+1 ISSUE #21

+1 ISSUE #25

+1 ISSUE #29

CS 1632 – Deliverable 3: Performance Testing

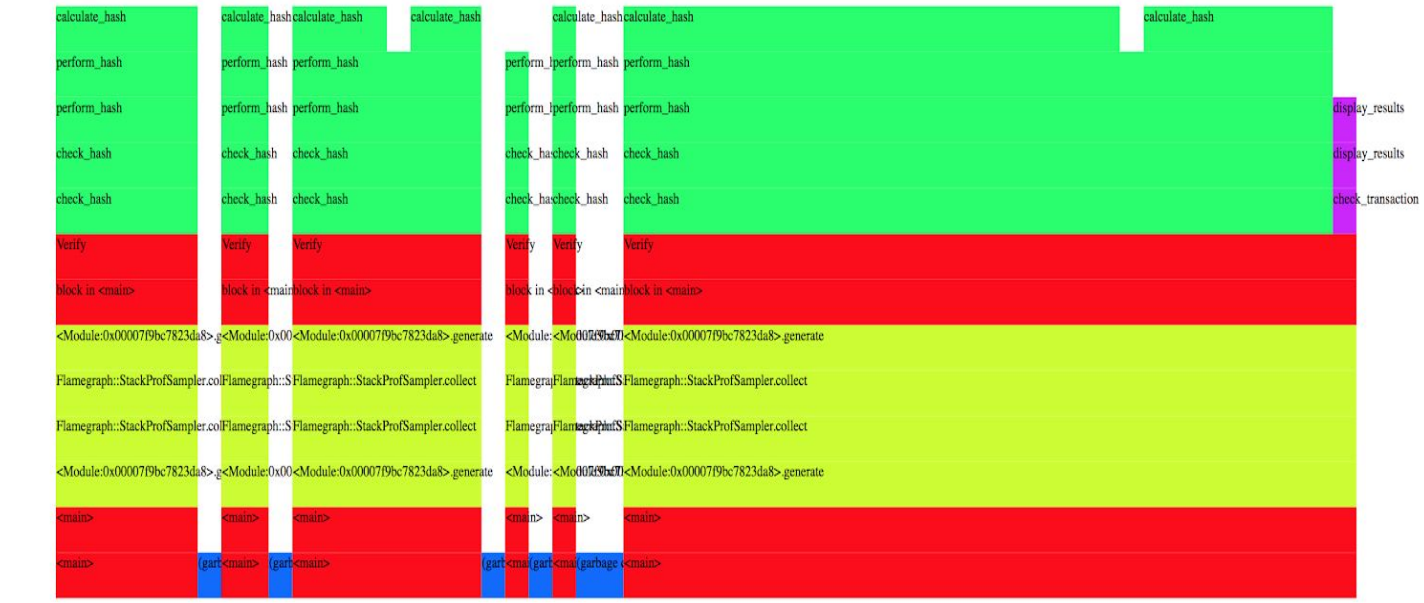
Lucas Brennan - lbrennan26 (github)

Michael Okonski - mh08 (github)

Github Repo - https://github.com/lbrennan26/CS1632_D3

Summary

The part of the deliverable that was most challenging was the hashing portion. We had a lot of trouble in terms of comparing hashed values especially in hex. It was saying every hash was not equal even when they were. We realized that it was easier for us to do equality comparisons not in hex because that was giving us expected results. We considered various edge and failure modes. Most of which involve user input on the command line. We do file checking to make sure that the file exists. Furthermore, there are other cases where the blocks are out of order or the blocks don't start with xero. In terms of transaction checking there is the case that a person’s billcoin balance is negative at the end of block. There are cases where illegal characters are used in the blockchain is invalid. When we used the flame graph tool we saw that the perform and calculate hash functions were taking up the most CPU time. We then saw this and made the following changes to speed the program up. We isolated the check previous hash and the check current hash function into separate functions. Then split the data into 8 separate arrays. Through Jruby and threading we were able to check the hash on all 8 arrays simultaneously. This allowed for substantial speedup almost at 300%. We were unable to do this with the check previous function due to the fact that we would have to share data between threads which was not something we were up for.



verifier_sim.rb (49 samples - 87.50%)

flamegraph-0.9.5 (49 samples - 87.50%)

hash_checker.rb (48 samples - 85.71%)

(6 samples - 10.71%)

transaction_checker.rb (1 sample - 1.79%)

Long. txt times: (eight threads)

Run 1: 10.9516 sec

Run 2: 10.9997 sec

Run 3: 10.9677 sec

Mean: 10.9730

Median: 10.9677