HPC-IODC AND WOPSSS 2018

Argonne NATIONAL LABORATORY

UNDERSTANDING AND TUNING HPC I/O: HOW HARD CAN IT BE?

PHIL CARNS
Mathematics and Computer Science Division
Argonne National Laboratory

UCHICAGO ARGONNE LLC

U.S. DEPARTMENT OF ENERGY

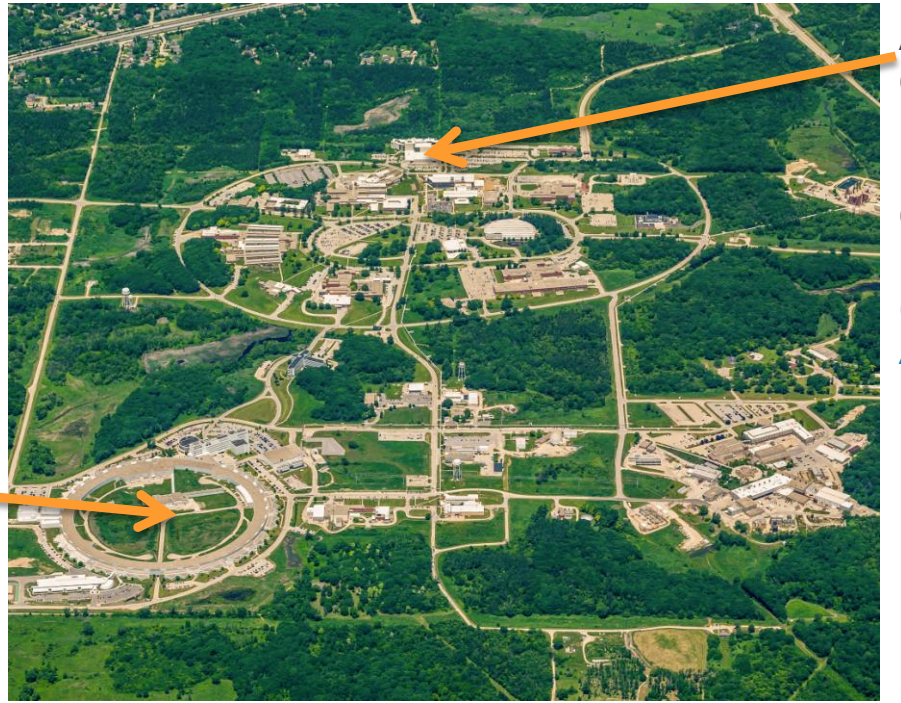
Argonne National Laboratory is a U.S. Department of Energy laboratory managed by UChicago Argonne, LLC.

June 28, 2018
Frankfurt Germany

A LITTLE ABOUT MYSELF AND MY COLLEAGUES

U.S. Department of Energy's Argonne National Laboratory

"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)

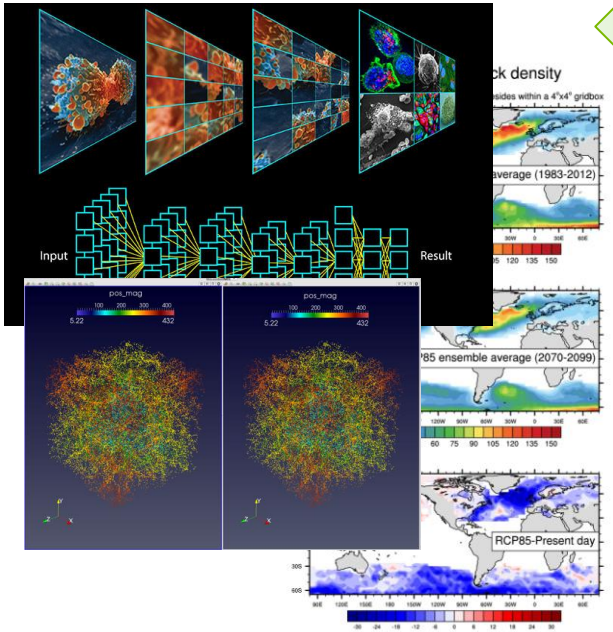
Advanced Photon Source

(Under construction: APS upgrade)

The Mathematics and Computer Science department carries out HPC research in support of DOE science.

THE ROLE OF DATA-INTENSIVE COMPUTER SCIENCE RESEARCH

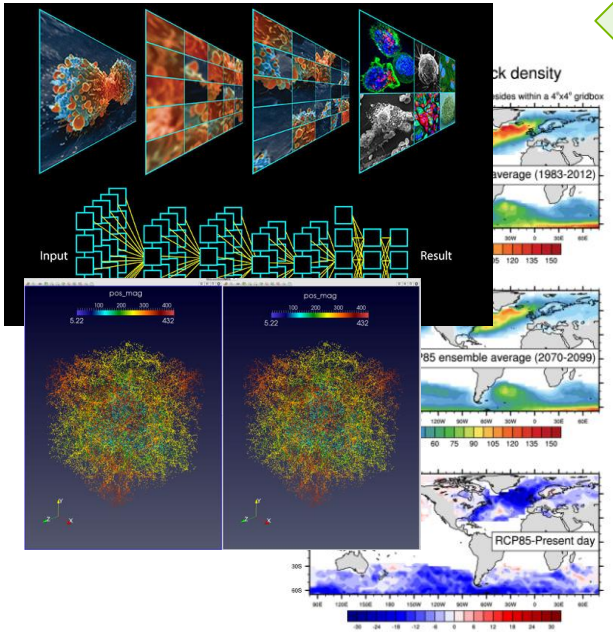
(one perspective)



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

THE ROLE OF DATA-INTENSIVE COMPUTER SCIENCE RESEARCH

(one perspective)



This entails:

- Characterizing access
- Modeling architectures
- Building and optimizing data services
- Putting new technology into the hands of scientists

OR MORE SIMPLY:
UNDERSTANDING AND TUNING HPC I/O

UNDERSTANDING AND TUNING HPC I/O

Can be expressed in terms of the OODA loop concept from strategy and control theory.

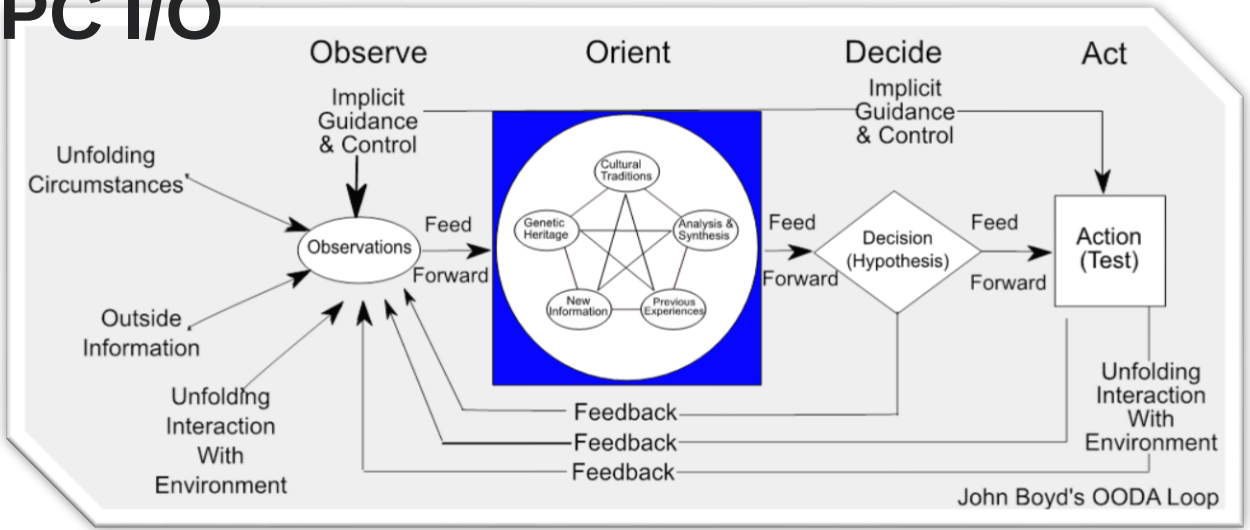


Figure by Patrick Edwin Moran

- **Observe:** instrument applications and systems
- **Orient:** interpret performance data in context
- **Decide:** determine how improve
- **Act:** implement improvements

UNDERSTANDING AND TUNING HPC I/O

Can be expressed in terms of the OODA loop concept from strategy and control theory.

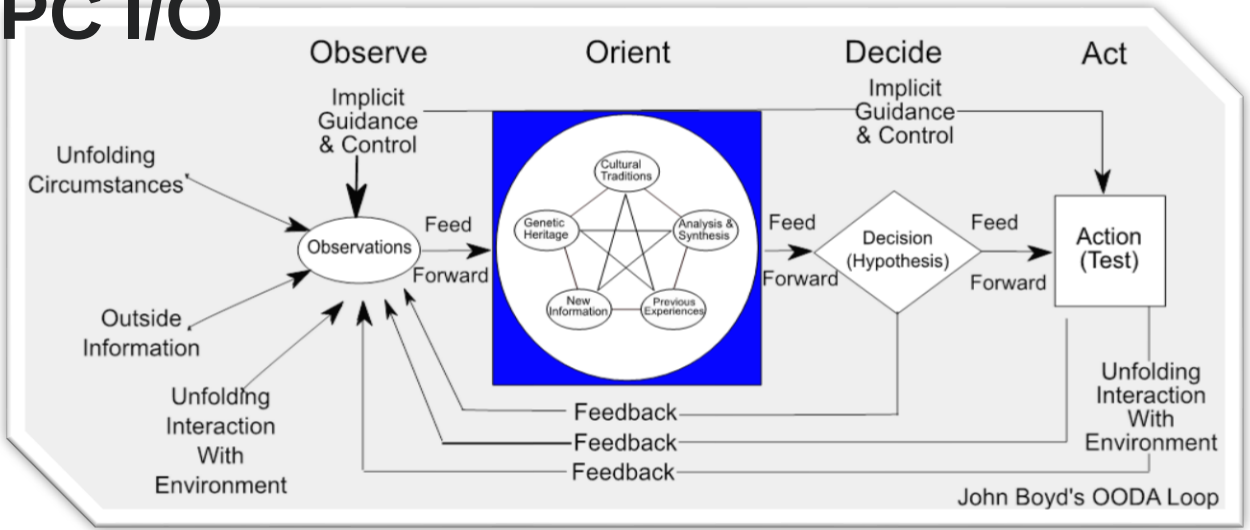
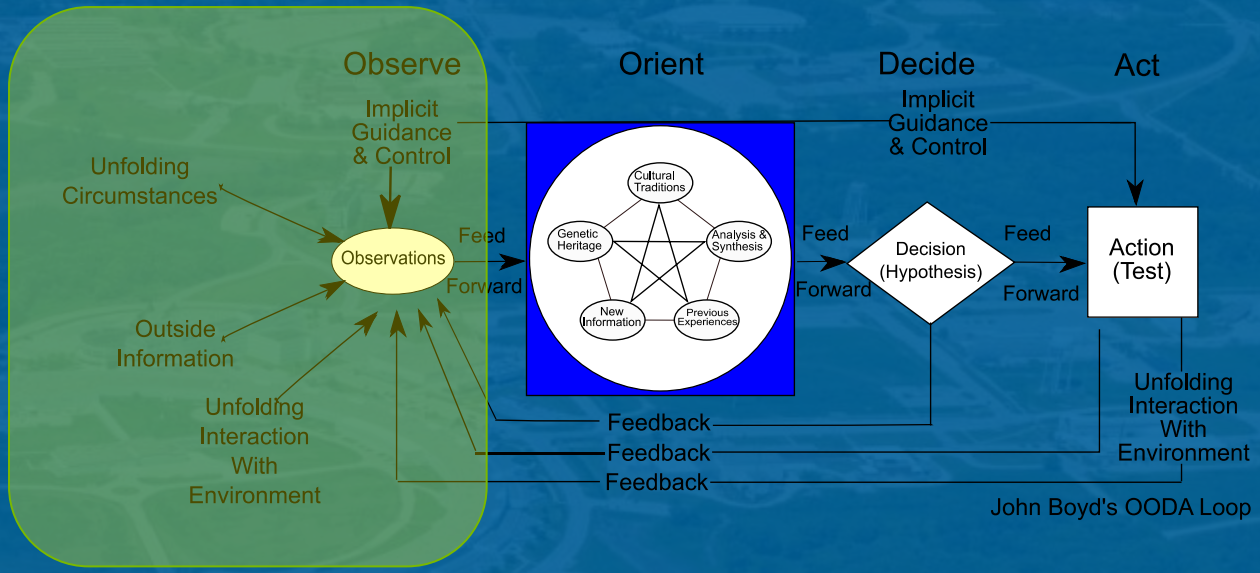


Figure by Patrick Edwin Moran

The concept is simple enough, but it may be difficult to apply to HPC I/O.

This presentation will explore implications, share experiences, and highlight challenges in understanding and tuning HPC I/O in this conceptual framework.

OBSERVE



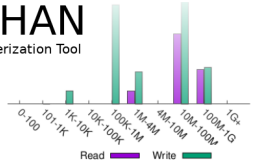
OBSERVING HPC I/O

- A wide variety of tools are available for this purpose
- They instrument different facets of HPC I/O
  - Application behavior
  - System behavior
  - Resource usage
  - Correctness
  - External data sources
- Data integration frameworks are also maturing

We (as a community) have made considerable strides in scalable data collection in the past decade, but challenges remain.

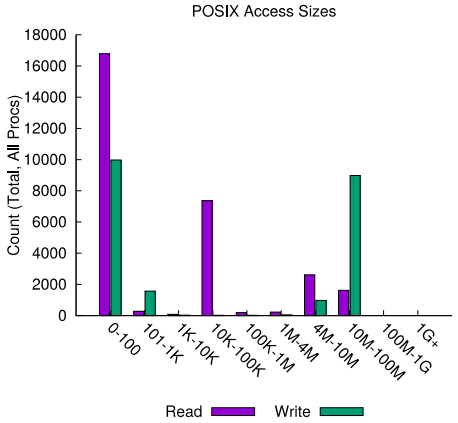
Figures provided by Shane Snyder
https://www.mcs.anl.gov/research/projects/darshan

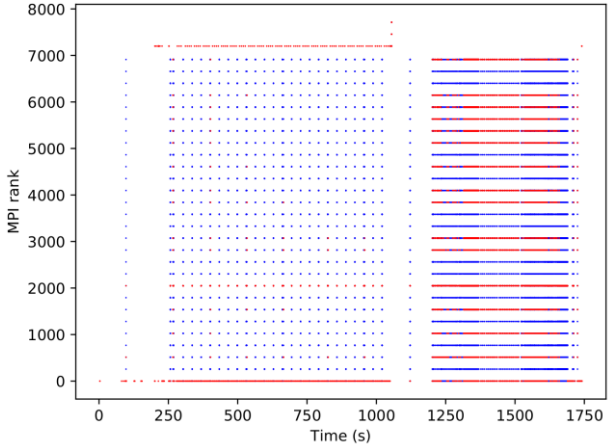
DARSHAN
HPC I/O Characterization Tool



CHALLENGES IN OBSERVATION

You always need more data, but sometimes not until after the fact





This calls for dynamic instrumentation based on policies or triggers

Always-on characterization tools can routinely provide data like this.

But sometimes you need higher fidelity data to find the cause of a complex performance problem.

- Both figures above are generated from Darshan data
- But the latter mode cannot be enabled at all times due to data collection cost

CHALLENGES IN OBSERVATION

Emerging technology

- Storage technology is a moving target. Examples:
  - Alternative data models (not just POSIX)
  - Ephemeral namespaces and resources
  - Non-volatile memory

Screenshot from https://github.com/pmem/pmdk

This tree contains a collection of libraries for using Non-Volatile Memory (NVM). There are currently nine libraries:

- **libpmem** -- basic pmem operations like flushing
- **libpmemblk**, **libpmemlog**, **libpmemobj** -- pmem transactions
- **libvmem**, **libvmmalloc**[1] -- volatile use of pmem
- **libpmempool** -- persistent memory pool management
- **librpmem**[1] -- remote access to persistent memory (EXPERIMENTAL)
- **libpmemcto** -- close-to-open persistence (EXPERIMENTAL)

and two command-line utilities:

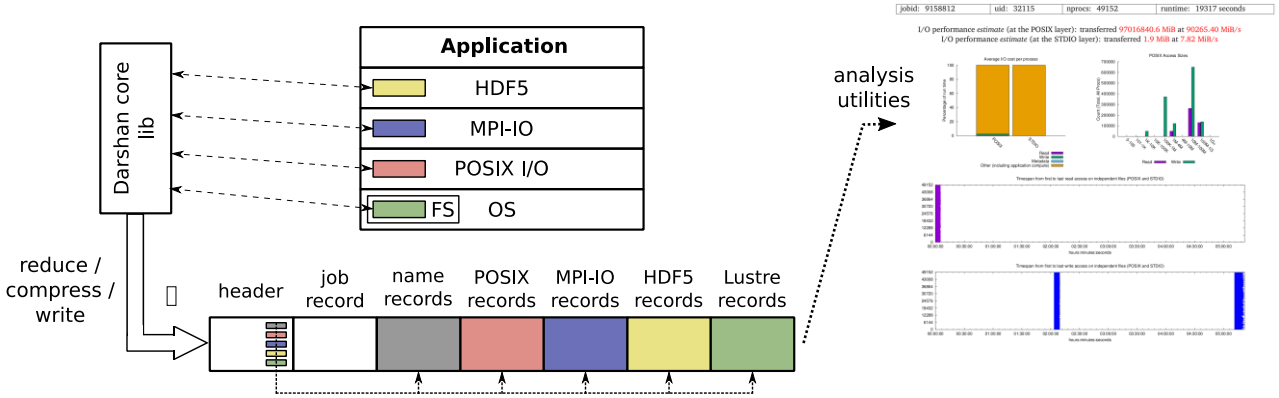
- **pmempool** -- standalone tool for off-line pool management
- **daxio** -- perform I/O on Device-DAX devices or zero a Device-DAX device

- Consider the (excellent) PMDK package for NVM access
- 9 libraries, 2 utilities, 0 performance instrumentation hooks!
- We can't wrap load/store operations with interception libraries

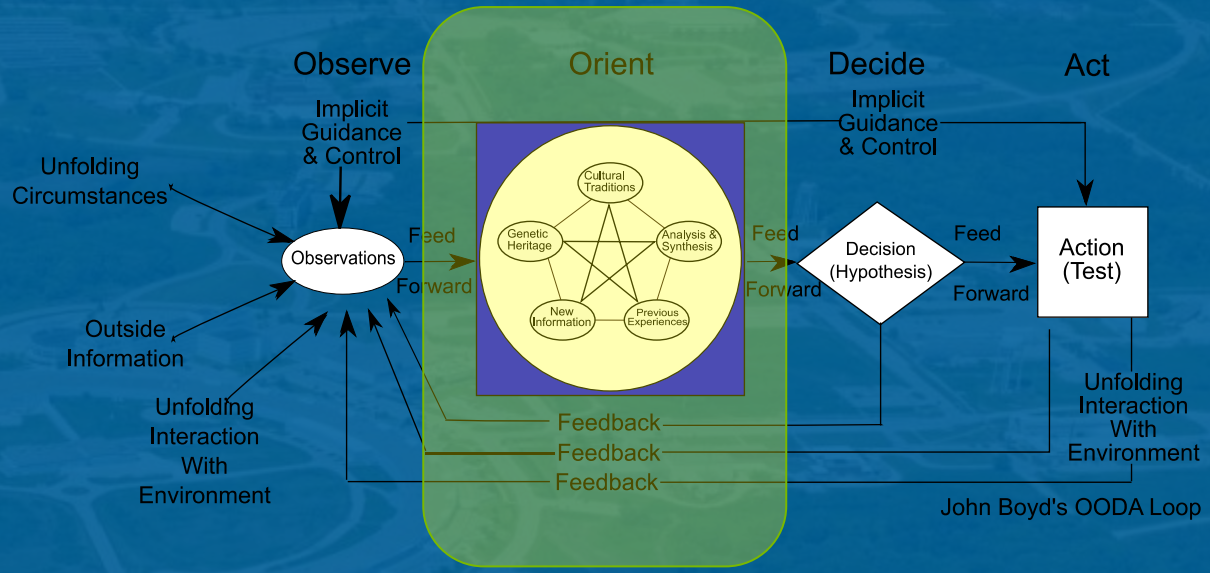
CHALLENGES IN OBSERVATION

Emerging technology

- One way to mitigate this problem is through modularity in instrumentation tools
- Still doesn't (by itself) tell you how to instrument pmem or a new data model
- But prevents us from re-inventing the wheel for scalable data collection



ORIENT



ORIENTING (INTERPRETING) HPC I/O DATA

- I/O performance can be misleading in isolation
- Must be considered in the context of application and system capabilities
- State of the art for assessing system capabilities:
  - Benchmarks
    - Hero runs, synthetic workloads, proxy applications
  - Comparison against historical trends
    - Time series data, especially from systems
  - Vendor specifications
  - I/O modeling

Do these things give us sufficient experience to orient our observations?

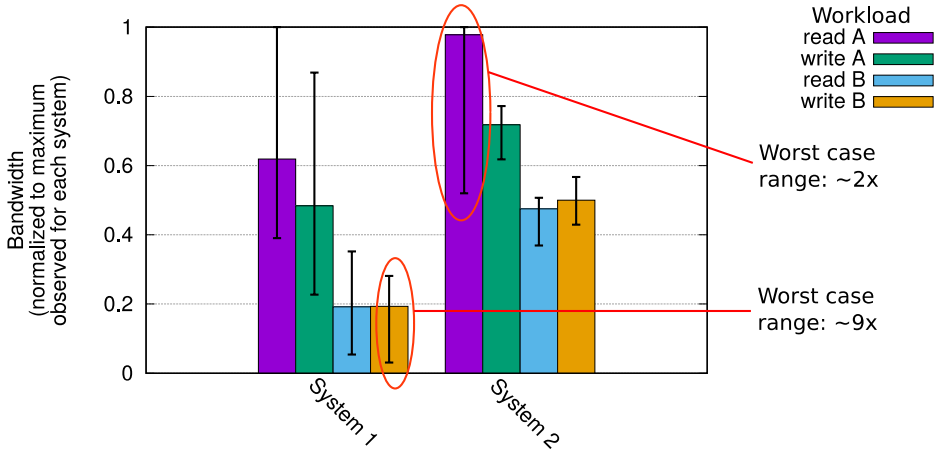
Could users apply a roofline model that incorporates I/O activity?

CHALLENGES IN ORIENTATION

Is the performance “good”, or at least good enough?

Consider the “**I/O system fingerprint**” for two example production systems in figure at right

- Data gathered with standardized, periodic, sampling of diverse workloads
- Produces more than just single scalar number for expected I/O performance (note the user education challenge here)
- Indicates strengths, weaknesses, and susceptibility to variance
- Helps orient expectations, but how do you know which part of the fingerprint is relevant?



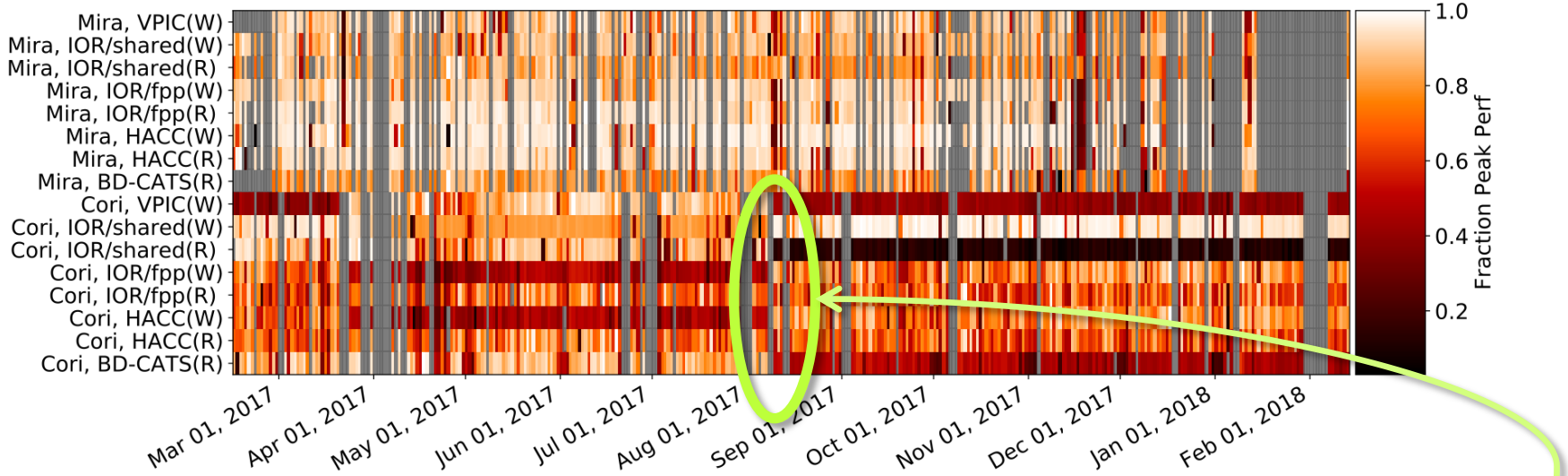
- This preliminary example show the median performance of several workloads over time on two major computing platforms.
- Performance is normalized to the maximum observed rate on each system, to focus on trends rather than absolute throughput.
- Whiskers indicate minimum and maximum sample values.

Glenn Lockwood et al., “A year in the life of a parallel file system”, to appear at SC18

TOKIO

CHALLENGES IN ORIENTATION

System behavior changes, even when expectations do not



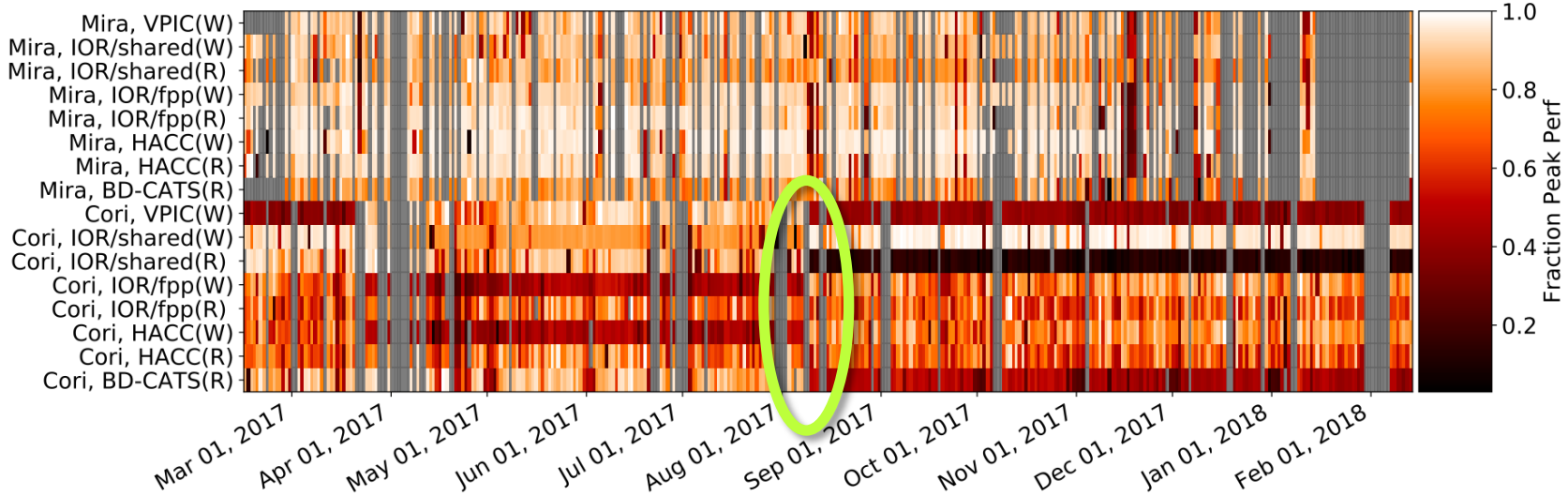
- The above heatmap shows the relative performance of 8 benchmarks on 2 large systems for 1 year: **Light=good, dark=bad, gray=N/A**
- Something changed in August 2017! Better at some things, worse at others

Glenn Lockwood et al., “A year in the life of a parallel file system”, to appear at SC18

TOKIO

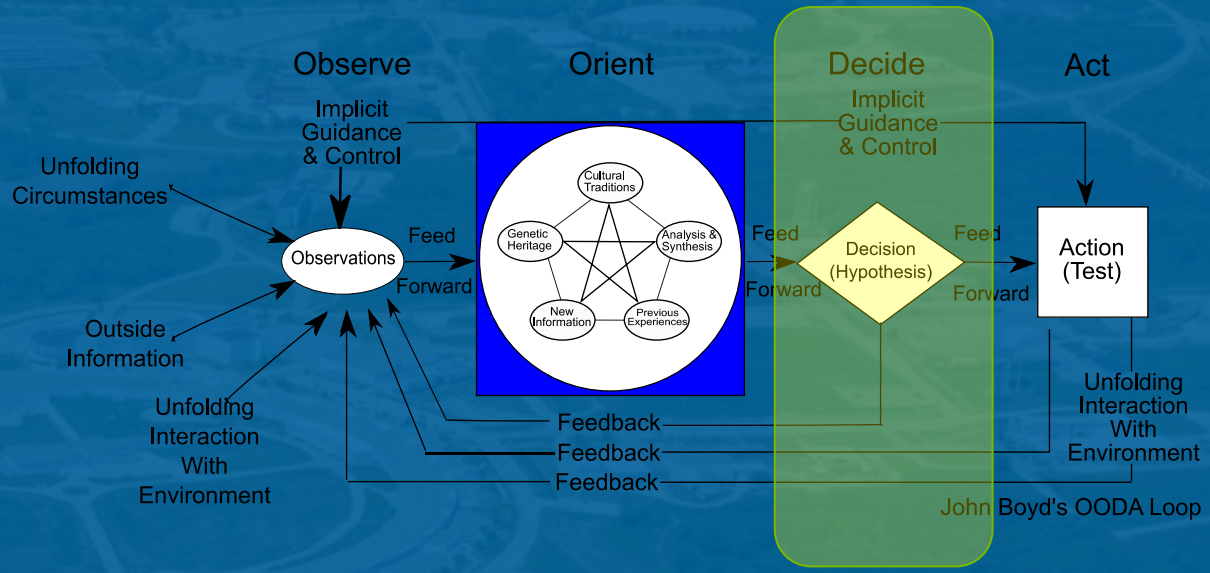
CHALLENGES IN ORIENTATION

System behavior changes, even when expectations do not



- Implications? We must reorient if the system behavior changes.
- Expectations (and more formally, models) don’t necessarily hold forever.
- This example was a sudden change; gradual changes are common as well.

DECIDE



DECIDING WHAT TO DO

- Who has responsibility when we determine that behavior *XYZ* is bad?
  1. **Meet the users where they are**: procure systems optimized for *XYZ*
  2. **Have the users come to you**: change applications to avoid *XYZ*
  3. **Middle ground**: some of both, system tuning and application tuning

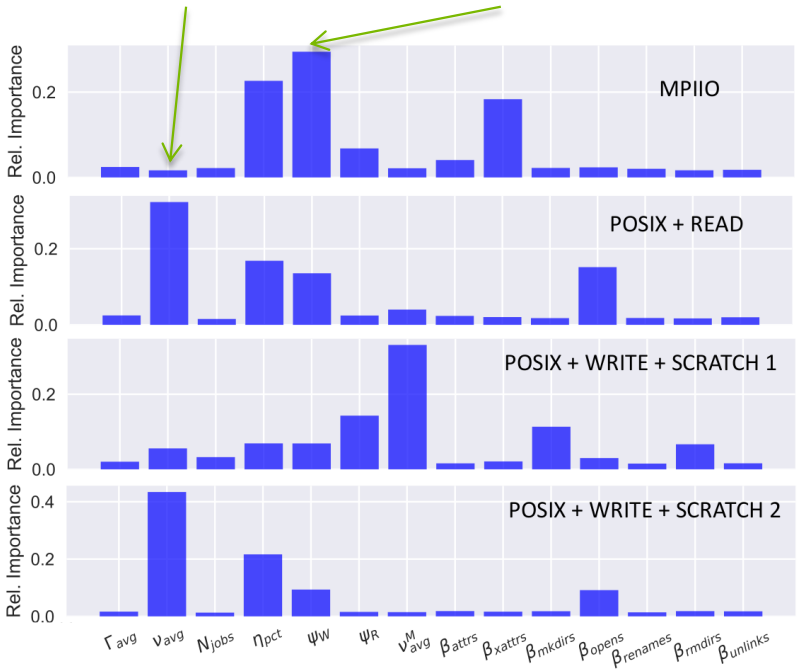
There's an interesting causality problem here: are users selecting the I/O strategy that they want (in many cases enabled by high level libraries), or the I/O strategy that they think the system can handle?

- Decision process
  - Ideally not just ad-hoc or based on expert intuition
  - Follow previous successful patterns
  - Even better, use I/O models to predict the impact of changes

CHALLENGES IN DECISION

It's not easy to project impact

- Can we automate this?
  - E.g. machine learning based performance models
- Different applications, even on the same system, have different responses to system and tuning parameters
- Figure at right shows the relative feature importance for 14 parameters in 4 application classes on an example system
- Some sensitive to metadata, background I/O, etc. to varying degrees



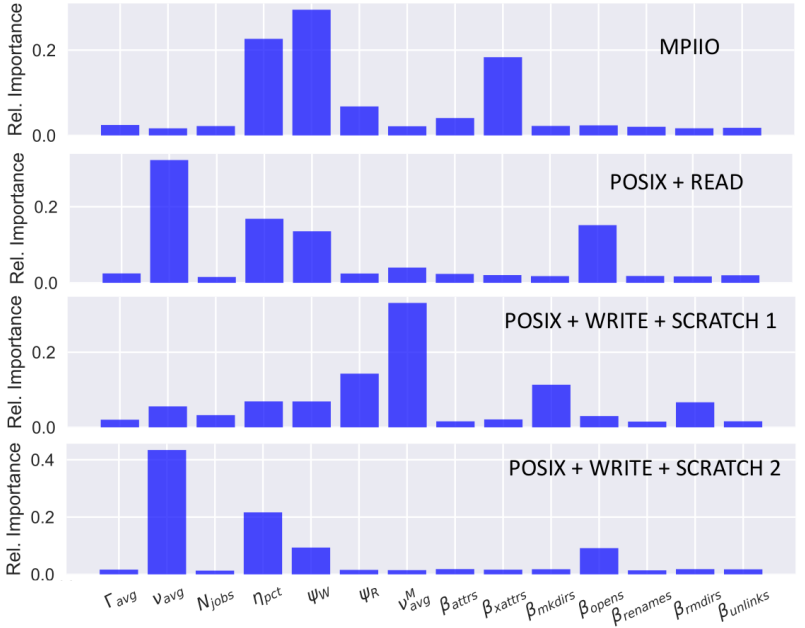
Sandeep Madireddy et al., "Machine Learning Based Parallel I/O Predictive Modeling: A Case Study on Lustre File Systems", ISC 2018.

RAPIDS

CHALLENGES IN DECISION

- Interactions between components are subtle
- Interactions between components differ for each application

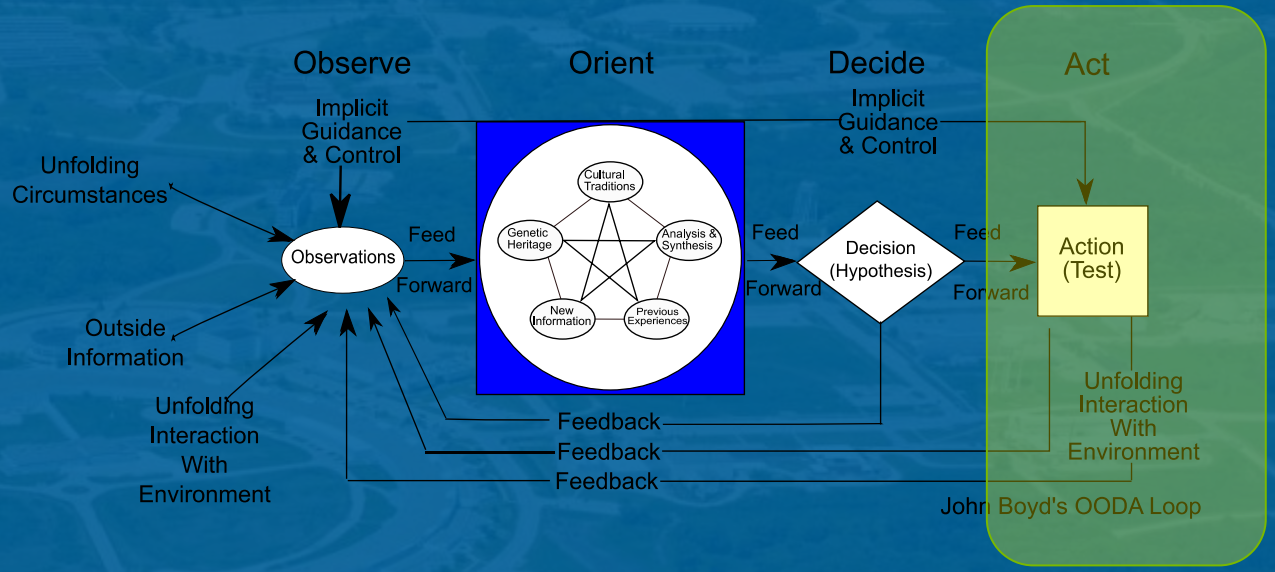
- It may be difficult to predict the impact of a tuning change



Sandeep Madireddy et al., “Machine Learning Based Parallel I/O Predictive Modeling: A Case Study on Lustre File Systems”, ISC 2018.

RAPIDS

ACT

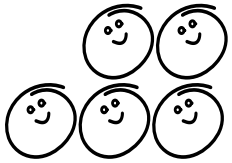


ACTING ON THE DECISION

- After observing, orienting, and deciding, it is time to act!
- Constraints: some theoretical optimizations cannot be implemented in practice
  - Application may have very good reason for their I/O strategy
  - Production considerations can limit the range of practical optimizations
  - Risk of compromising portability for users
- How can we be agile with our I/O strategies?
  - Tunable software, especially across layers
- One step further: customizable data services
  - Create data services that are tailored to the task at hand by composing reusable building blocks in different ways
  - This approach is easier if in-system storage resources are available

CHALLENGES IN ACTION

How to provide storage solutions that suit the problem



- Different applications have different data models and requirements
  - Interface/bindings
  - Provisioning
  - Semantics
  - Resilience
  - Coherence

This calls for more flexibility than is typically offered by a parallel file system!

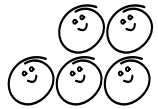
- Fortunately, they have many underlying building blocks in common:
  - Network transport
  - Local storage abstractions
  - Concurrency model
  - Meta services like group membership and telemetry
  - Placement algorithms

Argonne NATIONAL LABORATORY

Los Alamos NATIONAL LABORATORY

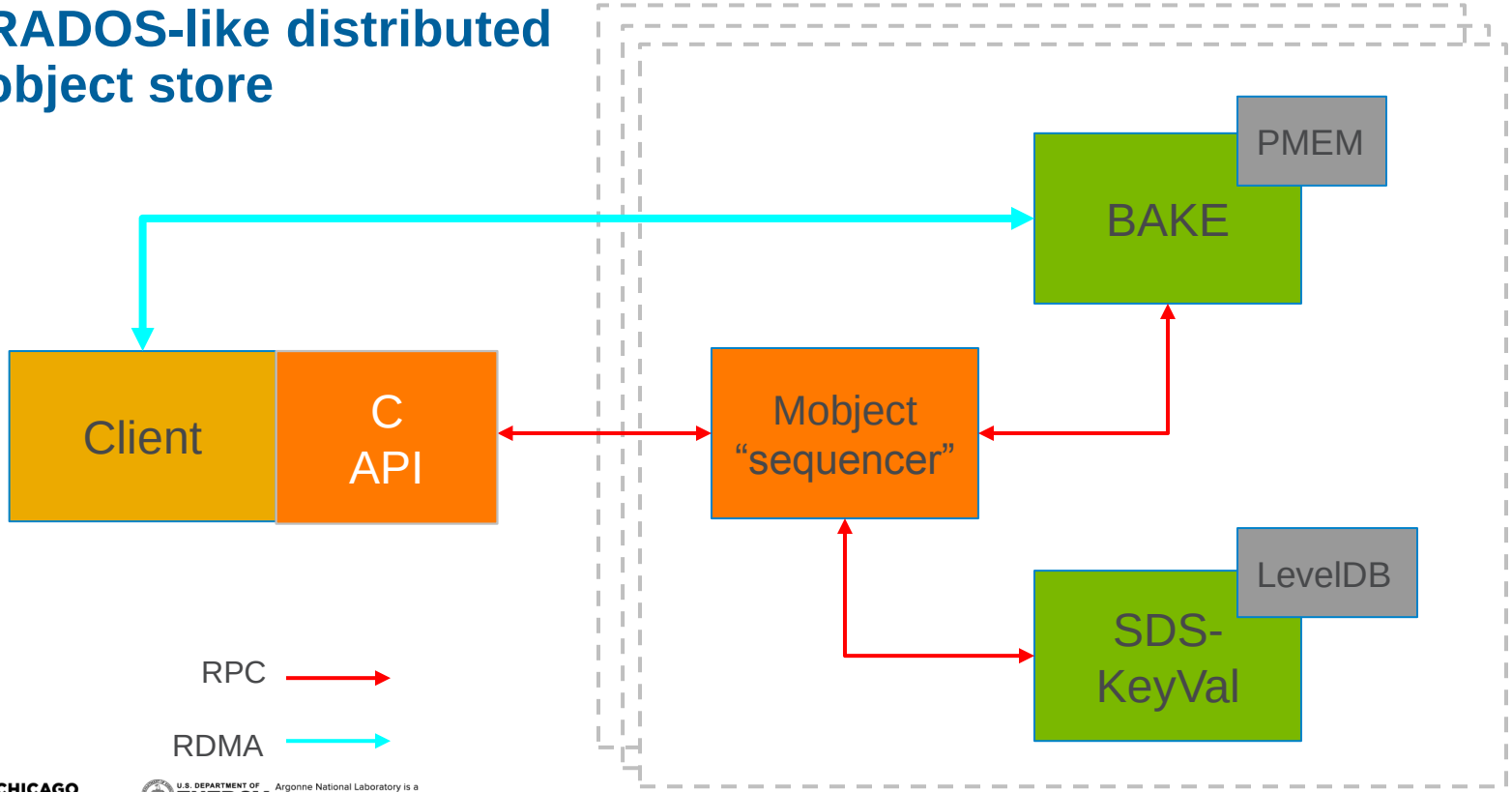
Carnegie Mellon University

The HDF Group

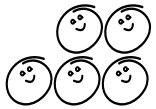


EXAMPLE: MOBJECT

RADOS-like distributed object store

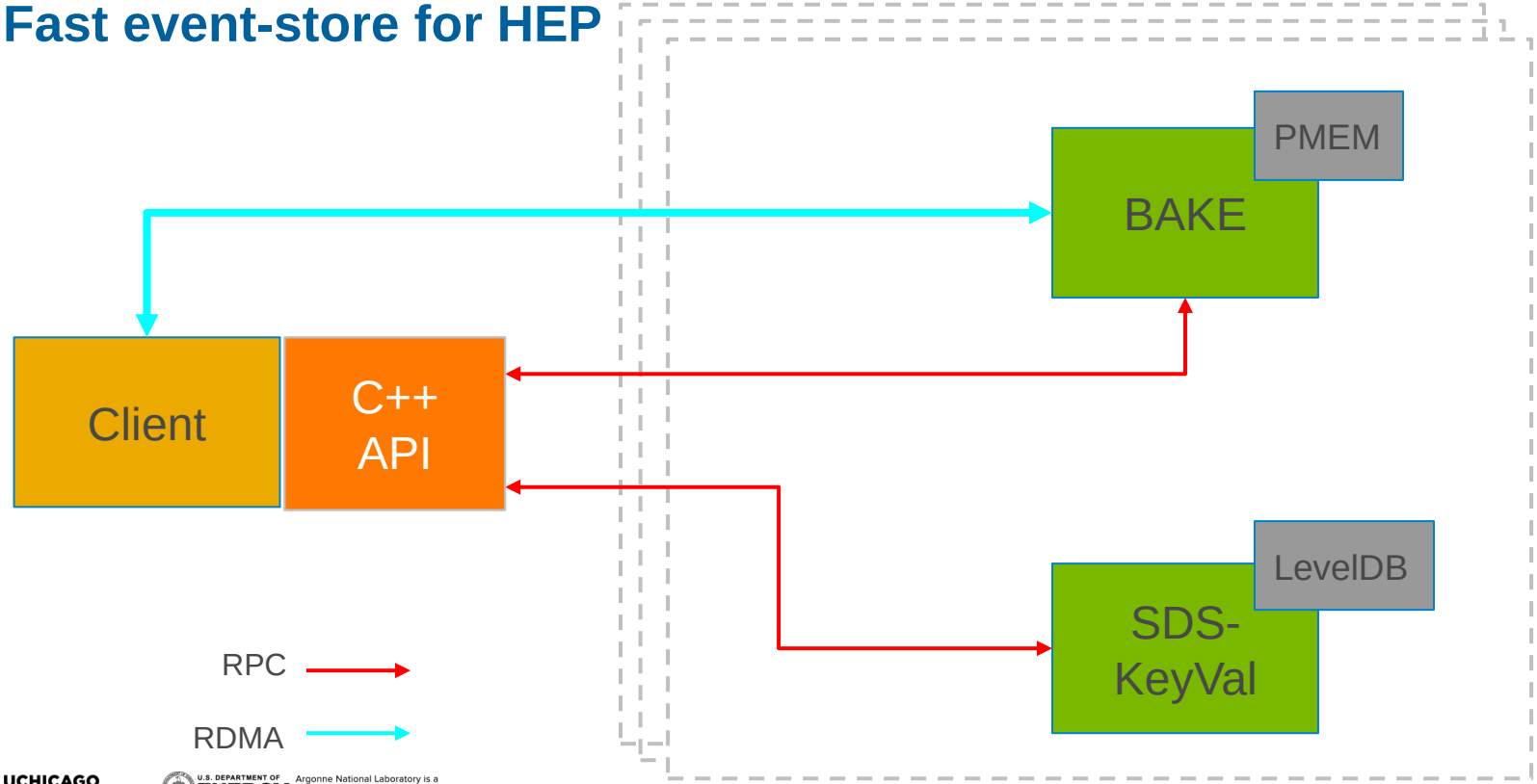


Figures provided by Matthieu Dorier



EXAMPLE: HEPNOS

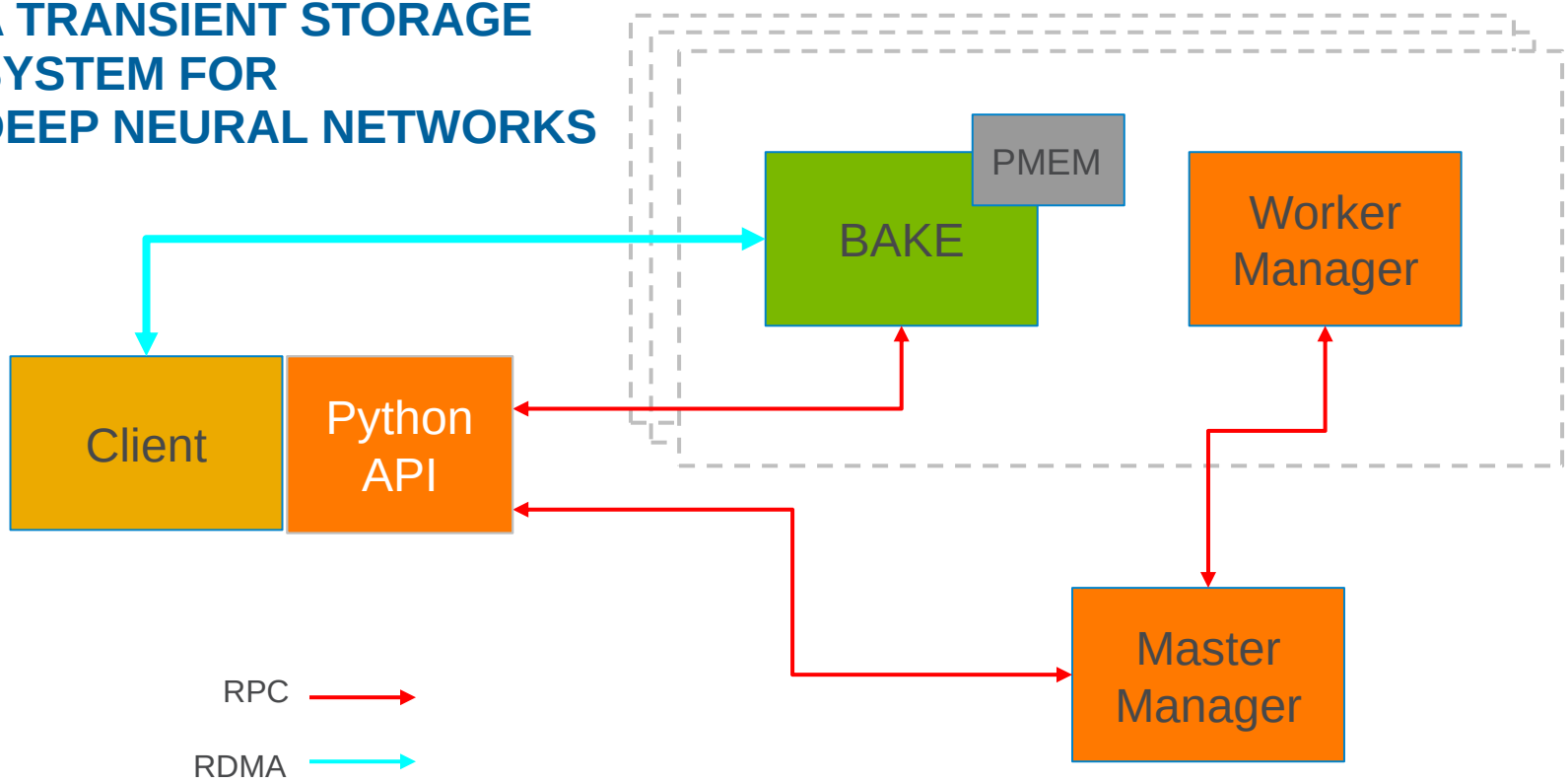
Fast event-store for HEP



Figures provided by Matthieu Dorier

EXAMPLE: FLAMESTORE

A TRANSIENT STORAGE SYSTEM FOR DEEP NEURAL NETWORKS

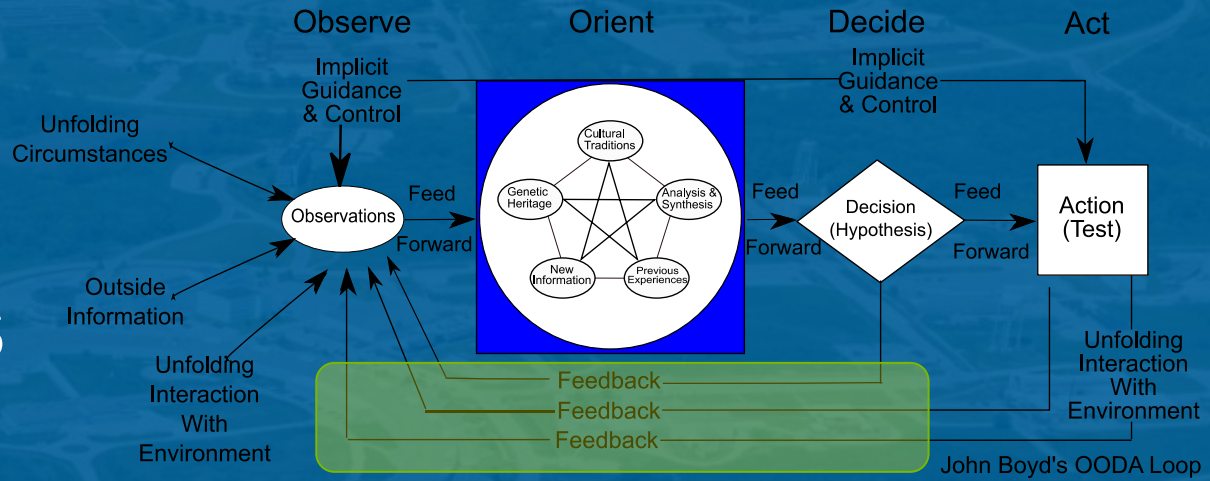


CHALLENGES IN ACTION

- The previous examples all used the same building blocks
  - But each service was customized to accommodate different usage models, data models, and expected workloads
- The building blocks are there, but determining how to best deploy and provision fully customized data services is still a manual process

- More generally: we can rethink what can be done in file systems, middleware, and customized services

PUTTING THE LOOP IN OODA LOOPS



DON’T FORGET THE LOOP

- There is no one point at which tuning is “done”
- Must continue to observe, integrate feedback, and iterate
- The UMAMI tool from the TOKIO project is one example of how to do this

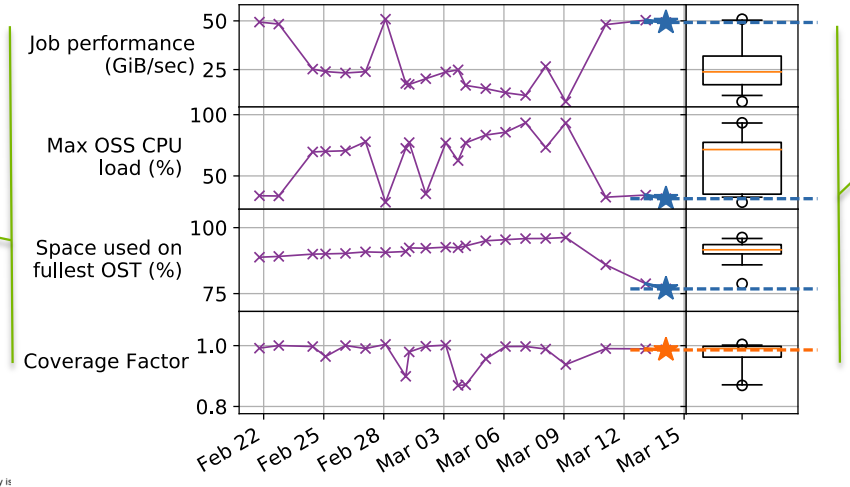
TOKIO

UMAMI

TOKIO Unified Measurements And Metrics Interface

- UMAMI is a pluggable dashboard that displays the I/O performance of an application in context with system telemetry and historical records

Historical samples (for a given application) are plotted over time



Each metric is shown in a separate row

Box plots relate current values to overall variance

Glenn Lockwood et al., “Umami: a recipe for generating meaningful metrics through holistic I/O performance analysis”, PDSW 2017

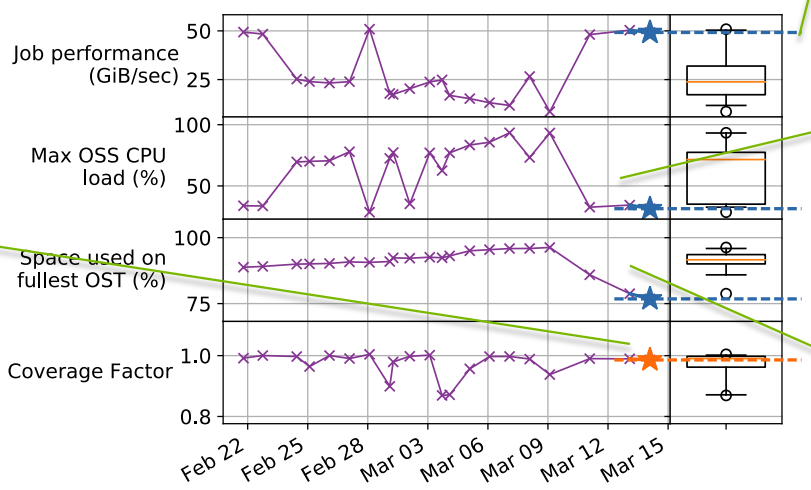
TOKIO

UMAMI

TOKIO Unified Measurements And Metrics Interface

- Broader contextual clues simplify interpretation of unusual performance measurements

Performance for this job is higher than usual



Server CPU load is low after a long-term steady climb

System background load is typical

Corresponds to data purge that freed up disk blocks

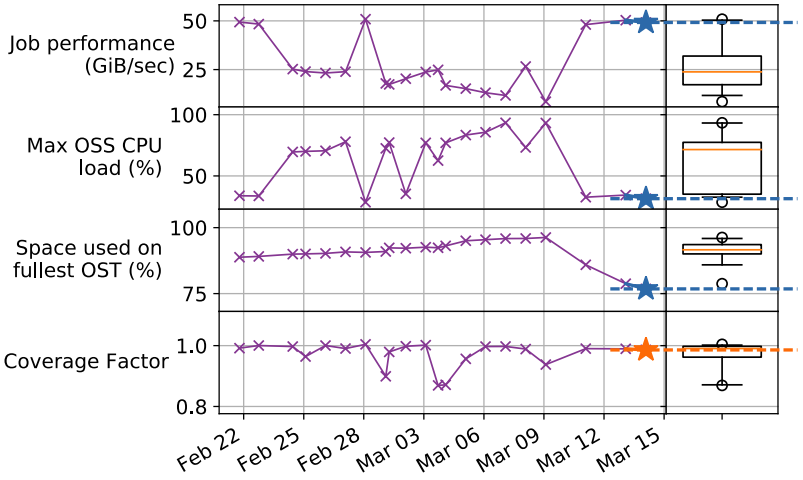
Glenn Lockwood et al., “Umami: a recipe for generating meaningful metrics through holistic I/O performance analysis”, PDSW 2017

TOKIO

UMAMI

TOKIO Unified Measurements And Metrics Interface

- UMAMI still requires some level of expert interpretation
- Can we automate the analysis?



WRAP UP

COMMENTARY ON UNDERSTANDING AND TUNING HPC I/O

How hard can it be? Pretty hard!

- Simply monitoring/instrumenting is hard enough by itself, but is not sufficient
  - Must also interpret, decide what to do, and implement improvements
- This presentation explored OODA loops as a way to frame the conversation [*]
- Continuous iteration and engagement with users is crucial
- Evolving platforms and applications present many opportunities!

[*] OODA loops are sometimes taught as part of a tactical strategy to disorient your opponent. Maybe don't apply that aspect of the model to your HPC system?!

Or maybe this explains why your users are breaking your file systems…

RESOURCES

https://www.mcs.anl.gov/research/projects/darshan

https://www.nersc.gov/research-and-development/storage-and-i-o-technologies/tokio/

https://www.mcs.anl.gov/research/projects/mochi

https://rapids.lbl.gov/

This work was supported by the U.S. Department of Energy, Office of Science, Advanced Scientific Computing Research, under Contract DE-AC02-06CH11357.