Uncovering Access, Reuse, and Sharing Characteristics of I/O-Intensive Files on Large-Scale Production HPC Systems

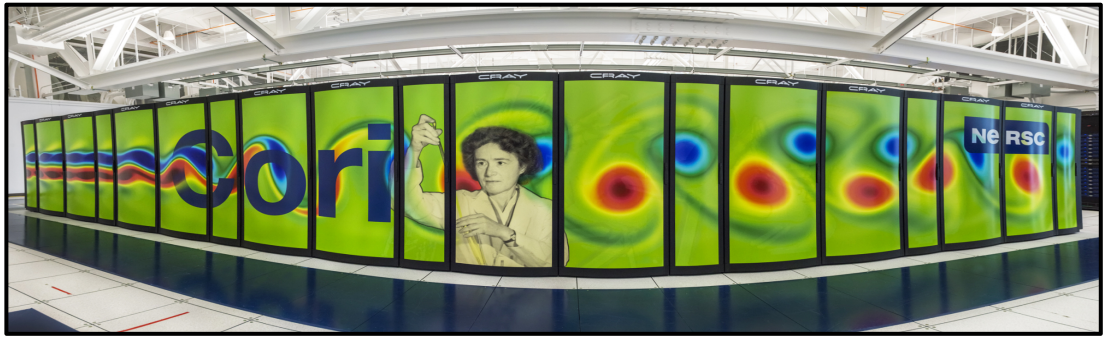
Tirthak Patel
Northeastern University

Suren Byna, Glenn K. Lockwood, Nicholas J. Wright
Lawrence Berkley National Laboratory

Philip Carns, Robert Ross
Argonne National Laboratory

Devesh Tiwari
Northeastern University

HPC systems suffer from severe I/O bottleneck!



What is This Study About?

In-depth characterization and analysis of access, reuse, and sharing characteristics of I/O-intensive files on a top 500 supercomputer.

4-month period, ~84 million logs, ~52 million files, ~9k applications, 650 users, and ~13 PB I/O data (7 PB read and 6 PB write).

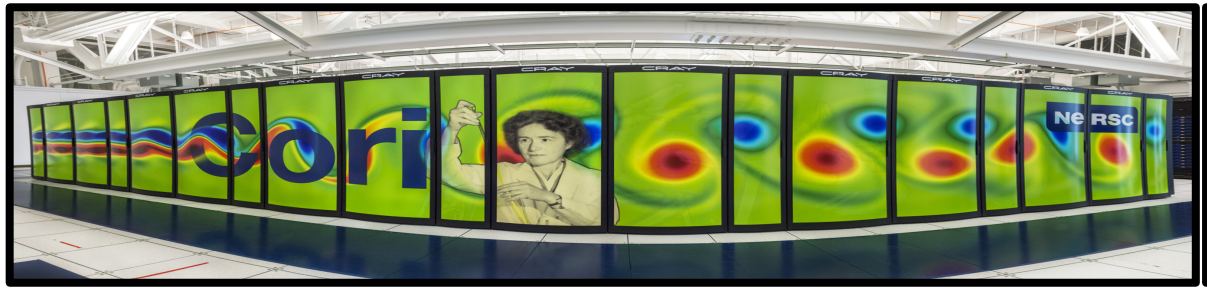
What is This Study About?

In-depth characterization and analysis of access, reuse, and sharing characteristics of I/O-intensive files on a top 500 supercomputer.

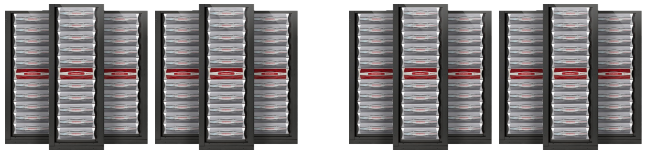
4-month period, ~84 million logs, ~52 million files, ~9k applications, 650 users, and ~13 PB I/O data (7 PB read and 6 PB write).

Examples include: (1) whether HPC files are ready-heavy, write-heavy, or both; (2) sharing of a file across multiple applications; (3) I/O variability within a run across runs!

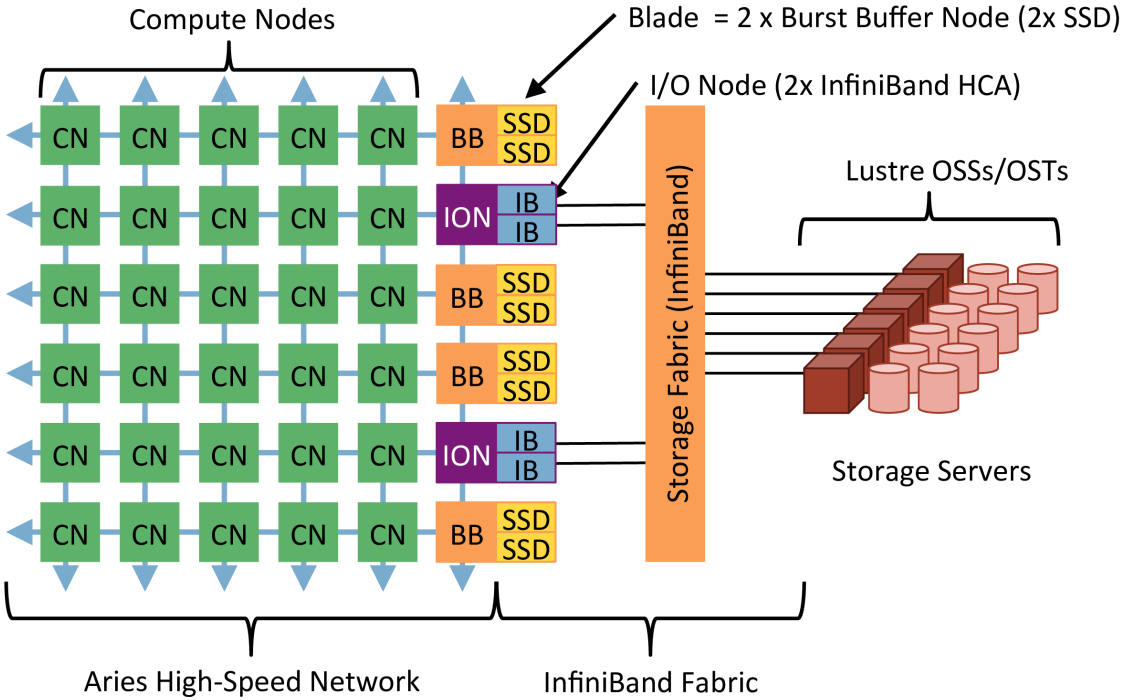
Our Study Platform



Shared Parallel Storage System (scratch space)

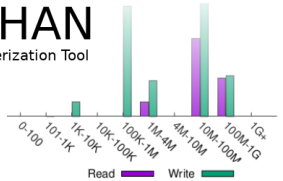


Our Study Platform



2018 R&D 100 WINNER

DARSHAN
HPC I/O Characterization Tool



Application-level I/O tracing using Darshan to collect file I/O access patterns

Take it with a grain of salt! (or three)

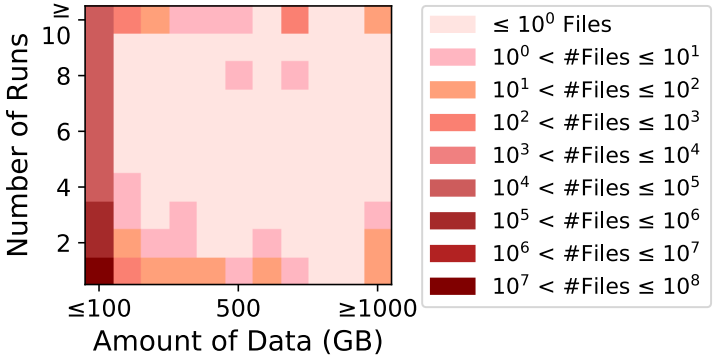
Findings may be true for other systems, but we don't make this claim. Inherently influenced by NERSC policies, filesystem, and workloads.

Some information is not available for correlation. Performing what-if analysis and beyond Darshan's tracing capability is not possible.

Burst buffer effects do not threaten the validity today but needs to be revisited in a few years.

Terms & Definitions

Identifying I/O-intensive files

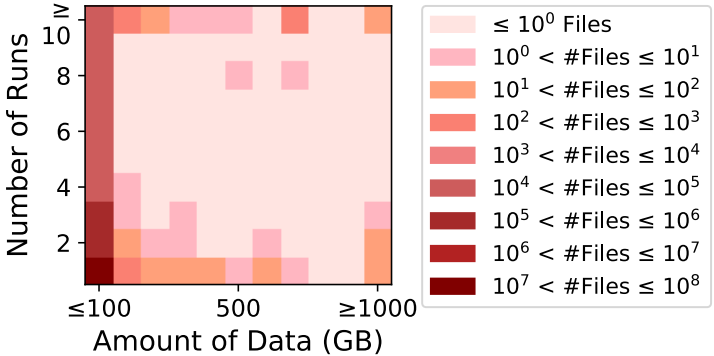


The files that transfer >100 GB and are accessed by multiple applications

8 PB of data transfer, 8.5k files, ~400k jobs

Terms & Definitions

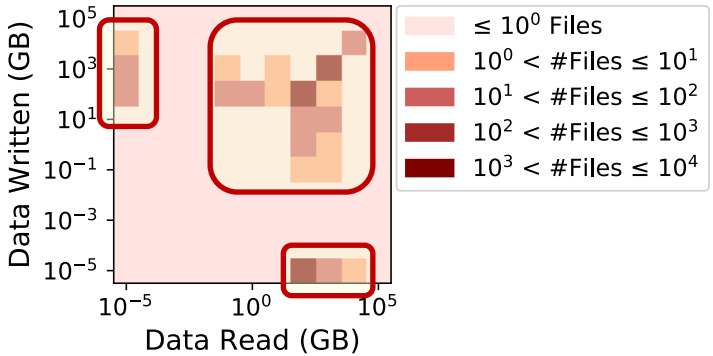
Identifying I/O-intensive files



The files that transfer >100 GB and are accessed by multiple applications

8 PB of data transfer, 8.5k files, ~400k jobs

Dominant I/O type for I/O-intensive files



Write-Heavy (WH) Files (~7%)
Read-Write (RW) Files (~71%)
Read-Heavy (RH) Files (~22%)

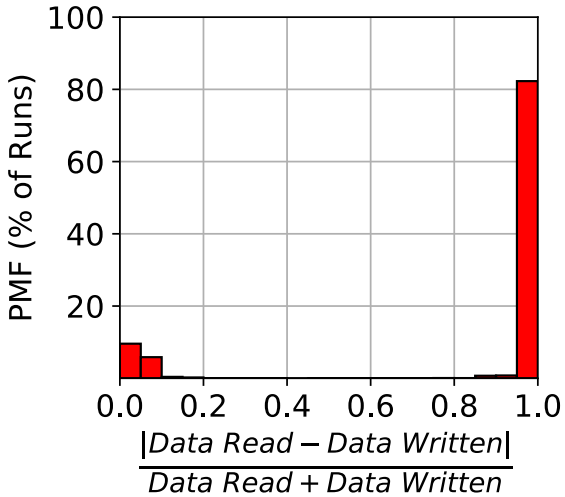
Terms & Definitions

Classification of jobs / runs accessing files

Traditionally, HPC applications are assumed to perform both read and write I/O during the same run in different phases, but does that continue to remain true?

Terms & Definitions

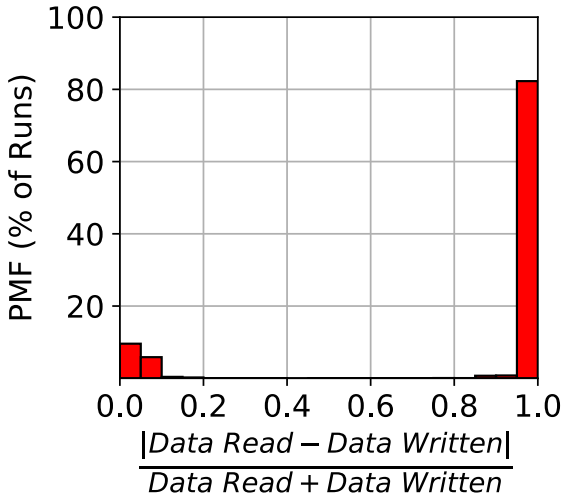
Classification of jobs/ runs accessing files



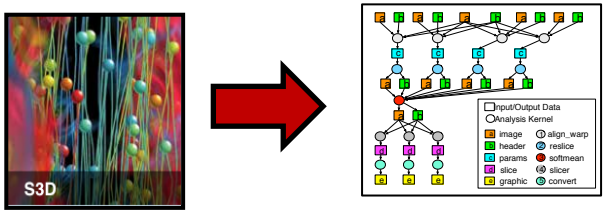
Runs can be classified as “read runs” (68%) or “write runs” (32%)

Terms & Definitions

Classification of jobs / runs accessing files



Runs can be classified as “read runs” (68%) or “write runs” (32%)

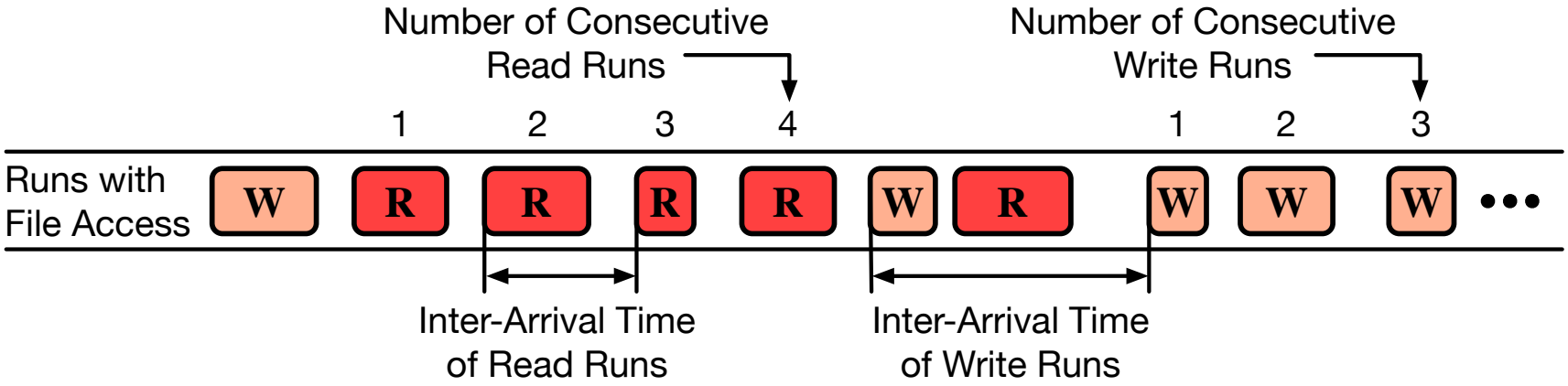


Non-monolithic applications provide the opportunity to better schedule different components of a large workflow to avoid I/O contention among different workflows.

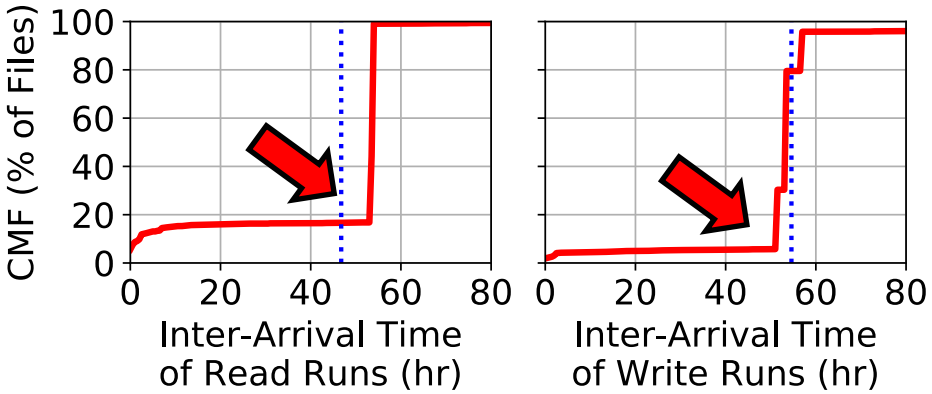
File Reuse Characteristics

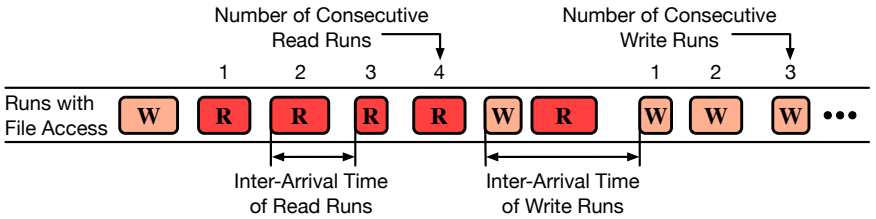
Inter-Arrival of Runs

Inter-arrival times of “read” and “write” runs accessing the same I/O-intensive files



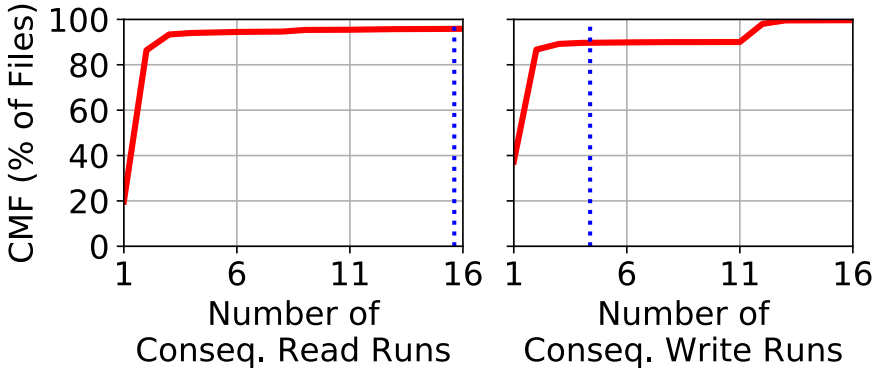
Read & Write Run Inter-arrival Times

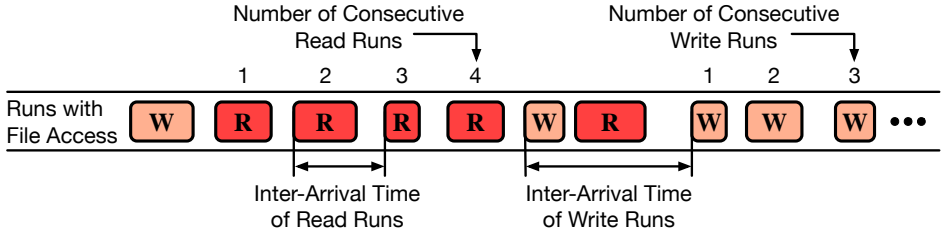




Most I/O-intensive files get re-accessed after a relatively long period (>50 hours) - much longer than the avg. runtime of jobs!
The inter-arrival times for both read and write runs are similar.

Consecutive Runs of the Same Type

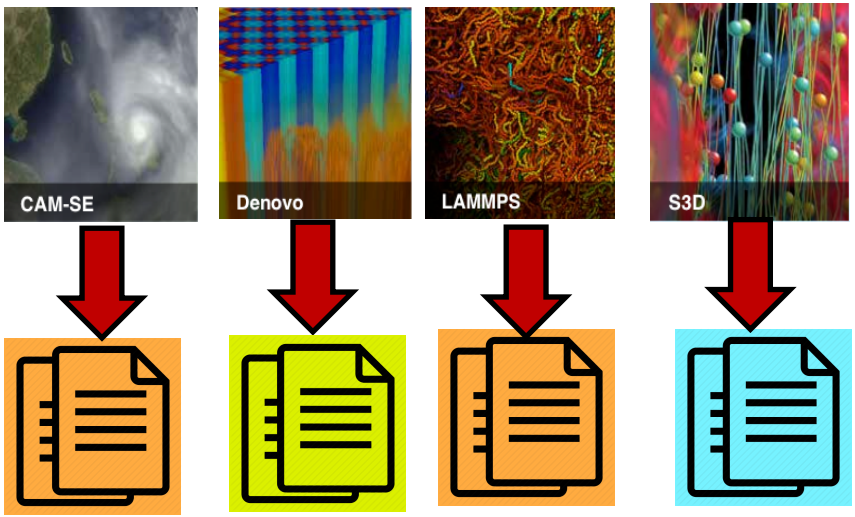




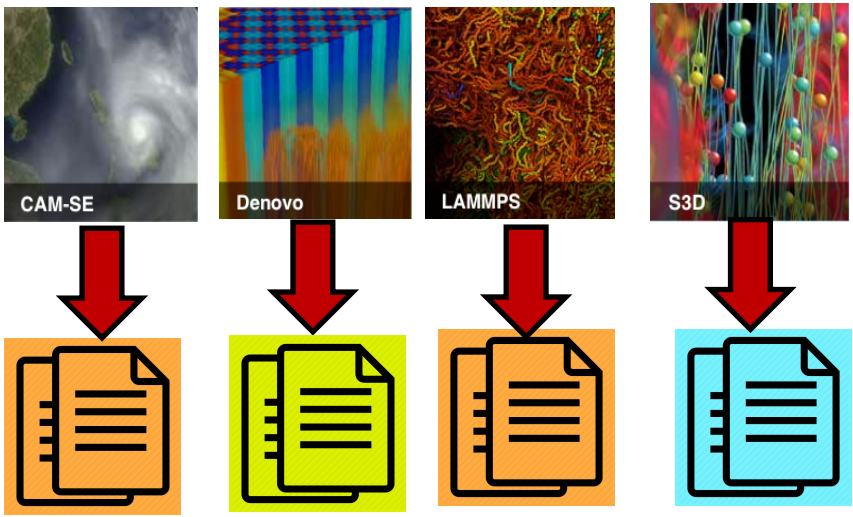
Some HPC files experience a very long string of consecutive read runs, in contrast with smaller consecutive write runs.

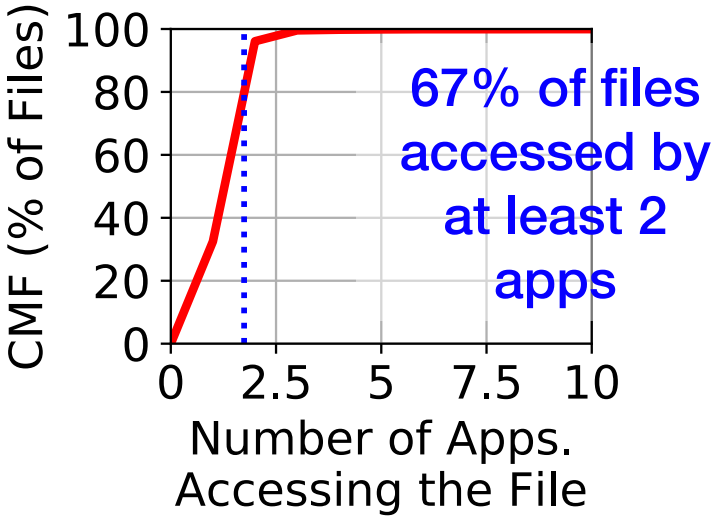
Partitioning of I/O servers to separately serve RH files (which perform many consecutive reads) and RW files (for read and write runs) can boost I/O performance.

File Sharing Across Applications



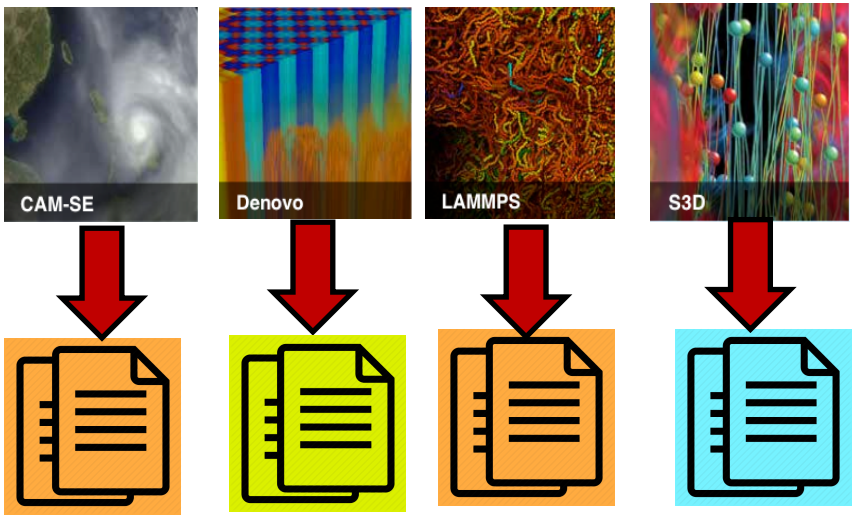
File Sharing Across Applications

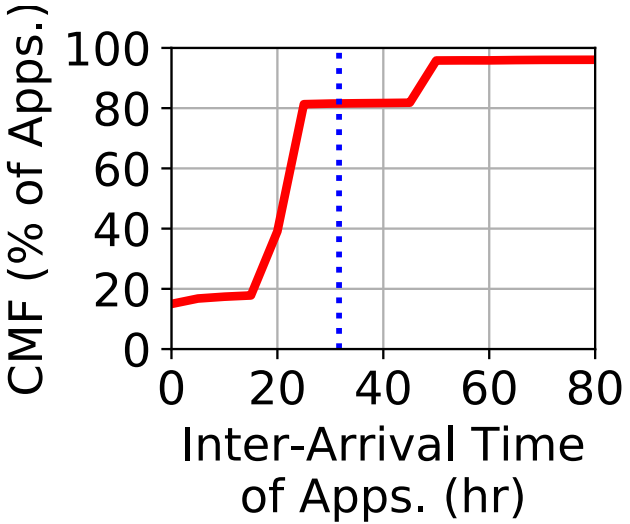




Majority (~86%) of files accessed by multiple applications are RW files. Only 12% of these shared files are RH files and only 2% are WH files.

File Sharing Across Applications





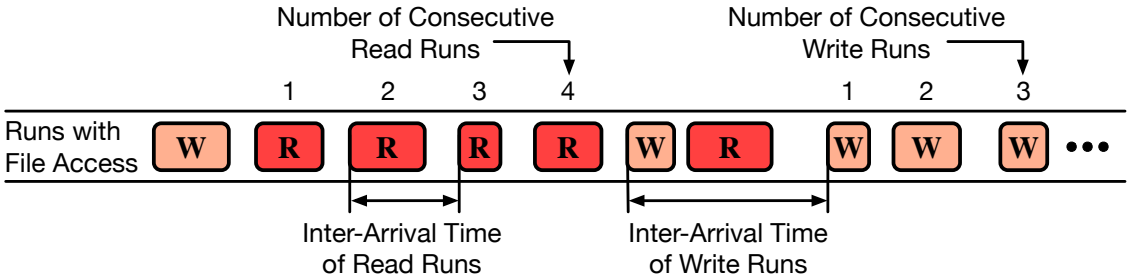
The producer and the consumer are launched significantly apart in time - potential caching across applications has to be carefully managed.

Data Transfer Characteristics
Per I/O Run and Per OST

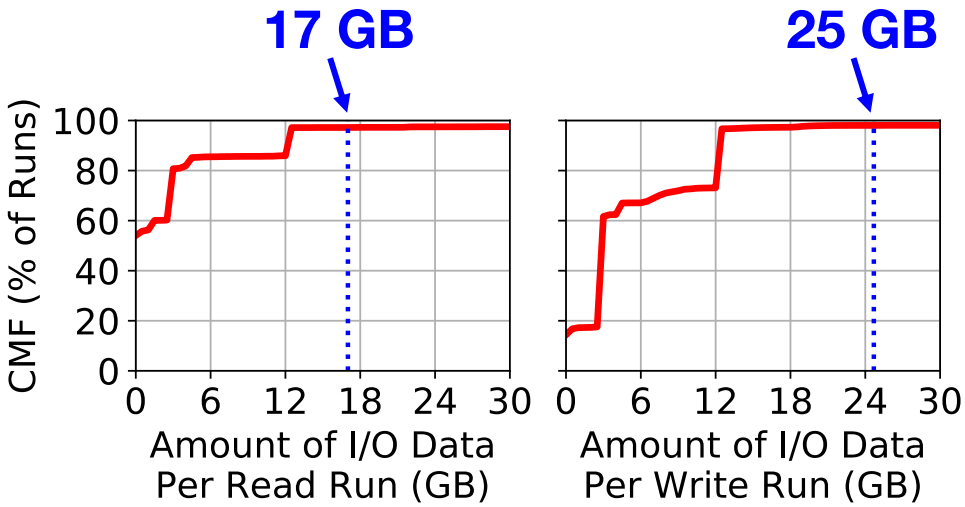
Hypothesis

The total amount of data read is higher than the total amount of data written. The number of read runs are higher than the number of write runs.

Is it accurate to hypothesize that the amount of data read per run is higher than the amount of data written per run?



Data Transfer Per Read & Write Run

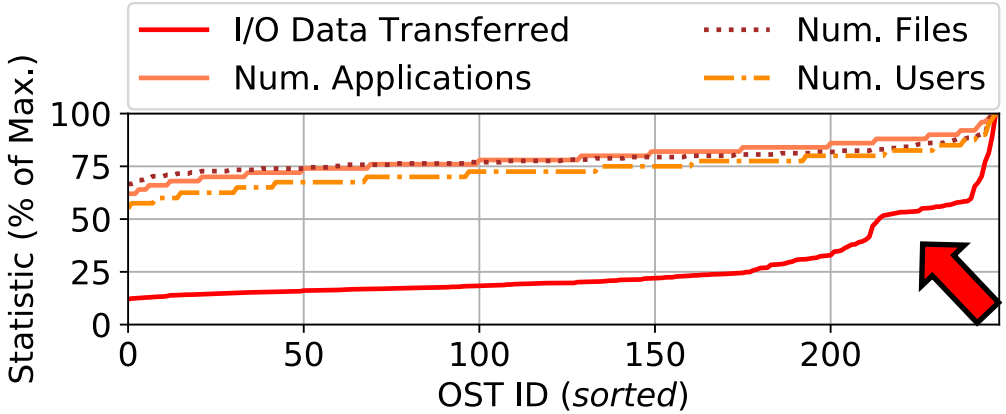


Surprisingly, write runs transfer more amount of data than read runs per run. This finding can be exploited to manage I/O contention better at the system-level by limiting the number of concurrently executing write runs.

Hypothesis

OST are capacity-balanced and have similar amount of data transfer.

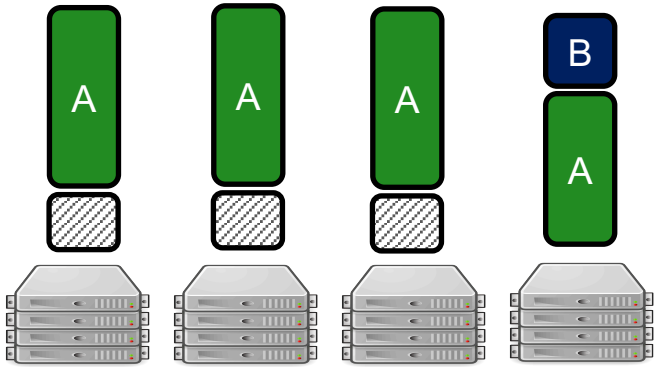
Data Transfer Per OST



OSTs are balanced in terms of number of files but not amount of data transferred per OST -- emphasizing the need for dynamic file migration, and replication of read-only data.

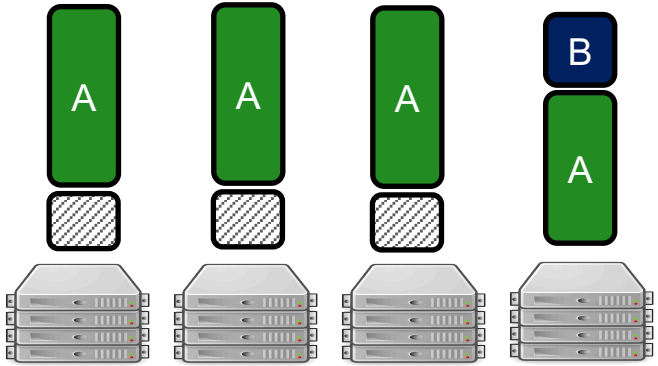
I/O Variability

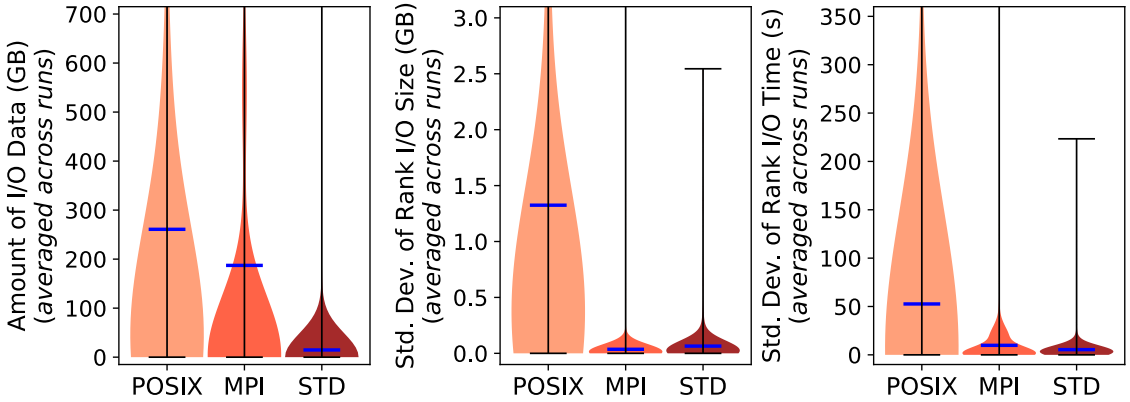
Intra-run I/O Variability



Faster ranks having to wait for the slower ranks to finish I/O before they can resume computation, thus wasting precious compute cycles on the HPC system

Intra-run I/O Variability



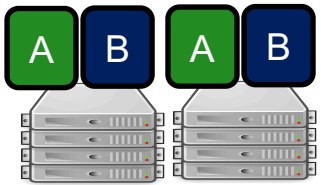


Faster ranks having to wait for the slower ranks to finish I/O before they can resume computation, thus wasting precious compute cycles on the HPC system

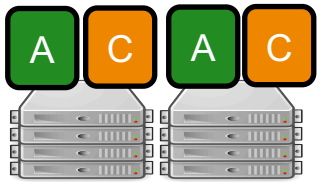
High degree of variability in I/O time of ranks/processes with similar I/O size belonging to the same application.

Inter-run I/O Variability

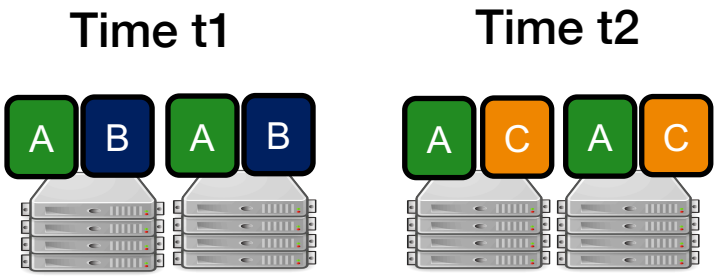
Time t1

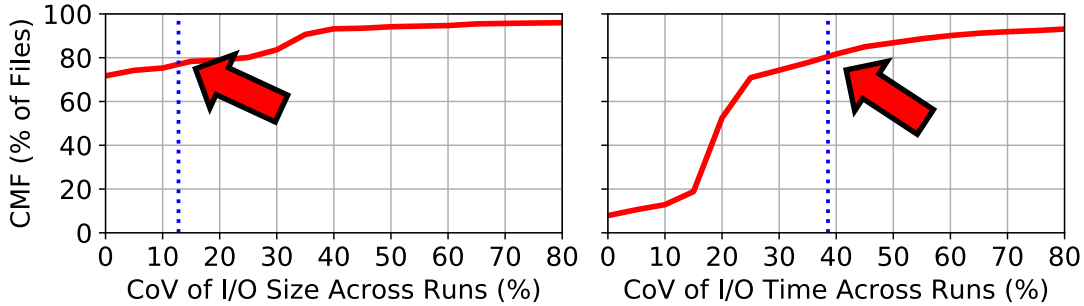


Time t2



Inter-run I/O Variability





HPC files have consistent run-to-run data transfer behavior but observe significantly different I/O time across runs.

Read-heavy files which have the least variability in I/O data size, but the most variability in I/O time - indicating the need for special attention to read-heavy files.

Conclusion

- There is an open opportunity to leverage the favorable characteristics of HPC I/O (classifiable files, single-I/O-type runs, long inter-arrival times) to better manage HPC I/O.

- Lack of coordinated I/O management at the OST level exacerbates the I/O variability and contention challenges.

- Inter-run and intra-run I/O variability not only cause runtime variability but also waste I/O bandwidth and compute cycles