MSST 2018: NON-VOLATILE MEMORY API SESSION

INCORPORATING NVM INTO DATA-INTENSIVE SCIENTIFIC COMPUTING

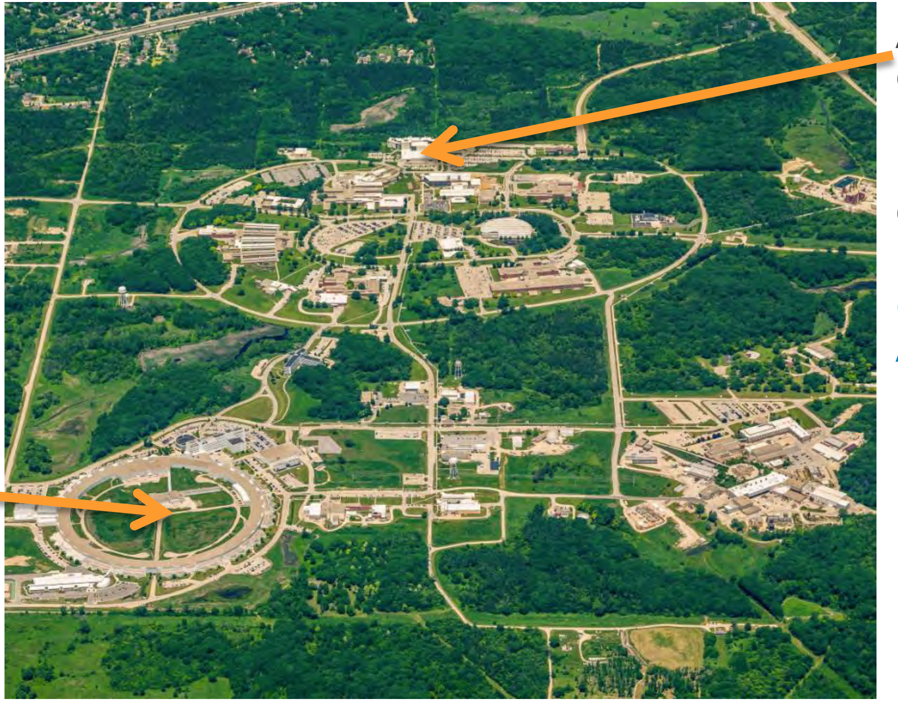
PHIL CARNS
Mathematics and Computer Science Division
Argonne National Laboratory

May 15, 2018
Santa Clara, CA

ARGONNE NATIONAL LABORATORY

U.S. Department of Energy

"Argonne is a multidisciplinary science and engineering research center"



Argonne Leadership Computing Facility

IBM Blue Gene/Q (Mira)
Cray XC40 (Theta)

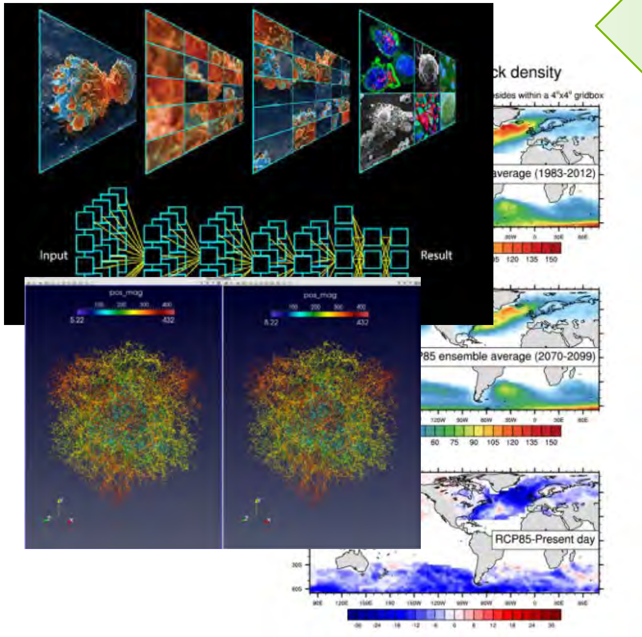
(Under construction:
A21 exascale system)

Advanced Photon Source

(Under construction:
APS upgrade)

THE ROLE OF ANL/MCS DATA-INTENSIVE SCIENCE RESEARCH

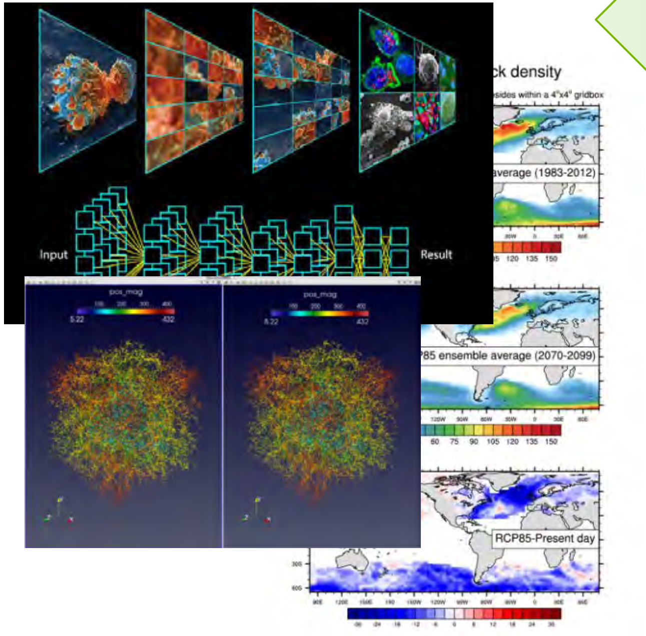
(one perspective)



Techniques, algorithms, and software to bridge the “last mile” between scientific applications and storage systems

THE ROLE OF ANL/MCS DATA-INTENSIVE SCIENCE RESEARCH

(one perspective)



This entails:

- Characterizing access
- Modeling architectures
- Optimizing data services
- **Incorporating new technology such as NVM**

DATA-INTENSIVE SCIENTIFIC COMPUTING

Constraints on NVM integration from an end-user perspective

- Efficiency
  - CPU hours (and storage) are a scarce commodity
  - This has a direct impact on scientific time to solution
- Portability
  - Applications must execute on multiple platforms
  - The science itself will outlive all of those platforms
- Ease of use
  - Scientists would like to focus on their problem domain
  - Not the mysterious ways of vendor_api_write_foo()

DATA-INTENSIVE SCIENTIFIC COMPUTING

Potential solutions in the storage design space

- Efficiency
- Portability
- Ease of use

1. A global parallel file system
   – POSIX is portable and easy to use (or at least well understood)
   – Re-engineering needed to address latency shifting by orders of magnitude
   – Semantics and API make this challenging
2. “Here are some NVM devices: have fun!”
   – Dedicated developers will always be able to maximize efficiency with this approach
   – Not enough ninja programmers for this to be a viable long term option
3. Specialized data services
   – *There are challenges and opportunities*
   – ***NVM APIs** can help*

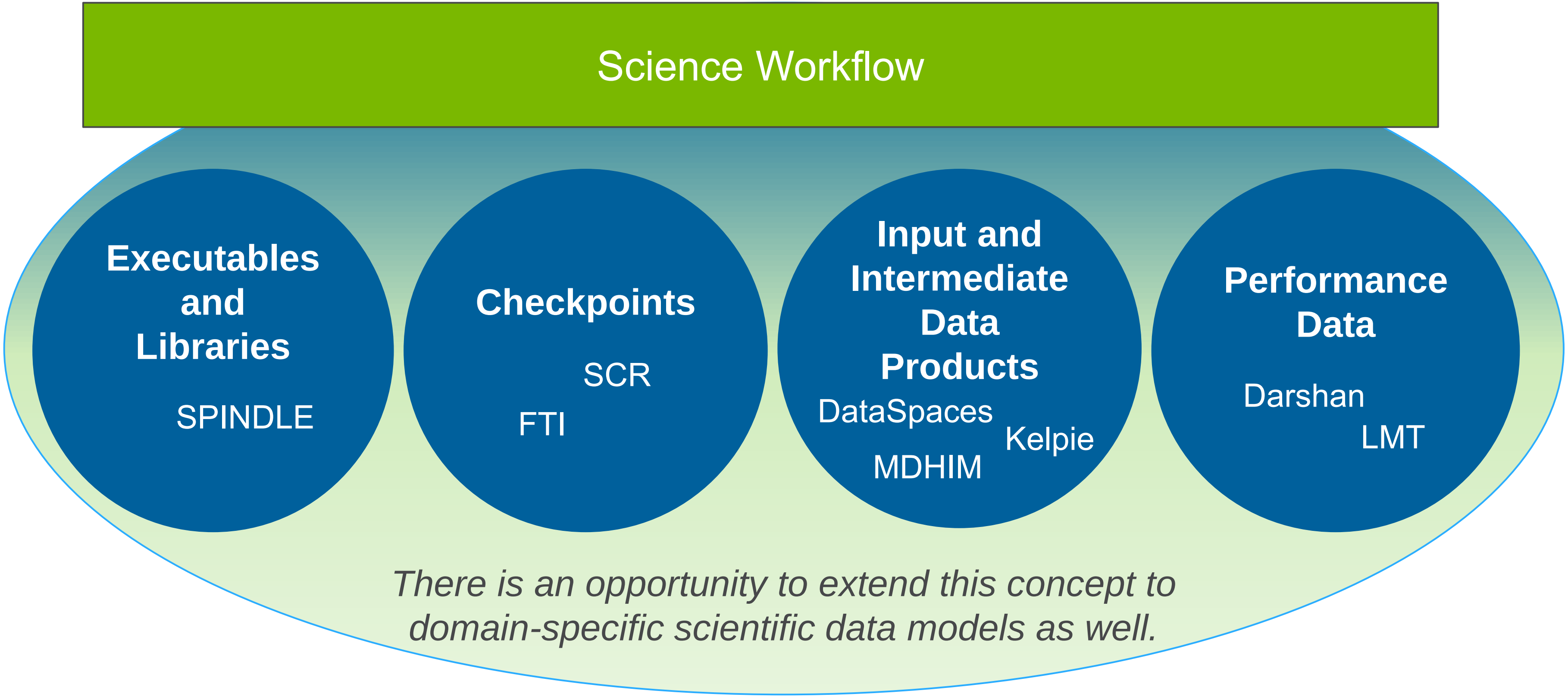
WHAT DO WE MEAN BY SPECIALIZED DATA SERVICES?

SPECIALIZED DATA SERVICES

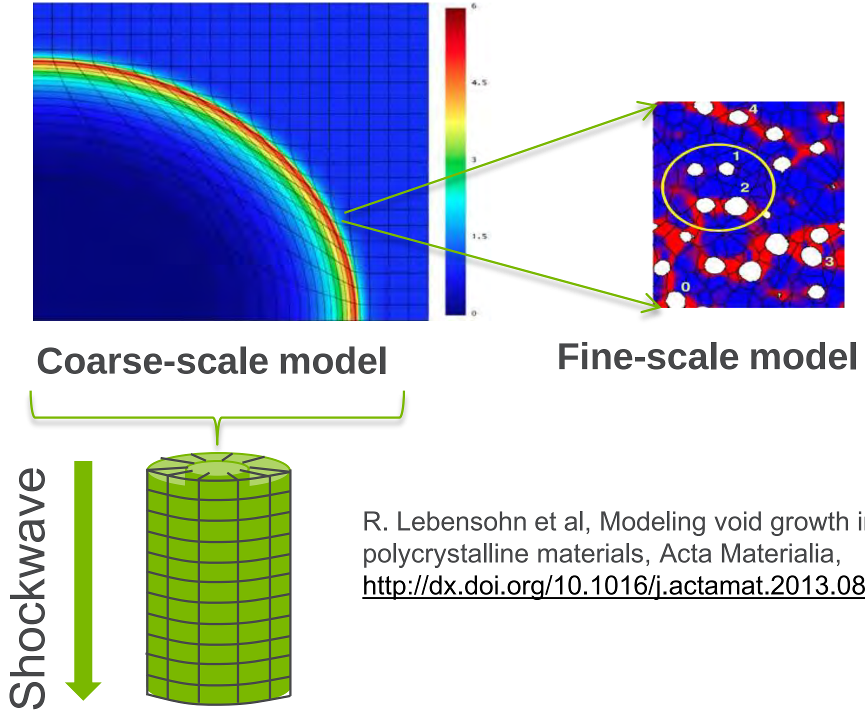
- Semantics and capabilities tailored to a problem domain
- Provisioned and instantiated on-demand
- Abstracting storage technology from the application
- Target more than just checkpointing
- *A way to leverage NVM characteristics by bypassing conventional storage software infrastructure*

Examples are already common in HPC!

AN ECOSYSTEM OF DATA SERVICES



A SCIENTIFIC DATA MODEL EXAMPLE: MULTI-SCALE SIMULATION



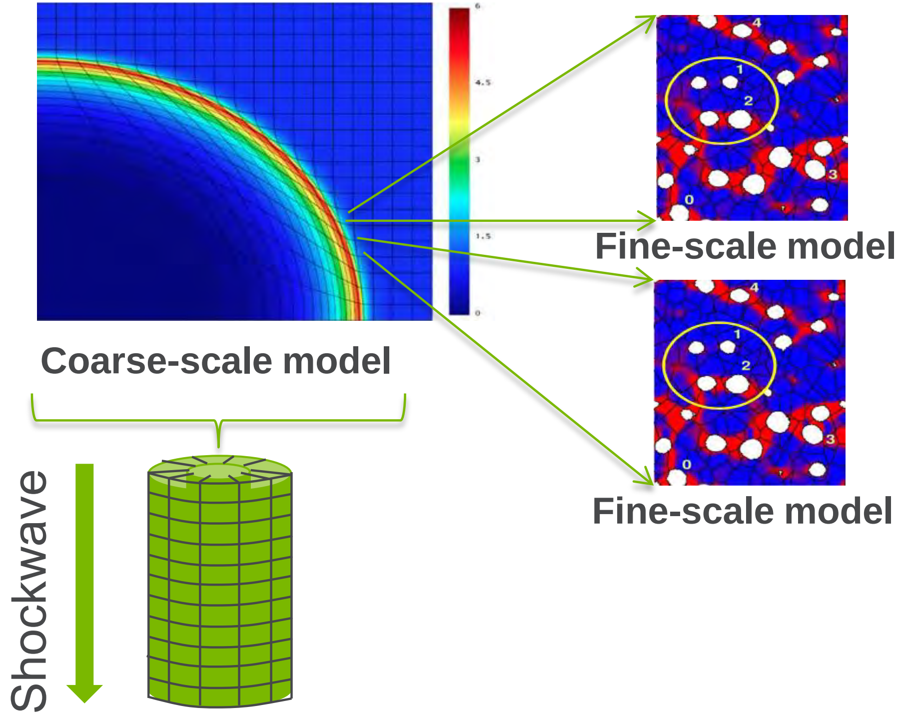
R. Lebensohn et al, Modeling void growth in polycrystalline materials, Acta Materialia, http://dx.doi.org/10.1016/j.actamat.2013.08.004.

Multi-scale models simulate across multiple time and length scales.

This example is a hydrodynamics unstructured mesh with an FFT-based PDE solver.

We will use it to illustrate a motif that occurs in other problem domains as well and highlights the need for reusable building blocks.

A SCIENTIFIC DATA MODEL EXAMPLE: MULTI-SCALE SIMULATION

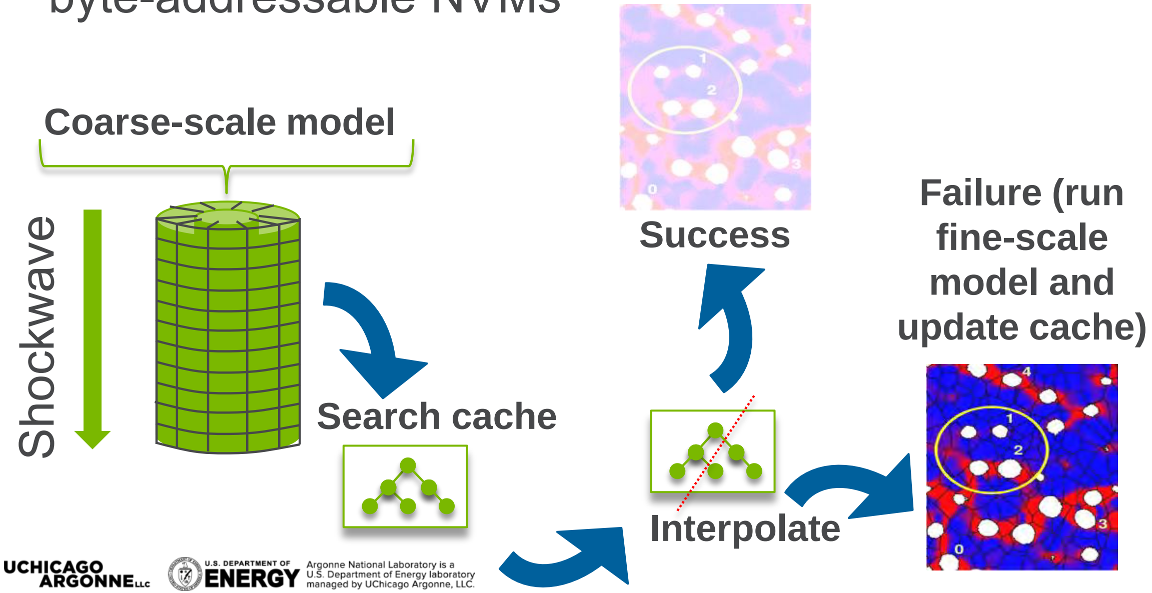


- Phenomena such as shock waves propagate through course-scale model
- This sometimes requires recomputation of similar (or identical) fine-scale models

If the fine-scale model is expensive, then we should cache fine-scale results for later use.

COMPUTATIONAL CACHING AS A SPECIALIZED DATA SERVICE

- Search cache for nearest neighbors in parameter space, interpolate, and check error bounds
- Could be a distributed data service that leverages low latency, byte-addressable NVMs



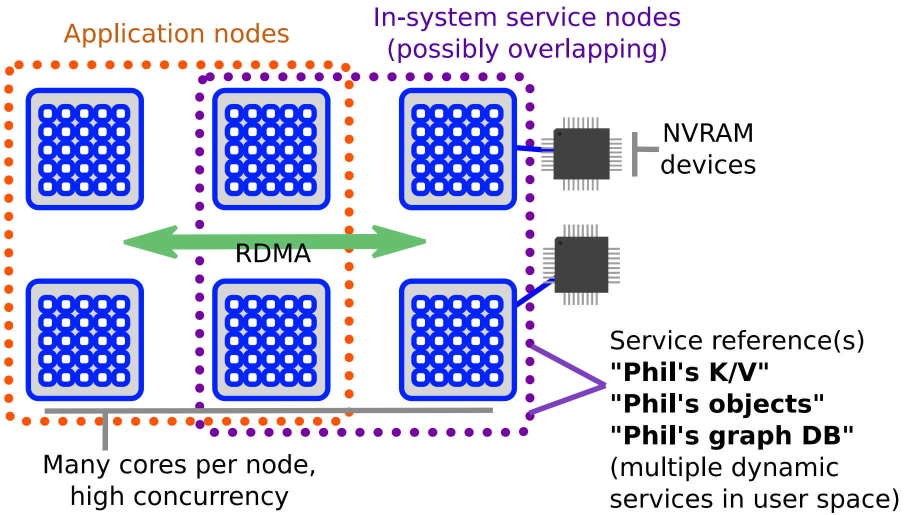
This isn't a standard file system or database.

NVM APIs:

- Give us building blocks for new data models
- Let us differentiate classes of memory

TECHNICAL CHALLENGES FOR SPECIALIZED DATA SERVICES IN HPC

- Where is the NVM?
  - Local to compute nodes, remote access, or remote access via fabric?
- Integration with custom HPC networks
  - Dragonfly, torus, fat tree, exotic APIs
- Concurrency
  - Applications with > 100 thousand processes
- Access mode
  - User-space access helps to enable dynamic services on time-shared systems

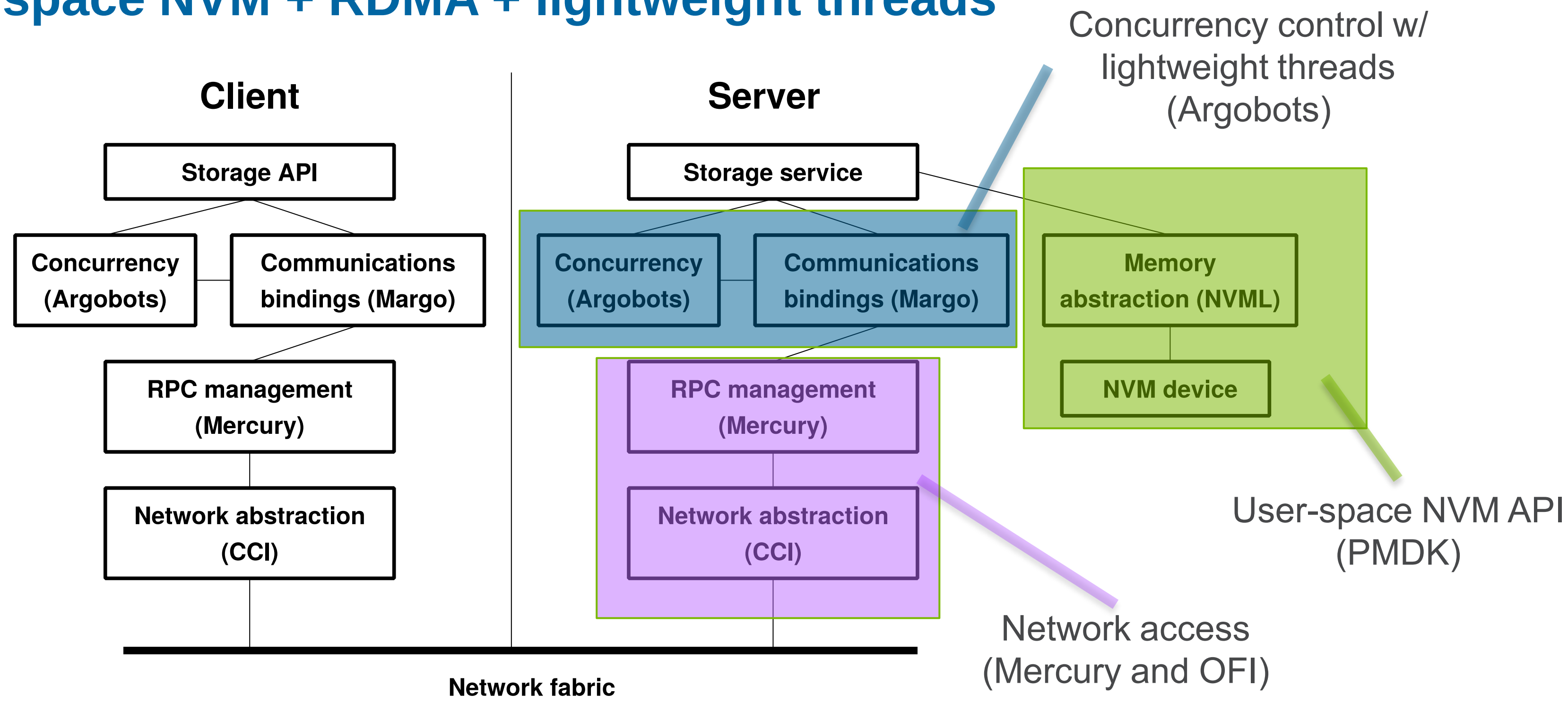


The Mochi Project
ANL, LANL, CMU, HDFG
https://www.mcs.anl.gov/research/projects/mochi

BUILDING SPECIALIZED DATA SERVICES WITH NVM

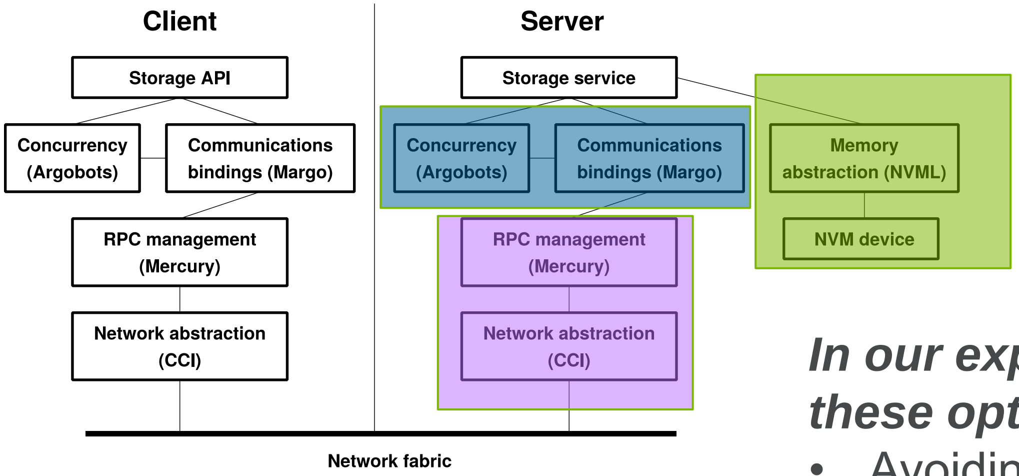
ARCHITECTING AN NVM-BACKED DATA SERVICE

User space NVM + RDMA + lightweight threads



ARCHITECTING AN NVM-BACKED DATA SERVICE

User space NVM + RDMA + lightweight threads



Modularity helps with extensibility, portability, and reuse, but is this too many layers/components?

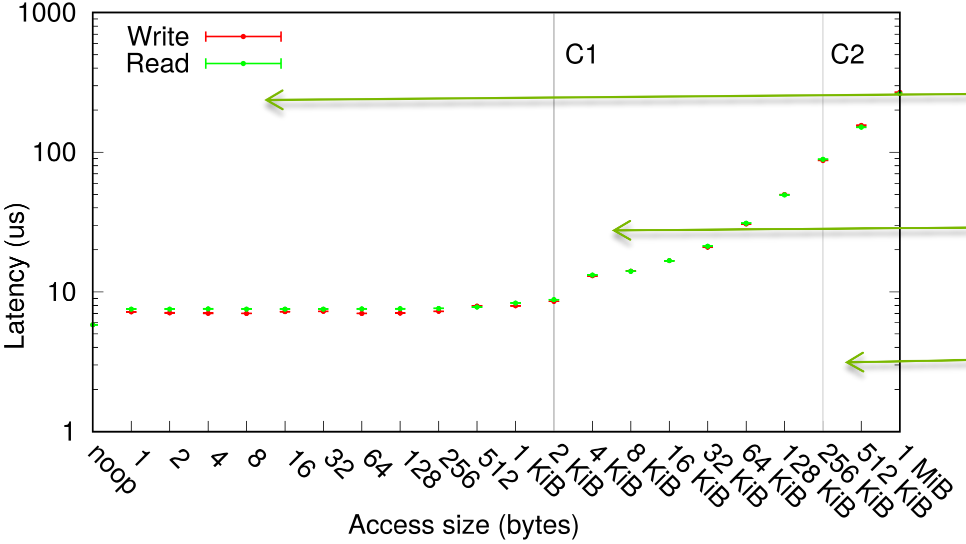
In our experience, no. We prioritize these optimizations instead:

- Avoiding privileged mode transitions
- Avoiding context switches in general
- Avoiding memory copies
- Reducing CPU load

ACCESS LATENCY

How much latency do those software layers add?

- RAM in place of pmem
- No busy polling
- Each access is at least 1 network round trip, 1 libpmem access, and 1 new thread



Protocol modes:

- Eager mode, data is packed into RPC msg
- Data is copied to/from pre-registered RDMA buffers
- RDMA "in place" by registering memory on demand

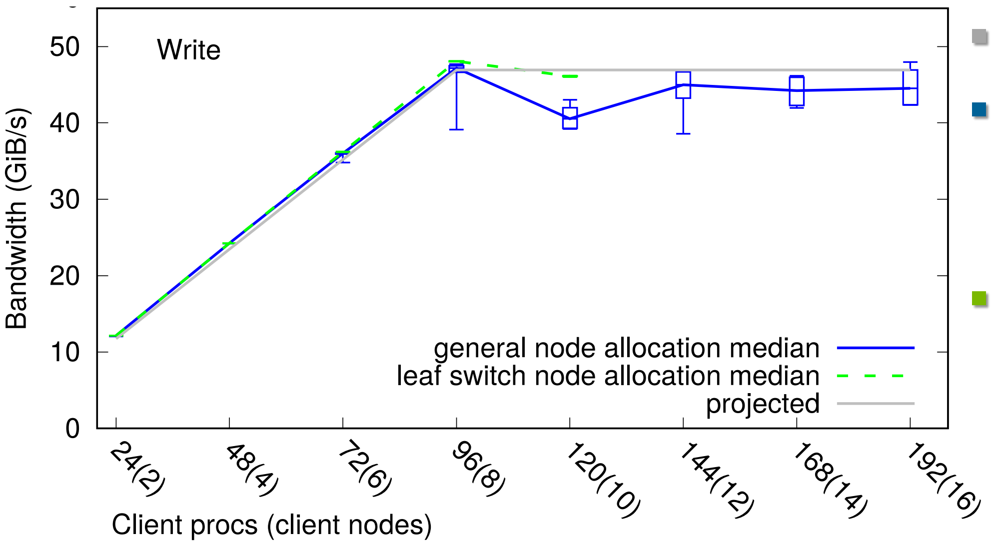
Crossover points would be different depending on transport

ACCESS LATENCY

Observations and questions

- Single digit microsecond access latencies: could it be tuned further?
  - Consider adaptive polling
  - Optimize memory allocation
  - OFI providers (and others) are improving rapidly
- What about the long tail?
  - Previous slide shows confidence interval for 10,000 samples at each point, and the intervals are quite narrow
  - But there are outliers: worst noop sample was > 70 microseconds
  - This leads to the dreaded jitter problem in HPC
- The cost of memory copy vs. registration is a key factor in optimization

AGGREGATE BANDWIDTH



- Grey line is projected maximum
- Blue line is a normal allocation
  - Whiskers (min and max) show significant variance
- Green line is an allocation with all nodes on one leaf switch
  - Whiskers (min and max) show very little variance

- Same system as in previous example
- 8 servers (1 per node)
- Up to 192 application processes (12 per node)

AGGREGATE BANDWIDTH

Observations and questions

- New problems arise when storage latency isn't the longest pole in the tent:
  - E.g., network topology (In this example, internal switch routing)
  - Consider dynamic routing and congestion-avoidance algorithms?
  - Better internal service instrumentation?
  - Make the storage system topology-aware?
- The service can saturate aggregate bandwidth relatively easily
- PMDK atomics help avoid serialization
  - Especially when creating and destroying objects
- How does this software architecture hold up at larger scales?

COMMENTARY ON THE ROLE OF NVM APIS IN SCIENTIFIC COMPUTING

- We surely appreciate faster file systems and databases, but there are many other possibilities to consider
- NVM is easier to integrate into HPC if it gets along with our other technologies
  - RDMA networks, user-space provisioning, lightweight concurrency
- Bottlenecks aren't where they used to be
- Some degree of standardization is helpful
  - Minimize burden on developers for portability
- What is the role of PMoF?
  - Important technology, but not a full solution for concurrency and flow control
- Right now focus is on "get it to work, fast!", but focus will shift over time: characterization, elasticity, multi-objective optimization, and more

THANK YOU!

THIS WORK WAS SUPPORTED BY THE U.S. DEPARTMENT OF ENERGY, OFFICE OF SCIENCE, ADVANCED SCIENTIFIC COMPUTING RESEARCH, UNDER CONTRACT DE-AC02-06CH11357.

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.