Enabling Portable I/O Analysis of Commercially Sensitive HPC Applications Through Workload Replication

James Dickson, Steven Wright, Stephen Jarvis - University of Warwick

Satheesh Maheswaran, Andy Herdman, Duncan Harris - UK Atomic Weapons Establishment

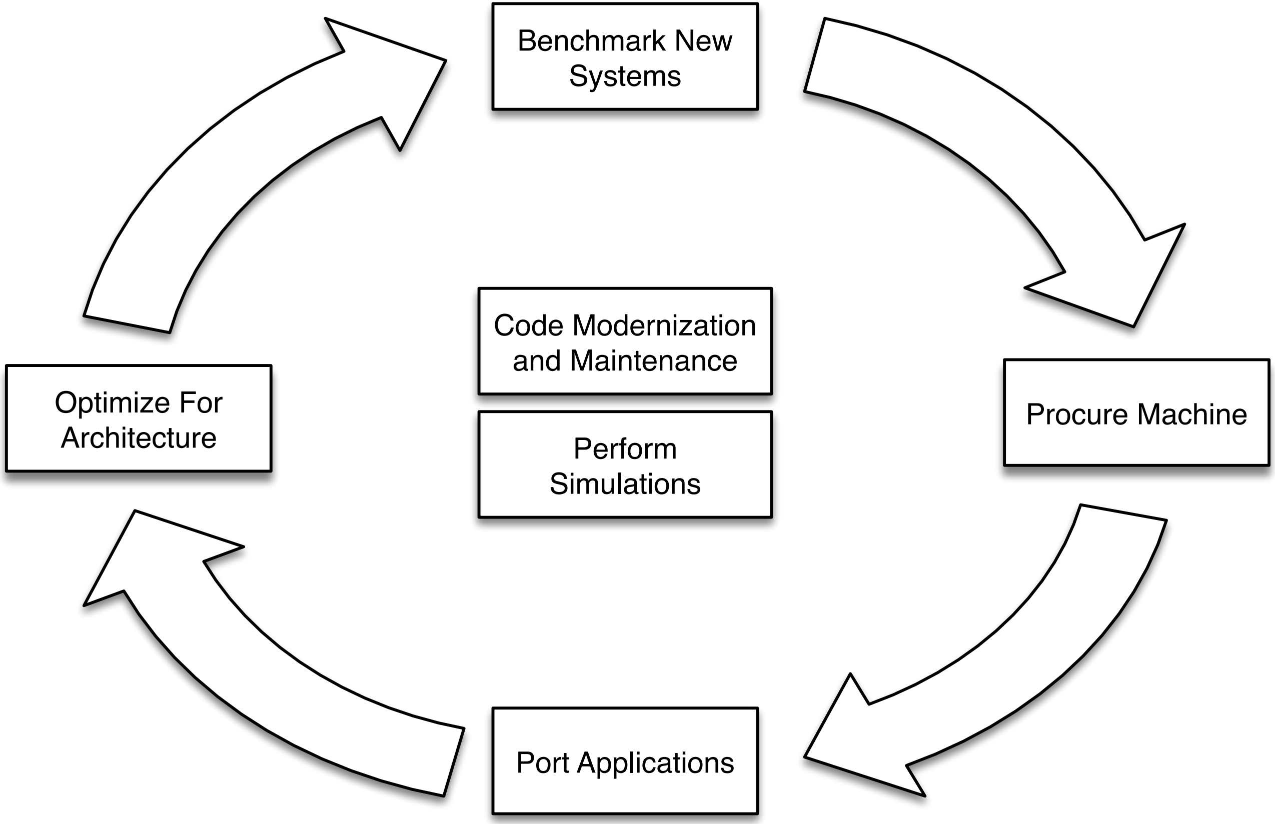
Mark C. Miller - Lawrence Livermore National Laboratory

Outline

- Motivation
- Background
- Implementation
  - MACSio development
  - Workload design characteristics
- Case Study
  - Target application I/O patterns
  - Data replication accuracy
  - Workload performance analysis
- Conclusion

Motivation

- The I/O patterns used in applications are varied and often left up to the user
  - Checkpoint every x time steps… but where does the x come from?
- How can we get a handle on the I/O performance in applications with such variability?
- We can continue to develop file systems, but if applications are not doing efficient I/O then we are not seeing anywhere near the theoretical peak performance
- Should we tune file system to applications or applications to file system?
- Would an alternative high level library work better for a particular suite of applications?
- What sort of I/O performance are we going to get from the next system?
- How can we benchmark a real I/O workload from a large sensitive code?



Benchmark New Systems

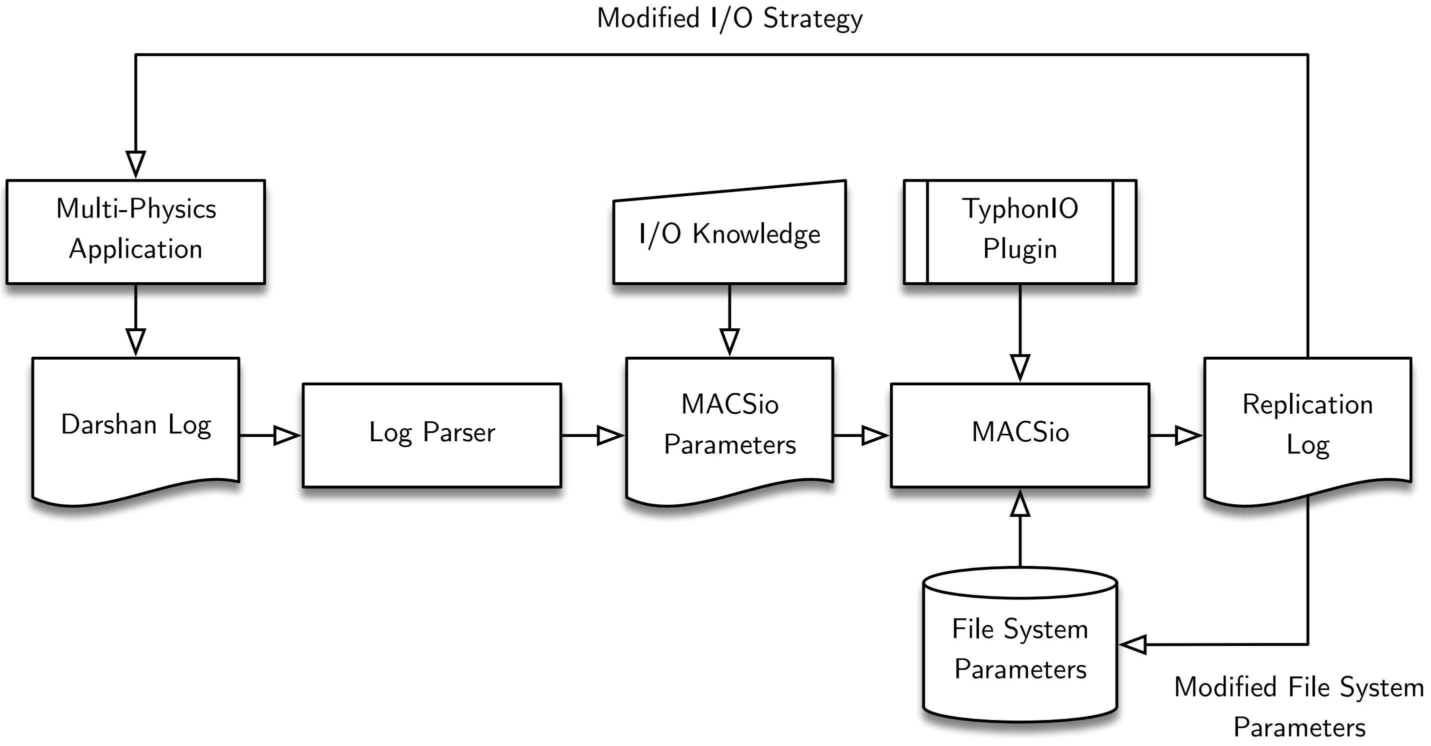
Procure Machine

Port Applications

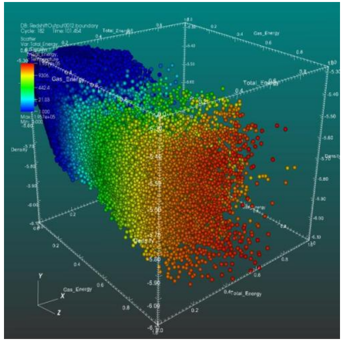
Optimize For Architecture

Code Modernization and Maintenance

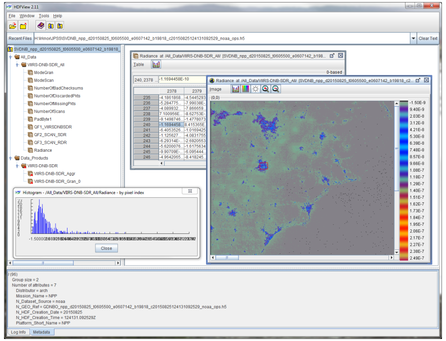
Perform Simulations


Modified I/O Strategy
Multi-Physics Application
I/O Knowledge
TyphonIO Plugin
Darshan Log
Log Parser
MACSio Parameters
MACSio
Replication Log
File System Parameters
Modified File System Parameters

I/O Purpose



Visualization



H:\lrknox\JPSS\SVDNB_npp_d20150825_t0605500_e0607142_b19818_c20150825124131092529_noaa_ops.h5

Radiance at /All_Data/VIIRS-DNB-SDR_All/ [SVDNB_npp_d20150825_t0605500_e0607142_b198...

240, 2378 ... -1.1694458E-10

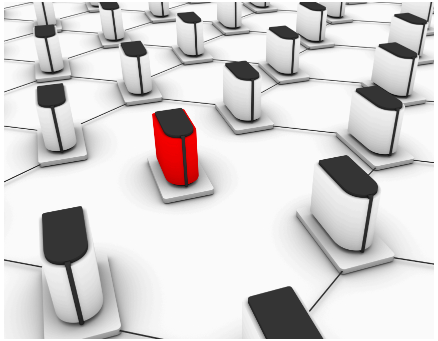
	2378	2379
235	-4.1861868...	-4.5445293
236	-5.284775...	-7.99038E-...
237	-4.089932...	-7.866659...
238	7.100956E...	-8.62753E-...
239	-9.1498746...	-1.4778073
240	-1.1694458...	8.415365E...
241	-6.4053526...	-1.0169425
242	-1.125627...	-4.0831755
243	-6.29314E-...	-2.6920553
244	-5.6200076...	-1.6175634
245	-9.90709E-...	-6.095444...
246	-4.9642065...	-8.418245...

Radiance at /All_Data/VIIRS-DNB-SDR_All/ [SVDNB_npp_d20150825_t0605500_e0607142_b19818_c2...

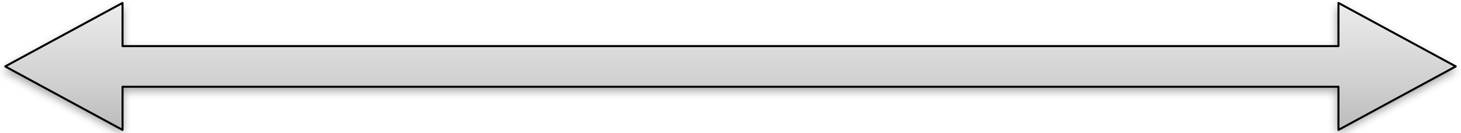
Histogram - /All_Data/VIIRS-DNB-SDR_All/Radiance - by pixel index

```
/ (96)
  Group size = 2
  Number of attributes = 7
    Distributor = arch
    Mission_Name = NPP
    N_Dataset_Source = noaa
    N_GEO_Ref = GDNBO_npp_d20150825_t0605500_e0607142_b19818_c20150825124131092529_noaa_ops.h5
    N_HDF_Creation_Date = 20150825
    N_HDF_Creation_Time = 124131.092529Z
    Platform_Short_Name = NPP
```

Experimental Results



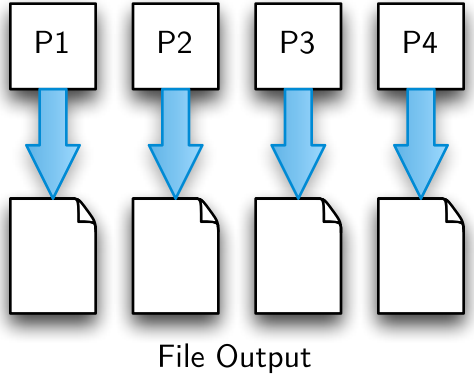
Failure Recovery



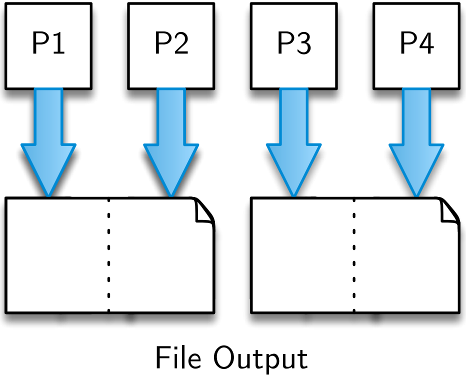
Higher Frequency

More Data

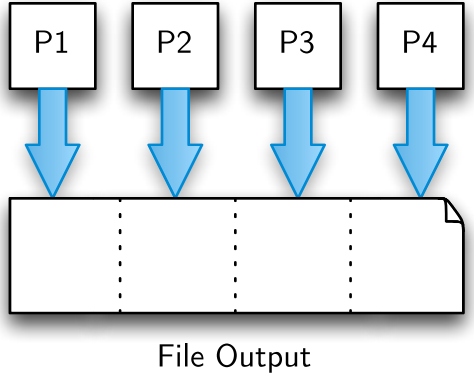
I/O Strategies



N-N



N-M



N-1

I/O Profiling

- Tracing
  - Comprehensive picture of individual I/O operations
  - Collection and storage of this granularity of data does not scale well
- Characterization
  - Don’t store individual calls, instead maintain counters with much lower overhead
  - Less granularity but won’t present the same scaling problems

Production Application

- Multi-Physics code - “M-Phys”
  - Large production application for simulation incorporating multiple physics packages
  - Many important problems can be modeled with this one (extremely complex) application
  - Dataset and control flow dependent on simulation type and user aims, so I/O pattern can have many different forms
  - In general, the two types of file that are written are for checkpointing and visualization

MACSio

- "Multi-purpose Application-Centric, Scalable I/O Proxy Application"
- Two key characteristics:
  - Level of Abstraction: POSIX, MPI-IO, SILO, HDF5 and beyond…
  - Degree of Flexibility: dump type, dataset composition, user defined data objects
- Multi-purpose achieved through plugin based design, if you have a library or interface to work with, write a plugin!

MACSio Development

- Original application behavior has been adapted to more accurately represent real codes
  - Can handle mixed datasets in the same run, so checkpoints and visualization dumps of different composition can be interspersed
  - Changed the main application loop to be time step based making it easier to replicate irregular sequences of file accesses
  - Added a plug in to issue I/O calls using the TyphonIO library to match the target application

Extracting Workload Characteristics

- Data sizes
  - Characterization data reports the size of the data file written but we don't learn any information about the structure of the dataset
  - Need to translate the known file size back to dataset parameters in MACSio

$$\boxed{\begin{aligned}\text{FileSize} = \text{Processors}(\text{PartSize}(\alpha \cdot \text{Variables} + \beta \\ + \gamma \cdot \text{Variables} + \delta) + \psi \cdot \text{Variables} + \eta\end{aligned}}$$

  - Constants α, β, γ, δ, ψ, η are derived empirically from a dataset composition scaling study

Extracting Workload Characteristics

- Simulating dataset growth
  - Use a growth factor sequence based off of the difference between consecutive checkpoints
- Matching execution pattern
  - Ordering of file accesses and spacing extracted from begin/end timestamps

Extracting Workload Characteristics

- Visualization Scheme
  - Data is written to a single plot file, with new states appended to the end on each dump (followed by flush to file rather than a file close)
  - Because of the limited data reported by Darshan, we only know the total size of this file at the end of the run, so have to map the dataset growth pattern learned from checkpoints

Extracting Workload Characteristics

- Visualization Scheme

$$VisTotal = Vis_0 + Vis_1 + \ldots + Vis_n$$

$$Vis_{n+1} = F_n \cdot Vis_n$$

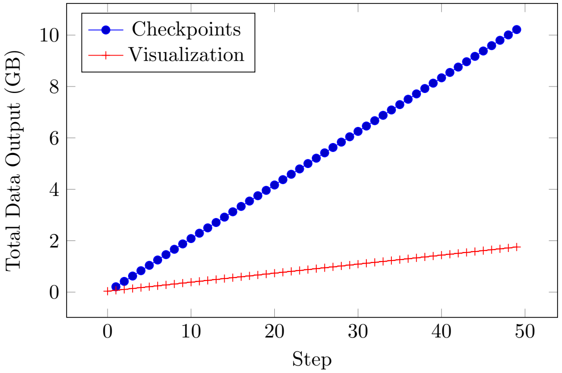
$$Vis_0 = \frac{VisTotal}{1 + F_0(1 + F_1(\ldots(1 + F_n)))}$$

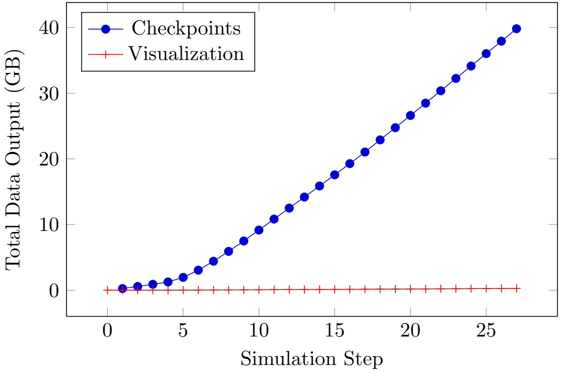
Replication Case Study

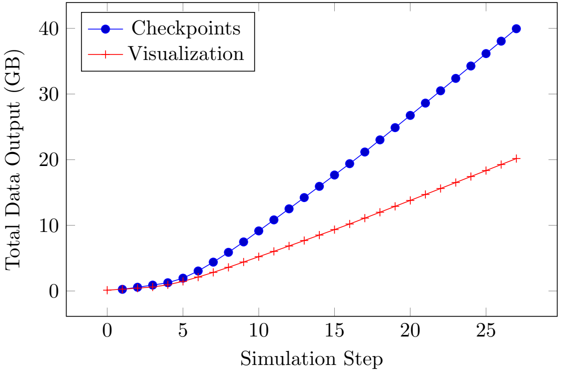
- Focus on five different production problems that are simulated with the M-Phys application, each using different physics packages

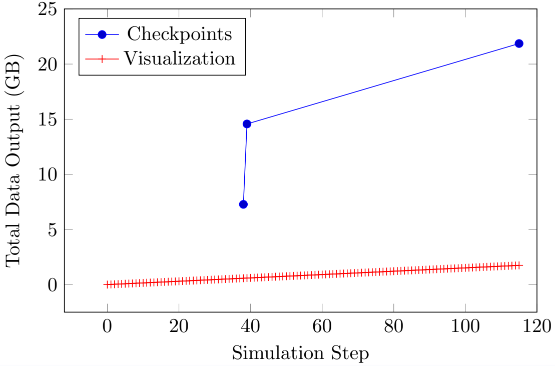
	A	B	C	D	E
Checkpoints	49	27	27	3	3
Checkpoint Sizes	213 MB	259 MB - 1.9 GB	259 MB - 1.9 GB	7.3 GB	7.7 GB
Visualization States	50	28	28	116	88
Total Visualization Output	1.75 GB	258 MB	20.2 GB	1.8 GB	1.4 GB
Number of Ranks	16	80	80	80	80
Original Runtime	119 Minutes	131 Minutes	131 Minutes	20.5 Hours	21 Hours

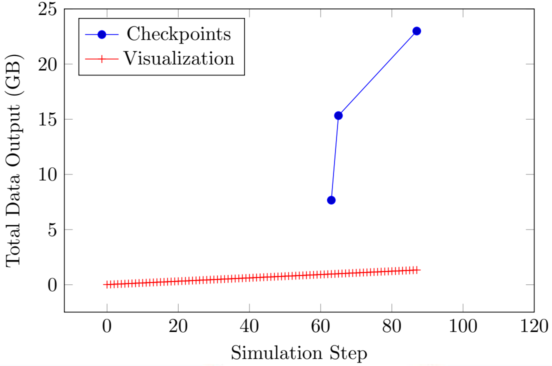
Simulation Patterns











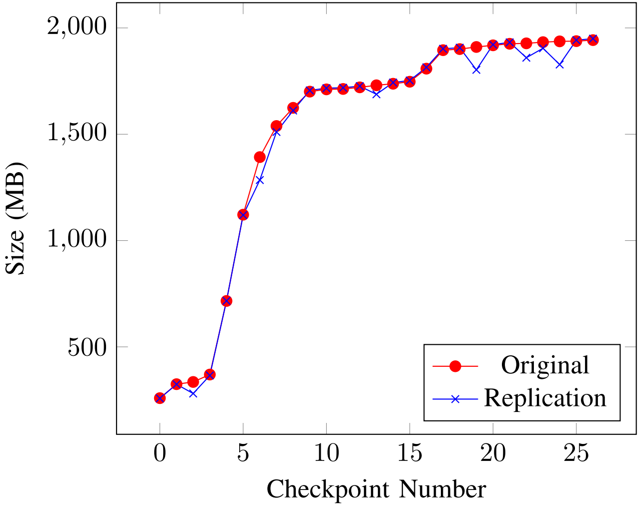
Experimental Platforms

	Titan	Archer	Cab	Taurus	Tinis
Platform	Cray XK7	Cray XC30	Appro Xtreme-X	Bullx DLC 720	Lenovo NeXtScale nx360
Compute Nodes	18,688	4,920	1,296	1,456	203
Interconnect	Cray Gemini	Cray Aries Dragonfly	InfiniBand QDR	InfiniBand FDR	QLogic TrueScale InfiniBand QDR
Parallel File System	40PB Lustre	1.3PB/1.5PB Lustre	5PB Lustre	2.8PB Lustre	0.5PB GPFS
Storage Targets	1008	48/56	80	96	12

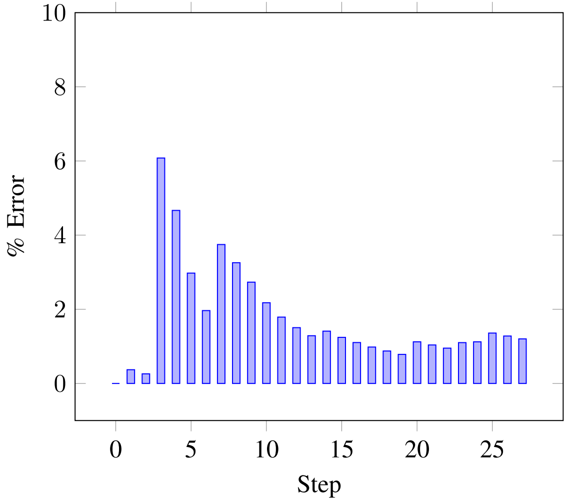
MACSio Replication

	A	B	C	D	E
Mean % error in checkpoint size	0.17%	1.90%	3.05%	1.14%	0.46%
Max % error in checkpoint size	0.43%	16.22%	16.27%	1.16%	0.46%
% error in total Visualization output	8.29%	0.23%	5.72%	1.88%	1.96%
% error in Visualization output per state	0.16%	0.01%	0.20%	0.02%	0.02%

MACSio Replication

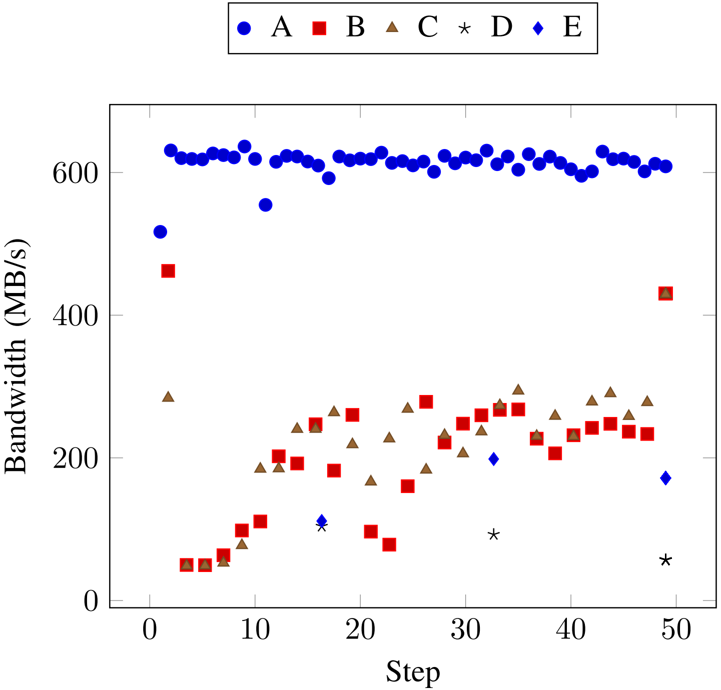


Growth of the dataset during the simulation, demonstrated by the size of each checkpoint



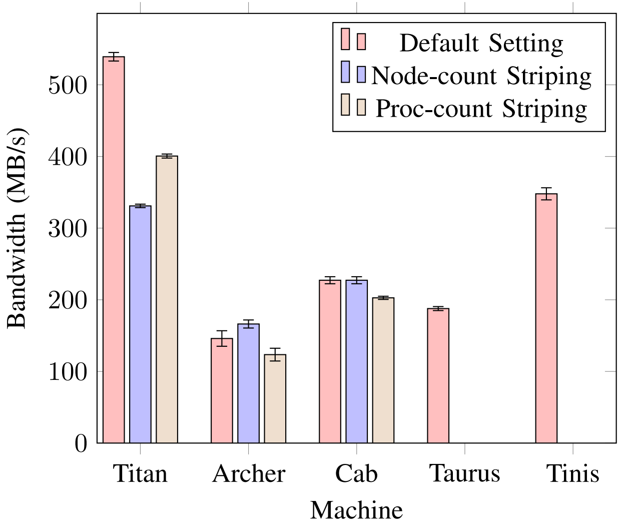
Percentage error between the dataset size of the original and replicated workloads

Reference Performance

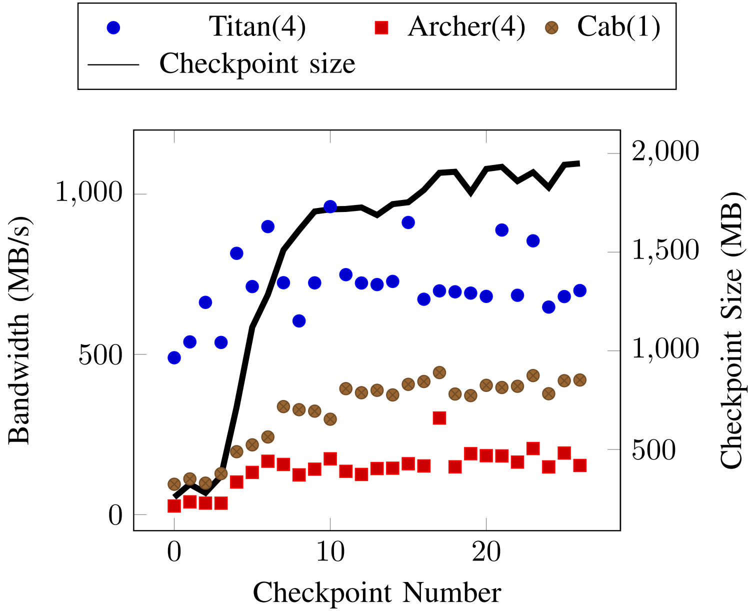


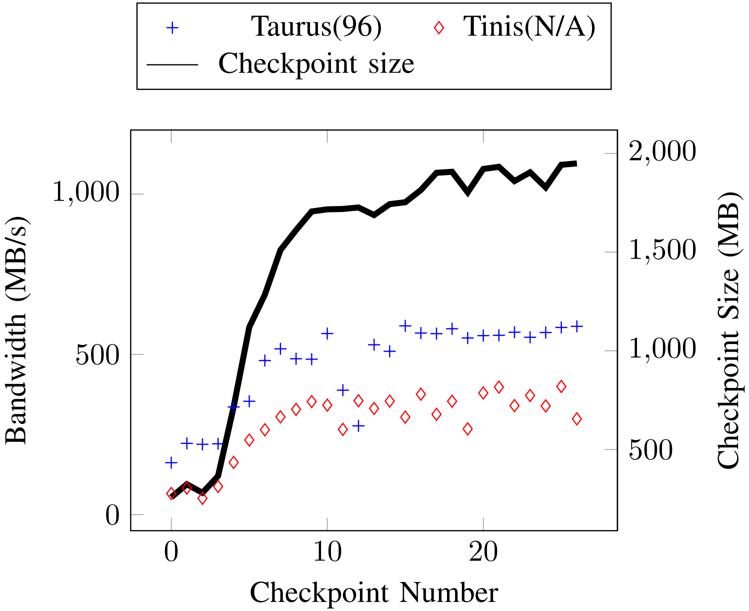
Problem A

- None of the target machines achieve the same average bandwidth for any of the stripe configurations
- Smallest of the problems in terms of scale is most likely to fail in saturating the relevant system components with I/O traffic

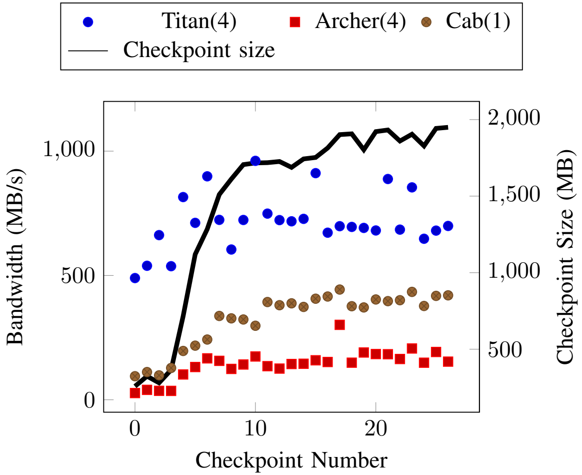


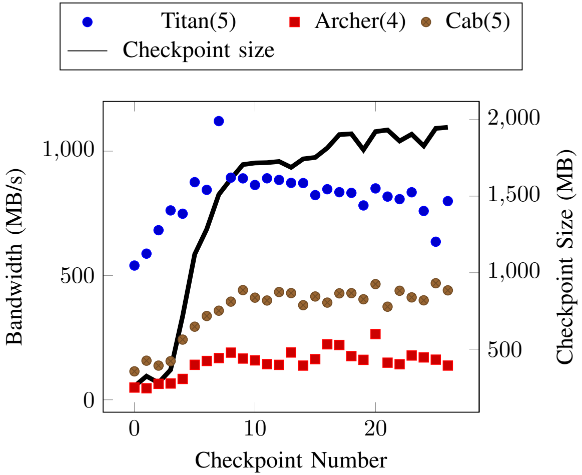
Problems B and C

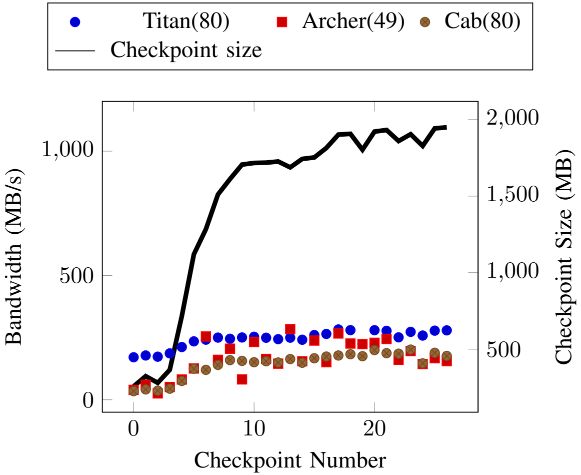




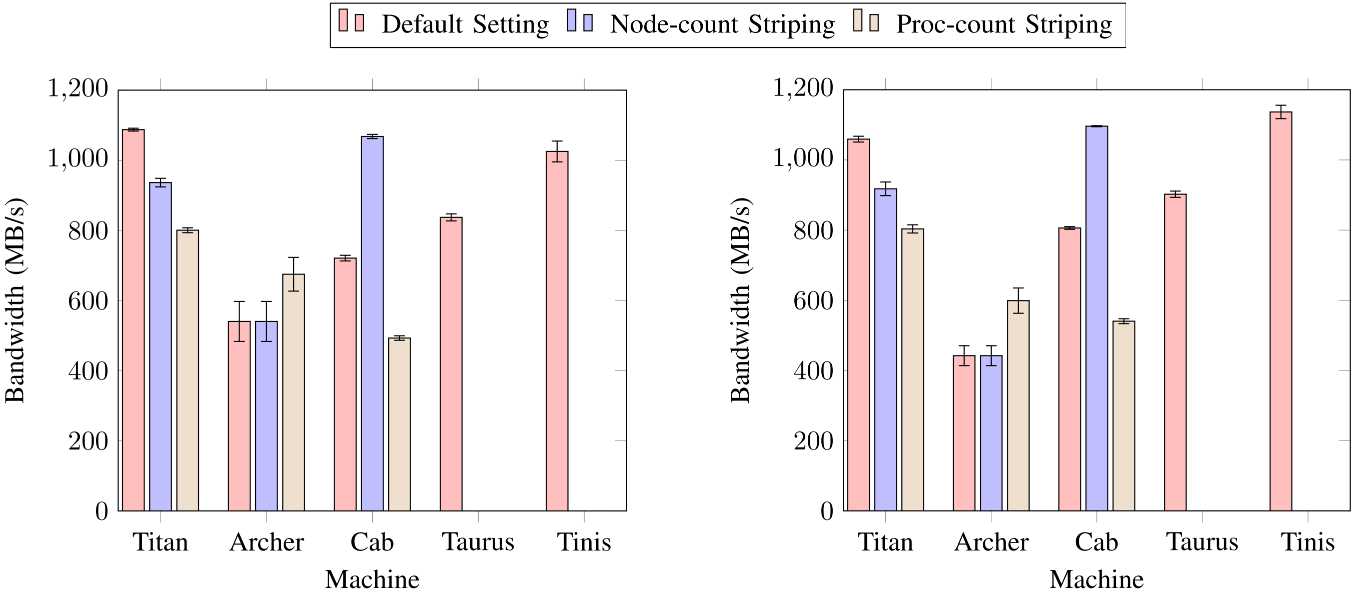
Problems B and C







Problems D and E

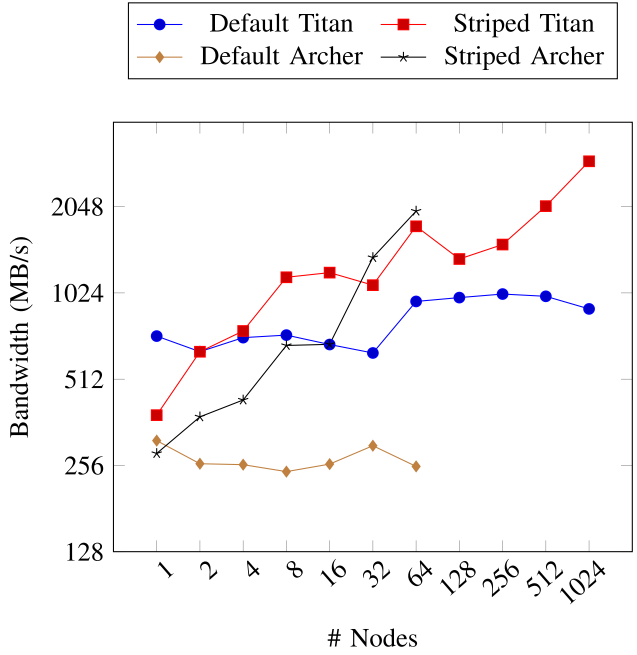


Problem D

Problem E

Predicting Scaling Behavior

- Problem E used due to irregularity of I/O pattern and largest dataset per checkpoint
- Fix the dataset size per rank and scale the number of nodes used



Conclusion

- We have worked towards a semi-automated workflow for the profiling, characterization, replication and analysis of application I/O in a production code
- It is evident that one application is doing a lot of different things with regards to I/O, and being able to establish a technique for identifying this and experimenting in open and portable ways
- We were able to take our replication input parameters and replicate workloads on a range of different platforms, identifying how file systems configuration differences will affect the I/O footprint

Future Work

- Consider the introduction of high fidelity tracing with a focus on avoiding scalability issues
- Incorporate more complex dataset composition into MACSio to increase accuracy and solve the file growth mismatch
- Deploy the different classes of workload against a newly procured system to investigate file system configuration
- Improve tools to allow for integration into future procurement benchmarking exercise

Acknowledgements

- UK Atomic Weapons Establishment Technical Outreach Program
- UK Engineering and Physical Sciences Research Council and EPCC
- Oak Ridge National Laboratory
- Lawrence Livermore National Laboratory
- Technische Universität Dresden
- University of Warwick Centre for Scientific Computing

Thank You

Questions?

J.Dickson@warwick.ac.uk