Question: Is there any difference in performance when running the same experiment on the multithreaded server and on the single threaded server? What is likely the bottleneck in your system? How much concurrency is available in various parts, such as dispatch, worker, logging? Can you increase concurrency in any of these areas and, if so, how?

Answer: There is a slight difference in performance when running the same experiment on the multithreaded server and on the single threaded server. The multithreaded server had a time of about 120ms while the single threaded server had a time of about 150ms.

The likely bottleneck in my system is the fact that I am locking a mutex whenever I am inputting a connection to a buffer, which prevents other threads from getting previous connections from the buffer while new connections are being inputted. Additionally, logging is another bottleneck as the threads will lock the mutex when doing a pwrite() and incrementing the global offset, which will make the other threads having to wait.

There exists concurrency between threads executing its reply methods, however, there is a lack of concurrency between threads accepting connections and logging. Threads will generally lock other threads out when taking a connection out of the buffer, and when doing a pwrite() on a global offset and incrementing it. Currently, I do not believe I can increase concurrency on acceptance of connections as the lack of concurrency there is for race conditions, but there can be an improvement for dispatching. What I can do is have two separate buffers, one for accepting connections and another for putting connections in. The threads will take from the acceptance buffer without having to wait for the dispatch to finish inserting connections into the buffer and the dispatch will input connections without having to wait for a thread to finish taking a connection. Once the threads run out of connections in their own buffer, I would import the buffer from the dispatcher to the threads and reset the buffer assigned to the dispatcher. With this, both the threads and dispatcher can run concurrently without the need of waiting for one another every time there is a connection requested.