Efficient Data Movement and Computation via In-Flight Analysis

THE OHIO STATE UNIVERSITY

Noah Lewis, Suren Byna

ABSTRACT

- **In-flight analysis** refers to performing computations on data while it is in transit between the source and the destination.
- As workloads continue to grow in scale, optimally scheduling compute and managing data movement grows increasingly difficult.
- PDC's (Proactive Data Container) [1] current in-flight analysis implementation is restrictive because:
  - Transformations are attached per region transfer.
  - Transformations are unable to be chained for more complex workflows.
  - PDC has limited visibility into the overall data pipeline.
- Our work explores how to overcome these challenges to make in-flight analysis more accessible, robust, and performant across diverse HPC workflows.

MOTIVATION

- Deciding where computation runs and where data should be moved is a challenge in large-scale workflows.
- Poor decisions lead to unnecessary data movement, resource underutilization, and longer runtimes.
- In typical applications, clients process data through multi-stage pipelines to produce results.
- Writing custom software to manage data movement introduces significant complexity.
- Optimally scheduling compute adds further challenges.
- A method to reduce this complexity by automating computation and data movement decisions is needed.

EXISTING WORK

- Many cloud providers offer some form of in-flight analysis.
- Popular cloud services are AWS Lambda, Google Cloud Functions, and Azure Functions.
- Apache Spark [3] and similar frameworks are used in HPC for in-flight computations.
- However, they mainly focus on computation and offer limited support for optimizing data movement and storage for large-scale HPC workflows.

PDC'S CURRENT STATE

- The left figure shows PDC's previous in-flight implementation.
  - Transformations could not be chained.
  - Transformations are set per read/write of each region or object.
- The right figures show performance gains when using transformations.

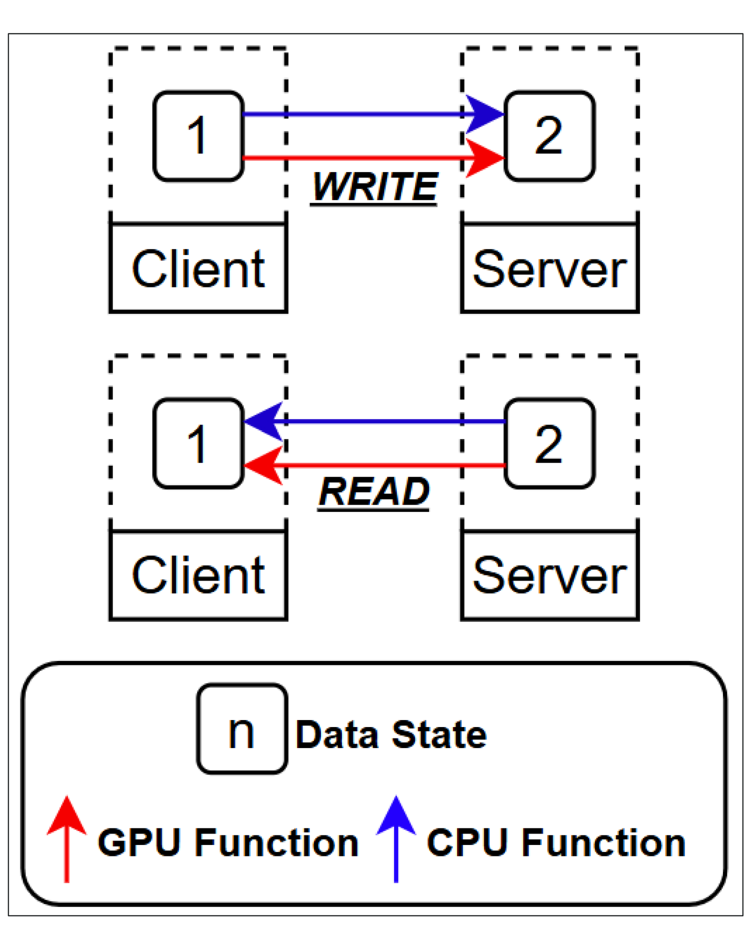

Fig. 1. The transformation used is dynamically selected by PDC at runtime using heuristics.

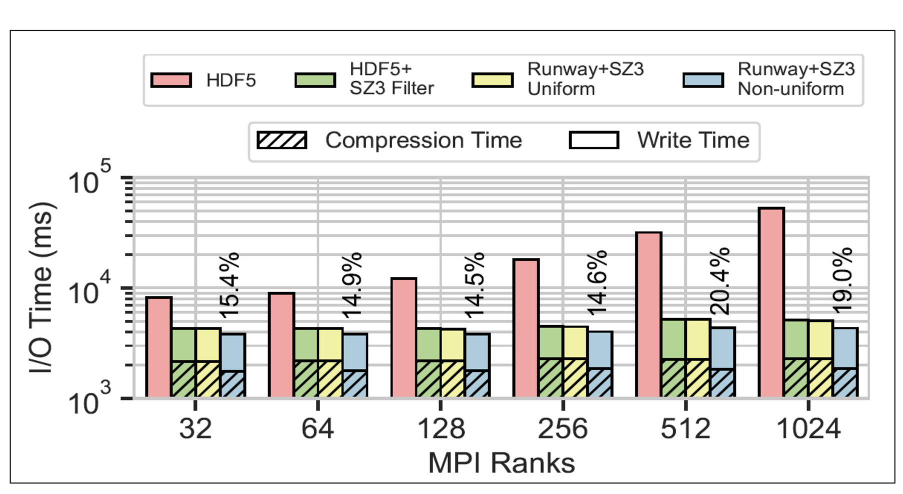

Fig. 2. I/O time when performing SZ3 compression [2].

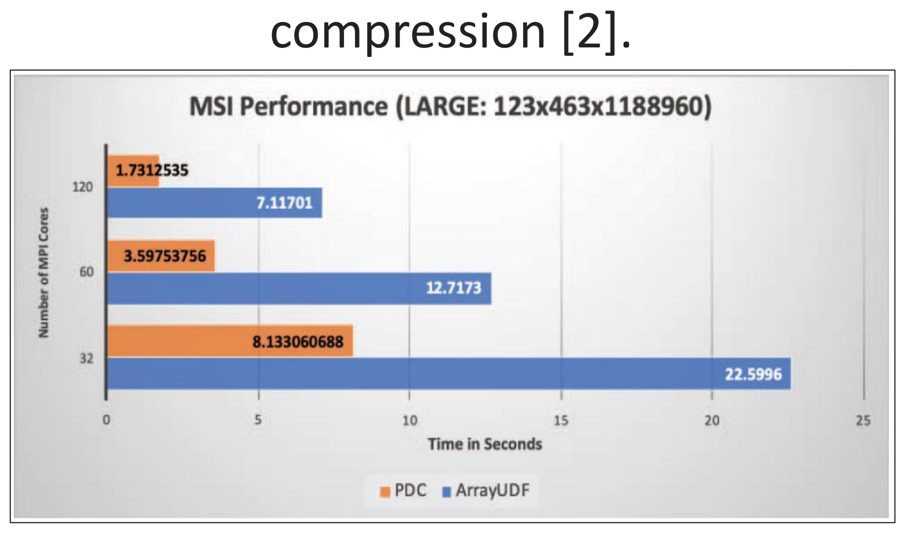

Fig. 3. The performance of OpenMSI: A 6-point stencil [1].

NEW IN-FLIGHT ANALYSIS DESIGN

- Client's construct **Directed Graphs** consisting of:
  - **Data States**: The state of a region at a specific point in the graph.
  - **Transformations**: Functions which take input and output regions as parameters each with a unique data state.

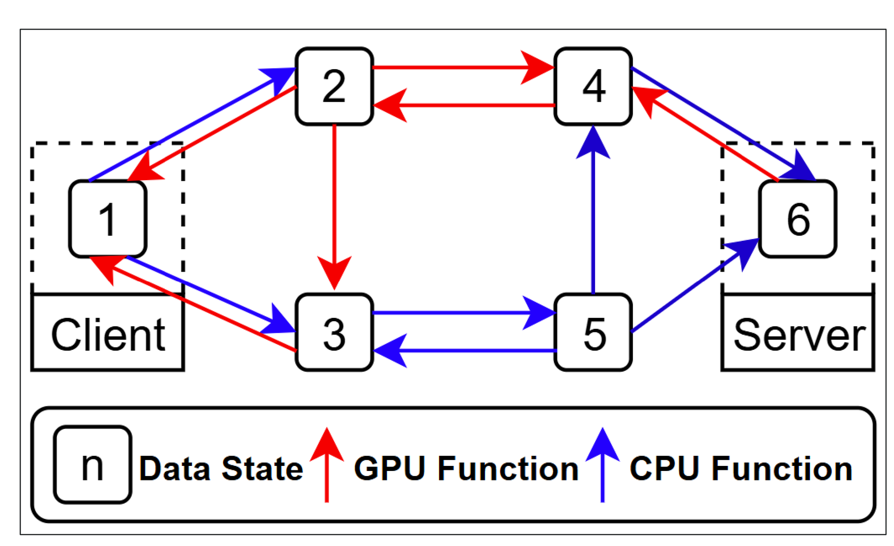

Fig. 4. Client reads and writes transparently pass through the graph once it is attached to objects or regions. Common transformations include compression, type conversion, and encryption.

- Directed graphs can be persisted and reused across various workflows.

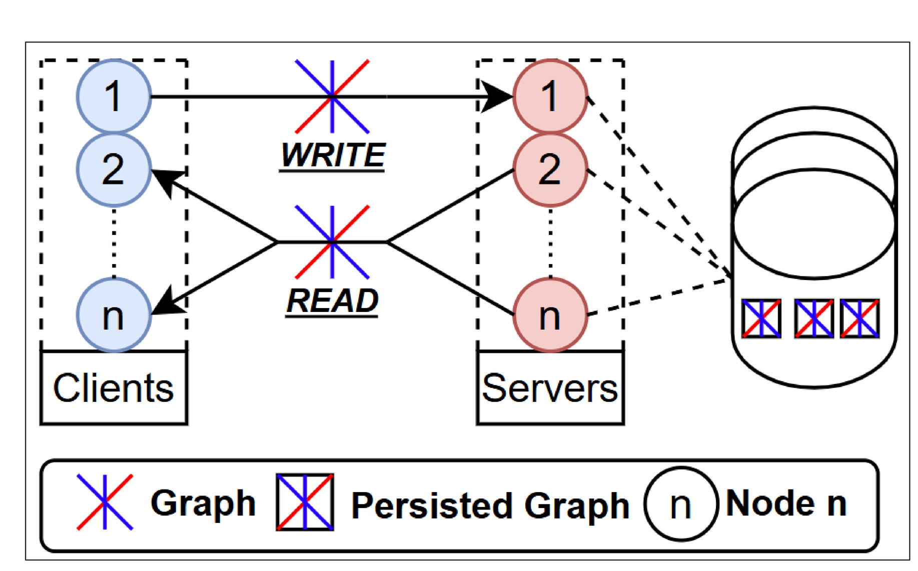

Fig. 5. Clients manage directed graphs with open, read, and close operations analogous to those for PDC objects.

CONCLUSION

- As HPC applications scale in size and complexity, scheduling compute and managing data movement becomes increasingly difficult.
- In-flight analysis offers a potential solution to reduce this complexity, enabling the underlying compute and storage systems to make real-time, informed decisions.
- PDC's new client API should ease user development and allow PDC to optimally schedule compute and manage data movement.

NEXT STEPS

- Finalize the client API design and implementation to ensure it is expressive and easy to use.
- Identify and categorize workflows that would benefit from in-flight analysis integration.
- Incorporate GPU-accelerated analysis and assess the benefits of GPUDirect Storage (GDS) during data transit.
- Enable clients to specify performance, security, and efficiency objectives that PDC pursues by leveraging directed graphs and system heuristics.

REFERENCES

[1] R. Warren *et al.*, "Analysis in the Data Path of an Object-Centric Data Management System," *2019 IEEE 26th (HiPC)*

[2] J. Ravi, S. Byna and M. Becchi, "Runway: In-transit Data Compression on Heterogeneous HPC Systems," *2023 IEEE/ACM 23rd (CCGrid)*

[3] Zaharia, Matei, et al. "Apache Spark: A Unified Engine for Big Data Processing." *Communications of the ACM*, vol. 59, no. 11, Nov. 2016, pp. 56–65, doi:10.1145/2934664.