Utilizing In-Memory Storage for MPI-IO

Julian Kunkel, Eugen Betke

Deutsches Klimarechenzentrum (DKRZ)

Abstract

In contrast to disk or flash based storage solutions, throughput and latency of in-memory storage promises to be close to the best performance. Kove®'s XPD® offers pooled memory for cluster systems. However, the system does not expose access methods to treat the memory like a traditional parallel file system that offers POSIX or MPI-IO semantics.

Contributions of this poster are:

1. Implementation of an MPI-IO (wrapper) driver for the XPD
2. Thorough performance evaluation of the XPD using IOR (MPIO)

This MPI independent file driver enables high-level I/O libraries (HDF5, NetCDF) to utilize the XPD's pooled memory.

Approach

The developed MPI-IO file driver[a] is selectable at runtime via LD_PRELOAD. It checks the file name for the prefix "xpd:" and routes the accesses otherwise to the underlying MPI. Important MPI-IO functions for HDF5 and IOR are implemented. During the MPI_open/close the Infiniband connections to the XPD's are established/destroyed.

IOR is used for benchmarking performance and barriers between the phases (open, write, read, close) are used to synchronize the processes.

The performance analysis varies the parameters:

- Access granularity:
  16 KiB, 100 KByte[b], 1 MiB, 10 MiB
- Processes-per-node (PPN): 1 to 12
- Nodes: 1 to 98
- Connections: 1 to 14
- Access pattern: sequential and random[c]
- File size: 20 GiB per connection [d]

Performance metrics:

M1. Time for open/close (used to test scalability of the connections)
M2. Throughput read/write reported by IOR
M3. Throughput read/write (computed based on the time for the read/write phase)

Each configuration is run at least three times. Since the throughput reported by IOR includes overhead of open/close and initially that times turned out to be significant, those aspects have been investigated separately.

A subset of measurements is run on the Lustre of DKRZ's supercomputer Mistral.

[a] `http://github.com/JulianKunkel/XPD-MPIIO-driver`
[b] Base 10 has been used on purpose as this leads to unaligned access for file systems, i.e., 100 KByte = 10^5 Bytes. All other cases are base 2.
[c] As expected for a DRAM based storage system, they did not show significant differences. Thus, the poster only contains values for random I/O.
[d] The capacity of the XPD is shared among all users.

Overview

Performance of all (7500) conducted runs:

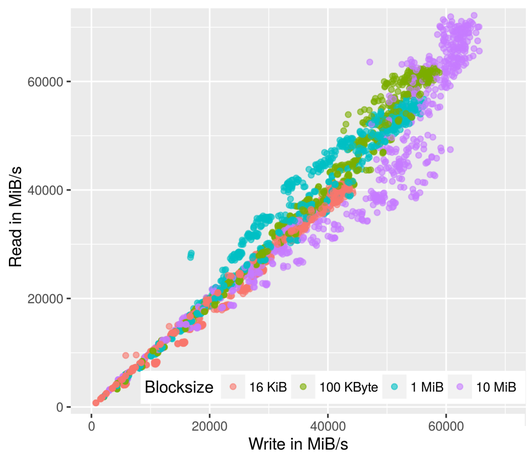


Fig. 1: Observed throughput computed based on the read/write phase (M3.)

Observations:

- Read/write behaves symmetric
  Pearson correlation coef.: 0.969
- Open/close overhead reduces throughput of $M2 \sim= 0.9 \cdot M3$
- Best performance:
  - 65,600 MiB/s (write)
  - 72.200 MiB/s (read)

  $\Rightarrow$ 5155 MiB/s per IB FDR link (read)

Increasing Connections

Understanding the performance behavior when increasing the number of connections reveals scale-out behavior. The test uses always 14 client nodes. Results for reads are shown, write is similar.

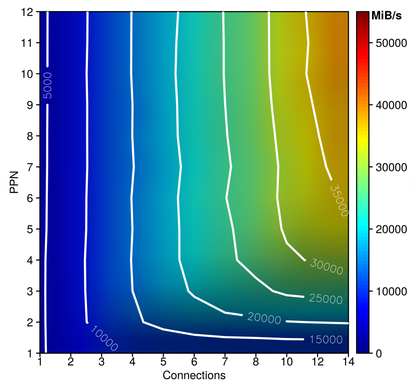


Fig. 2: Granularity: 16 KiB

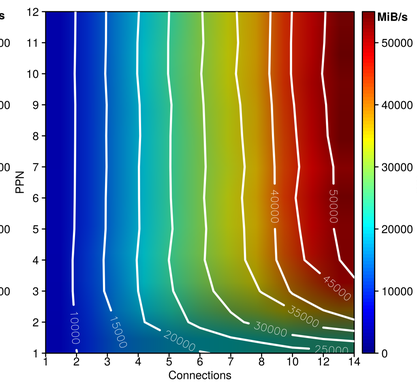


Fig. 3: Granularity: 100 KB

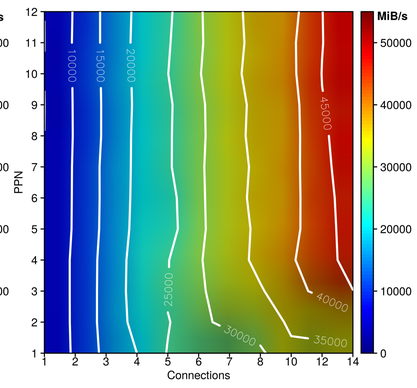
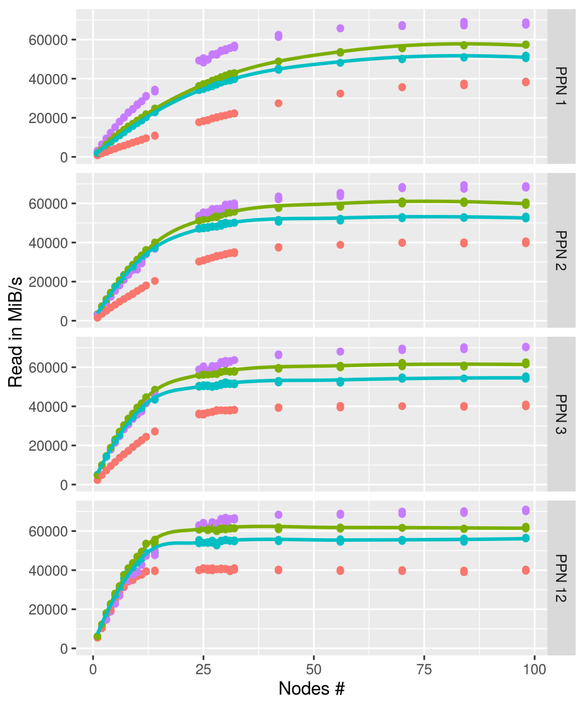


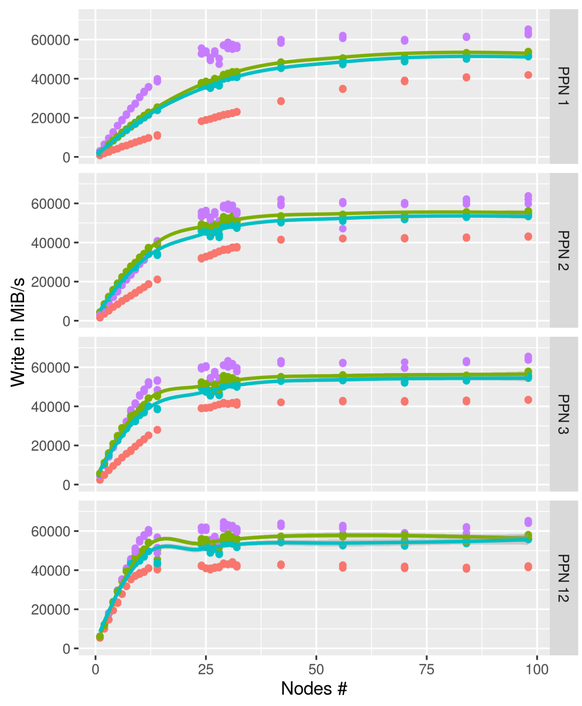
Fig. 4: Granularity: 10 MiB

Scaling Behavior

Results for measuring performance varying blocksize (10 MiB, 1 MiB, 100 KB, 16 KiB), nodes and PPN.



(a) Read



(b) Write

The graphs contain fitting curves for 1 MiB and 100 KB. Graphs for PPN=5 and PPN=8 look similar.

Observations:

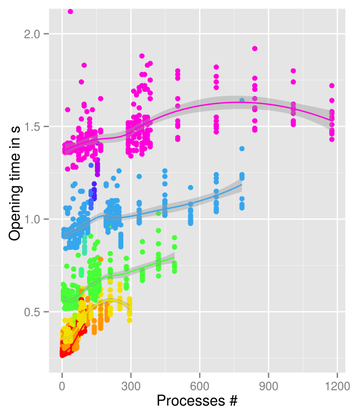
- With small block sizes, I/O becomes limited by network latency and CPU speed
- An increase of PPN or client nodes improves overall throughput until hardware is saturated
- Robust scaling behavior, with PPN=12 and 14 client nodes, peak performance is achieved
- Regardless of PPN, with 14 nodes (== 14 IB links), the 14 server links are at $> 50\%$ saturated

Opening/Closing of Files

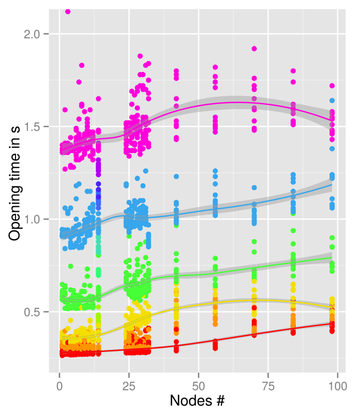
The connection to the XPDs is established during MPI_File_open() and disconnection during close. This allows to define the XPD volumes and striping factor to be used at open time.

Fig. 5: Results for 14 connections including fitting curves for PPN={1,2,3,5,8}

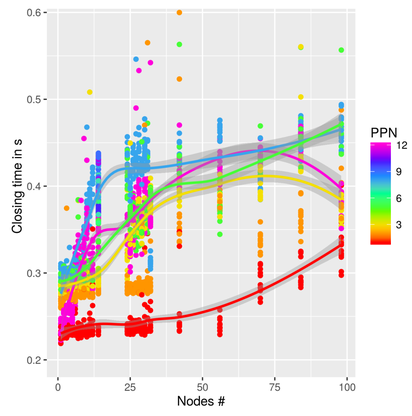
(a) Open vs. process count



(b) Open vs. node count



(c) Close vs. node count



A few (random) outliers in the close figure are up to 1.1s, they are purged from the graph.

Estimating scaling of connection times using linear models

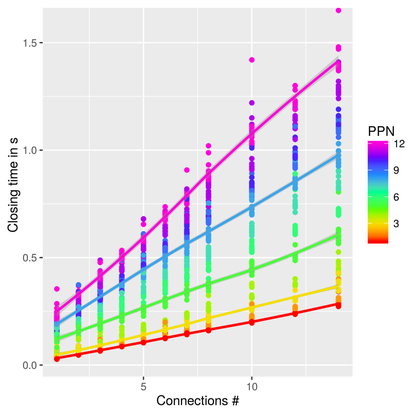
To estimate the behavior for large systems, a linear model is built: $t(PPN, nodes) = I + c_{ppn} \cdot PPN + c_{nodes} \cdot nodes$
Where I and all coefficients starting with c are determined.

Type	Intercept	c_{ppn}	c_{nodes}	R^2
Open	0.0705	0.1090	0.00235	0.966
Close	0.2490	0.0076	0.00166	0.406

To evaluate the impact for the number of connections (the tests have been run on only 14 nodes), another model is built. Since for each PPN the connections are established, they must be multiplied:
$t = I + c_{ppn} \cdot PPN + c_{conn} \cdot conn + c_{PPN:conn} \cdot PPN \cdot conn$

Type	Intercept	c_{ppn}	c_{conn}	$c_{PPN:conn}$	R^2
Open	-0.0031	0.0133	0.00586	0.00695	0.98
Close	-0.012	0.0128	0.01960	0.00007	0.90

Fig. 6: Open times for varying connections on 14 nodes



Given that 14 connections are used on a system with 10.000 nodes, 24 PPN, the prediction of the model for open would be 26s. This would allow true burst-buffer scenarios (i.e., the XPD as a write-behind cache) or cases in which only a subset of data regions is needed.

Using the second model and 10000 connections to XPDs it would take about 1700s and 214s to open and close a single file, respectively.

System Description

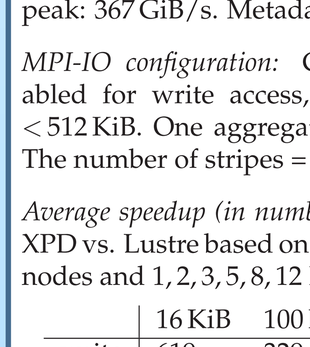
The test system is Cooley, the visualization cluster of Mira on ALCF:

- 126 compute nodes equipped with two 2.4 GHz Haswell E5-2620
- FDR Infiniband
- Kove® XPD® L3
- 3 XPDs with 6+4+4=14 FDR connections

Performance Variability

A low performance variability is important for tightly coupled applications.

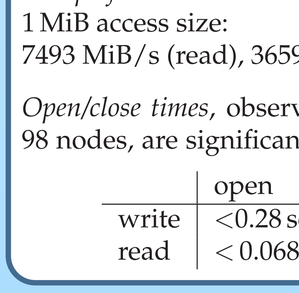
Fig. 7: Density of the variability range across all conducted experiments (span across three repeats each).



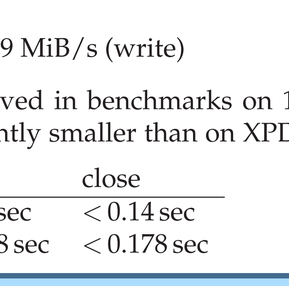
- Mean(read) = 1.23%
- Mean(write) = 1.78%
- 99% of all measurements vary $< 10\%$
- 14 (0.6%) are $> 10\%$

Fig. 8: Boxplots for 100 repeats on 14 nodes

(a) Read



(b) Write



Comparison to Lustre

DKRZ's phase2 Lustre system consisting of 68 OSS and 33 PByte of storage capacity. Theoretical peak: 367 GiB/s. Metadata: 210.000 Ops/s

MPI-IO configuration: Collective I/O was enabled for write access, only for granularities < 512 KiB. One aggregator per node was used. The number of stripes = 2 · number of nodes.

Average speedup (in number of times) of using the XPD vs. Lustre based on random I/O of 2, 4, 8, 14 nodes and 1, 2, 3, 5, 8, 12 PPNs:

	16 KiB	100 KB	1 MiB	10 MiB
write	619	329	10	10
read	887	79	19	15

Best performance is achieved on 14 nodes, 5 PPN, 1 MiB access size:
7493 MiB/s (read), 3659 MiB/s (write)

Open/close times, observed in benchmarks on 1-98 nodes, are significantly smaller than on XPD:

	open	close
write	<0.28 sec	< 0.14 sec
read	< 0.068 sec	< 0.178 sec

Observations & Conclusions

- Read performance ≈ write performance
- Random I/O ≈ sequential I/O
- Highly scalable in terms of
  - client nodes
  - number of connections
- Bottlenecks are CPU and network latency
  - in particular for small blocksizes
- Pre-registered memory can boost I/O
  - used for 100 KByte (faster)
  - not used for 1 MiB (slower)
- Excellent access time variability
  - read: $\leq 2.5\%$; write: $\leq 5\%$
- File opening/closing times
  - must be investigated for big systems
  - can be improved (according to Kove)

In the future, we aim to optimize buffer settings, work towards a full MPI-IO compatible wrapper library and deal with data migration between XPD and file system.

Acknowledgements

Thanks to Kove for their support and discussion. Thanks to our sponsor William E. Allcock for providing access and feedback. This research used resources of the Argonne Leadership Computing Facility, which is a DOE Office of Science User Facility supported under Contract DE-AC02-06CH11357.