Accelerating Communications in High-Performance Scientific Workflows

J. Gregory Pauloski
Kyle Chard & Ian Foster (Advisors)
jgpauloski@uchicago.edu

SC24 Atlanta, GA | hpc creates.

Motivation: Data Flow Challenges & Opportunities in FaaS & Workflow Systems

- **What are the limitations of existing solutions?**
- **How to decouple data flow w/o rewriting apps?**
- **What high-level patterns can the proxy model enable?**
- **Can stateless frameworks support stateful apps?**
- **What mediated communication channels are best?**
- **How can learnings accelerate large-scale science?**

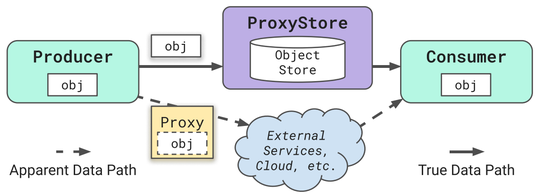
ProxyStore: Wide-Area Pass-by-Reference

/proxystore/proxystore — SC '23 Paper

Applications of ProxyStore

- **Workflow systems:** Reduce communication overheads
- **Federated FaaS platforms:** Bypass cloud data transfer
- **Edge computing:** Enable P2P transfer over diverse networks

Proxy Model



Object Proxy: *Transparent reference to object in global store*

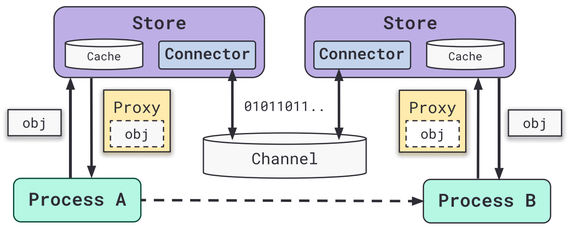
- → **Transparent:** Proxy behaves as *target* object (forwards ops)
- → **Lazy:** Factory invoked when proxy first accessed (*resolving*)
- → **Factory:** A callable (e.g., func) that returns the target object
- → **Pass-by-Ref:** Eventual user of proxy receives data copy
- → **Pass-by-Value:** No copies when proxy is not used

```python
import numpy; from proxystore.proxy import Proxy

x = numpy.array([1, 2, 3])  # Lambda function is simple factory
p = Proxy(lambda: x)  # Proxy can do everything numpy array can

assert isinstance(p, Proxy) and isinstance(p, numpy.ndarray)
assert numpy.array_equal(p, [1, 2, 3])
assert numpy.sum(p) == 6
```

ProxyStore Design



Store: *High-level abstraction of an object store for creating proxies*

- → `Store.proxy()` methods
- → Interfaces with connector
- → Object/propery caching
- → Custom serialization
- → Asynchronous resolution
- → Performance monitoring

Connector: *Low-level interface to a mediated storage*

- → Can be any mediated communication channel (e.g., object store, etc.)
- → Many implementations
- → Easy to extend

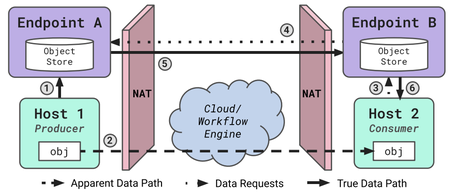
Mediated Communication Channels

Redis · globus · ØMQ · KeyDB · daos Storage Stack · Mochi · Endpoints · File Systems · UCX-Py

Proxy Model Benefits

- **Efficiency:** Proxies are lightweight to communicate
- **Compatibility:** Proxies are interoperable with existing code
- **Optimization:** Amortizes costs/partial object resolution
- **Security:** Ensure data are only resolved where permitted

P2P Apps with NAT Hole Punching



Circumvent expensive cloud transfer with easy P2P data transfer!

Easy-to-Use Python API

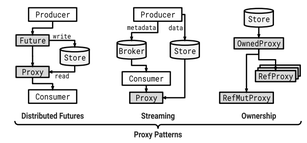
Try out ProxyStore! `$ pip install proxystore[all]`

```python
from concurrent.futures import ProcessPoolExecutor
from proxystore.connectors.redis import RedisConnector
from proxystore.store import Store

with Store('demo', RedisConnector('localhost', 6379)) as store:
    with ProcessPoolExecutor() as pool:
        proxy = store.proxy(list(range(1, 100000)))
        future = pool.submit(sum, proxy)
        print(future.result())
```

Proxy Patterns

Preprint



High-level patterns make the low-level proxy paradigm easier to use, simplifying the creation of sophisticated task-based apps.

Proxy Ownership

- → Rust-inspired borrowing and ownership semantics
- → DAG-based task structures
- → Tasks can mutably or immutably borrow proxies
- → Auto dereferencing and memory management
- → Object lifetimes

Distributed ProxyFutures

- → Implicit and explicit usage
- → Data flow dependencies
- → Execution engine agnostic
- → Any comm. channel

ProxyStream

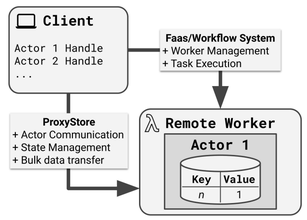
- → Stream-by-proxy model
- → Low-latency + high-bandwidth

Stateful Actors

Work-in-Progress

ProxyStore Actors

- → Compatible with any task-based execution engine (Dask, Globus Compute, Ray, etc.)
- → P2P/wide-area deployment with ProxyStore Endpoints
- → Temporal decoupling with actors mailbox model



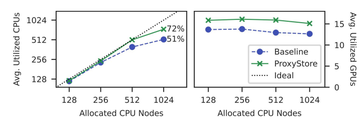
Have an interesting use case? Reach out and let us know!

Better Science

App Papers

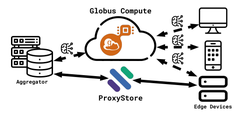
Multi-site Molecular Design

Colmena + Parsl + Proxystore



Hierarchical Fed. Learning

github.com/h-flox/flox



Other Apps

1000 Genomes: Enable implicit task dependencies and reduce makespans with ProxyFutures

GenSLM: Improve scaling in Parsl apps

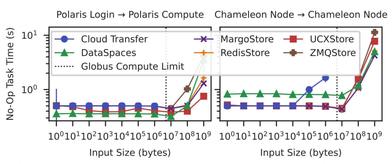
DeepDriveMD: Reduce ML inference latency in AI-steered ensemble simulations with proxy streaming

MOF Generation: Optimize memory usage

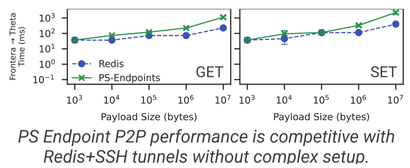
PSBench

/proxystore/benchmarks

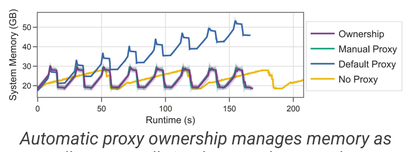
Evaluate mediated communication channels and data flow performance



Distributed in-memory connectors use RDMA for low-latency and high-bandwidth transfers.



PS Endpoint P2P performance is competitive with Redis+SSH tunnels without complex setup.



Automatic proxy ownership manages memory as well as manually implemented approaches.

Benchmarks: Round trip task time, P2P transfer, data flow optimization, streaming throughput, memory usage

Executors: Dask, Globus Compute, Parsl, ProcessPoolExecutor

Data management: Baseline, Proxystore, IPFS

TaPS

/proxystore/taps — eScience '24

Task Performance Suite

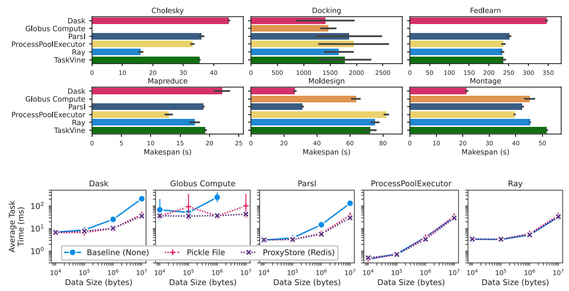
- → Reference real science apps for benchmarking workloads
- → Compare task execution engines and data management systems

Plugin System

Apps: Cholesky, Docking, Failures, Fed. Learning, MapReduce, Moldesign, Montage, Synthetic

Execution Frameworks: dask, python, Globus Compute, RAY, Parsl, TaskVine

Data Management: Files, ProxyStore



Performance Insights

- → Apps highlight strengths/weaknesses across frameworks
- → Pass-by-ref. key to reduce overheads (ObjectRef or Proxy)

Conclusions & Future Work

- → Proxy is powerful abstraction for performant/portable science apps
- → Reduces key performance bottlenecks in task execution frameworks
- → Proxy model can enable high-level application patterns
- → TaPS can enable reliable/reproducible benchmarking for community
- → Future Work: Stateful actors, better P2P transfer, new comm. tools
- → Future Work: Enable future exascale science apps with ProxyStore

THE UNIVERSITY OF CHICAGO · Argonne NATIONAL LABORATORY · globus labs

Code + Documentation github.com/proxystore

Papers + Portfolio gregpauloski.com