Gabriel Borges

Professor Wu

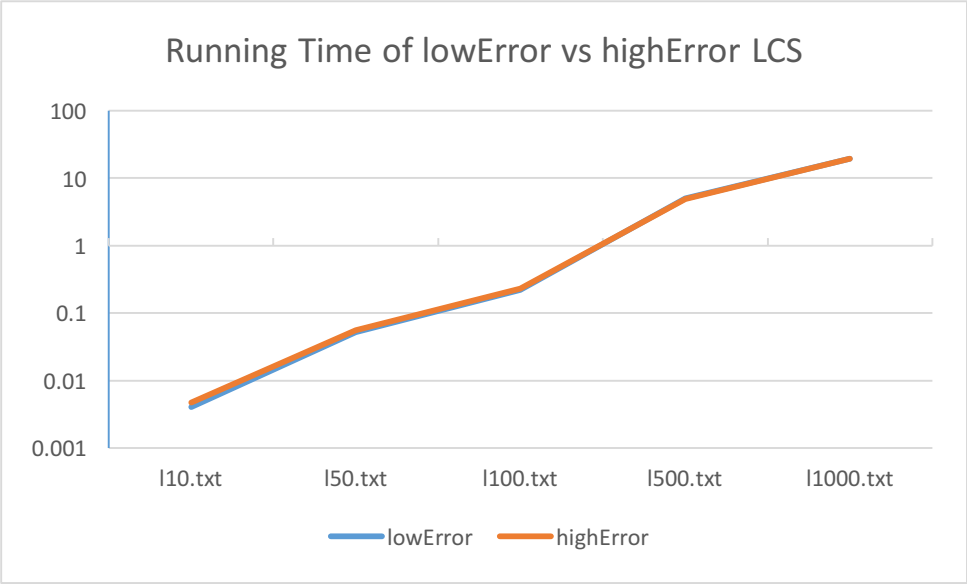
CSE 3500 – Programming Assignment 2

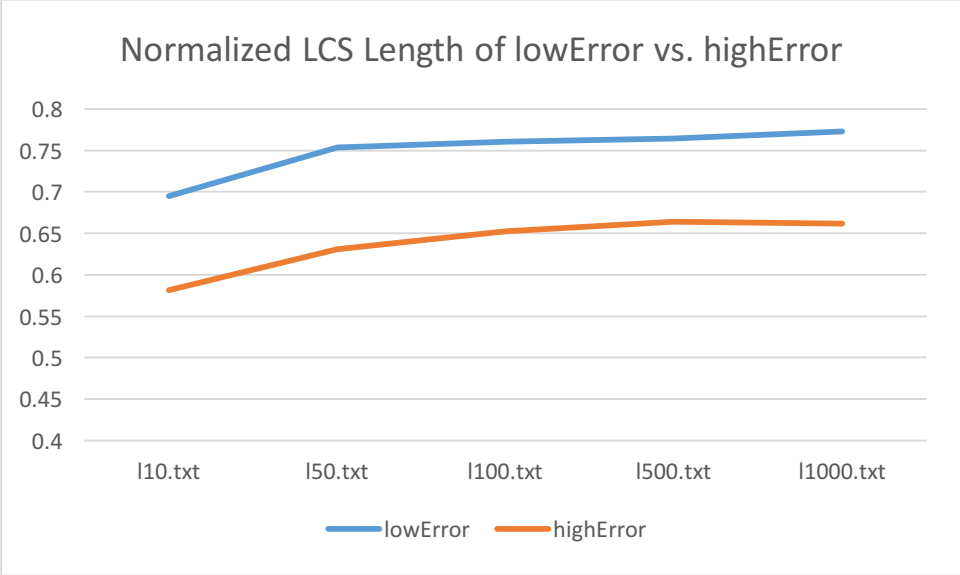
4/13/16

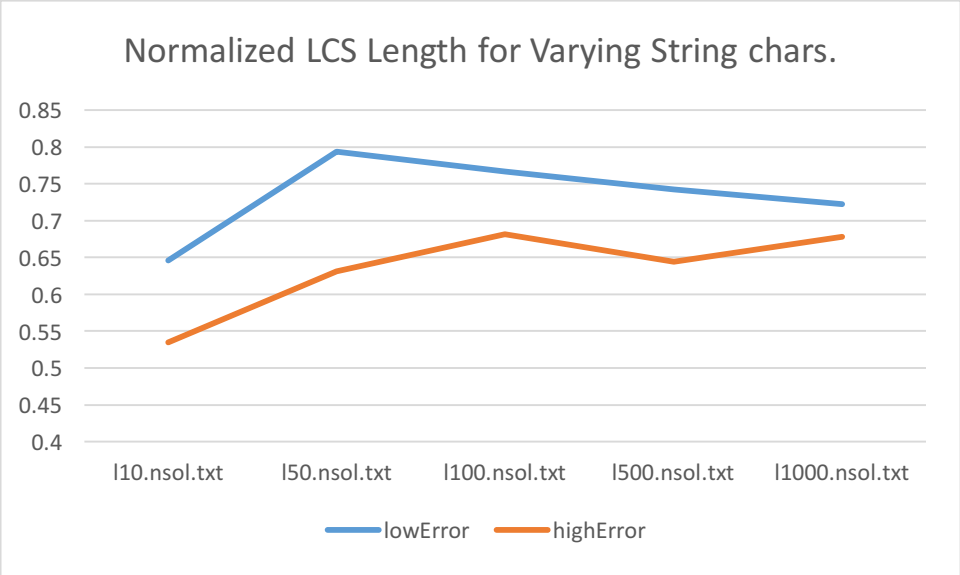
Programming Assignment 02

Settings: My computer is running on a 2.4GhZ-i5 processor with 8GB of ram. My chosen language for this project is python.

Results: Raw data has been included within an excel spreadsheet and submitted with this assignment. Therefore, no chart is provided within this document, only the charts for clarity.







Some notes about the results (in answer to the questions). Firstly, the interesting thing is that running-time is relatively unaffected for files where the error rate is high or low: The algorithm runs essentially the same. The only impact is the amount of data to go through, which has a strong affect on the runtime. Also, the normalized length of the LCS varies between those files with low and high error quite significantly. On average, the normalized length of the LCS for the files with low error was between 0.75 and 0.8, whereas the average normalized length of the LCS for the files with high error was between .6

and .67. As for the files where the consensus was varied, it is clear (more so on the raw data versus the above graph) to tell when the two strings being compared were of a different consensus. The LCS for those strings has a normalized variance of around 20%. By comparing the raw data, it is possible to discern which string actually came from a different consensus: you simply take whichever string is having an affect that causes the variance. I.E. if string 1 compared with string2 have an average normalized length of around .75, but when they are both compared with string3 they have a normalized length of .5, it can be deducted that string3 is from a different consensus.

Conclusion:

One of the things I've learned, is the implementation of a dynamic programming algorithm is actually quite simple once you have the main idea of how the algorithm works down. The computer can then solve seemingly complex problems quite efficiently. Focusing on the particulars of this assignment, one of my major takeaways is that the computation of an LCS is inherently dependent mostly on its length: The calculation for an LCS in general does not depend on the actual makeup of the string at all when it comes to dynamic programming.

Source Code:

Not copy/pasting my source code here, but it will be attached as part of the upload. An example output is:

```
The LCS of strings 4 and 5 has a length of 5 and the string: ATCTG

The LCS of strings 4 and 6 has a length of 5 and the string: ATCCT

The LCS of strings 4 and 7 has a length of 6 and the string: ACTCCT

The LCS of strings 4 and 8 has a length of 4 and the string: CTCC

The LCS of strings 4 and 9 has a length of 6 and the string: AATCCT

The LCS of strings 4 and 10 has a length of 5 and the string: ATCCT

The LCS of strings 5 and 6 has a length of 6 and the string: TATCTC

The LCS of strings 5 and 7 has a length of 6 and the string: TATCTC

The LCS of strings 5 and 8 has a length of 5 and the string: TTCTC

The LCS of strings 5 and 9 has a length of 5 and the string: ATCTC

The LCS of strings 5 and 10 has a length of 5 and the string: TATCC

The LCS of strings 6 and 7 has a length of 7 and the string: TACCTTC

The LCS of strings 6 and 8 has a length of 6 and the string: TTCTCC

The LCS of strings 6 and 9 has a length of 7 and the string: ATCCTTC

The LCS of strings 6 and 10 has a length of 7 and the string: TATCCTT

The LCS of strings 7 and 8 has a length of 6 and the string: TTTCCC

The LCS of strings 7 and 9 has a length of 7 and the string: ACCTTCT

The LCS of strings 7 and 10 has a length of 6 and the string: TATCCC

The LCS of strings 8 and 9 has a length of 5 and the string: TGCCC

The LCS of strings 8 and 10 has a length of 5 and the string: TACCC

The LCS of strings 9 and 10 has a length of 6 and the string: ATCCCT

Runtime: 0.00639986991882 seconds
gborges0727 (master *) CSE3500-ProgrammingAssignment02 $
```