Programming Assignment 1:
Deriving the estimates for Latency, Bandwidth, and Network Buffer Size
Marcus Blaisdell

To derive the estimated values for Latency, Bandwidth, and Network Buffer size, we must test the network empirically. To do this, I used the standard MPI system functions and sent a message of increasing size across the network, tracking the amount of time required to send each message and plotting these values in tables and graphs for evaluation.
I began with the smallest possible message size of one byte and then double the size of the message on each successive iteration until the message reaches 2MB in size. This allows me to observe any trends in the speed of the message across the network.
In order to stress the network, I use non-consecutive processors to force the message to move across the network and not be limited to internal communications within one chip. I used the example provided in lecture and sent the message from processor 4 and receive it on processor 1.

Deriving the estimates:

There are two methods for which the estimates must be derived. The Blocking method, and the non-Blocking method. For each method, I needed to get at least ten examples to be able to create reliable averages. To achieve this, I used much of the sample code provided in "send_recv_test.c" and modified it to begin sending a message of size 1 byte, and using a while loop, continue to double it on each iteration until the size reached 2MB. I used a unix script file to run the program 10 times. I attempted to accomplish this all in the C code but I received an error when trying to close the MPI port and so as a work-around, I utilized the script file to individually run the program multiple times, forcing the MPI to go out of scope and allowing me to open a new session. I use this method for both Blocking and nonblocking. The script file is "loopRun.sh". This proved to be effective and was, on average, running ten iterations of each communication method in approximately 20 seconds.

As I examined the first set of ten runs, I noticed that many of them had excessive run times compared to some of the other runs. The runs with smaller times showed the distinctive upturned curve at the larger message sizes that was expected and some showed a consistently level curve for the majority of the message sizes. If we examine these graphs closely, we can see that the large times seem to be obscuring the details by taking an excessive amount of time to process any sized message and were therefore not useful for deriving my estimates. I then repeated my tests to get a total of thirty complete runs. Of the 30, I documented 18 in this report, Table 1 and Charts 1-18, to demonstrate the issue. To derive the estimates, I had to filter the data to remove data points that were preventing me from observing the necessary trends, Table 2.
The same issue was noted for the nonblocking method and the same filtering was applied to derive the estimates.

Blocking								
Run	Time for first send	Time for first receive	send 1MB	send-2MB	receive 1MB	receive 2MB	Send Slope per MB	Receive Slope per MB
1	2073	2200	505	926	490	894	421	404
2	2207	17210	495	2163	478	2158	1668	1680
3	3294	3013	1915	1893	1890	1905	-22	15
4	8501	16555	1906	1892	1908	1888	-14	-20
5	6582	13783	464	814	456	783	350	327
6	12952	13100	1809	1800	1876	1889	-9	13
7	3146	8484	1810	1808	1871	1909	-2	38
8	8610	7858	1889	-998128	1874	-998136	-1000017	-1000010
9	2106	2237	1874	1895	1790	1785	21	-5
10	6574	6348	1845	1876	1799	1817	31	18
11	2189	6222	447	810	458	800	363	342
12	7926	2794	1833	1827	1803	1811	-6	8
13	10280	6942	492	860	441	790	368	349
14	2157	2283	509	809	532	791	300	259
15	2254	12704	452	1871	484	3040	1419	2556
16	6601	6744	486	827	480	807	341	327
17	2289	2151	503	907	458	804	404	346
18	4689	4987	457	873	435	826	416	391
Min	2073	2151					-1000017	-1000010
Max	12952	17210					1668	2556
Average	5246.111	7534.17					-55220.4	-55164.5556

Table 1: Chart of relevant data points for determining the Latency and Bandwidth for Blocking Parallel Communications.

In table 1, we see that the data is noisy and does not provide usable estimates. By analyzing the data and determining which samples had real value and which were obscuring the data, I was able to filter the data to be usable, table 2.

Blocking									
Run	Time for first send	Time for first receive	send 1MB	send-2MB	receive 1MB	receive 2MB	Send Slope per MB	Receive Slope per MB
1	2073	2200	505	926	490	894	421	404
5	6582	13783	464	814	456	783	350	327
11	2189	6222	447	810	458	800	363	342
13	10280	6942	492	860	441	790	368	349
14	2157	2283	509	809	532	791	300	259
16	6601	6744	486	827	480	807	341	327
17	2289	2151	503	907	458	804	404	346
18	4689	4987	457	873	435	826	416	391
Min	2073	2151					300	259
Max	10280	13783					421	404
Average	4607.5	5664					370.38	343.125

Table 2: A filtered revision of the chart of relevant data points for determining the Latency and Bandwidth for Blocking Parallel Communications.

Latency:
2,073us, transmission only,
4,224us, transmit and receive

The minimum time for the first transmission is 2,073 micro-second. This indicates that the network is capable of sending its first transmission within that amount of time indicating that this is the minimum latency. If we consider that the latency is the total amount of time to complete a transmission from the point it is sent to the point it is received, then we sum the minimum values of the time for the first send and the time for the first receive to obtain the value 4,224 micro-seconds.

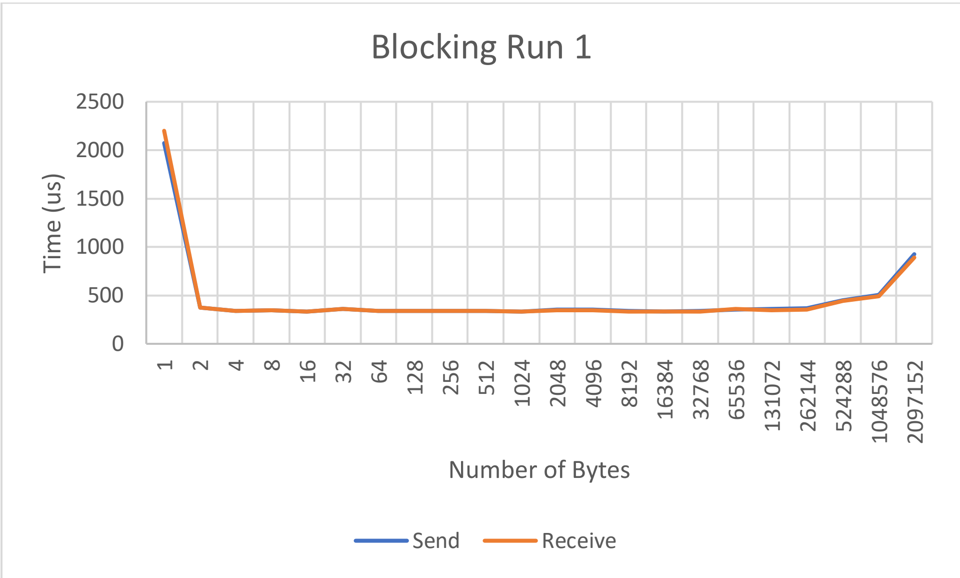
Bandwidth:
370.38us/MB transmit
343.125us/MB receive

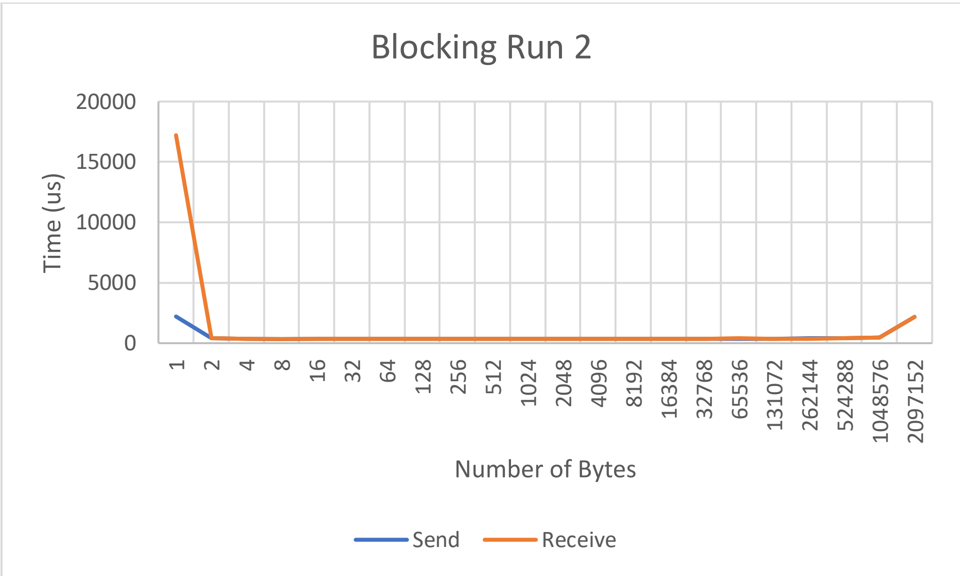
By examining the first table, we see that one sample had an erroneous time value that was negative and very large. There are also multiple samples that have a negative slope for the transmission speed between 1MB and 2MB. The transmission time should not decrease as the size of the message increases. By examining the data, we see that in these cases, the transmission time is greater than 1 milli-second and we can see several examples that have times less than 1 milli-second indicating that the system is capable of transmitting a 2MB message in less than 1 milli-second. Therefore, if we remove all samples with erroneous values, samples with negative slope, and samples with 1MB or 2MB transmission times greater than 1ms, then we can see that the slopes become more consistent and we can see that the average slope for transmission is

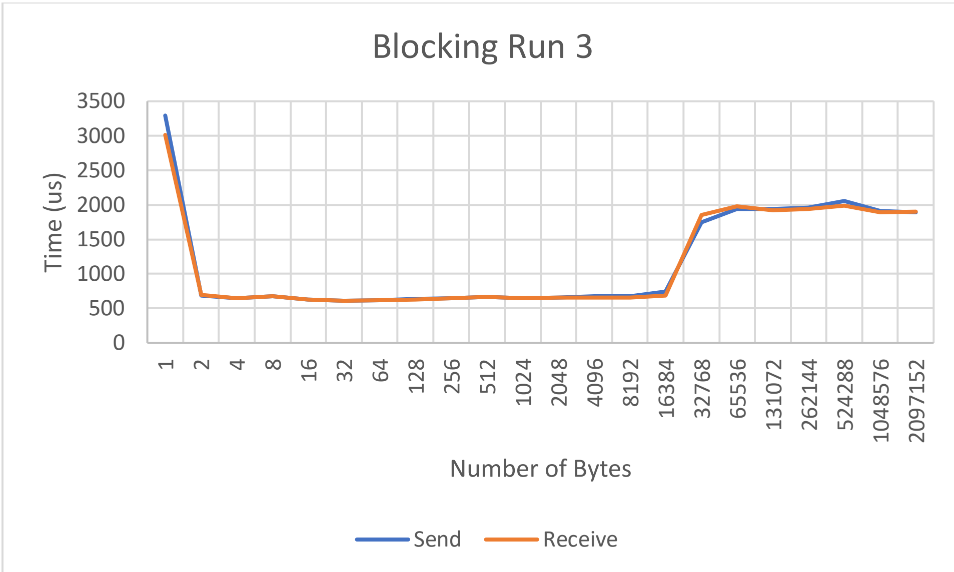
370.38 micro-seconds per megabyte and the average slope for receipt is 343.125 micro-seconds per megabyte.

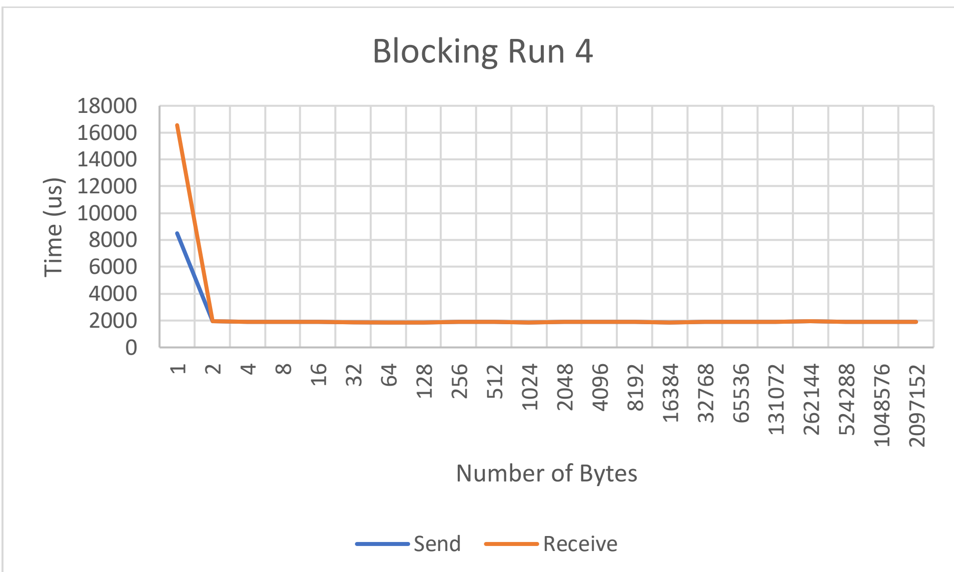
Network buffer size:
1MB to 2MB

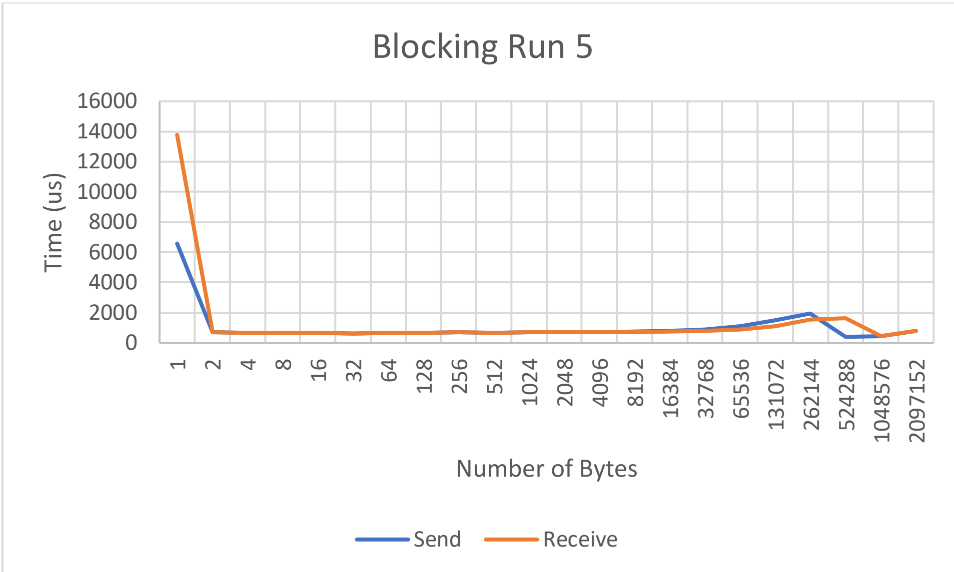
By examining the graphs, we can see a notable upturn in the transmission speed between 1MB and 2MB in graphs 1, 2, 5, 11, 13, 14, 15, 16, 17, 18 that indicates that the network buffer size is between 1MB and 2MB.

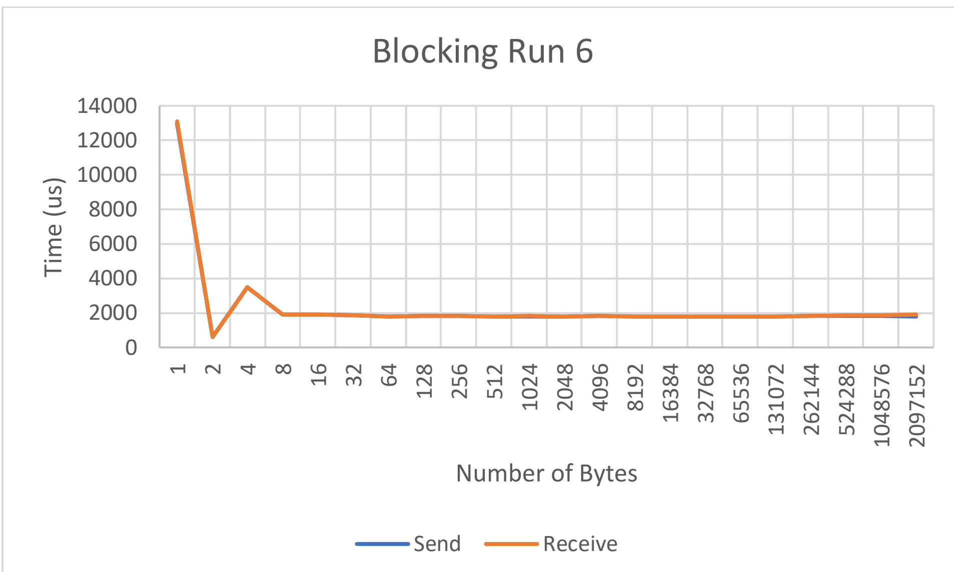


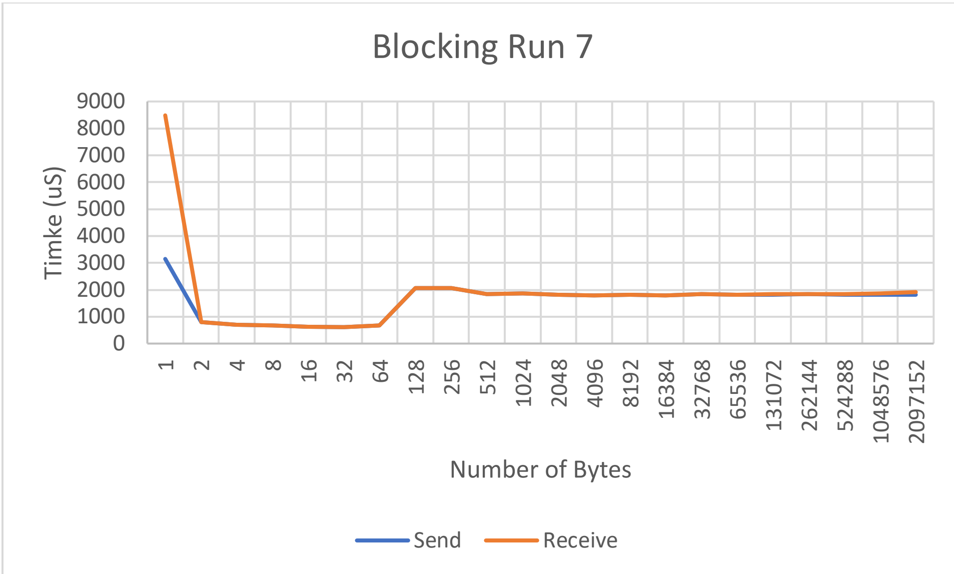



Blocking Run 3
3500
3000
2500
2000
1500
1000
500
0
Time (us)
1
2
4
8
16
32
64
128
256
512
1024
2048
4096
8192
16384
32768
65536
131072
262144
524288
1048576
2097152
Number of Bytes
Send
Receive

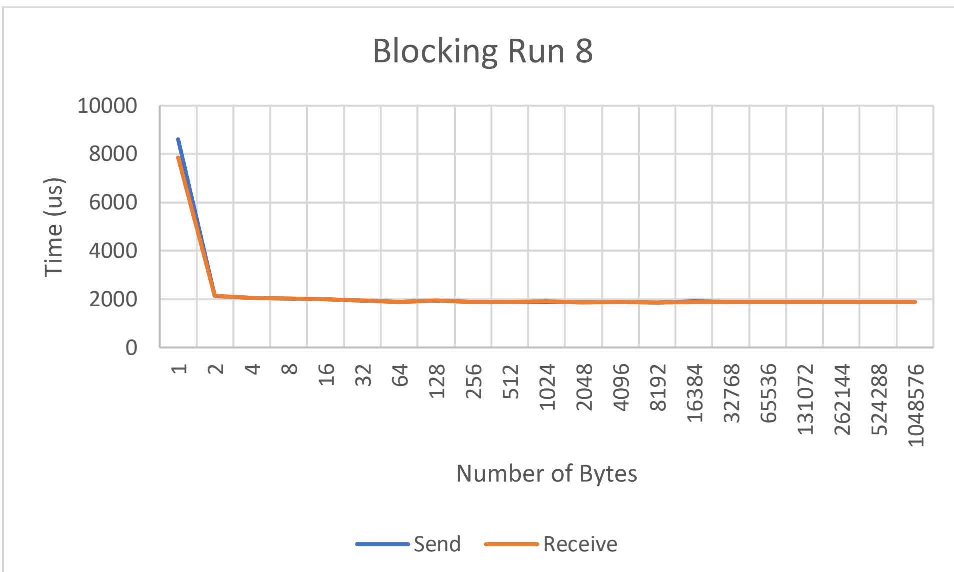

Blocking Run 4
18000
16000
14000
12000
10000
8000
6000
4000
2000
0
Time (us)
1
2
4
8
16
32
64
128
256
512
1024
2048
4096
8192
16384
32768
65536
131072
262144
524288
1048576
2097152
Number of Bytes
Send
Receive


Blocking Run 5
16000
14000
12000
10000
8000
6000
4000
2000
0
Time (us)
1
2
4
8
16
32
64
128
256
512
1024
2048
4096
8192
16384
32768
65536
131072
262144
524288
1048576
2097152
Number of Bytes
Send
Receive

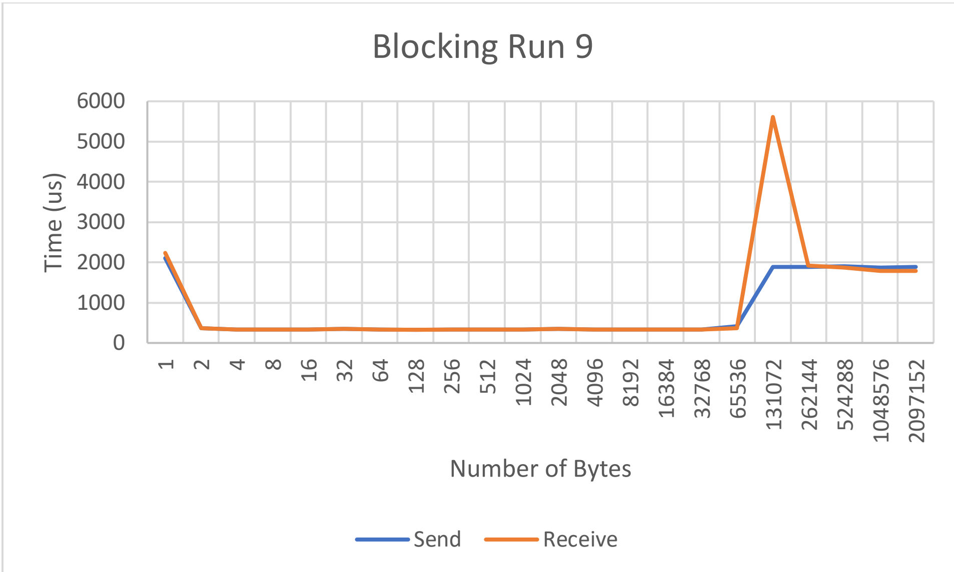

Blocking Run 6
14000
12000
10000
8000
6000
4000
2000
0
Time (us)
1
2
4
8
16
32
64
128
256
512
1024
2048
4096
8192
16384
32768
65536
131072
262144
524288
1048576
2097152
Number of Bytes
Send
Receive

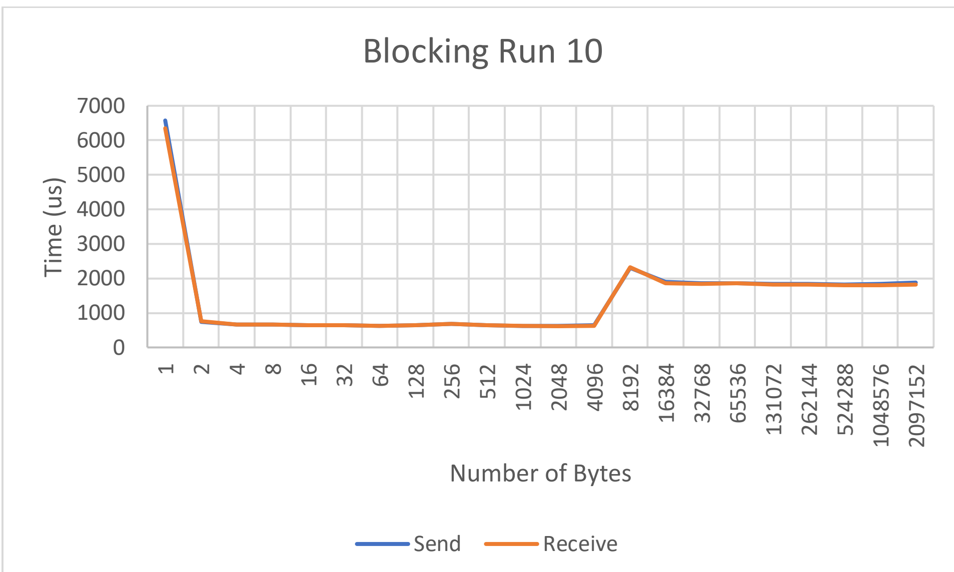


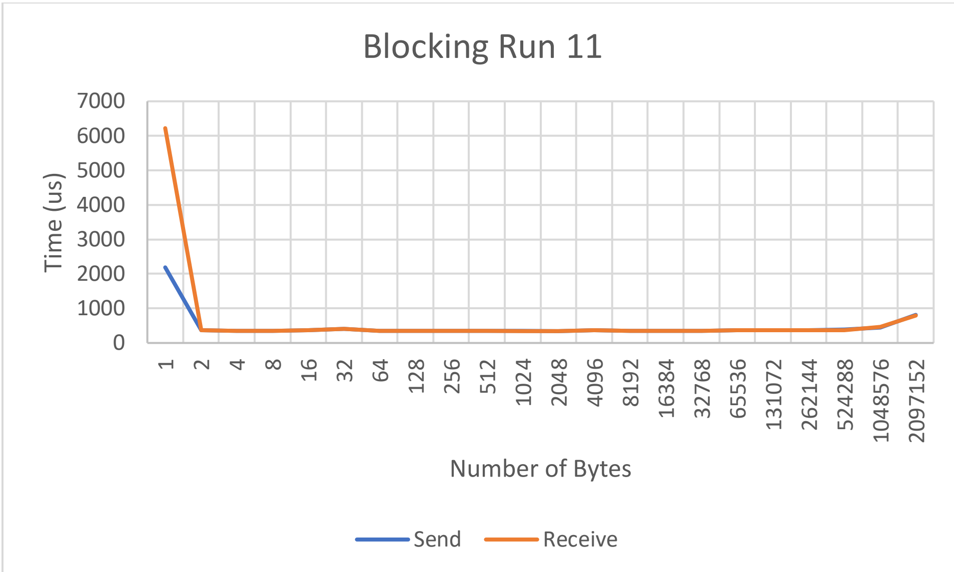
Blocking Run 7
Timke (uS)
9000
8000
7000
6000
5000
4000
3000
2000
1000
0
1
2
4
8
16
32
64
128
256
512
1024
2048
4096
8192
16384
32768
65536
131072
262144
524288
1048576
2097152
Number of Bytes
Send
Receive



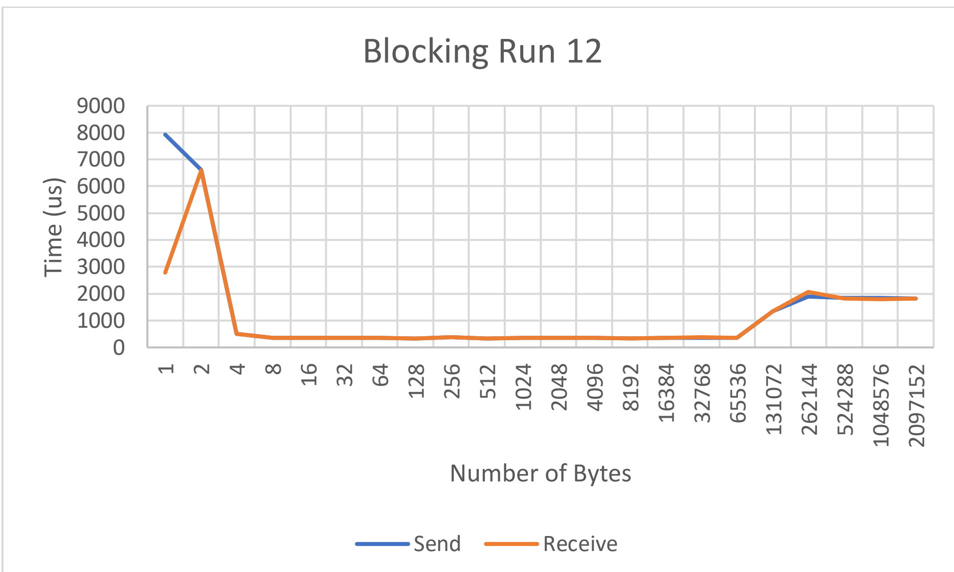
Blocking Run 8
Time (us)
10000
8000
6000
4000
2000
0
1
2
4
8
16
32
64
128
256
512
1024
2048
4096
8192
16384
32768
65536
131072
262144
524288
1048576
Number of Bytes
Send
Receive


Blocking Run 9
6000
5000
4000
3000
2000
1000
0
Time (us)
1
2
4
8
16
32
64
128
256
512
1024
2048
4096
8192
16384
32768
65536
131072
262144
524288
1048576
2097152
Number of Bytes
Send
Receive

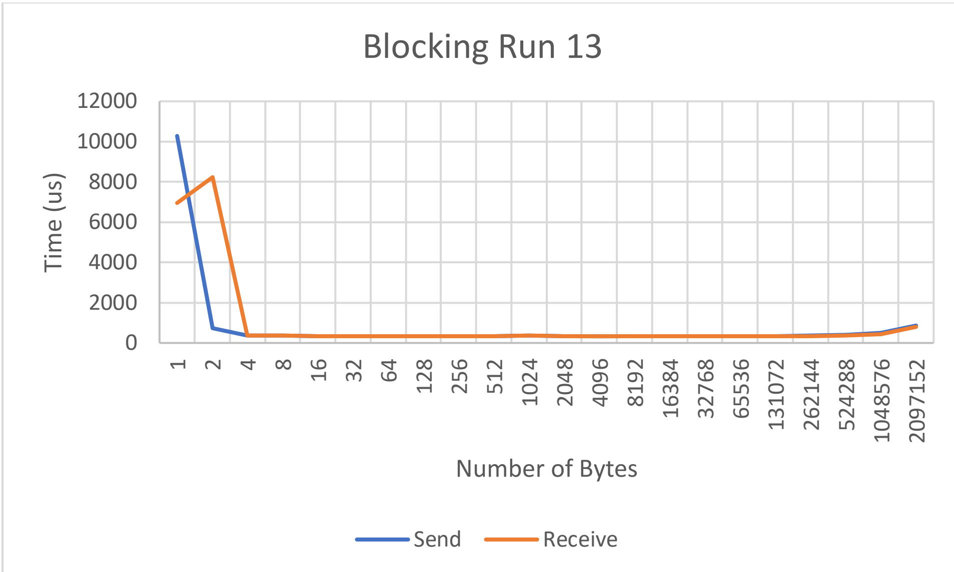

Blocking Run 10
7000
6000
5000
4000
3000
2000
1000
0
Time (us)
1
2
4
8
16
32
64
128
256
512
1024
2048
4096
8192
16384
32768
65536
131072
262144
524288
1048576
2097152
Number of Bytes
Send
Receive

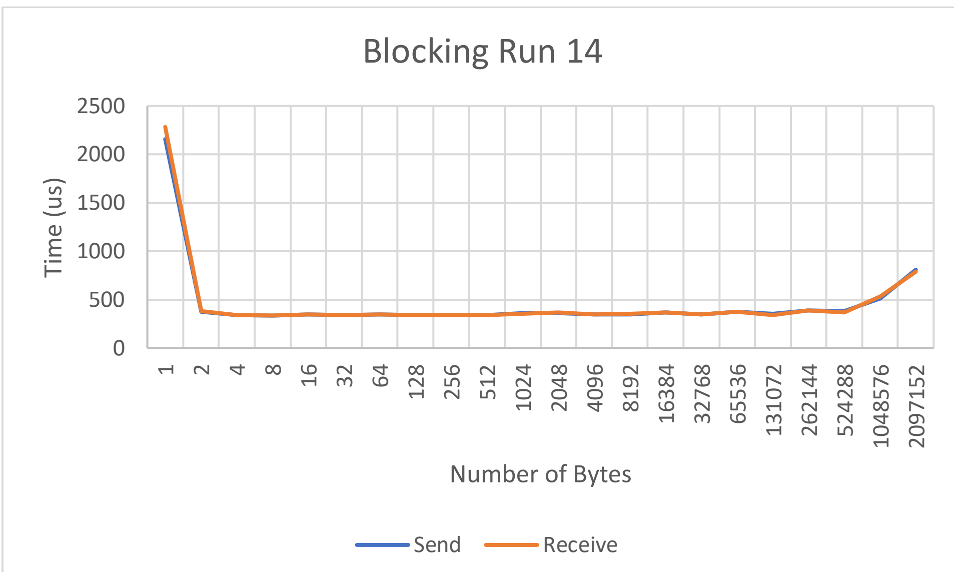


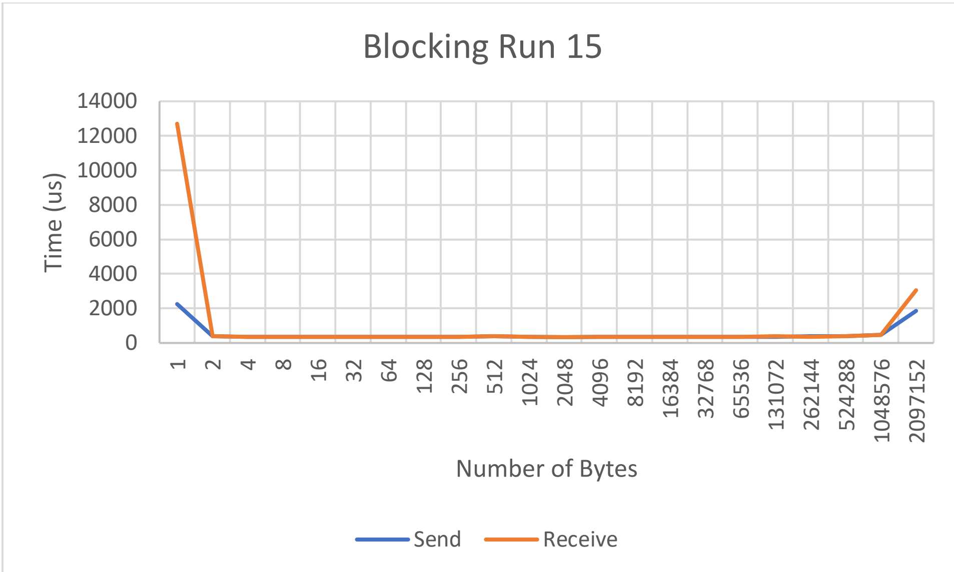
Blocking Run 11
7000
6000
5000
4000
3000
2000
1000
0
Time (us)
1
2
4
8
16
32
64
128
256
512
1024
2048
4096
8192
16384
32768
65536
131072
262144
524288
1048576
2097152
Number of Bytes
Send
Receive

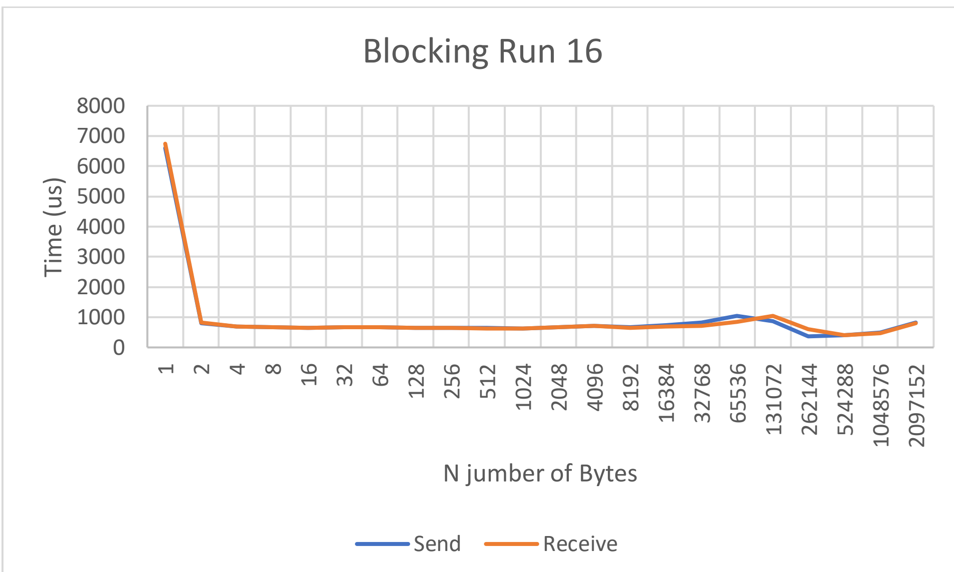


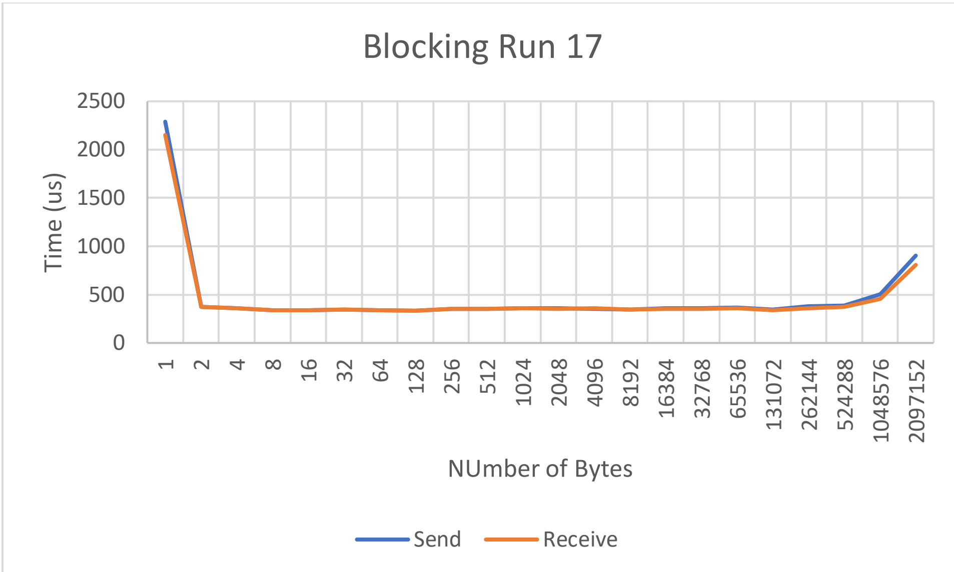
Blocking Run 12
9000
8000
7000
6000
5000
4000
3000
2000
1000
0
Time (us)
1
2
4
8
16
32
64
128
256
512
1024
2048
4096
8192
16384
32768
65536
131072
262144
524288
1048576
2097152
Number of Bytes
Send
Receive

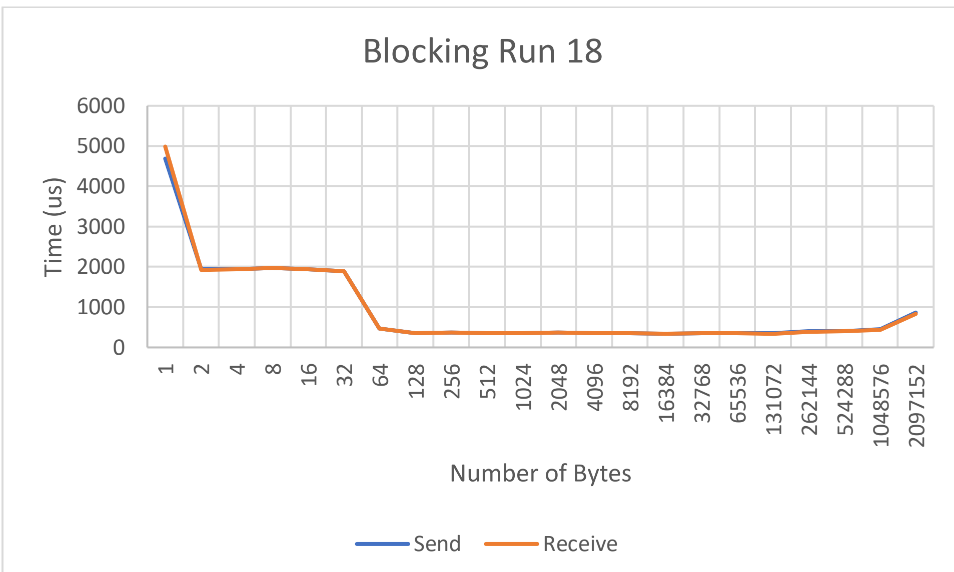

Blocking Run 13
12000
10000
8000
6000
4000
2000
0
Time (us)
1
2
4
8
16
32
64
128
256
512
1024
2048
4096
8192
16384
32768
65536
131072
262144
524288
1048576
2097152
Number of Bytes
Send
Receive


Blocking Run 14
2500
2000
1500
1000
500
0
Time (us)
1
2
4
8
16
32
64
128
256
512
1024
2048
4096
8192
16384
32768
65536
131072
262144
524288
1048576
2097152
Number of Bytes
Send
Receive


Blocking Run 15
Time (us)
14000
12000
10000
8000
6000
4000
2000
0
1
2
4
8
16
32
64
128
256
512
1024
2048
4096
8192
16384
32768
65536
131072
262144
524288
1048576
2097152
Number of Bytes
Send
Receive


Blocking Run 16
Time (us)
8000
7000
6000
5000
4000
3000
2000
1000
0
1
2
4
8
16
32
64
128
256
512
1024
2048
4096
8192
16384
32768
65536
131072
262144
524288
1048576
2097152
N jumber of Bytes
Send
Receive





Non Blocking:

Non-Blocking								
Run	Time for first send	Time for first receive	send 1MB	send-2MB	receive 1MB	receive 2MB	Send Slope per MB	Receive Slope per MB
1	2081	9265	478	1930	468	2782	1452	2314
2	2074	2193	1947	1912	1914	1884	-35	-30
3	2113	4144	513	926	505	892	413	387
4	2123	14135	445	799	479	875	354	396
5	2146	4499	498	852	548	849	354	301
6	8644	17929	515	864	476	799	349	323
7	12815	19276	483	807	462	797	324	335
8	2742	3598	463	2165	448	2536	1702	2088
9	3162	2953	1891	1900	1896	1910	9	14
10	4778	4931	1872	1885	1867	1900	13	33
Min	2074	2193					-35	-30
Max	12815	19276					1702	2314
Average	4267.8	8292.3					493.5	616.1

Non-Blocking								
Run	Time for first send	Time for first receive	send 1MB	send-2MB	receive 1MB	receive 2MB	Send Slope per MB	Receive Slope per MB
3	2113	4144	513	926	505	892	413	387
4	2123	14135	445	799	479	875	354	396
5	2146	4499	498	852	548	849	354	301
6	8644	17929	515	864	476	799	349	323
7	12815	19276	483	807	462	797	324	335
Min	2113	4144					324	301
Max	12815	19276					413	396
Average	5568.2	11996.6					358.8	348.4

Latency:
2,113us Transmit only,
6,257us Transmit and receive.

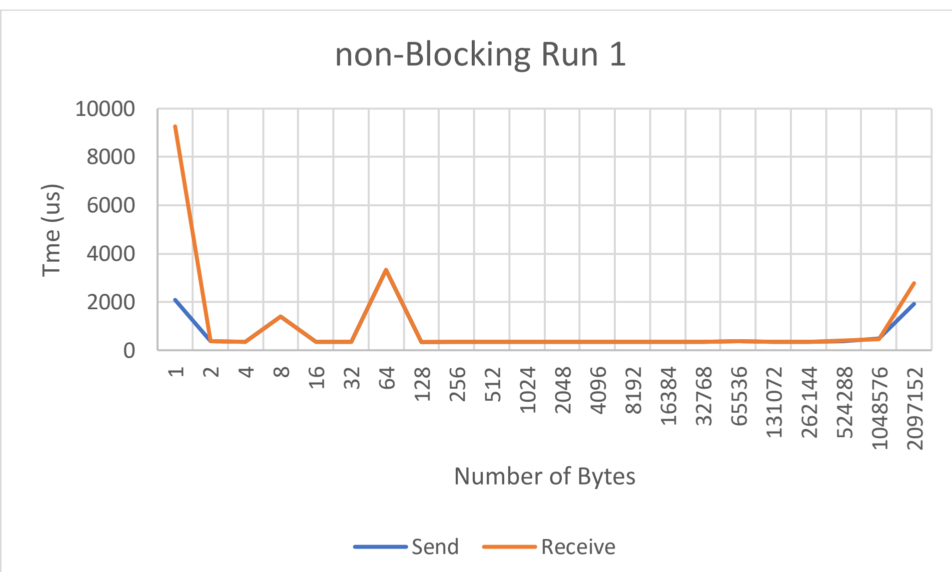
The minimum time for the first transmission is 2,113 micro seconds. This indicates that the network is capable of sending its first transmission within that amount of time indicating that this is the approximate minimum latency. If the latency is considered to be the amount of time required to both transmit and receive a message and we sum these minimum values, we see a value of 6,257 micro-seconds.

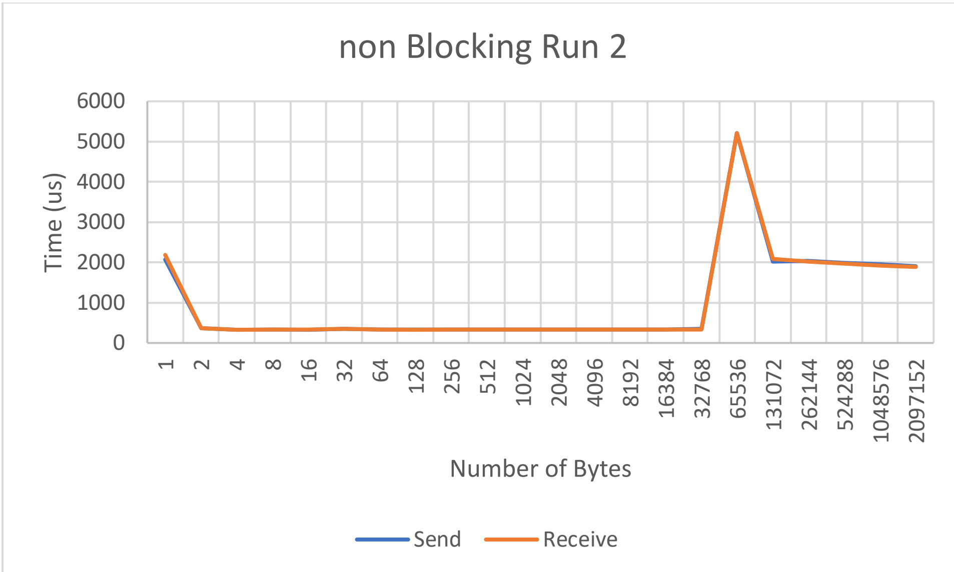
Bandwidth:
358.8us, transmit
348.4us, receive

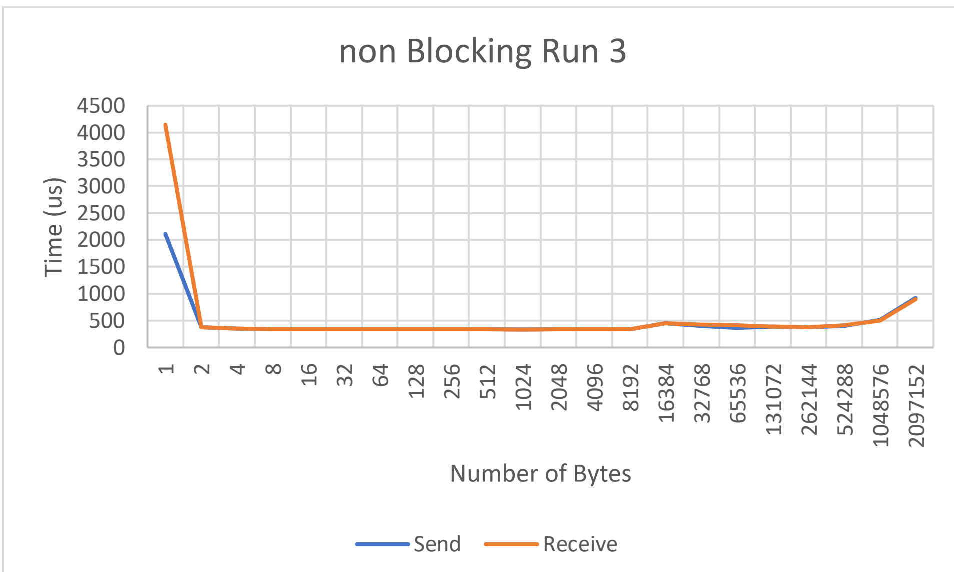
By examining the first table, we see that there are multiple samples that have a negative slope for the transmission speed between 1MB and 2MB. The transmission time should not decrease as the size of the message increases. By examining the data, we see that in these cases, the transmission time is greater than 1 milli-second and we can see several examples that have times less than 1 milli-second indicating that the system is capable of transmitting a 2MB message in less than 1 milli-second. Therefore, if we remove all samples with negative slope, and samples with 1MB or 2MB transmission times greater than 1ms, then we can see that the slopes become more consistent and we can see that the average slope for transmission is 358.8 micro-seconds per megabyte and the average slope for receipt is 348.4 micro-seconds per megabyte.
In comparison to the Blocking method, the average transmission time of the non-Blocking method is approximately 1.03 times faster and the average receipt time of the Blocking method is 1.015 times faster than the non-Blocking method.

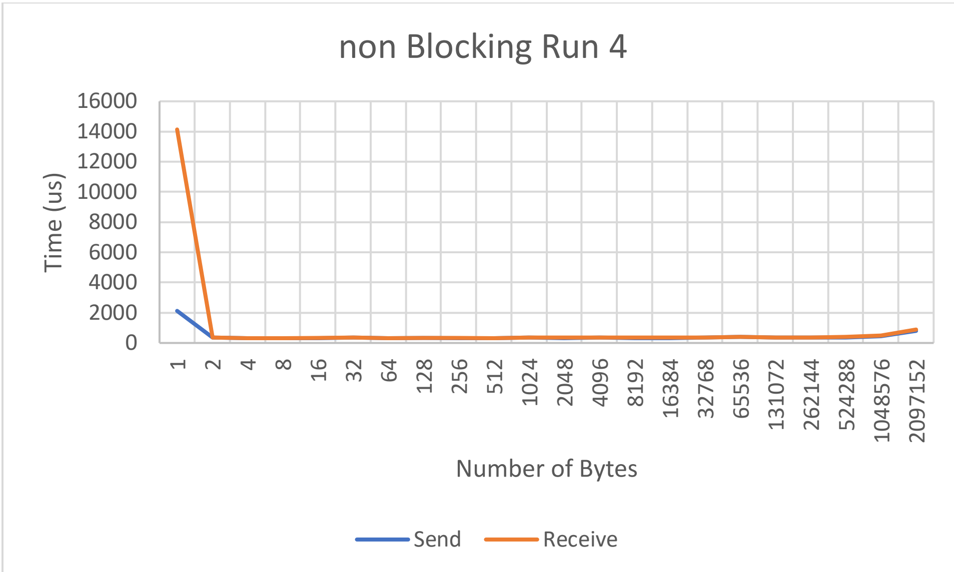
Network buffer size:
1MB to 2MB

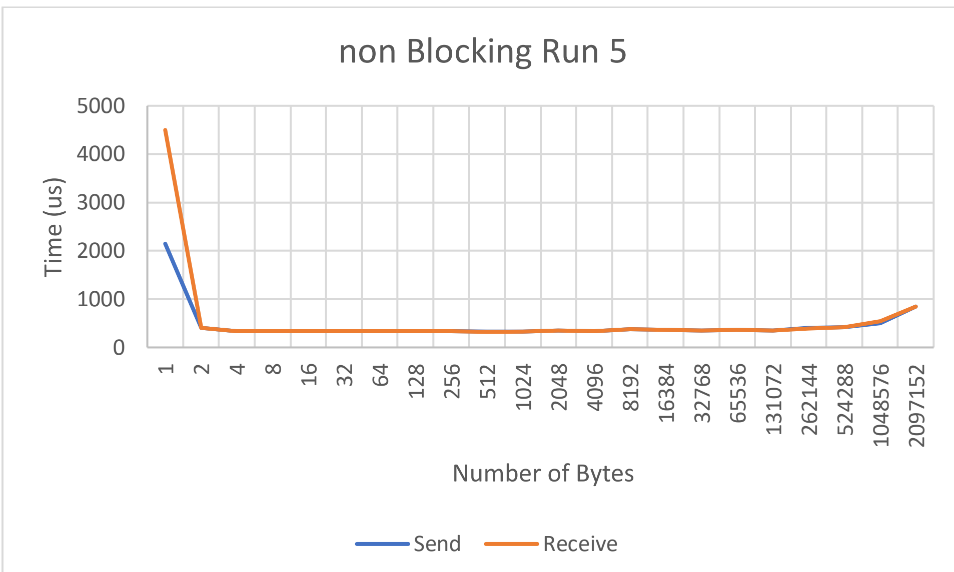
By examining the graphs, we can see a notable upturn in the transmission speed between 1MB and 2MB in graphs 1, 3, 4, 5, 6, 8 that indicates that the network buffer size is between 1MB and 2MB which is consistent with what was observed in the Blocking method.

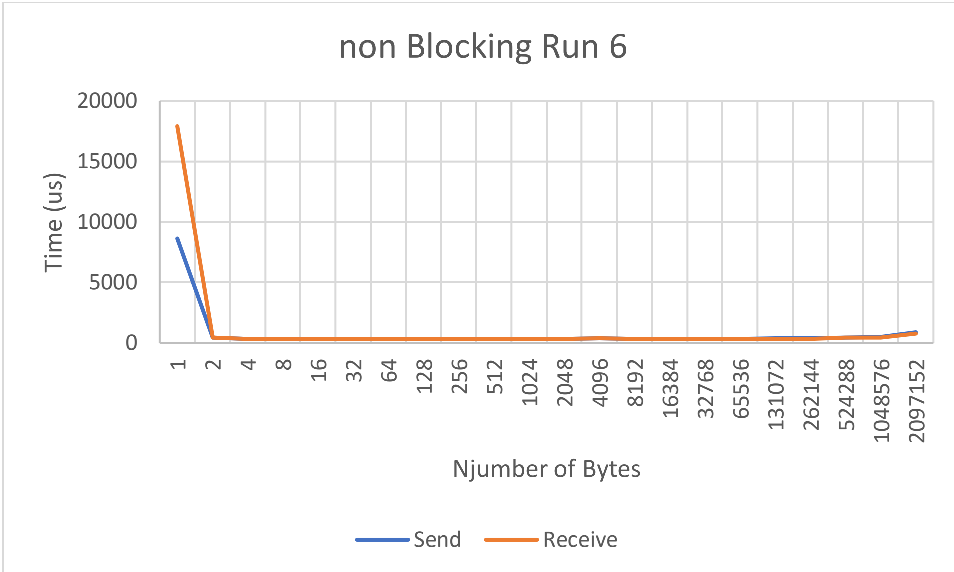


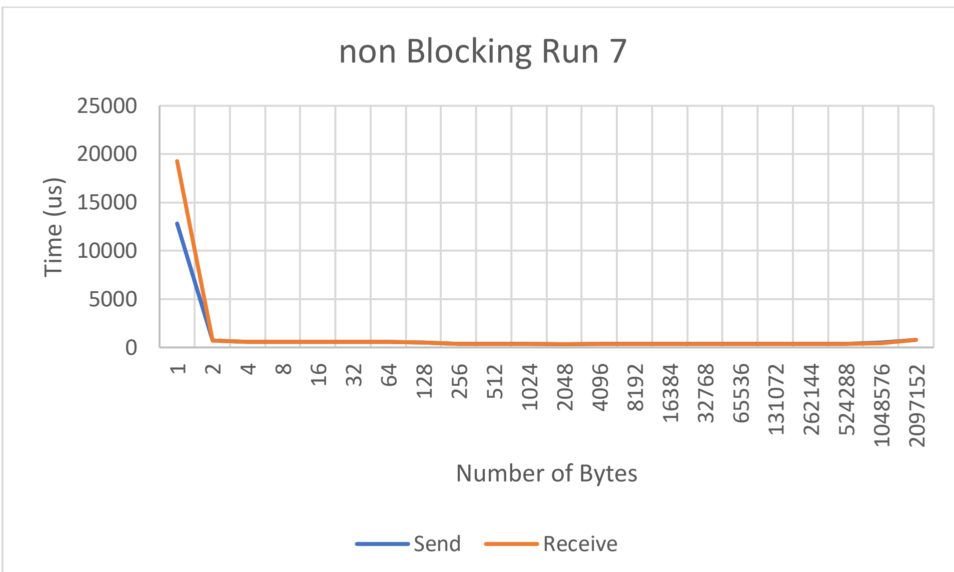

non Blocking Run 2
6000
5000
4000
3000
2000
1000
0
Time (us)
1
2
4
8
16
32
64
128
256
512
1024
2048
4096
8192
16384
32768
65536
131072
262144
524288
1048576
2097152
Number of Bytes
Send
Receive

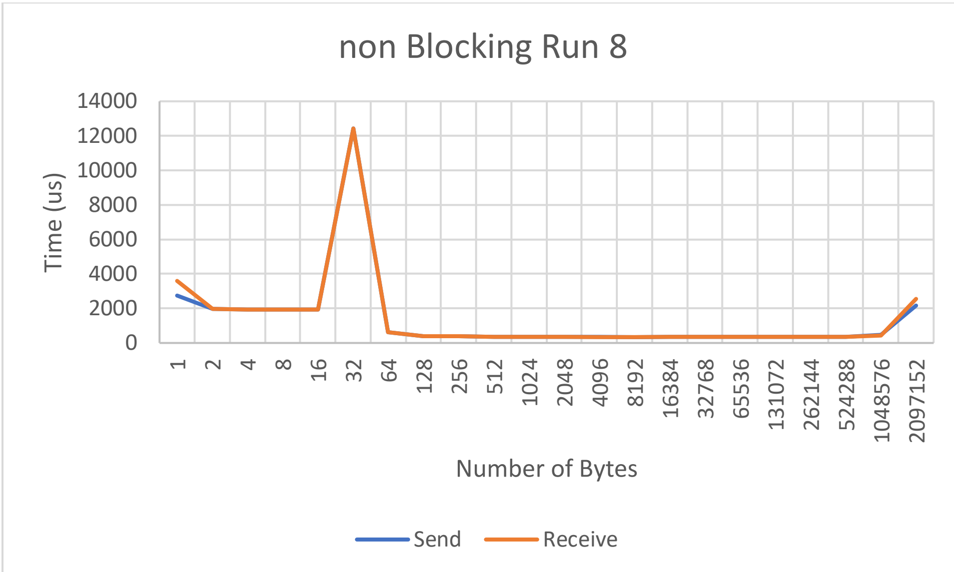

non Blocking Run 3
4500
4000
3500
3000
2500
2000
1500
1000
500
0
Time (us)
1
2
4
8
16
32
64
128
256
512
1024
2048
4096
8192
16384
32768
65536
131072
262144
524288
1048576
2097152
Number of Bytes
Send
Receive

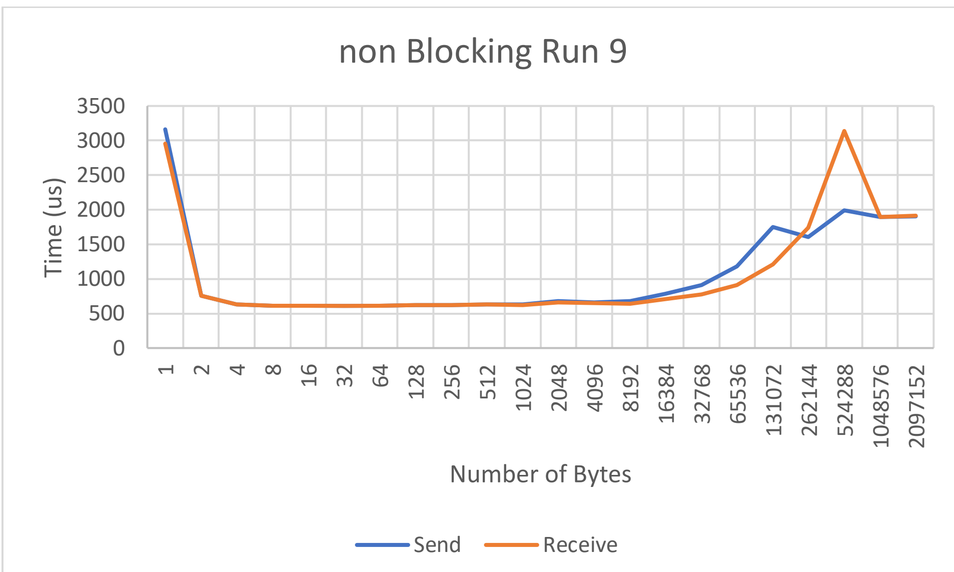

non Blocking Run 4
16000
14000
12000
10000
8000
6000
4000
2000
0
Time (us)
1
2
4
8
16
32
64
128
256
512
1024
2048
4096
8192
16384
32768
65536
131072
262144
524288
1048576
2097152
Number of Bytes
Send
Receive

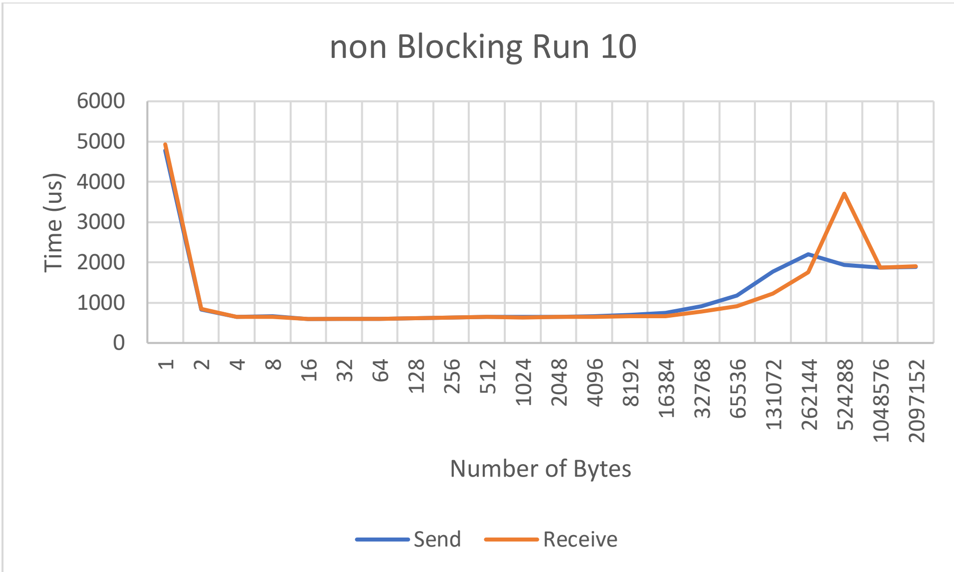

non Blocking Run 5
5000
4000
3000
2000
1000
0
Time (us)
1
2
4
8
16
32
64
128
256
512
1024
2048
4096
8192
16384
32768
65536
131072
262144
524288
1048576
2097152
Number of Bytes
Send
Receive


non Blocking Run 6
20000
15000
10000
5000
0
Time (us)
1
2
4
8
16
32
64
128
256
512
1024
2048
4096
8192
16384
32768
65536
131072
262144
524288
1048576
2097152
Njumber of Bytes
Send
Receive


non Blocking Run 7
25000
20000
15000
10000
5000
0
Time (us)
1
2
4
8
16
32
64
128
256
512
1024
2048
4096
8192
16384
32768
65536
131072
262144
524288
1048576
2097152
Number of Bytes
Send
Receive


non Blocking Run 8
Time (us)
14000
12000
10000
8000
6000
4000
2000
0
1
2
4
8
16
32
64
128
256
512
1024
2048
4096
8192
16384
32768
65536
131072
262144
524288
1048576
2097152
Number of Bytes
Send
Receive


non Blocking Run 9
Time (us)
3500
3000
2500
2000
1500
1000
500
0
1
2
4
8
16
32
64
128
256
512
1024
2048
4096
8192
16384
32768
65536
131072
262144
524288
1048576
2097152
Number of Bytes
Send
Receive


non Blocking Run 10
6000
5000
4000
3000
2000
1000
0
Time (us)
1
2
4
8
16
32
64
128
256
512
1024
2048
4096
8192
16384
32768
65536
131072
262144
524288
1048576
2097152
Number of Bytes
Send
Receive