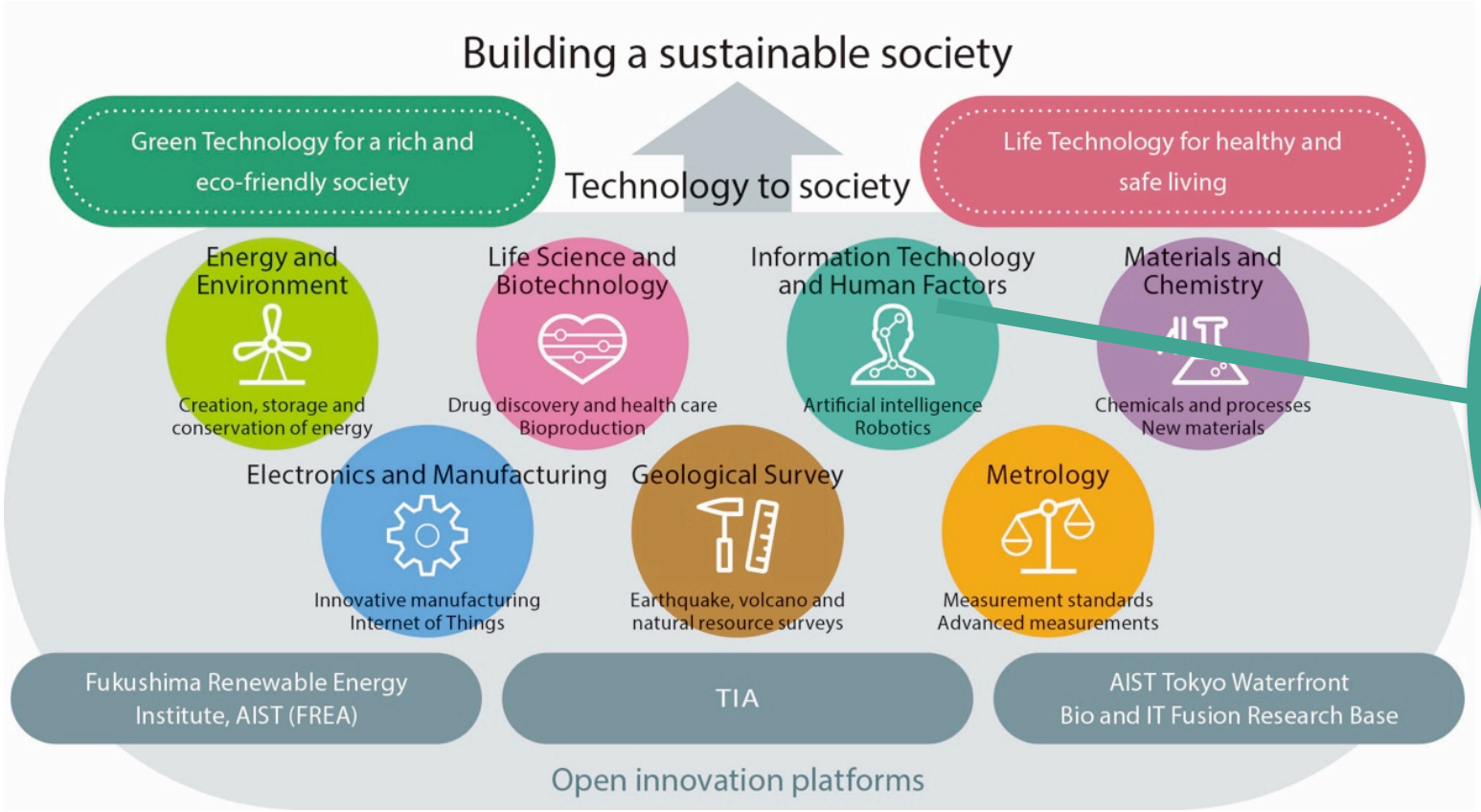
Building Software Ecosystems for AI Cloud using Singularity HPC Container

National Institute of
Advanced Industrial Science and Technology (AIST)
Artificial Intelligence Research Center

Hitoshi Sato

About AIST



Artificial Intelligence Research Center
@Odaiba, Tokyo

Real World Big-Data Computing – Open Innovation Laboratory
Joint Lab w/ Tokyo Tech

ITRI

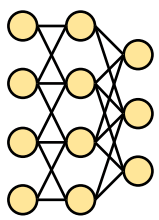
Information Technology Research Institute
HQ @Tsukuba

One of **the public research organization** in Japan under METI*, focused on **"bridging"** the gap between **innovative technological seeds** and **commercialization**.

*Ministry of Economy, Trade and Industry

Current Status of AI R&D in Japan:

AI requires the convergence of Algorithm, Big Data, and Computing Infrastructure



Algorithm

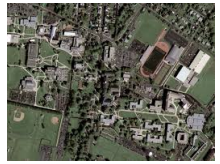
AI Venture Startups

AI/BD Centers & Labs in National Labs & Universities

Deep Learning, Machine Learning, etc.

Cutting edge infrastructures,
Clouds, Supercomputers, etc.

Computing Infrastructure



Big Companies AI/BD R&D (also Science)

Big Data

We lack the cutting edge infrastructure dedicated to AI and Big Data, open to public use

AIST ABCI: AI Bridging Cloud Infrastructure

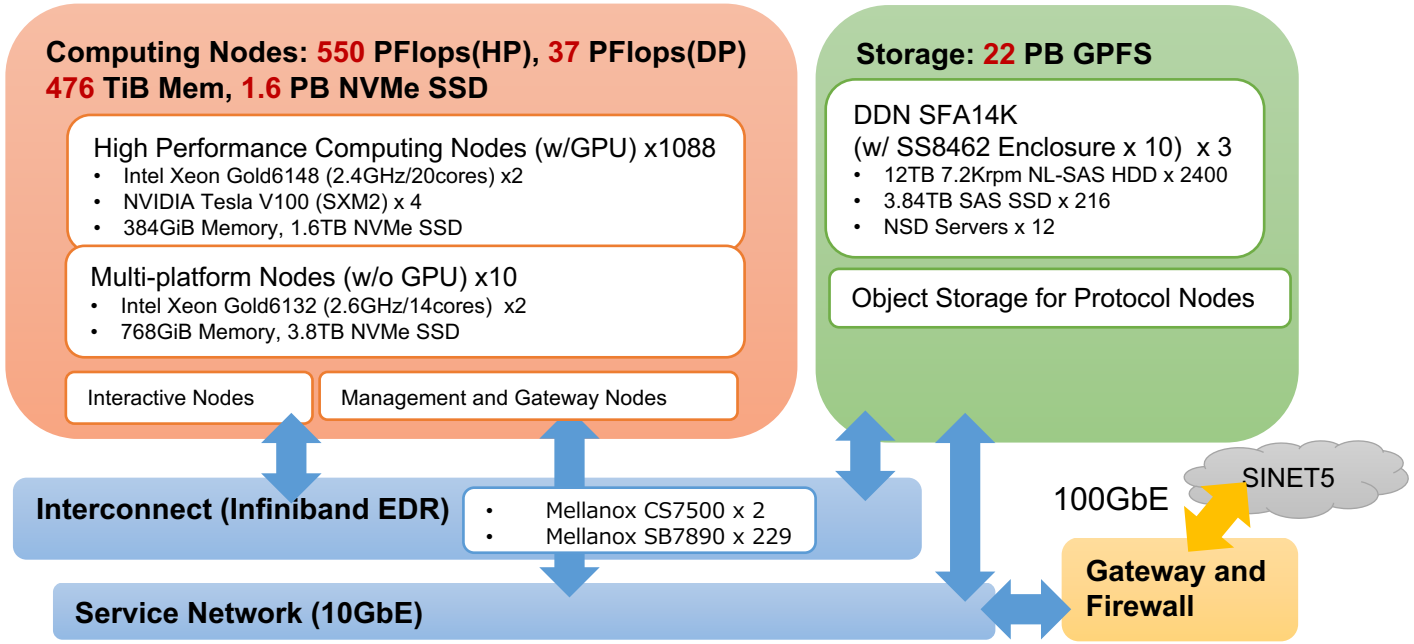
as the worlds first large-scale Open AI Infrastructure

- **Open, Public, and Dedicated** infrastructure
  for AI & Big Data Algorithms, Software and Applications
- Platform to accelerate joint academic-industry R&D for AI in Japan
- Top-level compute capability w/ **0.55EFlops(HP), 37PFlops(DP)**

Univ. Tokyo Kashiwa II Campus
Operation Scheduled in 2018

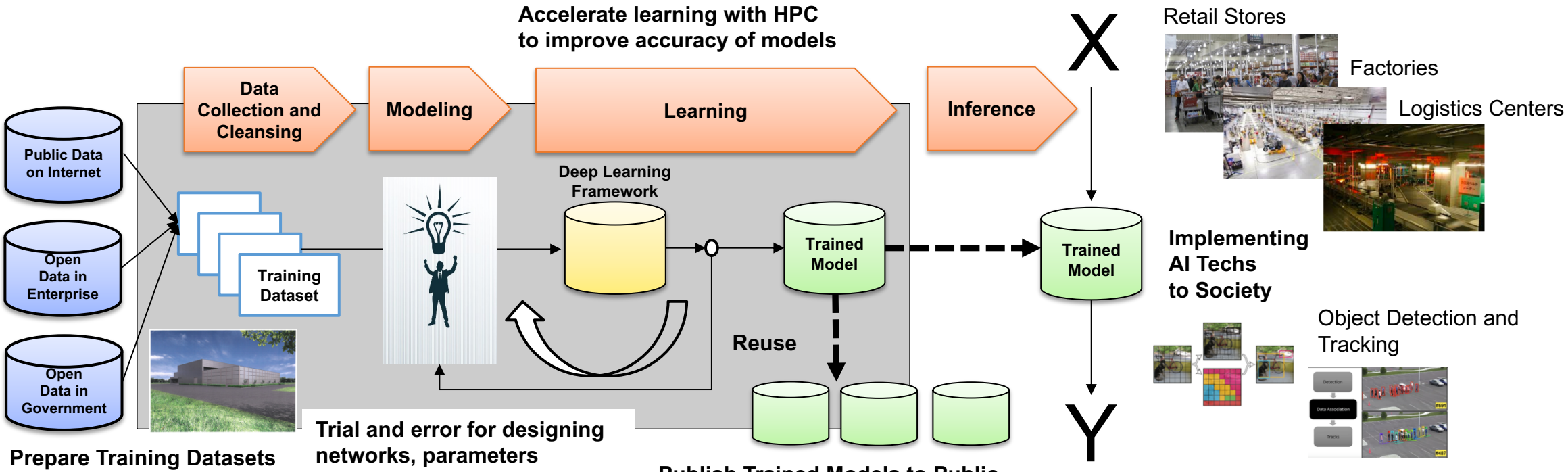
ABCI Overview

- **1088x** compute nodes w/ **4352x NVIDIA Tesla V100 GPUs**, **476TiB of Memory**, **1.6PB of NVMe SSDs**, **22PB of HDD-based Storage** and **Infiniband EDR network**
- **Ultra-dense IDC design from the ground-up** w/ **20x thermal density** of standard IDC
- **Extreme Green** w/ **ambient warm liquid cooling, high-efficiency power supplies**, etc., commoditizing supercomputer cooling technologies to clouds **(～2.2MW, ～60kW/rack)**



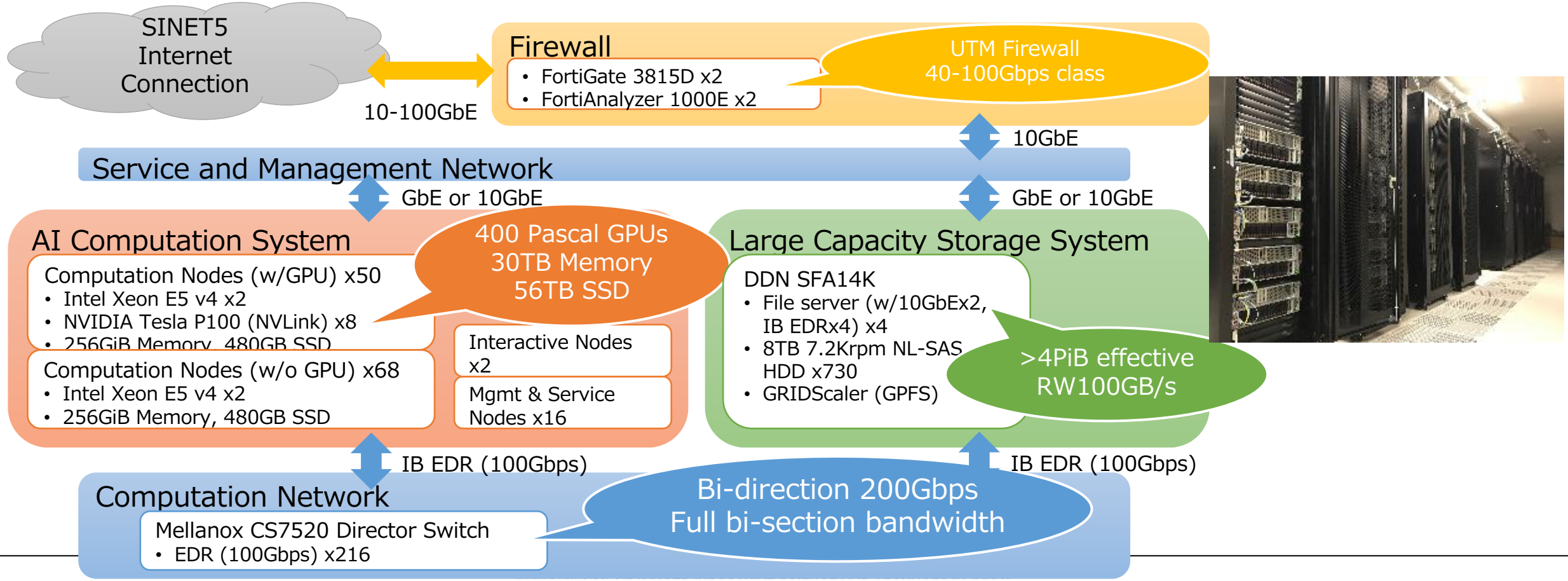
ABCI Mission

- Accommodate open ecosystems of data and software for AI R&D
  - **Sharable/Reusable/Reproducible algorithm methods, training data w/ labels, trained models**, etc.
- Creating various social applications that use AI (ML, DL, etc.)



ABCI Prototype: AIST AI Cloud (AAIC) ～June 2017

- **400x NVIDIA Tesla P100s and Infiniband EDR** accelerate various AI workloads including ML (Machine Learning) and DL (Deep Learning).
- Advanced data analytics leveraged by **4PiB shared Big Data Storage and Apache Spark** w/ its ecosystem.

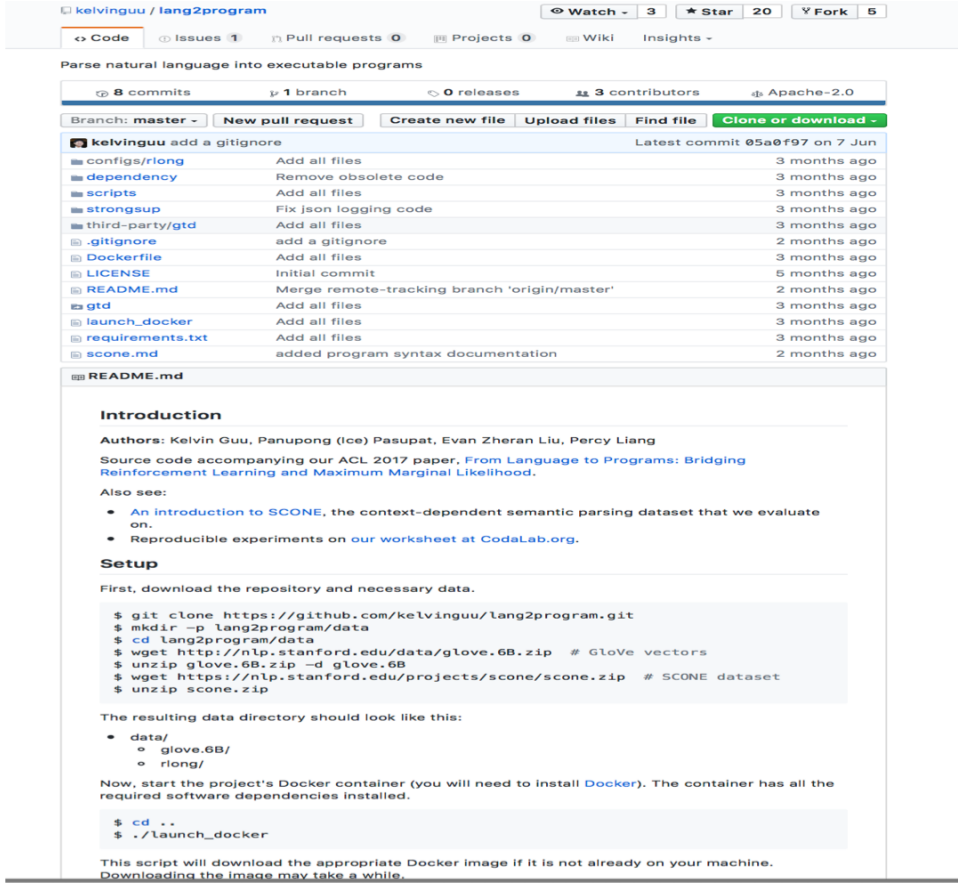


Example of Real AI/DL Apps Usage

- lang2program
  - https://github.com/kelvinguu/lang2program
  - Implementation for ACL 2017 (http://acl2017.org/) paper
  - Provided as Dockerfile
    - Including Tensorflow, CUDA, CuDNN, PostgresQL, Python pip Packages, etc.
    - Various software dependencies for installation

Can traditional shared HPC systems support such DL Apps?

- Not allowed Docker for security reason (rootless access required)
- Elaborate efforts for manual installation



Software Ecosystem for HPC in AI

Different SW Ecosystem between HPC and AI/BD/Cloud

Existing Clouds

BD/AI User Applications

Machine Learnig MLlib/ Mahout/Chainer

Graph Processing GraphX/ Giraph /ScaleGraph

SQL/Non-SQL Hive/Pig

Java · Scala · Python + IDL

MapReduce Framework Spark/Hadoop

RDB PostgresQL

CloudDB/NoSQL Hbase/Cassandra/MondoDB

Distributed Filesysem HDFS & Object Store

Coordination Service ZooKeeper

VM(KVM), Container(Docker), Cloud Services (OpenStack)

Linux OS

Ethernet TOR Swtiches High Latency/Low Capacity NW

Local Node Storage

x86 CPU

Application Layer

- Cloud Jobs often Interactive w/resource control REST APIs
- HPC Jobs are Batch-Oriented, resource control by MPI

System Software Layer

- Cloud employs High Productivity Languages but performance neglected, focus on data analytics and dynamic frequent changes
- HPC employs High Performance Languages but requires Ninja Programmers, low productivity. Kernels & compilers well tuned & result shared by many programs, less rewrite
- Cloud focused on databases and data manipulation workflow
- HPC focused on compute kernels, even for data processing. Jobs scales to thousands of jobs, thus debugging and performance tuning
- Cloud requires purpose-specific computing/data environment as well as their mutual isolation & security
- HPC requires environment for fast & lean use of resources, but on modern machines require considerable system software support

OS Layer

Hardware Layer

- Cloud HW based on Web Server "commodity" x86 servers, distributed storage on nodes assuming REST API access
- HPC HW aggressively adopts new technologies such a s GPUs, focused on ultimate performance at higher cost, shared storage to support legacy apps

Existing Supercomputers

HPC User Code

Numerical Libraries LAPACK, FFTW

Various DSLs

Workflow Systems

Fortran · C · C++ + IDL

MPI · OpenMP/ACC · CUDA/OpenCL

Parallel Debuggers and Profilers

Parallel Filesystem Lustre, GPFS,

Batch Job Schedulers PBS Pro, Slurm, UGE

Linux OS

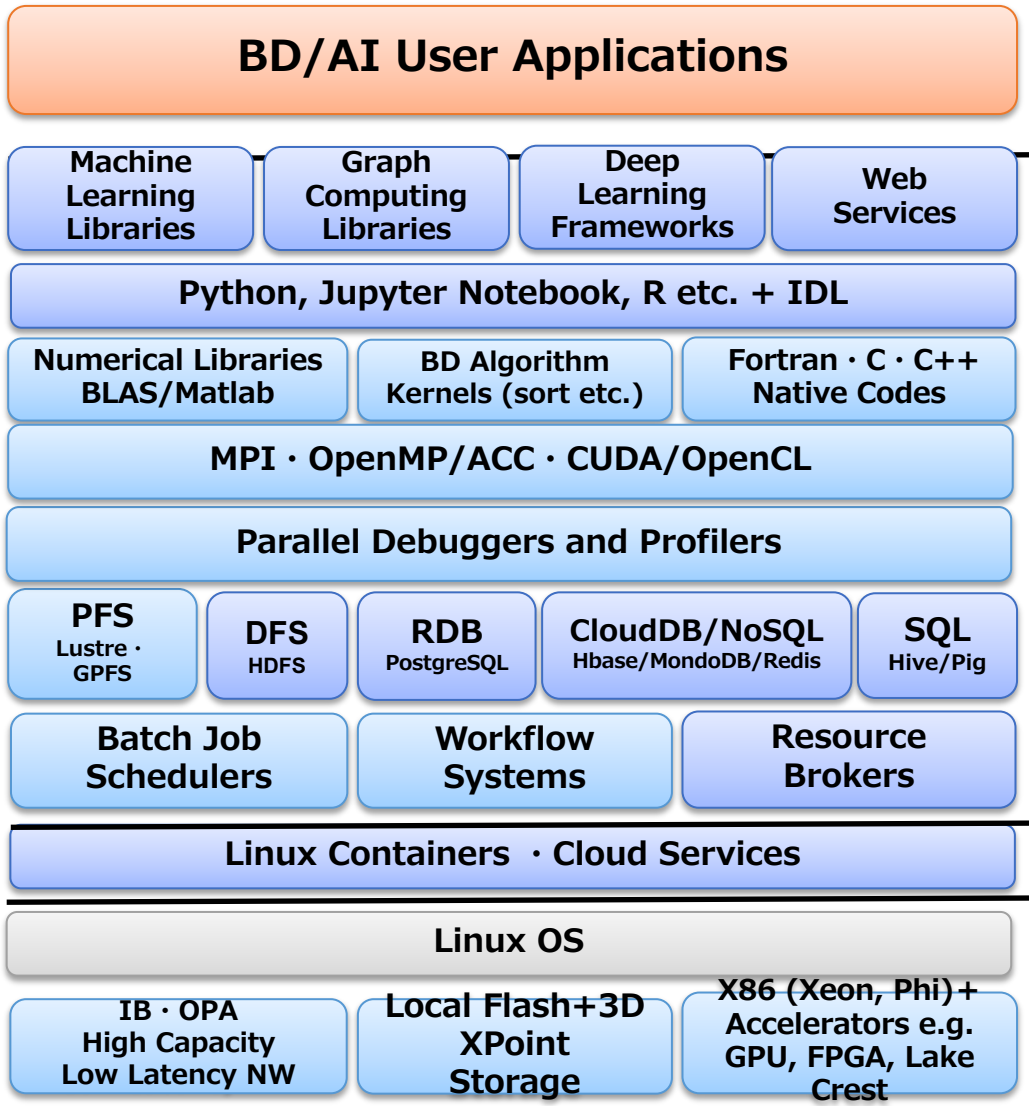
InfiniBand/OPA High Capacity Low Latency NW

High Performance SAN+Burst Buffers

X86 + Accelerators e.g. GPUs, FPGAs

Various convergence research efforts underway but no realistic converged SW Stack yet => achieving HPC – AI/BD/Cloud convergence key ABCI goal

Basic Requirements for AI Cloud System



Application

- ✓ Easy use of various ML/DL/Graph frameworks from Python, Jupyter Notebook, R, etc.
- ✓ Web-based applications and services provision

System Software

- ✓ HPC-oriented techniques for numerical libraries, BD Algorithm kernels, etc.
- ✓ Supporting long running jobs / workflow for DL
- ✓ Accelerated I/O and secure data access to large data sets
- ✓ User-customized environment based on Linux containers for easy deployment and reproducibility

OS

Hardware

- ✓ Modern HPC computing facilities based on commodity components

Singularity

- **Singularity**
  - HPC Linux Container developed at LBNL and Sylabs Inc.
    - http://singularity.lbl.gov/
    - https://github.com/singularityware/singularity
  - Rootless program execution, storage access
    - No daemons w/ root privilege like Docker
      - Computing resources are managed by job schedulers such as UGE
    - Root privilege is given by commands w/ setuid
      - Mount, Linux namespace creation, Bind paths between host and container
  - Existing Docker images available via DockerHub
  - HPC software stacks available
    - CUDA, MPI, Infiniband, etc.

Singularity Usage

User Machine

Using given container images

Using Docker container images via DockerHub

```
sudo singularity build container.img docker://ubuntu
```

Using Singularity container images via SingularityHub

```
sudo singularity build container.img shub://ubuntu
```

Using user defined recipe

Writing a Recipe file for container image creation

```
sudo singularity build container.img Singularity
```

Using container images from interactive operations

Sandbox directory

```
sudo singularity build –sandbox tmpdir/ Singularity
```

Container image file

```
sudo singularity build –writable container.img Singularity
```

Package software installation (yum, apt etc.)

```
sudo singularity shell –writable container.img
```

Copy a container image file (scp etc.)

container.img

Shared HPC Systems

Shell Execution (shell)

```
singularity shell container.img
```

Command execution (exec)

```
singularity exec container.img …
```

User defined execution (run)

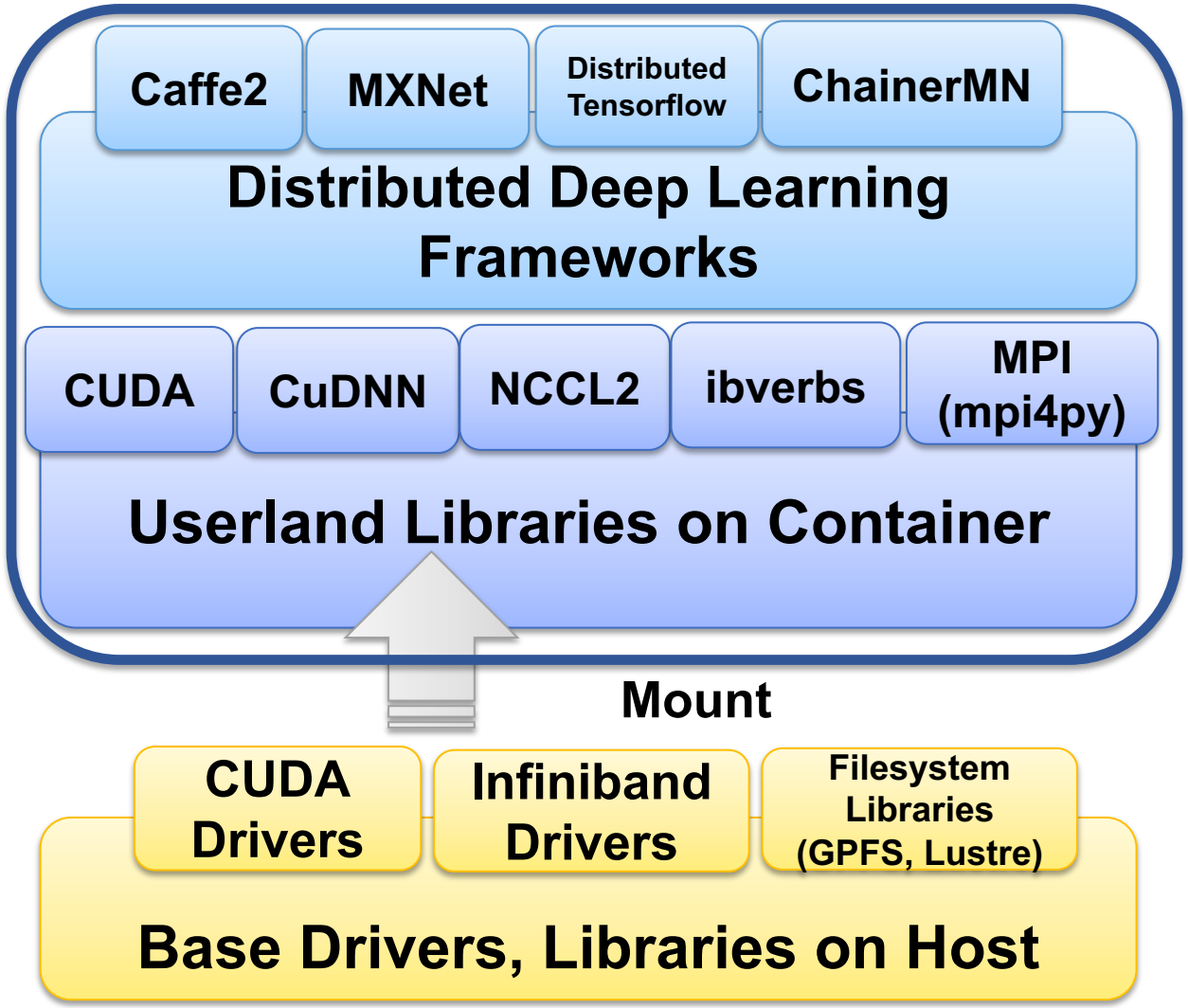
```
singularity run container.img
```

What can we do with Singularity?

- **Can we use GPU, Infiniband, etc. ? => Yes**
  - Mount host installed device drivers and libraries to container.
  - cf. nvidia-docker
- **Can we use MPI? => Yes**
  - Mount host installed MPI to container.
    (but a few merit w/ container..)
  - Install MPI programs inside the container, and launch the binary from the outside.
    (Same MPI configuration may be required between host and container)
    - mpirun –np N singularity exec container.img mpi_program
- **Can we use mpi4py? => Yes**
  - Install MPI inside the container
  - Install mpi4py inside the container
  - Launch python script from the outside

How to Create Container Images for Distributed DL

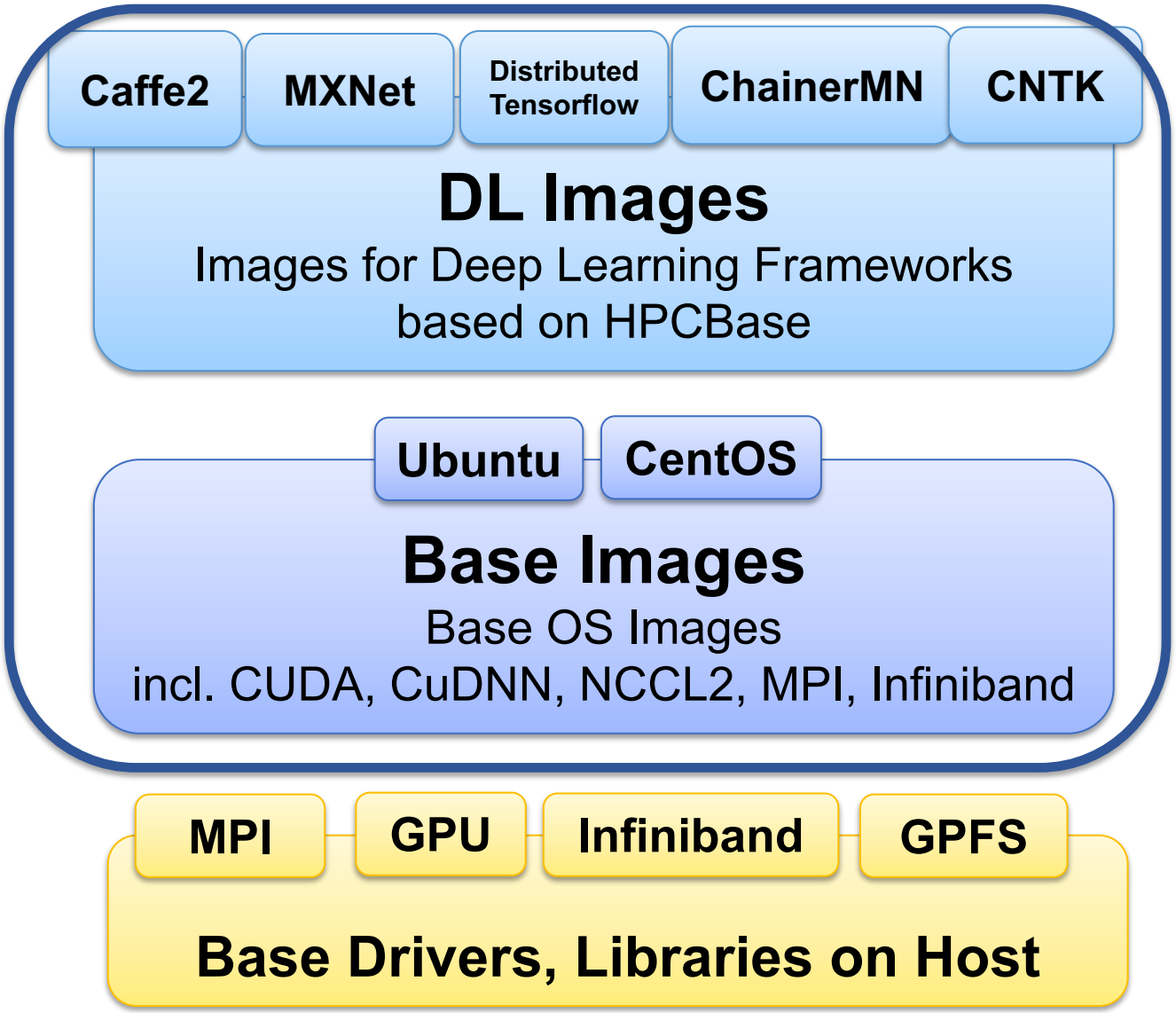
- Host
  - Mount native device drivers and system libraries to containers
    - GPU, Infiniband, etc.
    - cf. nvidia-docker
- Containers
  - Install user space libraries
    - CUDA, CuDNN, NCCL2, ibverbs, MPI, etc.
  - Install distributed deep learning frameworks
    - Optimized build for underlying computing resources



AICloudModules: Docker-based Container Image Collection for AI

- Base Images
  - Optimized OS Images for AI Cloud
    - Including CUDA,CuDNN, NCCL, MPI, etc., Libraries
    - Excluding CUDA, Infiniband, GPFS etc., native drivers on host
    DL Images
- DL Images
  - Installed Deep Learning Frameworks and AI Applications based on Base OS Images

https://hub.docker.com/r/aistairc/aaic/
as a prototype

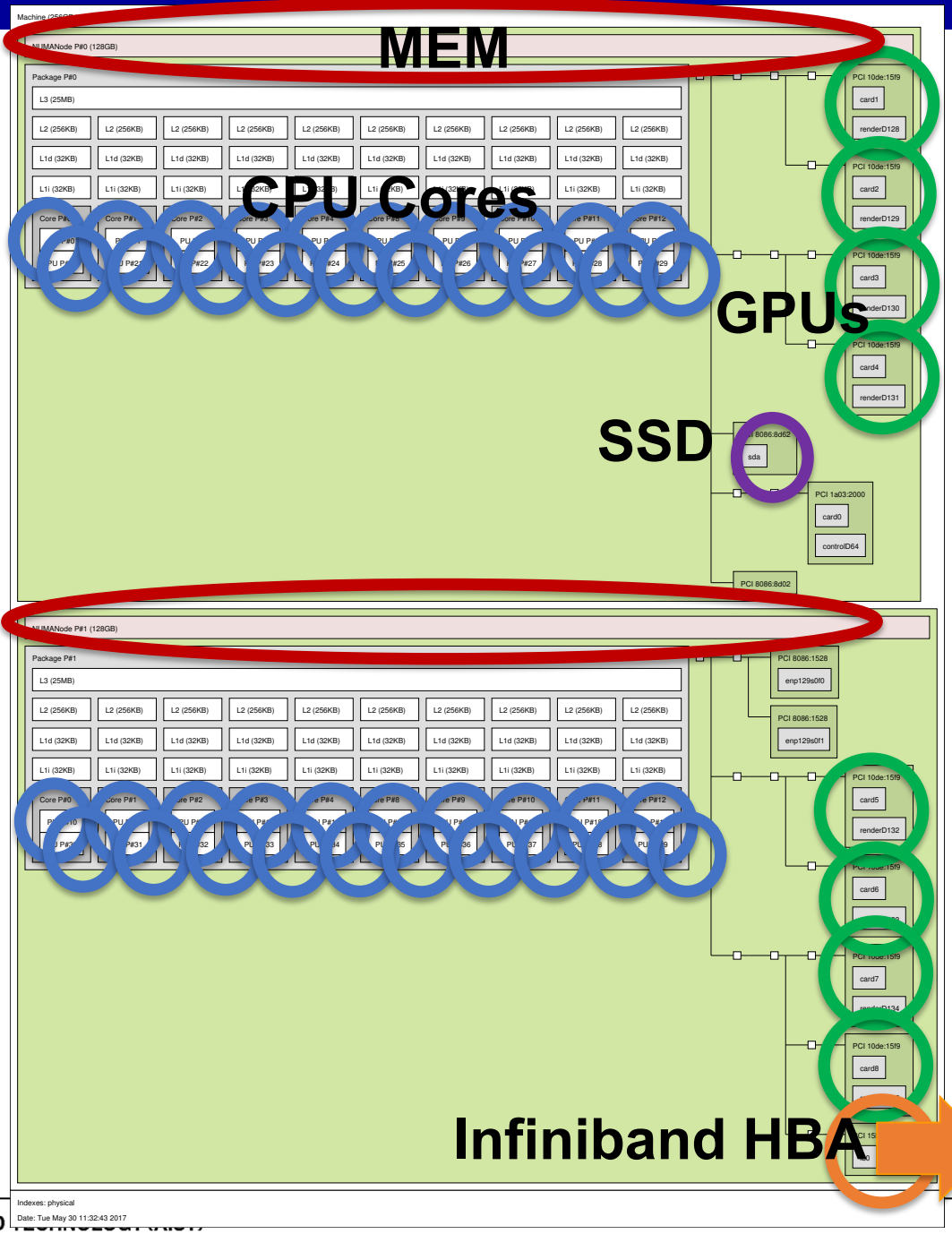


Preliminary Performance Studies

- Emulated AI Workloads
  - Computation (GEMM)
  - Memory Bandwidth (STREAM)
  - Network (OSU Micro Benchmarks)
  - Storage I/O (FIO)
  - Distributed Deep Learning (Imagenet1K)
- Experimental Environment
  - AAIC
  - Baremetal
    - CentOS 7.3, Linux kernel v3.10.0, gcc v4.8.5, glibc v2.17
  - Singularity Container
    - Singularity v2.4
    - Ubuntu 16.04, Linux kernel v3.10.0, gcc v5.4.0, glibc v2.23
  - Base Software on Both
    - CUDA 8.0.61.2 CuDNN v6.0.21 NCCL v2.0.5, OpenMPI v2.1.1

Computing Node Configuration of AAIC (up to 50x nodes)

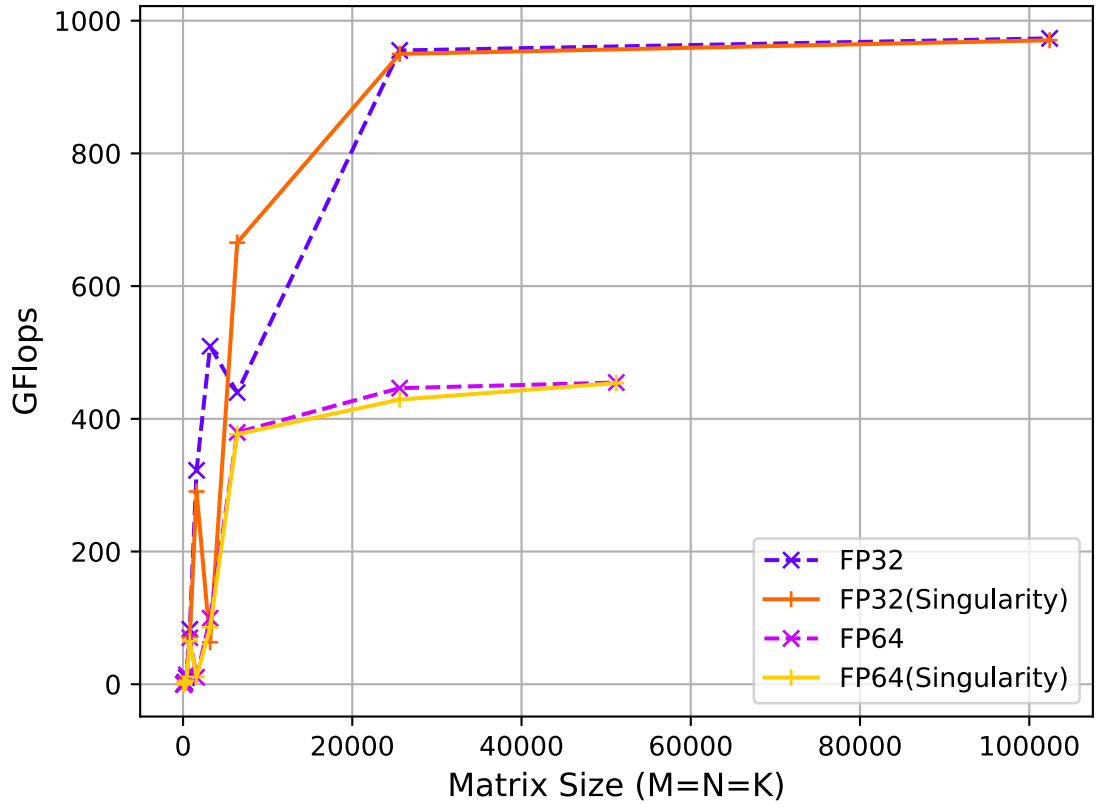
- **NVIDIA TESLA P100 x 8**
- **Intel Xeon E5-2630 v4 x 2 sockets**
  - 10 cores per socket
  - Hyper Threading (HT) enabled
  - **40 cores per node**
- **256GiB of Memory**
- **480GB of SSD**
- **EDR Infiniband**
  - Connected to other computing nodes and GPFS storage



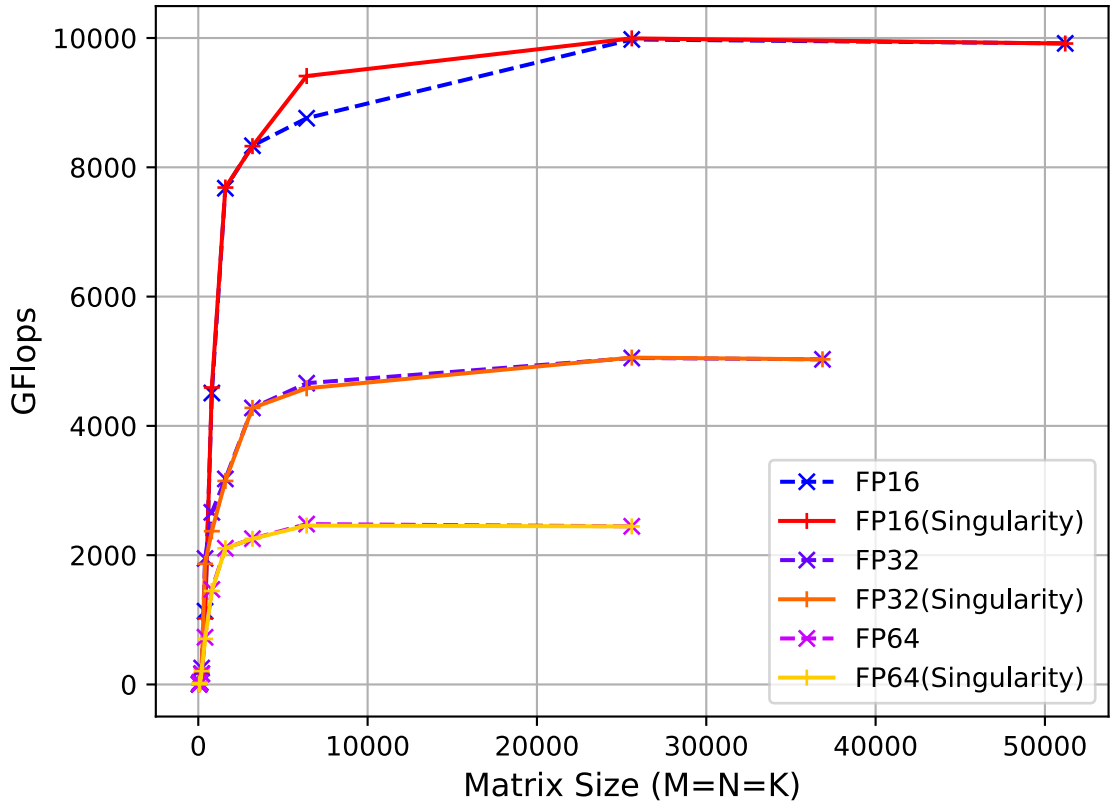
Computation (GEMM)

Better

CPU

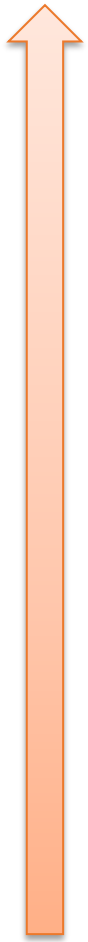


GPU

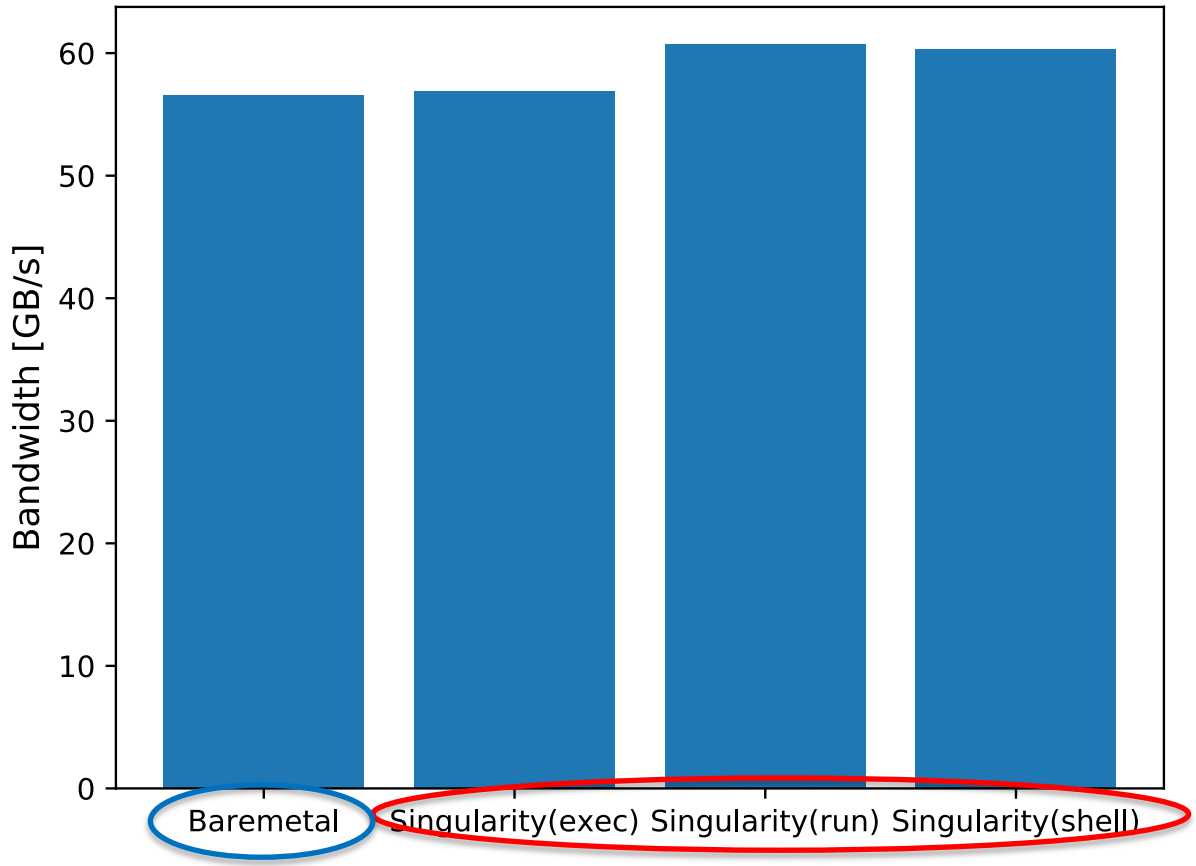


Memory Bandwidth (STREAM)

Better



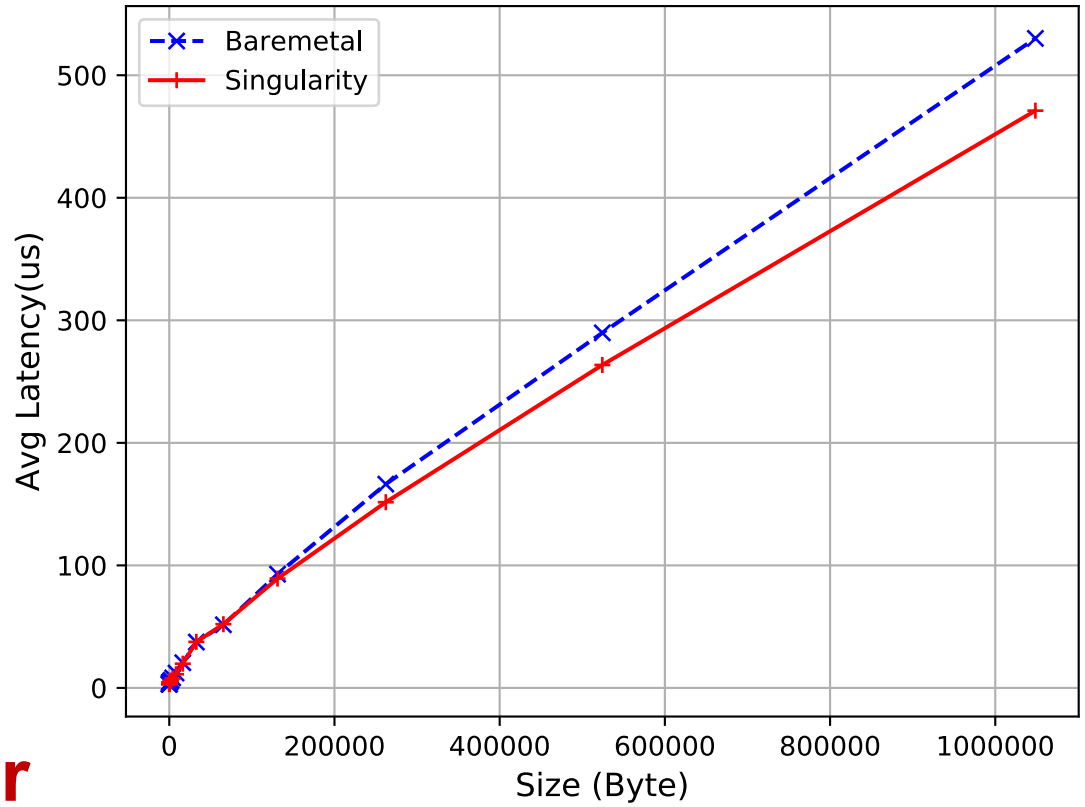
Triad



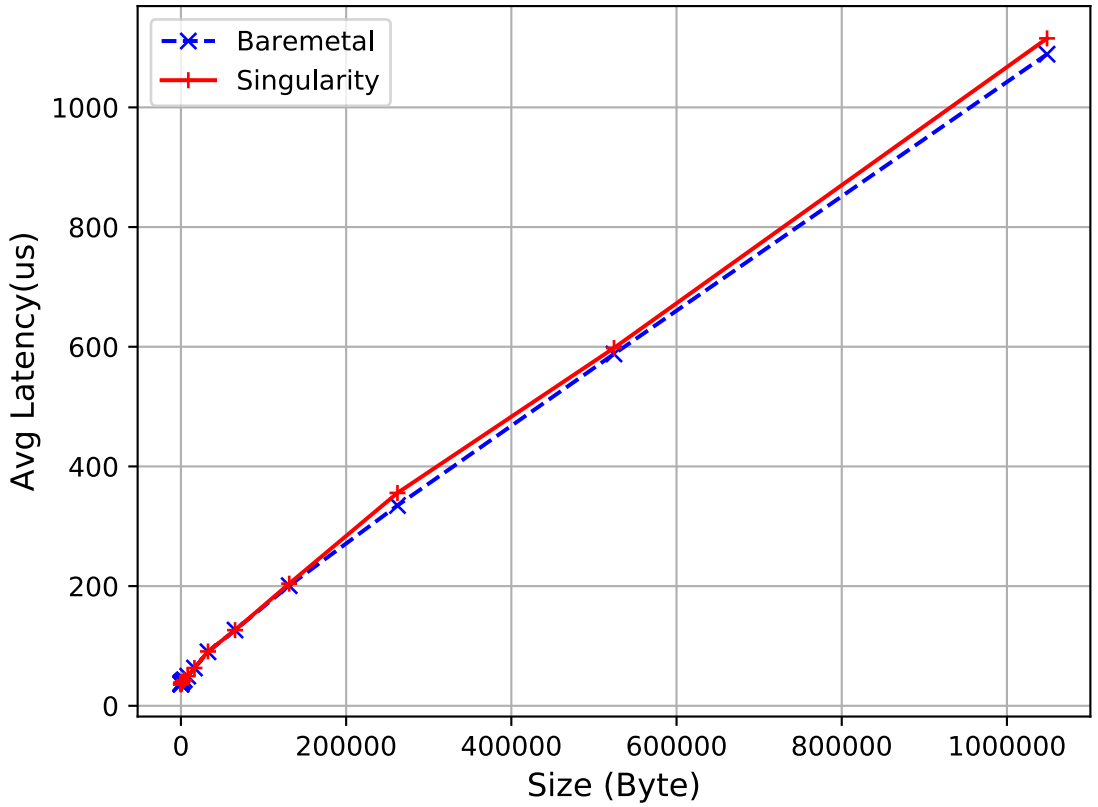
Network (OSU Micro Benchmarks v5.3.2)
AllReduce (Ave. Latency, 2nodes, 8gpus/node)

Better

AllReduce(h2h)

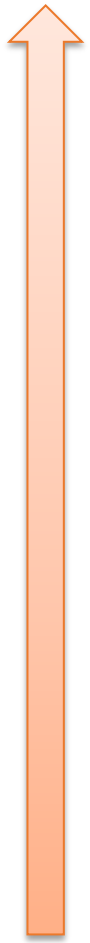


AllReduce(d2d)

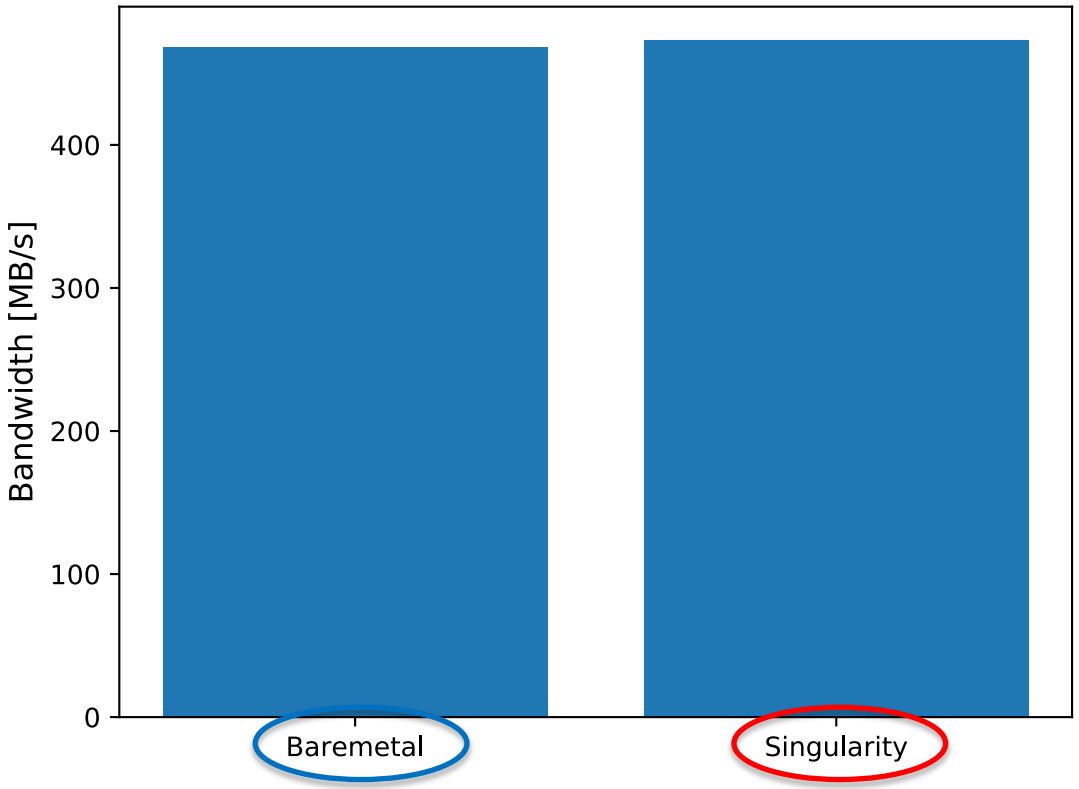


Storage I/O (FIO v2.19) Throughput

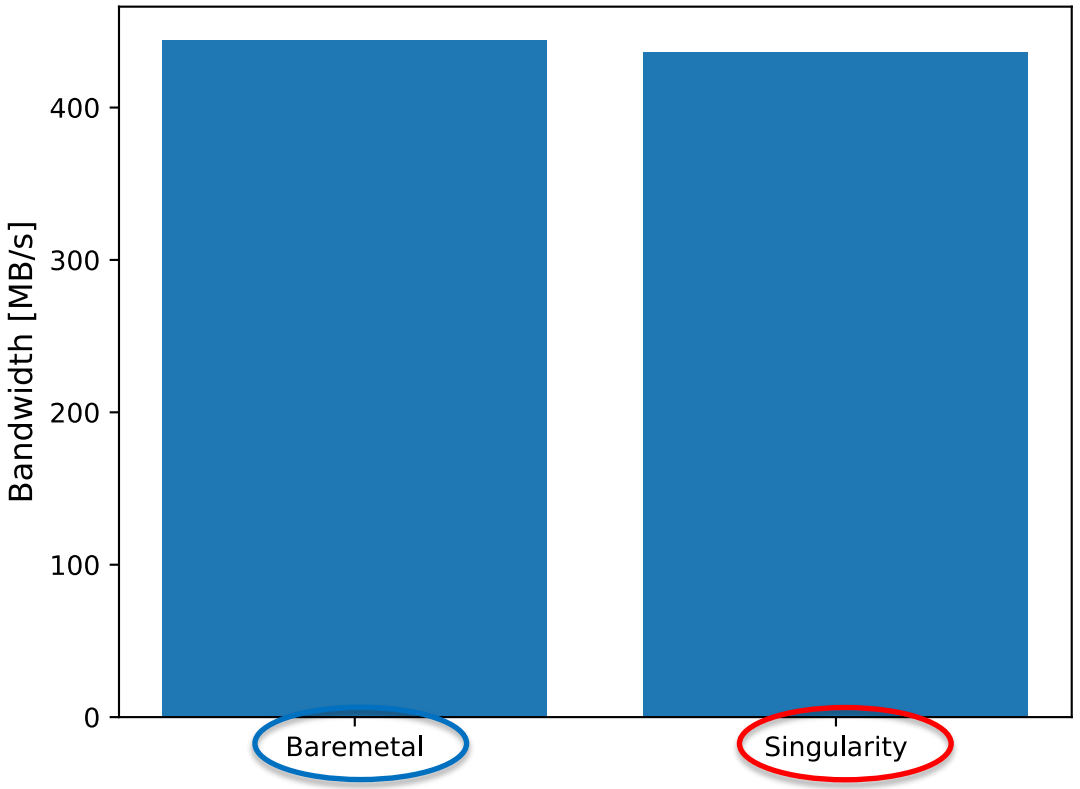
Better



READ

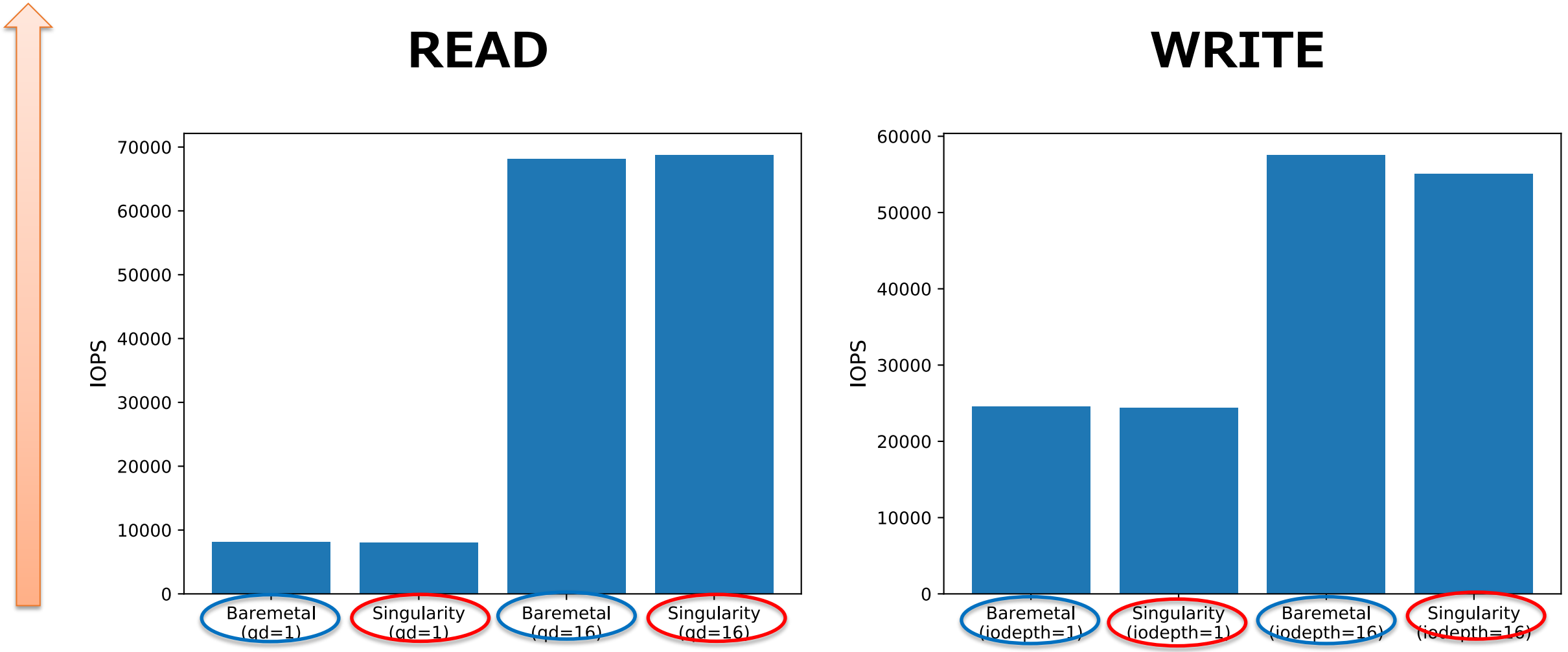


WRITE



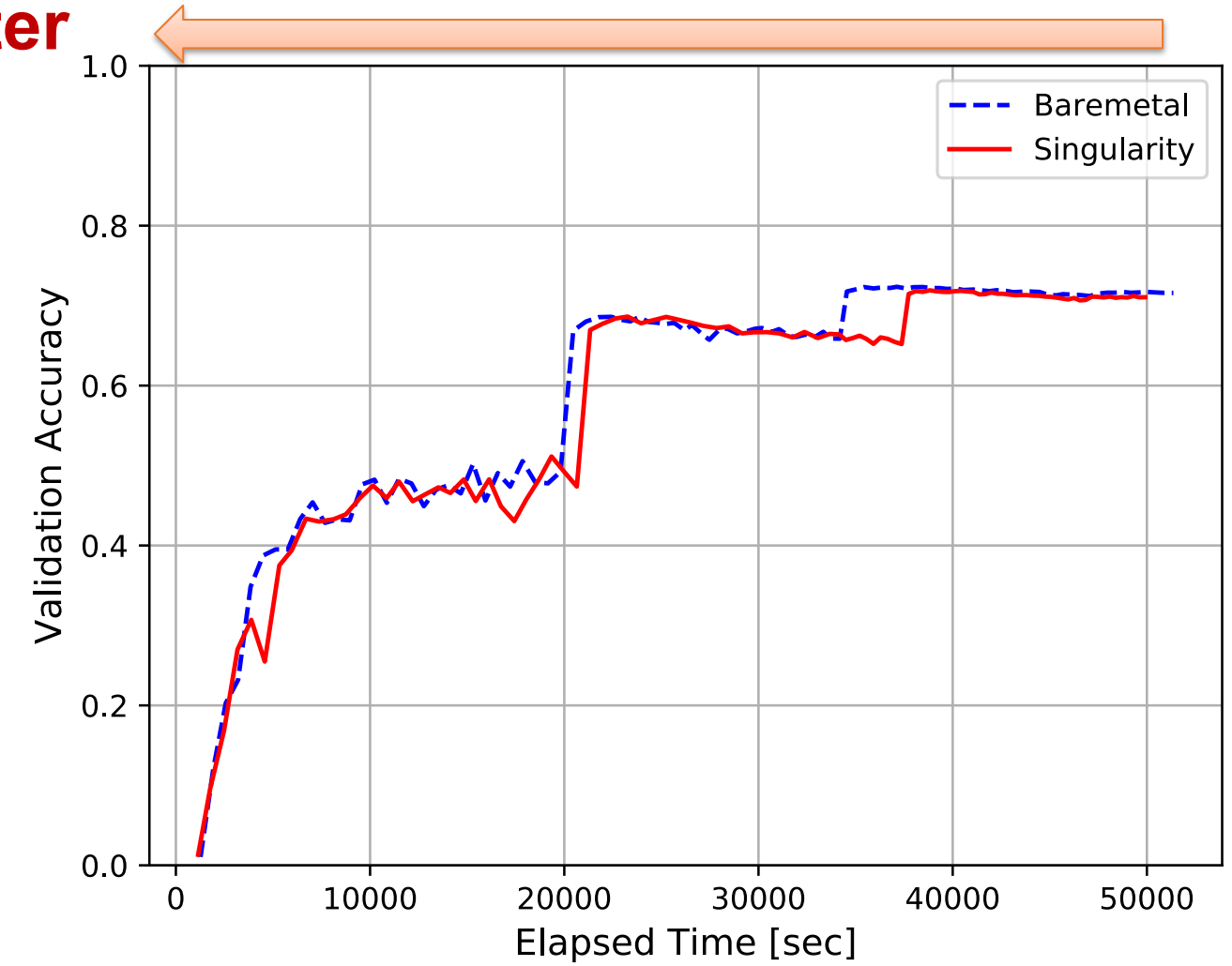
Storage I/O (FIO v2.19)
IOPS (qdepth=1, 16)

Better



Performance Studies for Distributed Deep Learning w Singularity

- Environments
  - AAIC 8 nodes (64 GPUs)
  - Framework: ChainerMN v1.0.0
    - Chainer 2.1.0, Cupy 1.0.3, mpi4py 2.0.0, Python 3.6.1
  - Container
    - Singularity v2.3.1
    - Ubuntu 16.04, gcc-5.4.0, glibc-2.23
  - Baremetal
    - CentOS 7.3, gcc-4.8.5, glibc-2.17,
- Settings
  - Dataset: Imagenet-1K
  - Model: ResNet-50
  - Training:
    - Batch size: 32 per GPU, 32 x 64 in totla
    - Learning Rate: 0.1 per 30 epoch
    - Optimization: Momentum SGD (momentum=0.9)
    - Weight Decay: 0.0001
    - Training Epoch: 100



Discussion

- Singularity
  - Competitive performance to Baremetal
    - Not only Computation, but Memory Bandwidth, Storage I/O, Network
  - Easy to handle OS differences (CentOS vs. Ubuntu)
    - cf.) glibc version differences
  - Simple installation to shared computing resources
    - configure, make, make install (but, root privilege required)
  - Remaining Some Engineering Issues
    - Software Stability
    - Security Issues

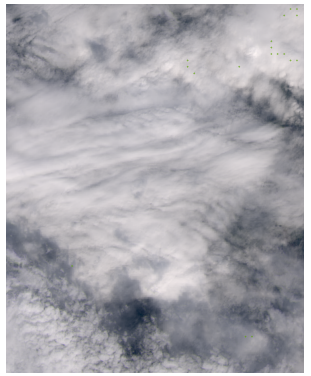
Discussion (cont'd)

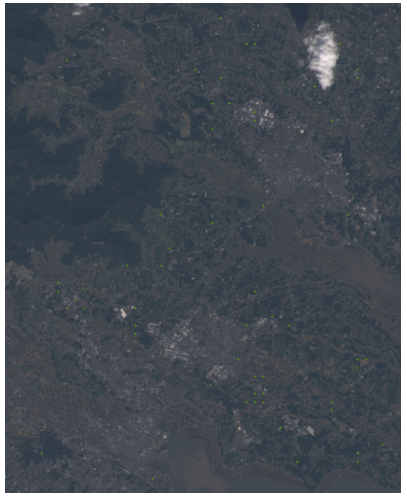
- Are there any problems to run uncertain user's containers?
  - Basically no problems, but it depends on the site policy
    => Define the issues !!
  - Singularity provides some security mechanisms:
    - Singularity Check command to analyze installed binaries inside the container
    - File-owner- or Path-based limitation to launch Singularity container
    - Read/Write restriction to container image files
- Do we need special container images to the site?
  - Basically, NO. Singularity can run on/with ARM and OPA, etc. (If Linux)
  - Optimized builds are required for best performance
- Can we use ISV Applications?
  - Difficult to include inside the container due to license issues
  - Alternatively, mount host installed ISV apps to container

Discussion (cont'd)

- Can we run demonized programs, such as web services and databases, etc.?
  - Yes. Singularity instance command helps to run such instances in the background.
  - User privilege

=> Under investigation using
our AI application prototype (Web Services)





Power Plant Detection from Satellite Data

https://github.com/gistairc/MUSIC4P3

Summary

- Software Ecosystems for AI Cloud
  - AI needs both HPC and Cloud/Big Data technologies
    - Hardware, Software, etc.
  - How to fill the chasm?
- Case study of Singularity (User-level Container)
  - Containerization of Distributed Deep Learning Frameworks
  - Preliminary Performance Studies w/ Emulated AI Workloads
    - Competitive performance to Baremetal

ABCI AI Bridging Cloud Infrastructure

The worlds first large-scale Open AI Infrastructure

Serving your AI needs in Spring 2018