I made a function that did the inversion and simplification with OR that Bryan described. I also made a function that slightly modified the format that the old json file from the CircuitOptimizer program. The next thing to do would be to compare the cost of each of the gates and see where mine improve on the netlist Bryan sent and where, if anywhere, it is worse.

Then I will convert the json file to a dictionary that maps each truth value to a list of netlists. There should be one netlist for each circuitString.

There were 9 differences in the NOR file. These are the results from the NOR comparison:

00010110: [9, 10]

00101001: [8, 9]

01001001: [8, 9]

01100001: [8, 9]

01101011: [8, 9]

01101101: [8, 9]

01111001: [8, 9]

11100101: [7, 8]

11101001: [10, 11]

While comparing the netlist to the circuitString, I noticed that an OR should actually not cost anything because it is just two promoters in front of a gene so I made the adjustment. Surprisingly the differences were not all on the same truthValues as for the NOR

There were 9 differences in the OR file. These are the results from the OR comparison:

00010110: [9, 10]

00011010: [6, 7]

10000110: [7, 8]

10010010: [7, 8]

10010100: [7, 8]

10011110: [7, 8]

10110110: [7, 8]

11010110: [7, 8]

11101001: [8, 9]

The next step would be converting the circuits in my file to a list of netlists and saving them as a json file.

I completed this process.