State of the Art I/O Tools

EPCC

February 28, 2018

Elsa Gonsiorowski

LLNL-PRES-746909

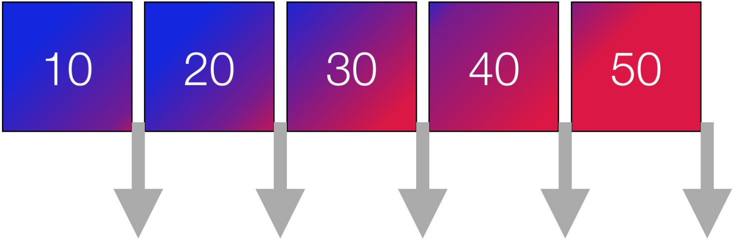
This work was performed under the auspices of the U.S. Department of Energy by Lawrence Livermore National Laboratory under contract DE-AC52-07NA27344. Lawrence Livermore National Security, LLC

Lawrence Livermore National Laboratory

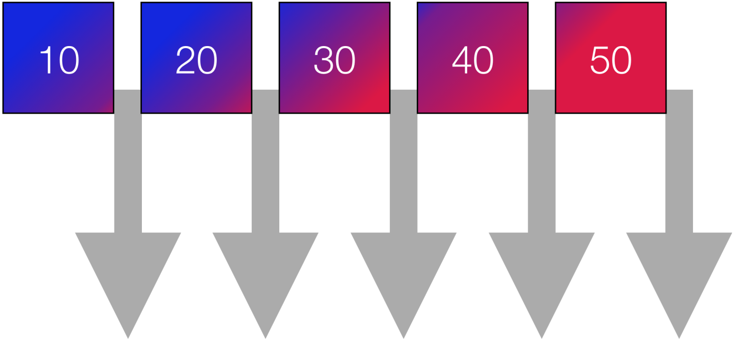
Outline

- Motivating Example
  - Questions from Applications
- Measuring I/O Performance
  - MACSio
- The I/O Stack
- Burst Buffer Technologies
  - SCR and Performance Portability
- Additional Projects
  - IO-500

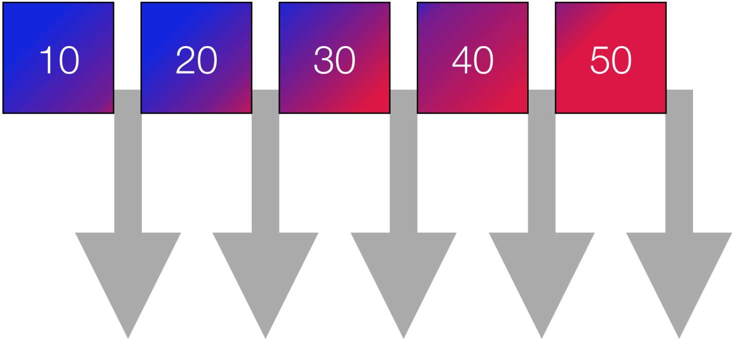
Simulation Output



Simulation Output



Simulation Output



I/O Performance hasn't changed

Motivation

As computation performance increases
I/O must be re-evaluated.

Questions from Applications

1. Where do we fall in the I/O envelope?
2. Parameters to achieve best performance?
3. How do we best use new storage tiers?

Where do we fall in the I/O Envelope?

Given:

- Peak system I/O performance
- Current application performance
- I/O pattern or trace
- … other details?

Answer:

- Where is the application losing performance?
- What will gains can be made?

Where do we fall in the I/O Envelope?

Current Examples

- Use IOR and mdtest to measure peak system performance
- I/O Specific proxy application
- Lots of work

Where do we fall in the I/O Envelope?

Unposed Questions

- What is the point of this I/O?
- Could this use-case be achieved in a more efficient way?
- How do we enable in-situ or co-situ processes?

High-level questions

Parameters to achieve best performance?

Given:

- Tuning of peak performing benchmark
- Current application I/O

Answer:

- What file system settings need to be tuned?
- Is metadata a bottleneck / file locking?

Parameters to achieve best performance?

Current Examples

- None.
- Validation of simulation models with counters, no analysis of real applications

Parameters to achieve best performance?

Unposed Questions

- Can any of this be detected at a lower level?
- Automatic tuning of the file system during a workload
- How can this drive future procurements?

Lower level and inter-level questions

How do we best use new Storage Tiers?

Given:

- Scientific need
- System limitations

Answer:

- Which I/O patterns perform best
- Resiliency models

How do we best use new Storage Tiers?

Current Examples

- Defensive I/O Assumption
  - Optimal checkpoint interval
  - SCR with system-specific configuration
- Lossy compressions
  - HDF5 ZFP Compression

How do we best use new Storage Tiers?

Unposed Questions

- Interactions between resource schedulers and application
  - pre-stage / post-stage
  - dynamic job allocation resources
- What is the scientific need? How much precision is needed?
- Work flows to manage data movement

Questions requiring full-stack knowledge

Measuring I/O Performance

- Benchmarking
- Profiling
- Proxy Applications

Benchmarking

- IOR
- mdtest
- benchio
- IO_Bench
- MPI Tile IO
- b_eff_io
- SPIOBENCH
- iozone
- MADbench2

Mainly testing POSIX interface, with some MPI-IO.

Profiling

- Darshan
- Vampir

Proxy Applications

- MACSio
- HACC_IO / GenericIO

High Level-of-Abstraction

- Application-level I/O
- Utilize multiple layers of I/O middlewares
- Representative mesh data

Tunable I/O Patterns

- File-per-process
- Single shared file
- Middle ground: M files to N processes

Plug-in based Architecture

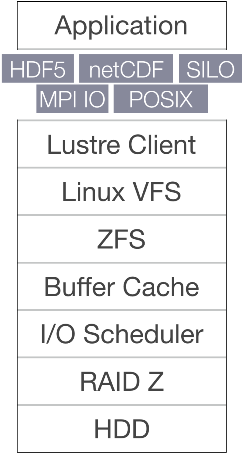
- HDF5
- netCDF
- SILO
- TyphonIO
- ADIOS *coming soon*

The I/O Stack

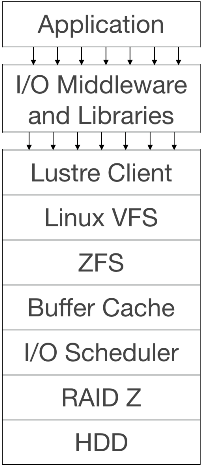
Application
I/O Middleware and Libraries
Lustre Client
Linux VFS
ZFS
Buffer Cache
I/O Scheduler
RAID Z
HDD

Courtesy of John Bent

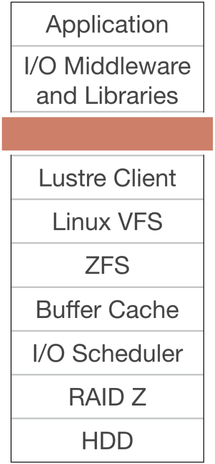
The I/O Stack


Application
HDF5
netCDF
SILO
MPI IO
POSIX
Lustre Client
Linux VFS
ZFS
Buffer Cache
I/O Scheduler
RAID Z
HDD

The I/O Stack


Application
I/O Middleware and Libraries
Lustre Client
Linux VFS
ZFS
Buffer Cache
I/O Scheduler
RAID Z
HDD

The I/O Stack


Application
I/O Middleware and Libraries
Lustre Client
Linux VFS
ZFS
Buffer Cache
I/O Scheduler
RAID Z
HDD

Burst Buffer Technologies

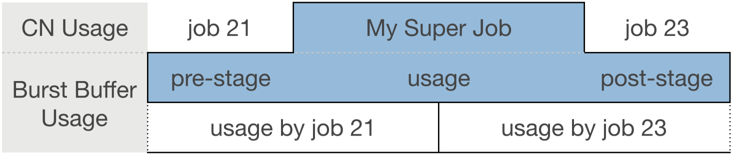
Type	Technology	Location
Node Local	IBM BBAPI	LLNL (Sierra)
Machine Global	Cray Datawarp	LANL (Trinity)

Burst Buffer Technologies

Type	Technology	Location
Node Local	IBM BBAPI	LLNL (Sierra)
Machine Global	Cray Datawarp	LANL (Trinity)

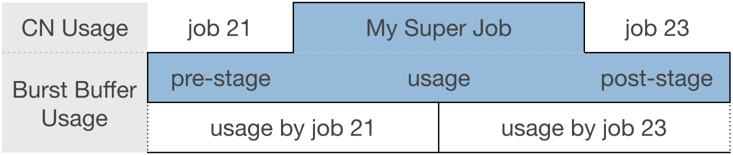
How can an application utilize this layer for I/O workloads?

Burst Buffers Use Case



- Relies on integration with resource scheduler
- Different for machine-global vs. node-local storage
- Does not address inter-job data movement

Burst Buffers Use Case



Perfect for Checkpoint/Restart

SCR Goal

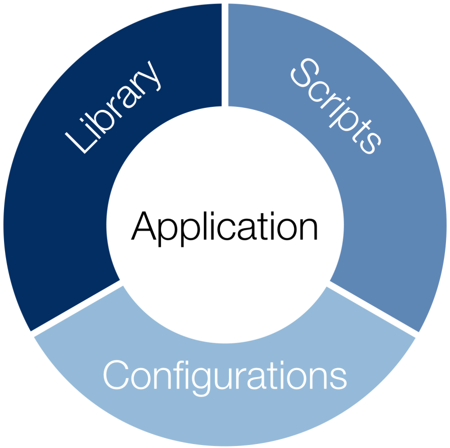
Enable checkpointing applications to take advantage of system storage hierarchies

SCR Goal

Enable checkpointing applications to take advantage of system storage hierarchies

- Efficient file movement between storage layers
- Data redundancy operations

SCR Components



SCR Component: Backend Library

- Redirect application files
- Synchronous & asynchronous flush operations
  - Hardware specific capabilities
- Data redundancy
- Support for both checkpoint & output data

SCR Component: Frontend Scripts

- On Startup Locate most recent checkpoint and fetch for restart
- Within Allocation Detect application crash or system failures and trigger restart
- During Execution Manage datasets
- Resource Scheduler Integration Pre- and post-stage data movement

SCR Component: Configurations

- Define the levels of the hierarchy
- Define modes/groups of failure
- Define checkpointing and data residency needs

SCR Component: Configurations

- Define the levels of the hierarchy
- Define modes/groups of failure
- Define checkpointing and data residency needs

Machine Portability

VELOC

- Combining two codes: FTI and SCR
- FTI: variable-based checkpointing scheme
- Will support existing FTI and SCR applications

ECP
EXASCALE COMPUTING PROJECT

UnifyCR

- User-level file system
- Shared namespace across distributed burst buffers
- I/O interception layer

MPI File Utils

Use parallel processes to perform file operations

- Executed within a job allocation
- `dbcast`: broadcast from PFS to node-local storage
- `dcp`: multiple file copy in parallel
- `drm`: delete files in parallel
- *many more*

`https://github.com/hpc/mpifileutils`

IO-500

Site		Score	BW (GiB/s)	MD (KIOP/s)
JCAHPC	JPN	101.48	471.25	21.85
Kaust	SAU	70.90	151.53	33.17
Kaust	SAU	41.00	54.17	31.03
JSC	DEU	35.77	14.24	89.83
DKRZ	DEU	32.15	22.77	45.39

vi4io.org, February 2018.

Lawrence Livermore National Laboratory