Report on Project 1

Message Communication Time.

CS415

February 20, 2017

Evan Su

Ping Pong Time Test

Overview:

The ping pong test is the measurement of time for a message to be sent to another process and back again. One goal of the test is to figure out which method of communication is faster, message passing within the same box or message passing between boxes. The other goal is to determine the average ping pong time for each message passing method.

Test Methodology:

A program, called mpi_pingpong, is written to test the send and receive time between processes. The program will measure the time elapsed for message sent to another process and back to the sender. The size of the message is 2 integers. The size of 2 integers was arbitrary chosen. The program will measure the time of 50 sends and receives. The results are outputted on the terminal in the format of CSV (comma separated values). For each trial, the program will run twice; once for the same box and another for different boxes.

Data Analysis:

The results of the test are summarized in Table 1.1. Raw data is also available in the zip called "Data and Analysis." The name of the file sets are "one_box#" and "two_box#" with # corresponding with the trial number. The results show that message passing in the same box is about 2 - 3 microseconds. Whereas, message passing between different boxes is around 52 microseconds. The fast speed of same box message passing is a result of shared memory. The time of shared box is limited to how fast each process can read and write to the shared memory. Message passing between two different boxes had the addition time cost of sending a packet through a network.

Conclusion:

Message passing between processes in the same box is significantly faster than in different boxes. The same box is at least 20 times faster. The high time cost of network usage implies that any message passing should ideally be between processes in the same box.

Packet Size Test

Overview

MPI message passing between two boxes uses packets. The goal of the packet size test is to determine the behavior of MPI message passing at different message sizes. Specifically, the amount of data that can fit in a single packet.

Test Methodology:

A program, called mpi_packetCheck, is written to test time elapsed for a message sent to another process and back to the sender. The program will test each possible message size up to 10,000 integers. Message size of 10,000 integers was chosen because the data from 1,000 integers was insufficient and 100,000 integers used too much computational time. The program sends integers between two process on two different boxes to measure the communication time. Each message size test ran 50 times before the program tests the next amount.

Data Analysis:

Four trials were performed and the results were plotted on graph 2.1. Raw data is also available in the zip called "Data and Analysis." The name of the file set is "Timing#" with # corresponding with the trial number. Each trial ran on different days on different times. Trial 1 had the least network activity of 1 user. Trail 4 had the most network activity of 5 users.

The first 40 integer sized packets had a consistent large time of 140 microseconds compared to size packets afterwards with a time of 35 seconds. The large time must be caused by a special send routine for small packets. Around the 42nd integer sized packet, the time drastically decreased and subsequence packet sizes increased linearly until 500 integers. The message passing routines must be creating packets of exact message sizes. After 500 integers, the communication time then increased linearly but at a slower rate until 1947 integers. The message passing routines increased the size of the packets linearly but not exactly sized anymore. After 1947 integers, the communication time continually spiked up and plateaued. At this point, messages began to be split into different packets.

The trials had time spikes at different places but the size of the spikes was consistent. Each time spike is 143 microseconds in height. This implied that each time a spike occurred, an additional two packet (for back and forth) was made and sent. The number of packets that each process used can be determined by dividing the Ping-Pong time for a message by 143 microseconds. For example, between 2000 to 2780 integers, all trials used at least 2 packets. Looking at the upper bound of each number of packet interval for all trials (1800, 2800,5005, etc.), all packet carried at most 2^16 bits of data. For instance, around 2800 integers, the number of packets that carry the message jumped from 2 packets to 3 packets. Before the jump, the 2 packets must be full which implies that each packet is carry 1400 integers. 1400 integers are approximately 2^15.4 bits. Testing each upper bound show that no packet will exceed 2^16 bits.

Conclusion:

The maximum packet size appears to be 2^16 bits. Packets should be larger than 40 integers to avoid time cost of special packet handling methods. The send routine will increase packet size linearly until the limit of a packet is near or is reached. The send routine will often split the message into another packet before reaching the limit.

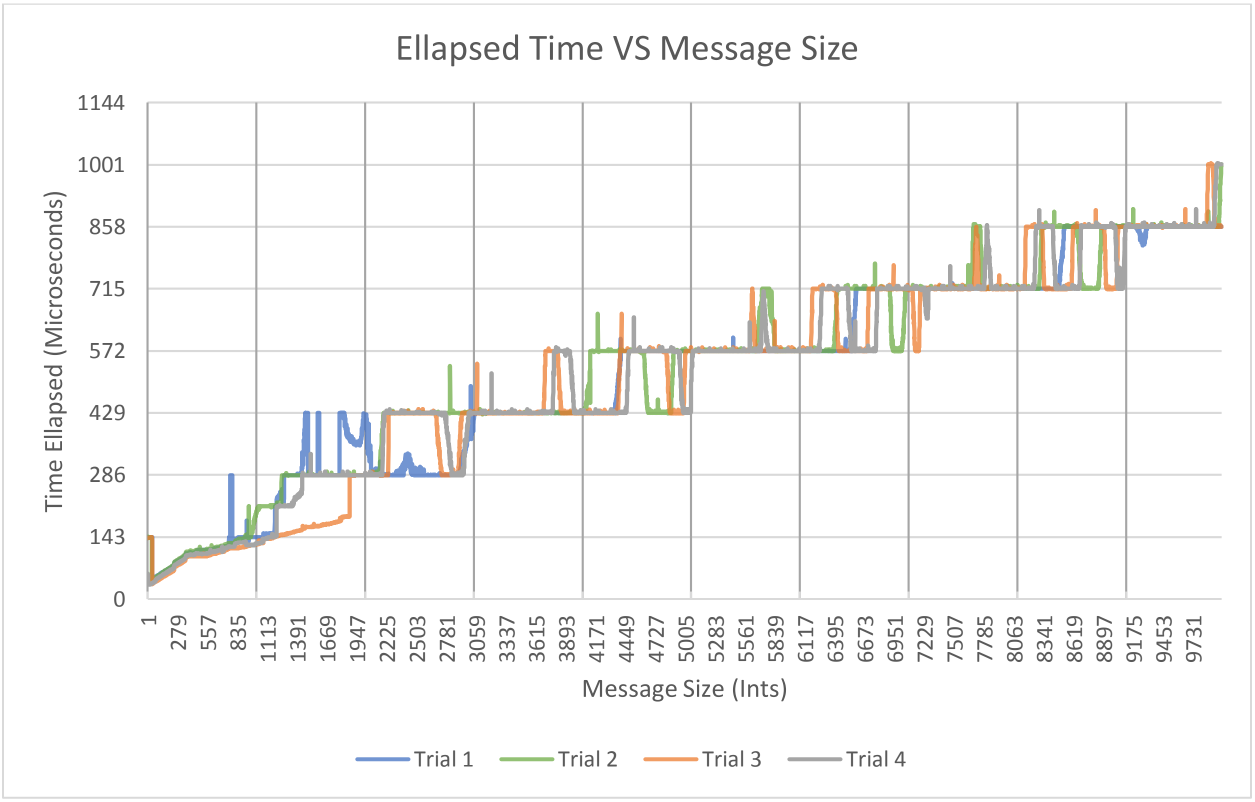
Graphs and Tables

Table 1.1

	One Box				Two Box			
Trial	MIN	MAX	Average	SD	MIN	MAX	Average	SD
1	1	5	2.142857	0.645497	57	73	59.71429	3.5
2	2	3	2.291667	0.45934	46	60	52.16327	3.52578
3	2	9	2.857143	1.040833	49	81	52.91837	6.214354

Time is measured in microseconds

Graph 2.1


Ellapsed Time VS Message Size
Time Ellapsed (Microseconds)
Message Size (Ints)
Trial 1
Trial 2
Trial 3
Trial 4