

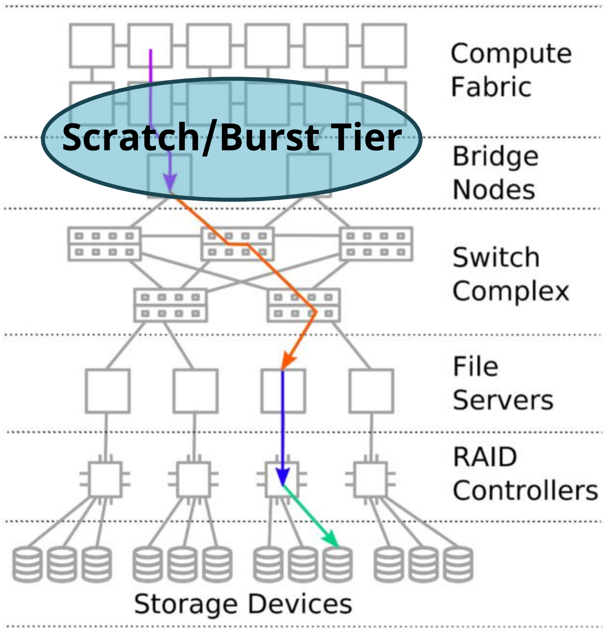
# Building a Future of HPC Performance Engineering Rooted in Data, Tools, and Trust

***Sarah M. Neuwirth***
Johannes Gutenberg University Mainz, Germany
neuwirth@uni-mainz.de

iWAPT 2025 Workshop, IEEE IPDPS, June 2025

# Motivation
## *Heterogeneous and Complex HPC Infrastructures*

- HPC infrastructure *too complex*, humans are *overwhelmed*
- Complexity and scope increase the *urgency*
  - *New computational paradigms* (AI/ML apps vs. BSP-style HPC)
  - *New architectural directions* (e.g., IPU, RISC-V, data flow)
  - *Heterogeneity overall*: node architectures, within the system, storage and parallel file system during application design (e.g., ML within HPC applications)
  - *New operations paradigms* (e.g., cloud, container)
  - Simplistic approaches to increasing compute demand result in *unacceptable power costs*
- Difficult for humans to optimally adapt applications to systems and to detect and diagnose vulnerabilities





Carns, P., 2023. *HPC Storage: Adapting to Change*. Keynote at REX-IO'23 Workshop.

Ciorba, F., 2023. *Revolutionizing HPC Operations and Research*. Keynote at HPCMASPA'23 Workshop.

B. Settlemyer, G. Amvrosiadis, P. Carns and R. Ross, 2021. *It's Time to Talk About HPC Storage: Perspectives on the Past and Future*, in Computing in Science & Engineering, vol. 23, no. 6, pp. 63-68.

# Vision: From Data to Decisions

*Holistic Monitoring for Intelligent HPC Operations*

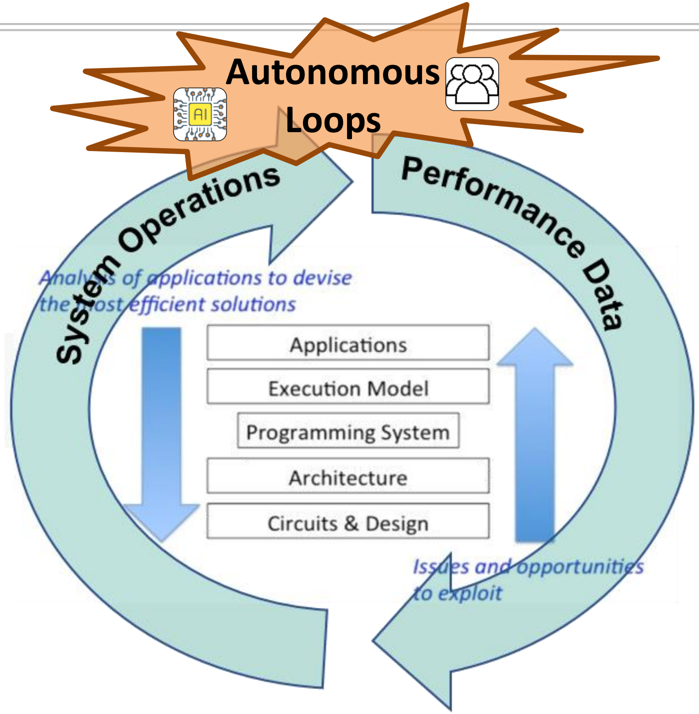
**What We Need:**

- Continuous monitoring, archiving, and analysis of operational + performance data
- Unified visibility into applications, system software, and hardware layers

**Why It Matters:**

- Enables automated feedback loops using AI/ML
- Supports dynamic workload & architecture analysis
- Powers adaptive, actionable responses

**Goal:** Efficient, explainable HPC operations driven by *autonomous analyze-feedback-response loops*





Gentile, A., 2021. *Enabling Application and System Data Fusion*. Keynote at MODA'21 Workshop.

Ciorba, F., 2023. *Revolutionizing HPC Operations and Research*. Keynote at HPCMASPA'23 Workshop.

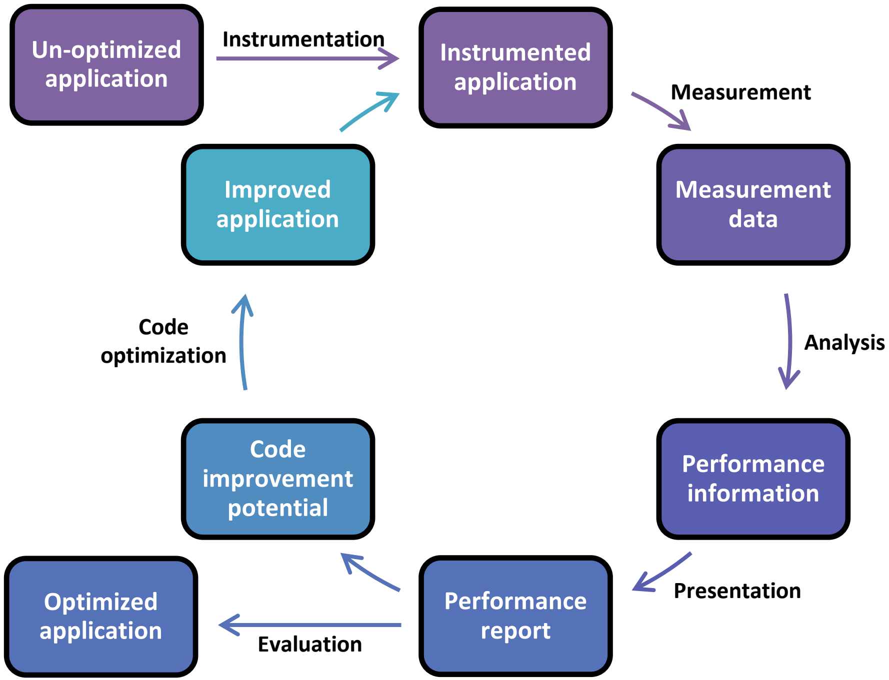
Dagstuhl Seminar 23171, 2023. *Driving HPC Operations With Holistic Monitoring and Operational Data Analytics*. https://www.dagstuhl.de/23171

# Diagnosing the Performance Trust Gap

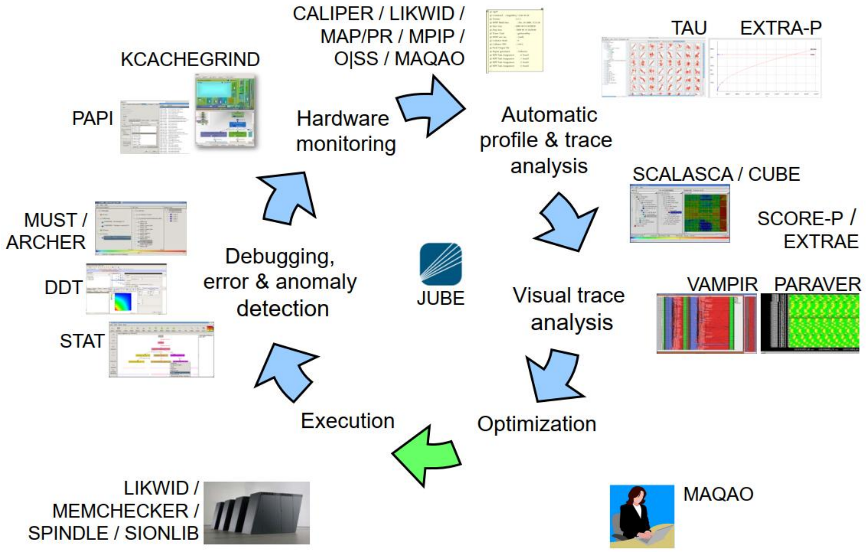
*Why current methods fall short*

# Diagnosing the Performance Trust Gap

## *The Traditional Performance Optimization Cycle*




### Example: Overview of the VI-HPS Tools

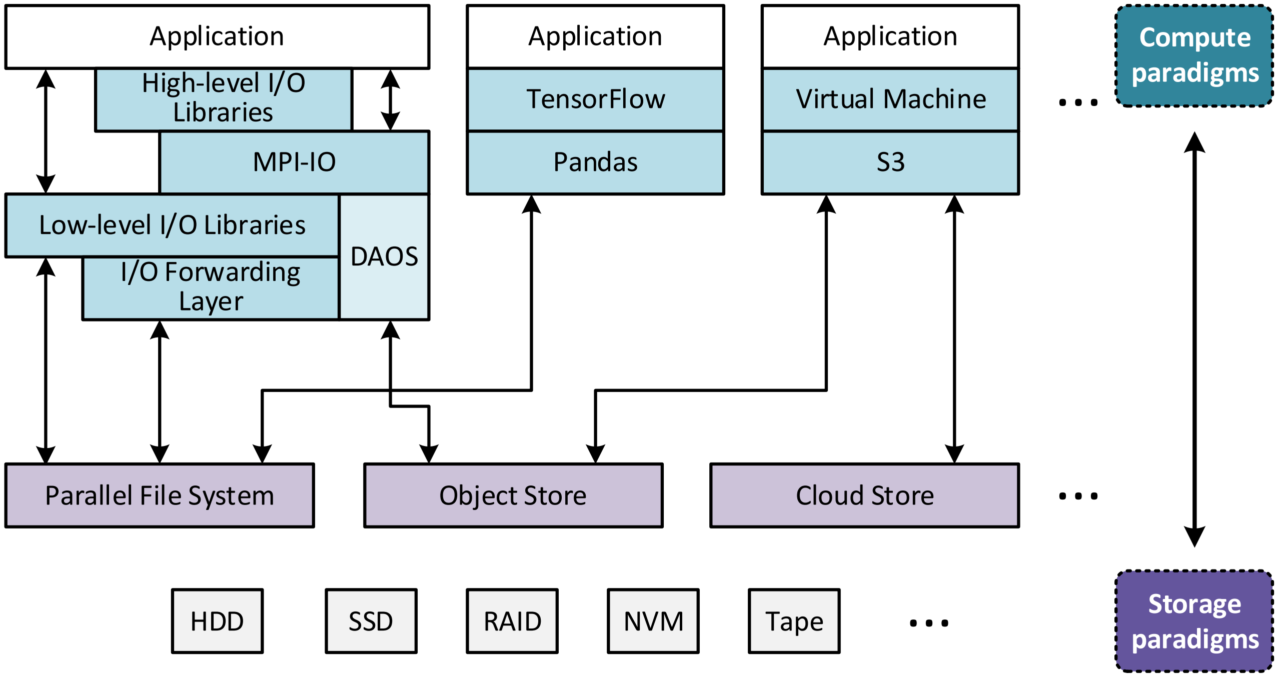



Performance Engineering Overview, https://doc.zih.tu-dresden.de/software/performance_engineering_overview/

Virtual Institute – High Productivity Supercomputing (VI-HPS), https://www.vi-hps.org/tools/tools.html

# Diagnosing the Performance Trust Gap

*Example: Parallel I/O Stack and its Performance Factors*




**Application**

- Number of processes
- Request sizes
- Access patterns
- I/O operation
- Data volume

**Network**

- Message sizes
- Network topology
- Network paths
- Network type

**File System**

- Type of file system
- Disk types
- Stripe sizes
- File hierarchy
- Shared access

# Diagnosing the Performance Trust Gap
*Example: Status of I/O Characterization Tools*

**Blue Waters, Mira, and Theta popular Darshan log sources used for research:**

- https://bluewaters.ncsa.illinois.edu/data-sets
- https://reports.alcf.anl.gov/data/
- ftp://ftp.mcs.anl.gov/pub/darshan/data

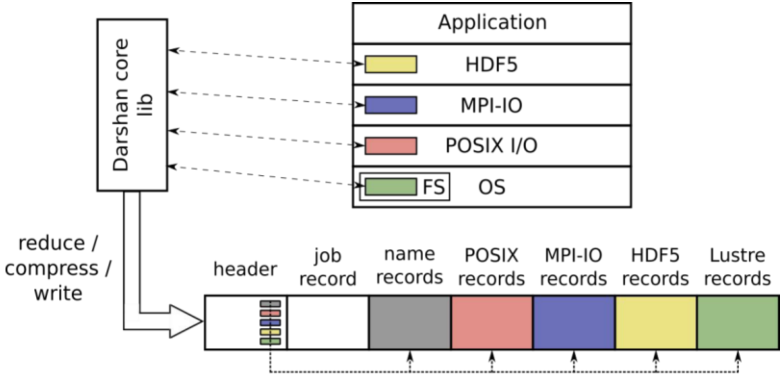
**Open questions:**

- How relevant are the logs to current systems?
- How do we know the integrity of the logs?

**Community comments:**

- "Darshan is one of the first tools to be deactivated in the event of I/O problems."
- "Darshan cannot grasp the complexity of state-of-the-art parallel storage systems."

**Darshan I/O Characterization Tool**




Snyder, S., 2022. *Darshan: Enabling Insights into HPC I/O Behavior*. ECP Community BoF Days.

**What are the implications of these questions and observations?**

# Diagnosing the Performance Trust Gap

## *Fragile Pipelines: Tool-Driven, Not Insight-Driven*

| Category | Examples | Srengths | Limitations |
|---|---|---|---|
| ***Application-level*** | Darshan, Recorder, Score-P | Fine-grained function tracing | No visibility into system-wide interactions |
| ***System-level*** | LDMS, DCDB, TACCStats | Aggregated I/O performance metrics | Cannot correlate application performance with system metrics |
| ***End-to-end*** | Ganglia, Nagios, Apollo | Holistic view of system utilization | Lacks deep profiling at kernel and network levels |

- **Siloed Tool Views:** *Each tool sees a layer. None explain the whole system.*
  => In case of I/O: App-level profilers (e.g., Darshan) vs. system tools (LDMS, DCDB)
- **Workflow Scripts Instead of Workflows:** *Custom scripts per experiment = unscalable, unrepeatable.*
  => Benchmarking tools (e.g., iperf, sockperf) often require client/server logic incompatible with SLURM
- **Discarded Insights:** *Performance data is ephemeral; models are not reused.*
  => No structure for reuse → repeated effort, lost opportunities

# Diagnosing the Performance Trust Gap

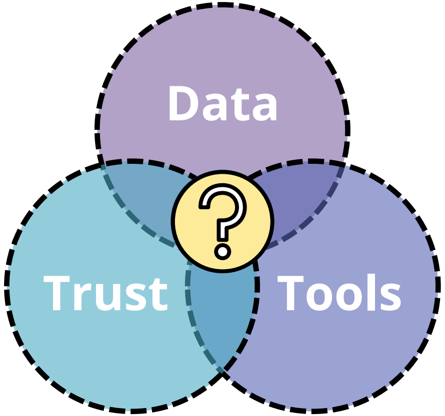
*Why We Need to Rethink HPC Performance Engineering*

- Heterogeneous Architectures
- Parallelism and Scalability
- Diversity of Workloads
- Dynamic Resource Allocation
- Software Stack Complexity
- Large-scale Data Movement
- Benchmark Suitability

Reproducibility

Portability

Comparability



# Reproducible, Tool-Agnostic Workflows for Performance Insight

*From ad hoc pipelines to robust, model-aware workflows*

# Reproducible, Tool-Agnostic Workflows

*Goal: Holistic & Automated Monitoring and Analysis Cycle*

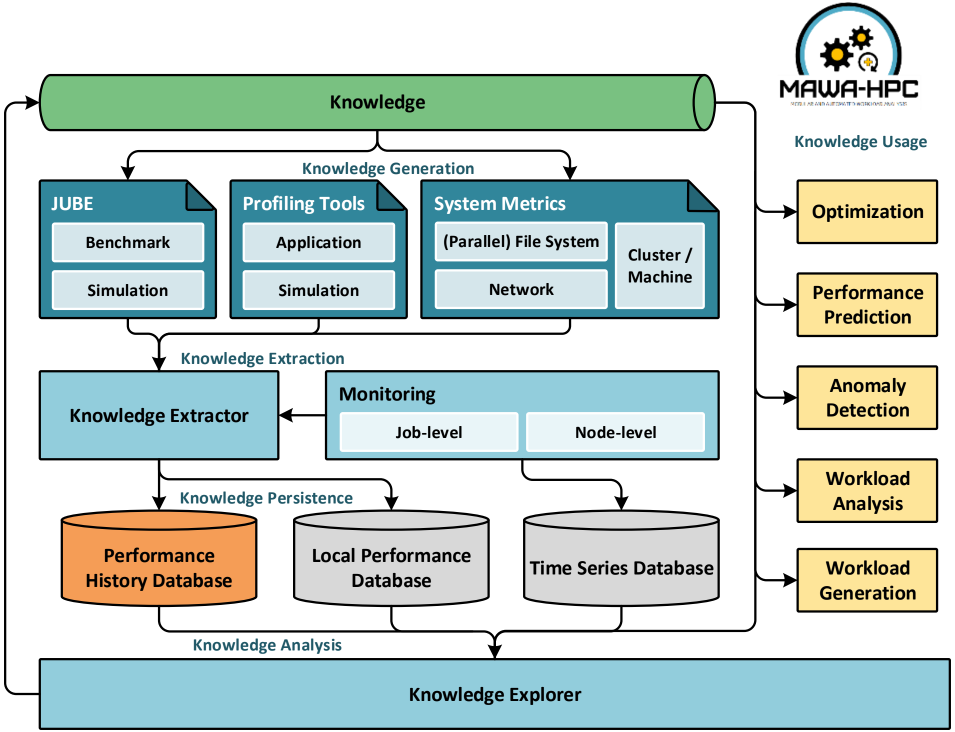
***Idea:*** Develop and implement standardized and tool-independent approach for HPC workload and application analysis

***Goal:*** Establish a ***performance history database*** to categorize systems, workload behaviors, and characteristic patterns for different science domains


Zhu, Z., Bartelheimer, N. and Neuwirth, S., 2023. *MAWA-HPC: Modular and Automated Workload Analysis for HPC Systems*. ISC'23.

Bartelheimer, N., Zhu, Z., and Neuwirth, S., 2023. *Toward a Modular Workflow for Network Performance Characterization*. IPDPSW'23.

Zhu, Z., and Neuwirth, S., 2023. *Characterization of Large-Scale HPC Workloads With Non-Naïve I/O Roofline Modeling and Scoring.* ICPADS'23.

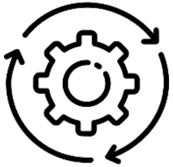



# Reproducible, Tool-Agnostic Workflows

*Reproducible Benchmarking and Measurement*

- ***Benchmarking:*** Process of comparing system performance using standardized tests and metrics.
- ***Reproducibility:*** Ability to obtain the same results with the same system and test conditions.
- ***Importance of Reproducibility:***
  - *Consistency:* Enables fair and accurate comparisons between systems
  - *Confidence:* Trust in benchmark results for decision-making
  - *Research Validity:* Essential for scientific studies and product evaluations


- ***Key Principles of Reproducible Benchmarking:***
  - *Documentation:* Record hardware and software configurations, test settings, and data
  - *Version Control:* Maintain consistent test suites and tools
  - *Automation:* Minimize human error by automating test execution
  - *Standardization:* Use industry-standard benchmarks and metrics
  - *Multiple Runs:* Conduct tests multiple times to verify results



# Reproducible, Tool-Agnostic Workflows

## *Reproducible Benchmarking: Integration is Key*

### Repeatability

- Ensures consistent results from the same setup
- Builds trust in computational outcomes
- Forms the basis for scientific validation

Schifrin, A., 2023. *Automated Performance Characterization of HPC Systems.* Bachelor thesis, Goethe University Frankfurt.

### Comparability

- Standardizes benchmarking across models or methods
- Allows fair evaluation of new approaches
- Helps identify performance trade-offs

Bartelheimer, N. and Neuwirth, S., 2023. *Toward Reproducible Benchmarking of PGAS and MPI Communication Schemes.* ICPADS'23.

### Modularity

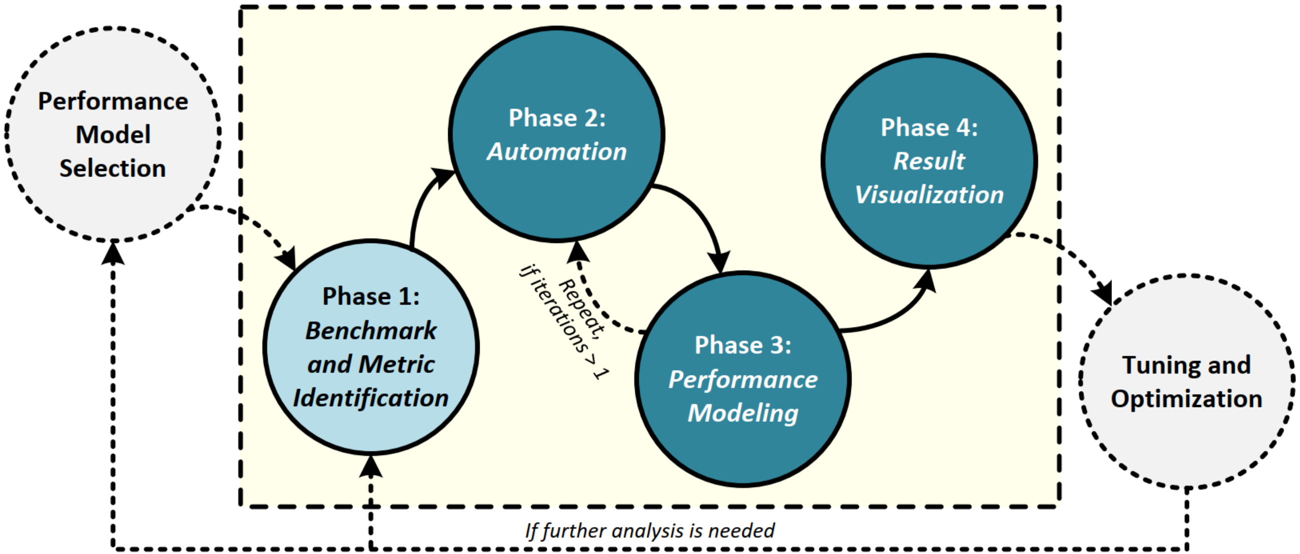
- Enables flexible software architecture
- Facilitates integration of AI/ML components
- Supports code reuse and maintainability

Zhu, Z., Wang, C. and Neuwirth, S., 2025. *Advancing HPC Performance Modeling with an Interactive, Automated and Tool-Agnostic ML-Driven Workflow*. SSDBM'25. (to appear)

# Reproducible, Tool-Agnostic Workflows

*Workflow Design for Reproducible Benchmarking*

Core component of the benchmarking framework is the **JUBE Benchmarking Environment**




```
config
├ platform
│ ├ fuchs
│ │ ├ platform.xml
│ │ └ platform-likwid-specs.xml
│ └ cesari
│   ├ platform.xml
│   └ ...
├ likwid-benchmark.xml
├ likwid-specs.xml
├ qperf-benchmark.xml
├ qperf-specs.xml
├ ...
├ likwid.run.in
└ ...
```

Bartelheimer, N., Zhu, Z., and Neuwirth, S., 2024. *Automated Network Performance Characterization for HPC Systems*. International Journal of Networking and Computing.

JUBE Documentation: [https://apps.fz-juelich.de/jsc/jube/jube2/docu/index.html](https://apps.fz-juelich.de/jsc/jube/jube2/docu/index.html)

# Reproducible, Tool-Agnostic Workflows

*Reproducible Benchmarking: Example Configuration*

**Benchmark-independent, Platform-specific**

`platform.xml`

```xml
<parameterset name="systemParameter">
    <parameter name="nodes" type="int">2</parameter>
    <parameter name="taskspernode" type="int">1</parameter>
    <parameter name="threadspertask" type="int">1</parameter>
    <parameter name="tasks" mode="python" type="int">$nodes * $taskspernode</parameter>
    <parameter name="timelimit">00:30:00</parameter>
</parameterset>
```

```xml
<parameterset name="executeset">
    <parameter name="submit">sbatch</parameter>
    <parameter name="submit_script">submit.job</parameter>
    <parameter name="starter">srun</parameter>
</parameterset>
```

**Benchmark-specific, Platform-independent**

`likwid-specs.xml`

```xml
<parameterset name="copy_params">
    <parameter name="benchmark_set_cp" tag="copy">        copy</parameter>
    <parameter name="benchmark_set_cp" tag="copy_avx">    copy_avx</parameter>
    <parameter name="benchmark_set_cp" tag="copy_avx512">copy_avx512</parameter>
</parameterset>
```

```xml
<parameterset name="alloc_params">
    <parameter name="alloc" tag="W">-W</parameter>
    <parameter name="alloc" tag="w">-w</parameter>
</parameterset>
```

```xml
<patternset name="likwid_pattern">
        <pattern name="likwid_cycles" type="int" dotall="False" mode="pattern">Cycles:\s+$jube_pat_int</pattern>
        <pattern name="likwid_cpu_clock" type="int" dotall="False" mode="pattern">CPU Clock:\s+$jube_pat_int</pattern>
</patternset>
```

**Benchmark-specific, Platform-specific**

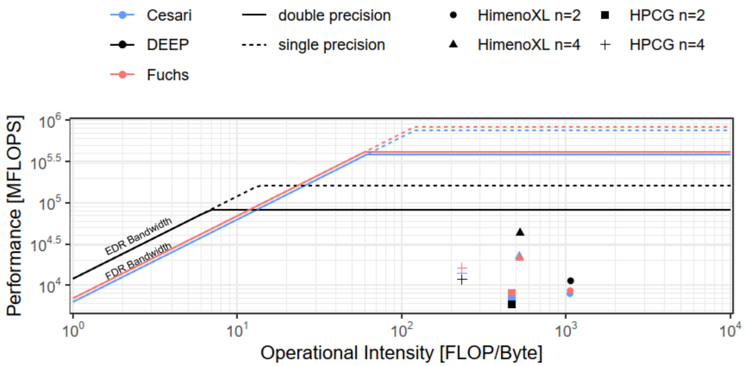
`platform-likwid-specs.xml`

```xml
<parameterset name="thread_domain_params">
    <parameter name="thread_domain" tag="td-n">  N</parameter>
    <parameter name="thread_domain" tag="td-s0">S0</parameter>
    <parameter name="thread_domain" tag="td-s1">S1</parameter>
    <parameter name="thread_domain" tag="td-d0">D0</parameter>
    <parameter name="thread_domain" tag="td-d1">D1</parameter>
    <parameter name="thread_domain" tag="td-c0">C0</parameter>
    <parameter name="thread_domain" tag="td-c1">C1</parameter>
    <parameter name="thread_domain" tag="td-m0">M0</parameter>
    <parameter name="thread_domain" tag="td-m1">M1</parameter>
</parameterset>
```

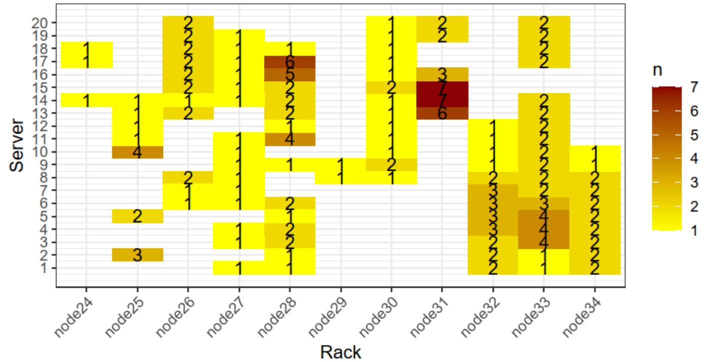
```xml
<parameterset name="suffix_params">
    <!--
        CHANGEME: this parameter can be changed and represents the parameter part after the <size>
        if not empty, the suffix must start with the ":" (colon) character
    -->
    <parameter name="suffix"></parameter>
</parameterset>
```

# Reproducible, Tool-Agnostic Workflows

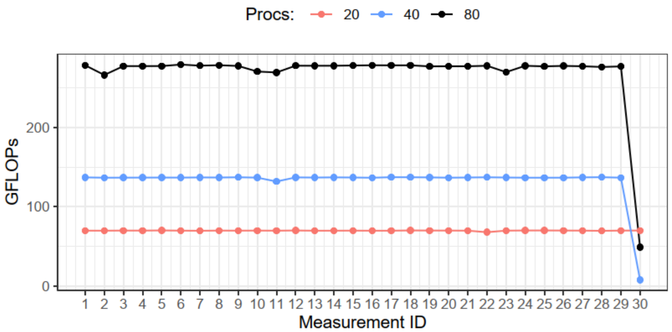
*Example: Automated Performance Characterization*



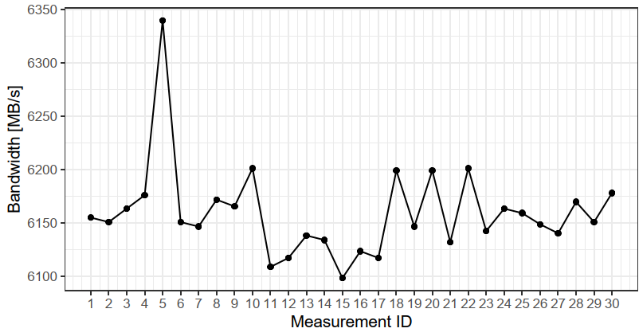

**Roofline model for Himeno and HPCG results.**




**Heat map of the allocated nodes (overall benchmark runs).**




**Himeno benchmark over 15 days / 2 measurements per day.**




**RDMA point-to-point performance over 15 days / 2 measurements per day.**

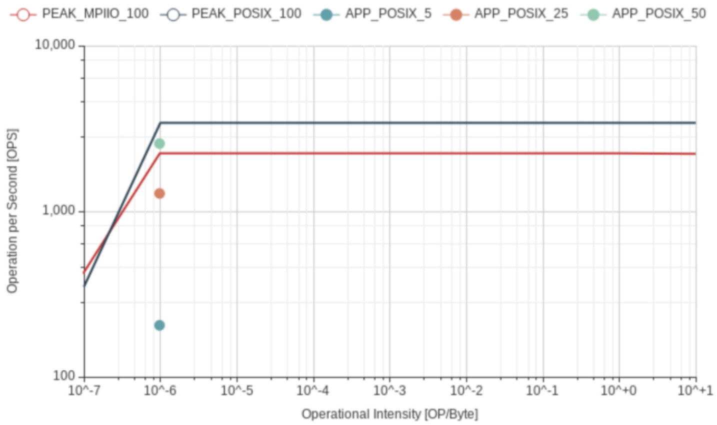
# Reproducible, Tool-Agnostic Workflows

## *I/O Roofline Characterization: Initial Concept*

Zhu, Z., Bartelheimer, N. and Neuwirth, S., 2023. *An Empirical Roofline Model for Extreme-Scale I/O Workload Analysis.* IPDPSW'23.

Zhu, Z., and Neuwirth, S., 2023. *Characterization of Large-Scale HPC Workloads With Non-Naïve I/O Roofline Modeling and Scoring.* ICPADS'23.

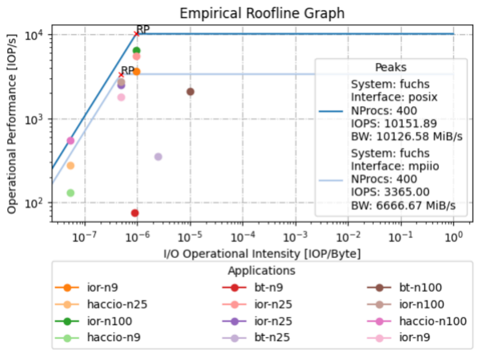
- Traditional Roofline Model…
  - is based on looking at the *relationship between work and traffic*
  - provides intuitive approach through simple bound and bottleneck analysis
- ***I/O Roofline Model*** is based on IOPS and the I/O bandwidth
  => *I/O interface specific*, i.e., POSIX, MPIIO, etc.
  - *IOPS*: number of reads and writes that a storage system can perform per second
  - *Bandwidth*: total amount of data read or written per second
- **X-axis:** *I/O Operational Intensity* $= \frac{\text{Total I/O Operations}}{\text{Read Bytes+Write Bytes}}$
- **Y-axis:** $P = \min(\text{Peak IOPS}, \text{Peak I/O Bandwidth} \times \text{I/O Intensity})$
  where $P$ is the attainable perf., $P_{peak}$ is the peak perf., $b$ is the peak bandwidth, and $I$ is the arithmetic intensity

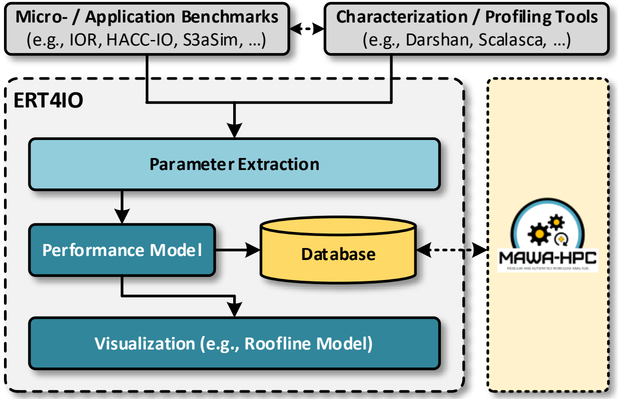


# Reproducible, Tool-Agnostic Workflows

## *I/O Roofline Characterization: Workflow Implementation*

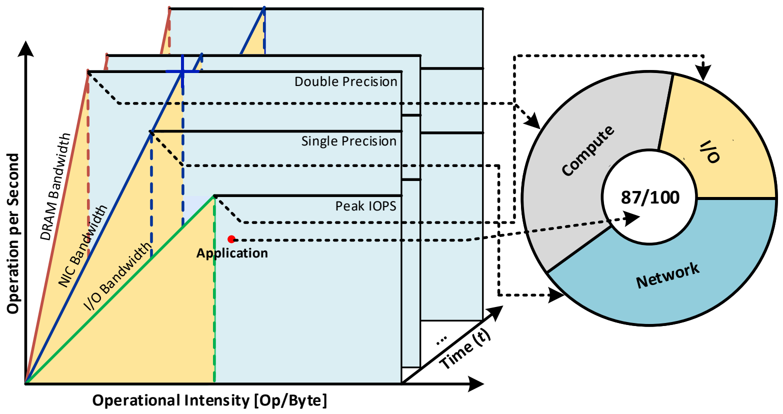
- *ERT4IO (Empirical Roofline Tool for I/O)* provides automated I/O Roofline characterization
  - Parameter extraction from Darshan logs
  - Generates Roofline visualization
- Forwards results to MAWA-HPC framework
  - Enables further data analysis
  - Preserves and shares knowledge via performance history database with the HPC community
- Applicable to various use cases
  - Systems with different configurations and hardware can be compared and evaluated
  - Intuitive estimation of an application's I/O performance
  - Identifying performance bottlenecks and anomalies



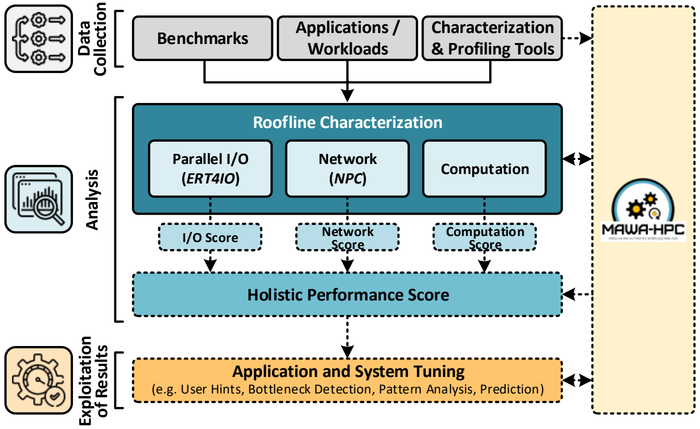
# Reproducible, Tool-Agnostic Workflows
## *Multi-dimensional Performance Modeling (WiP)*

- *Goal:* provide a comprehensive view of application and system performance ⇨ *emerging workloads*
- Multi-dimensional performance models, for example Roofline model, to account for multiple performance factors (e.g. network, compute power, and parallel I/O)
- Including time as an additional dimension, the Roofline model can provide insight into an application's performance over time, enabling the identification of performance anomalies




Zhu, Z., Bartelheimer, N. and Neuwirth, S., 2023. *MAWA-HPC: Modular and Automated Workload Analysis for HPC Systems*. ISC'23.

Bartelheimer, N., Zhu, Z., and Neuwirth, S., 2023. *Toward a Modular Workflow for Network Performance Characterization*. IPDPSW'23.

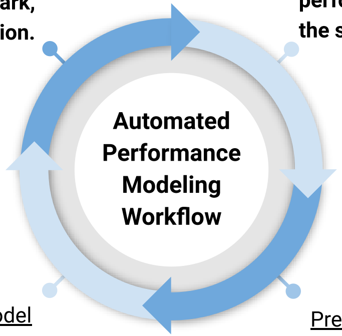


# Reproducible, Tool-Agnostic Workflows
## *Automated ML-Driven Performance Modeling Workflow*

Zhu, Z. and Neuwirth, S., 2024. *Interactive and Tool-Agnostic ML-Driven Workflow for Automated HPC Performance Modeling*. SC'24.

<u>Experiment Design</u>
**Design the experiment and set up the application, benchmark, or simulation.**

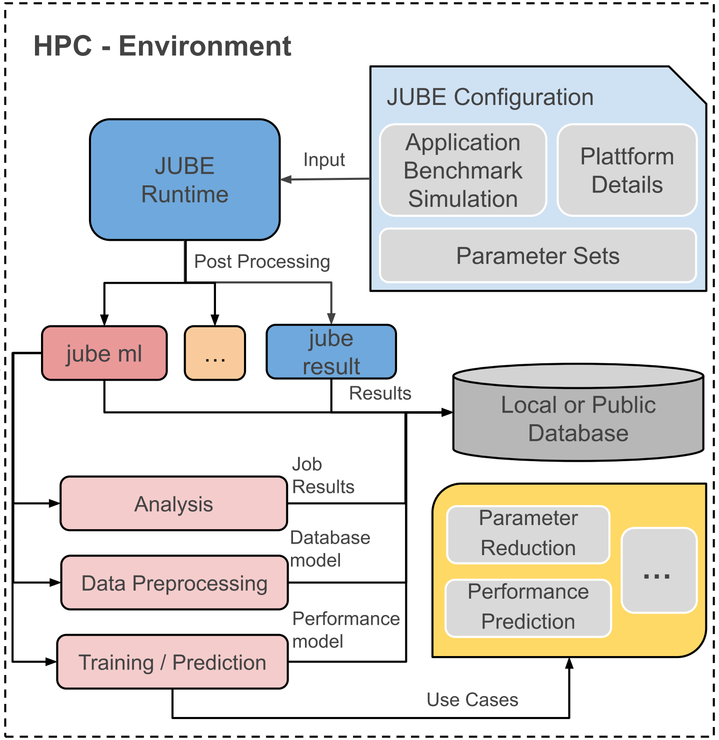
<u>Measurements</u>
**Run the experiment and collect and analyze performance results from the simulation.**




**Integration in JUBE**

<u>Create Performance Model</u>
**Split data into training and validation sets, train the model, validate it, and tune parameters until an acceptable model.**

<u>Pre Processing</u>
**Perform feature engineering and select the target. If necessary, clean the data.**

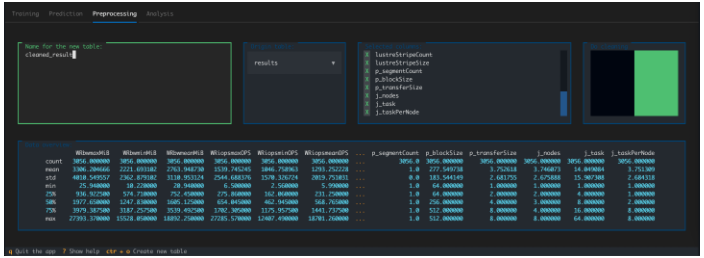



Zhu, Z., Wang, C. and Neuwirth, S., 2025. *Advancing HPC Performance Modeling with an Interactive, Automated and Tool-Agnostic ML-Driven Workflow*. SSDBM'25. (to appear)

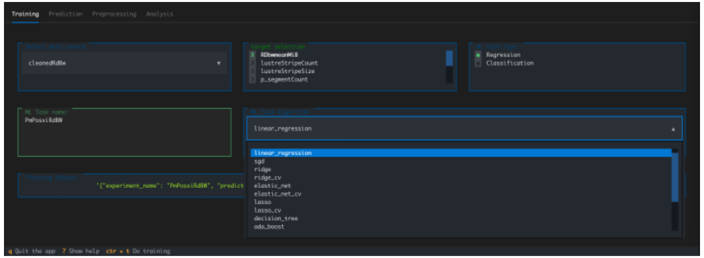
# Reproducible, Tool-Agnostic Workflows
## *JUBE-ML Prototype Implementation*

- Prototype *extends the JUBE framework* with support for automatic ML-based performance modeling and supports variety of ML algorithms
- JUBE-ML is enhanced by the *sqlite-ml extension*
- JUBE-ML stage provides *four key functionalities*:
  - ***Data analysis:*** insights into performance results (e.g., min, max, mean) and lambda functions
  - ***Data preprocessing:*** cleans and filters data for ML, creates new tables with selected features & targets
  - ***ML model training:*** applies ML models (regression or classification) to build a performance model
  - ***Prediction:*** validates the model & enables different analysis scenarios, such as identifying irrelevant parameters and predicting system performance

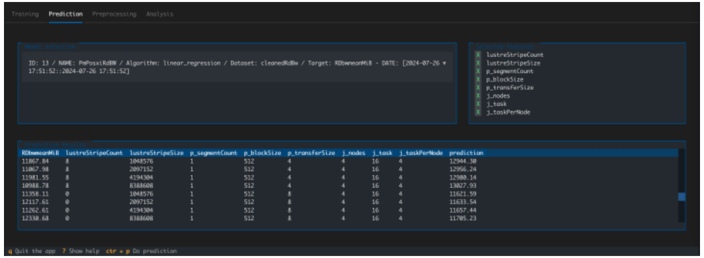
PRE-PROCESSING AND ANALYSIS



TRAINING PHASE



PREDICTION AND VALIDATION

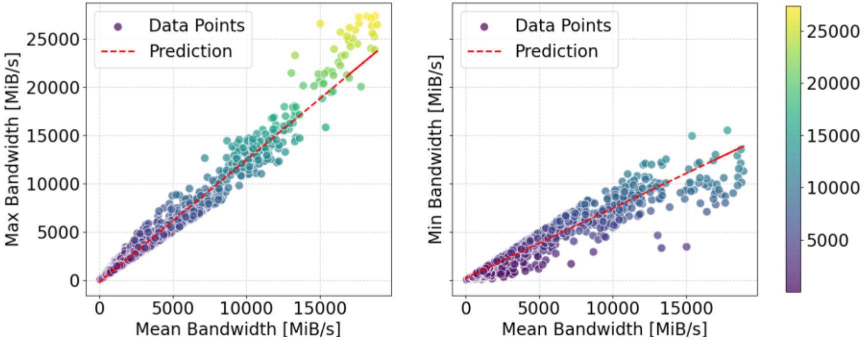


# Reproducible, Tool-Agnostic Workflows

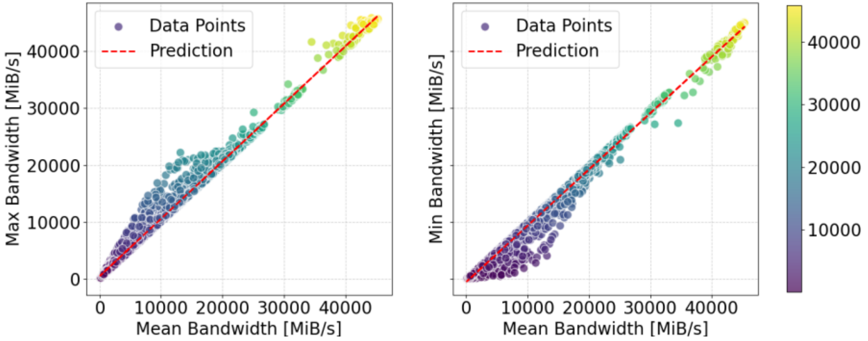
## *JUBE-ML Case Study: I/O Bandwidth*

- *Use Case:* I/O bandwidth modeling and prediction to demonstrate the JUBE-ML workflow
  - IOR benchmark to simulate various I/O workloads and generate performance data
  - Linear regression: 25% for training, 75% for validation
- Despite the small training sample, the model predicts both minimum & maximum bandwidths well

| | |
|---|---|
| Block size | 64MB, 256MB, 512MB |
| Transfer size | 1MB, 2MB, 4MB, 8MB |
| Lustre striping count | 0, 2, 4, 8 |
| Lustre striping size | 1MB, 2MB, 4MB, 8MB |
| Nodes | 1, 2, 4, 8 |
| Tasks per node | 1, 2, 4, 8 |




**POSIX write performance prediction with JUBE-ML.**




**POSIX read performance prediction with JUBE-ML.**

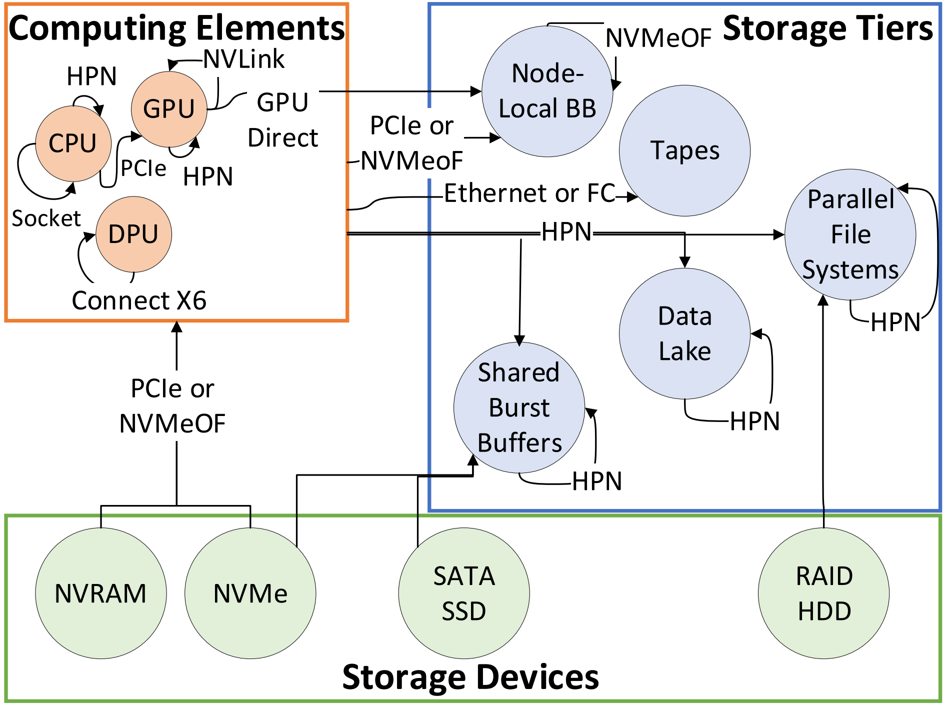
# Toward Explainable and Verified HPC Performance

*Bridging measurement and meaning through explainable models and verified traces*

# Toward Explainable and Verified HPC Performance

## *Master Architectural Plan (MAP)*

- Derived from current TOP500 and IO500
- Comprehensive framework designed to encapsulate the evolution of HPC storage architectures
- Primary objective of MAP is to provide a ***standardized yet adaptable blueprint*** to align software and monitoring tools across various HPC configurations
- ***Graph-based representation*** of components and interconnects for modeling data flows in HPC systems

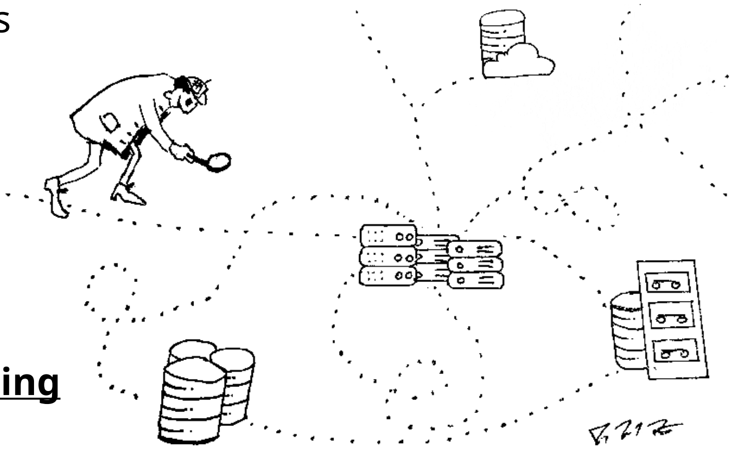



Neuwirth, S. and Devarajan, H., Wang, C., and Lofstead, J.., 2025. *XIO: Toward eXplainable I/O for HPC Systems*. SSDBM'25. (to appear)

# Toward Explainable and Verified HPC Performance

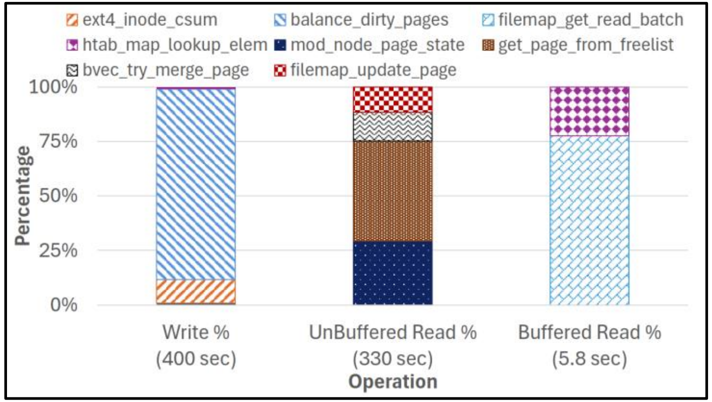
## *DataCrumbs: Comprehensive I/O Profiling*

- Only a few paths are covered by user-space monitoring tools
- Need access to more levels of software stack:
  - Multi-level software stack for high-level libraries
  - Multi-level kernel stack to understand buffering and stack cost
  - Multi-component to understand communication used for data movements
- **<u>DataCrumbs Approach: Low-Overhead Multi-Layer Profiling</u>**
  - Lightweight tool using eBPF (or SystemTap)
  - Attaching probes to user applications, middleware & kernel components => enabling transparent monitoring of relevant information
  - Sampling aggregated kernel data at configurable intervals => I/O patterns, buffer states, and system interactions
  - Correlating I/O function calls with kernel stack events, exposing bottlenecks such as page cache inefficiencies, metadata overhead, and system call latencies

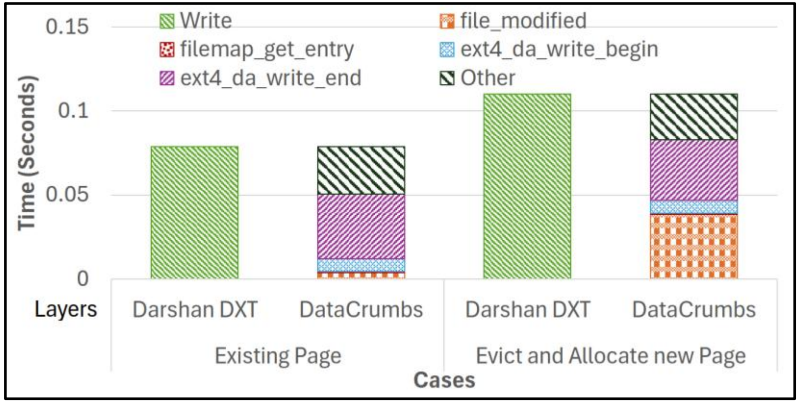


# Toward Explainable and Verified HPC Performance

## *Initial Evaluation: DataCrumbs vs. Darshan DXT*




**Performance breakdown with DataCrumbs for different read and write operations.**



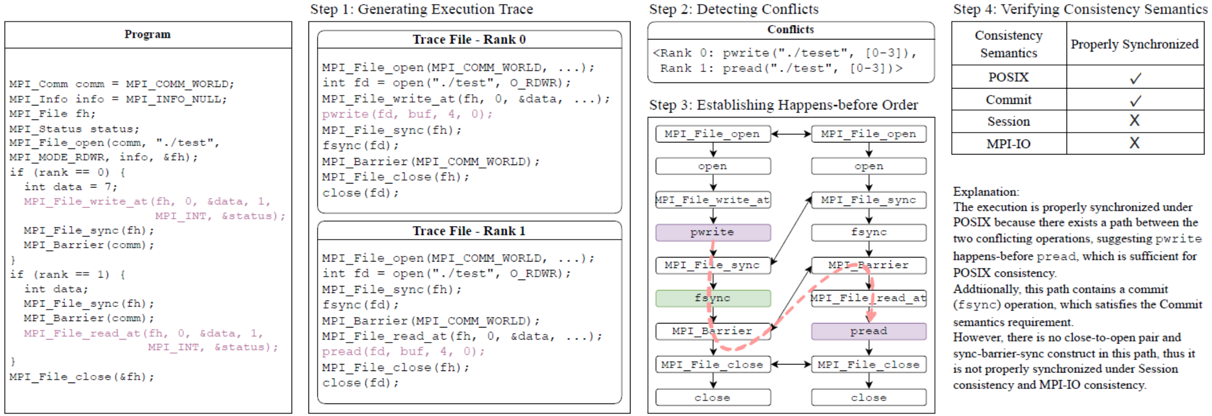

**Comparison of IOR performance breakdown with DataCrumbs and Darshan for two use cases.**

- ***Write operations (POSIX I/O):*** 82% of execution time spent on OS page cache management, highlighting impact of dirty page flushing
- ***Unbuffered read operations:*** High overhead due to frequent page cache misses
- ***Buffered read operations*** use different kernel functions
- ***Comparison of DataCrumbs and Darshan*** highlights need for visibility within kernel file system stack to understand performance variability within applications

Neuwirth, S. and Devarajan, H., Wang, C., and Lofstead, J.., 2025. *XIO: Toward eXplainable I/O for HPC Systems*. SSDBM'25. (to appear)

# Toward Explainable and Verified HPC Performance

*VerifyIO: Verifying Parallel I/O Consistency Semantics*

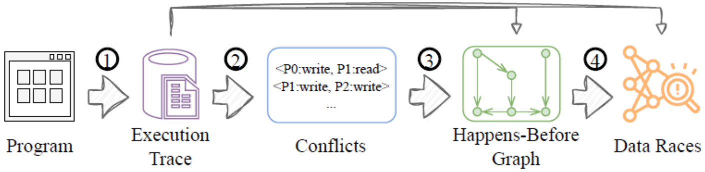



**Goals:**

- ***Ensure correctness*** on relaxed consistency systems
- ***Diagnose portability issues*** across backends
- ***Support future file systems*** beyond POSIX
- ***Enable developers*** to detect and fix semantic violations

**What is VerifyIO?** An open-source tool for trace-based verification of I/O consistency semantics in HPC applications.

**Why does it matter?** Emerging file systems and libraries relax consistency models (e.g., MPI-IO, Session), which can silently break application correctness.




Wang, C., Zhu, Z., Mohror, K., Neuwirth, S., and Snir, M., 2025. *VerifyIO: Verifying Adherence to Parallel I/O Consistency Semantics*. IPDPS'25. (to appear)

# Toward Explainable and Verified HPC Performance
## *FlexBench: From Traces to Tunable I/O Benchmarks*

**FlexBench is…**
- A benchmark generator that reconstructs and manipulates I/O patterns from execution traces
- Built on top of Recorder+, leveraging Context-Free Grammars (CFGs) to compress and describe I/O behavior

**Why do we need FlexBench?**
- Traditional tracing tools like Darshan or Recorder capture detailed I/O but lack replay and what-if capabilities
- FlexBench enables users to replay, analyze, and tune the I/O behavior of applications, even without modifying code

**Core Goals:**
1. Use CFGs to precisely describe application I/O patterns
2. Reproduce original I/O performance from filtered traces
3. Expose optimization opportunities (e.g., parameter tuning)

**Step 1: Recording the Full Trace**

**Step 2: Filtering for a Clean Trace**

**Step 3: Deriving the Benchmark**

**Step 4: Performance Optimization**

# Conclusions

# Conclusions

*Open Problems and Community Challenges*

## Benchmarking Trust

- What does it mean to trust a performance result?
- Can we quantify trust the way we quantify performance?

## Metadata Standardization

- Can we converge on trace metadata schemas across tools?
- How do we ensure trace context is captured and preserved?

## From Metrics to Meaning

- What constitutes a verified insight?
- Can we establish common ground between correctness verification and performance validation?

## Reproducibility at Scale

- How do we scale reproducibility beyond case studies?
- What community infrastructure (e.g. shared testbeds, curated traces) do we need?

**Trustworthy performance insights are sustainable insights.**

# Thank you for your Attention!

**Dr. Sarah M. Neuwirth**
*Professor of Computer Science*
Johannes Gutenberg University Mainz
*Email:* neuwirth@uni-mainz.de
*Website:* https://www.hpca-group.de/
*NHR South-West HPC Center:* https://nhrsw.de/