High Performance Distributed Object Storage Solutions

Saksham Goel | Nilanjan Daw | Rinku Shah | Pramod S Rao

Umesh Bellur | Purushottam Kulkarni

Systems and Networks Research Group
Department of Computer Science and Engineering
Indian Institute of Technology Bombay

10th December 2020

Outline

Setting the Context

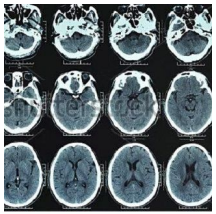
Survey of Existing Techniques

Technology Trends and Impacts

Action Plan

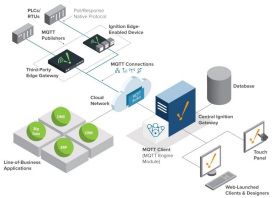
Computing at Scale and Storage

Big Data Analysis



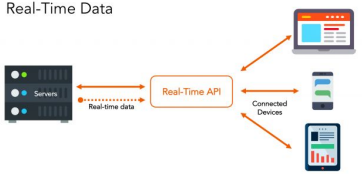
Covid-19 Protein Detection
Medical Image Processing
Oil and gas exploration
Climate Change Analysis

Edge / IoT Computing



Smart Homes
Smart Wearables
Virtual assistants
Autonomous vehicles

Real time Data Streaming



Content streaming services
Conference calling services
Real time systems

Low latency

High volume **High throughput**

Storage solutions are governed by application requirements

Technology Drivers of HPDOS

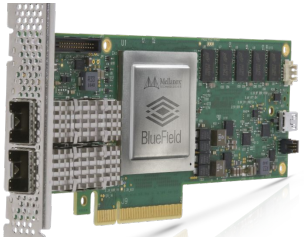
Memory

Storage Class Memories /
Non-Volatile RAM

Storage

NVMe
SSD
Flash Drives

Interconnects



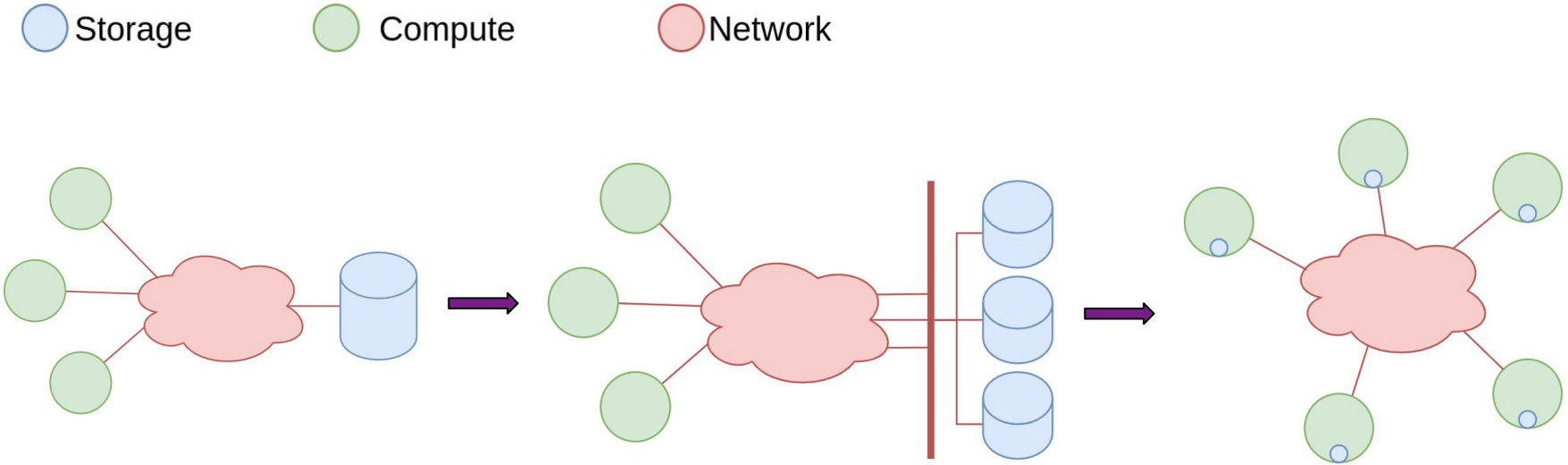
SmartNICs
RDMA
SR-IOV
DPDK

Low latency

High volume **High throughput**

Storage solutions are driven by technological advances

Architectural Continuum: Towards Hyperconvergence



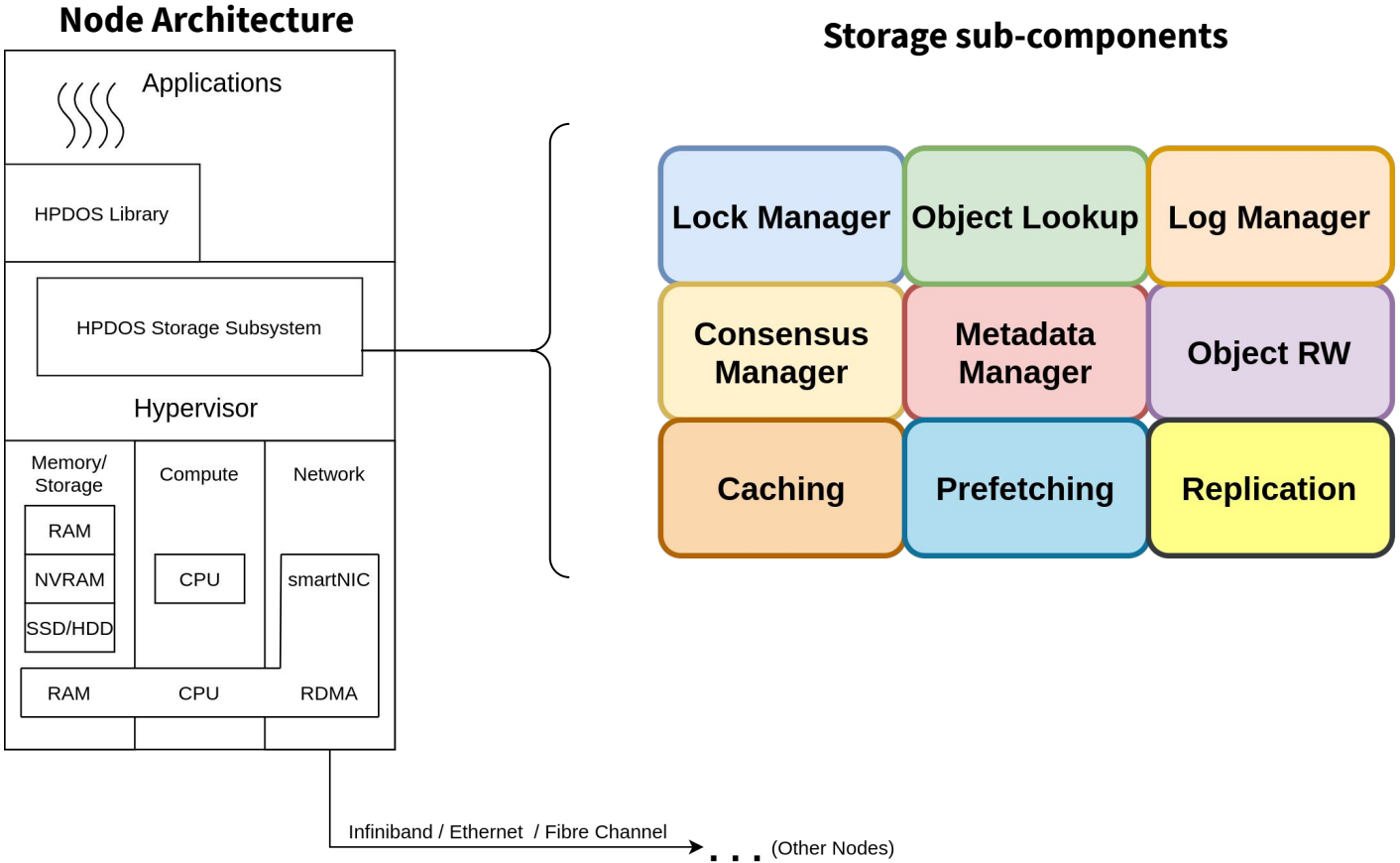
Networked attached storage

Storage clusters attached to a network

Hyperconverged storage + compute

* IBM Blue Gene

HPDOS in a Hyperconverged Setting



Outline

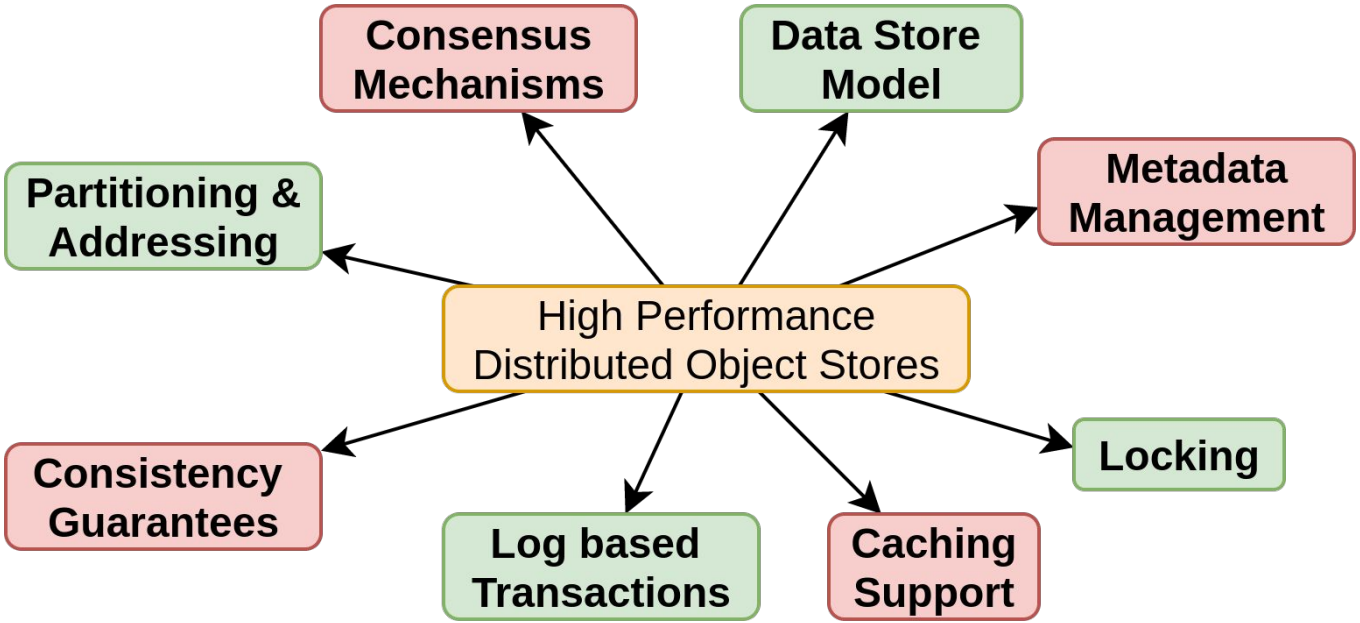
Setting the Context

Survey of Existing Techniques

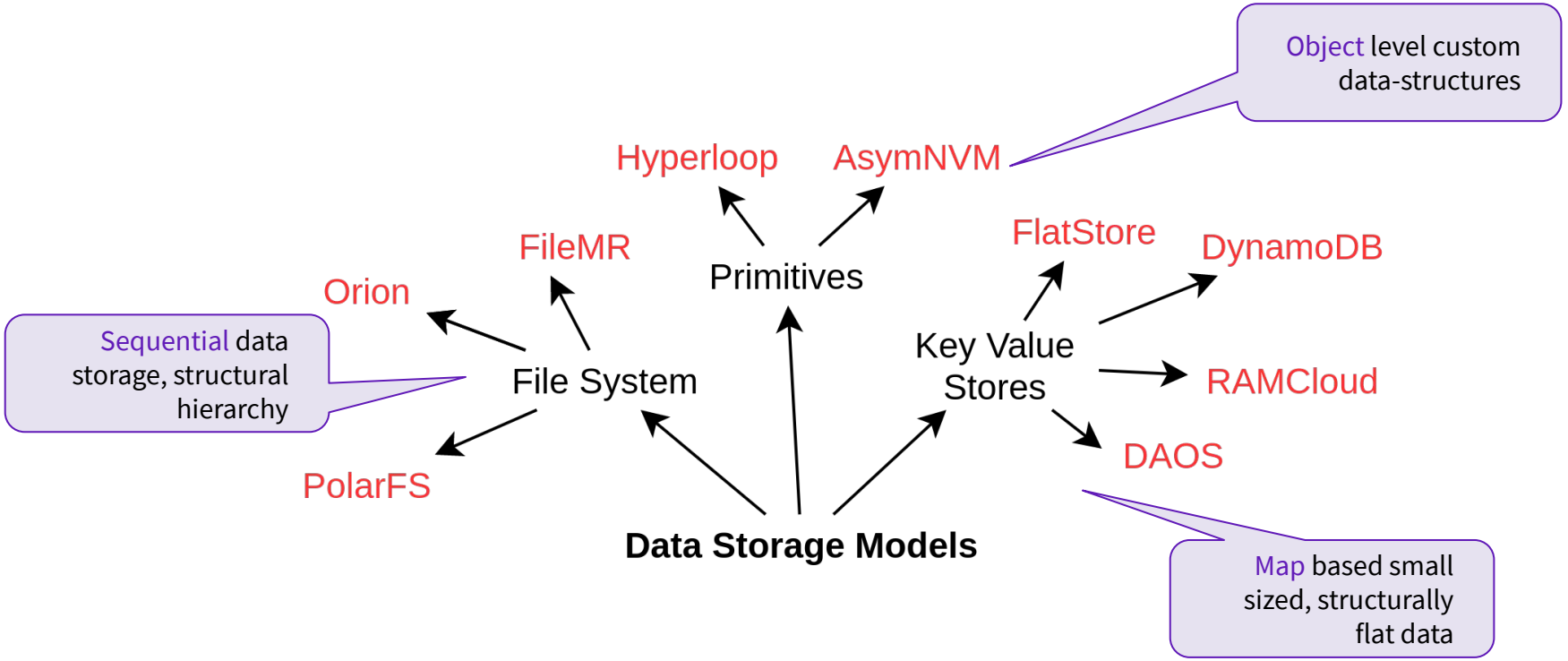
Technology Trends and Impacts

Action plan

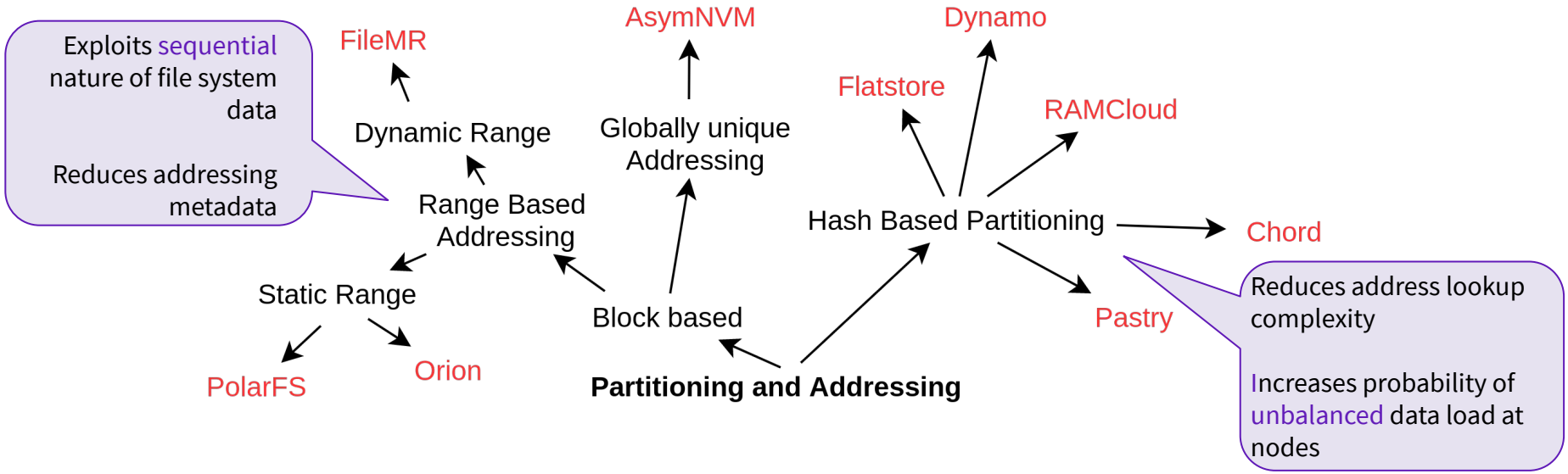
Structuring Existing Research in HPDOS



Data Store Models



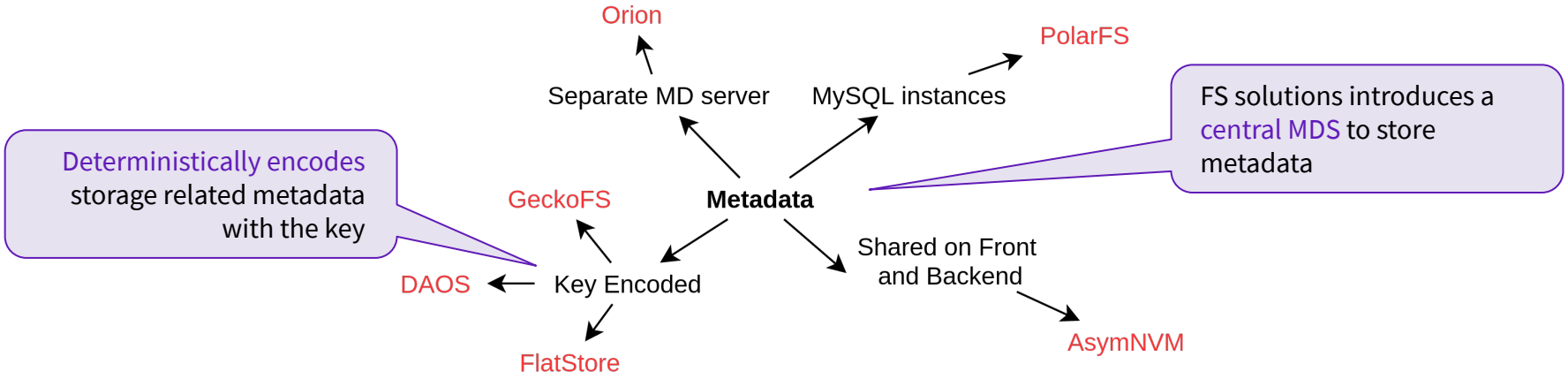
Data Partitioning and Addressing



File systems prefer Range based addressing

Key-value based stores overwhelmingly uses Hash Based partitioning and addressing.

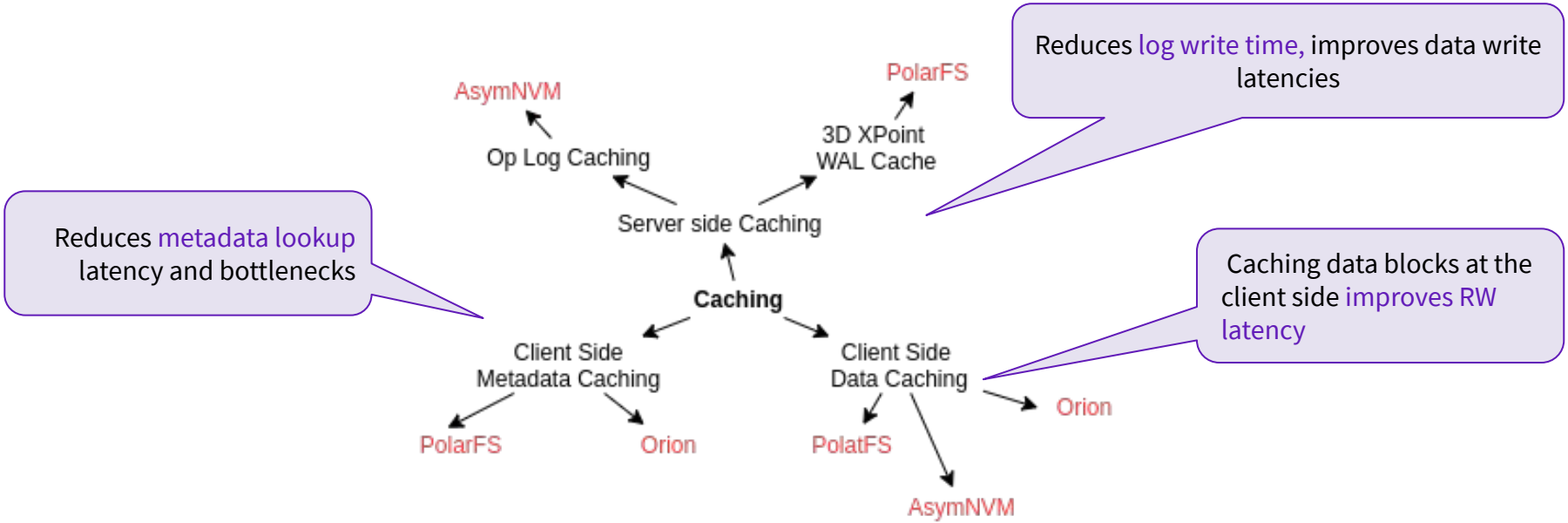
Metadata Management



Key Encoding reduces flexibility of the system to manage storage load.

Central server lies on the critical path for data Read/write increasing IO latency.

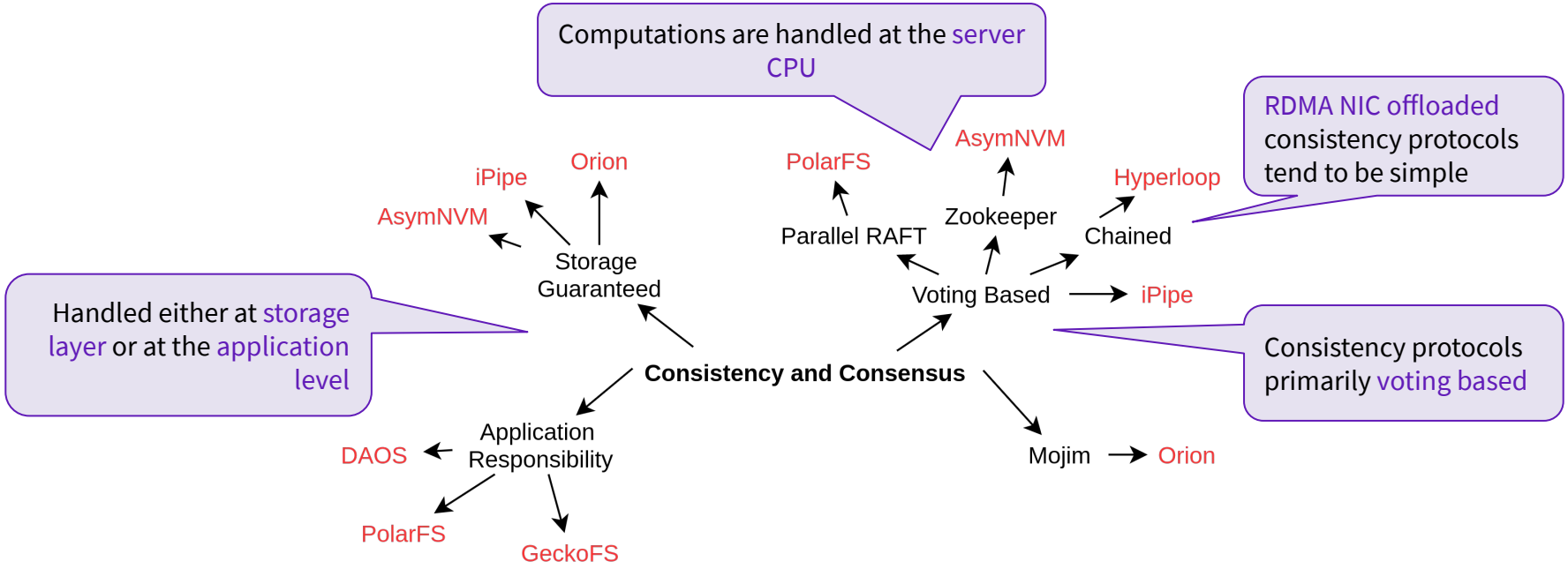
Cache Management



Caching increases system complexity, are CPU managed. Increases resource needs.

Some attempt has been made to offload KV caching to the NIC for hot keys.

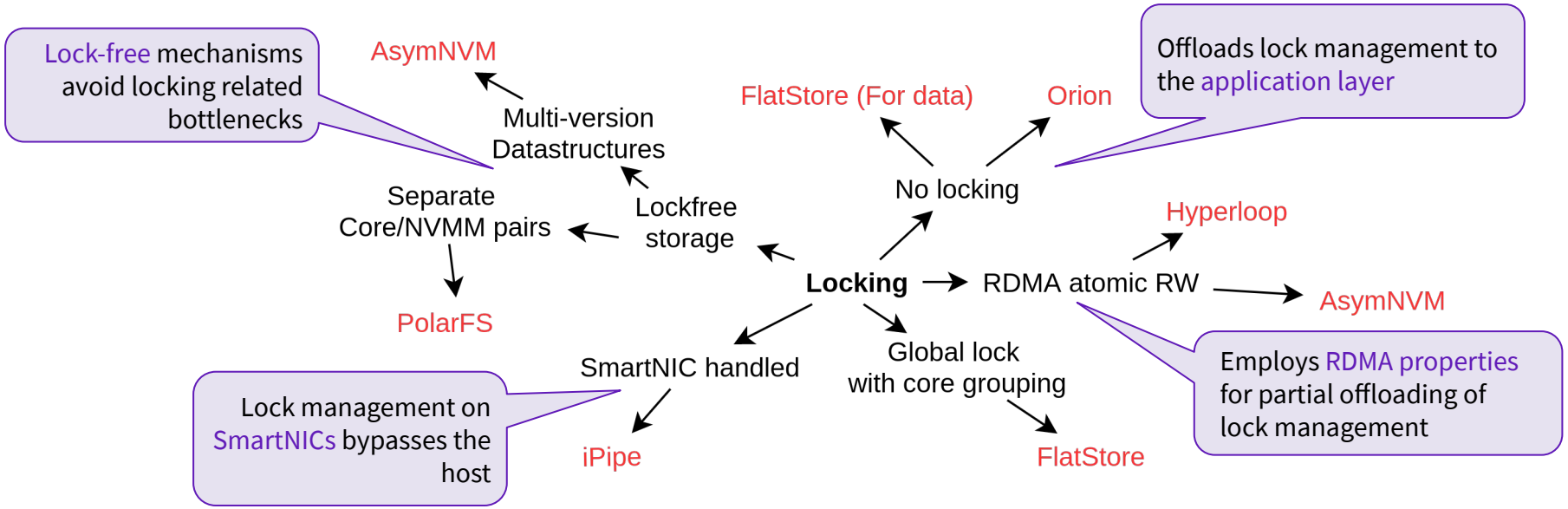
Consistency and Consensus



Consistency Management at the CPU wastes resources

RDMA NICs due to limited expressibility allows only simple protocols.

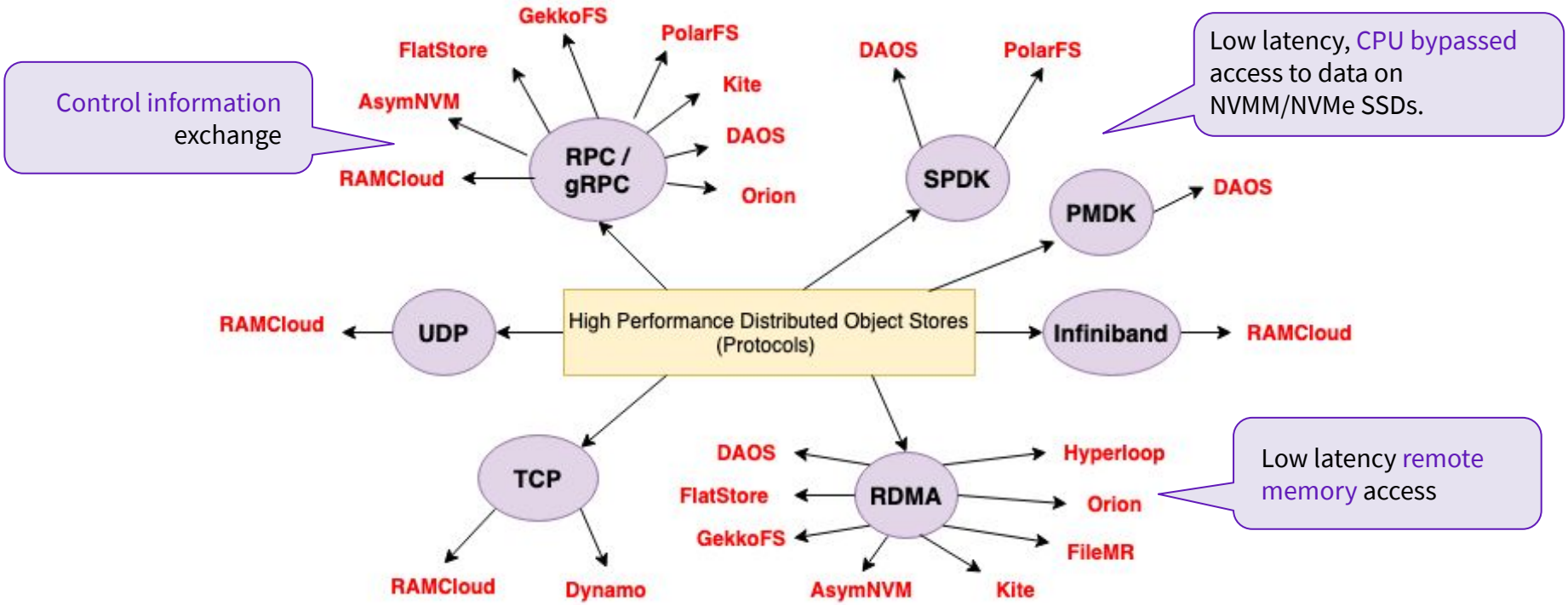
Transaction and Lock Management



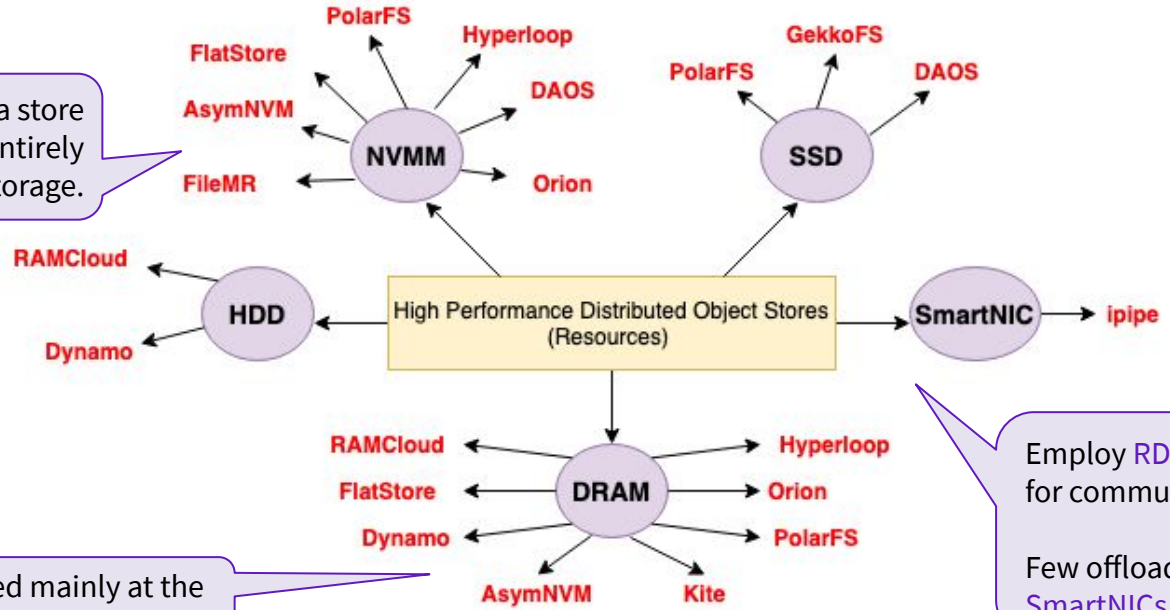
Solutions with locking support tend to offload mechanism to the NIC.

Transaction support unanimously uses log based techniques like WAL.

Communication Protocols Employed



Hardware Resources Employed



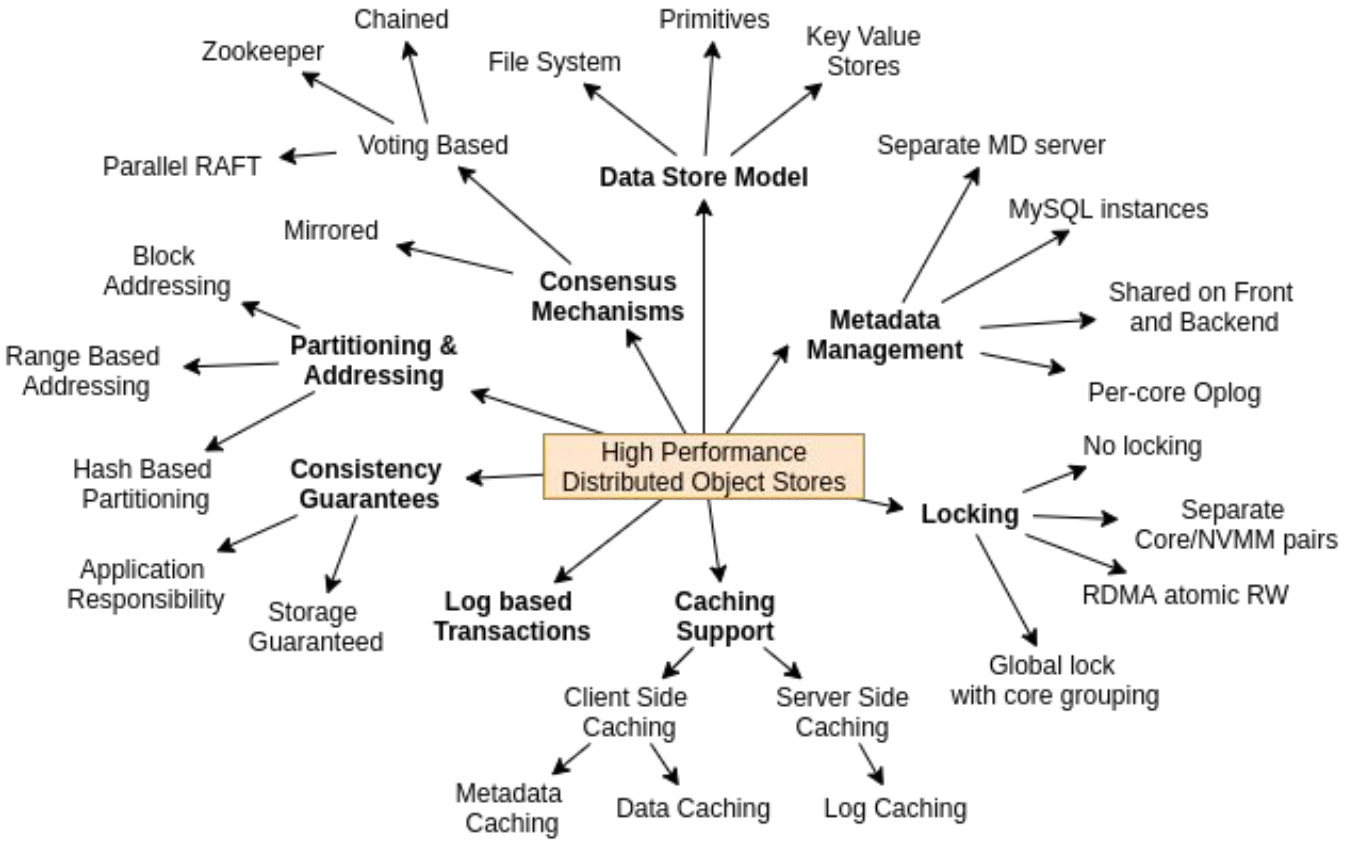
NVMM as data store sometimes entirely eliminating block-storage.

Cache is managed mainly at the DRAM.

Employ RDMA capability of NICs for communication

Few offload computation on SmartNICs.

A comprehensive view of all systems



Outline

Setting the Context

Survey of Existing Techniques

Technology Trends and Impacts

Action plan

Resolving the storage puzzle

SmartNIC Capabilities [1]	ASIC-based	SoC-based
#CPU cores	200 +	50 +
Latency	low	high
Bandwidth (in Gbps)	< 200	< 100
Special programming lang?	✓	x
Onboard DRAM	< 2 GB	< 16 GB

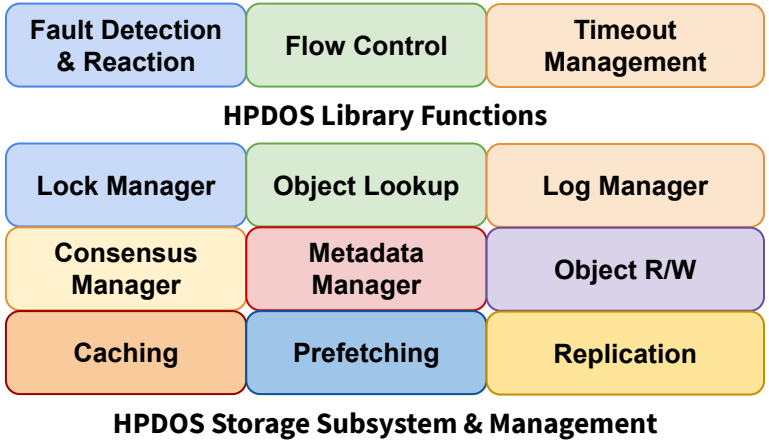
✚

RDMA capabilities
Zero-copy
Kernel bypass
No CPU involved

✚

Memory Capabilities [2]	DRAM	NVRAM	SCM	SSD
Latency (in µs)	~ 0.08	~ 5	~ 20	~ 100
Non-volatile?	x	✓	✓	✓
Symmetrical R/W?	✓	✓	x	x
Capacity (in TB)	< 0.064	< 0.48	< 32	< 8

Exploit technology capabilities to accelerate HPDOS performance



[1] Sean Choi et al. "λ -NIC: Interactive Serverless Compute on Programmable SmartNICs". arXiv 2019.

[2] Jakob Lüttgau et al. "Survey of Storage Systems for High-Performance Computing." *Supercomputing Frontiers and Innovations* [Online], 5.1 (2018): 31-58. Web. 8 Dec. 2020

Outline

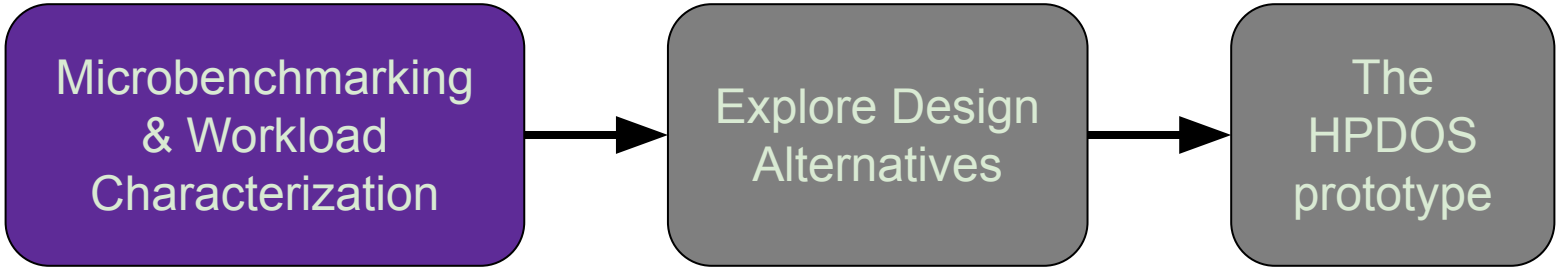
Setting the Context

Survey of Existing Techniques

Technology Trends and Impacts

Action plan

Action Plan


Microbenchmarking & Workload Characterization
Explore Design Alternatives
The HPDOS prototype

Microbenchmarking & Workload Characterization

Performance Characterization of SmartNICs	Micro-benchmarking of SmartNICs
Characterization of HPDOS workloads	Use HPC workloads & benchmark tools

Expected Outcomes

- Performance characterization of SmartNICs for HPDOS workloads
- Suggestions for policy design

Performance Characterization of SmartNICs

Sample questions for Micro-benchmarking

1. What is per-function resources (NIC and/or CPU) to performance mapping? [+] [*]
   a. With interfering NIC workloads
   b. With interfering CPU workloads
2. What is the impact of queuing discipline?
3. What is the (dynamic) capacity of a SmartNIC?*
4. If we employ resource reservation to solve noisy neighbor problem, what are the tradeoffs?*

* for both, SoC-based & ASIC-based NICs

[+] Performance metrics
- Throughput
- Latency
- Request drop ratio

Workload characterization

1. Modelling workload metrics and parameters
2. HPC Application workloads
    a. AI/ML
    b. Genomics
    c. Financial risk modelling
    d. Weather modelling etc.
3. Benchmarking tools - IOR, IOR 500, YCSB, COSbench
4. Drawbacks
    a. Wide variety of workloads for any single criteria of applications
    b. High overhead for workload tools
    c. Contrast in synthetic workloads and real time application workloads

Next Steps ...

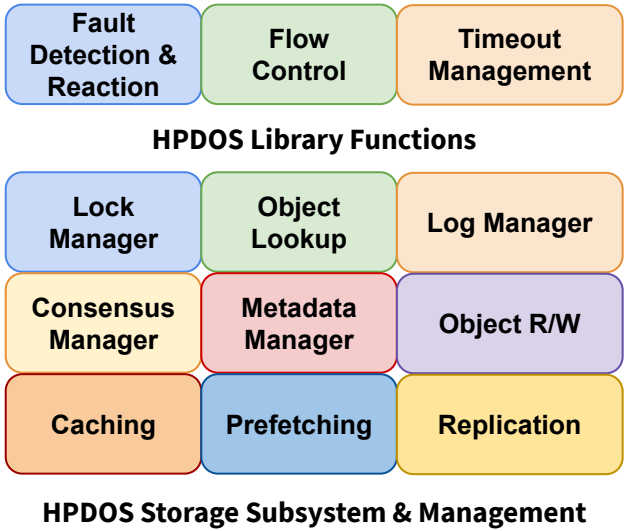
Microbenchmarks	Capturing traces	Generate Workloads	Workload targets

• Measure performance for individual components
  ○ Smart NIC
  ○ Memory etc.
• Identify bottlenecks early on.
	• Run real time applications
	• Capture the mix of operations
	• On which components certain types of requests are more.
		• Use traces to define application workloads
		• Generate **application specific synthetic** workloads
			• HPC workloads
			• AI/ML workloads

Action Plan


Microbenchmarking & Workload Characterization
Explore Design Alternatives
The HPDOS prototype

Identify Offloadable Storage Functions

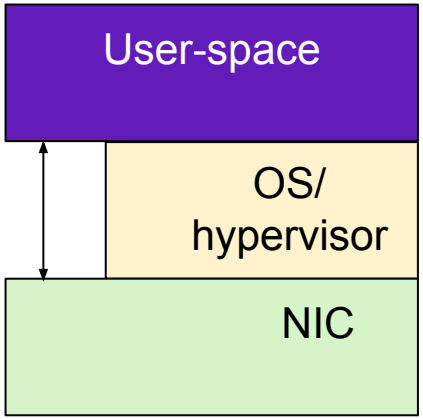


What storage functions can leverage SmartNICs for performance?

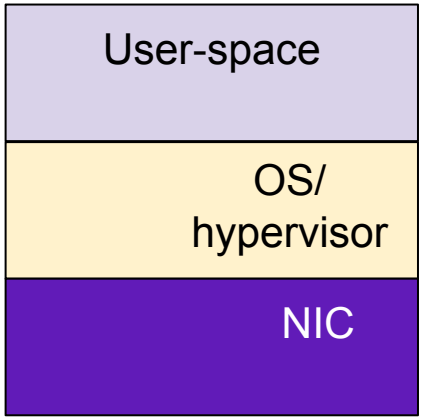
Design Questions

1. What storage functions can benefit with
   a. SmartNIC storage?
   b. SmartNIC compute?
2. How much do we offload?
   a. complete/partial
3. When do we offload?
   a. always/workload-specific/ ...
4. Where do we offload?
   a. source/dest smartNIC, ToR switch
5. How do the storage functions interact with each other?

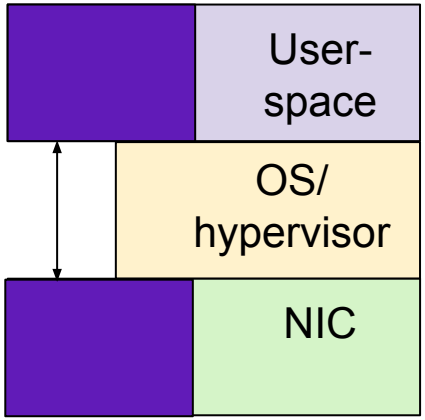
Exploring Design Alternatives



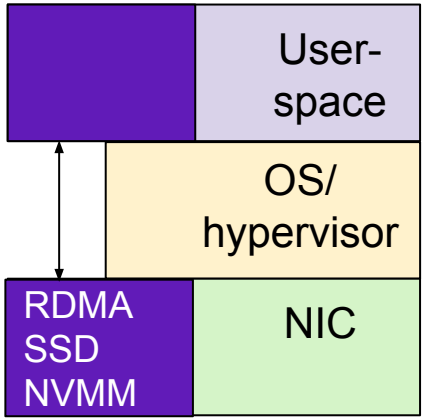
(A) Storage function runs in user-space (with/wo kernel bypass)



(B) Storage function runs on the NIC



(C) Storage function partially offloaded to NIC



(D) Storage function uses NIC capabilities (RDMA/SSD/NVMM)

HPDOS Storage Function

Explore performance tradeoff & cost for each design alternative!

HPDOS Design Directions

Hierarchical Caching

Hierarchical Caching
- Cache outside of DRAM? SmartNIC?
- Cache outside of host? ToR switch?

Challenges
- Cache coherency
- Limited NIC memory; design policies

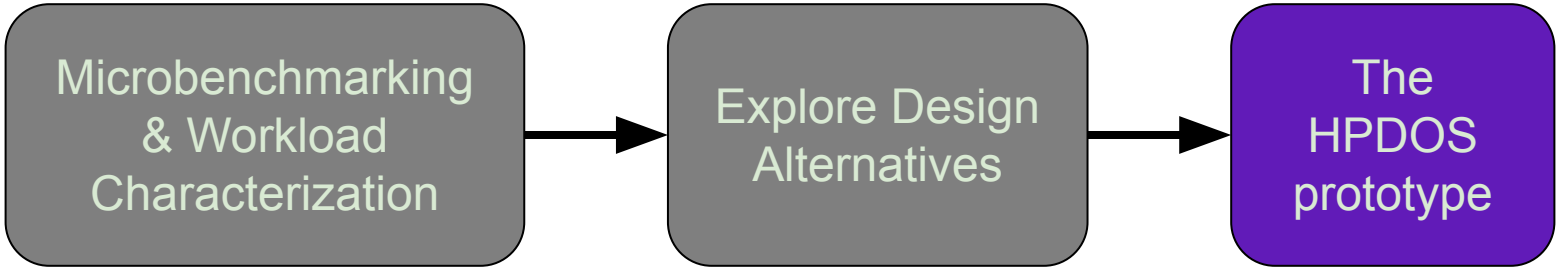
Prefetching of Metadata & Data

Prefetching of Metadata & Data
- What object type to prefetch?
- Prefetch destination?
  - SmartNIC/DRAM/NVRAM/SSD
- Prefetch from which replica?
- Where does prefetch logic run?

Distributed Storage Protocols

Distributed Storage Protocols
- How much compute to offload?
- Dynamic offload decisions
  - Overheads?
  - Decision-making period?
- Multi-tenant support; how?

Action Plan


Microbenchmarking & Workload Characterization
Explore Design Alternatives
The HPDOS prototype

Building the HPDOS System Prototype

1. Choose (dynamically) the right design option for each storage function
    a. What to offload (or not)? How much? When? Where?
2. Choose (dynamically) the interaction mechanism between the components
    a. Based on the outputs from (1)
3. Design policies for:
    a. Optimum performance
    b. Efficient resource utilization

Thank You

Saksham Goel — saksham.goel@cse.iitb.ac.in
Nilanjan Daw — nilanjandaw@cse.iitb.ac.in
Rinku Shah — rinku@cse.iitb.ac.in
Pramod S Rao — pramodrao@cse.iitb.ac.in
Umesh Bellur — umesh@cse.iitb.ac.in
Purushottam Kulkarni — puru@cse.iitb.ac.in

Systems and Networks Research Group
Department of Computer Science & Engineering
Indian Institute of Technology Bombay

`https://git.cse.iitb.ac.in/synerg/hpdos`

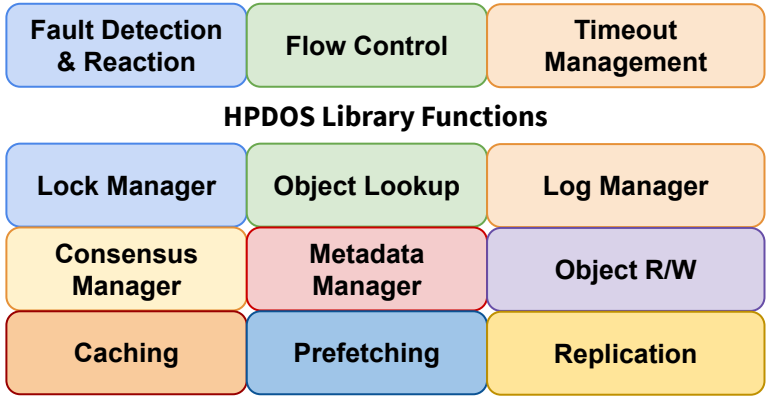
Backup slides from here

Resolving the storage puzzle

SmartNIC Capabilities [1]	ASIC-based	SoC-based
#CPU cores	200 +	50 +
Latency	low	high
Bandwidth (in Gbps)	< 200	< 100
Special programming lang?	✓	x
Onboard DRAM	< 2 GB	< 16 GB

RDMA capabilities
Zero-copy
Kernel bypass
No CPU involved

Memory Capabilities [2]	DRAM	NVRAM	SCM	SSD
Latency (in us)	~ 0.08	~ 5	~ 20	~ 100
Non-volatile?	x	✓	✓	✓
Symmetrical R/W?	✓	✓	x	x
Capacity (in TB)	< 0.064	< 0.48	< 32	< 8



Interconnect map	Compute + Storage
SmartNIC	NIC CPU + NIC storage
RPC, RDMA	Server CPU + DRAM
SPDK	SSD (no CPU)
PMDK	NVRAM (no CPU)

[1] Sean Choi et al. "λ -NIC: Interactive Serverless Compute on Programmable SmartNICs". arXiv 2019.
[2] Jakob Lüttgau et al. "Survey of Storage Systems for High-Performance Computing." *Supercomputing Frontiers and Innovations* [Online], 5.1 (2018): 31-58. Web. 8 Dec. 2020

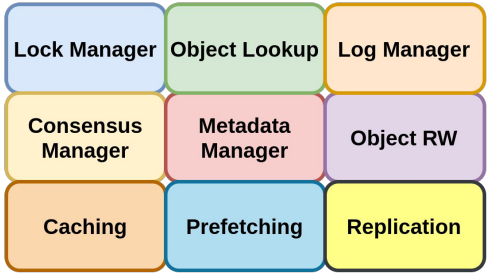
Resolving the storage puzzle

SmartNIC capabilities	ASIC-based	SoC-based
#CPU cores	200 +	50 +
Latency	low	high
Bandwidth (in Gbps)	< 200	< 100
Special programming lang?	✓	x
Onboard DRAM	< 2 GB	< 16 GB

Table 1: SmartNIC capabilities[1]

Interconnect technologies	Compute/Storage
SmartNIC	NIC CPU + NIC storage
RPC, RDMA	Server CPU + DRAM
SPDK	SSD (no CPU)
PMDK	NVRAM (no CPU)

Table 2: Interconnect mapping

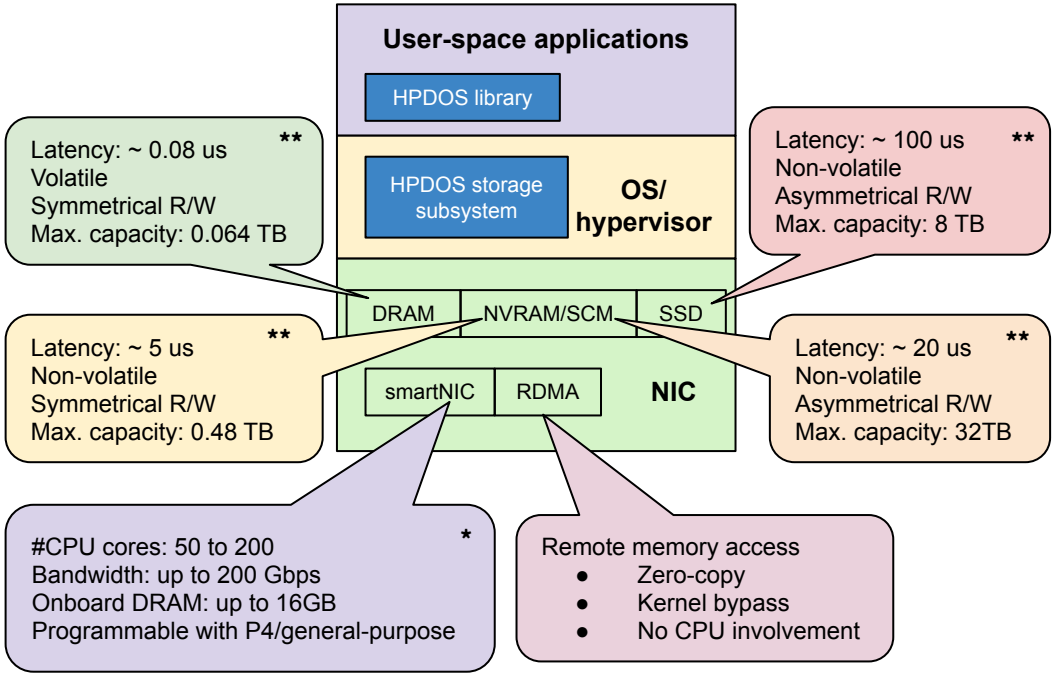


Memory capabilities	DRAM	NVRAM	SCM	SSD
Latency (in us)	~ 0.08	~ 5	~ 20	~ 100
Non-volatile?	x	✓	✓	✓
Symmetrical Read/Write?	✓	✓	x	x
Capacity (in TB)	< 0.064	< 0.48	< 32	< 8

Table 3: Capabilities of memory technologies[2]

[1] Sean Choi et al. “λ -NIC: Interactive Serverless Compute on Programmable SmartNICs”. arXiv 2019.
[2] Jakob Lüttgau et al. "Survey of Storage Systems for High-Performance Computing." *Supercomputing Frontiers and Innovations* [Online], 5.1 (2018): 31-58. Web. 8 Dec. 2020

Need of the hour: Low Latency and High Throughput



Distributed Object Storage requirements

- Fast storage access — RDMA, NVMM, SCM, SSD?
  - Object R/W
  - Metadata lookup
- Fast processing — Offload to SmartNIC?
  - Prefetch/caching policies
  - Consensus/consistency
  - Transaction management
- Fast communication — SmartNIC?
  - Fault detection & reaction
  - Consensus/consistency/locking

To improve DOS efficacy, use newer technologies?

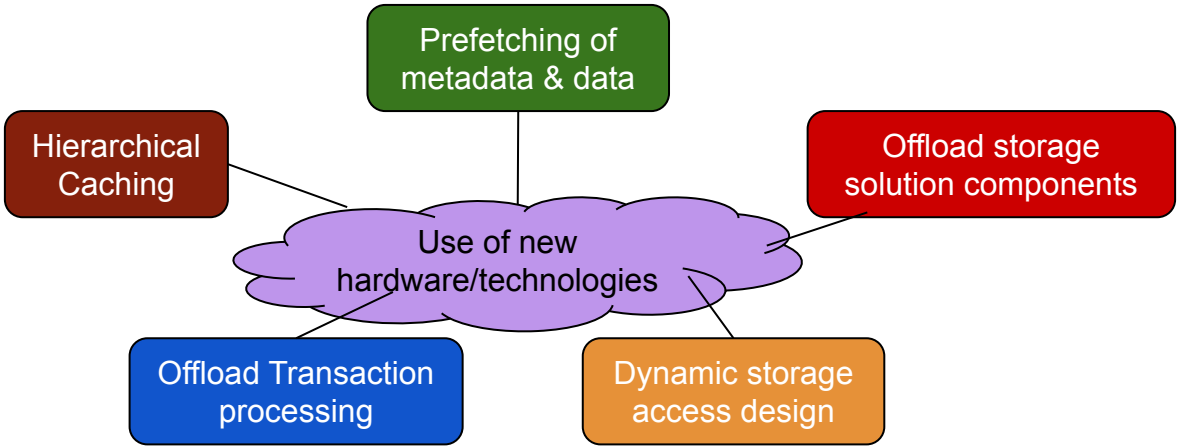
References:
* Sean Choi et al. "λ -NIC: Interactive Serverless Compute on Programmable SmartNICs". arXiv 2019.
** Jakob Lüttgau et al. "Survey of Storage Systems for High-Performance Computing." *Supercomputing Frontiers and Innovations* [Online], 5.1 (2018): 31-58. Web. 8 Dec. 2020

Future Design

Hierarchical Caching
- Cache outside of DRAM? SmartNIC?
- Cache outside of host? ToR switch?

Challenges
- Cache coherency
- Limited NIC memory; design policies

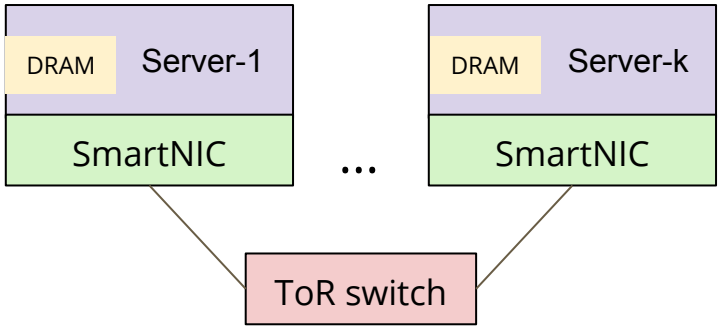
Prefetching of metadata & data
- What object type to prefetch?
- Prefetch destination?
  - SmartNIC/DRAM/NVRAM/SSD
- Prefetch from which replica?
- Where does prefetch logic run?



Hierarchical Caching

Design questions

- Leverage smartNIC memory to cache storage objects (data/metadata)
  - How much should we cache on a single NIC?
- Can we out-perform the host's caching capacity?
- What are the challenges?
  - Cache coherency between host and smartNIC
  - Limited on-board NIC memory; design policies
    - Provisioning
    - Ageing
    - Eviction of storage objects
  - Duality requires policy decisions
    - Local requests prefer server DRAM cache
    - Remote requests prefer smartNIC cache



Notes: Figure may not be necessary

Prefetching Metadata and Data

Design questions

- What object types should be prefetched?
  - For example, data vs. metadata
- What is the destination of prefetch? smartNIC, DRAM, NVRAM, or SSD?
  - Incoming requests => SmartNIC
  - Outgoing requests => DRAM cache
- Out of cache?
  - NVRAM for small objects; SSD for large objects
- Which replica should be used for prefetching?
  - Load balancing between replicas
  - Avoid network congestion
- Where does the prefetch logic reside? CPU, smartNIC, or ToR switch?

Prefetching is best-effort; limit prefetch under high load conditions

Offloading Storage Solution Components

Design questions

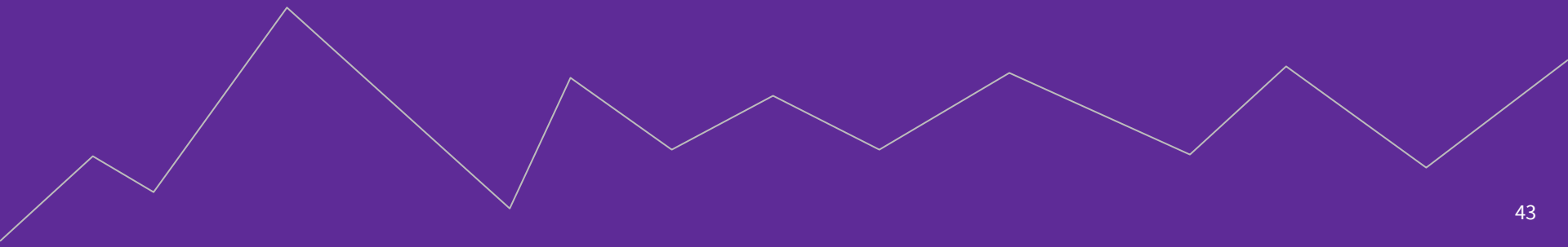
- How much compute should be offloaded to the smartNIC?
  - most frequently accessed functions?
- Request characteristics are dynamic; offload decisions should be dynamic
  - What are the performance benefits of dynamic offload?
  - What are the overheads?
  - What should be the decision-making period?
- How do we manage resource provisioning and partitioning in case of multitenancy?

Collaboration between the Resources

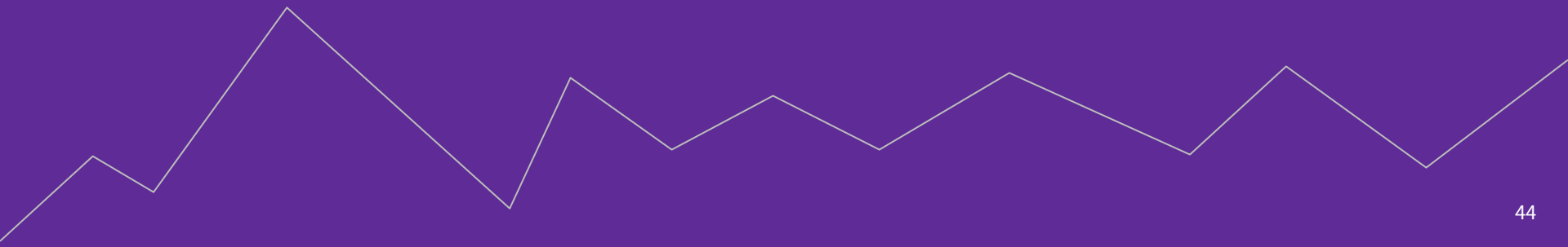
Design questions

- Can we offload transaction processing to smartNIC?
  - Offload transport protocol processing (gRPC) and use RDMA to fetch data to NIC memory
- How do we dynamically utilize the storage access methods? SmartNIC, RDMA, and CPU?
  - Design policies
  - Provide control knobs to dynamically provision work between them

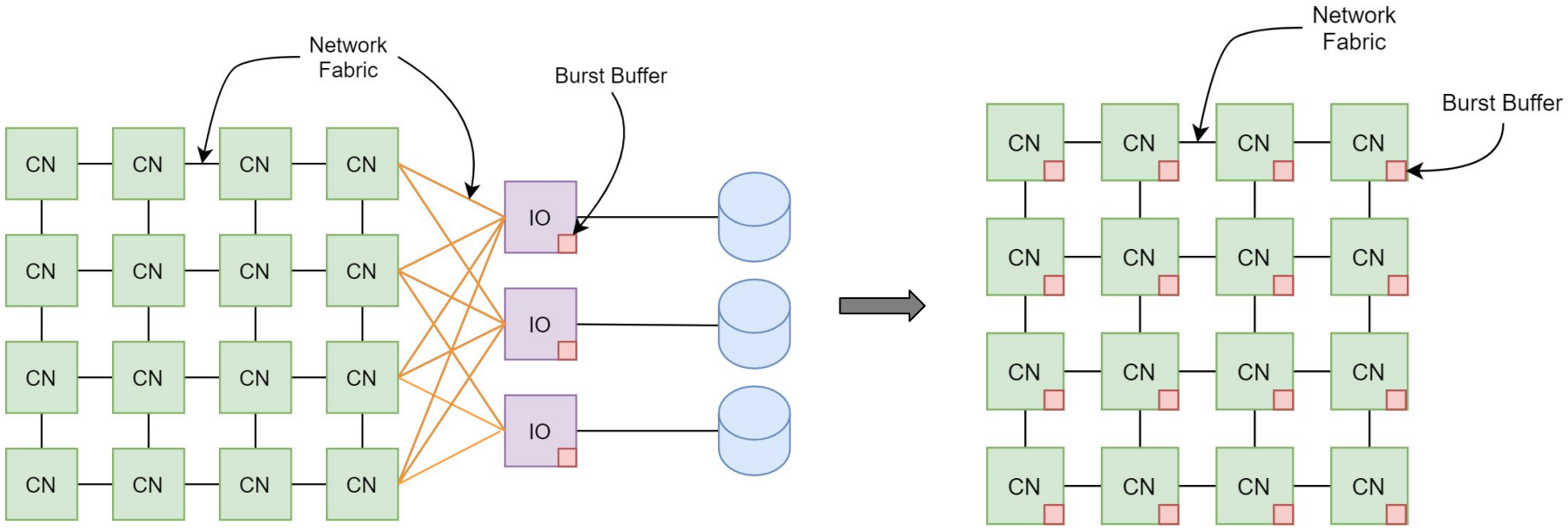
Motivation and Setup



Workloads

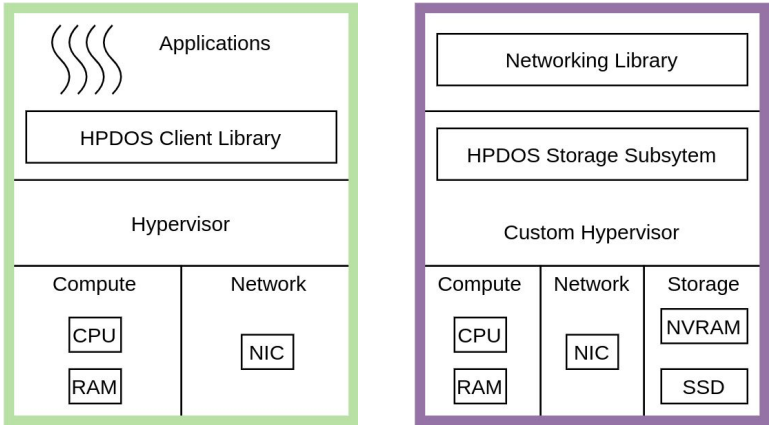


Architectural Continuum: Towards Hyperconvergence

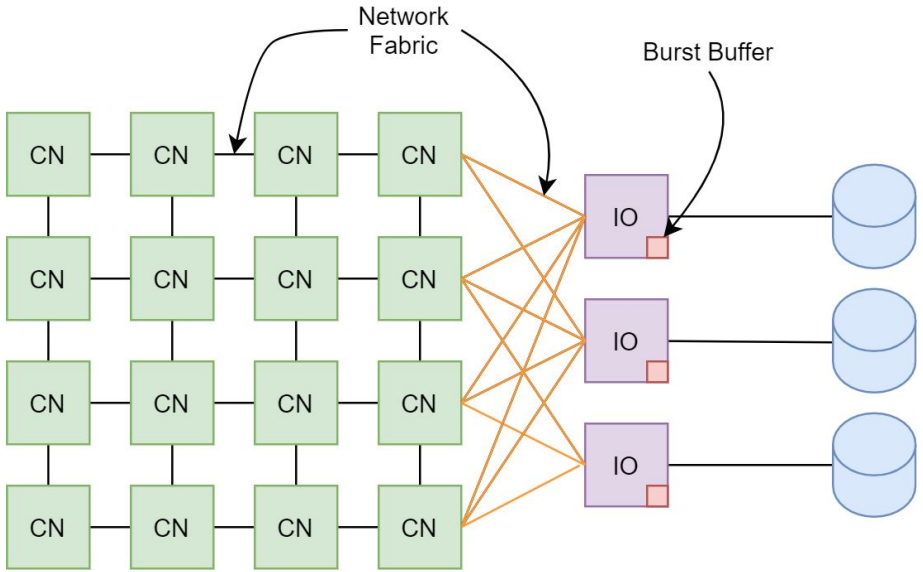


* IBM Blue Gene

Setup : Shared I/O Nodes



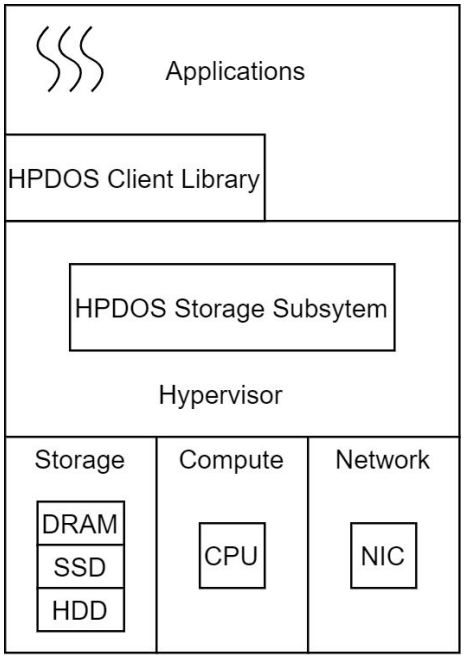
Node Architecture



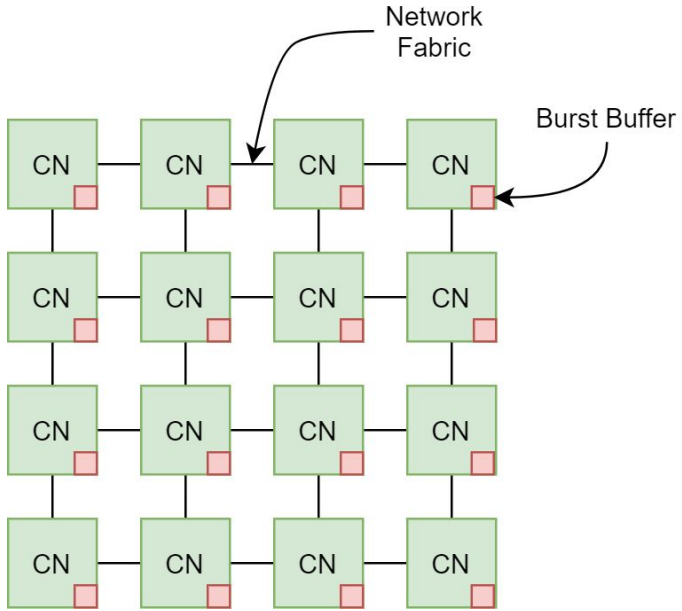
Network Cluster

* IBM Blue Gene

Setup : Hyperconverged

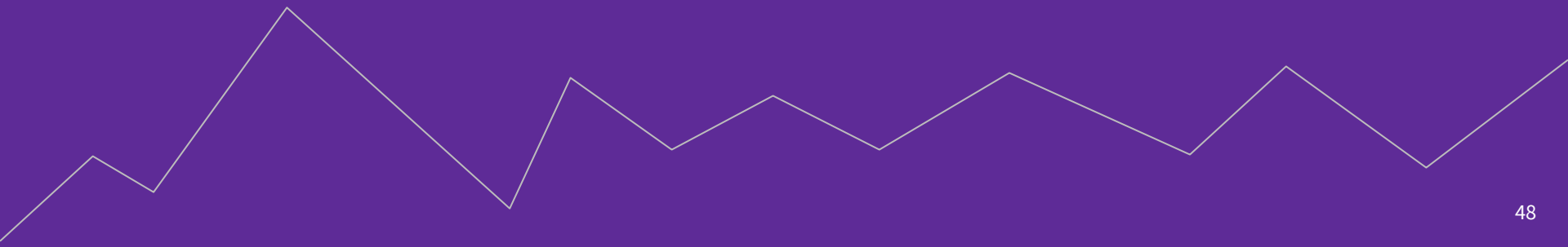


Node Architecture

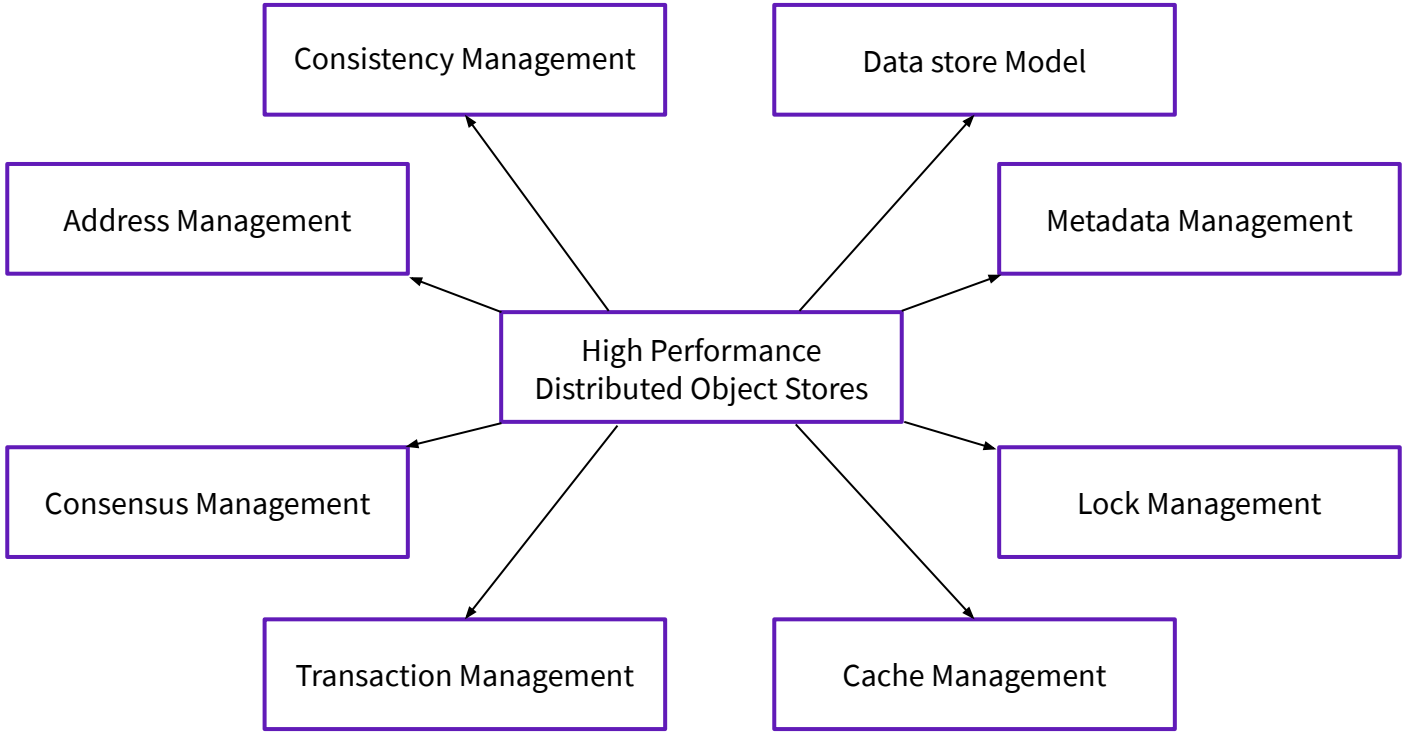


Network Cluster

Survey of Techniques

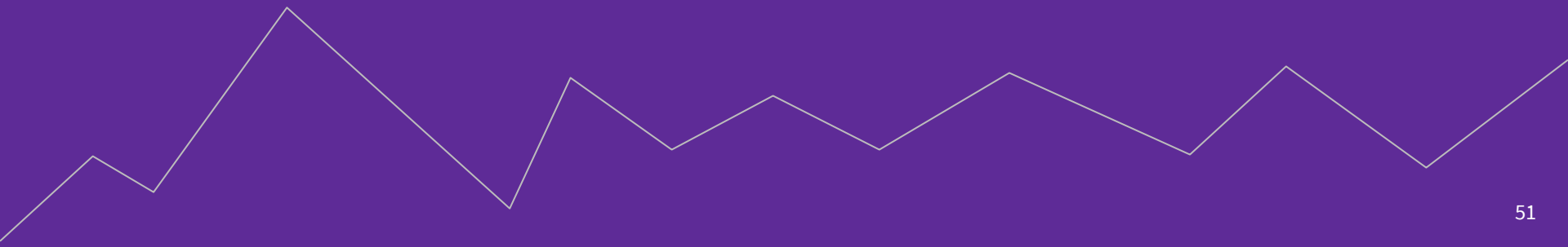


Structuring Existing Research in HPDOS

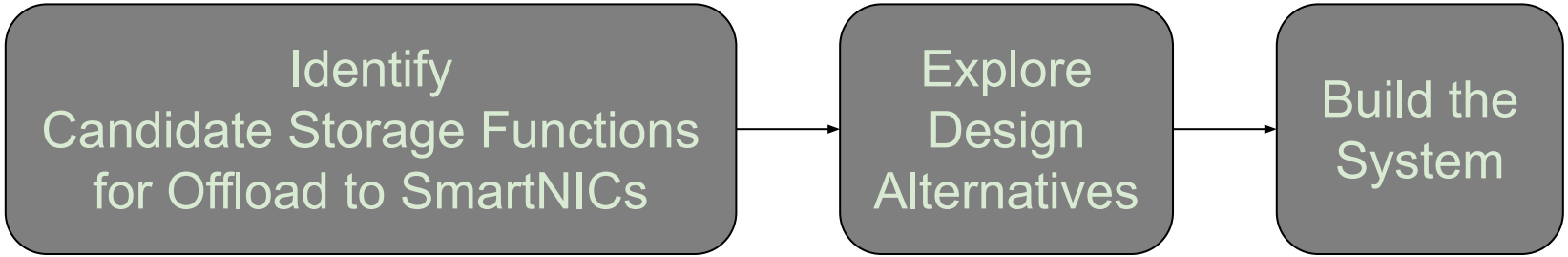


Technology Trends
&
Storage solutions

Next steps



Action Plan


Identify
Candidate Storage Functions
for Offload to SmartNICs
Explore
Design
Alternatives
Build the
System

Workloads

- Existing benchmark tools
  - COSbench
  - YCSB
  - IOR
- Drawbacks
- Next Steps
  - Workload Characterization
  - HPC application workloads
  - HPC workload example

IOR (Interleaved OR Random)

01 **Features**

- Parallel I/O Benchmark for distributed file systems
- IOR 500 uses IOR
- Mainly for HPC workloads
- Measures metadata performance for file systems

02 **Pros**

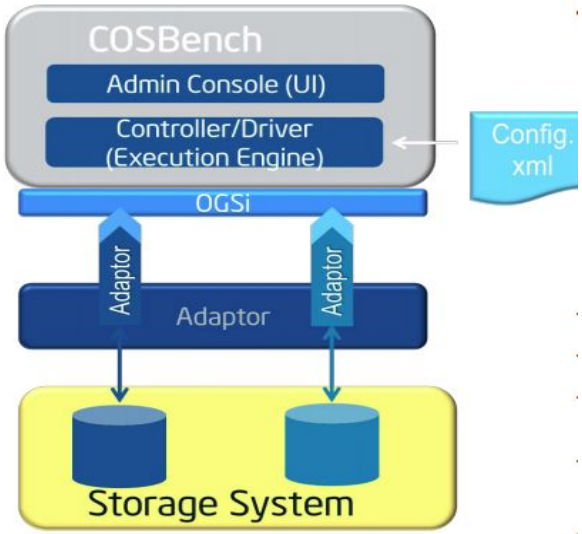
- Extensively tunable benchmark parameters
- Optimal for Synthetic HPC workloads
- Ability to perform microbenchmarks as well
- Becoming a De facto standard way to measure the HPC's I/O capability for clusters

03 **Cons**

- Does not provide distribution models

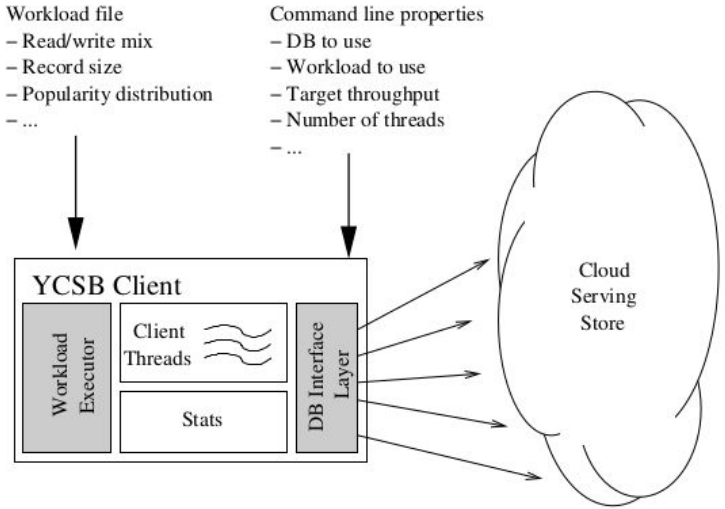
COSbench

1. Cloud object storage benchmarking
2. Distributed load testing framework
3. Pros:
   a. GUI to load and run workloads
   b. Tunable workloads (Object sizes, load balancing etc.)
   c. Web-based real time performance monitoring
4. Cons:
   a. Does not support benchmarking for file systems.



YCSB (Yahoo! Cloud serving benchmark)

1. Key-value and cloud serving benchmark
2. Consists of pre-defined core workloads
3. Pros:
    a. Easy to use and compare multiple storage systems.
    b. Tunable workloads with multiple distribution models
4. Cons:
    a. Network intensive to use.



IOR (Interleaved OR Random)

1. Parallel I/O benchmarking for distributed file systems.
2. IOR 500
    a. uses IOR
    b. de facto standard way to measure the HPC's I/O capability for clusters
3. Pros:
    a. Measures performance of metadata
    b. Extensively tunable for benchmark parameters
    c. Optimal for HPC workloads
4. Cons:
    a. Does not provide distribution models

Drawbacks

1. Wide variety of workloads for any single criteria of applications
    a. Makes it difficult to characterize different types of workloads
2. High overheads for workload tools
3. Difference in Synthetic workloads and real time applications

HPC Workloads

1. Wide variety of characteristics for different HPC applications
2. Mainly FLOPs and integer operations
3. HPC workloads exhibit reuse patterns
4. HPC Application workloads
    a. Genomics
    b. Weather modelling
    c. Financial risk modelling
    d. AI/ML etc.

HPC genomics workload

1. Lots of variation in workflows used for research
   a. Different sequencers, applications and workflows
2. FDA approved diagnostic tools result in more workflows
3. Typical I/O properties observed in a genomics HPC workload
   a. Highly CPU intensive
   b. Consists of millions of small to medium files
   c. Write intensive, predominantly sequential
   d. Read I/O is random access
   e. Probability for distribution of reads are uniform