Enterprise MLOps Cloud Open Source

OPEN Compute Project ® APAC Tech Day 2023

lablup

Backend.AI:

Open-source AI hyperscaler platform for everyone

Jeongkyu Shin
Lablup Inc.

backend AI

lablup

CLOUD NATIVE COMPUTING FOUNDATION SILVER MEMBER

THE LINUX FOUNDATION SILVER MEMBER

Mission

Solve speed imbalance
between modern scientific research and its application fields,
By proposing new paradigms for computing-based research
via cloud computing and AI technologies.

Service / Product

Backend.AI (https://www.backend.ai)

A sophisticated GPU PaaS that
Simplify, Unify and Accelerate processes
which enable users to train & serve ML models
on cloud or on-premises with ease.

Technologies

- In-house container orchestrator
- Linux kernel system call virtualization
- Driver-level GPU partitioning virtualization
- Hybrid cloud platform technology
- Directly connected security layer between nodes
- Real-time application layer composition within containers

Customer References

SAMSUNG ELECTRONICS · kt · LG · LOTTE · DSME · CJ · Samsung Research

한국은행 THE BANK OF KOREA · 건강보험심사평가원 · ETRI · TTA · KAERI · KIST · 대한민국해군 REPUBLIC OF KOREA NAVY

GNU 경상국립대학교 · GIST · KMU · CIAST

70+ and growing!

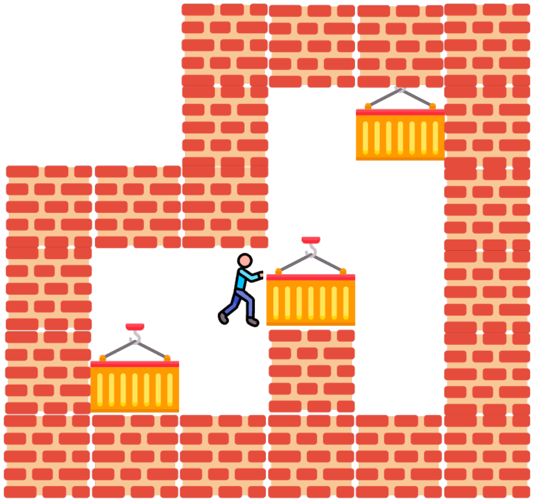
Make AI Accessible

We strive to make AI accessible
by everyone, everywhere.

In order to achieve this goal, we address:
The complexity issues of AI systems, and
The cost issues when applying AI in real-world situations.

4 Topics

- Problem & Our approach
- Backend.AI + Sokovan
  - Summary
  - History
- Characteristics
  - Multi-level scheduler
  - Dynamic allocation
  - NUMA-aware resource mapping
  - Multi-node clustering for training / inference
  - Node subsystems
- Practical cases
- Demo



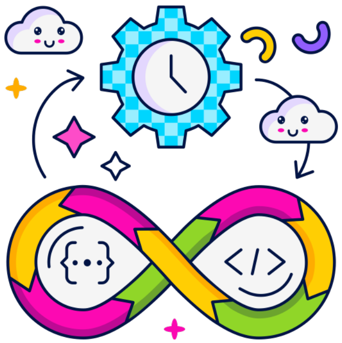
backend AI

We'll Get You Every Last Bit.

Backend.AI: **The Problem**

ML/AIOps System Requirements

- **A new era of the AI world**
  - Unprecedented pace of evolution
    - ✓ 90 days release cycles of TensorFlow in 2018-2019
    - ✓ 1.5 years gap between the NVIDIA GPU generations
  - BigData-like *scale* requirements + HPC-like *performance* requirements
  - *Batch* jobs + *Interactive* jobs
- **HPC challenges**
  - Sensitivity to resource mapping and hardware layouts
  - All the latest hardware acceleration technologies (GPU, NVLink, RDMA, ...)
  - Heterogeneity of the infrastructure
- **AI challenges**
  - Fast cycles of experimentation & deployments
  - Complexity of managing software stacks



7 AI/HPC Workload : Practical Problems

Heterogeneity of resource demands

CPU-intensive: Data preprocessing, analysis, feature extraction...

GPU-intensive: model training, validation, A/B test, latency-free inference...

I/O-intensive: data manipulation, batch, inter-GPU communication...

Complexity of I/O acceleration options

Inter-node GPU–GPU peering

GPUDirect I/O

GPUDirect Storage

Resizable-BAR

Tiered storage caching

HBM2,3 / DDR5 / CXL2,3

Block storage + Filesystem

Never-ending compatibility issues

Software

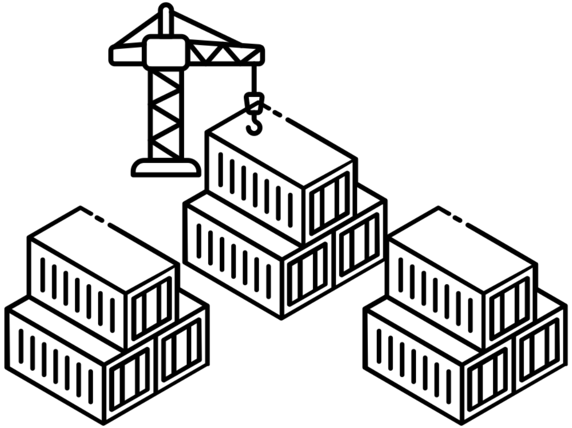
F77/F90 to Julia, TensorFlow 1/2, PyTorch 1/2, JAX, Haiku, ...

Hardware

CPU-only / SSE, AVX-based, CUDA, ROCm, Google TPU/Coral, GraphCore IPU, Habana ...

Our Approach (v1)

- **Let's make containers as the intrinsic abstraction of the workload units**
- **Containers**
  - Minimal performance impacts
  - Faster deployments
  - Isolation of complex software stacks
  - Reproducible setups
- **What did we have in 2015...?**
  - Slurm, IBM LSF
  - Docker v1.7 ~ v1.9
  - Kubernetes v0.x (Google Borg)
  - No nvidia-docker yet... (v1.0 released in 2017)



9 Problem

- Q. Could we combine the strong parts of Slurm and Kubernetes?

Slurm

- ✓ HPC-oriented batch job scheduler
- ✓ Tailored for long-running computing tasks
- ✓ Manual NUMA-aware job placement
- × Multi-tenant security
  - Requires the "host" mode networking even with containers
- × Automatic node setups
  - Packages, container images, etc.

Kubernetes

- ✓ Microservice-oriented container orchestrator
- ✓ Tailored for short-lived user requests
- ✓ Multi-tenancy & Auto-scaling
- × Suboptimal abstraction for resource-demanding batch workloads
  - Requires "acrobatics" to adjust many knobs hidden somewhere (e.g., pod preemption policy, HPA sync period, sidecar container lifecycle, pipeline storage, ...)

We can ultimately accomplish what we need to, but it takes more effort than it "should".

https://betterprogramming.pub/kubernetes-was-never-designed-for-batch-jobs-f59be376a338

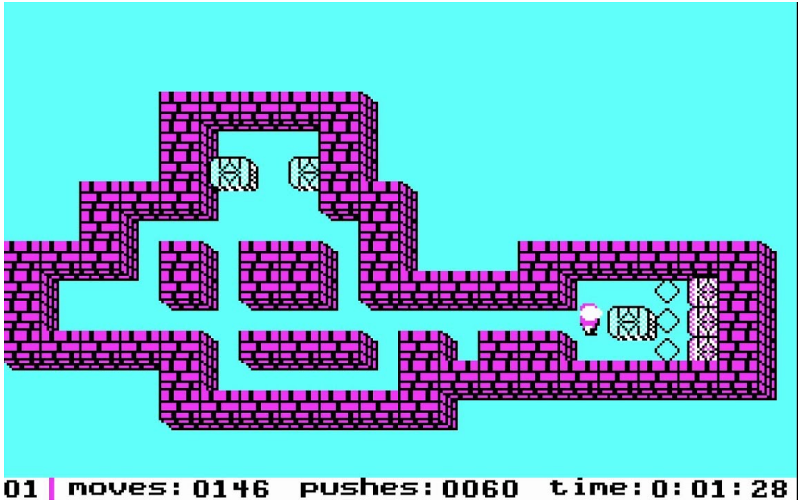
10 Our Approach (v2)

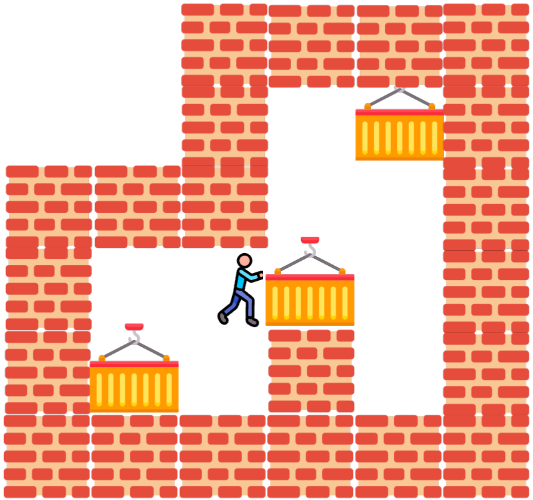
- Let's build a new container orchestrator for AI/HPC from the ground up!
  - Embrace **both batch (training) & interactive apps (dev & inference)**
    - ✓ Job queues & scheduler (**Sokovan**) for batch jobs like ML training, data processing, ...
    - ✓ App proxy for interactive apps like Jupyter notebooks, code-server, Triton Server, ...
  - **Unleash the potential of latest hardware advancements** (NUMA, RDMA, GPUDirectStorage, ...)
  - Full-fledged **enterprise-grade administration** (users, keypairs, projects, billing, stats, ...)

- Pros
  - ***Super-fast***: native integration with hardware details (NUMA, GPUDirectStorage, etc.)
  - ***Super-customizable***: plugin architecture for schedulers, accelerators, storage, etc.
- Cons
  - Extra efforts to integrate with the existing ecosystem (...but we have Docker!)

Sokovan: **Introduction**

12 Sokovan: From sokoban game





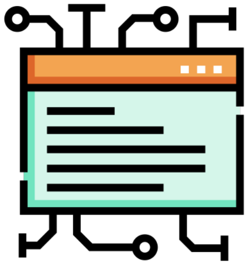
Sokovan: Design Principle

14 Sokovan: Design Principle

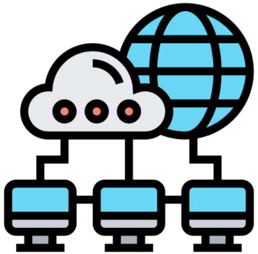
- **Flexible compute session (No *Pod*!)**
  - Bundles one or more containers created on the fly (no pre-occupation)
    - ✓ Containers are more like volatile processes with an overlay filesystem attached.
  - Implements persistent storage via volume mounts
- **Customizable scheduler**
  - Heuristic FIFO, DRF (dominant resource fairness), user-written algorithms
- **Multi-tenancy first**
  - Goal: serve as a public SaaS
  - Dynamic namespacing & partitioning instead (resource groups, scoped configuration)
  - Decouples user/project from Linux user/group (e.g., for sharing data volumes)
    - ✓ e.g., SSO plugins, Keystone integration

Sokovan: Component Design

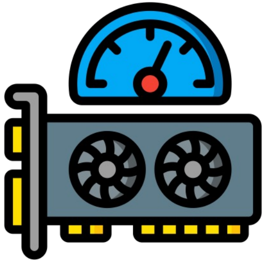
- Fully acceleration-aware, multi-tenant, batch-oriented job scheduling



- Combines a cluster-level node assignment scheduler and a node-level resource/device assignment scheduler



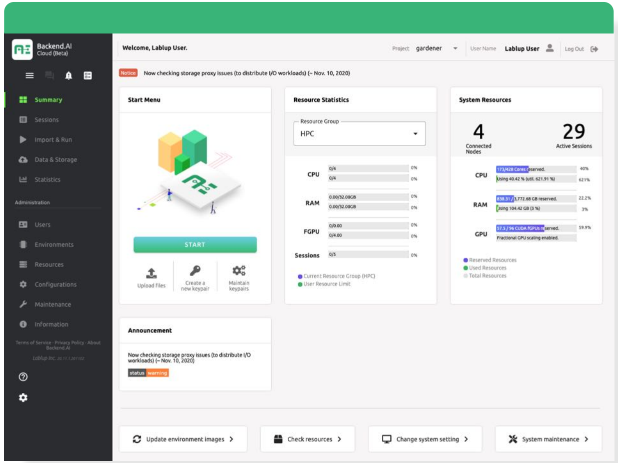
- Job subsystem: manages docker, `containerd` and k8s cluster agents



- Fully integrates multiple hardware acceleration technologies into various system layers to unleash the potential performance

Backend.AI: Open-source AI Platform for Everyone

All-in-one Enterprise Operating Platform for AI Development and Services



PyTorch, DL4J DEEPLEARNING4J, dmlc mxnet, PaddlePaddle, TensorFlow, Caffe, Microsoft Cognitive Toolkit, torch, Keras

backend AI

Product Goals

- Provide **top-class computing performance** with **latest hardware acceleration**
- **Maximize utilization of computing resources** by concurrent **multi-tenancy**
- Hide the underlying hardware & software complexity from AI devs and data scientists
- **Automate resource management** & scaling
- Provide **enterprise-grade stability** with professional support
- Cooperate with **your favorite tools and frameworks**

Backend.AI: Tech stack

- Open-source (as a component of Backend.AI)
- Monorepo with Pantsbuild
- Multi-architecture support
  - x86-64, Arm64 (aarch64), RISC-V (w/ selected board)
- Operating System
  - Linux, Windows (WSL), macOS
- Runtime backend
  - Baremetal / OpenStack + Docker / Podman
  - Docker (Snap) / Docker (systemd) / Docker (native) / Docker Desktop / OrbStack
- Prerequisites
  - Python 3.11 (23.03) / stand-alone python
  - PostgreSQL 14 / Redis 7 / etcd 3.5

Backend.AI: History

- Debut at PyCon KR 2015 (Aug 2015)
- Open-sourced since 2017 (Sorna, Backend.AI)
- OpenStack-ready talk at OpenInfra Days Seoul 2018
- Backend.AI Container Pilot component is now known as Sokovan (Dec 2022)
- Now operates many AI clusters / supercomputers around the world
  - Runs ~10,000 Enterprise GPUs

aws

Microsoft Partner Network

NAVER CLOUD PLATFORM

NVIDIA

NVIDIA DGX-Ready Software Partner

PURESTORAGE

Tech. Alliance Partner

NetApp

DELL

Cloud Service Provider

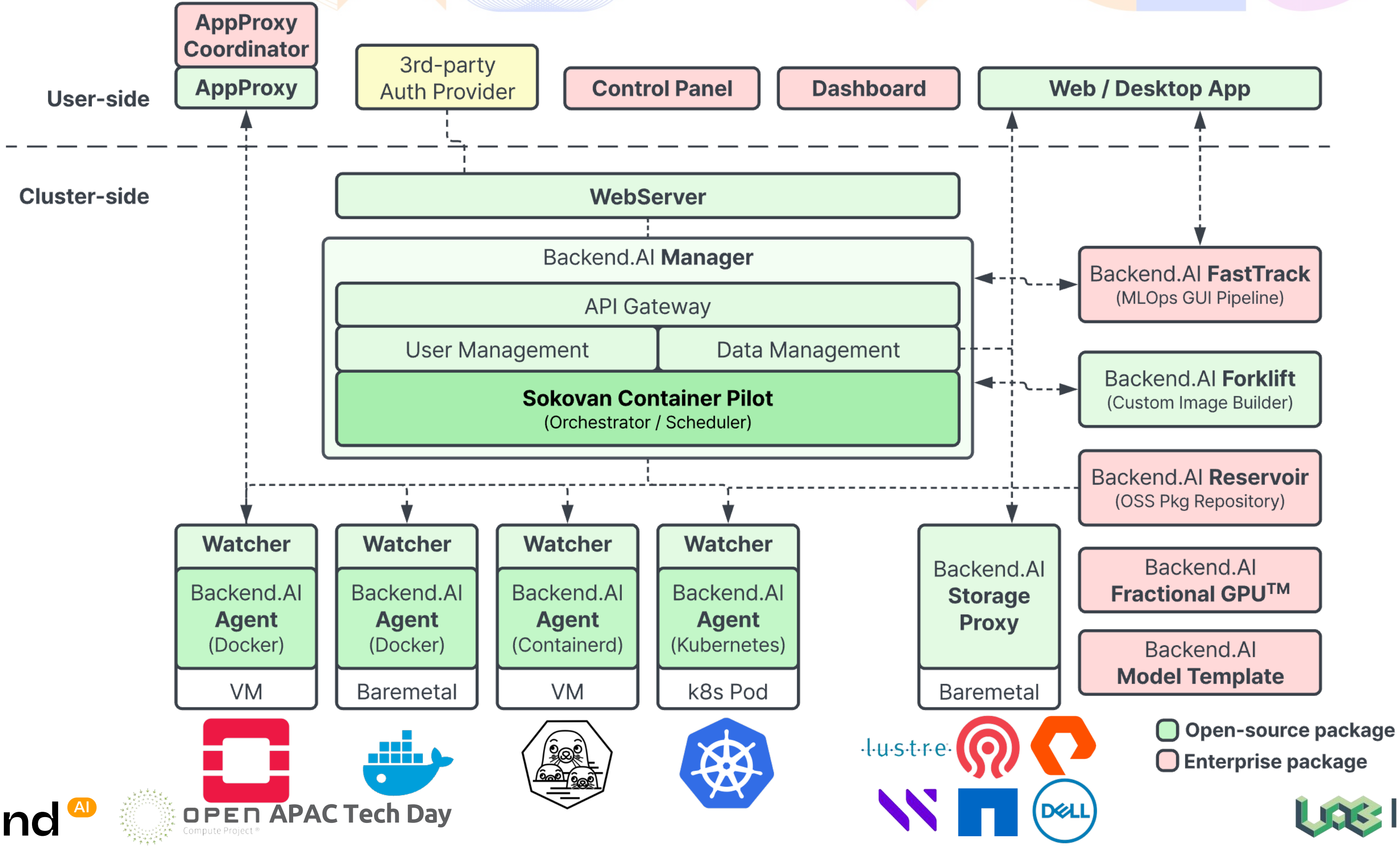
SAMSUNG ELECTRONICS · kt · LG · LOTTE · DSME · CJ · Samsung Research

BANK OF KOREA · 건강보험심사평가원 · ETRI · TTA · KAERI · KIST · 대한민국해군 REPUBLIC OF KOREA NAVY

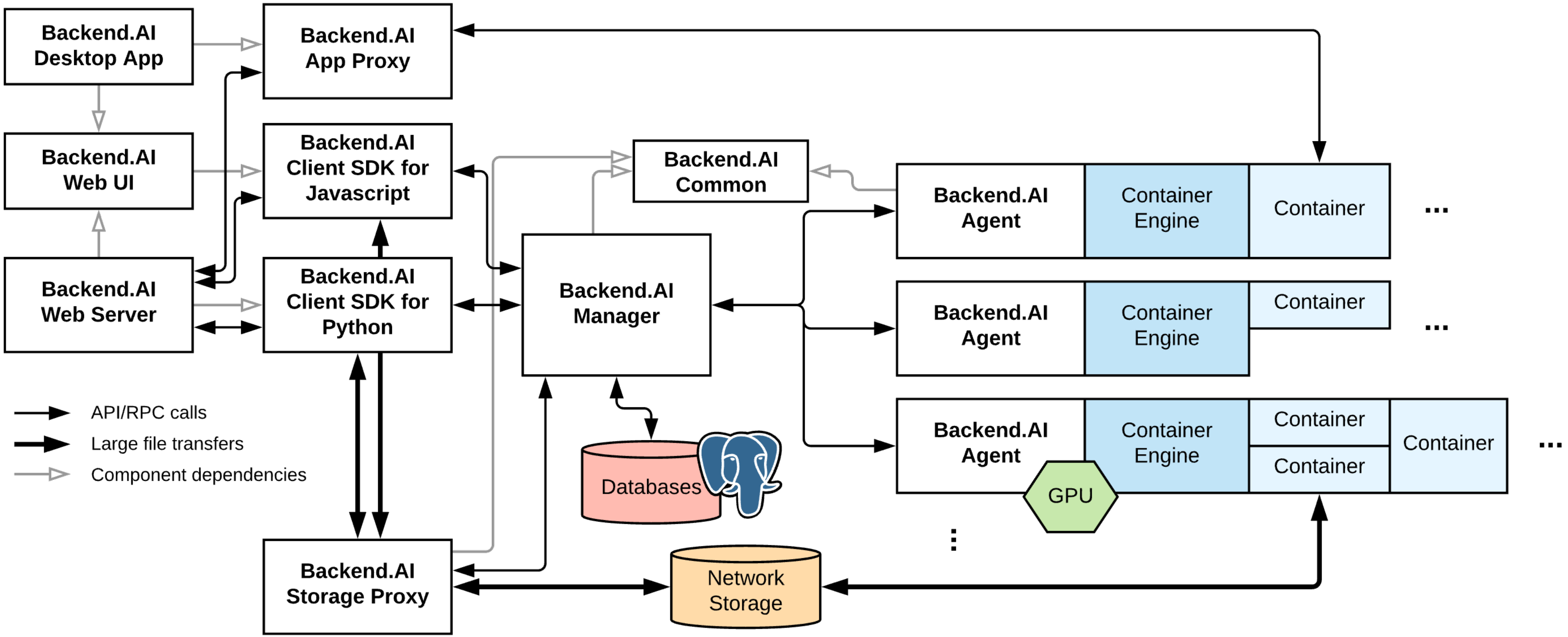
GACHON UNIVERSITY · GNU 경상국립대학교 · GIST · KMU · POSTECH · HANYANG UNIVERSITY · CIAST

70+ and growing!

Backend.AI: Components



20 Backend.AI: Architecture



One-Liner to Kickstart Your Journey

- Development setup:

```
$ git clone https://github.com/lablup/backend.ai
$ cd backend.ai
$ bash ./scripts/install-dev.sh      # Default agent
$ bash ./scripts/install-dev-k8s.sh  # (Optional for k8s agent)
```

- Production setup:

```
$ pip install backend.ai-manager              # Manager
$ pip install backend.ai-agent                # Compute agents
$ pip install backend.ai-storage-proxy        # Storage proxy
$ vi ~/.config/backend.ai/{manager,agent,storage-proxy}.toml
```

Sokovan: **Characteristics**

Sokovan: Harnessing Cutting-Edge Capabilities

Multi-level Scheduler	Dynamic & Fractional GPU Allocation	NUMA-aware Resource mapping	Multi-node multi-container clustering
Heterogeneous Agent Backends	Resource Group & Namespacing	GPU/NPU Abstraction	I/O Acceleration plane

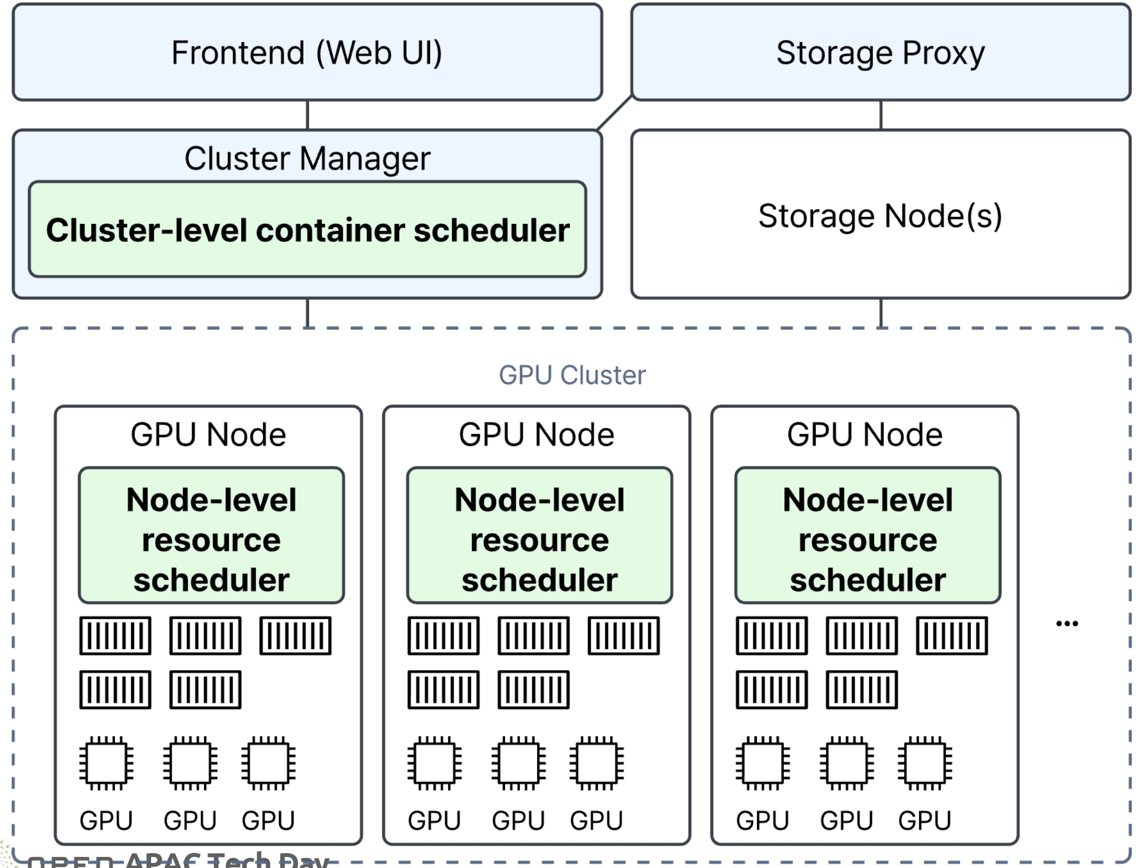
24 Sokovan: Harnessing Cutting-Edge Capabilities

Since we do not have enough time...

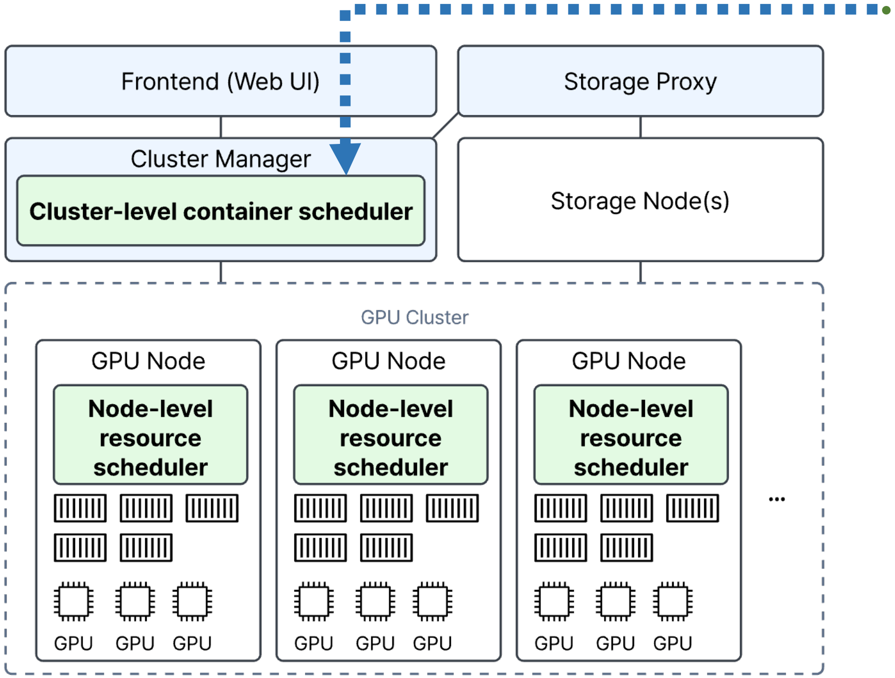
Multi-level Scheduler	**Dynamic & Fractional GPU Allocation**	**NUMA-aware Resource mapping**	**Multi-node multi-container clustering**
Heterogeneous Agent Backends	Resource Group & Namespacing	GPU/NPU Abstraction	I/O Acceleration plane

If you are interested in others, please come to us after the talk!

Multi-level scheduler



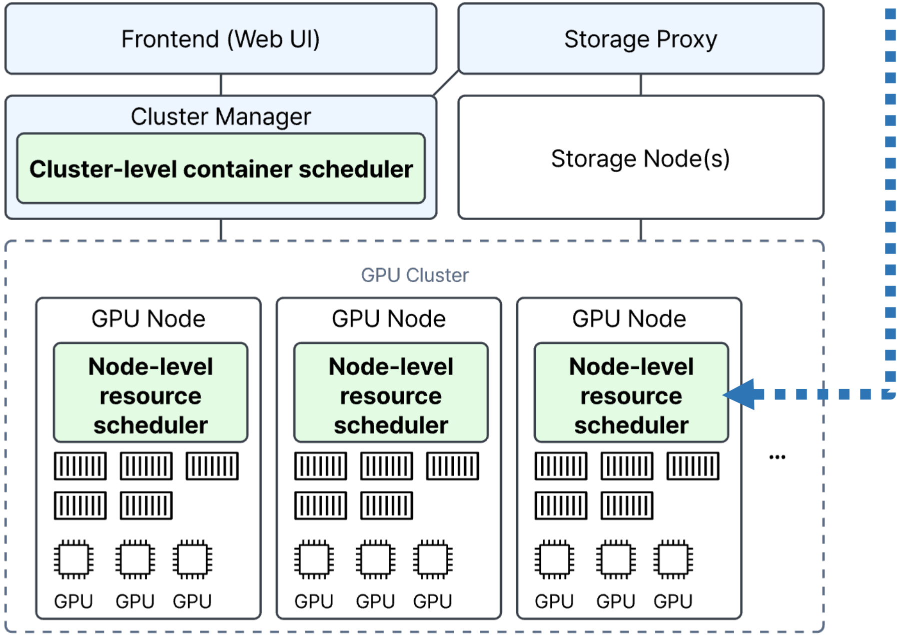
Multi-level scheduler / Cluster-level



Cluster-level scheduler (Manager)

- Controls the density and priority of workloads
- Performs iterative two-phase scheduling per resource group
  - ✓ Which session to schedule first?
  - ✓ Which node to assign the selected session's containers?
- The scheduler plugin interface
  - ✓ Each plugin defines the implementation for the above two phases.
- Included schedulers
  - ✓ Heuristic FIFO (to prevent HoL blocking)
  - ✓ LIFO
  - ✓ DRF (dominant-resource fairness)

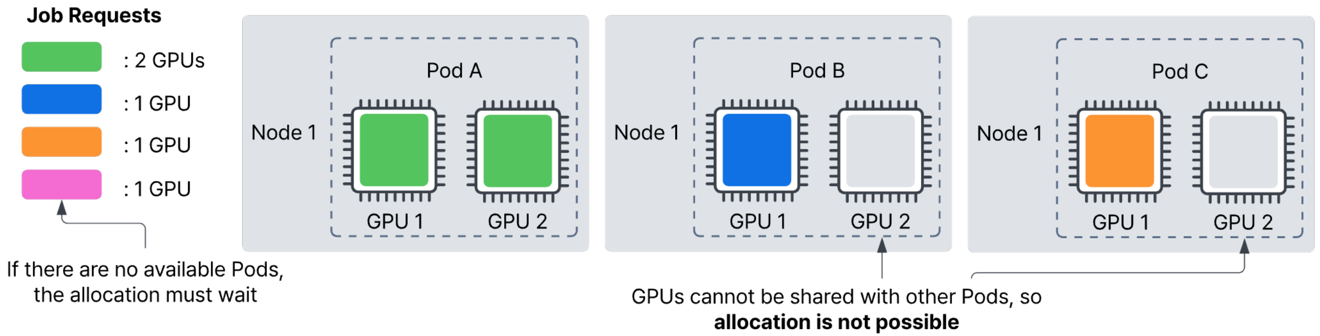
27 Multi-level scheduler / Node-level



- **Node-level resource scheduler (Agent)**
  - Optimizes the per-container performance by smartly mapping containers and devices (CPU cores, GPUs, etc.)
  - The compute plugin interface
    - ✓ Each plugin reports the hardware config with the capacity and layouts
  - Included compute plugins
    - ✓ CPU and memory (intrinsic)
  - Extensions
    - ✓ NVIDIA CUDA, AMD ROCm, Google TPU, Graphcore IPU, ...
    - ✓ Utilizes the NUMA topology information provided by NVML and `libnuma`
    - ✓ Auto-configures NCCL based on Infiniband RDMA and GDS (GPU Direct Storage)

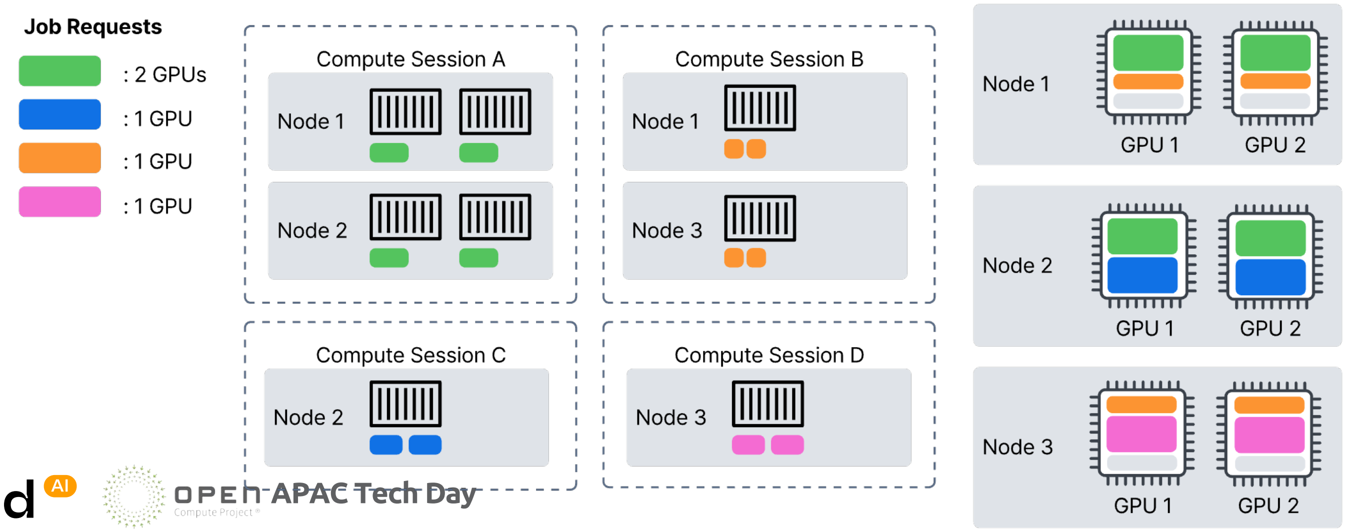
Dynamic GPU Allocation: Powering Up with Sokovan

- **Generic Kubernetes Pod-based GPU resource allocation**
  - Maps GPU and other computing resources in the Pod level only
  - Creates Pods in prior and assigns Jobs to the Pods
  - Some jobs may be pending due to inflexibility of sparing resources from existing Pods

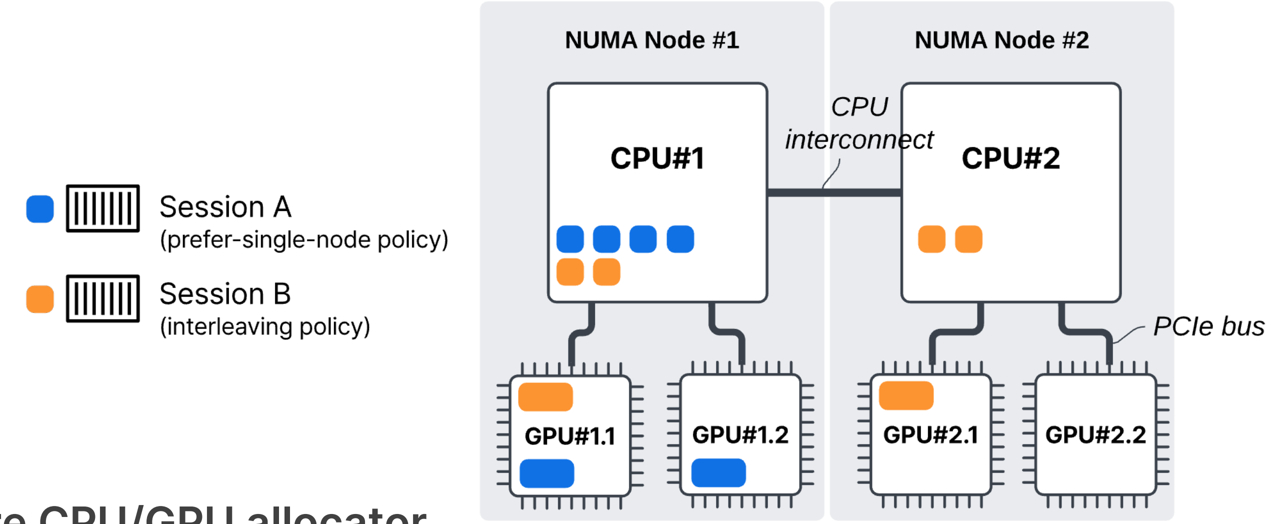


29 Dynamic GPU Allocation: Powering Up with Sokovan

- **Dynamic GPU allocation with Sokovan / Backend.AI**
  - Accommodates all Jobs (in contrast to above) with higher GPU utilization
  - Fractional GPU scaling allows more fine-grained resource distribution
  - Dynamically creates and deletes the sessions upon job scheduling decision
  - Allocates and reclaims the resources as soon as the Session is created and deleted



NUMA-aware resource mapping



- NUMA-aware CPU/GPU allocator
  - Offers two different policies: **interleaving / prefer-single-node**
  - Auto-configures the CPU affinity mapping of containers <u>based on GPU assignments</u>
  - Fully compatible with **Weka.io** Agents configured for GPU Direct Storage which requires every NUMA node that has assigned GPUs to be activated in containers
  - Supports an arbitrary number of NUMA nodes (1/2/4/8/...)

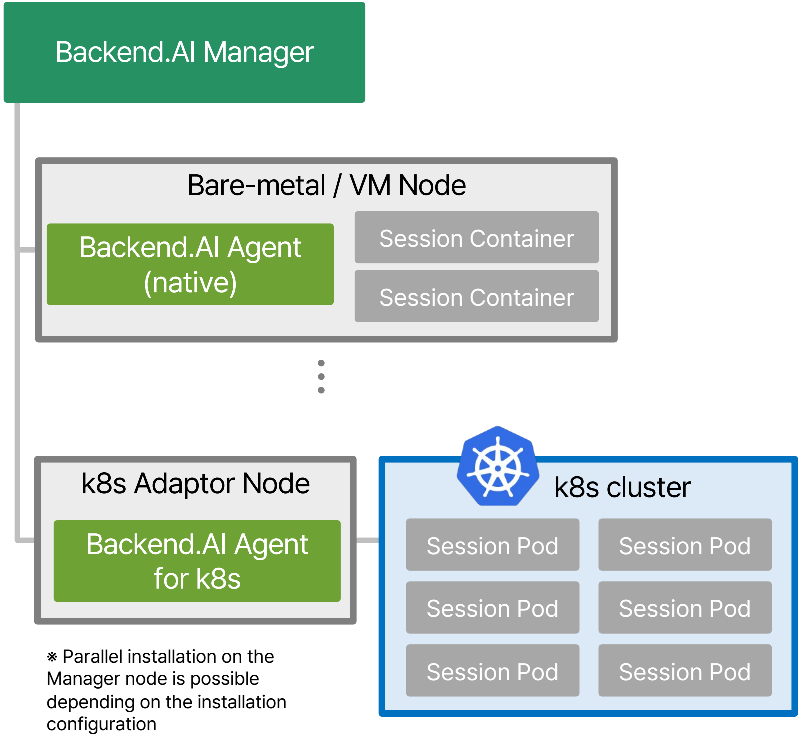
Multi-node multi-container clustering

- **Bundles multiple distributed containers into a single compute session**
  - Interconnect (control-plane): overlay networks (multi-node) / bridge networks (single-node)
  - Interconnect (data-plane): NCCL + Infiniband RDMA
  - Interconnect (storage-plane): Infiniband + GPUDirectStorage
  - Users interact with the primary ("main1") container.
  - Containers may have different roles with each role's own indices.

Shell Environment Variable	Example	Equivalent in..
BACKENDAI_SESSION_ID	3614fdf3-0e04-40c3-88cc-c573549a528a	Session
BACKENDAI_KERNEL_ID	3614fdf3-0e04-40c3-88cc-c573549a528a	Kernel
BACKENDAI_CLUSTER_MODE	single-node	Session
BACKENDAI_CLUSTER_SIZE	3	Session
BACKENDAI_CLUSTER_HOST	main1	Kernel
BACKENDAI_CLUSTER_HOSTS	main1,sub1,sub2	Session
BACKENDAI_CLUSTER_REPLICAS	main:1,sub:2	Session

Heterogeneous Agent Backends

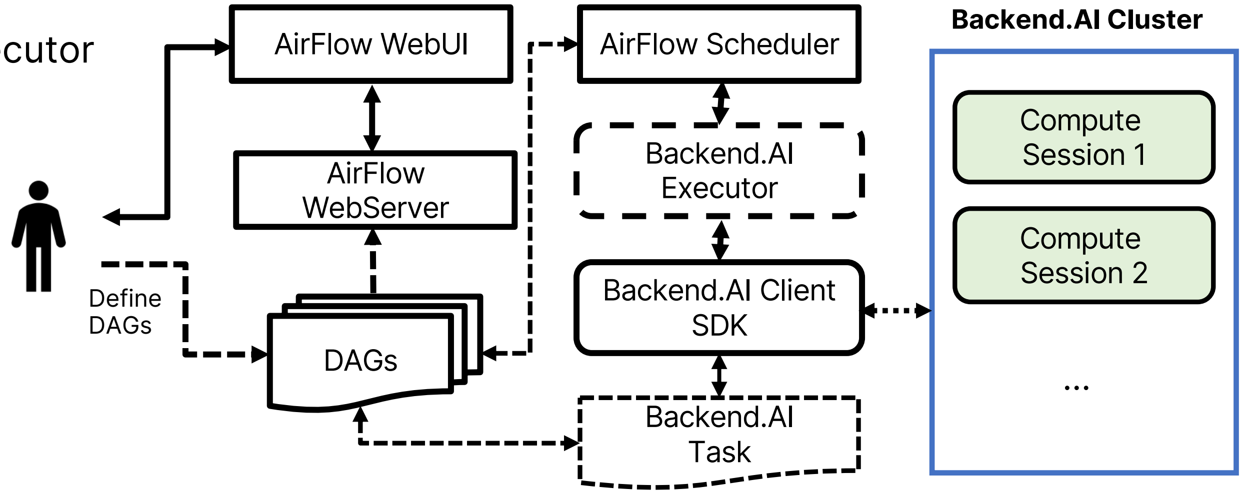
- **Integrates other work unit provisioners**
  - Work unit may be a container, VM, or native Linux process
  - ***Kubernetes*** agent backend
    - ✓ Attach an entire k8s cluster like a single compute agent
    - ✓ Scheduling / queueing is handled by Sokovan: the k8s-side queue is always empty
  - ***OpenStack*** agent backend *Alpha
    - ✓ Integrated OpenStack VM management
    - ✓ Unified API for both container / VMs
- **API-level compatibility layer *Alpha**
  - Let k8s clients to control Backend.AI
  - API Conformance: targeted to Backend.AI 23.09



e.g. k8s subsystem as compute agent

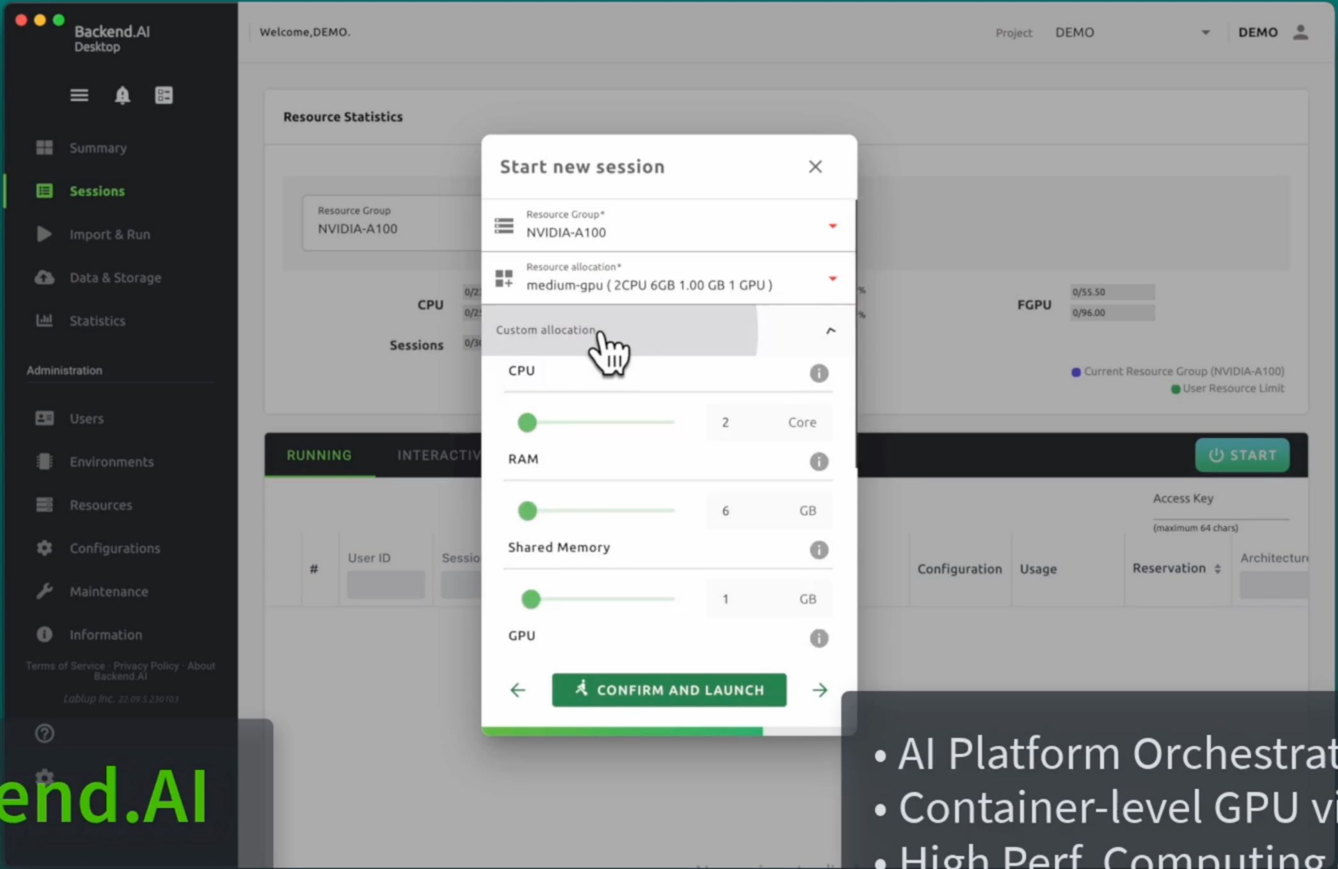
33 Integration with MLOps

- Apache AirFlow
  - Run as task / executor



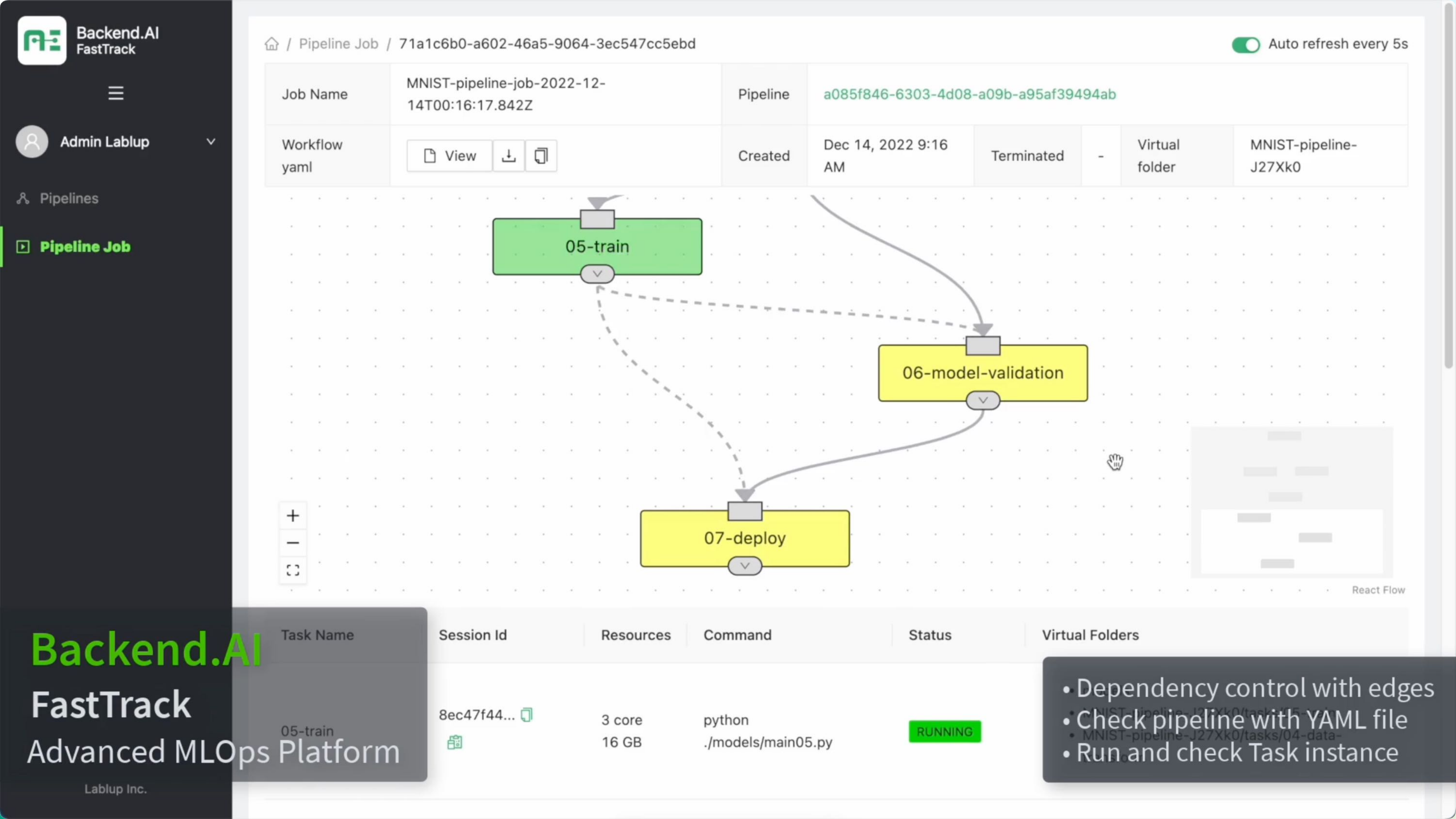
- MLFlow
  - MLFlow can be run as **instant MLOps platform** with Backend.AI session
- FastTrack
  - Lablup's own MLOps platform

Demo



Backend.AI
WebUI

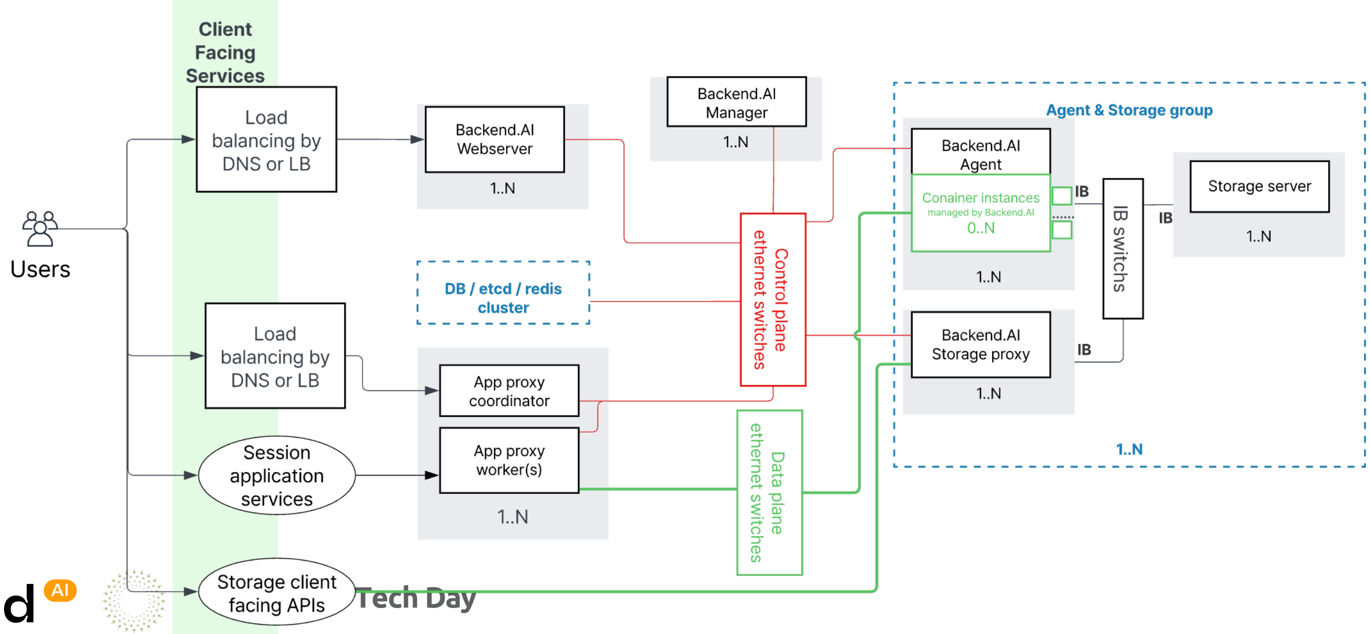
- AI Platform Orchestration
- Container-level GPU virtualization
- High Perf. Computing optimization
- App & Storage in compute session


Backend.AI
FastTrack
Admin Lablup
Pipelines
Pipeline Job
Pipeline Job / 71a1c6b0-a602-46a5-9064-3ec547cc5ebd
Auto refresh every 5s
Job Name
MNIST-pipeline-job-2022-12-14T00:16:17.842Z
Pipeline
a085f846-6303-4d08-a09b-a95af39494ab
Workflow yaml
View
Created
Dec 14, 2022 9:16 AM
Terminated
-
Virtual folder
MNIST-pipeline-J27Xk0
05-train
06-model-validation
07-deploy
React Flow
Task Name
Session Id
Resources
Command
Status
Virtual Folders
05-train
8ec47f44...
3 core
16 GB
python ./models/main05.py
RUNNING
Backend.AI
FastTrack
Advanced MLOps Platform
Lablup Inc.
• Dependency control with edges
• Check pipeline with YAML file
• Run and check Task instance

Backend.AI: **Field Studies**

Practical cases

- General System configuration
  - Sokovan orchestrator: simultaneously achieves overall cluster system optimization and node-level optimization, installed on Backend.AI manager and agents
  - Network: Completely split planes for user / data (eth), storage (IB) and inter-node GPU comm. (IB)



Training large language models to the theoretically maximum performance

Test specification			
16-node cluster GPU: NVIDIA A100 80GB x 8 (Max. FLOPS per GPU: **150 TFLOPS**) Clustering platform: Backend.AI 22.03.8			
Cluster	**CPU**	**RAM**	**GPU**
Per-node	AMD EPYC 7742 64-core x 2	1024GB 640GB (GPU)	NVIDIA A100 80GB x 8
Total	AMD EPYC 7742 64-core x 32	16384GB 10240GB (GPU)	NVIDIA A100 80GB x 128

Test condition	
World size	128
Data parallel size	128
Model parallel size	1
Batch size	64
Parameter size	7.66B (=7661.3M)
Tested	2022/12/05 08:20:28

Summary				
Trial	**GPU#**	**# of param.**	**FLOPS per GPU**	**Total FLOPS**
#1	128	7.66B	**145.39 TFLOPS**	18.60 PFLOPS
#2	128	7.66B	**145.50 TFLOPS**	18.62 PFLOPS

- **Model / system**
  - Training with *Megatron-Deepspeed* (ZeRO-2 optimizer)
  - Automatic GPU-GPU network configuration
  - GPUDirect storage for training data I/O
- **Achievements**
  - Approached the **maximum theoretically achievable GPU performance**
  - Less than **1% difference from that achieved in bare-metal workloads** based on Slurm

Applying GPUDirect Storage to the large container-based AI cluster

Test specification			
13-node cluster Clustering platform: Backend.AI 22.03.8			
Cluster	**CPU**	**RAM**	**GPU**
Per-node	AMD EPYC 7742 64-core x 2	1024GB 640GB (GPU)	NVIDIA A100 80GB x 8
Storage	Samsung PM1733/5 PCIe x4/dual port 4TB SSD x4		

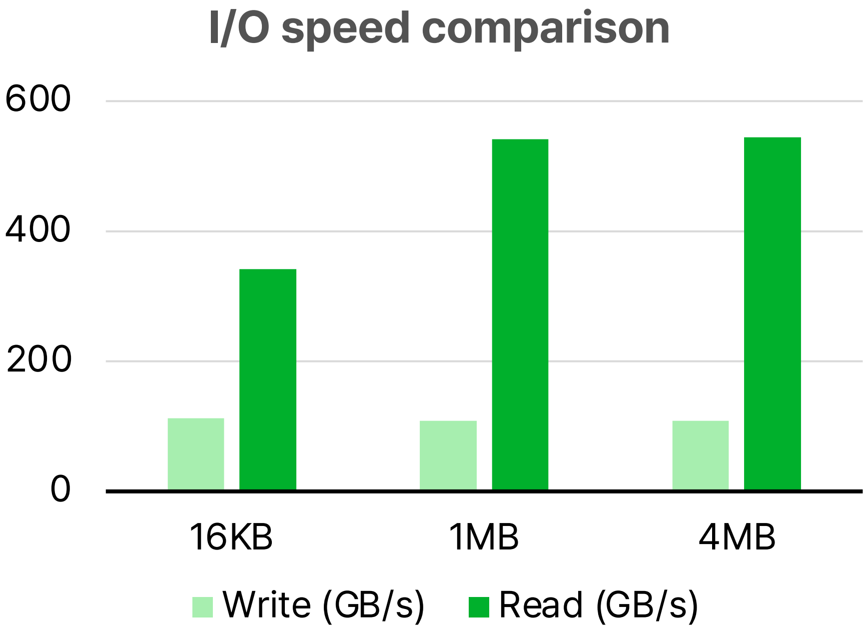
Test condition	
# of processors	100
Tested	2022/11/30 16:10:31

Summary

File size	I/O Type	Max. Speed (Mb/sec)	Max. Speed (OPS)
16KB	Write	114724.27	7342353.15
	Read	350137.17	22498779.04
1MB	Write	111114.38	111114.38
	Read	554428.82	554428.82
4MB	Write	110763.55	27690.89
	Read	557929.82	139482.45

- Magnum IO GPUDirect Storage + Weka.io
  - Achieving network storage access of **150Gb/s or more per second**.
  - The world's first implementation for GPUDirect Storage in a container-based AI cluster

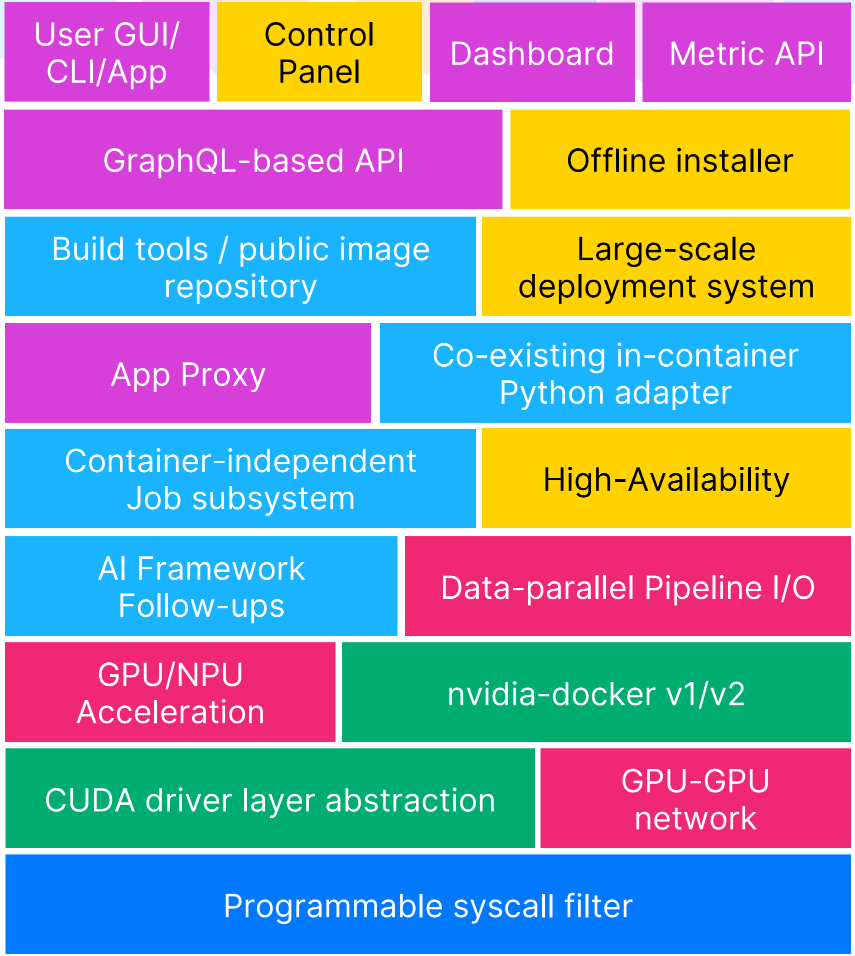
Applying GPUDirect Storage to the large container-based AI cluster



- **Magnum IO GPUDirect Storage + Weka.io**
  - Achieving network storage access of **150Gb/s or more per second**.
  - The world's first implementation for GPUDirect Storage in a container-based AI cluster

41 Recap

- **Designed a new orchestrator based on a completely different abstraction**
  - Easily hackable
  - Solved the various limitations of container for the HPC/AI field
- **Optimized the allocation and deployment of acceleration hardware**
  - GPU, NPU, Network
  - Exploit the full potential performance in multi-node GPU setups
- **Performance comparable to bare-metal workloads in GPU-accelerated clusters**
  - GPU-to-GPU networking and GPUDirect Storage in multi-node setups
  - Achieved the theoretically maximum performance on container clusters



...and more!

Thank You!

Question?

lablup/backend.ai

jshin@lablup.com

backend AI