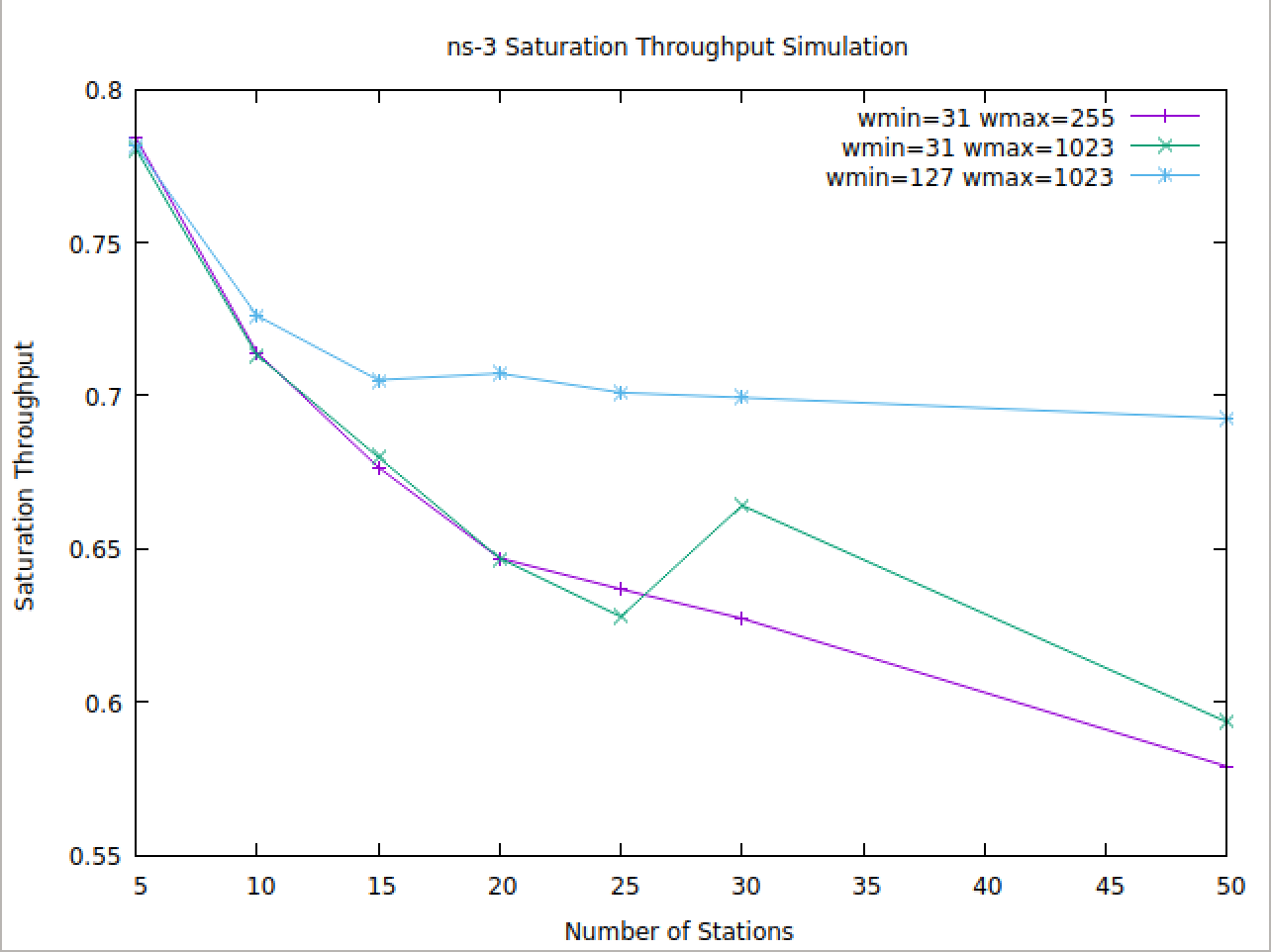


Figure 1: ns-3 Saturation Throughput

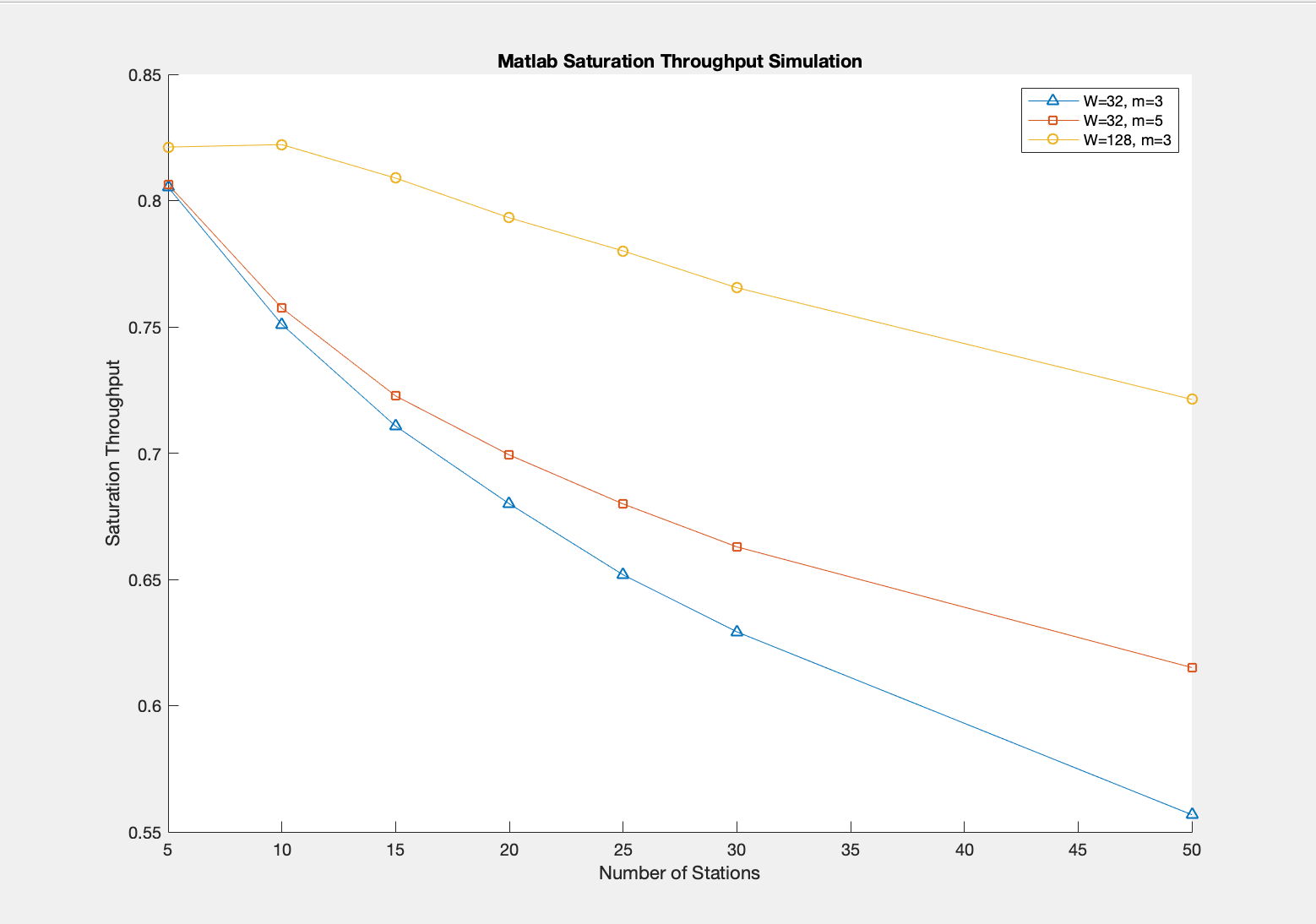


Figure 2: Matlab Saturation Throughput

### Simulation Parameters

|  |  |
| --- | --- |
| Packet Size | 2048 |
| Channel Bit Rate | 6 Mbps |
| Slot Time | 50 us |
| SIFS | 28 us |
| DIFS | 128 us |
| ACK\_TIMEOUT | 300 us |

# Comparison with Bianchi’s Plots

To the best of our ability, we have overlaid the plots to show that they are of the same form.

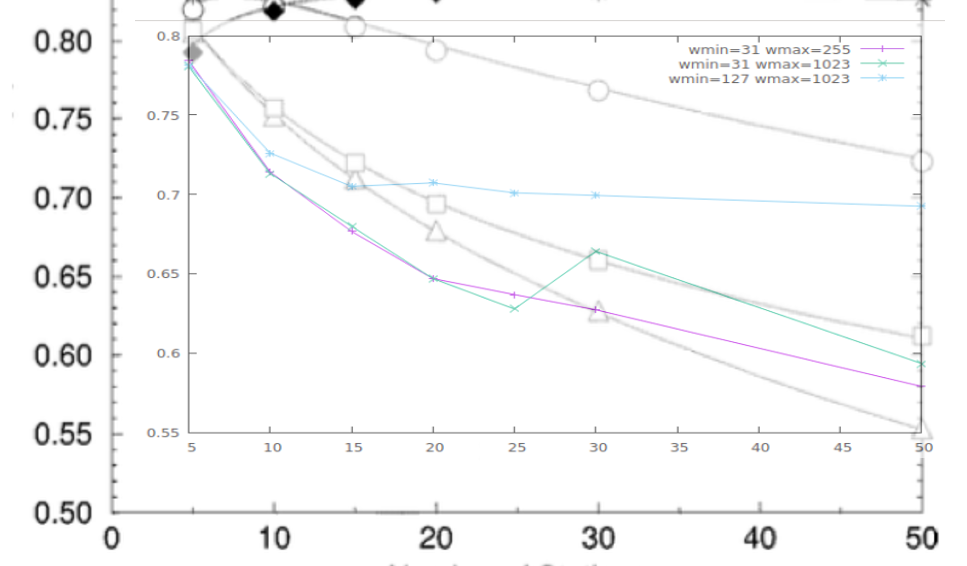


Figure 3: ns-3 results overlayed with Bianchi Figure 4

The two plots have some very important similarities:

* The Bianchi plots and our plots for the same W, m values start in approximately the same place
* The Bianchi plots and our plots for the same W, m values end in approximately the same place
* The slopes of the lines for the same W, m values are relatively the same
* The ordering of the lines remains relatively consistent
* A larger allowable contention window leads to higher throughput for larger networks

The values however, are not exactly the same. Here are some reasons for which we might have gotten slightly different values than Bianchi:

* Different packet payload size, limited in our case by our PHY simulator settings. This results in slightly lower throughput for our simulations due to smaller packet size
* Different Phy simulators. This might explain why our throughputs are consistently lower than Bianchi’s

# Comparing Simulation and Analytical Results

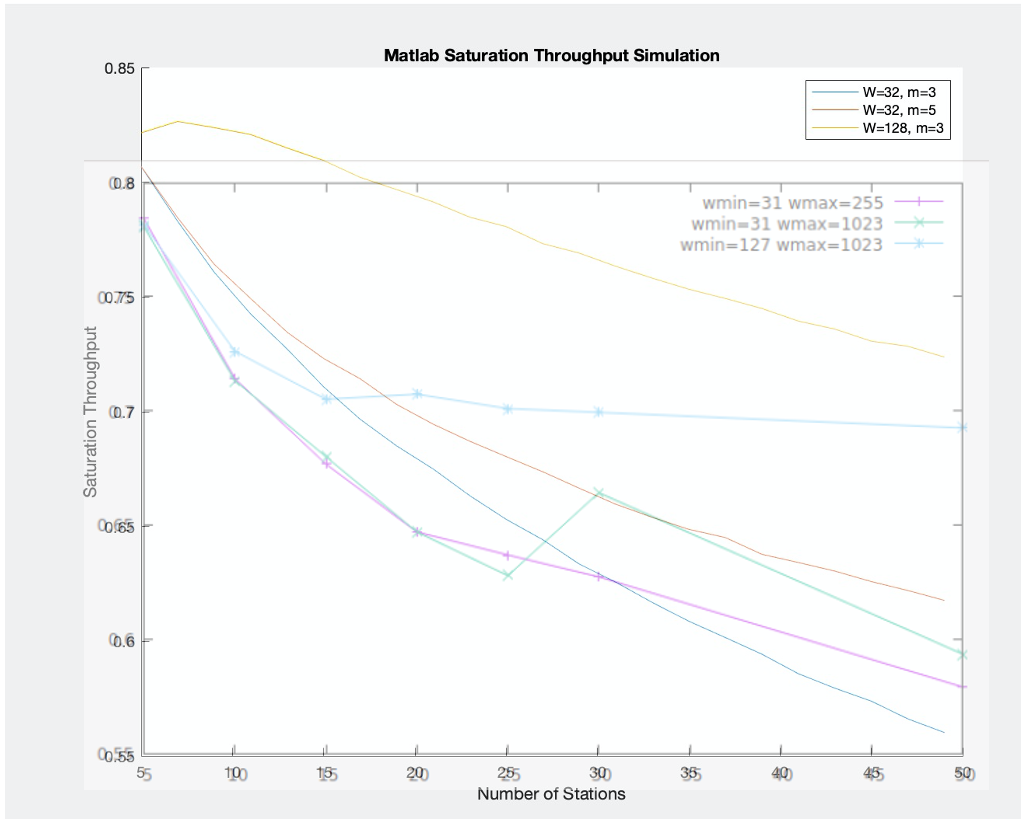


Figure 4: Matlab results overlayed with Bianchi Figure 4

We were very happy with the results of these overlaid plots. Here are some noticeable similarities:

* The MATLAB and NS3 plots for the same W, m values start in approximately the same place
* The MATLAB and NS3 plot for the same W, m values end in approximately the same place
* The slopes of the lines for the same W, m values are relatively the same
* The ordering of the lines remains relatively consistent between NS3 and MATLAB
* A larger allowable contention window leads to higher throughput for larger networks

However, the MATLAB output and the NS3 simulations did not result in exactly the same plots due to these differences:

* The MATLAB plot took into account sizes of various fields in the headers, the NS3 plots used default values
* The MATLAB plot used the same packet size as Bianchi’s simulations. The NS3 code used a packet size that would maximize normalized saturation throughput
* The MATLAB simulation was able to plot for each number of stations [1:50] whereas the NS3 simulation was heavier and only plotted for a few predefined values {5, 10, 15, 20, 25, 30, 50}

# NS3 and MATLAB Code

Please visit the project Github: <https://github.com/candre97/Saturation-Throughput-of-CSMA-CA> for the NS3 and MATLAB code.

# References

[1] G. Bianchi. IEEE 802.11-saturation throughput analysis. IEEE Communications Letters, 2(12):318–320, Dec 1998. URL: https://pdfs.semanticscholar.org/4a5c/ f874e9469815113c7ea93ff97317bdb52a90.pdf, doi:10.1109/4234.736171.