SAND2019-2591PE

High-Performance Computing Storage System Challenges for Theoreticians

Jonathan W. Berry

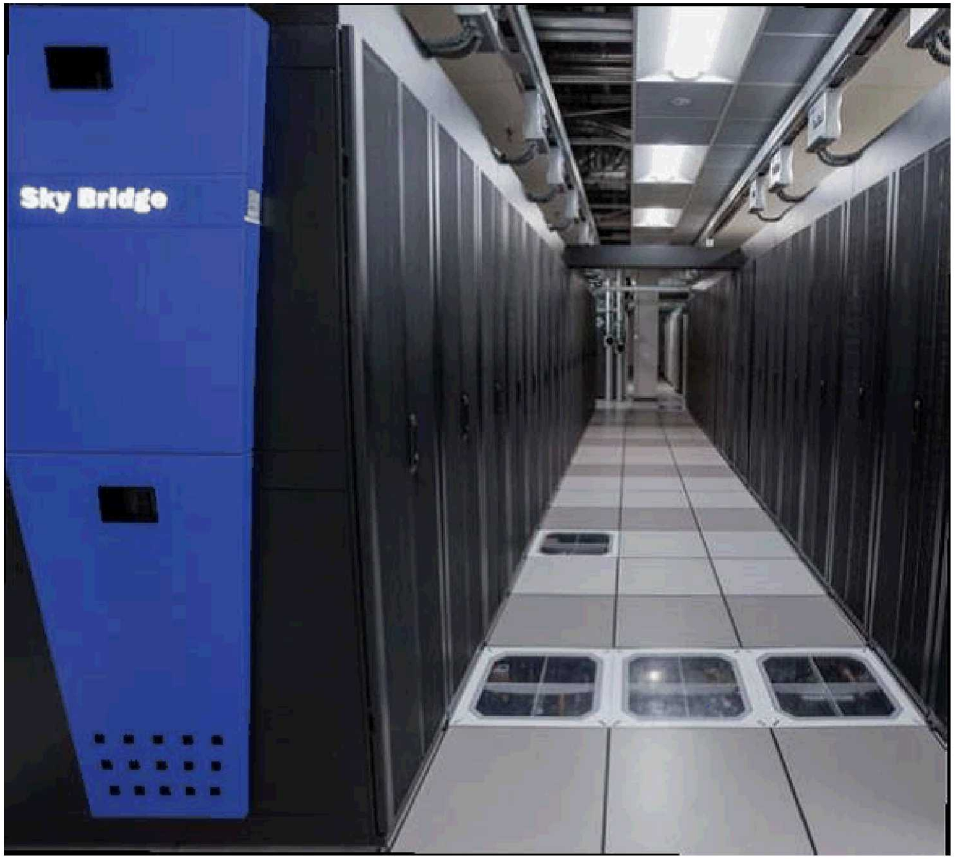
Sandia National Laboratories

March 10, 2019

Supported by the Laboratory Directed Research and Development program at Sandia National Laboratories, a multimission laboratory managed and operated by National Technology and Engineering Solutions of Sandia, LLC., a wholly owned subsidiary of Honeywell International, Inc., for the U.S. Department of Energy's National Nuclear Security Administration under contract DE-NA-0003525.

Outline/Intro

- Caveat: I’m not a filesystems person
- This talk is about challenges I’ve learned about by discussions with Sandia experts
- I’ll conclude with some motivations for data structures research



Laboratory Directed Research & Development

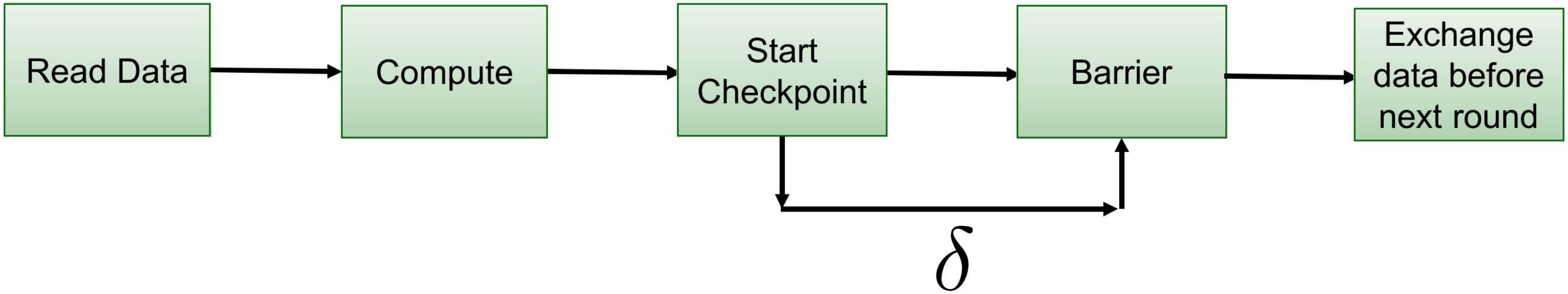
High-Performance Computing (HPC)

- **Scientific Computing HPC Applications**
  - Primary motivation: modeling and simulation
  - Style of computing: Single Program/Multiple Data (SPMD)
  - Programming model: C/C++/Message Passing Interface (MPI)
  - Primary emphasis: Network, Compute nodes
  - Storage: Parallel Distributed Filesystems
  - Storage challenge: ***Checkpoint/restart***
- **Data Analysis Applications**
  - Primary motivation: graph algorithms, machine learning, etc.
  - Style of computing: shared-memory
  - Programming model: (e.g. PGAS)
  - Storage challenge: ***Stream and load the data***
- ***Caveat: Vendor emphasis is not HPC (think: Gaming!)***

I/O Challenges: Scientific Computing

- Warning: decades of HPC I/O literature out there

- ***Checkpoint/restart:*** the once and future problem



Read Data

Compute

Start Checkpoint

Barrier

Exchange data before next round

δ

- All processes tend to slam the filesystem at the same time.
- Challenge: $\delta \to 0$
- Interesting result from Loncaric (Los Alamos N.L.):

$$\frac{\text{JMMTI}}{\delta} \geq 200 \text{ is a good regime}$$

("Job Mean Time To Interrupt")

I/O Challenges: Scientific Computing

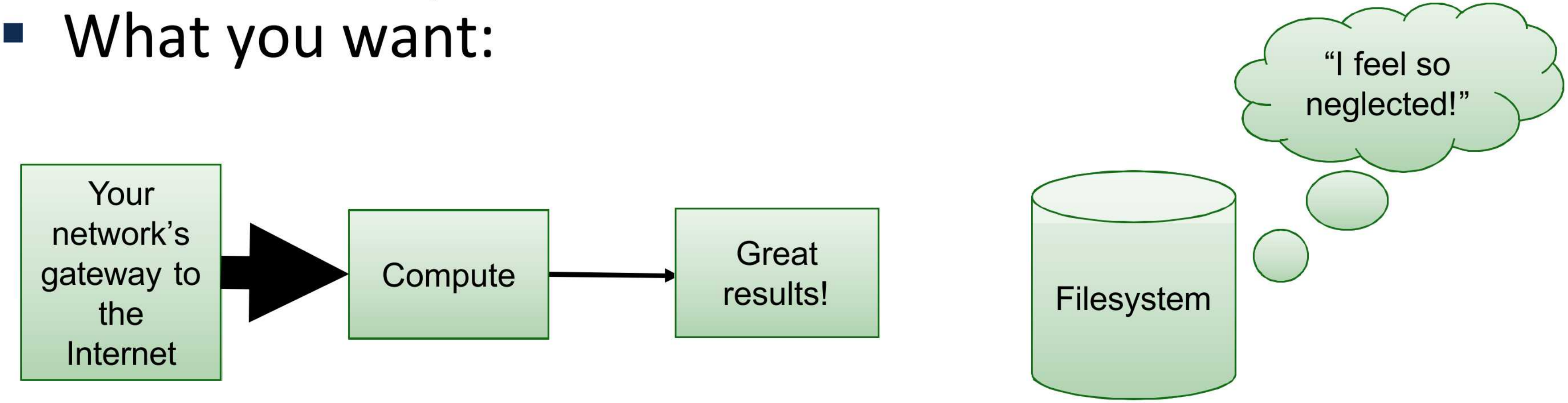
- ***Launch:*** another pain point
  - "copying the binaries can take 12 minutes (1500 nodes); we want 0s" - Lee Ward (Sandia National Labs)
  - But this might be a vendor problem rather than a systems/algorithms problem
- ***Typical I/O-relevant acceptance tests (there are many)***
  - ***"clobber-create"*** : create new files with unique fnames
  - ***"IOR (Interleaved or Random)"*** : random access
  - ***"Bonnie++"*** : file system benchmarking tool for measuring I/O performance
- ***How to have more impact in HPC (e.g.)***
  - ***Lustre*** (parallel & distributed filesystem) is open-source. BetrFS-style write-optimization ideas in Lustre would have much more HPC impact here than stand-alone

HPC Filesystems: Lab Expert POC's

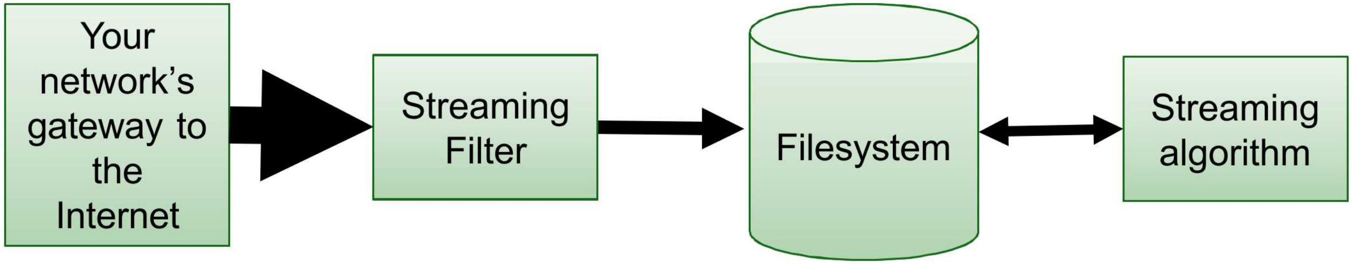
- Matthew Curry (Sandia National Labs)
- Robin Goldstone (Lawrence Livermore National Labs)
- Gary Gryder (Los Alamos National Labs)
- Glenn Lockwood (Lawrence Berkeley National Labs)
- Jay Lofstead (Sandia National Labs)
- Robert Ross (Argonne National Labs)
- Brad Settlmeyer (Los Alamos National Labs)
- Lee Ward (Sandia National Labs)

I/O Challenges: HPC Data Analysis

- Motivation: cyber streams, etc.
- What you want:

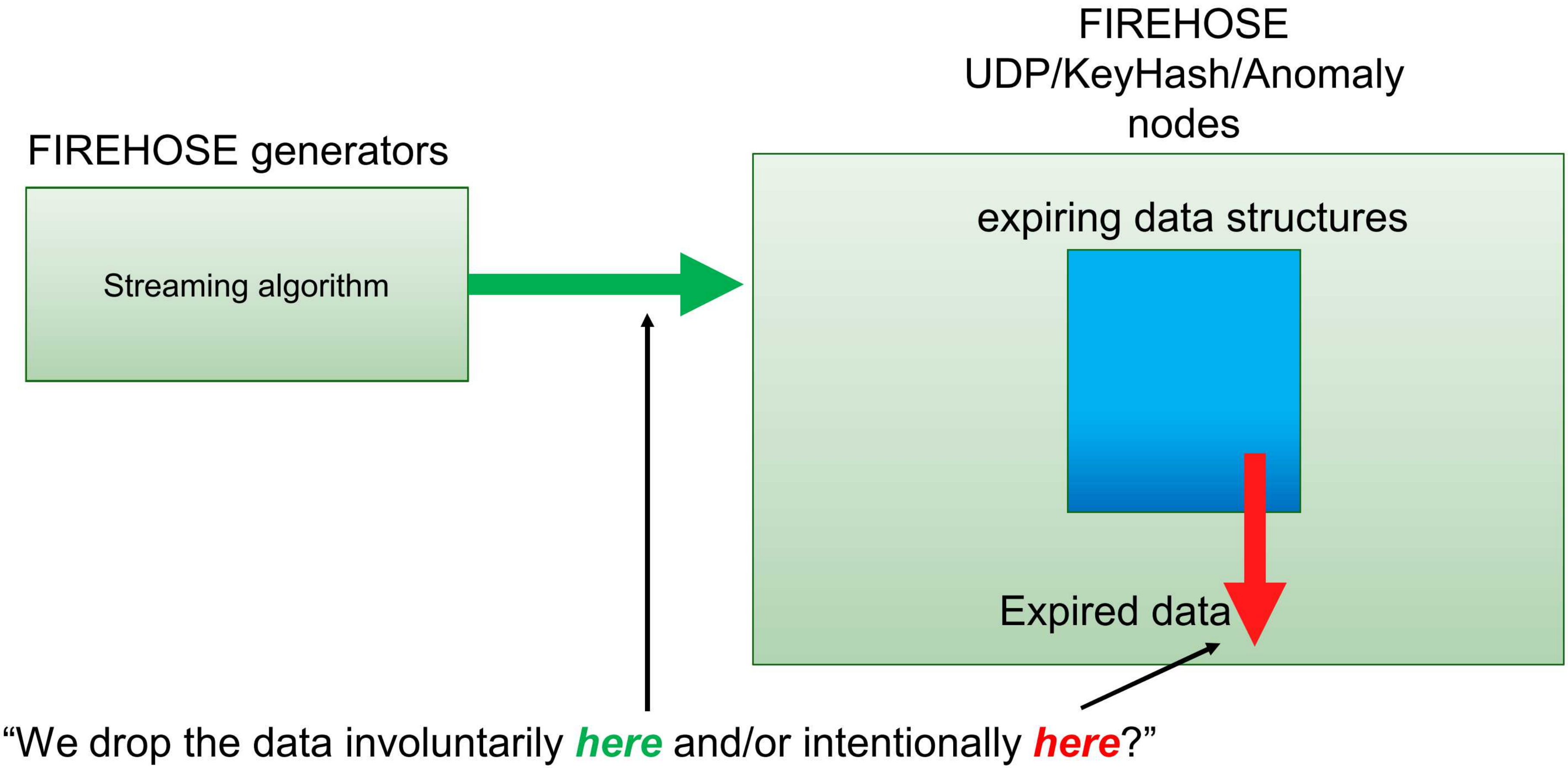


- What you're probably stuck with:



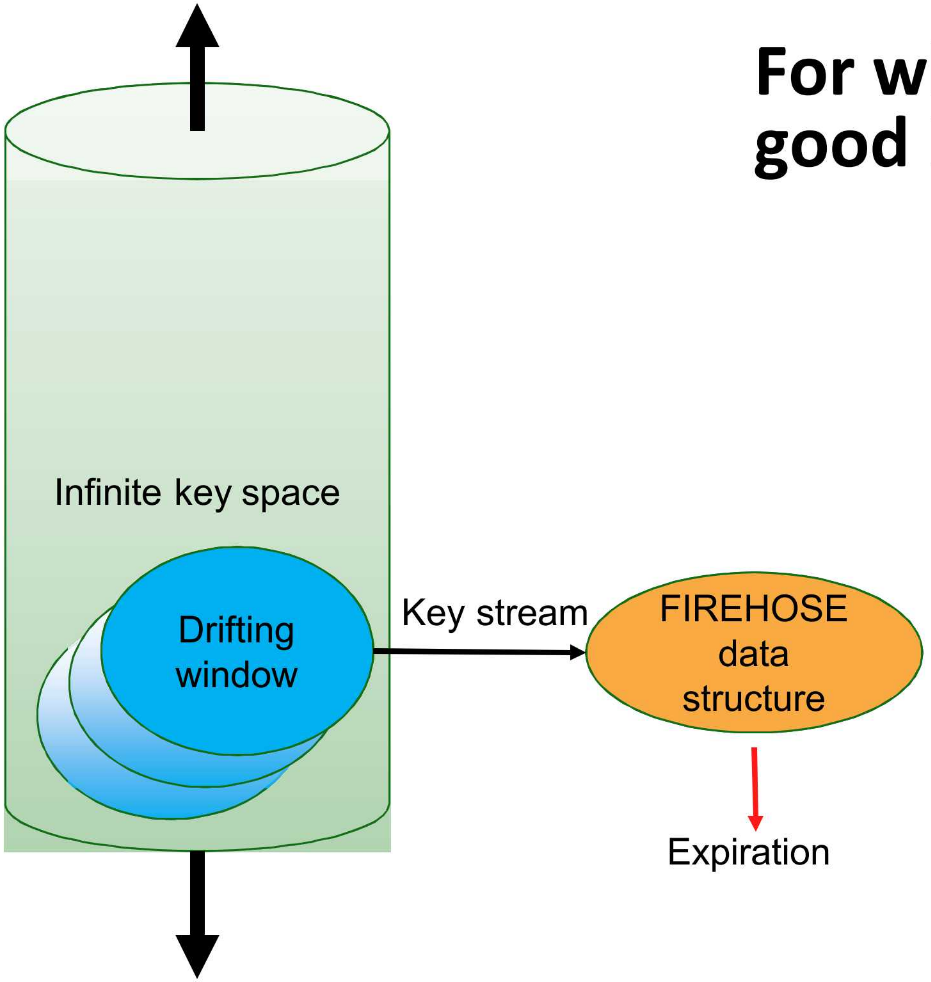
FIREHOSE “Active” Generator

With an infinite key space, we have been forced to ask different questions

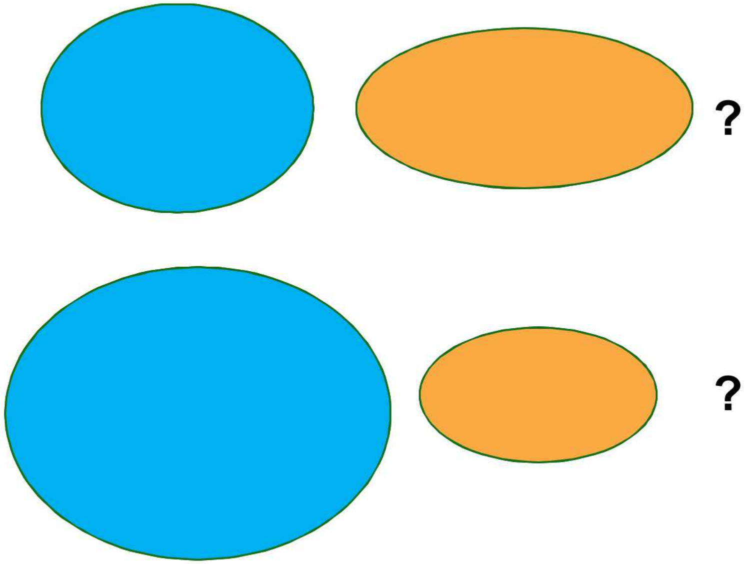


“We drop the data involuntarily ***here*** and/or intentionally ***here***?”

Active Generator Window Size



For what size ratios can we expect good FIREHOSE accuracy?



Waterslide: Where do We "Drop?"

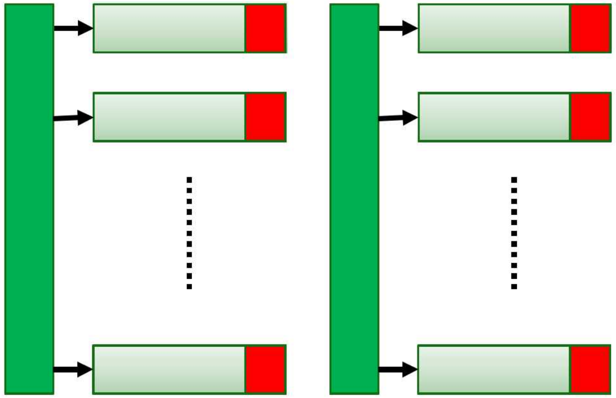
- Run FIREHOSE reference impl. in ***waterslide*** (open source)
- Look at "confusion matrix" (result) of FIREHOSE
- How many packets did we drop (generator->analytic)?
- How many reportable keys (>=24 occurrences) did we report (50M keys generated)

Table Size	Generator Window Size	Reportable keys	Reported keys	**Packet drops**
2^20	2^20	94,368	62,317	0
2^20	2^21	63,673	15,168	0
2^20	2^22	17,063	9	0

https://github.com/waterslideLTS/waterslide

What is Happening?

- **Waterslide uses 'd-left hashing'**
  - Two rows of buckets
  - Constant-size
  - Fast
  - Waterslide adds LRU expiration *per bucket*



Broder, Andrei, and Michael Mitzenmacher. "Using multiple hash functions to improve IP lookups." *INFOCOM 2001*

- **1/16 of all data is always subject to immediate expiration in steady state**

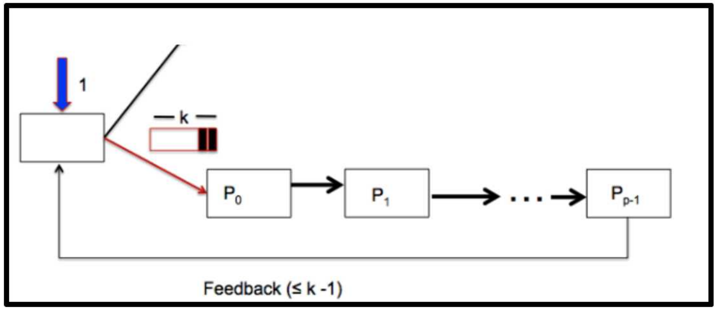
- **As active generator window grows, FIREHOSE accuracy quickly goes to zero**

Even when window size is only 4x data structure size, most reportable data are lost before being reported on.

Motivation for Data Structure Research

- **Premises**
  - Global key space often greater than data structure size
  - Data structure insertion time is not the bottleneck in distributed FIREHOSE
  - A working global expiration strategy could preserve accuracy for larger ratios of generator window size to data structure size

- **Approaches**
  - You’ll hear about “*Popcorning*”
  - You’ll also hear about the “*x-stream*” model



Berry, et al. "Maintaining connected components for infinite graph streams." *Proceedings of the 2nd International Workshop on Big Data (KDD), ACM 2013*

Conclusions

- Go forth and help the HPC community!

- Thank you!

jberry@sandia.gov