CS546 Parallel and distributed Processing

Programming Assignment

Report

The assignment carries out Evaluation of the simple MPI-IO program. It has two parts.

1. One is to simply gain some familiarity with MPI-IO. MPI-IO will likely be part of whatever I/O stack you will be using in an HPC environment. We want each MPI process to write its rank to a common file using The MPI-IO interface

2.The second part is to build a parallel I/O benchmark using MPI-IO. In this Benchmark, we will open a temporary file, write random data to a file, and close the file. Then we will open it again, read the data from the file and close it again. All operations are performed by each rank to a common file.

MPI IO provides a rich interface allowing us to describe Noncontiguous accesses in memory, file, or both and Collective I/These implementations allows to perform many transformations that result in better I/O performance

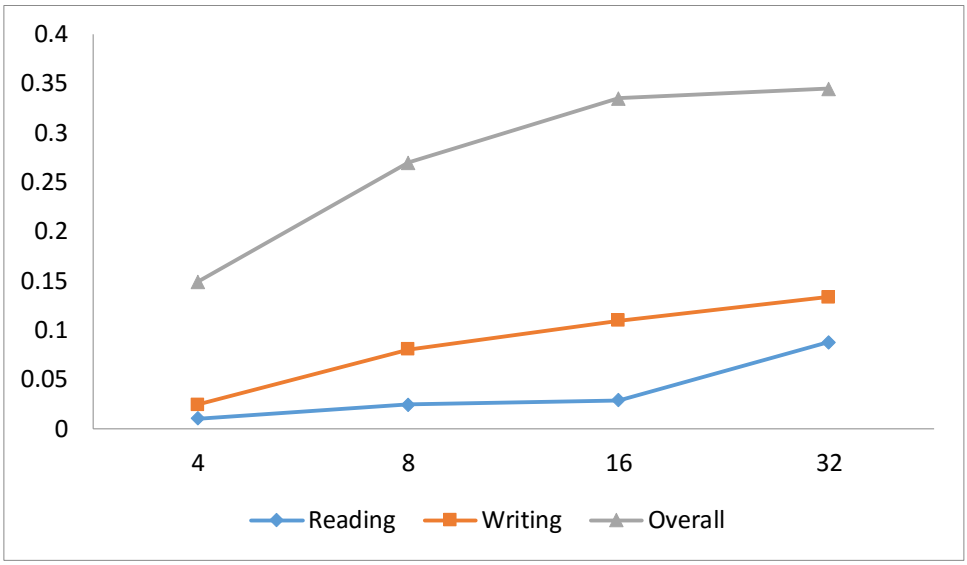
I've evaluated the parallel I/O benchmark using MPI-IO In this bench Mark, evaluated time taken for writing random data into a common file and again opening the same file, then reading data from a same common file ,calculated bandwidth and overall time for both reading and writing on Amazon Aws which includes orangeFS file system.

The below tables and plots for each 4,8,16 and 32 Processes. The plots were drawn based on time and number of processes for each size of the rank is 1,4,8 and 16 MB. And evaluated Bandwidth for each 1, 4,8 and 16 MB data based on overall time.

1. 1 MB

Number of Processes	4	8	16	32
Reading(Seconds)	0.010784	0.024703	0.02928	0.088029
Writing(Seconds)	0.014252	0.056358	0.08907	0.045937
Overall(Seconds)	0.124399	0.188942	0.255095	0.211373

Graph: X-Axis: Number of Processes
Y-Axis: Elapsed time in seconds

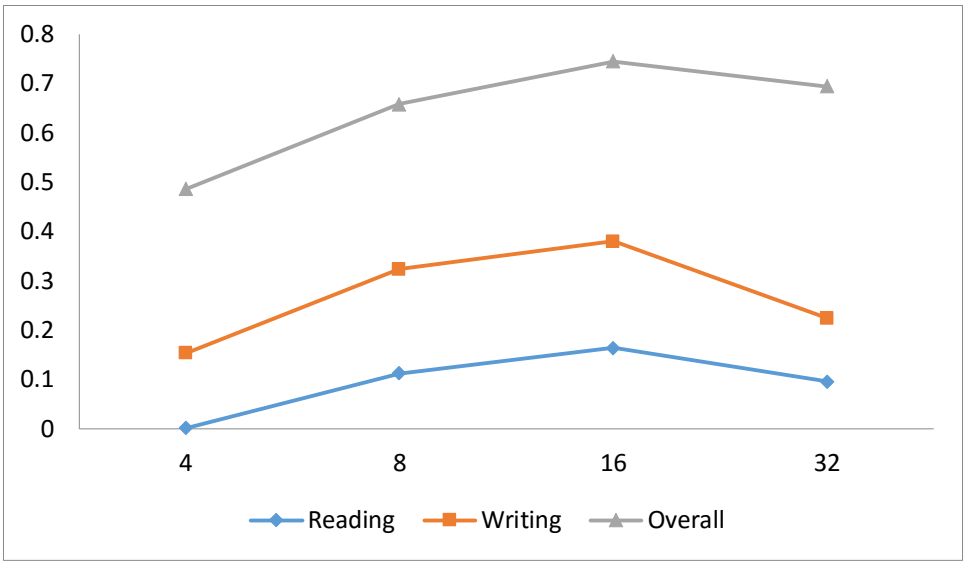


Observation: From the above graph, we can observe that writing data into a file take more time than reading because in writing, data generated randomly for each rank takes more time .For 1 MB for each rank, when multiple clients trying to write data in common file, leads more time as number of clients increases. In reading, it directly open a file, read data into buffer takes less time. In 16 and 32 clients, all the clients are might be waited to read data from a file, elapsed time increases from 16 to 32 clients. We will getting better performance in 4 processes. Overall time is combined opening a file ,writing data inti file ,reading from a file and closing a file takes more time when number of clients increases.

2. 4 MB

Number of Processes	4	8	16	32
Reading(Seconds)	0.002538	0.11338	0.164491	0.09642
Writing(Seconds)	0.151171	0.210455	0.215941	0.128848
Overall(Seconds)	0.332611	0.344895	0.365535	0.469698

Graph: X-Axis: Number of Processes
Y-Axis: Elapsed time in seconds

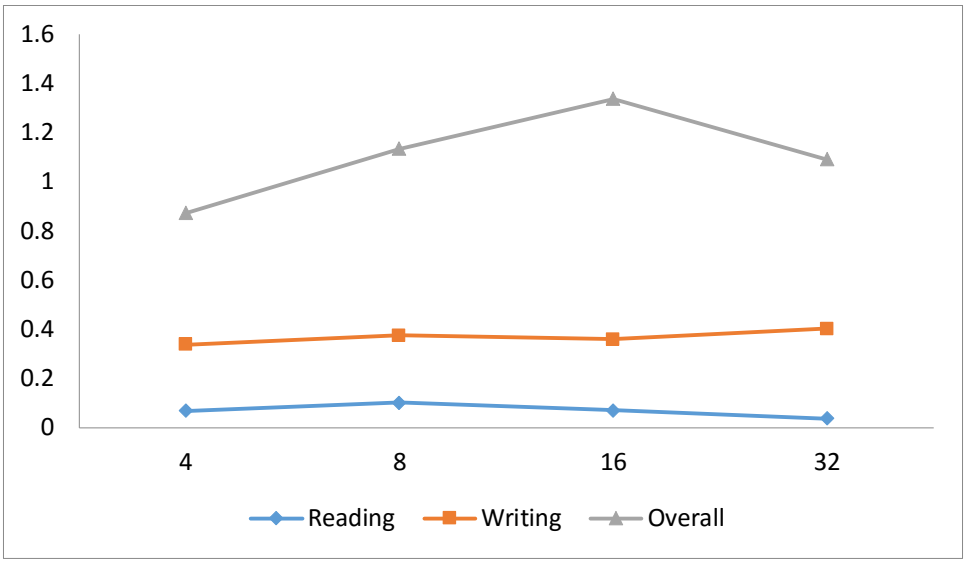


Observation: From the above graph, we can observe that writing data into a file take more time than reading because in writing, data generated randomly for each rank takes more time .For 4 MB for each rank, when multiple clients trying to write data in common file, leads more time as number of clients increases up 16 clients. In 32 clients, writing time takes less compared to 16 processes because of network. In reading, it directly open a file, read data into buffer takes less time. In 32 clients, all the clients are might not be waited as it have good network to read data from a file, elapsed time decreases from 16 to 32 clients. We will getting better performance in 4 processes. Overall time is combined opening a file ,writing data inti file ,reading from a file and closing a file takes more time when number of clients increases. Overall time decreases from 16clients to 32 clients.

3. 8 MB

Number of Processes	4	8	16	32
Reading(Seconds)	0.06733	0.101771	0.069533	0.03716
Writing(Seconds)	0.270111	0.27358	0.290682	0.36521
Overall(Seconds)	0.536745	0.760288	0.979166	0.689233

Graph: X-Axis: Number of Processes
Y-Axis: Elapsed time in seconds

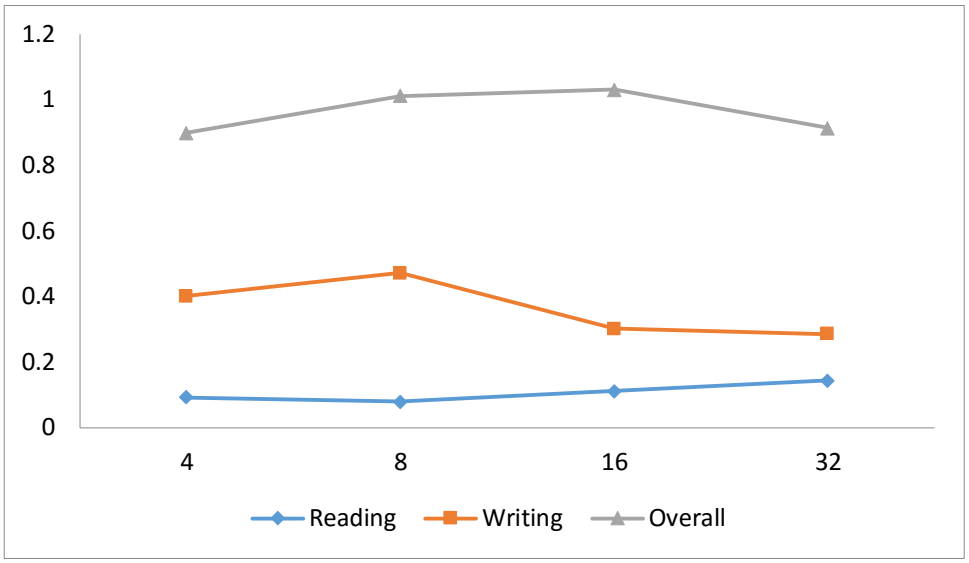


Observation: From the above graph, we can observe that writing data into a file take more time than reading because in writing, data generated randomly for each rank takes more time .For 8 MB for each rank, when multiple clients trying to read and write data in common file, performance is better till 32 clients, almost constant time form 4 processes to 32 clients for both reading and writing. Overall time is combined opening a file, writing data inti file, reading from a file and closing a file takes more time when number of client's increases up 16, then it gives better performance in 32 clients. Overall time increases from 16clients to 32 clients.

4. 16 MB

Number of Processes	4	8	16	32
Reading(Seconds)	0.091878	0.078694	0.112854	0.143901
Writing(Seconds)	0.309532	0.392422	0.18961	0.142298
Overall(Seconds)	0.498406	0.542259	0.730373	0.628222

Graph: X-Axis: Number of Processes
Y-Axis: Elapsed time in seconds

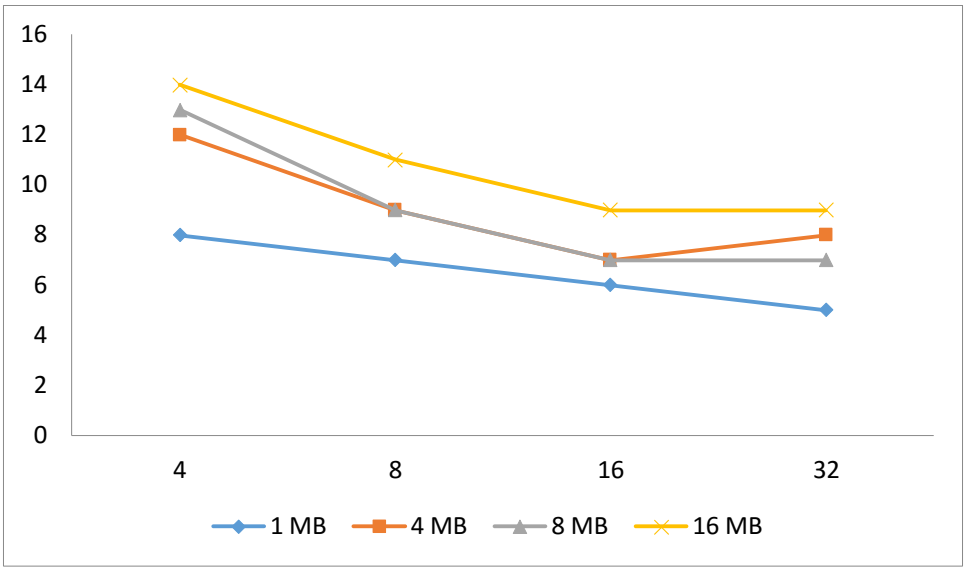


Observation: From the above graph, we can observe that writing data into a file take more time than reading because in writing, data generated randomly for each rank takes more time .For 16 MB for each rank, when multiple clients trying to write data in common file, leads more time as number of clients increases up to 8 clients, later it gives better performance because of fast I/O network. In reading, it directly open a file, read data into buffer takes less time. In 16 and 32 clients, all the clients are might be waited to read data from a file, elapsed time increases from 16 to 32 clients. We will getting better performance in 16 processes for writing whereas reading gives better performance for 4 clients. Overall time is combined opening a file ,writing data inti file ,reading from a file and closing a file takes more time when number of clients increases.

5. Bandwidth(MB/Sec)

Number of Processes	4	8	16	32
1 MB	8	7	6	5
4 MB	12	9	7	8
8 MB	13	10	7	7
16 MB	14	11	9	9

Graph: X-Axis: Number of Processes
Y-Axis: MB/Sec



Observation: From the above graph, we can observe that bandwidth decreases as number of clients increases because I‘ve evaluated bandwidth based on Overall time. Since overall times takes more time for opening a file pointer, generating random data, writing, reading and closing a file takes more time.as overtime increases, bandwidth decreases.(bandwidth is inversely proportional to time). And bandwidth slightly increases from 16 to 32 clients because fast I/O network. As the size of MB increases for each rank, Bandwidth increases.