Accelerating Communications in High-Performance Scientific Workflows

J. Gregory Pauloski

Advised by Kyle Chard and Ian Foster

University of Chicago & Argonne National Laboratory

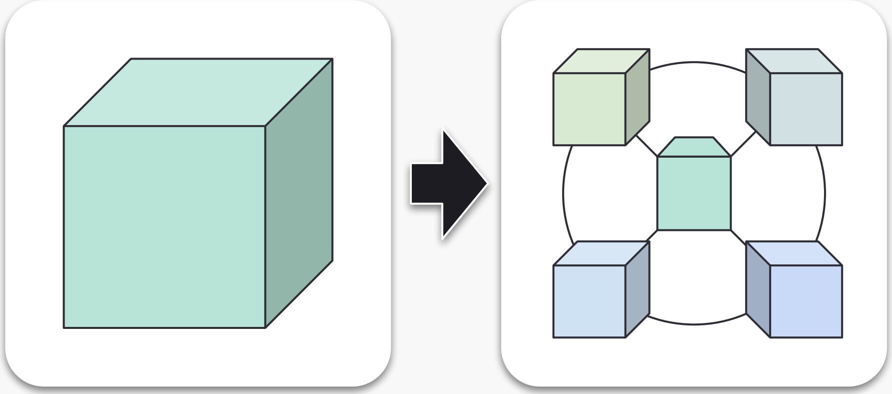
21 November 2024 — Atlanta, Georgia

A Shift in *Scientific Programs...*

Scientists want to utilize compute in more places.

Why?

- ➔ Faster & more reliable networks
- ➔ Specialized accelerators
- ➔ Data locality
- ➔ Performance requirements
- ➔ Compute availability & costs
- ➔ Better cloud management



Monolithic Programs ***Composition of Loosely Coupled Components***

Modern Science Applications are *Task-centric*

Applications are composed as a set of **discrete tasks** designed to **automate** computational processes to achieve a **scientific goal**

Benefits

- Heterogeneous Resources
- Software Modularity
- Monitoring
- Performance
- Reproducibility
- *and many more!*

Applications [1]

- Bioinformatics
- Cosmology
- High Energy Physics
- Materials Science
- Molecular Dynamics
- *and many more!*

Challenges [2]

- Coupling AI/ML/Quantum
- Cloud and HPC Integration
- Data Flow/Provenance
- Standards/Interoperability
- Performance
- *and many more!*

[1] "Scientific Workflows: Moving Across Paradigms" (**https://dl.acm.org/doi/10.1145/3012429**)
[2] Workflows Community Summit (**https://arxiv.org/abs/2304.00019**)

Federating Scientific Workflows

Distribute computational tasks across federated devices? Possible.

- ➔ Globus Compute distributed FaaS Model

Manage intermediate data between tasks? Limited.

- ➔ Interoperability between distributed/parallel frameworks is challenging
- ➔ Cloud object storage is reliable/available but expensive for data-intensive apps
- ➔ P2P CDNs are good for edge devices but bad for clusters

Manage communication between independent components? Limited.

- ➔ Easy in cloud-native apps (microservice architectures)
- ➔ Hard in federated apps (requires ad-hoc solutions)

Communication in Federated Science Workflows

New programming techniques ***enable and accelerate*** *task-oriented* ***science applications*** *executed* ***across the computing continuum****.*

P1	What are the limitations in existing distributed computing framework?	eScience '24 (Best Paper)
P2	How to represent and efficiently move objects in federated applications?	SC '23 & HPPSS '24
P3	How to design high-level data flow patterns?	Under Review
P4	How to build and deploy stateful agents across federated systems?	In Progress

Better, easier, & faster science! — MLHPC '21, IJHPCA '22, HCW '23, IJHPCA '24 & Others Under Review/In Progress

Task Performance Suite

ProxyStore

Proxy Patterns

Federated Agents

Task Execution Frameworks

Manage the execution of tasks in parallel across arbitrary hardware.

Workflow Management Systems

Define, manage, and execute workflows represented by a directed acyclic graph (DAG) of tasks

Explicit

DAG defined via configuration file or domain specific language

Implicit

Task dependencies derived through dynamic evaluation of a procedural script

Concurrent Executors

On-demand asynchronous execution of tasks

The Status Quo

Ad Hoc Benchmarks

- Framework-specific examples/demos
- Custom, single-use evaluation scripts for a publication
- Forks of real science applications

Problems

- Code is **framework-specific**
- Ad-hoc scripts subject to **code rot**
- Porting applications can be **onerous**
- Subtle **errors** in ported applications can lead to **inaccurate comparisons**

SimGrid: a Generic … Large-Scale Distribut…

Univer…

Developing accurate and scalable simu… management systems with WRENCH

Application skeletons: Construction and use in …

Daniel S. Katz[a,*], Andre Merzky[b], Zhao Zhang[c], Shantenu Jha[b]

[a] Computation Institute, University of Chicago & Argonne National Laboratory, Chicago, IL, USA

[b] RADICAL Laboratory, Rutgers…

[c] AMPLab, University of Califo…

WfCommons: A Framework for Enabling Scientific Workflow Research and Development

Tainã Coleman[a,c,*], Henri Casanova[b], Loïc Pottier[a], Manav Kaushik[c], Ewa Deelman[a,c], Rafael Ferreira da Silva[a,c,*]

WfBench: Automated Generation of Scientific Workflow Benchmarks

Tainã Coleman*, Henri Casanova† Ketan Maheshwari‡, Loïc Pottier*, Sean R. Wilkinson‡
Justin Wozniak§, Frédéric Suter‡, Mallikarjun Shankar‡, Rafael Ferreira da Silva‡

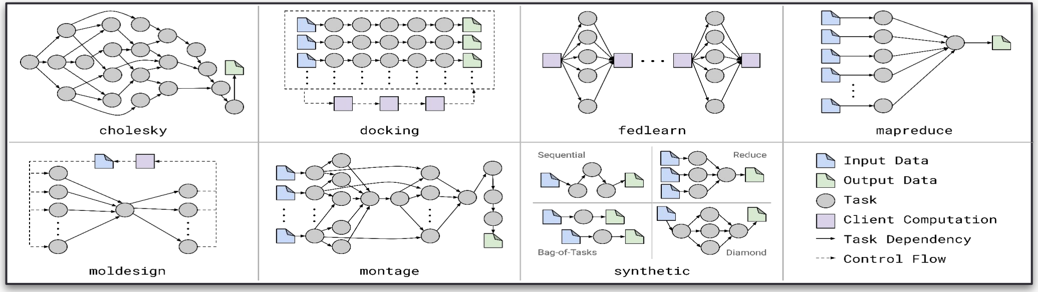
…niversity of Hawaii, Honolulu, HI, USA
…onal Laboratory, Oak Ridge, TN, USA

Prior work focused on **simulations** and **synthetic workloads**

TaPS — Goals

- ➔ Provide reference/standard applications for benchmarking workloads
- ➔ Benchmark task executors & data management systems
- ➔ Robust and reproducible configuration system
- ➔ Guide future research

Type	Name	Domain	Task / Data Type(s)
Real	**cholesky**	Linear Algebra	Python / In-memory
	docking	Drug Discovery	Executable, Python / File
	fedlearn	Machine Learning	Python / In-memory
	mapreduce	Text Analysis	Python / File, In-memory
	moldesign	Molecular Design	Python / In-memory
	montage	Astronomy	Executable / File
Synthetic	**synthetic**	—	Python / In-Memory
	failures	—	*Depends on base app*



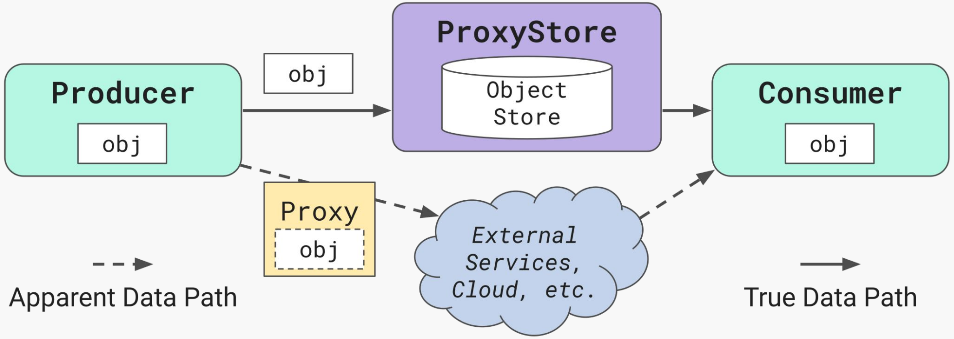
Task Performance Suite

ProxyStore

Proxy Patterns

Federated Agents

ProxyStore



Data flow management library for distributed Python workflows

- Represent and efficiently move objects in federated applications
- Proxy ***transparently*** decouples control and data flow
- Best of both ***pass-by-reference*** and ***pass-by-value***
- Use any mediated communication method via plugins

Proxy Objects

What is a proxy (in this context)?

- Self-contained wide-area **reference** to a **target** object
- Transparently resolve target **just-in-time** when first used

What are the benefits?

- Performance (pass-by-reference, async resolve, skip unused objects)
- Reduce code complexity
- Partial resolution of complex objects
- Access control

```python
from proxystore.connectors import RedisConnector
from proxystore.store import Store
from proxystore.proxy import Proxy

def foo(x: Bar) -> ...:
    # Resolve of x deferred until use
    assert isinstance(x, Bar)
    # More computation...

with Store('demo', RedisConnector(...)) as store:
    x = Bar(...)
    p = store.proxy(x)  # Anything can be proxied
    assert isinstance(p, Proxy)
    foo(p)  # Proxies can be passed-by-ref anywhere
```

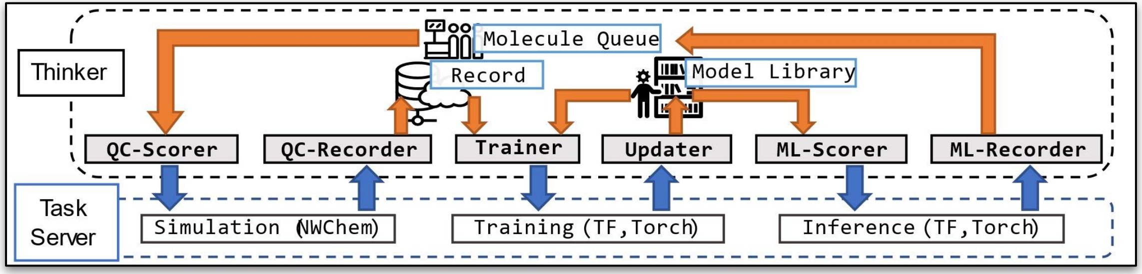
Connectors

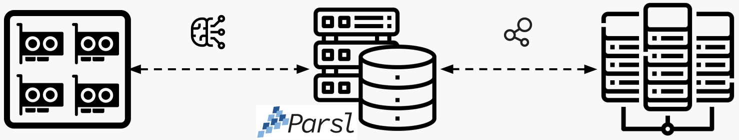
- Many **mediated** methods supported (mediated methods because producer/consumer may be temporally decoupled)
- `Connector` = Python **`Protocol`**
- **`MultiConnector`**: Policy-based routing between instances

Protocol	Storage	Intra-Site	Inter-Site	Persistence
File System	Disk	✓		✓
Redis/KeyDB	Hybrid	✓		✓
Margo	Memory	✓		
UCX	Memory	✓		
ZMQ	Memory	✓		
Globus	Disk		✓	✓
DAOS	Disk*	✓		✓
P2P Endpoint	Hybrid	✓	✓	✓

Multi-site Active Learning

Science Goal: Use quantum chemistry simulations and surrogate ML models to efficiently identify electrolytes with high ionization potentials in a candidate set.





20 GPU Workstation
- Training Tasks
- Inference Tasks

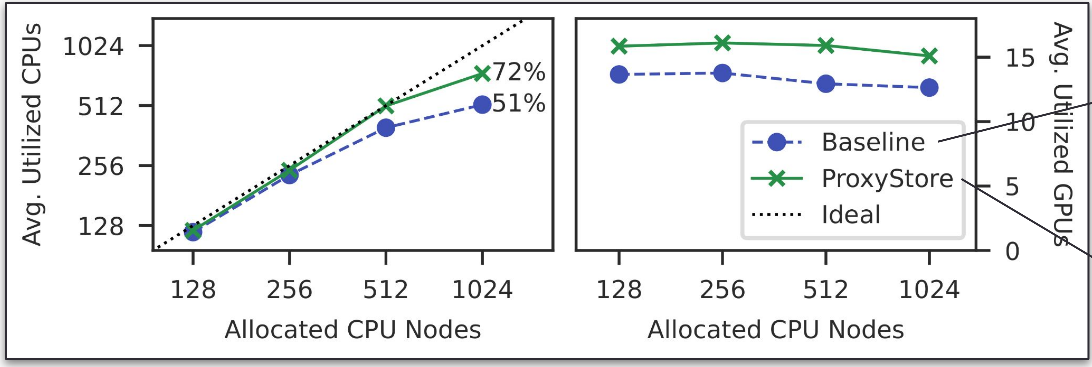
Workstation/Head Node
- Submit work
- Process results

1024 Theta KNL Nodes
- Simulation Tasks

Logan Ward, J. Gregory Pauloski, Valerie Hayot-Sasson, Ryan Chard, Yadu Babuji, Ganesh Sivaraman, Sutanay Choudhury, Kyle Chard, Rajeev Thakur, and Ian Foster. *Cloud services enable efficient AI-guided simulation workflows across heterogeneous resources*. In Heterogeneity in Computing Workshop at IPDPS. IEEE Computer Society, 2023.

Multi-site Active Learning

Systems Goal: Reduce task communication overheads in workflow system to increase system utilization and task throughput.



Baseline: Parsl manages all intermediate data (transfer via manually created SSH channels)

ProxyStore: MultiConnector
- *Simulation*: Redis
- *Training*: P2P Endpoints
- *Inference*: Globus Transfer/ P2P Endpoints

Takeaways
- Reduce overheads
- Re-used data only communicated once
- Orchestrator can choose ideal communication method
- No changes to task code needed

- Task Performance Suite
- ProxyStore
- <u>**Proxy Patterns**</u>
- Federated Agents

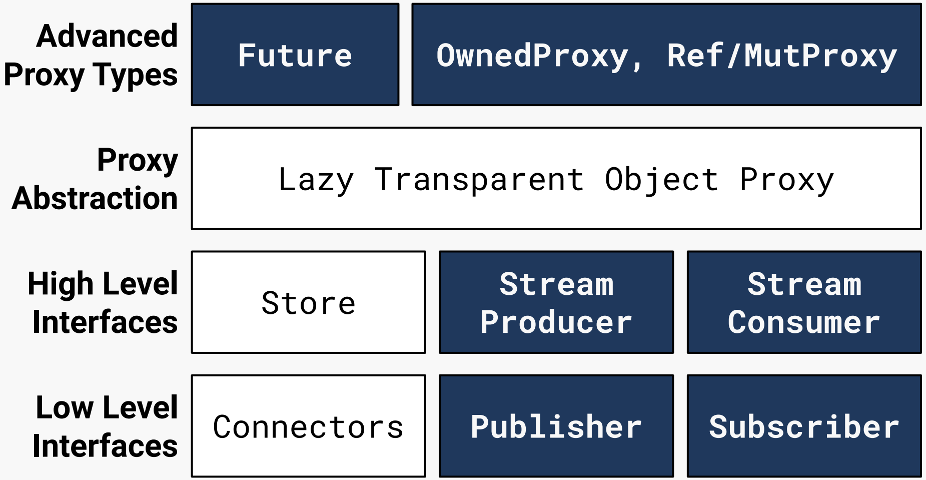
Yet...

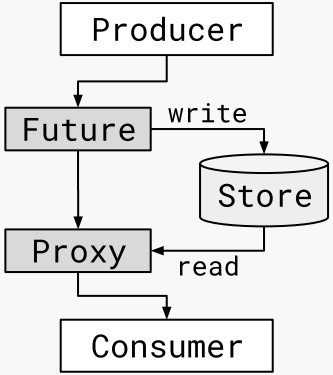
Object proxy is a ***low-level*** paradigm:

- ➔ A great building block within larger frameworks
- ➔ Has known limitations

What are ***higher-level*** proxy patterns?

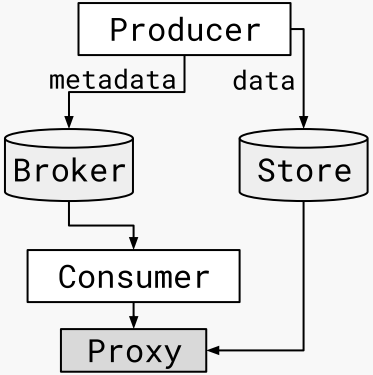
- ➔ Accelerate development of more sophisticated applications
- ➔ Address limitations





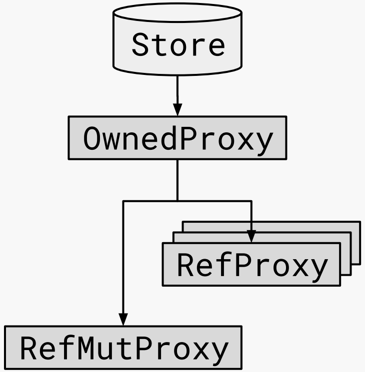
ProxyFutures

- ➔ Create a proxy before the target exists
- ➔ Inject data flow dependencies into compute tasks



ProxyStream

- ➔ High-throughput & low-latency streams
- ➔ Decouple event & metadata notification from bulk-data transfer



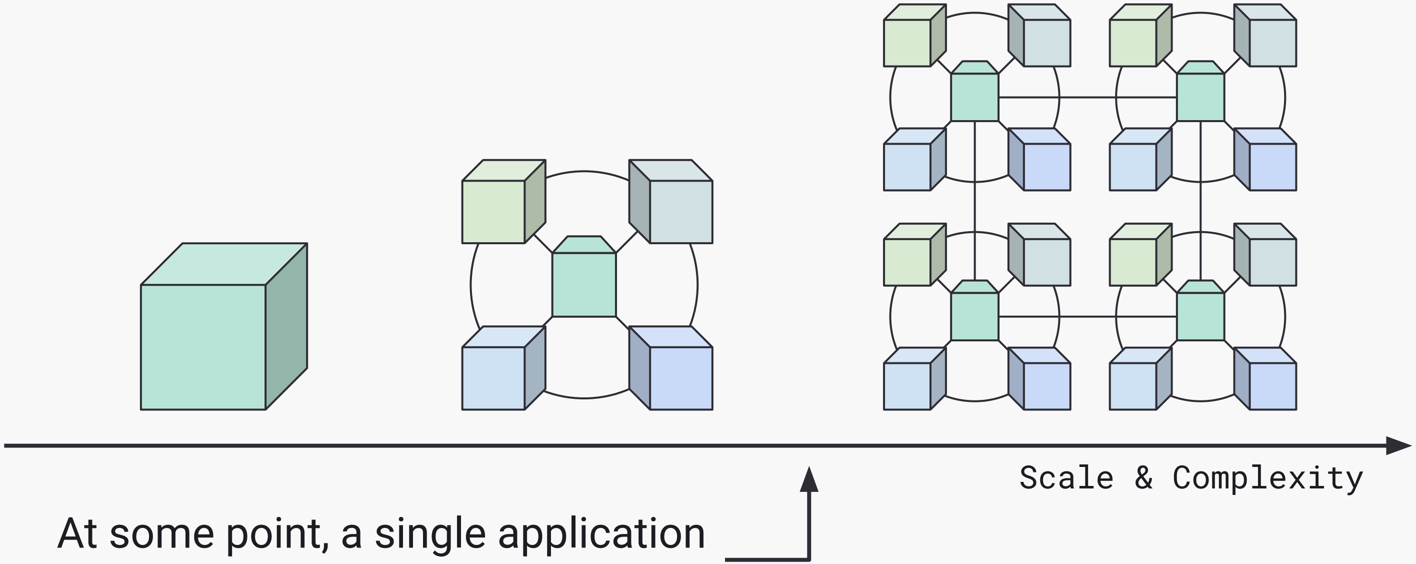
Proxy Ownership

- ➔ Automatically manage proxy lifetimes
- ➔ Borrow proxies safely (immutable or mutable) at runtime

- Task Performance Suite
- ProxyStore
- Proxy Patterns
- **Federated Agents***

** Exciting name pending…*

Scaling Distributed Science Apps



At some point, a single application
no longer makes sense

Agent Architectures for Science

What is an Agent?

- ➔ Entity with an internal state, set of actions it can perform, and a control loop that determines what actions to perform
- ➔ Agents can use message passing to invoke actions on each other

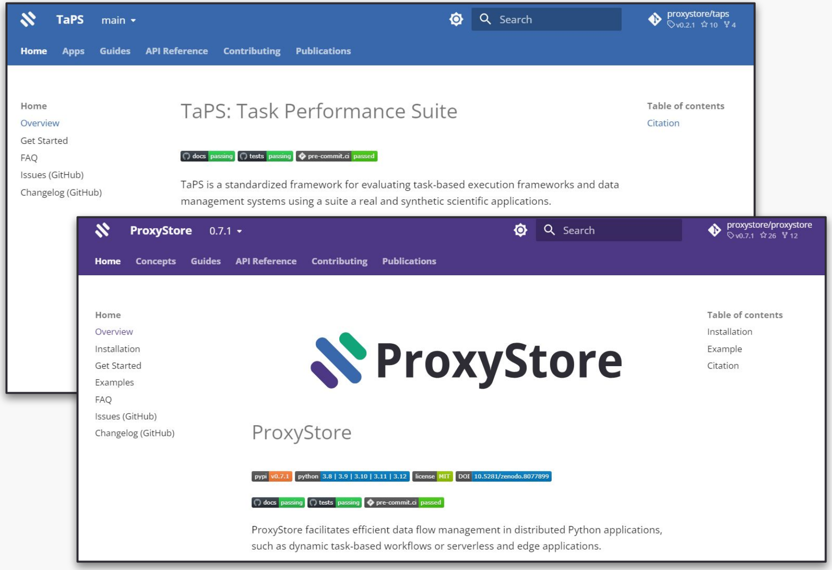
Why use agents for science?

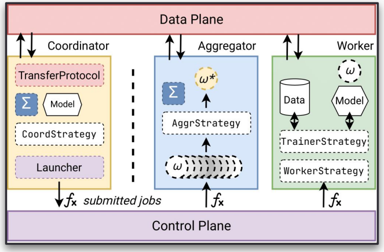
- ➔ Turn components into independent services (microservice-like)
- ➔ Agents operate autonomously but still cooperatively
- ➔ More natural expression of multi-stage computational science processes

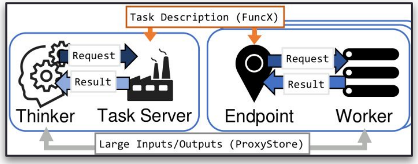
Federated Agents — Goals

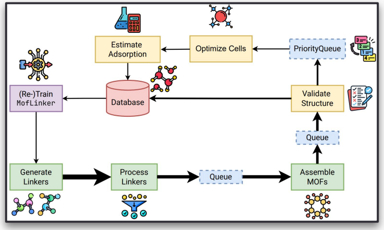
- **Middleware** with minimal set of features necessary to build any kind of agents (embodied, cooperative, AI, etc.)
- Mechanisms for **federated execution**
  - Globus Compute Launcher
  - ProxyStore peer-to-peer communication methods
  - AMQP (Hybrid-cloud w/ RabbitMQ) and DHT (Full P2P w/ Kademlia) Exchanges
- **Support research** in self-driving lab, AI-in-the-loop workflows, federated learning, etc.

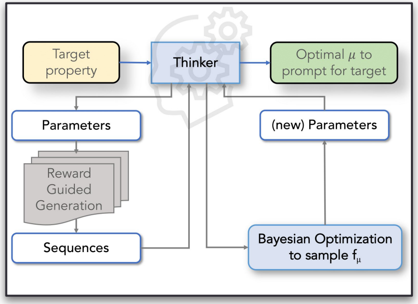
Impact

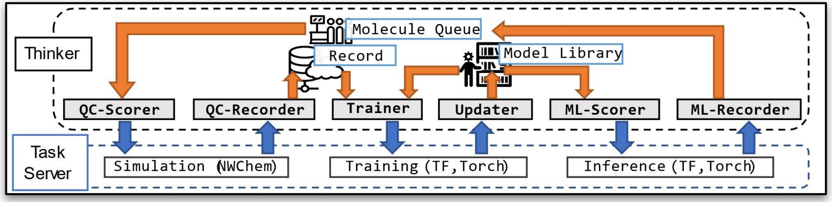












Translating open-source software into new science

Accelerating Communications in High-Performance Scientific Workflows

- ➔ **TaPS**: *Support research in distributed/parallel execution* —— eScience '24 (Best Paper)
- ➔ **ProxyStore**: *Better object references for federated environments* —— SC '23 & HPPSS '24
- ➔ **Proxy Patterns**: *Better data flow patterns with object proxies* —— Under Review
- ➔ **Federated Agents**: *Build science agents for autonomous discovery* —— In Progress

Better, easier, & faster science!

MLHPC '21, IJHPCA '22, HCW '23, IJHPCA '24 & Others In Review/Progress

Questions?

Contact:

Greg Pauloski
jgpauloski@uchicago.edu

github.com/proxystore

Reference:

github.com/proxystore
gregpauloski.com

Acknowledgements:

- Argonne National Laboratory under U.S. Department of Energy Contract DE-AC02-06CH11357
- National Science Foundation under Grant 2004894 and Grant 2209919
- ExaLearn Co-design Center of the Exascale Computing Project (17-SC-20-SC)