Introduction to MadSys

YONGWEI WU

http://madsys.cs.tsinghua.edu.cn/publications.html

What is MadSys?

- MadSys (Mad System) group is doing cutting-edge system research, especially focuses on the design, implementation, evaluation and application of the parallel and distributed systems. The group belongs to the High Performance Computing Institute of Tsinghua CS Department.
- Our research interests include Runtime Environment, Data Storage/Management, Virtualization, Resource Management, Distributed Scheduling, Performance Analysis, Reliability/Fault Tolerant and other related topics.

Group Members:

Yongwei Wu

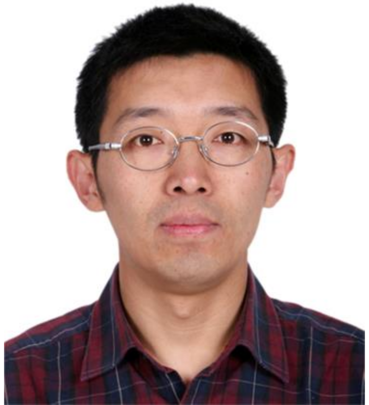
Professor, Vice Head

6-8 PhD Students

8-10 Master Students

Kang Chen

Associate Professor



Jinlei Jiang

Associate Professor

Recent Research Fields

- Distributed Storage
- Graph Computing
- Non-volatile Memory Systems
- Cloud Computing

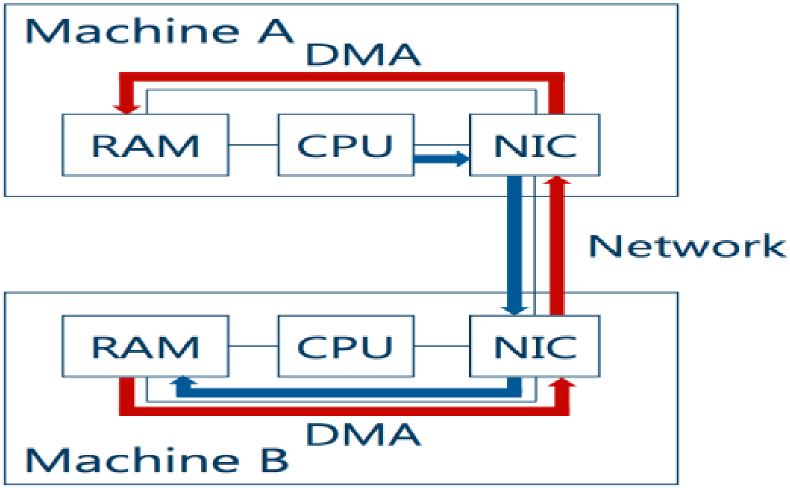
Distributed Storage

Our Research

1. RFP: Investigated the correct way of using RDMA for distributed (storage) systems
2. Triones: How to store data over multiple cloud providers for some specific benefit.
3. MeePo: Build a system for real-time group-data-sharing on demand in enterprise network
4. TStor: High-Scale (32+16/64+32) Erasure Code Enabled Distributed File System

1.1 RDMA devices are fast

- Apply Infiniband and RDMA into distributed storage systems
- InfiniBand: high performance networking hardware
- RDMA: Remote Direct Memory Access protocol
  - CPU/kernel bypassing and zero-copy
  - 2X~4X performance improvement compared with TCP/IP



FaRM, NSDI' 2014 RFP

1.2 Good Performance vs. Programmability

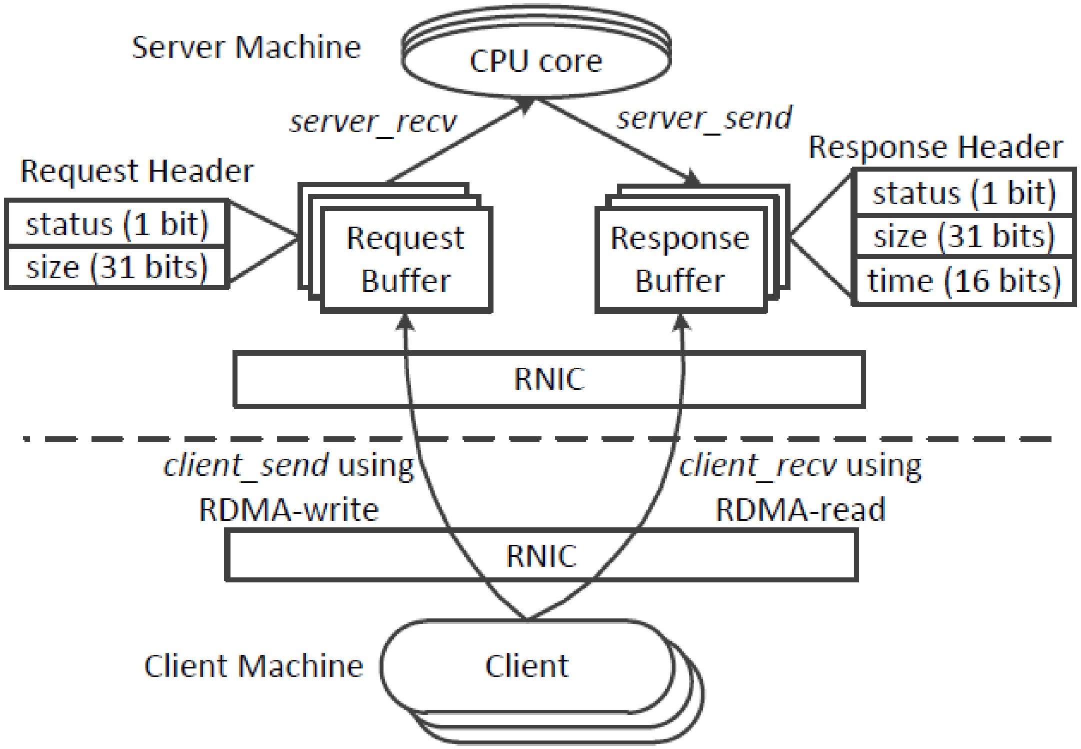
- Currently, two design paradigms are used using RDMA
- Server-Reply Paradigm
  - Just replace traditional TCP/IP socket interfaces with RDMA verbs
  - Good programmability but poor performance
- Server-Bypass Paradigm
  - Make use of bypassing features of RDMA and totally change the way TCP/IP works
  - Good performance but poor programmability

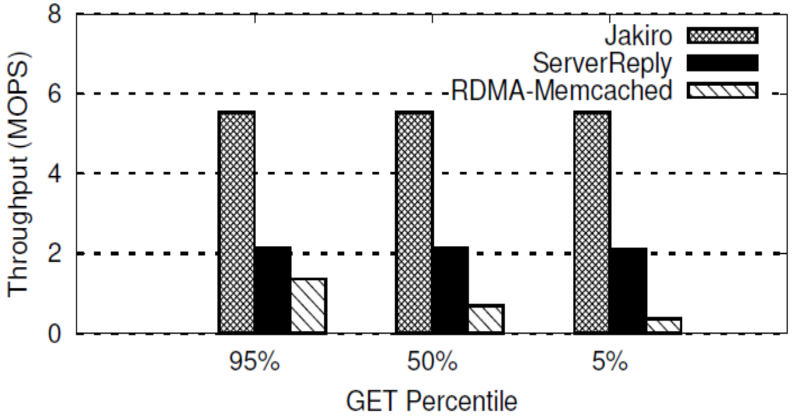
System developers have to trade-off between performance and programmability while choosing their way of programming from server-reply and server-bypass paradigms.

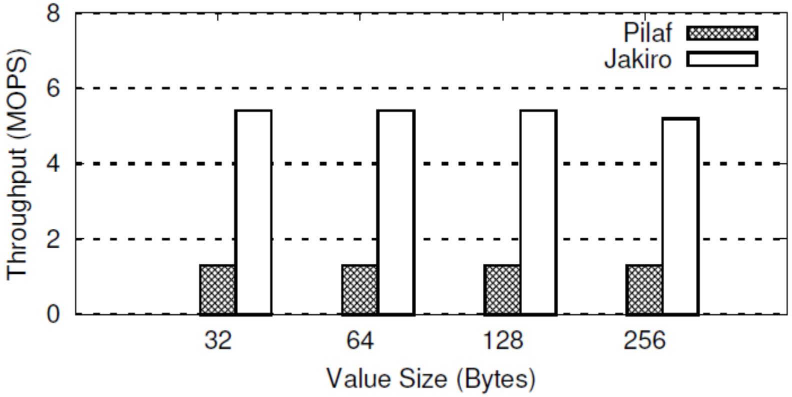
RFP

1.3 Remote Fetching Paradigm

- We present Remote Fetching Paradigm
  - Server process requests; Clients fetch results from server remotely
  - Achieve good programmability and high performance
  - 1.6X~4X performance improvement over server-reply and server-bypass paradigms







RFP

2.1 How to store data over multiple cloud providers

- Data grows exponentially in distributed storage systems
    - Cost of native data centers increase dramatically
- Non-private data is being transferred to public multi-cloud storage
    - Storage resources of public cloud storage providers (such as Amazon-S3 or Rackspace-Cloudfiles) are more cheaper
- Erasure Coding could be used to further reduce storage cost
    - Multi-cloud storage consists of erasure coding parameter (***n*,*k***) and ***n*** cloud storage providers
    - Tolerate the errors or unavailability of ***k*** providers

2.2 Why not build a model?

- Factors in multi-cloud storage with erasure coding are much complex

Vendor Lock-in

$$V = \frac{1}{n}$$

Fault-Tolerance Level

$$F = n - k$$

Latency

$$L_{X_{s,r}} = \frac{1}{k} \times \max_{1 \le i \le N} \{x_i \times l_{i,r}\}$$

Availability

$$A = 1 - \sum_{m=n-k+1}^{n} \sum_{j=1}^{\binom{n}{m}} \prod_{i \in \mathbb{N}_j} (1 - a_i) \prod_{i' \in \mathbb{N}'_j} a_{i'}$$

Cost

$$\begin{cases} C = \sum_{j=1}^{5} \sum_{i=1}^{T} \int x_i \times (Y_1 + Y_2)\, dt \\ Y_1 = \sum_{g=0}^{G_{i,j}-1} (D_{i,j,g} \times U_{i,j,g}(t)) \\ Y_2 = \left(Y_3 - \sum_{g=0}^{G_{i,j}-1} D_{i,j,g}\right) \times U_{i,j,G_{i,j}}(t) \\ Y_3 = \left(M_j \times R_j(t) + R'_{i,j}\right) \\ \overrightarrow{M} = (\frac{1}{k}, \frac{1}{n}, \frac{1}{k}, \frac{k}{n}, 1) \end{cases}$$

- Previous works do not consider the optimization issue, so they cannot optimally use cloud storage resources, which are charged by providers

Triones

2.3 Understand the problem systematically

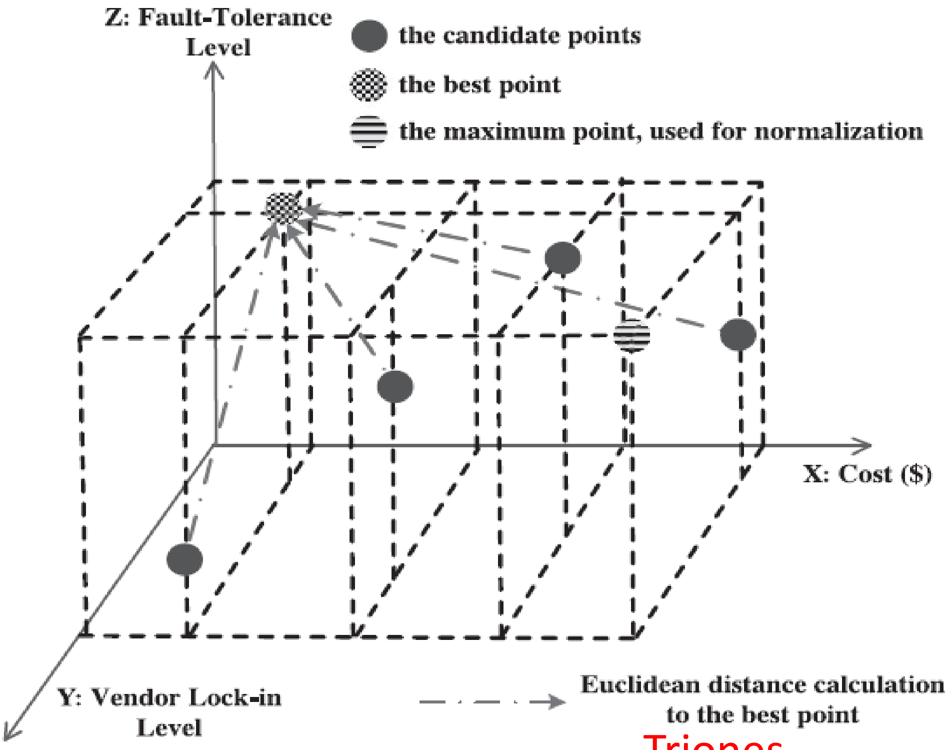
- We present Triones, a systematic model to address optimization issue in multi-cloud storage, and get your specific target: much lower latency with a little higher cost.
  - Non-linear programming to define data placement in multi-cloud storage
  - Get the optimum result throu
    ation

$$
\begin{cases}
Minimize & : & f_1(X,P) \\
 & : & f_2(X,P) \\
 & : & \cdots \\
Subject\ to & : & f_1^{'}(X,P) < Con_1 \\
 & : & f_2^{'}(X,P) < Con_2 \\
 & : & \cdots
\end{cases}
$$

$$X = \{x_1, x_2, \cdots, x_N, k\}$$

$$P = [\overrightarrow{P_1},\ \overrightarrow{P_2}, \cdots, \overrightarrow{P_N}]^{\mathrm{T}}$$

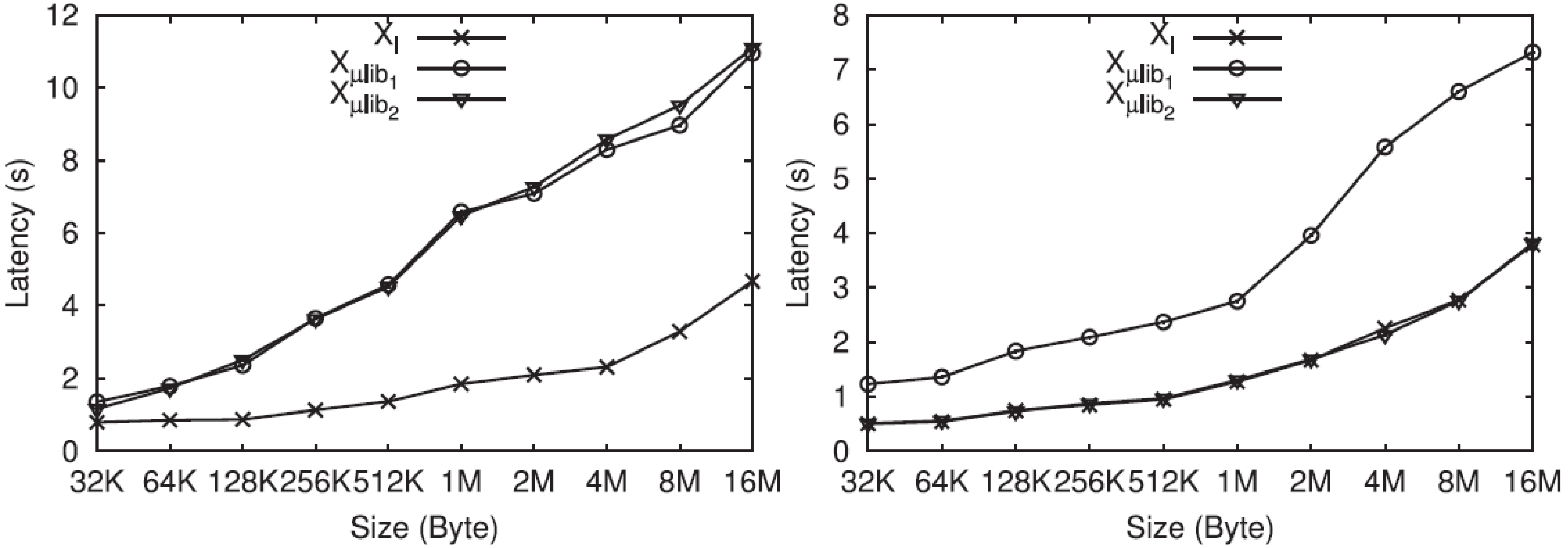
$$\overrightarrow{P_i} = (H_{i,1},\ H_{i,2},\ \cdots,\ H_{i,K})$$



Triones

2.4 The Effectiveness -- Latency

- Compared with random models， Triones reduces access latency by 50%



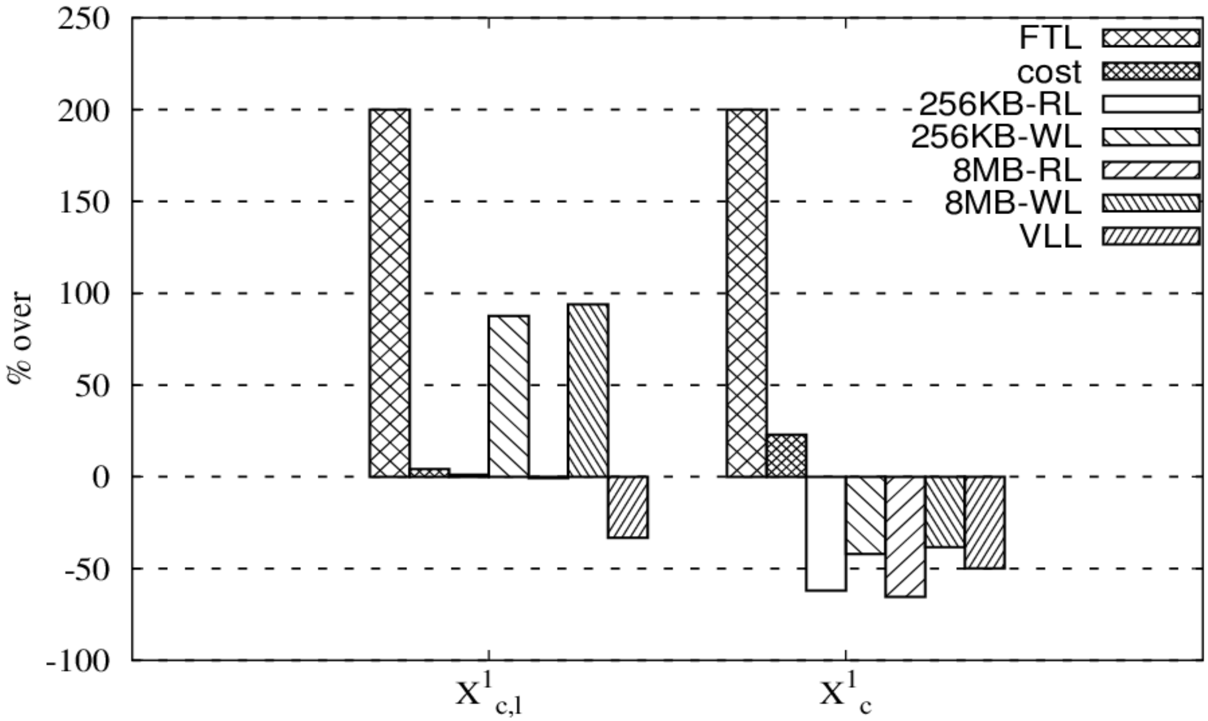
(a)Write

(b)Read

Triones

2.5 The Effectiveness -- Cost

- Compared with models for single-objective optimization，Triones improves fault-tolerance level by 200%, reduces vendor lock-in level by 49.85%, and reduces access latency by 30% to 70%, with only 22.97% more cost



FTL: Fault-Tolerance Level

RL: Read Latency

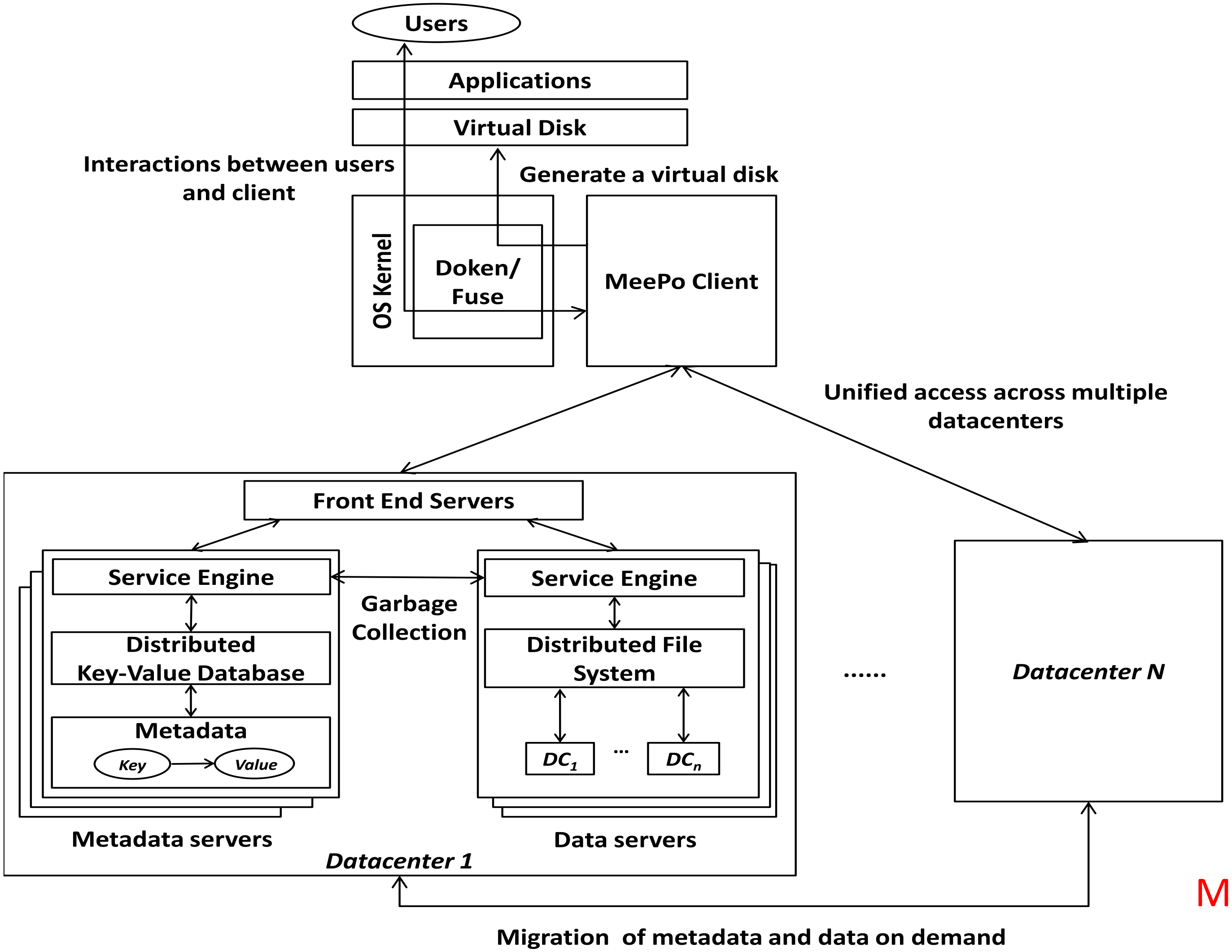
WL: Write Latency

VLL: Vendor Lock-in Level

Triones

3.1 Group Data Sharing Storage System

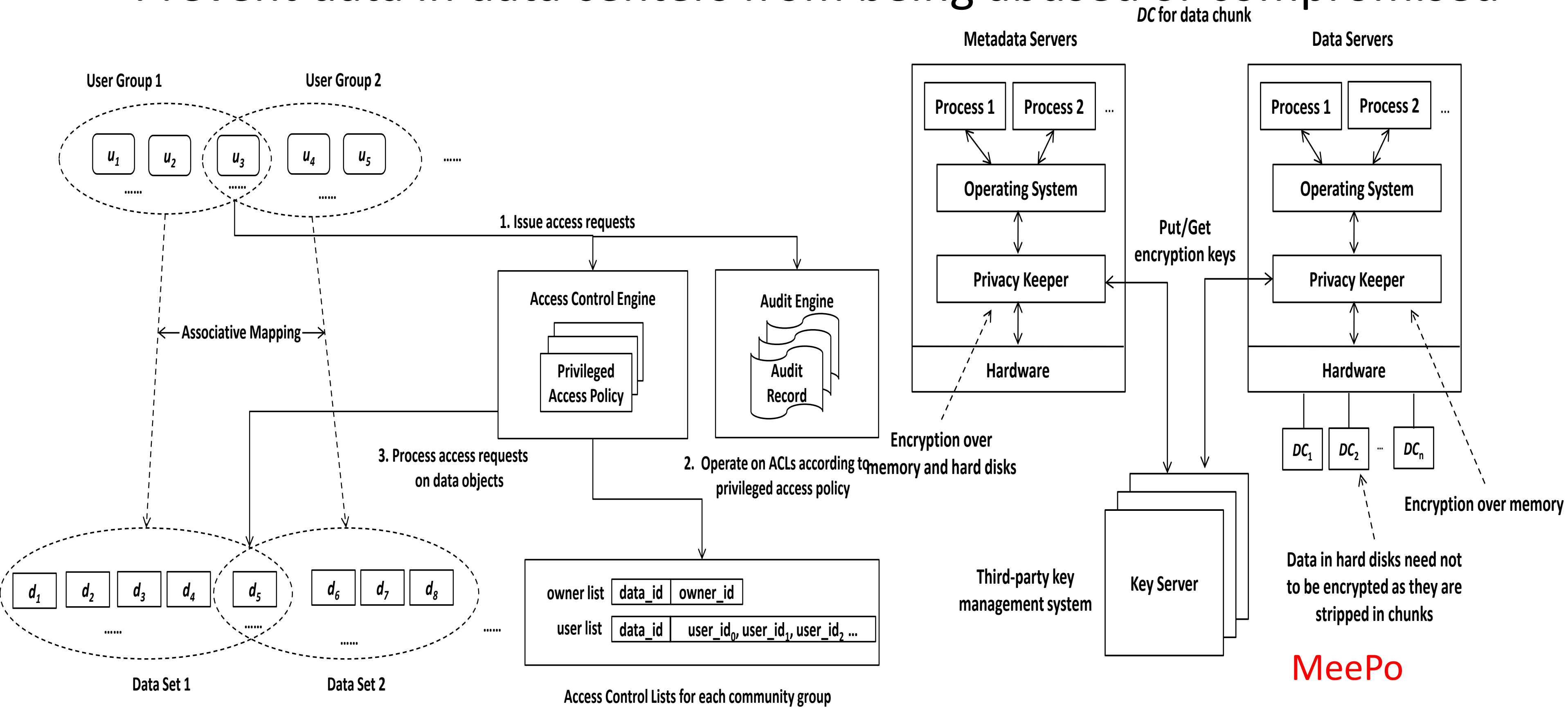
- We have designed and implemented MeePo to support group data sharing in enterprise with enterprise internal LAN



MeePo

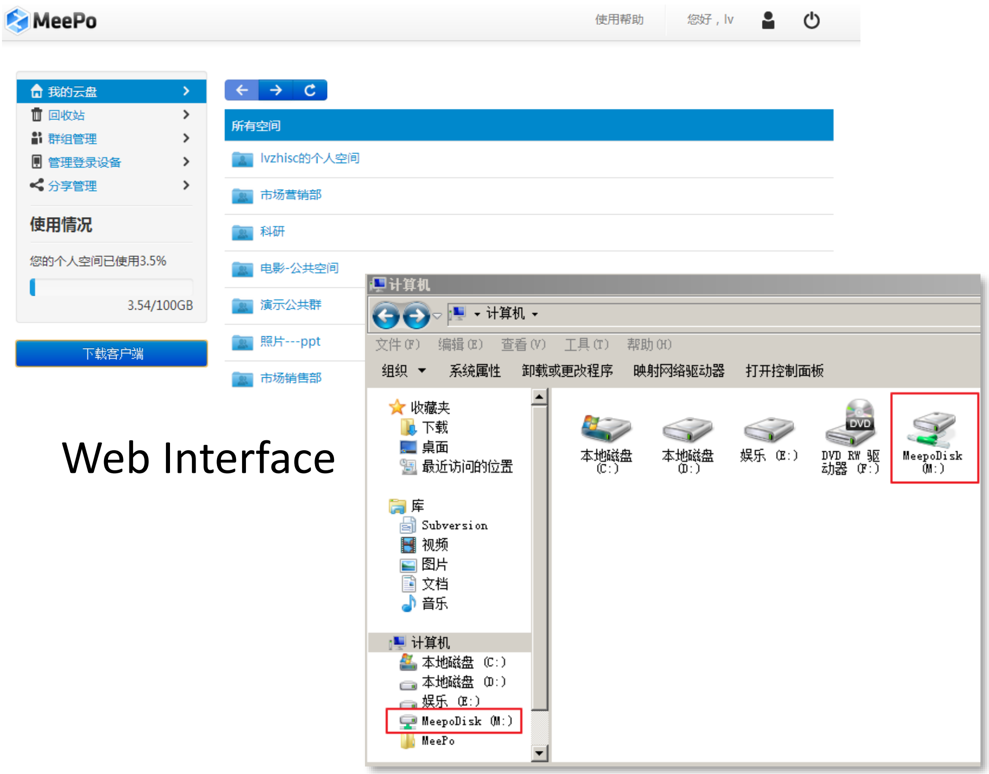
3.2 The MeePo Architecture

- Virtual Disk Mechanism
  - Sharing data is as convenient as operating files in native disks
- Privilege-Based Access Control
  - Prevent sharing data in groups from being disrupted
- Privacy Protection in Date Center
  - Prevent data in data centers from being abused or compromised



3.3 The MeePo System

- MeePo
  - Deployed in 51 universities/research institutes and 20 companies
  - The number of users registered or being served reaches 1,500,000, the number of groups created reaches 6,000
  - At Tsinghua University: 30,000 users, 600 groups, 500TB+ data



MeePo 使用帮助 您好，lv

我的云盘
回收站
群组管理
管理登录设备
分享管理

使用情况

您的个人空间已使用3.5%

3.54/100GB

下载客户端

所有空间

lvzhisc的个人空间
市场营销部
科研
电影-公共空间
演示公共群
照片---ppt
市场销售部

Web Interface

计算机

计算机

文件(F) 编辑(E) 查看(V) 工具(T) 帮助(H)

组织 系统属性 卸载或更改程序 映射网络驱动器 打开控制面板

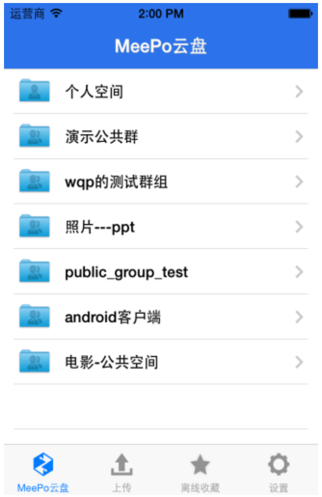
收藏夹
下载
桌面
最近访问的位置

库
Subversion
视频
图片
文档
音乐

计算机
本地磁盘 (C:)
本地磁盘 (D:)
娱乐 (E:)
MeepoDisk (M:)
MeePo

本地磁盘 (C:)
本地磁盘 (D:)
娱乐 (E:)
DVD RW 驱动器 (F:)
MeepoDisk (M:)

For Windows (Similar for Linux, MacOS)



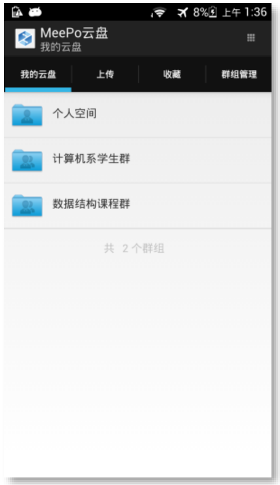
运营商 2:00 PM

MeePo云盘

个人空间
演示公共群
wqp的测试群组
照片---ppt
public_group_test
android客户端
电影-公共空间

MeePo云盘 上传 离线收藏 设置

iOS



8% 上午 1:36

MeePo云盘
我的云盘

我的云盘 上传 收藏 群组管理

个人空间
计算机系学生群
数据结构课程群

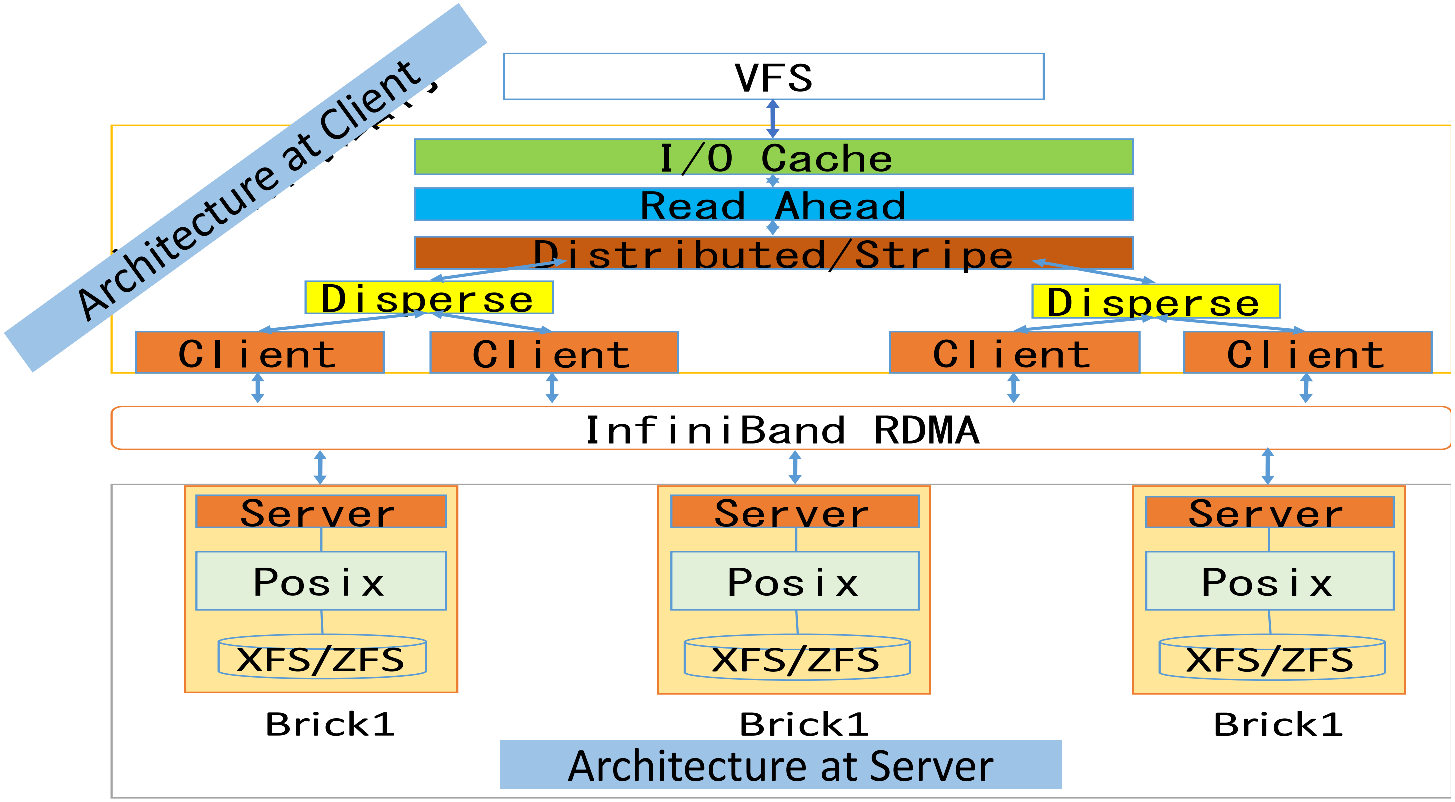
共 2 个群组

Android

MeePo

4.1 Enable Large Scale Encoding/Decoding

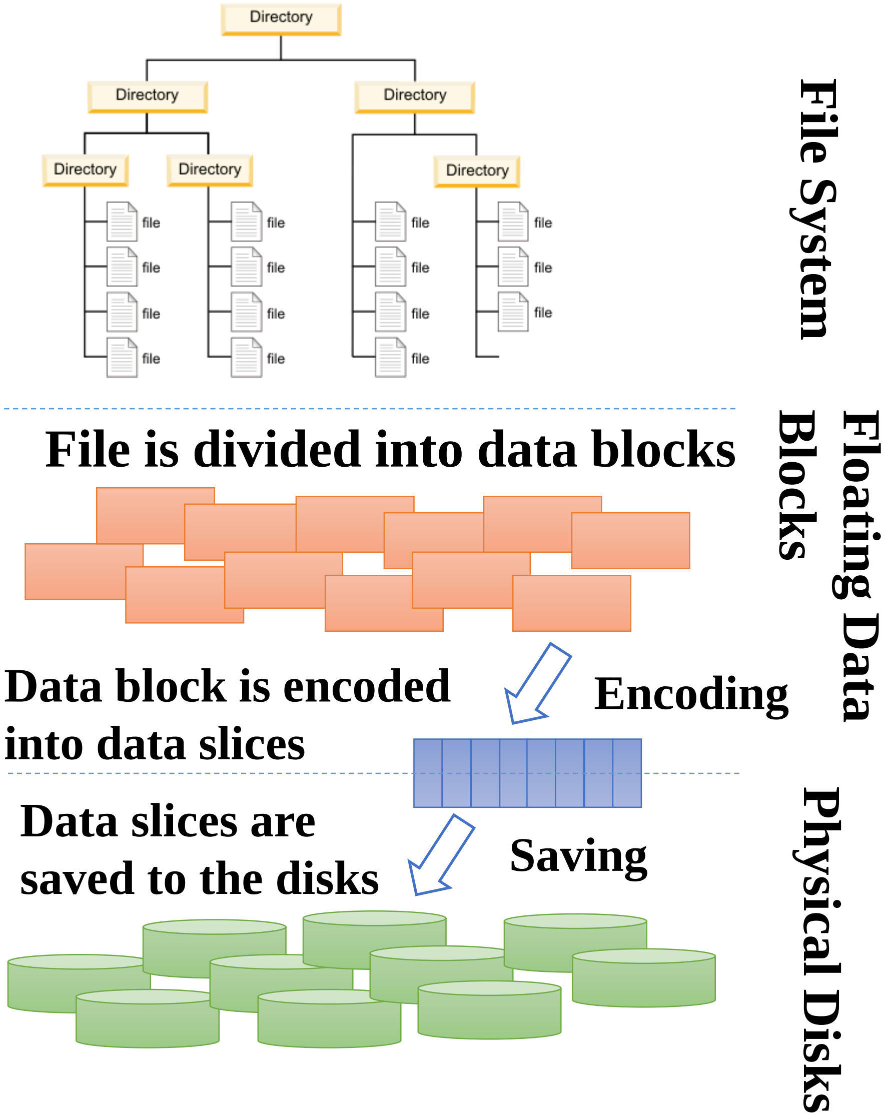
- (N+32)-level protection, providing very high availability . Computing complexity is increased exponentially.
- Improving Encoding/Decoding Performance
    - Using parallel encoding/decoding, AVX2 instructions, Binary Division
- The throughput reaches 1GB/s (High availability without performance degradation)



TStor

4.2 Tstor Architecture

- File data is saved as data blocks
- The data block is encoded into data slices through the erasure code
- Data Slice is saved to physical disk by data server
- Data slices of the same data block are distributed to different disks
- TStor forms three layers: physical disks (saves data slices), floating data blocks, and file systems



File System

File is divided into data blocks

Floating Data Blocks

Data block is encoded into data slices

Encoding

Data slices are saved to the disks

Saving

Physical Disks

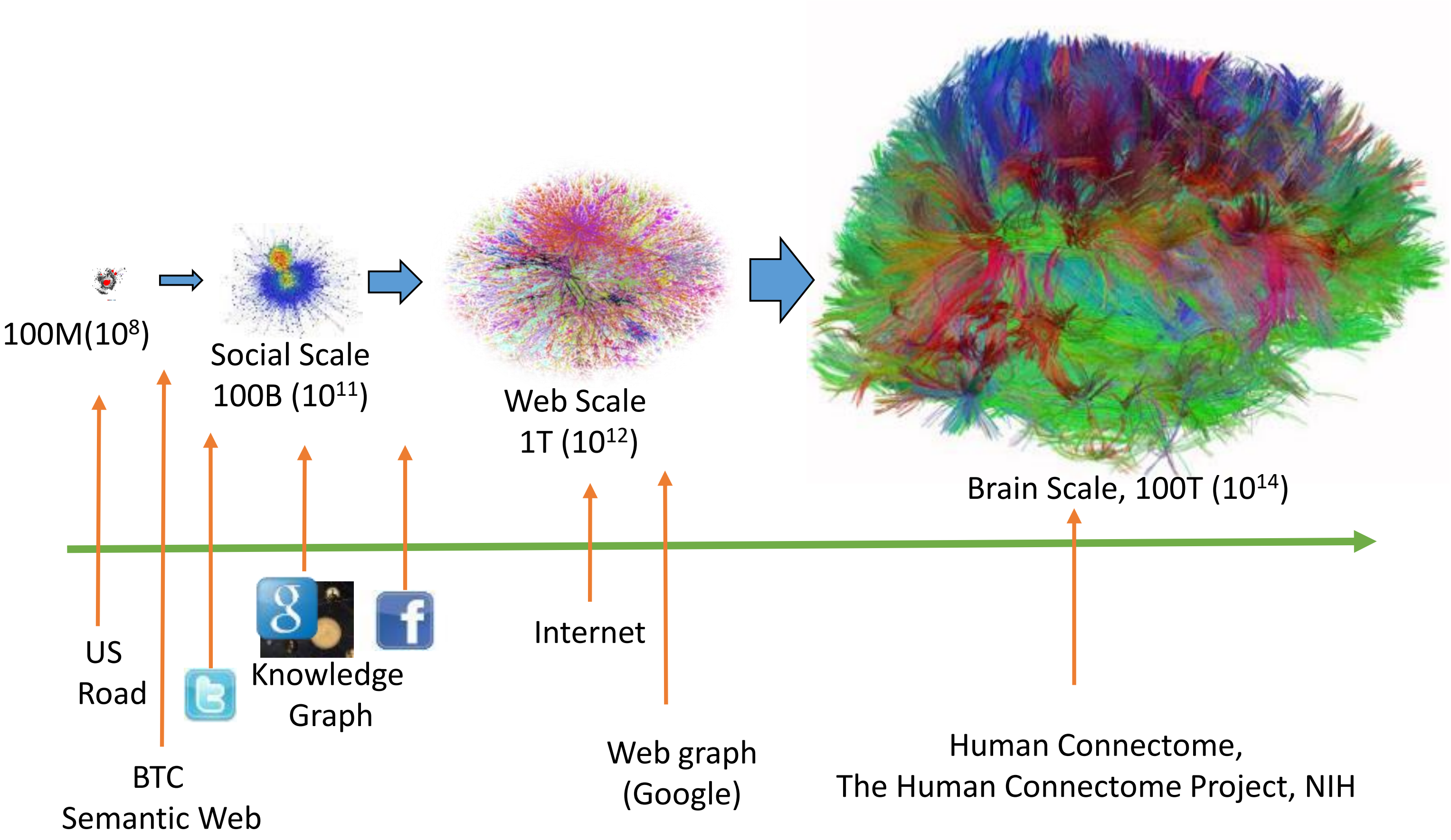
4.3 Tstor: Enable Large Scale Encoding/Decoding

Model	Object Write	Object Read	NFS Write	NFS Read
32 + 16	1.2GB/s	969MB/s	880MB/s	845MB/s
16 + 8	1.3GB/s	775MB/s	997MB/s	707MB/s
8 + 4	1.4GB/s	702MB/s	882MB/s	655MB/s

Data Recovery Capability:1.18 GB/s,8TB SATA DISK/2 hours (15 hours for 4+2 model)

Graph Computing

Why Graph Computing Matters?



100M(10^8)

Social Scale 100B (10^{11})

Web Scale 1T (10^{12})

Brain Scale, 100T (10^{14})

US Road

BTC Semantic Web

Knowledge Graph

Internet

Web graph (Google)

Human Connectome, The Human Connectome Project, NIH

Acknowledgement: Y. Wu, WSU

Data-Parallel V.S. Graph-Parallel

Data-Parallel

Graph-Parallel

Map Reduce

Feature Extraction

Cross Validation

Computing Sufficient Statistics

GraphLab

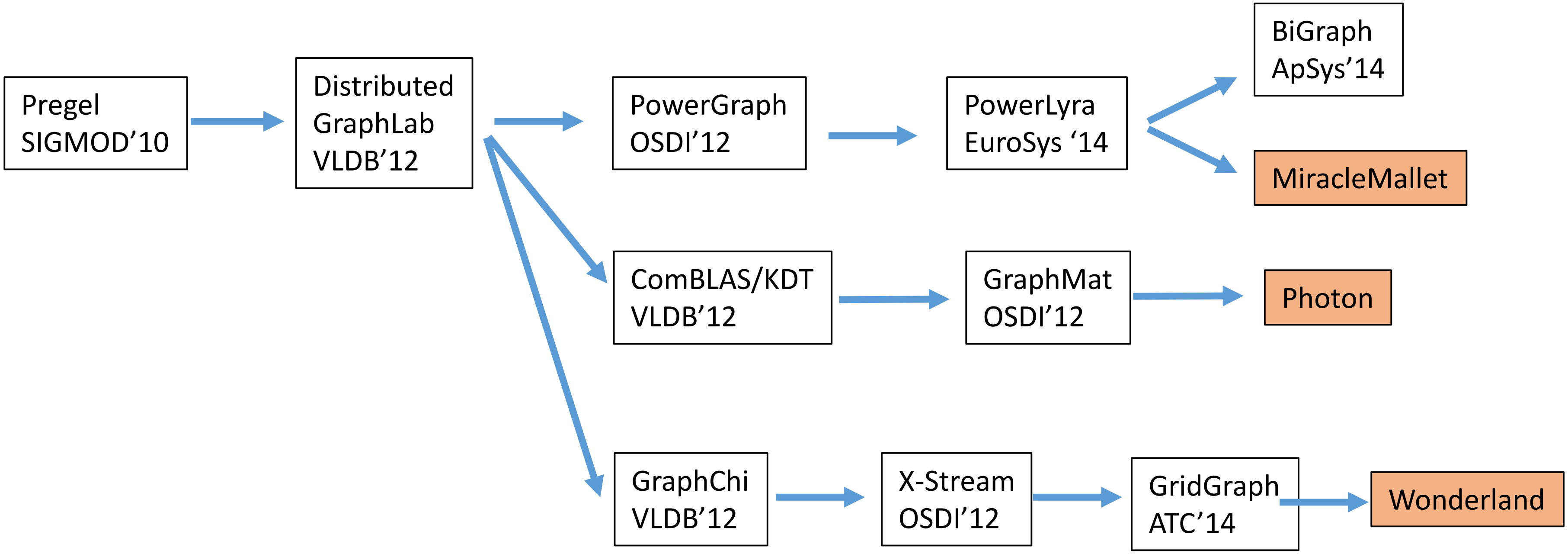
Graphical Models
Gibbs Sampling
Belief Propagation
Variational Opt.

Semi-Supervised Learning
Label Propagation
CoEM

Collaborative Filtering
Tensor Factorization

Data-Mining
PageRank
Triangle Counting

What’s going on with Graph Computing?



* Ones in orange are our work

Our Research

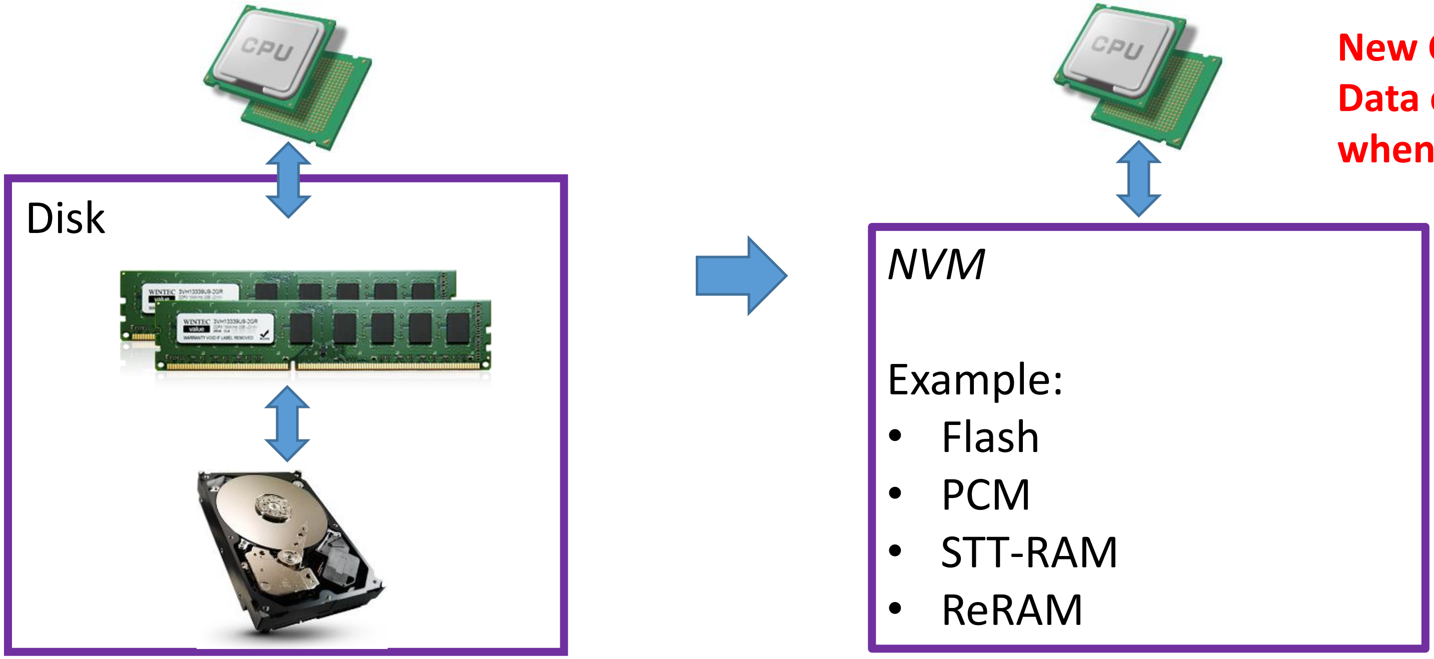
- MiracleMallet: 3D Graph Partitioning, Exploring the hidden dimension in graph processing for high performance computing
- Photon: Improving the Data Locality of Graph Computing. The current data layout incurs large amount of interleaved memory access. Making optimization for the locality of source vertex of each edge will often hurt the locality of target vertex or vice versa.
- Wonderland: Existing graph abstraction techniques typically assume either fully in-memory or distributed environment. Wonderland is based on graph abstraction

Non-Volatile Memory Systems (NVM)

Data Consistency for NVM

Traditional Memory-Disk architecture

Non volatile memory(NVM) architecture



Disk

NVM

Example:

- Flash
- PCM
- STT-RAM
- ReRAM

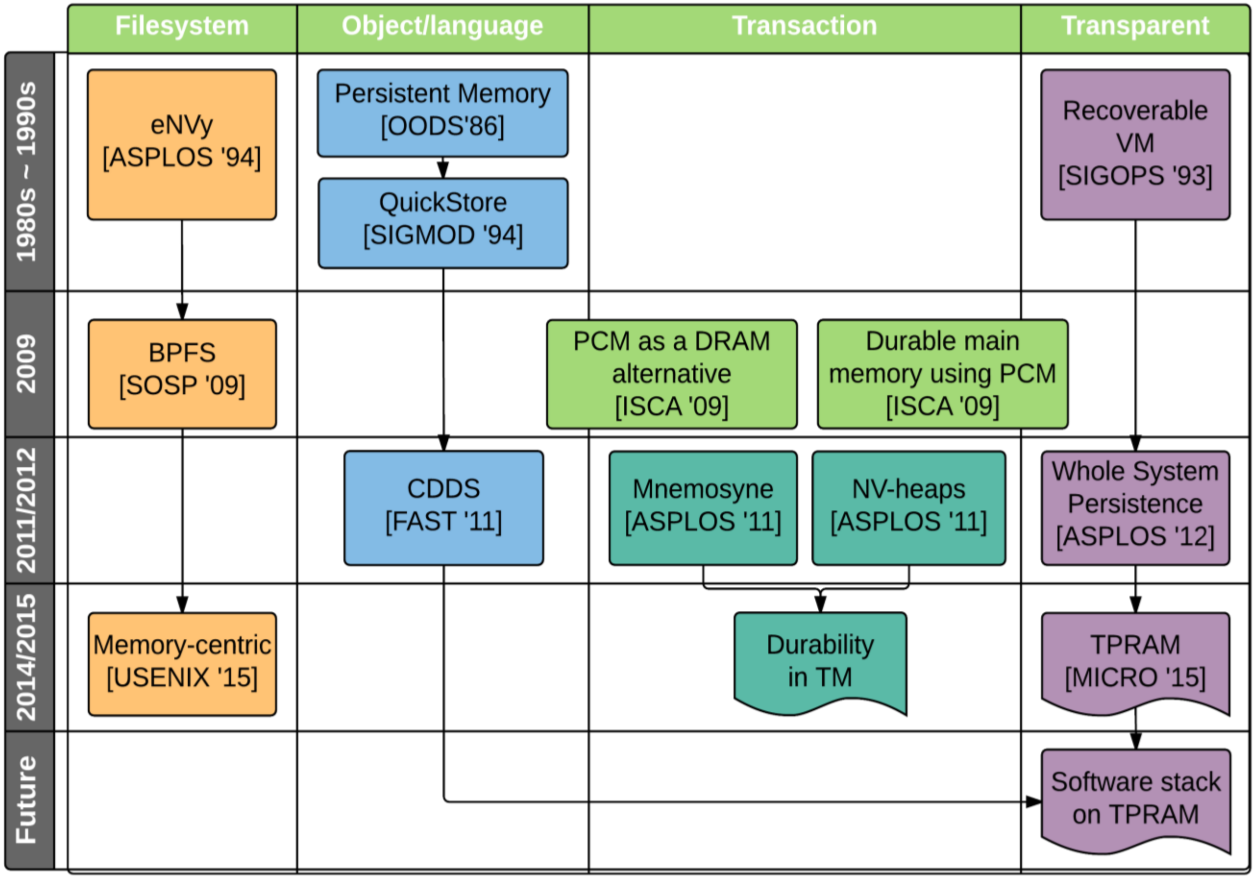
New Challenge：
Data consistency when crash

Our Research

[2]Data consistency for transaction memory. Our solution supports higher throughput and lower latency

[3] Providing software-transparent API for better programmability and proposing an efficient consistent dual-scheme check pointing mechanism for performance

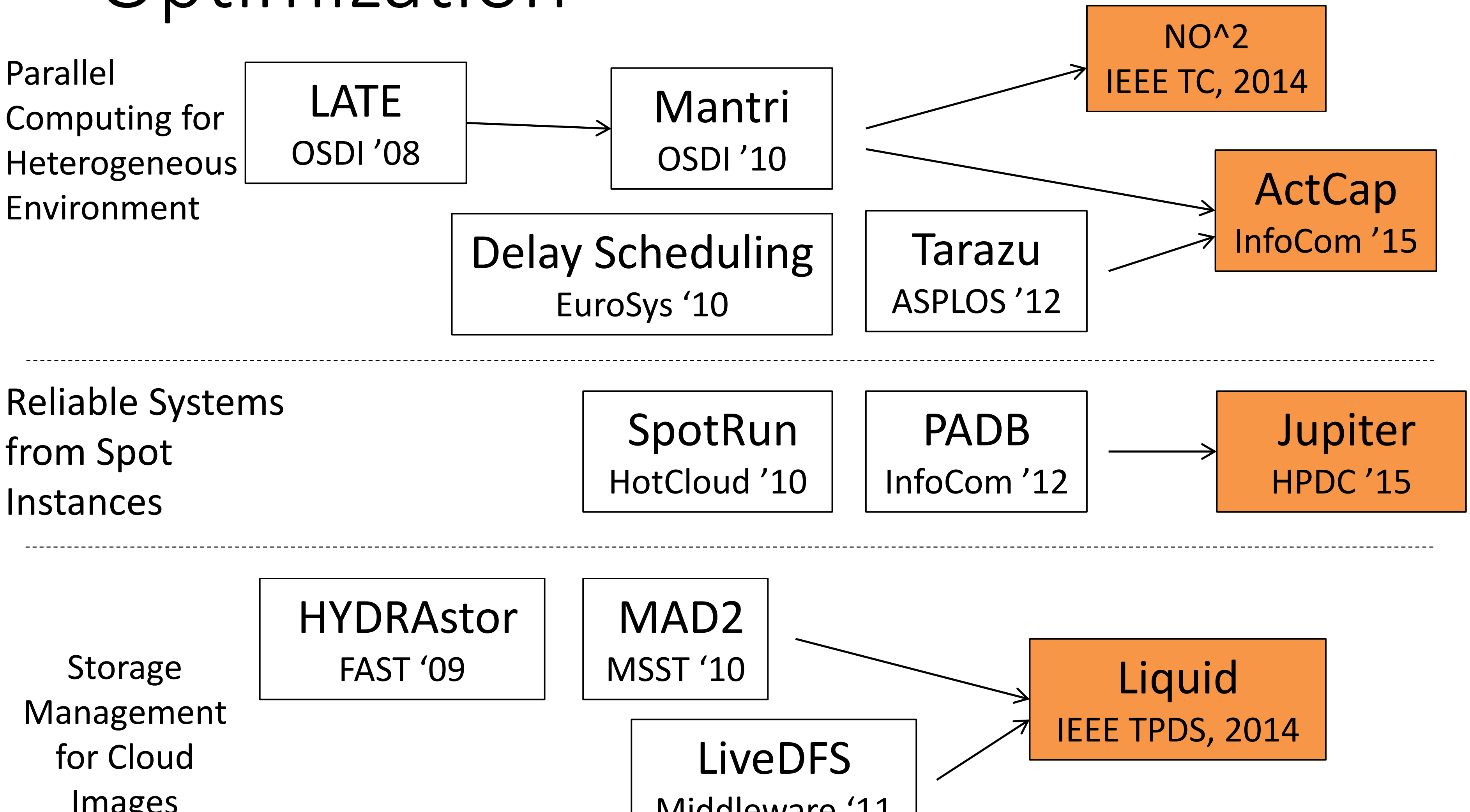
[1]Data consistency for file system, which used on mobile system, decreasing response time and energy consumption



	Filesystem	Object/language	Transaction	Transparent
1980s ~ 1990s	eNVy [ASPLOS '94]	Persistent Memory [OODS'86] QuickStore [SIGMOD '94]		Recoverable VM [SIGOPS '93]
2009	BPFS [SOSP '09]		PCM as a DRAM alternative [ISCA '09] Durable main memory using PCM [ISCA '09]	
2011/2012		CDDS [FAST '11]	Mnemosyne [ASPLOS '11] NV-heaps [ASPLOS '11]	Whole System Persistence [ASPLOS '12]
2014/2015	Memory-centric [USENIX '15]		Durability in TM	TPRAM [MICRO '15]
Future				Software stack on TPRAM

Cloud Computing

Our Reseach on Cloud Computing Optimization



Parallel Computing for Heterogeneous Environment

LATE
OSDI ’08

Mantri
OSDI ’10

NO^2
IEEE TC, 2014

ActCap
InfoCom ’15

Delay Scheduling
EuroSys ‘10

Tarazu
ASPLOS ’12

Reliable Systems from Spot Instances

SpotRun
HotCloud ’10

PADB
InfoCom ’12

Jupiter
HPDC ’15

Storage Management for Cloud Images

HYDRAstor
FAST ‘09

MAD2
MSST ‘10

LiveDFS
Middleware ‘11

Liquid
IEEE TPDS, 2014

* Ones in orange are our work

Facility

Facilities

- 2.5 PB Storage Cluster
- High Performance Distributed Computing Cluster (with Infiniband and RDMA)
- Tiny Cluster
- Many Core Machine
- Flash & SSD

Storage Cluster

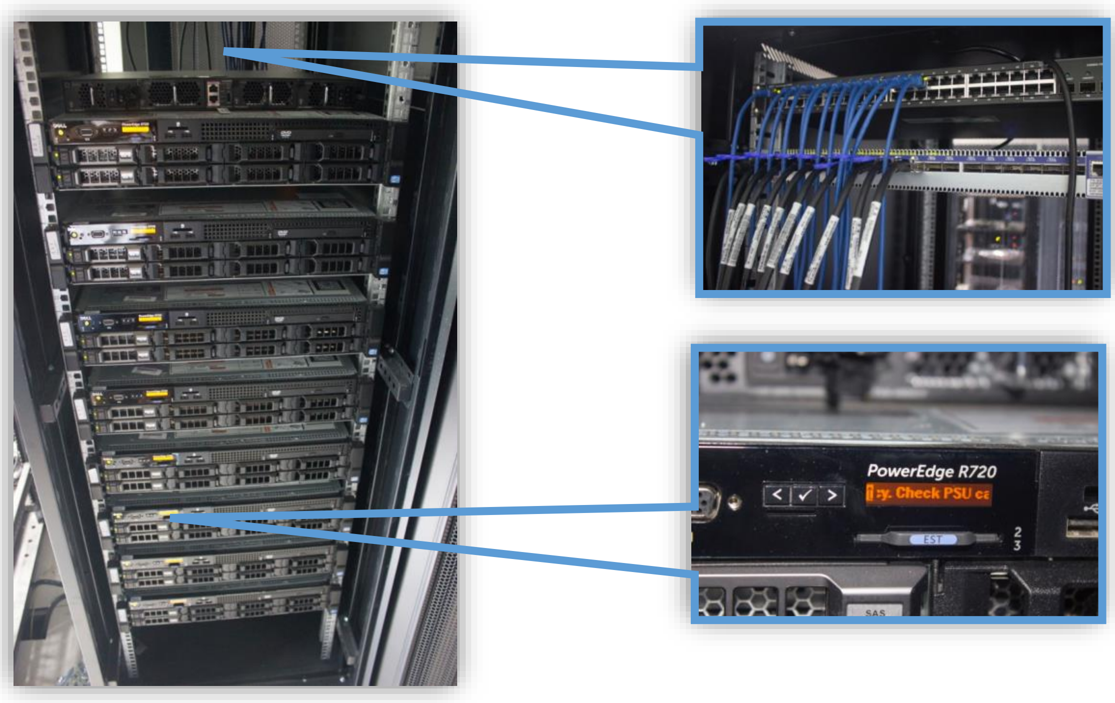
- 2.5 PB Storage Cluster

- ➢10 Servers
- ➢36TB each Server

- ➢18 Servers
- ➢36TB each Server
- ➢Connected by 40Gbps Infiniband

Computing Cluster

- High Performance Distributed Computing Cluster (8 Servers)

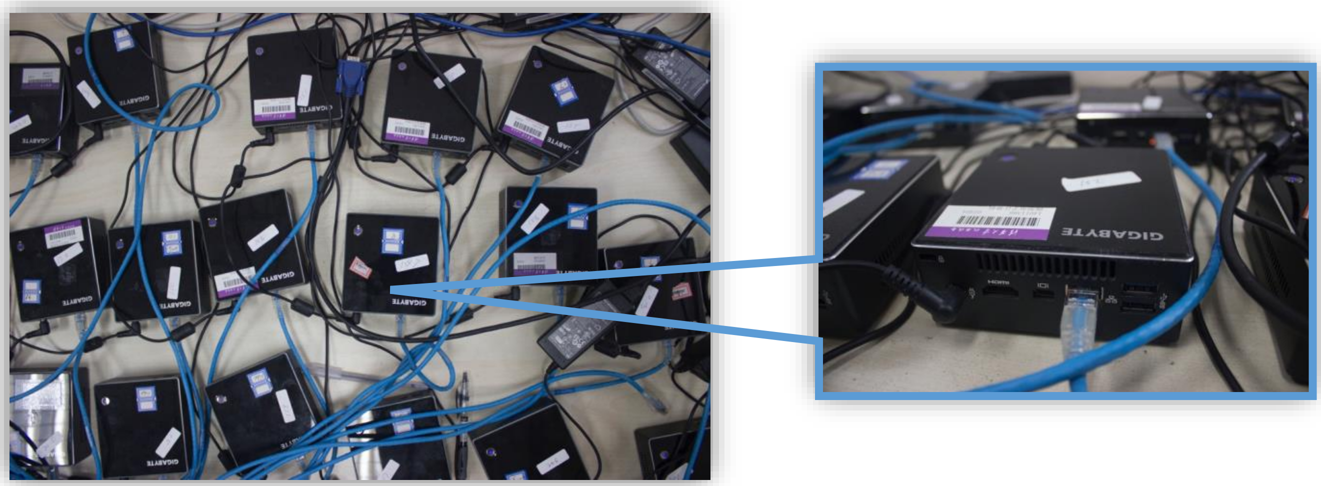


- 1Gbps Ethernet
- 20Gbps Infiniband
- *Mellanox MT27500* NIC
    - RDMA

- *Dell R720*
- 2 CPU
- *Intel Xeon CPU E5-2640 v*
    - 32 logical cores
    - 20MB LLC
- 96GB RAM

Tiny Cluster

- Tiny Cluster



➢*Intel Core i7*
➢16GB RAM
➢512GB SSD

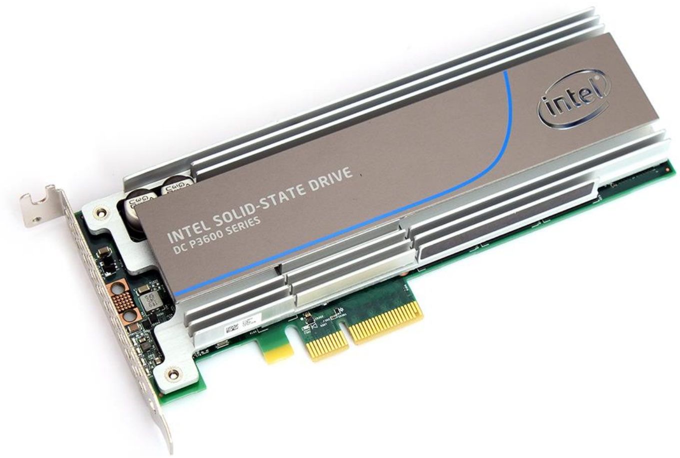
One Node with 120 cores

- Many Core Machine

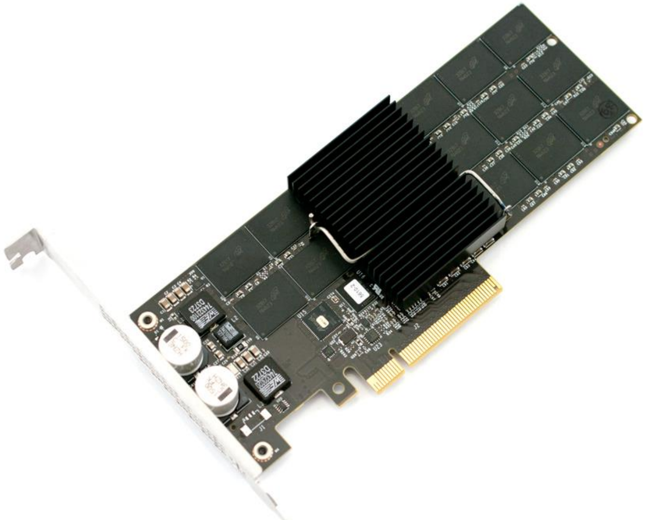
- *IBM System x3950 X6*
- NUMA architecture
- 120 physical cores
- 8CPU
- *Intel Xeon E7-8870 v2*
  - 15 physical cores
  - 30MB LLC
- 1TB RAM
- 2TB SSD

Experiment Environment

- PCIE SSD



- *Intel SSD DC P3600 1.2 TB*
    - 2.6GB/s
    - 160k IOPS
    - 20us latency



- *Fusion IO Atomics SX300*
    - 2.6GB/s
    - 285k IOPS
    - 15us latency

Thank You and for more information

http://madsys.cs.tsinghua.edu.cn/