Bryan traveled this week, so I may be working with Jing and helping her out with whatever she needs help with. She isn’t here yet, though, so in the mean time I will work on seeing if there is a way to simplify circuits to speed up the programs’ runtimes using suggestions proposed by Bryan last Friday. I will exam the 12 gate circuits from the speedy algorithm and compare it to their actual minimums of 8 and 9 gates and see if there is a way to go from the 12 gates to the 8 gates or 9 gates (or anything less than 12) by swapping out portions of the 12 gate circuit with other smaller pieces.

Before I do this, I need to get my wifi working again. Unfortunately, I had to restart my computer which closed the running python programs.

I discovered cProfile which tells you which methods in your program are using the most time. Apparently the findLeftParentheses is taking the most time. It accounts for approximately 9/15 seconds of runtime. It is being use to find the gate count. Using another method for gate counting, I was able to cut the time approximately in half for going up to 5 gates. I am hoping that this will mean a reduced time for higher gates also.

The output for cProfile for 6 gates using the new gateCounter:

ncalls tottime percall cumtime percall filename:lineno(function)

18601427 93.954 0.000 97.191 0.000 GateMinimization2.py:44(findSmallestParentheses)

1658931 61.672 0.000 178.842 0.000 GateMinimization2.py:31(gateCounter2)

17244486 24.607 0.000 24.607 0.000 {method 'replace' of 'str' objects}

1 4.131 4.131 192.372 192.372 GateMinimization2.py:552(minCircuitFinder4)

It is still not efficient enough. I just noticed that the time per call is really small (0.000) so the problem is with the number of calls (18601427 for findSmallestParentheses and 1658931 for gateCounter2). If I can reduce this then the problem will be solved. This boils back down to narrowing the list we search through or creating some sort of greedy algorithm. Also I should probably make a function that takes a truthValue as input and stops when you find the truthValue. This is easier to do so I will work on this first.

I made a function (minCircuitFinder5) that takes a truth value as an input and a number of gates with default being 5. It will check up to 5 gates and look for the desired truth value. If it cannot find it, it will return the dictionary with the list of truthValues and the dictionary with all the circuits. The next step is to see if there is a way to break down the truth value into two components that can nor together to give it. Then see if both of those components are in the found truth values. If one isn’t then it will break it down further into another pair of truth values. This portion I will do in a new python file and if I get it functional I will transfer it to the main file. The idea of this method is to work forward for a bit, and it we haven’t found it after however long, then we try to work backward until we find it. The working backwards way would use recursion and could potentially be, by itself a way of solving for a circuit. I will have to program it to find out.

I am getting stuck very badly. The recursion is very difficult to use. I may have to start from scratch tomorrow because it doesn’t seem to be working and I can’t find the problem. I may start with 2 inputs.

Spoke to Jing and she described what she is working on and sent me some papers that I will read to find out what she is working on.