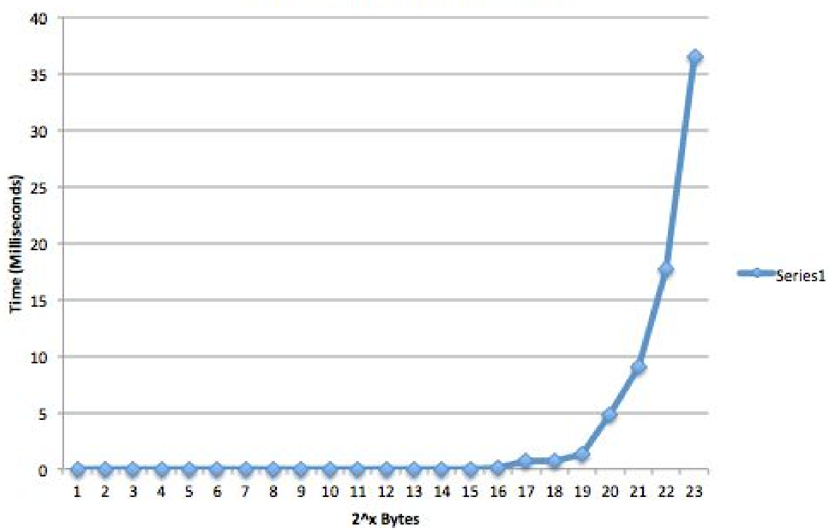
Samuel Carroll
CptS 411
Project 1 Writeup
September 17, 2019

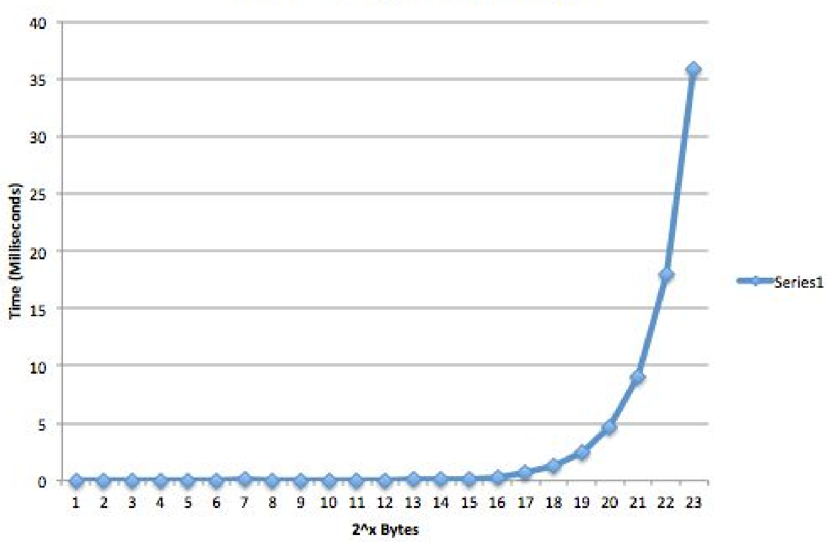
Intro:

For this assignment I chose to run my calculations using my non-blocking graphs. The way I set up the code I used a wait call immediately after calling MPI_Irecv, because of this my final results were fairly close for both blocking and non-blocking attempts as forcing a non-blocking call to wait right after it's called is effectively making it a blocking call. As I stated the two graphs I've included below are my non-blocking send and receive graphs; with the first graph being the send graph and the second graph being the receive graph.

Non-Blocking Send Graph



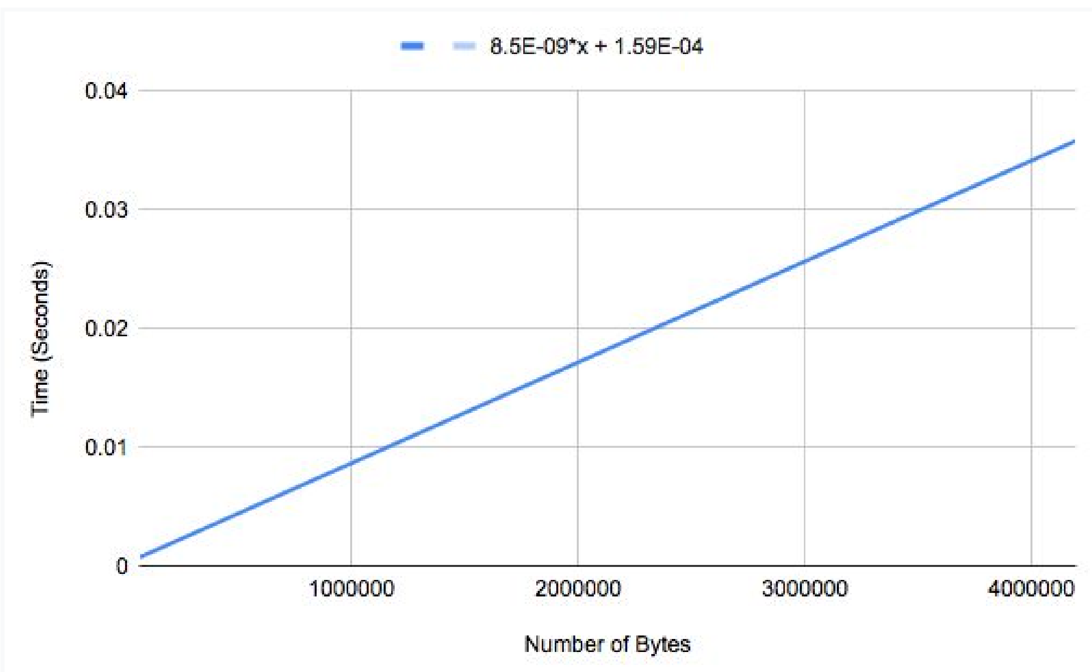
Non-Blocking Receive Graph



Calculations:

The first two calculations I focused on were latency and buffer size. For latency it comes from calculating the average time value for the flat portion of one of the graphs. For this I chose to use the non-blocking receive graph, in this case I used the number of bytes ranging from 2^1 bytes up to 2^15 which is the final point before it appeared to start turning up and no longer kept that flat line. For the average calculation I recorded a value of 0.03895 milliseconds, however latency is generally recorded using microseconds so in converting this the final value was 38.95 microseconds. For buffer size I again used the non-blocking receive and send data. For this one I was looking for the value where the send and receive times got to be almost identical to each other. I determined this value to be at 2^19 bytes (about ½ MB's), thus on the basis of my collected data I would estimate the network buffer size to be around 524,288 bytes.

My final calculation was to find the bandwidth across the system. For this portion I again chose to use the non-blocking receive graph. In this case I know that 1/bandwidth = μ, thus bandwidth = 1/μ, and I also know that μ is equal to the slope of the graph from the points on the graph where it starts to become more linear.



You'll notice the above graph looks different from the previous ones, this graph represents the line that occurs from 65,536 bytes to 4,194,304 bytes, which is the portion of the graph that I found to show a straight line that could provide a reliable slope for estimating bandwidth. In this case the slope was found to be 8.5E-9 as can be seen at the top of the above graph. With this slope as my μ value I can calculate bandwidth to be 1/(8.5E-9) = 11,764,705.8824 bytes/second, converting this to MB I got 117.65 MB/second for the estimated bandwidth of the pleiades cluster.