Kevin Lee
CSE 13S Winter 2022

Assignment 1 - WRITEUP.pdf

Introduction:

This writeup will primarily be focused on the three graphs produced, along with the UNIX commands that helped make it all possible. The three graphs I produced I named length.pdf, max.pdf, and his.pdf.

A lot of the UNIX commands I used were shown to me in Eugene's lab section, along with some commands coming from the lab document itself and the lecture on scripting.

length.pdf

I produced the length.pdf graph using a for loop and UNIX commands shown in Eugene's lab section.

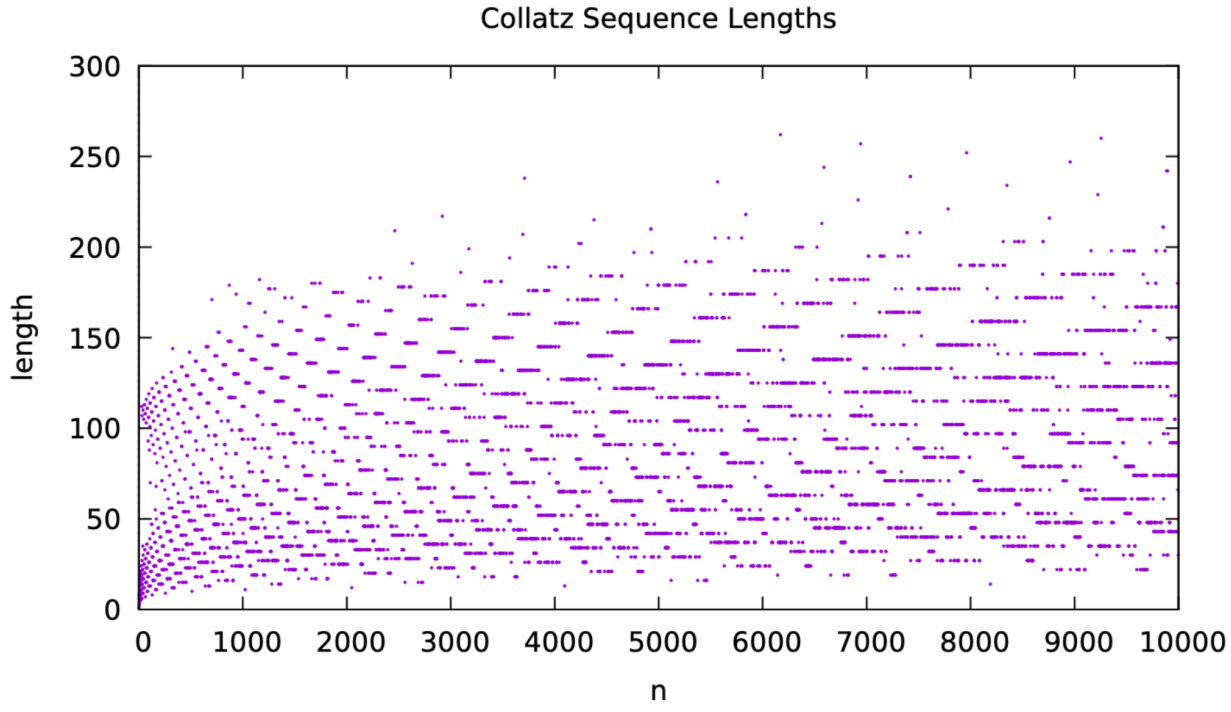
```
for n in {1..10000}; do
        ./collatz -n $n | wc -l >> /tmp/length.dat
done
```

The for loop can be seen in the first line, and we went from 1 to 10,000, to simulate the collatz sequence starting from points 1 to 10,000 inclusive. The second line is where we appended to our temporary file: /tmp/length.dat. The collatz code specifies that we can run the collatz sequence at a particular starting point by using ./collatz -n, followed by the starting point. In order to run every collatz sequence from 1 to 10,000, we used the aforementioned for loop and used the command ./collatz -n $n. We then piped this output into wc -l to specifically count for the number of lines, effectively giving us the length of the collatz sequence at the given starting point. We appended this output to our temporary file of /tmp/length.dat using the >> command.

Finally, we plotted on gnuplot.

```
set terminal pdf
set output "length.pdf"
set title "Collatz Sequence Lengths"
set xlabel "n"
set ylabel "length"
set zeroaxis
plot "/tmp/length.dat" with dots title ""
```

The first line initialized a new pdf. The second line specified the name of our plot to be set as length.pdf. Then we set the title of our plot to be Collatz Sequence Lengths per the asgn1 documentation. We set the xlabel and ylabel as n and length, once again per asgn1 documentation. We finally plotted the points in the last line of the code, plotting the data from our /tmp/length.dat file and with dots.



max.pdf

I produced the max.pdf graph, once again using a for loop, UNIX commands, and heavy inspiration from Eugene's lab section.

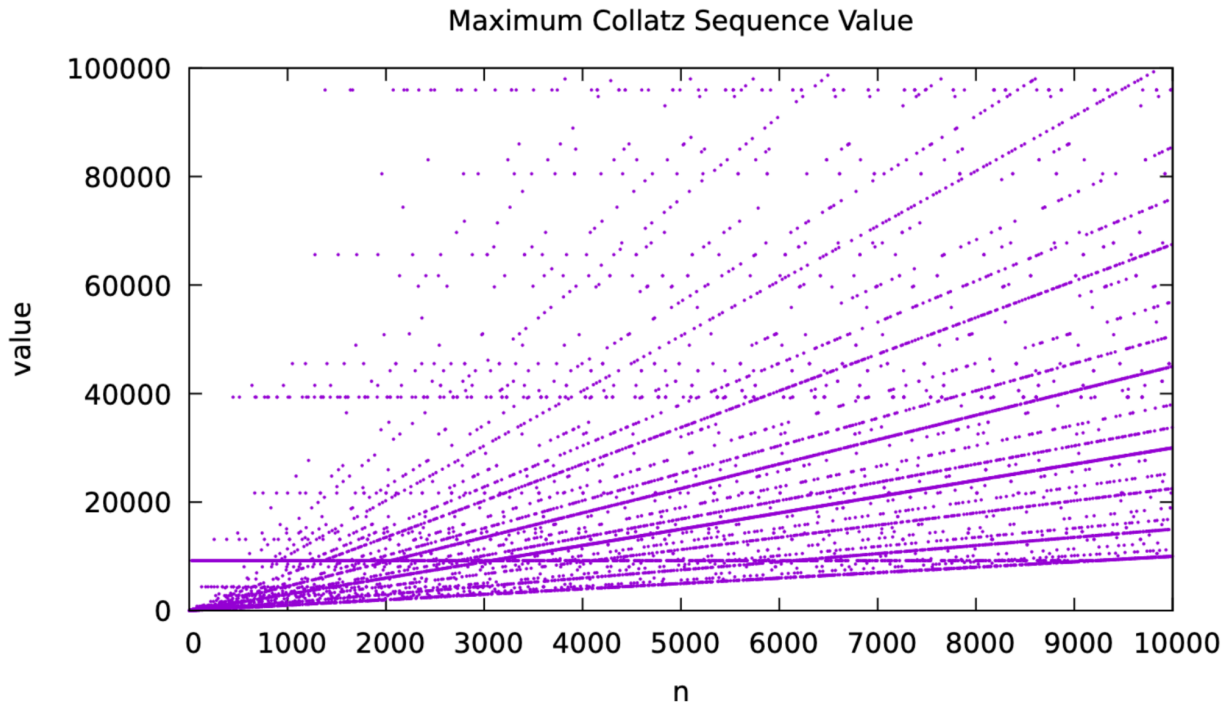
```
for n in {1..10000}; do
        ./collatz -n $n | sort -n | tail -n 1 >> /tmp/max.dat
done
```

I used the same for loop and ./collatz code to iterate through the exact same set of collatz sequences as last time. However, instead of looking for the length of the collatz sequence, we were looking for the maximum value the collatz sequence would reach. Once again we piped the output of ./collatz, however this time into sort -n, the -n specifying to sort by numerical value. We once again piped the output of the numerical sorting into tail -n 1, which specified that we would get the last value with -n 1 specifying that we only wanted the last value in the list. So in summary, we sorted the steps in a collatz sequence, giving us a list from smallest to largest, and then outputted the last value of this list, giving us the maximum value. This value, or the maximum value would then be appended to our temporary file /tmp/max.dat in much the same way as /tmp/length.dat.

Finally, we plotted on gnuplot

```
set terminal pdf
set output "max.pdf"
set title "Maximum Collatz Sequence Value"
set xlabel "n"
set ylabel "value"
set yrange [0:100000]
set zeroaxis
plot "/tmp/max.dat" with dots title ""
```

We initialized the graph in much the same way, only changing minor titles and axis names per asgn1 documentation. The only difference is that we set the range of the y axis from 0 to 100,000, to prevent our graphing being out of proportion because of a few outliers.



his.pdf

I produced this graph using UNIX commands shown in Eugene's lab section.

```
head -n 10000 < /tmp/length.dat | sort | uniq -c | less > /tmp/his.dat
```

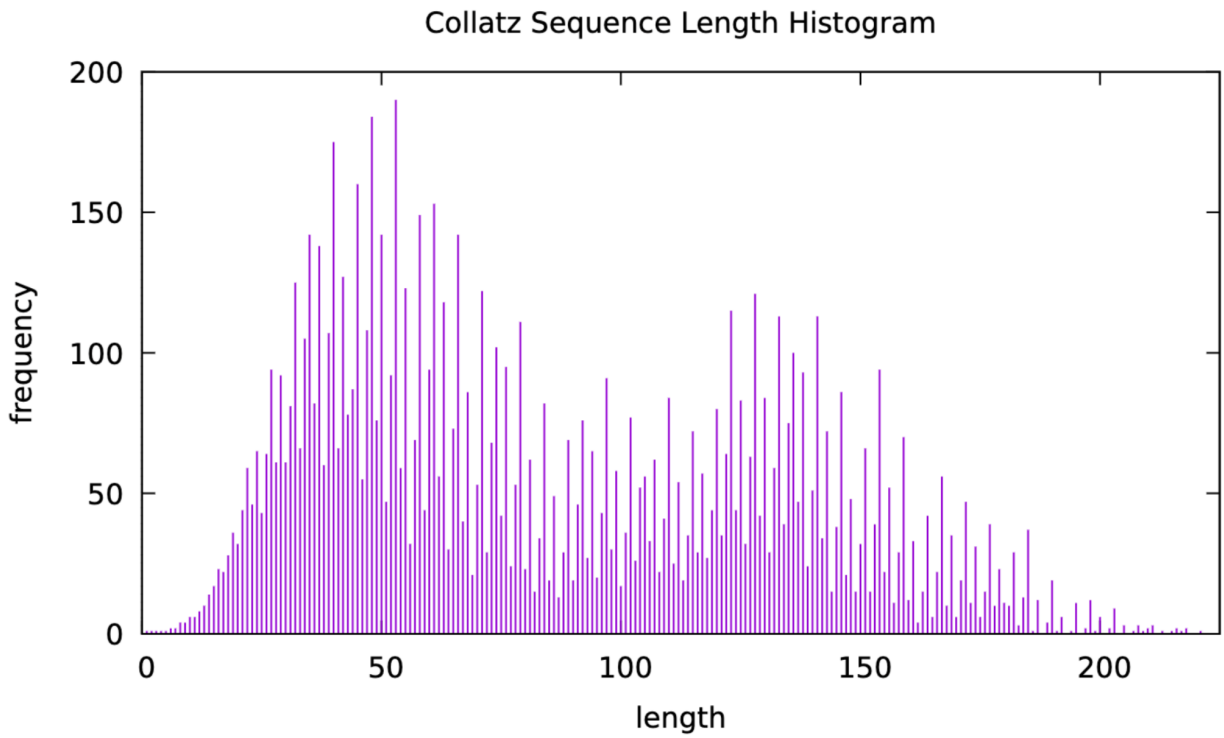
We can see that this code does not utilize a for loop, but does in effect, as we are using data from /tmp/length.dat, which was created using a for loop. The main function of this command is to find the frequency of a given length and redirect that output into /tmp/his.dat. The head -n 10000 command is to redirect whatever is inside <> to /tmp/his.dat. Head normally only redirects the first ten items of a list, however -n 10000 specifies that it should redirect up to 10000 items. Admittedly, this is a bit overkill, and could have done with only around 250, as there are only around 250 unique collatz lengths, however, -n 10000 makes sure that no data is missed, even if the collatz sequences all output a different length. Finding the frequency given a length is what is inside of the <>. First we pipe the output of /tmp/length.dat, or our length file to sort, to sort the lengths numerically. We then piped this to uniq -c, the -c specifying to prefix the lines or lengths by number of occurrences. Finally we piped into less, and redirected this file into /tmp/his.dat. The reason why we sorted the length values first, was that I noticed that if we did not do that, it would appear that the uniq -c program would miss

over some lengths, and would show there being say 2 occurrences of length 133, but saying that there were another 4 occurrences of length 133 later in the /tmp/his.dat file. I think what would happen is that the uniq -c command would only look for identical lengths for a little bit and then forget about it, leading to diverged lengths that were still the same.

Finally, we plotted on gnuplot.

```
set terminal pdf
set output "his.pdf"
set title "Collatz Sequence Length Histogram"
set xlabel "length"
set ylabel "frequency"
set yrange [0:200]
set xrange [0:225]
set zeroaxis
plot "/tmp/his.dat" using 2:1 with impulses title ""
```

The first few steps mirror that of the first two graphs. However, the differences start with setting the xrange. We set the yrange in max.pdf to prevent our graph from being out of proportion due to outliers and did so with the xrange as well. These were the bounds in the asgn1 documentation. Finally, instead of plotting the data as points using dots, we used impulses this time, which are boxes, but with 0 width, as per the example in the asgn1 doc. Finally, we switched the x and y columns, since we wanted to plot frequencies given the length, while our /tmp/his.dat had length given the frequency.



Conclusion

- This assignment introduced me to scripting and creating a shell script
- Used UNIX commands extensively for the first time
- Utilized gnuplot for the first time
- Implemented pipes and other concepts we learned in the scripting lecture