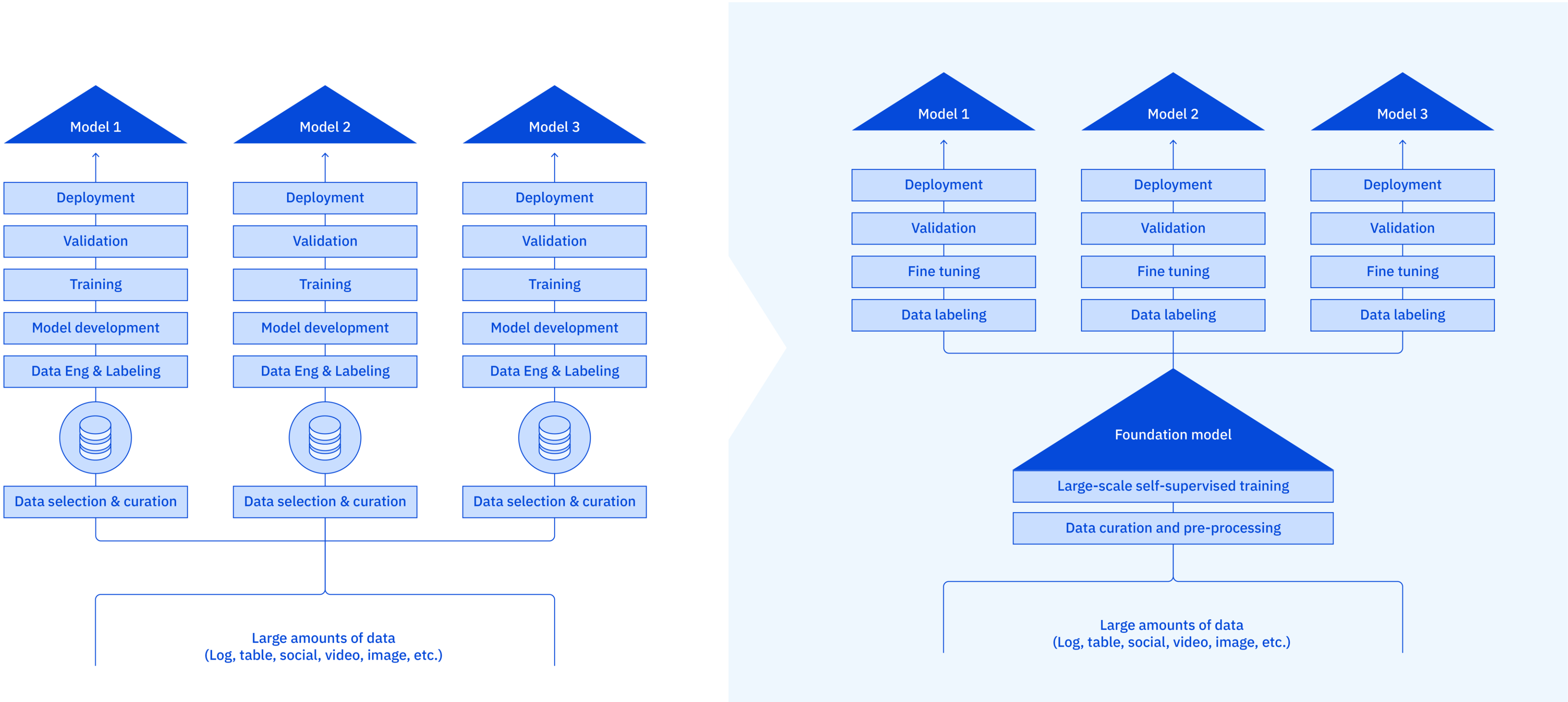
Insights Gained from Delivering Two Generations of AI Supercomputers and Storage Solutions in IBM Cloud

Dr. Seetharami R. Seelam
Distinguished Engineer
AI Infrastructure
IBM Research
sseelam@us.ibm.com

Agenda

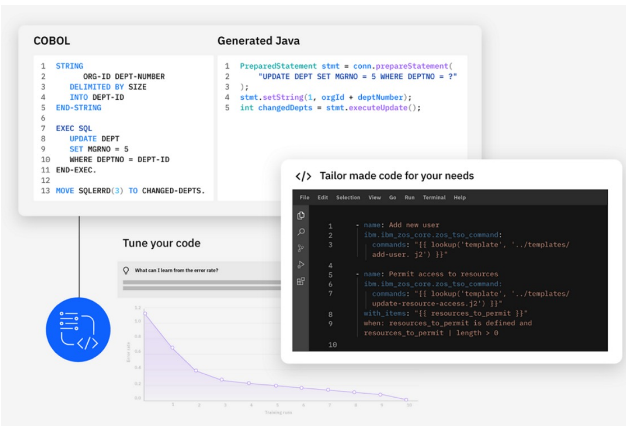
1. What’s new in AI
2. Vela Systems co-design
3. Summary

Starting in 2021, we started to observe a new trend in AI models: Foundation models

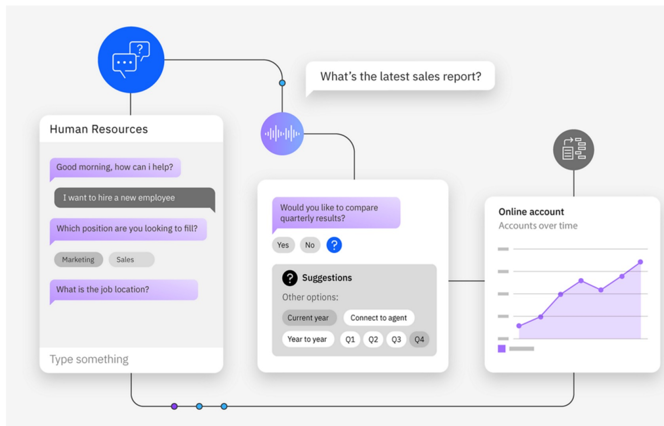


Applications of Foundation Models in the Enterprise

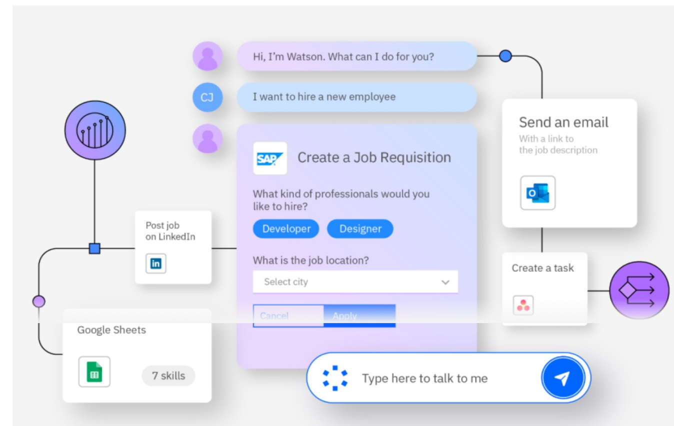
Code Assistants



Customer Service Assistants



Workflow Orchestration Assistants



IBM Granite Models powering applications in the Enterprise

Granite for Code
Trained on 116 programming languages, Granite code models (3b, 8b, 20b, 34b) are optimized for enterprise-grade software development workflows.

Granite for Language
Granite language models (7b open-source, 13B English, 20b multilingual, 8b Japanese) demonstrate higher accuracy and throughput at lower latency, while consuming only a fraction of GPU resources.

Granite for Time Series
Granite Time Series is a family of lightweight, pre-trained models for time-series forecasting trained on a collection of datasets spanning a range of business and industrial application domains.

https://www.ibm.com/products/watsonx-ai/foundation-models
https://huggingface.co/ibm-granite

AI Workloads **need flexible infrastructure**

…to support the whole AI workflow

Data preparation

Workflow of steps (e.g., remove hate and profanity, deduplicate)

Distributed training

Long-running job on massive infrastructure

Model tuning/adaptation

Model tuning with custom data set for downstream tasks

Inference

May have sensitivity to latency, throughput, power

Hours to days

10-2000+ low to mid-end CPU cores

10+ low to mid-end GPUs per

10-100+ concurrent jobs

on-prem **Public clouds**

weeks to months

10-500+ high-end GPUs (per job)

10+ concurrent jobs

on-prem **Public clouds**

minutes to hours

1-64+ mid to high-end GPUs (per job)

100+ concurrent jobs

on-prem **Public clouds**

sub-second API request

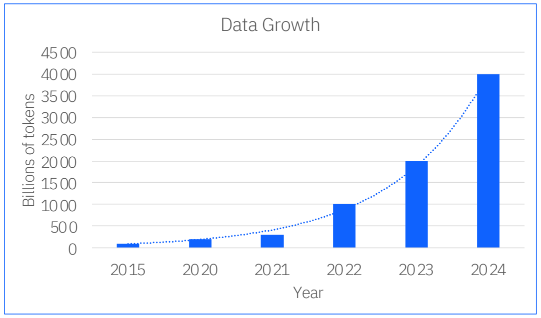
1-16+ GPUs per fine tuning task

Fraction to multiple GPUs per inference, or specialized accelerator

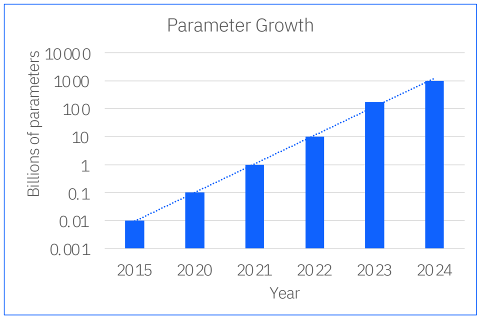
Thousands of API requests

on-prem **Public clouds** **Edge**

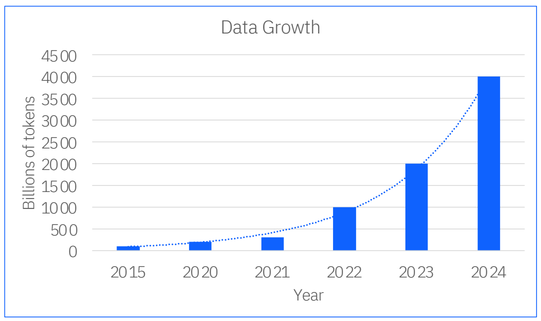
Foundation Models operate on large data

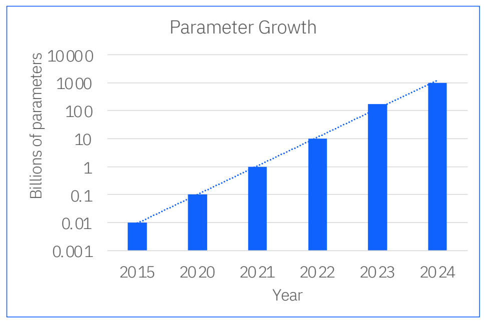


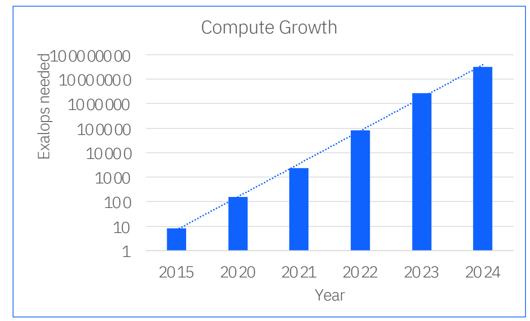
Foundation Models operate on large data to train large number of parameters



Foundation Models operate on large data to train large number of parameters **requiring large amount of compute**







e.g., 1E9
(D)

1E9
(N)

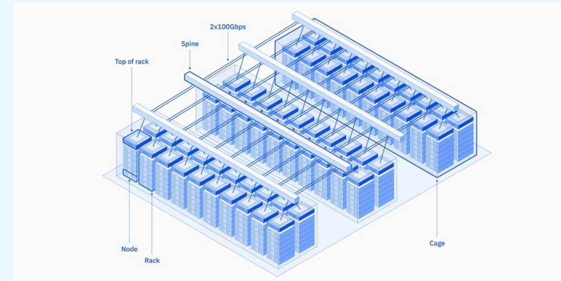
6*1E18
C=~(6*N*D)

Compute cycles required to train a model is linearly proportional to the product of data set size and model parameters

Agenda

1. What’s new in AI
2. Vela Systems co-design
3. Summary

Vela Systems



2022

Vela Part one
- Single Zone Availability
- 200 GigE TCP/IP

2023

Vela Part two
- RDMA Networking
- 400 GigE
- Double Dense Racks

2024

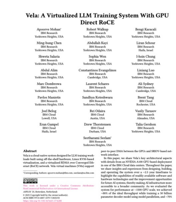
Vela2
- Multi Zone Availability
- 3.2 Tbps GPU RoCEv2 Network
- 400 GigE Cloud Network
- Native Cluster Networking
- Multi Tenant Solution

Vela-on-prem
- Vela deployed at customer sites
- Ethernet network
- H100 GPUs, IBM AIU


IBM — Vela and Blue Vela AI Infrastructure

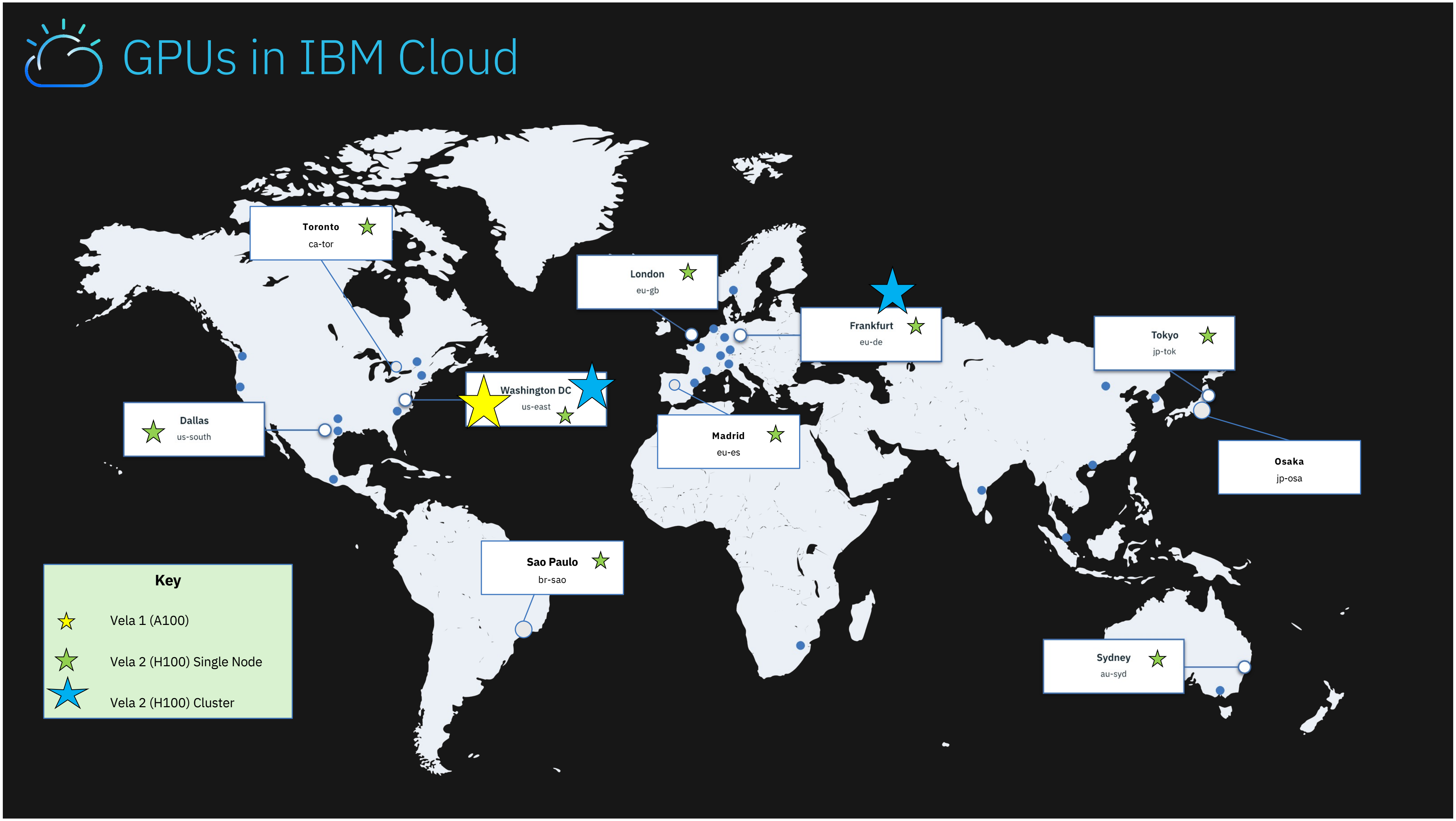
The infrastructure powering IBM's Gen AI model development

https://arxiv.org/pdf/2407.05467

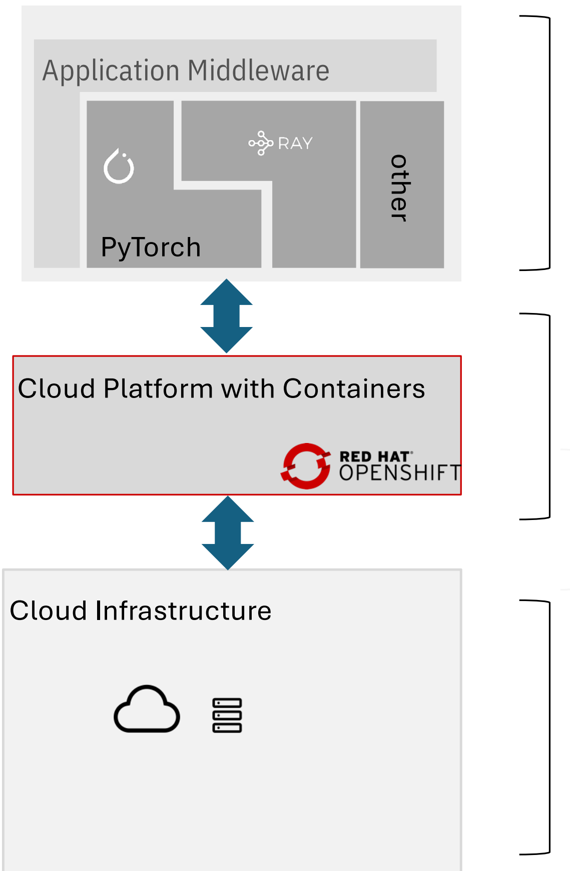

Vela: A Virtualized LLM Training System With GPU Direct RoCE

ASPLOS 2025

AMD MI300X
Intel Gaudi3
IBM AIU


GPUs in IBM Cloud
Toronto
ca-tor
London
eu-gb
Frankfurt
eu-de
Tokyo
jp-tok
Washington DC
us-east
Dallas
us-south
Madrid
eu-es
Osaka
jp-osa
Sao Paulo
br-sao
Sydney
au-syd
Key
Vela 1 (A100)
Vela 2 (H100) Single Node
Vela 2 (H100) Cluster

Co-design opportunities in three layers



Flexible middleware for training, fine tuning and inferencing

How to build optimized runtime libraries that can take advantage of the high performance infrastructure (e.g., FSDP, DDP)?

Cloud Platform

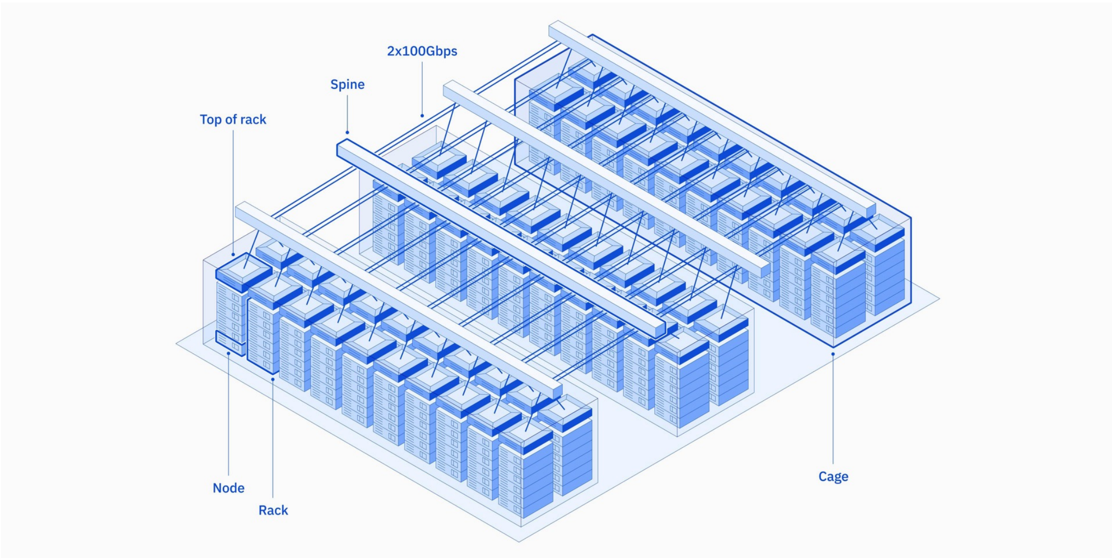
- How to enable high performance compute and network capabilities into containers?
- How to support AI workload life cycle?
- How to automate health checks?

Scale-out infrastructure (e.g. on prem or cloud)

- How to design an elastically scalable system?
- How to build high performance and resilient RDMA network
- How to maximize efficiency of power, space, and cooling?

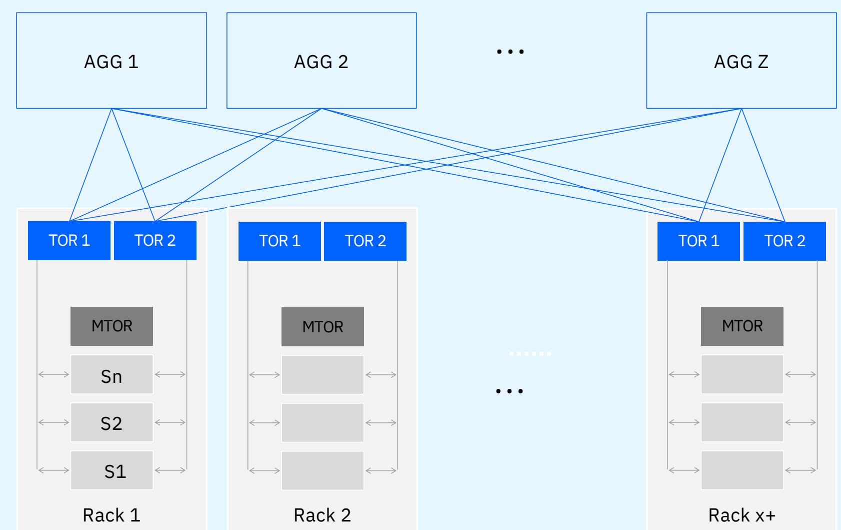
Vela AI System Design Principles

- Performance
  - Achieve state-of-the-art performance with Ethernet network
  - Support AI workload life cycle
- Cloud flexibility
  - Multi-tenancy
  - Varied cluster sizes
  - Elastically scalable to grow and deploy
- Power, Space, Cooling efficiency



Vela System architecture

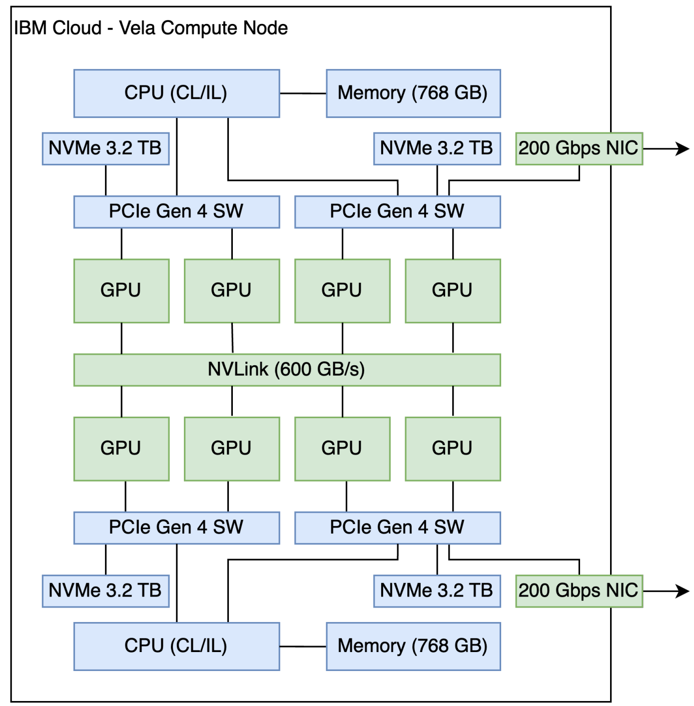
- Compute exposed as VMs
- Dual ported network cards, ethernet only
- Each port connects to a different top of the rack switch
- Multiple spine switches
- Packets can travel on multiple paths from source to destination
- Tiered storage and file system



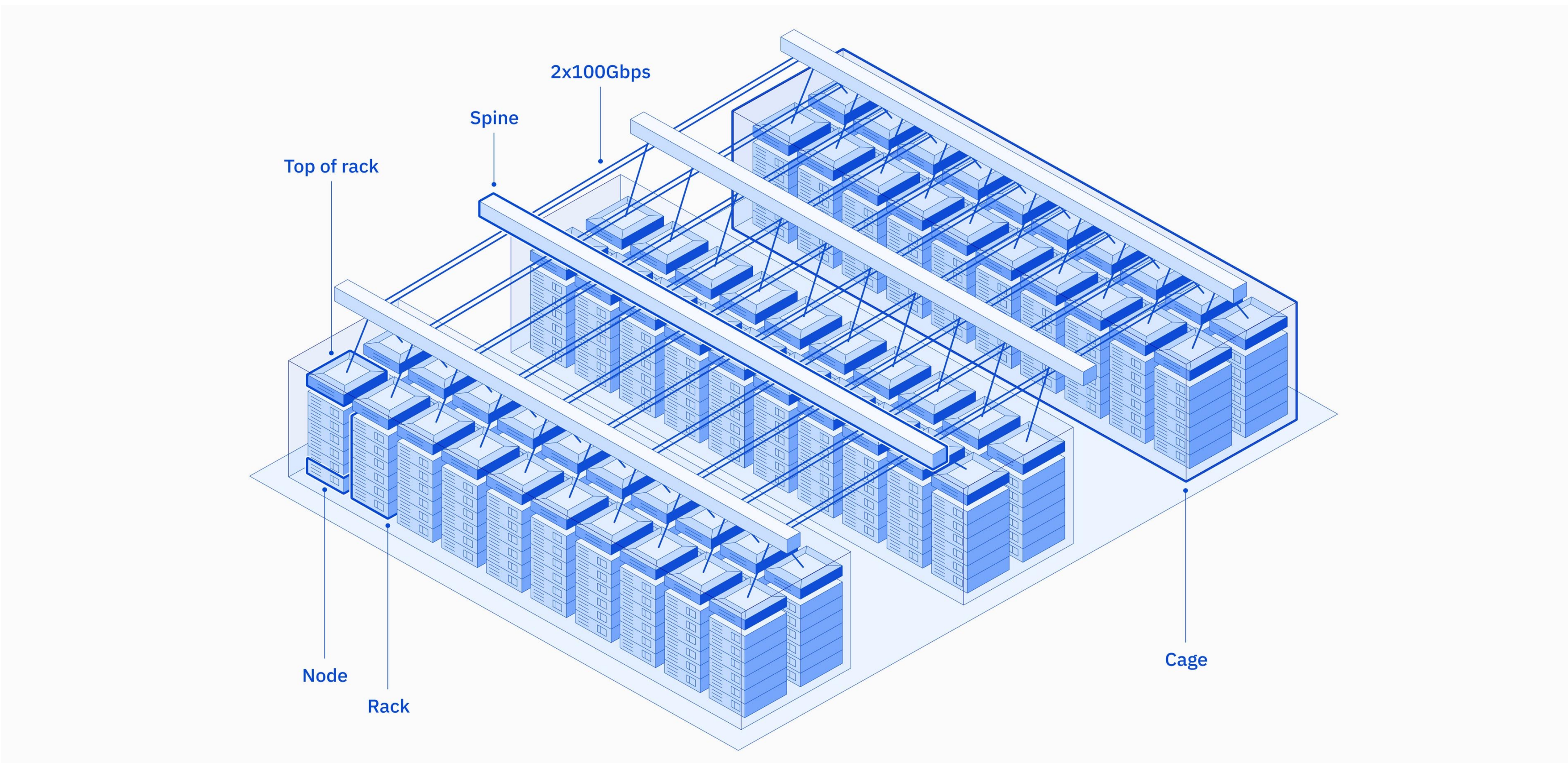
Vela and Vela2 system architecture

IBM Vela Compute Nodes

- Each physical node has:
  - 2x 24 core processors
  - 8x A100 80 GB GPUs
  - 4x 3.2 TB NVMe OPAL drives
  - 1.5 TB DDR4 Memory
  - 2x 2x100 Gbps CX-6 NICs
- Customers purchase a gx2-80x1280x8a100 VSI profile
  - 80 vCPUs, 1280 GB memory
  - 4x 3.2 TB NVMe Instance Storage drives
  - 4x 100 Gbps SR-IOV vNICs
- These are sold as ‘whole system VSIs’. The entire system is sold to a single tenant.
- Consumption as a VSI unlocks access to the broader IBM Cloud ecosystem



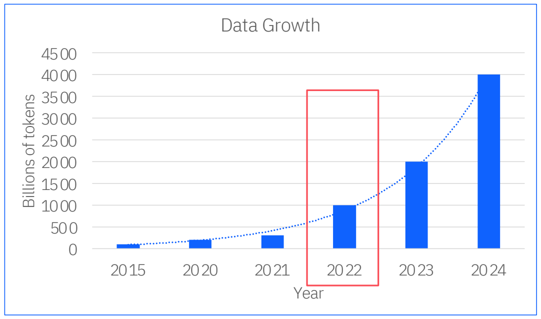
What is the Vela cloud native AI Supercomputer? – Part one (2021-2022)

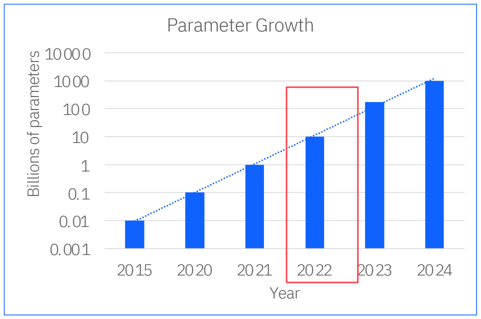


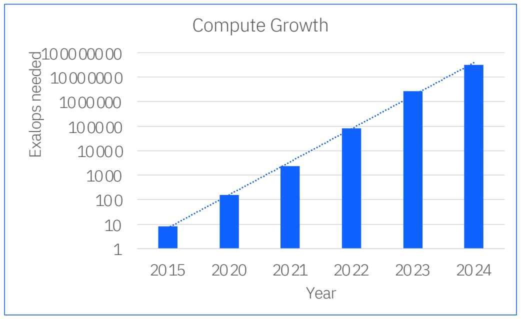
IBM Cloud Vela -- 2021- 2022

→ Bare-metal performance in the cloud
→ < 5% virtualization overhead
→ 90%+ GPU efficiency

Foundation Models operate on large data to train large number of parameters requiring large amount of compute





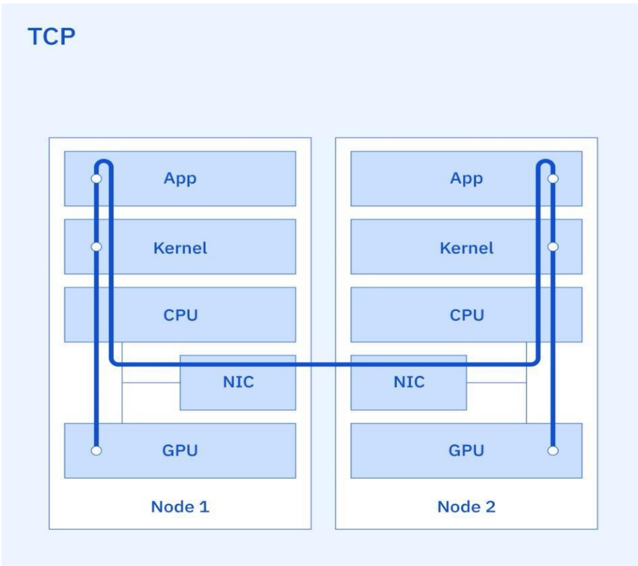


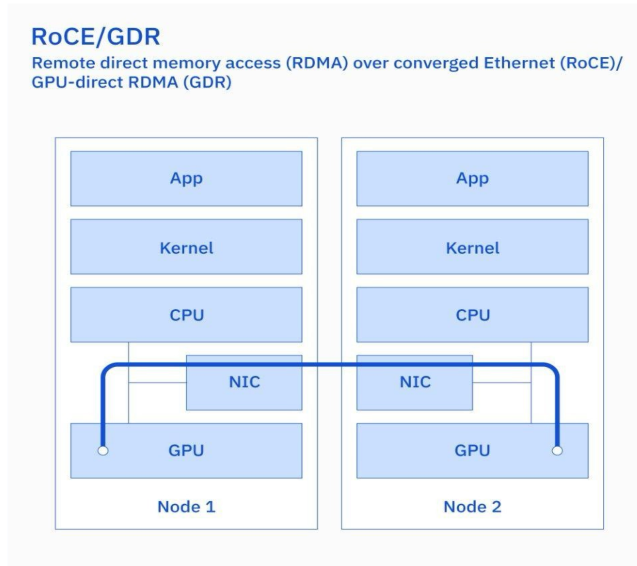
Need at least 2x more compute and at least 4x higher performance in 6 months

Vela – Part two (2023)

- Improve performance of workloads by at least 2x on vela part one
- Double the capacity
- Improve operations by 2x

Speeding up Vela by developing and deploying GPU Direct RDMA in IBM Cloud





- IBM Cloud Compute, SDN, and underlay-network optimized to support RDMA and GDR
- Improvements:
  - 2-4x network throughput
  - 6-10x network latency
- Trained Granite-20B and other models efficiently (2x): Watsonx code assistant for Z

TCP vs GDR Performance

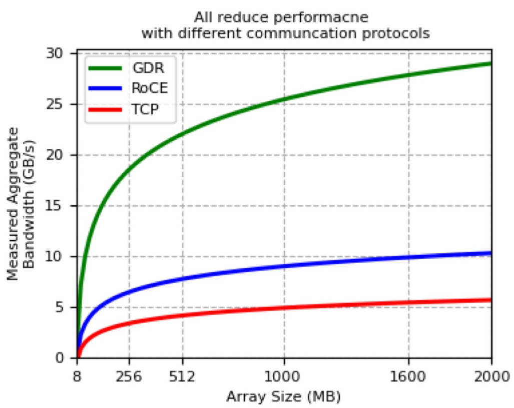


Figure 3: Performance of NCCL All Reduce collective with TCP, ROCE and GDR protocols

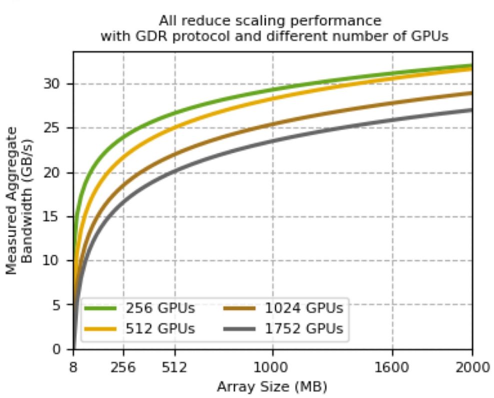
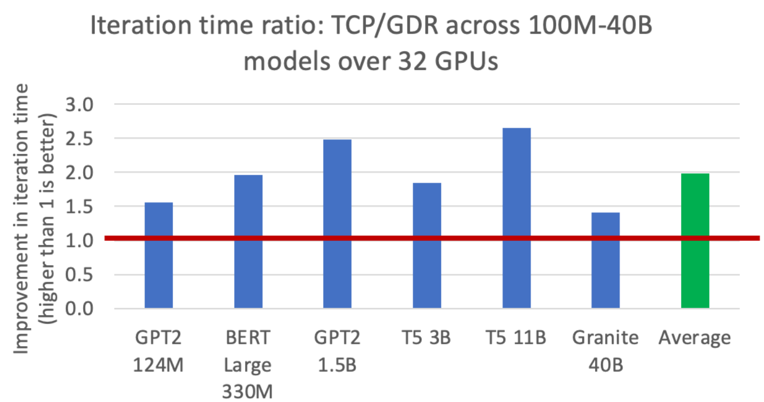
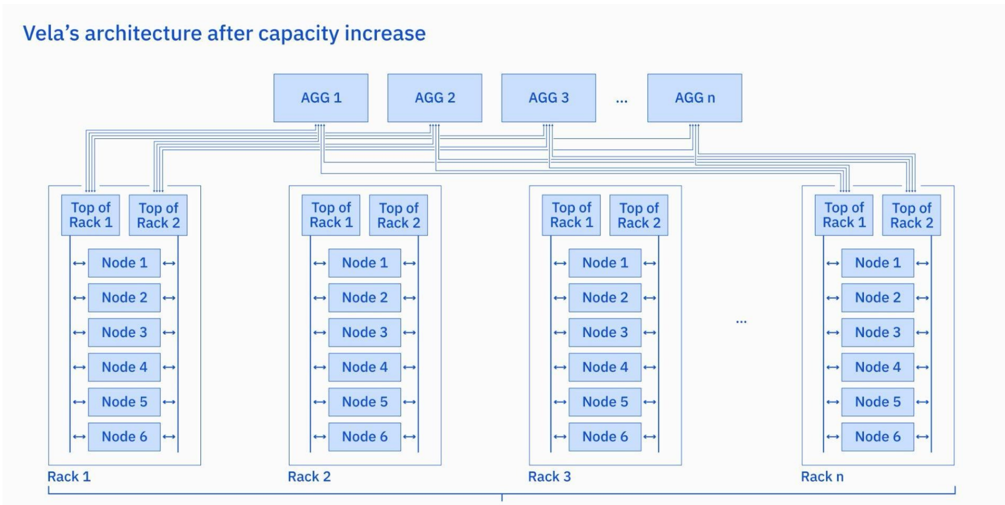


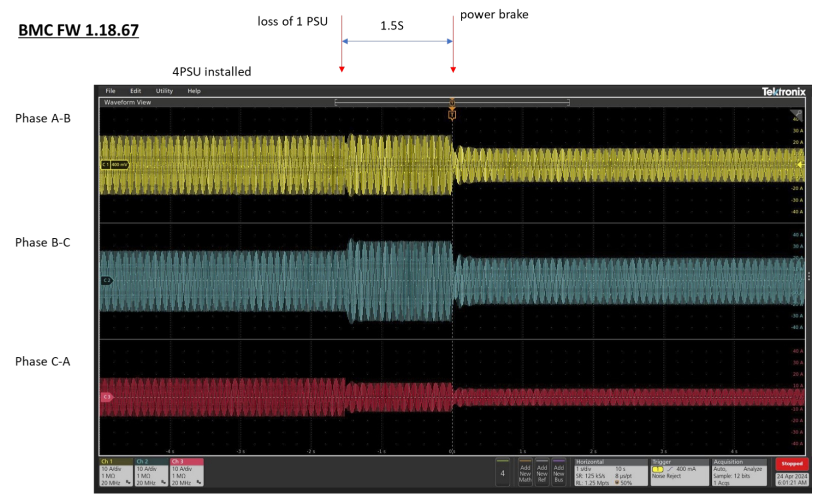
Figure 4: Performance of NCCL All Reduce collective with different number of GPUs



- At 32 GPUs, GDR is 1.4x to 2.6x better than TCP
- This will hold for larger number of GPUs and scales better for larger models

Power, Space, and cooling efficiency





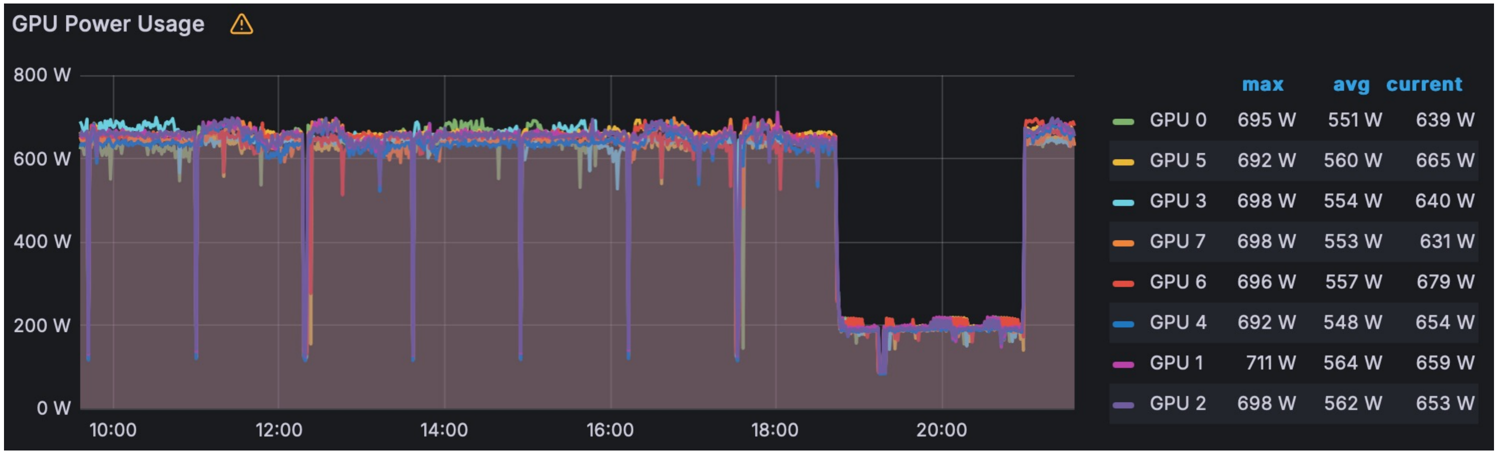
- Power break kicks in 1.5 sec after a PSU failure
- Typically, PDUs have a 5 second tolerance

- Doubled servers per rack: within the same space, power and network configuration
- "Overcommit" power – deployed power capping in case of power equipment failures
  - In case of a PSU/PDU failure, throttle down
  - Instead of always have excess power to handle failures – which requires 2x more power where only half is used at anytime.

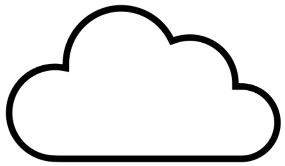
Vela cluster resilience

Jan 14: We found an issue with a power breaker in a data center, to resolve, they needed to de-energize eight PDUs. This work resulted in no access loss, as the power capping throttled the systems.

Power readings during power maintenance
Workload continue to execute albeit with lower performance

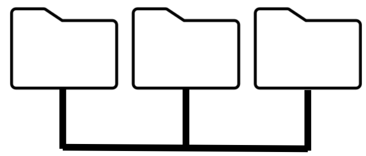


Point of view on storage for AI



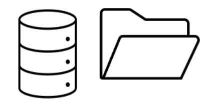
Object Storage

- Source for most data
- Key drivers: Cost, Scale, Performance



Distributed File Storage

- Cache for ”hot” data
- Input data, checkpoints, intermediate data
- Automated sync to COS



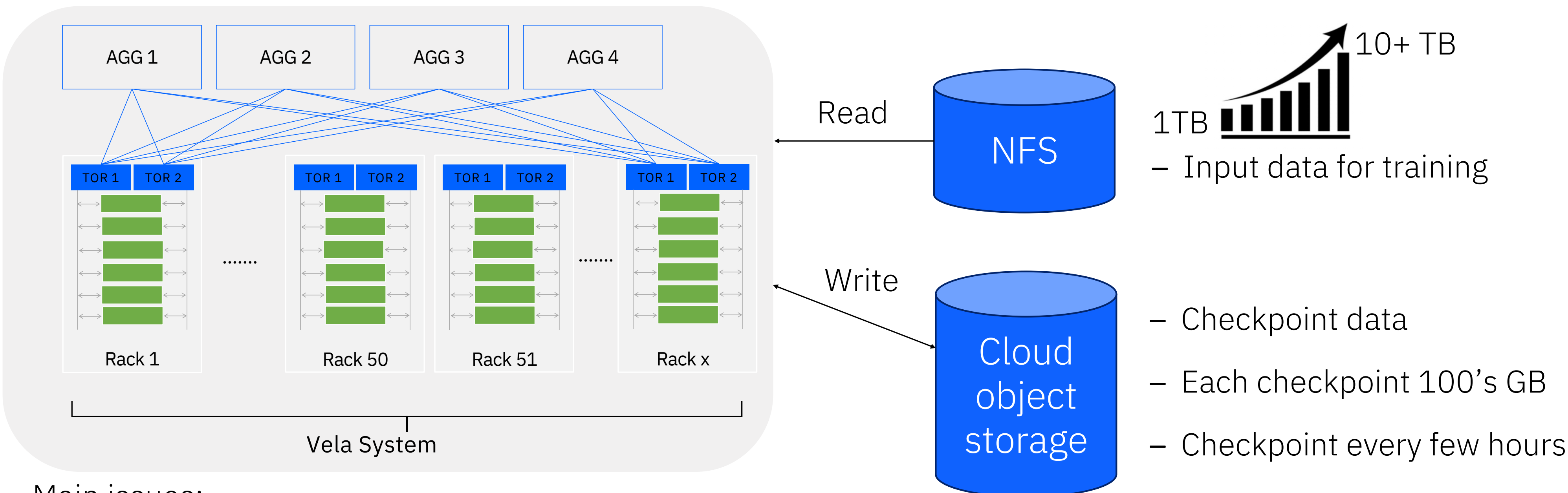
Local Storage

- Cache for frequently changing data (like KV Cache)
- Cache for model state to support multi-tenancy
- GPU Direct Storage
- Automated sync to COS

Research challenge:

- Make data migration between these layers transparent to users
- While avoiding interference with jobs (network and compute performance)
- Make DFS tier elastic

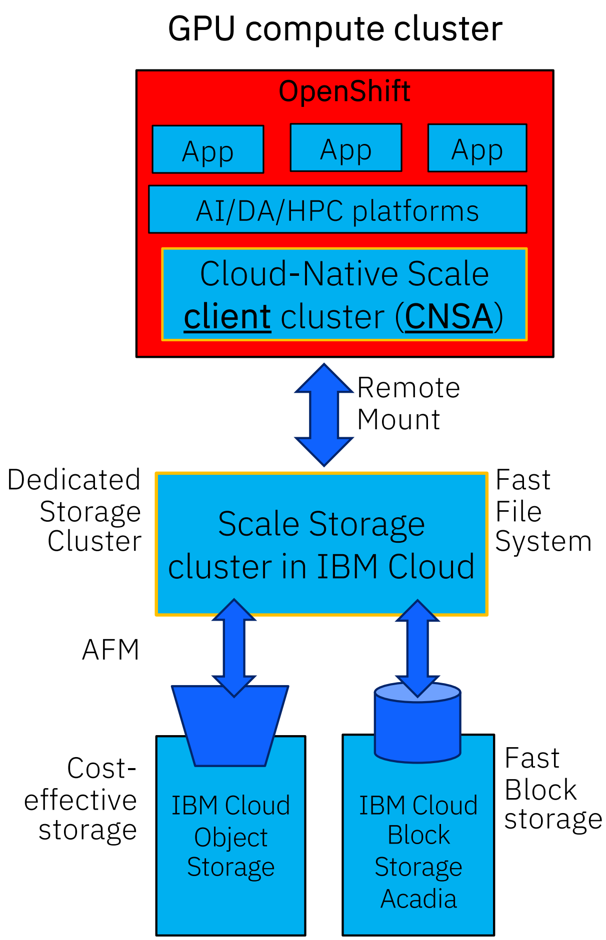
Cloud Vela BEFORE introducing IBM Storage Scale



Main issues:

- **Writes:** Checkpoint times are growing linearly with the size of the model. E.g., 50B takes ~3-5 minutes
- **Reads:** Input reading from NFS is becoming a bottleneck in training time
- **POSIX File System**: Shared namespace with strong consistency semantics for AI applications

Cloud Vela AFTER introducing IBM Storage Scale

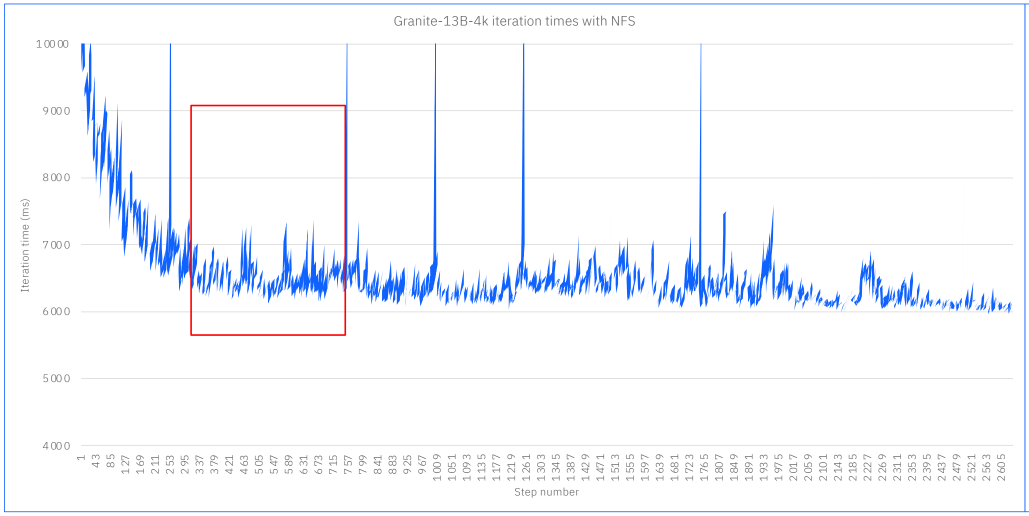


- Rely on **IBM Cloud** infrastructure [Cloud's offering of Scale (catalog docs)]
- **Dedicated storage cluster** on IBM Cloud instances
- Adapted to use **Acadia (Ceph) block storage** in IBM Cloud
  - 140 x 1TB x attached to 70 VMs
- **Cloud-Native Scale Access** (CNSA) on GPU compute cluster
  - >200 nodes
- GPU workers (i.e., node pool) can **dynamically expand and shrink**
  - Deployed a dedicated pool of CPU-only nodes for Quorum
- **Object storage** is the large shared cost-effective data repository
  - Two-tier architecture: AFM transparently moves data object storage ⟷ FS
- On-demand **shared POSIX file system** provisioning
- One volume for checkpointing; one volume for training data
  - Can accumulate ~10 days of checkpointing
  - Fit complete training dataset

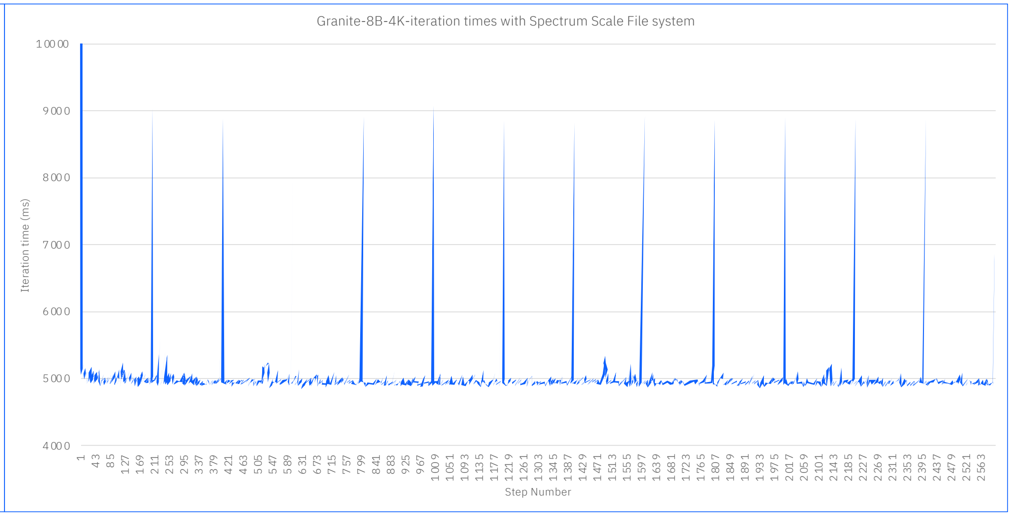
Largest CNSA deployment to date!

Promising Results

- ✓ Checkpoint times are **3x shorter** compared to COS. E.g., 50B takes ~50 seconds vs 3 minute
- ✓ Input read times are less variable



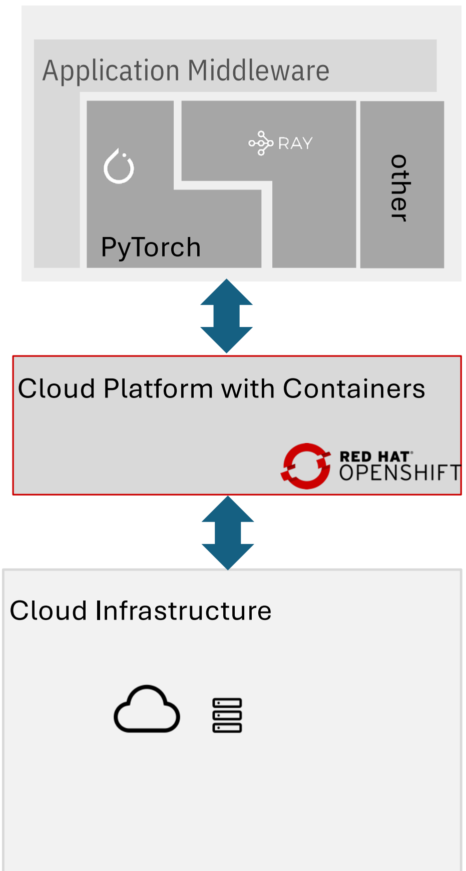
Step times: 9 – 6 seconds, up to 50% variation



Step times: 4.8 to 5.2 seconds, less than 10% variation

- ✓ Job restart no longer takes a long time to achieve steady state
- ✓ Step time variation reduces from 50% to under 10%

Co-design opportunities in three layers



Flexible middleware for training, fine tuning and inferencing

How to build optimized runtime libraries that can take advantage of the high performance infrastructure (e.g., FSDP, DDP)?

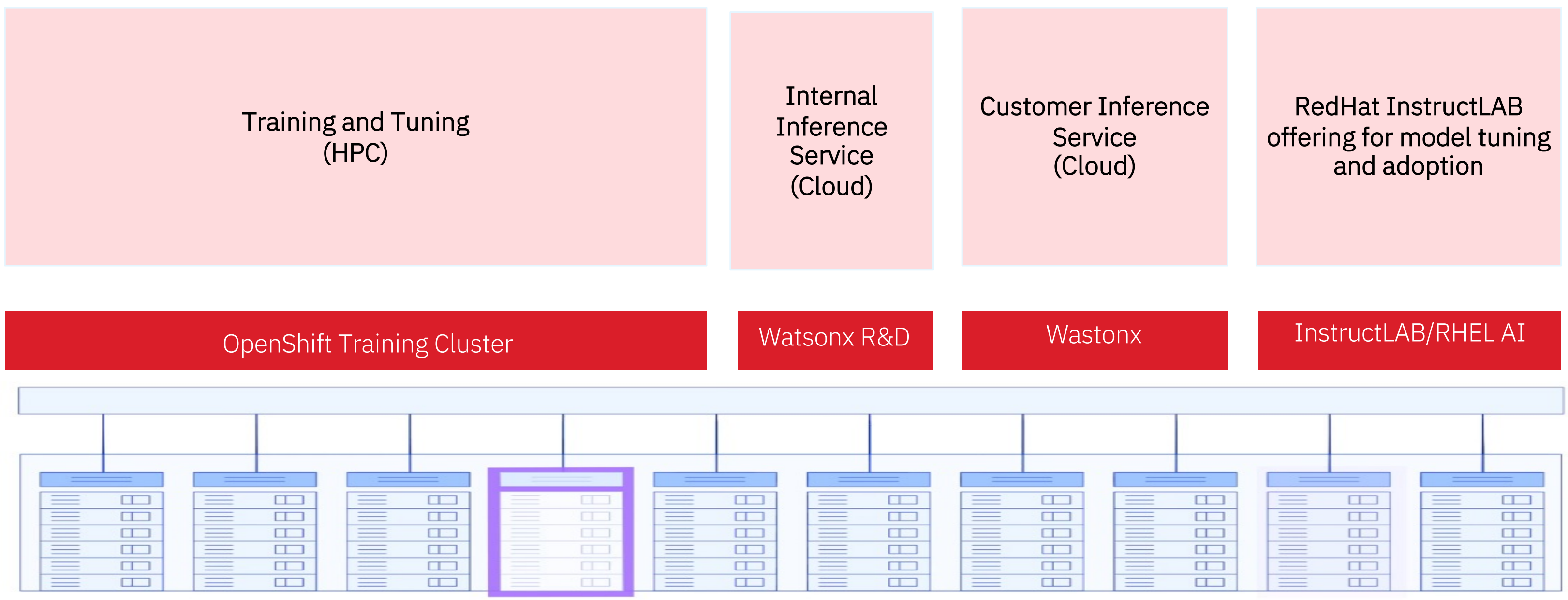
Cloud Platform

- How to enable high performance compute and network capabilities into containers?
- How to support AI workload life cycle?
- How to automate health checks?

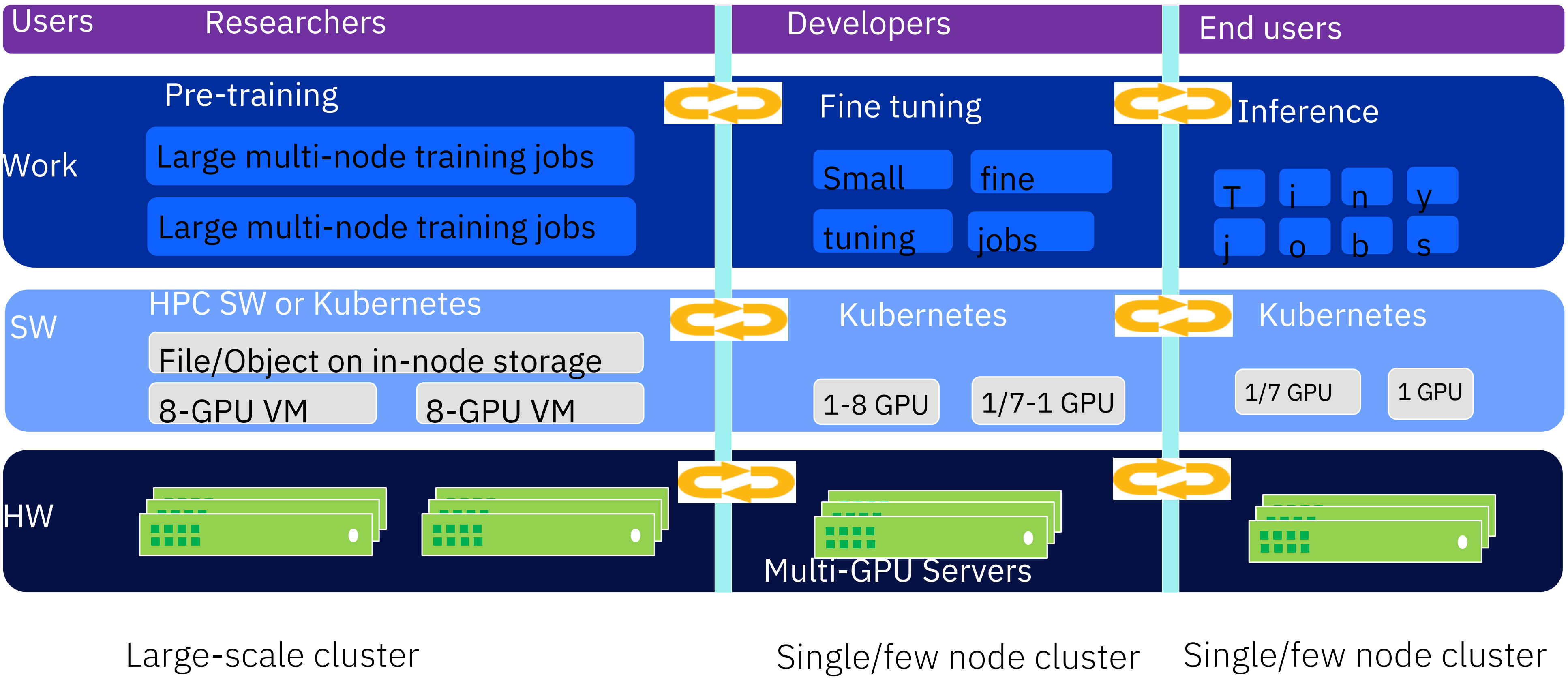
Scale-out infrastructure (e.g. on prem or cloud)

- How to design an elastically scalable system?
- How to build high performance and resilient RDMA network
- How to maximize efficiency of power, space, and cooling?

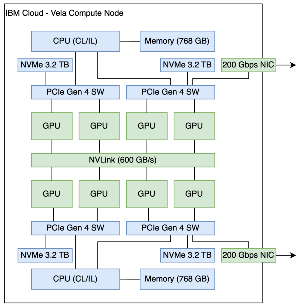
Vela workloads: Supporting full AI life cycle



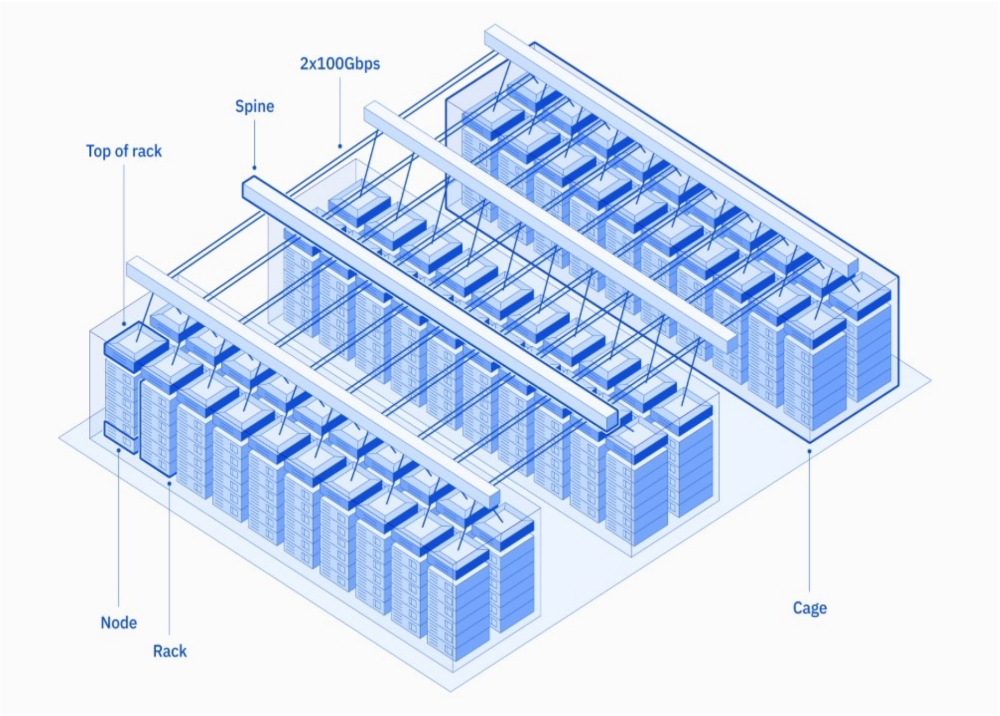
Requirements for pre-training, transfer learning and Inference



Improved diagnostics and operations

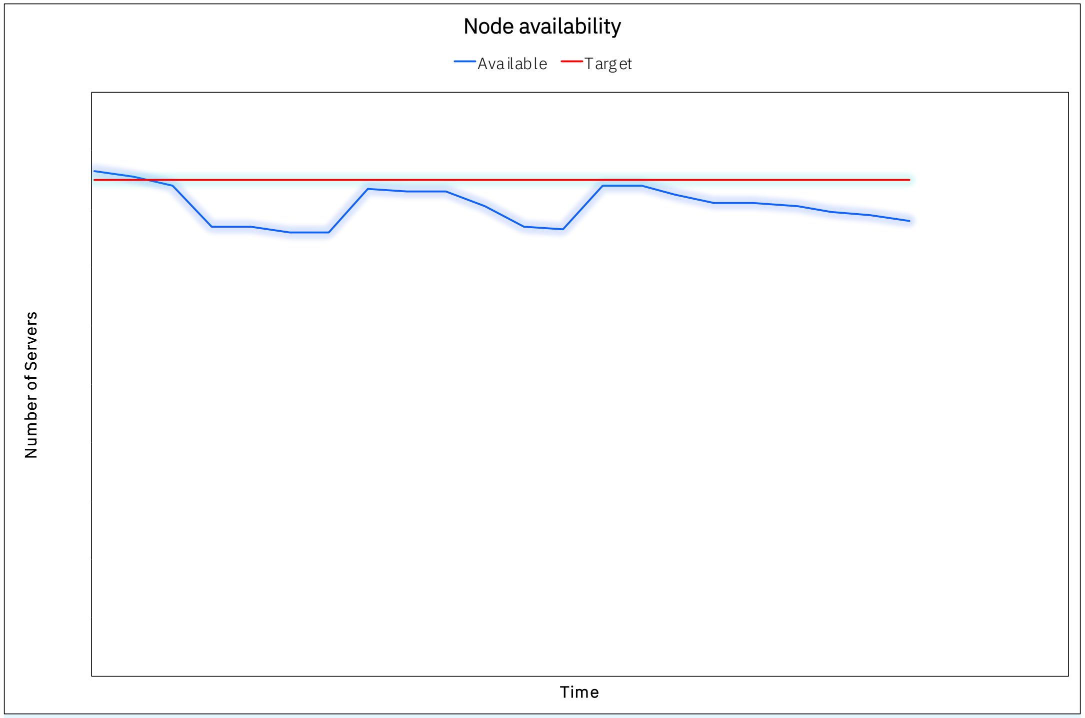


- AI servers have higher failure rates compared to traditional cloud systems
- Issues:
  - Component failure
  - Performance degradations
- Mitigations:
  - Alerts
  - Tools for rapidly detecting and isolating problem components
- This automation reduced the time to find and understand these issues by more than half.



Large-scale distributed infra needs tools…

- GPU node failures: 1 every 4 days
  - Top 3 issues: GPU failure or performance issue, network performance issues between GPUs , backend network and service issues
- This is not unique
  - META reports ~2 nodes lost per day while training OPT on Azure
    - 90 re-starts over the course of the training run; actual computation time ~ 30 days, total time to train > 2 months
    - https://github.com/facebookresearch/metaseq/blob/main/projects/OPT/chronicles/README.md



Key lessons:

- Continuous monitoring and isolation of problem nodes necessary to keep high utilization
- Automation in software that navigate around node failures can help large-scale AI training jobs complete faster ("auto-pilot")

Need tools to detect soft and hard failures

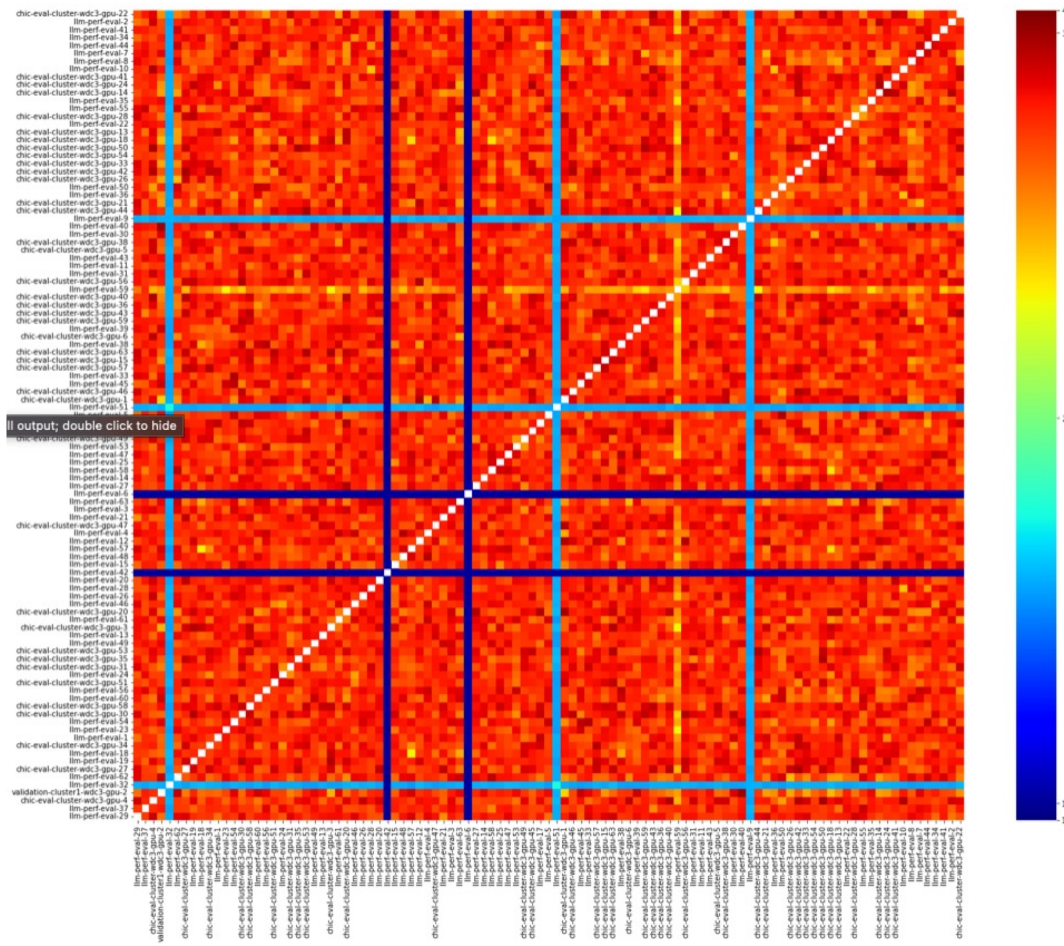
Example soft failures:

- PCI-E Link degradation
- Memory bandwidth slow down to due to corruptions
- One or more GPUs drop from the system

Examples of Hard failures:

- Node failure
- GPU failure

Heatmap of network bandwidth between nodes



GPU systems result in errors too… how do we deal with them?

Failure type	Root cause	Mitigations developed
Hardware failures (host crash)	• GPU HGX board failures • Memory DIMM failure • NVLink/switch failure	• Slack alert on host crash • Automatic VM restart • Automatic job restart
Subtle hardware failures (no host crash)	• Failure of GPUs • GPU HBM Memory corruption • PCI-E link failure • Port failures • Power feed failure	• Slack alert on port, GPU, other critical component failures • Alert based on host BMC logs • Enhanced metrics collection via Autopilot
Software failures	• PCI-E Link degradation • Cuda memory allocation error • HBM Memory row remaps	• Checks of PCI-E links • Alerts based on application logs • Periodic VM reboots

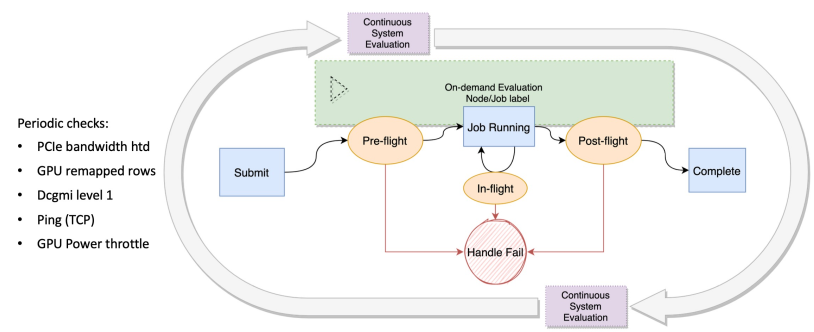
Table 1: Infrastructure failure types, root causes, and mitigations.

https://arxiv.org/pdf/2407.05467

Health checks

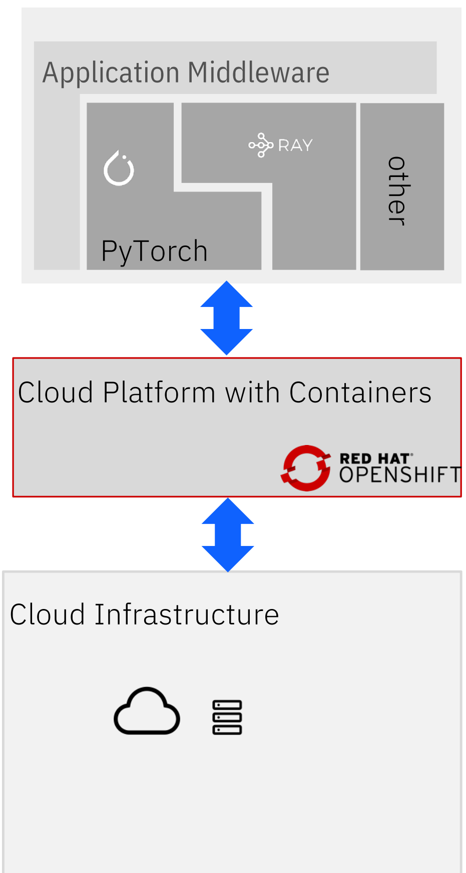
- *Continuous checks (invasive & non-invasive)*
- *Pre-flight checks*
- *In-flight checks*
- *On-demand checks (e.g., iperf)*

Integration with scheduler (Kueue) to avoid bad resources



Autopilot → Collaborations welcome

Co-design opportunities in three layers



Flexible middleware for training, fine tuning and inferencing

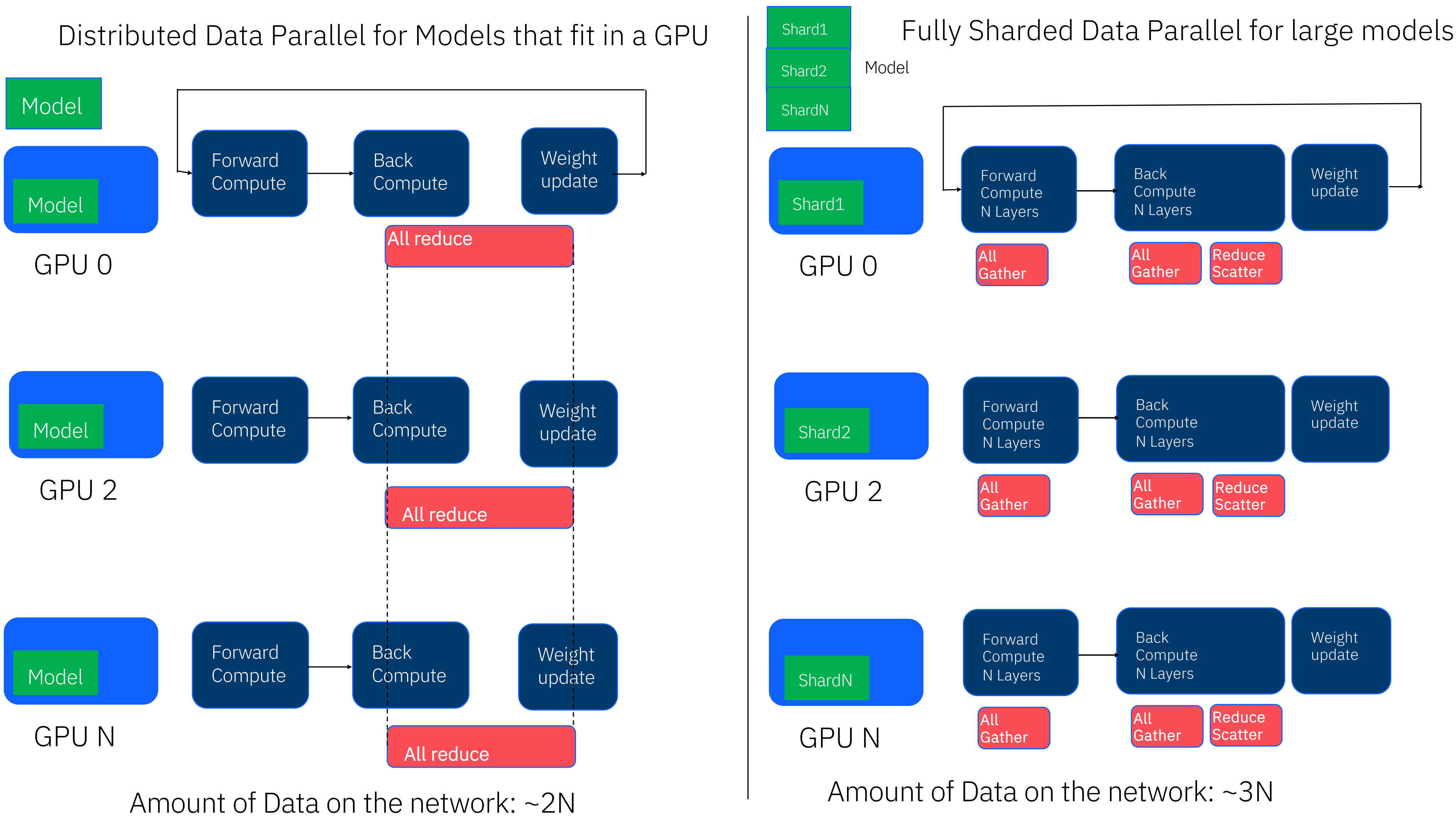
How to build optimized runtime libraries that can take advantage of the high performance infrastructure (e.g., FSDP, DDP)?

Cloud Platform

- How to enable high performance compute and network capabilities into containers?
- How to support AI workload life cycle?
- How to automate health checks?

Scale-out infrastructure (e.g. on prem or cloud)

- How to design an elastically scalable system?
- How to build high performance and resilient RDMA network
- How to maximize efficiency of power, space, and cooling?


Distributed Data Parallel for Models that fit in a GPU
Model
Model
Forward Compute
Back Compute
Weight update
All reduce
GPU 0
Model
Forward Compute
Back Compute
Weight update
All reduce
GPU 2
Model
Forward Compute
Back Compute
Weight update
All reduce
GPU N
Amount of Data on the network: ~2N
Fully Sharded Data Parallel for large models
Shard1
Shard2
ShardN
Model
Shard1
Forward Compute N Layers
Back Compute N Layers
Weight update
All Gather
All Gather
Reduce Scatter
GPU 0
Shard2
Forward Compute N Layers
Back Compute N Layers
Weight update
All Gather
All Gather
Reduce Scatter
GPU 2
ShardN
Forward Compute N Layers
Back Compute N Layers
Weight update
All Gather
All Gather
Reduce Scatter
GPU N
Amount of Data on the network: ~3N

Agenda

1. What's new in AI
2. Vela Systems co-design
3. Summary

Insights

1. AI workloads change rapidly
2. Design for enormous success
3. Deliver usable solutions and iterate on them
4. Automation is the key for operational success
5. Be a part of this journey

Acknowledgements

IBM Research

- Alim Alim
- Ali Sydney
- Apoorve Mohan
- Apo Kai
- Bengi Karacali
- Minghung Chen
- I-Hsin Chung
- Bob Walkup
- Laurent Schares
- Eric Luo

IBM Research

- Sophia Wen
- Liran Schour
- Pavlos Maniotis
- Shweta Salaria
- Sunyanan C.
- Takeshi Yoshimura
- Tatsuhiro Chiba
- Marc Dombrowa
- Bernard Metzler
- Talia Gershon
- ...

IBM Research

- Kevin O'connor
- Christopher Laibinis
- Carlos Fonseca
- Vasily Tarasov
- Swami Sundararaman
- Marc Eshel
- Paul Muench
- Scott Guthridge
- Frank Schmuck
- Thanh Pham

IBM Cloud

- Amilcar Arvelo
- Amitabha Biswas
- Bobby Clark
- Brent Tang
- Drew Thorstensen
- Eran Gampel
- Eric McKeever
- Felipe Telles
- Jason Van Patten
- ...

IBM Cloud

- Joel Belog
- John Rawlins
- Khuong Nguyen
- Preeti Kulkarni
- Rachappa Goni
- Richard Welp
- Robert Perry
- Saurabh Kumar Gupta
- Simon Mikulcik
- ...

...Many more that work on this topic everyday

References

1. Vela: A Virtualized LLM Training System With GPU Direct RoCE, ASPLOS 2025
2. The Infrastructure powering IBM's Gen AI model development: https://arxiv.org/pdf/2407.05467
3. Why we built an AI supercomputer in the cloud https://research.ibm.com/blog/AI-supercomputer-Vela-GPU-cluster
4. Supercharging IBM's cloud-native AI supercomputer https://research.ibm.com/blog/vela-ai-supercomputer-updates
5. ACM Middleware 2022: Keynote Talk - Hardware-Middleware System co-design for foundation models https://www.youtube.com/watch?v=mC5ZwMpQblQ
6. Introducing Vela: IBM's First AI-Optimized Cloud-Native Supercomputer in the Cloud https://www.nvidia.com/en-us/on-demand/session/gtcspring23-s51114/
7. How to Deploy a High-performance Distributed AI Training Cluster with NVIDIA GPUs and KVM https://www.nvidia.com/en-us/on-demand/session/gtcspring22-s42633/
8. Autopilot: https://github.com/IBM/autopilot
9. IBM Granite models: https://www.ibm.com/products/watsonx-ai/foundation-models
10. Granite Code Models: A Family of Open Foundation Models for Code Intelligence: https://arxiv.org/abs/2405.04324 Granite models: https://research.ibm.com/blog/granite-code-models-open-source