Dev Ganesh

REPORT FOR HW2

Method 1: I wrote code in **R** to instantitate **22 SOCK** cores. Each of these cores put together frequency tables for its set of files. After this, the code put together all the frequency tables into one table.

I ran my code on Hilbert and it took the 22 processors **15.2 minutes** to process all 81 CSV files. Before this, I ran my code locally on my laptop with just 2 processors and it took ~ **4 hours** to process the files.

Once all the files were processed, the time required to put the frequency tables together into one table, and the time needed to compute the mean, median and standard deviation was of the order of a **few seconds.**

With my parallel code, I get the **mean** to be **6.98276,** the **standard deviation** to be **30.22053** and the **median** to be **0**.

Method 2: I wrote a mapper and reducer in Python. I ran the code locally to check if it Is working and it processed the CSV files fine. When I ran it on the Hadoop machine, I kept getting a map failure error. The cause for this requires further investigation.

When comparing both these methods, both seem equally good from a computation standpoint. When I ran Duncan's Java code on the Hilbert machine, it took around 5 minutes to process all 81 CSV files. Based on this, my guess is that my own mapper and reducer would have taken around 15 minutes to process the files (since there is no combiner here). Thus, both methods seem equally efficient.