
KubeCon
CloudNativeCon
Europe 2023
TiKV

How to Leverage Volcano to Improve the Resource Utilization of AI Pharmaceuticals, Autonomous Driving, and Smart Buildings

William (LeiBo) Wang, Huawei Cloud
Email: wang.platform@gmail.com

Contents

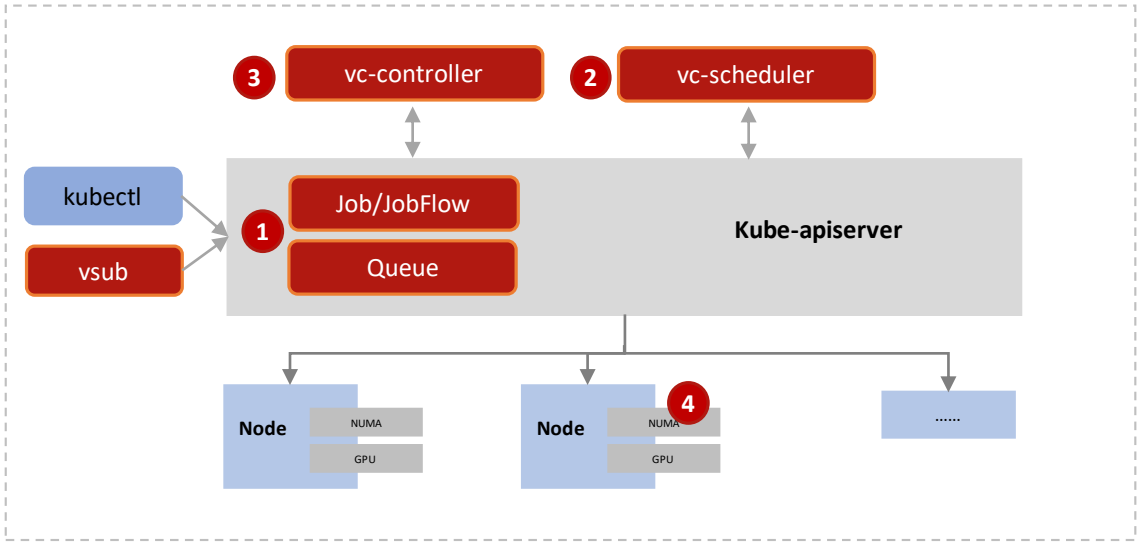
- Project Introduction
- Architecture
- New Challenges and Features
- Use Cases and community

Project Introduction

VOLCANO

- The first cloud native batch computing platform
- Open sourced in June 2019, donated to CNCF in 2020, and is currently a CNCF incubation project
- **2.9K** stars, **500+** global contributors
- **50+** enterprise production landing
- Supports **mainstream AI, big data, and HPC frameworks**
- Provides **rich advanced scheduling strategies** and **unified job management**
- Optimized for **large-scale resource pools** and supports heterogeneous computing

Volcano Architecture



1. **Volcano API**

   Core concepts such as Job, JobFlow, Queue, and PodGroup are provided to define batch workload.

2. **Volcano scheduler**

   Rich scheduling policies: fair-share scheduling, topology-based scheduling, SLA-based scheduling, preemption, backfill, elastic scheduling.

   Plug-in algorithm integration framework: allows users to customize algorithm policies by combining different actions and plugins to satisfy different scenarios.

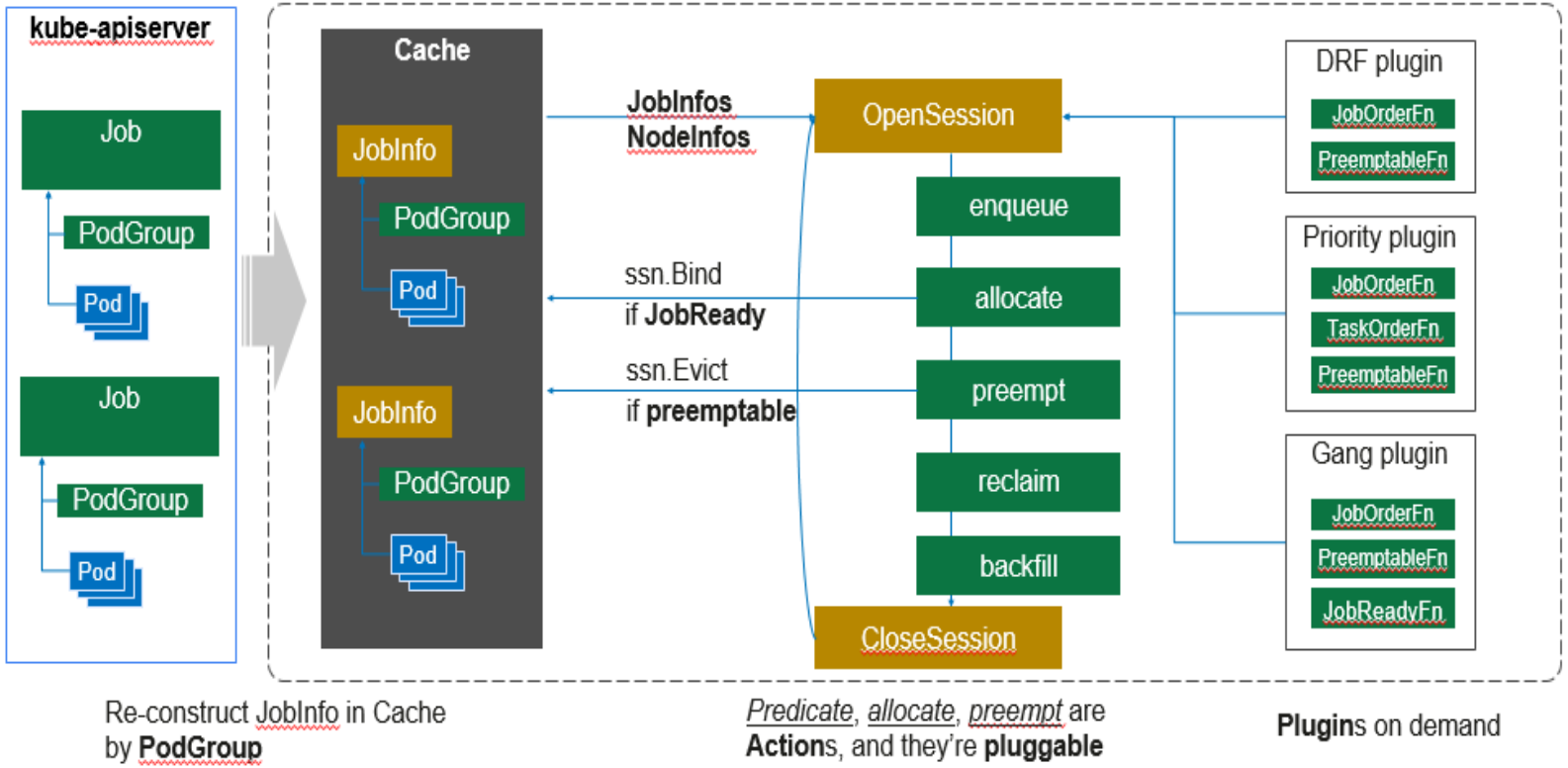
3. **Volcano controller**

   Core capabilities such as podgroup and job lifecycle management, job queue, queue resource reservation, queue capacity, and dynamic sharing of multi-tenant resources.

4. **Node plugin**

   Cooperate with scheduler to provide large-scale colocation support, topology-based scheduling optimization, and heterogeneous hardware support, including x86, Arm, GPU, Ascend, Kunlun, etc.

How Volcano Scheduler Works



- Volcano support AI and Bigdata workload through job based scheduling and plugin mechanism.
- Cache is designed to reduce interaction with kube-apiserver for better performance and construct relationship between Job and task. And it is extendable to interact with other resource manager besides Kubernetes.
- In each scheduling cycle, scheduler schedules jobs based on snapshot of cache for consistency.
- Action is first-level plugin and multi-actions are executed in sequence.
- Plugin is second-level plugin and can be registered in OpenSession function.

Volcano Journey

Internet

- Gang-scheduling
- Task-topology
- Priority based scheduling
- Integrate with Kubeflow
- Integrate with Argo
- ……

step1

Accelerate AI on K8s

Financial service

Smart building

- Integrate with Spark-operator
- Integrate with Flink-operator
- Make Volcano as Spark on K8s batch scheduler
- Resource reservation
- Avoid overcommit
- Performance tuning for throughput

step2

Accelerate Big data

Biopharmaceutical

- Enhance Job management
- Add Job dynamic scaling
- Add Job plugins for MPI and Tensorflow, Pytorch plugin in Volcano for better user experience
- Add the JobFlow for dependency management

step3

Simplify Job management

GPT

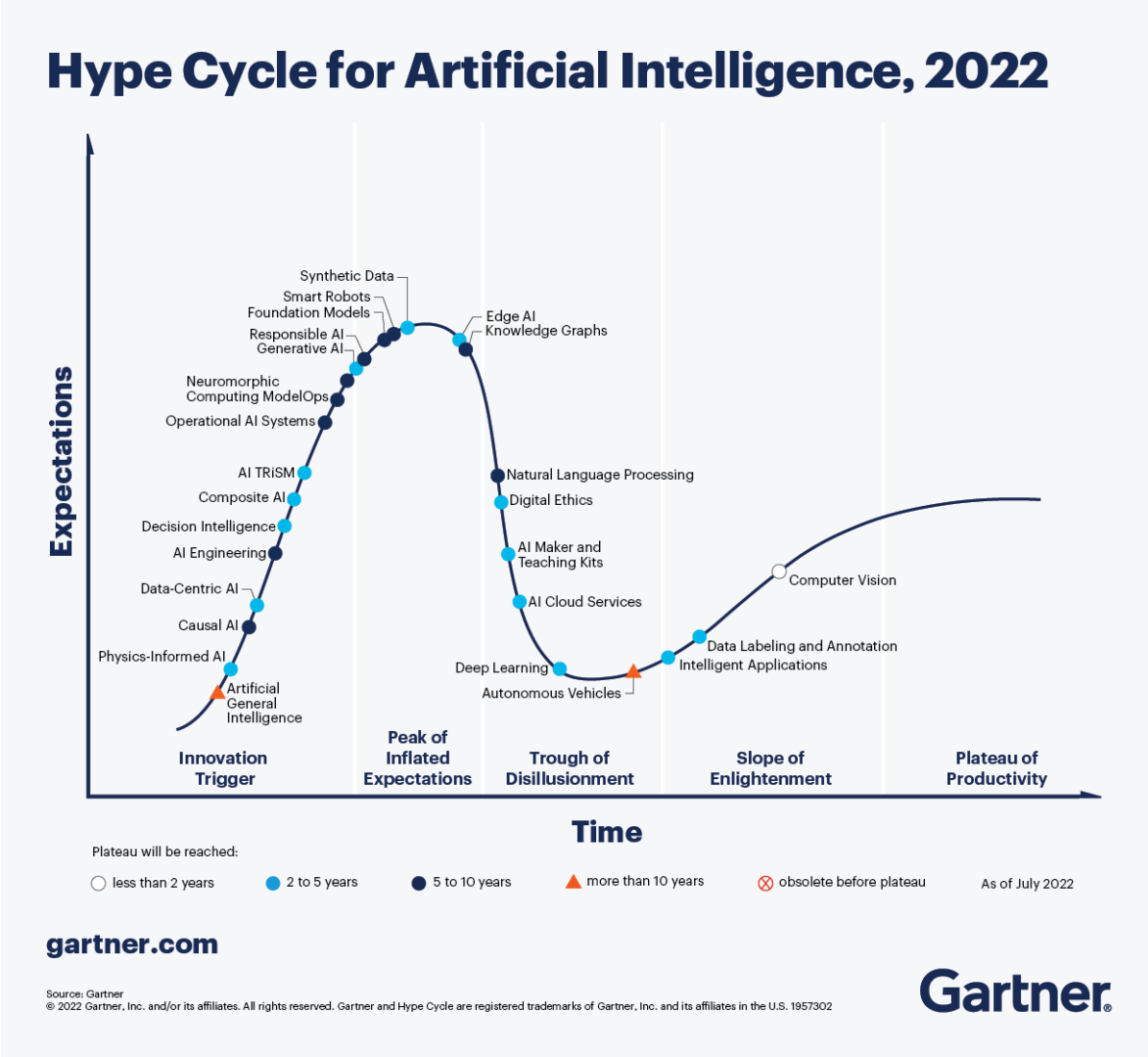
Autopilot

- GPU sharing (ongoing)
- Global scheduling on multiple clusters (ongoing)
- Cost awareness (pipeline)
- Colocation for online and offline container (ongoing)

step4

Improve resource utilization

Challenge 1: Requirement for Computing Power is Growing

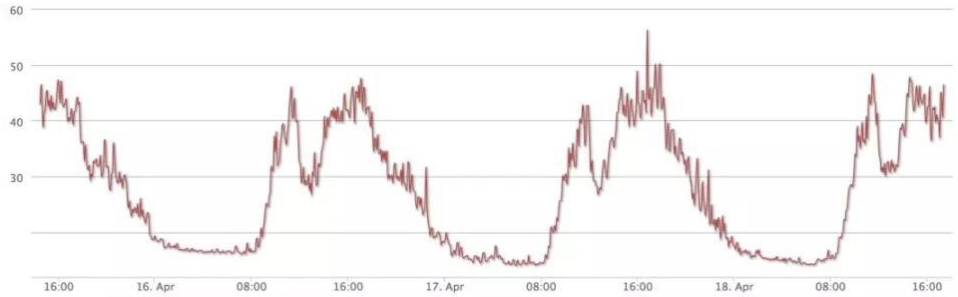


- AI technology has entered the stage of commercialization and requires more and more computing power.
- According to the analysis report released by OpenAI, since 2012, the computing power used in AI training has doubled every 3-4 months.
- Although advances in algorithm and architectures allow more learning to be done with less computing, the computing power demand remains high.
- Computing power is becoming the bottleneck of intelligent computing.

Challenge 2: Low Resource Utilization

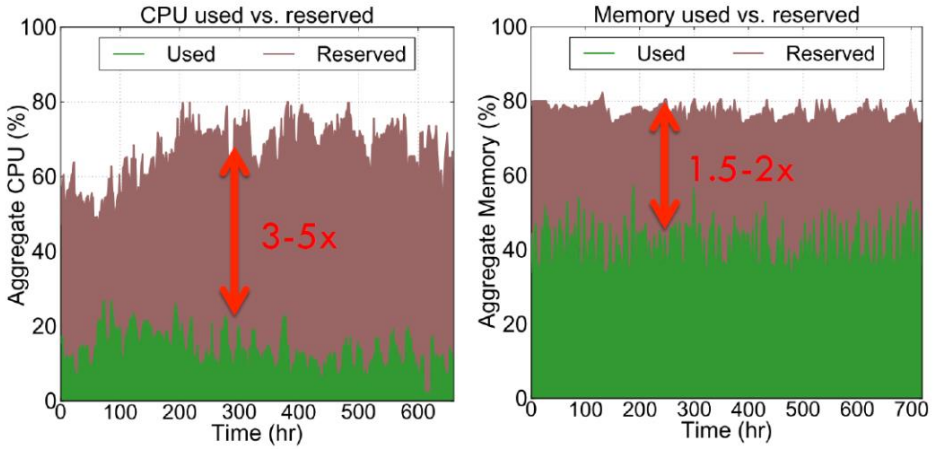
Resource requirements:

Obvious peaks and troughs



Twitter data center resource statistics:

Used resources < Reserved resources



*According to Gartner statistics, the average CPU utilization rate of enterprises is less than **15%.** There are many reasons for the low cluster utilization rate. Typical scenarios include:*

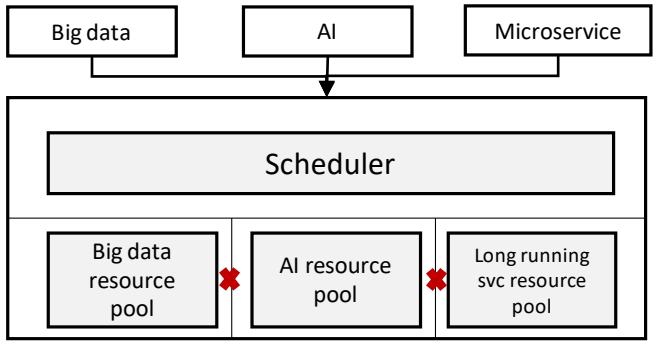
- Isolated resource pools cause severe **resource fragmentation**.
- Resources are applied for based on the peaks.
- **Unreasonable resource allocation**

Containerization cannot satisfy all scenarios, we still need to:

- **Unify resource pool** and ensure **fair scheduling** among multi-business teams.
- Ensure service **quality** while increasing application deployment **density**.
- Set some applications to **bind cores** and monopolize resources to ensure stability.
- **Prepare sufficient resources.**
- **Transparent scaling**

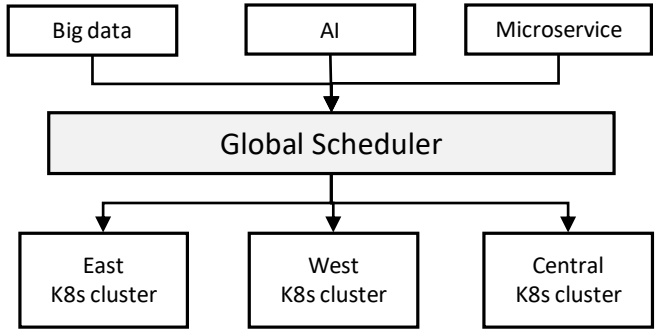
Unified Scheduling and Colocation Improve Resource Utilization

- Break up isolated resource pools for unified resource scheduling



K8s cluster

- Multi-cluster scheduling based on global resource view

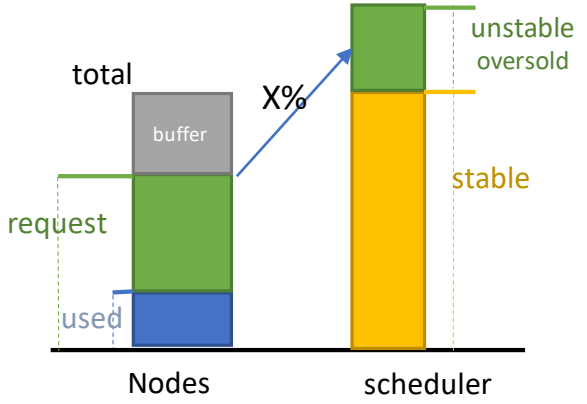


Distributed K8s clusters

- Co-locate multiple type applications
  - Colocation for MPI, Bigdata, and Function
  - Colocation for online and offline services

Stone	MPI	Weather forecast
Sand	Spark	Bigdata ETL
Water	Function	Monte Carlo

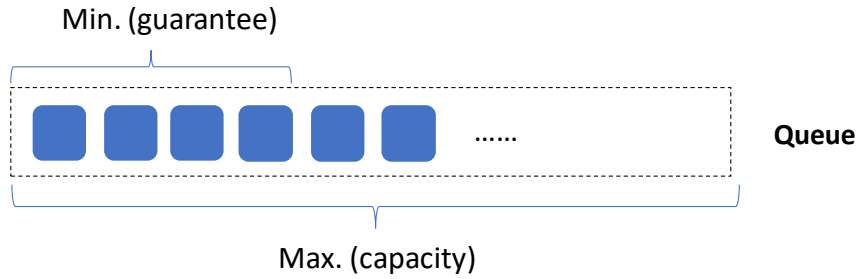
Improving resource allocation rate through offline service colocation

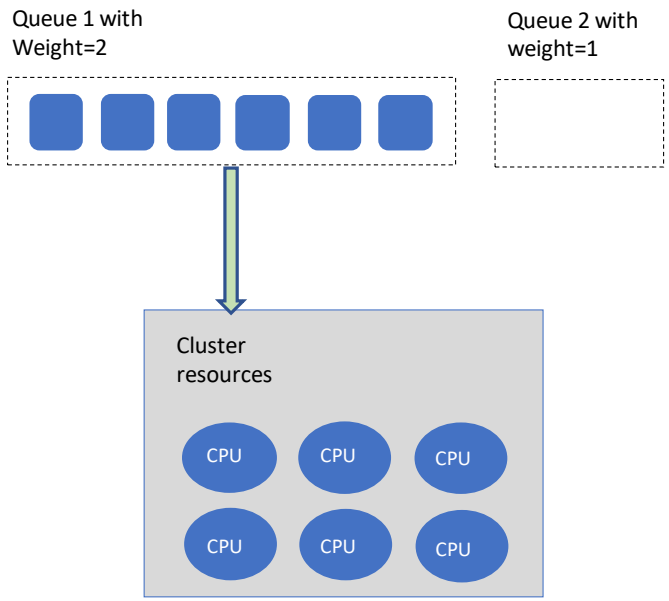


Improve utilization through online and offline colocation and resource oversold

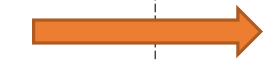
Scenario 1: Resource Share Model in Volcano

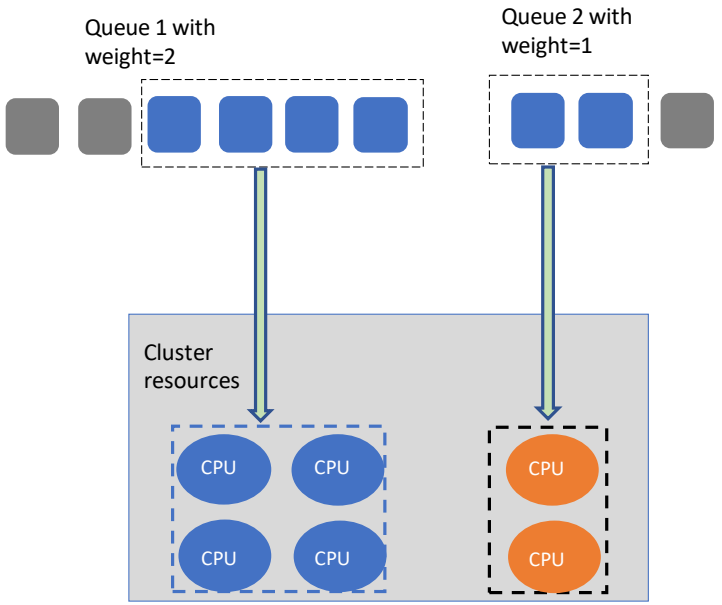
- Queue resource reservation/queue capacity
- Resource sharing for queues based on weight





Queue 2 is empty. Queue 1 can borrow resources from Queue 2.

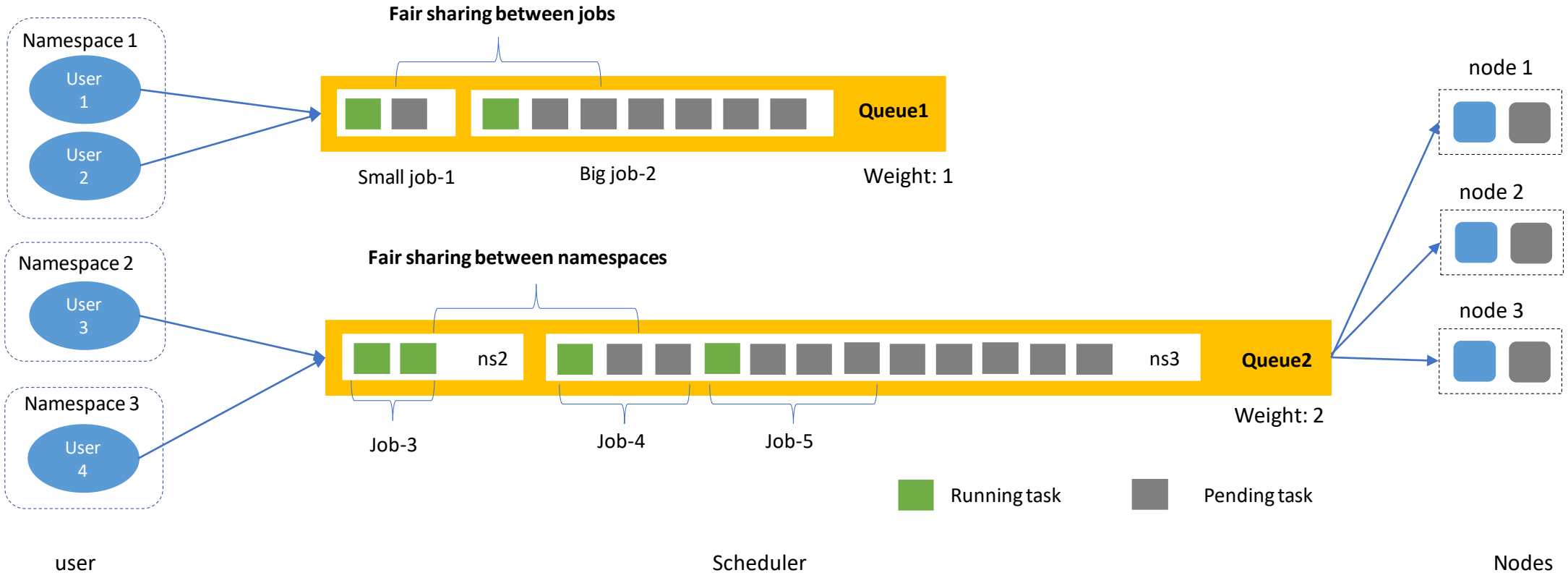




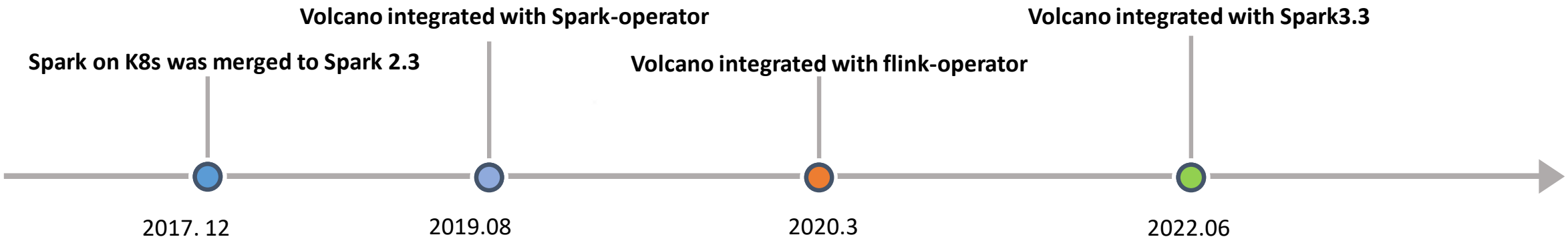
Queue 2 has workloads and will reclaim resources from Queue 1.

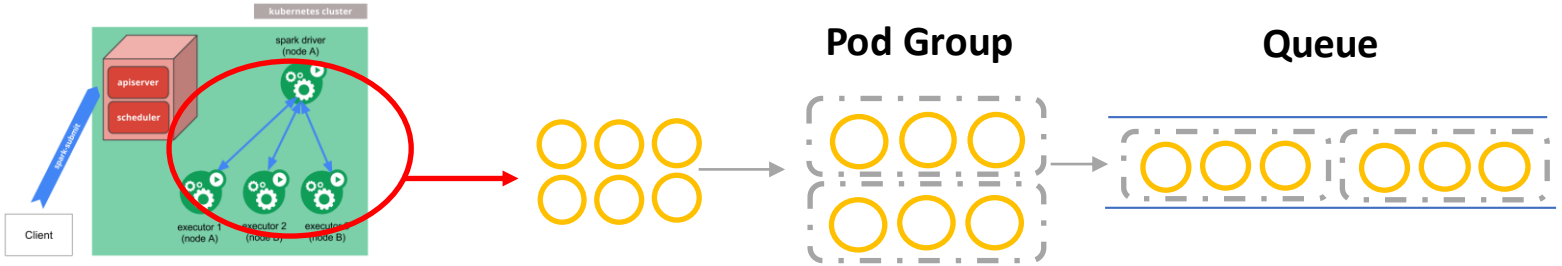
Scenario 1: Share Resource Fairly

- Sharing resource between jobs
- Sharing resource between namespaces
- Per-queue policy (FIFO, Priority, Fair share, …)



Scenario 2: The First Batch Scheduler for Spark on Kubernetes





1.	Support replicasets/job API	RESOLVED	Holden Karau
2.	Add the ability to specify a scheduler & queue	IN PROGRESS	Apache Spark
3.	Support backing off dynamic allocation increases if resources are "stuck"	OPEN	*Unassigned*
4.	Create a PodGroup with user specified minimum resources required	OPEN	*Unassigned*
5.	Support for specifiying executor/driver node selector	RESOLVED	Yikun Jiang
6.	Support the Volcano Job API	OPEN	*Unassigned*

[SPARK-36057: Support Customized Kubernetes Schedulers]

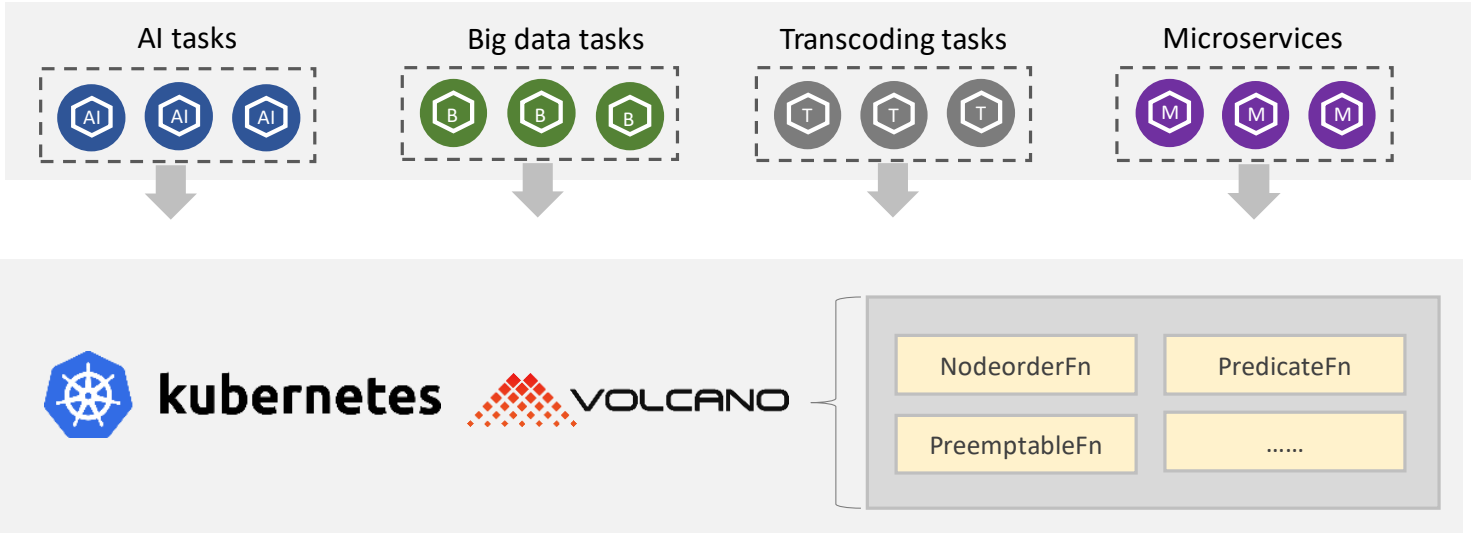
- **Benefits**
  - ✓ 1.5K+ pods/sec
  - ✓ Job-based scheduling
  - ✓ Job priority
  - ✓ Fair share
  - ✓ Queue
  - ✓ Resource reservation

Scenario 2: How to Use Volcano in Spark

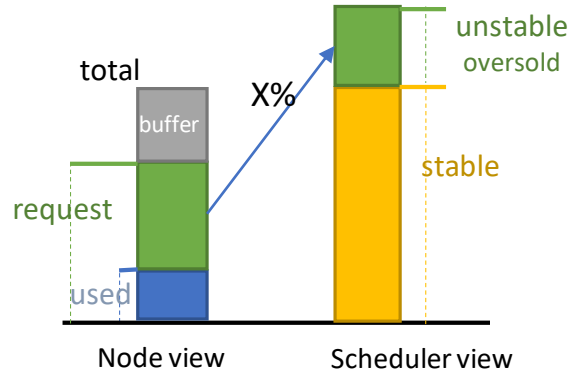

```
FEATURES="org.apache.spark.deploy.k8s.features.VolcanoFeatureStep"
~ bin/spark-submit  \
    --master k8s://https://127.0.0.1:60250  \
    --deploy-mode cluster  \
    --conf spark.executor.instances=1  \
    --conf spark.kubernetes.scheduler=volcano \
    --conf spark.kubernetes.driver.pod.featureSteps=$FEATURES \
    --conf spark.kubernetes.executor.pod.featureSteps=$FEATURES \
    --conf spark.kubernetes.scheduler.volcano.podGroupTemplateFile=/path/to/podgroup-template.yaml \
    --conf spark.kubernetes.namespace=spark \
    --conf spark.kubernetes.authenticate.driver.serviceAccountName=spark-sa  \
    --conf spark.kubernetes.container.image=spark:latest  \
    --class org.apache.spark.examples.SparkPi  \
    --name spark-pi  \
    local:///opt/spark/examples/jars/spark-examples_2.12-3.3.0-SNAPSHOT.jar
```

1. Custom scheduler
2. Custom feature step
3. Scheduler hints (podgroup template)

Scenario 3: Colocation Improves Resource Utilization

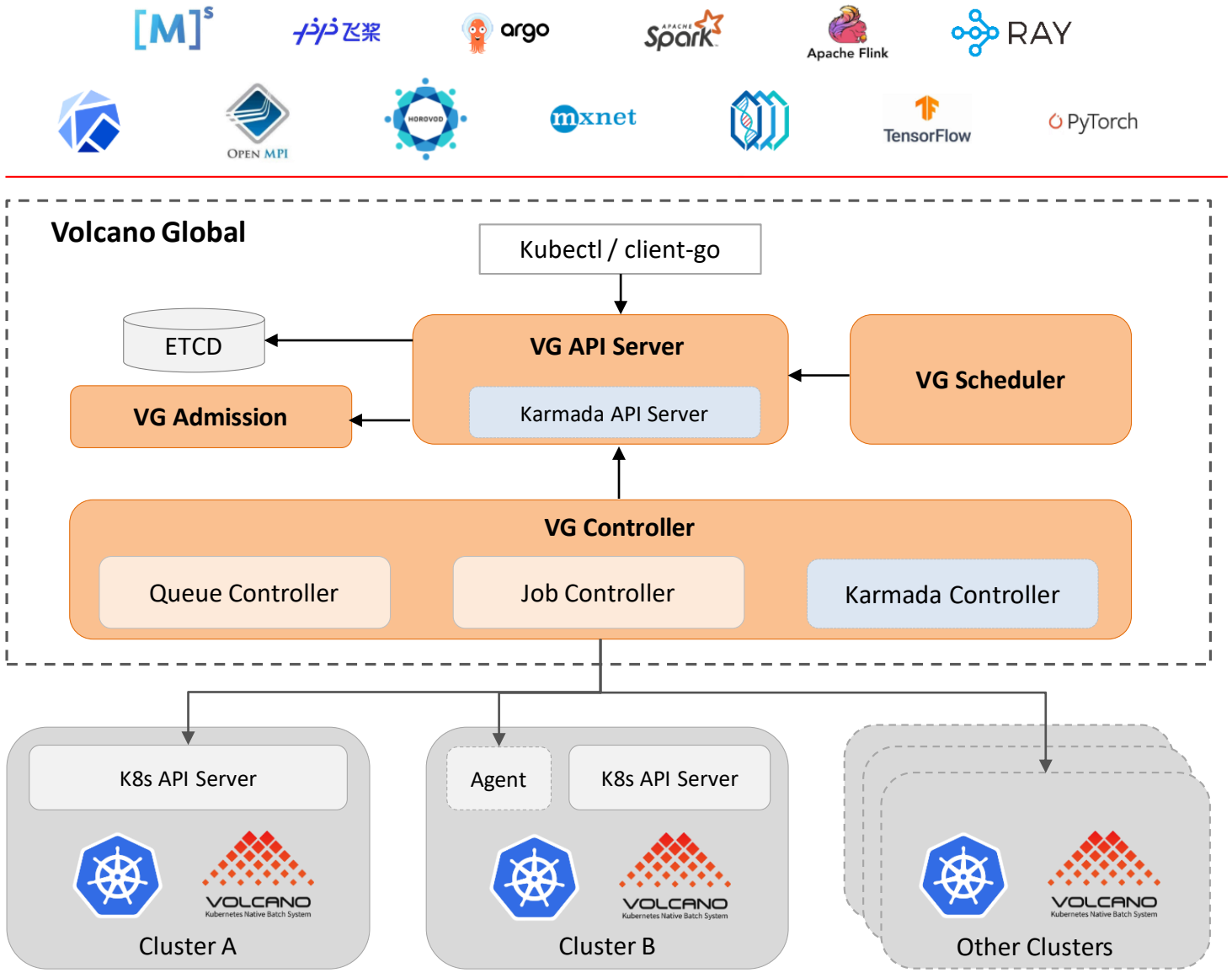


- **Resource oversubscription model**
  - ✓ Define priority of online and offline pod
  - ✓ Use allocated but unused resources to run low-priority task



- **Unified scheduler for online and offline workloads**
  - ✓ Support QoS awareness scheduling for online and offline workload.
  - ✓ Support rich scheduling policy for online and offline workloads, e.g. topology awareness, drf.
- **QoS management (enhanced OS & SLO agent):**
  - ✓ Oversubscription resource calculating, reporting, and interference checking etc.
  - ✓ CPU/Memory/Cache/Network/Disk isolation in OS.

Scenario 4: Cross-Cluster, Global Scheduling of AI and Big Data



Volcano Global (sub-project)

- **API server:** Resource management portal
- **Scheduler:** Scheduling resources to proper cluster
- **Controller Manager:** Resource status management and distribution
- **Admission:** Verifying tasks

Features

- Managing batch workload scheduling across multiple clusters in a unified manner
- Construct global resource view and reducing fragments
- Support fair scheduling for multi-tenant
- Support the cost awareness scheduling

Use Case 1: ING Migrates Big Data Platform from Yarn to K8s

DAP Landing

DAP Genesis

- Platform entry-point
- Project management

Buildpacks.io

jupyter

TensorFlow

Spark

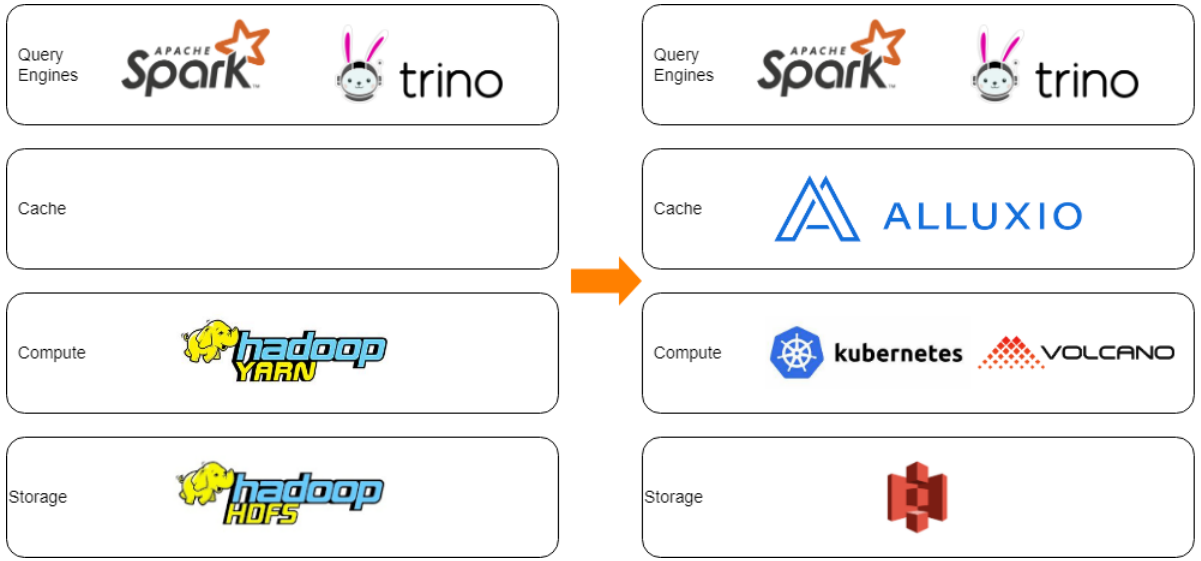
Data science in a box (Advanced analytics toolbox)

Amundsen

- Data discovery
- Metadata engine

Apache Superset

- SQL+BI toolset
- Dashboarding



Information reference: https://volcano.sh/en/blog/ing_case-en/

About ING

*ING (International Netherlands Groups) is a world-leading asset management company with services in more than **40 countries**. Its core business is **banking, insurance and asset management**. ING introduces cloud native infrastructure to create a next-generation, self-service big data analysis platform.*

Challenges

- Unified platform scheduling for interactive services, resident services, and offline analysis services
- Fair resource allocation to ensure business SLA and resource sharing for high utilization
- Fair multi-user resource allocation and quick response to high priority tasks

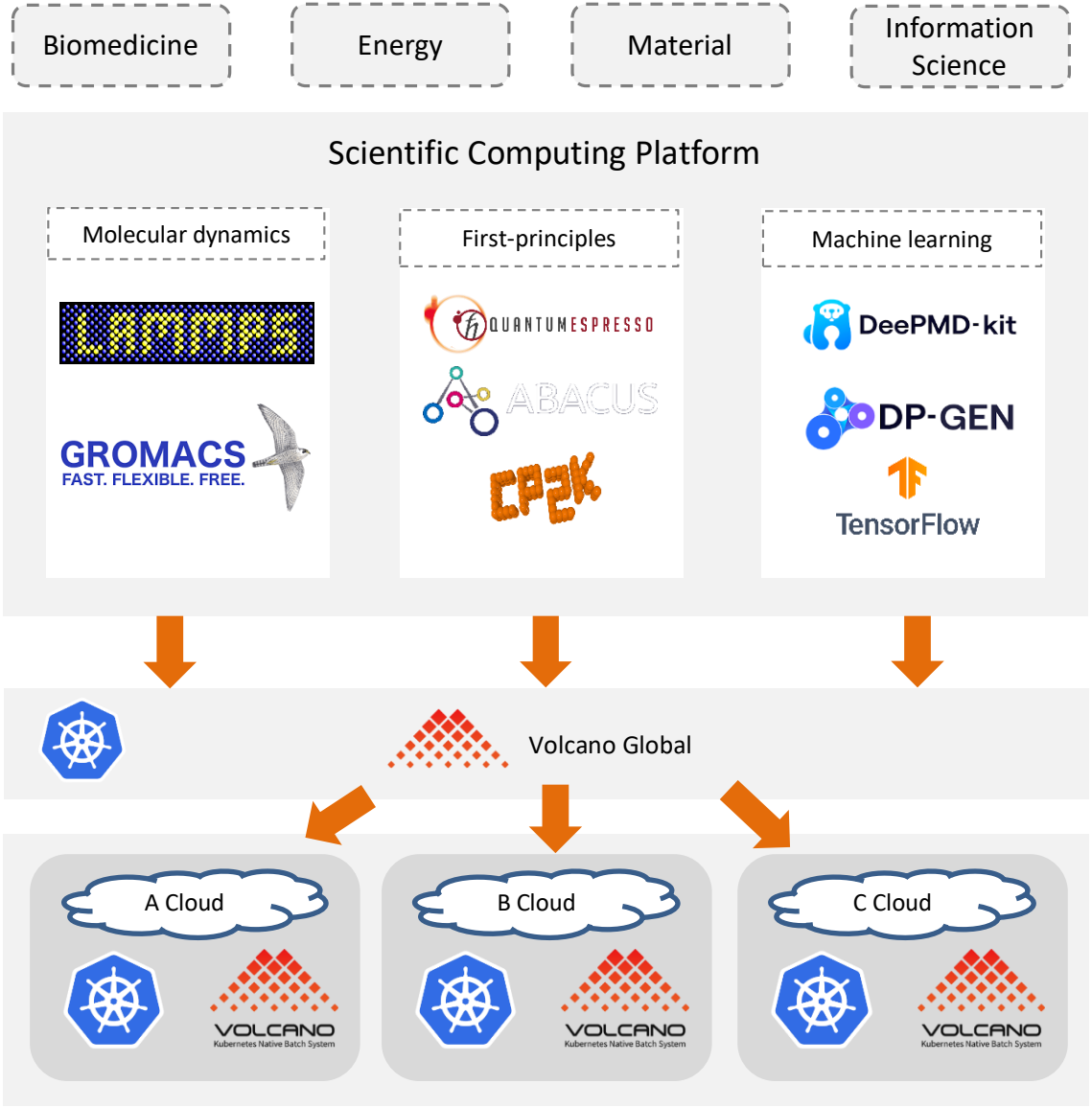
Solutions

- DRF algorithm for fair share resource allocation
- Job based preemption for quick response
- Queue dynamic resource sharing for high utilization
- Spark on Volcano optimization

Benefits

- Migrate workload from **Yarn to Kubernetes** smoothly
- Cloud native DAP platform serving **17 countries/regions**, **1,100 users**, with an annual growth rate of **8.1%**
- The number of projects running on the DAP platform has increased to **450+**

Use Case 2: Unified Management of Distributed Cluster Resources



Abort Platform

*An enterprise uses AI and molecular simulation algorithms to create a next-generation micro-scale industrial design and simulation platform for biomedicine, energy, materials, and information science and engineering research. The goal is to achieve high performance computing based on **multi-cloud** and **multi-supercomputing** cluster. The intelligent scheduling of tasks provides users with a "multiple, fast, and economical" computing experience.*

Challenges

- The business is distributed in **multiple independent clusters from different region and cloud provider**, and the operation and maintenance management are complicated.
- Drug discovering workload requires massive computing power, and the single region resources cannot meet the demand.
- Cost sensitive, and requires cost awareness scheduling and fair share scheduling.

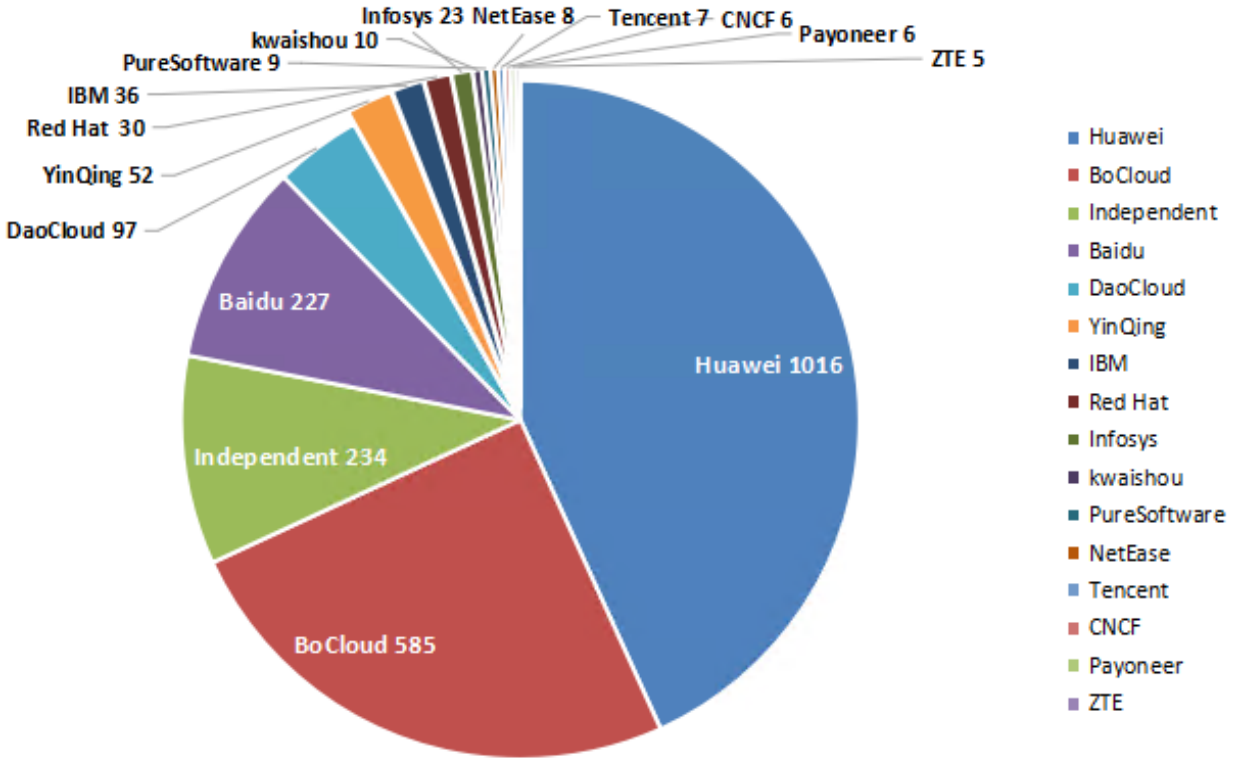
Solutions

- Provides a global scheduling based on Volcano to make multi-cluster management more easier.
- Support the scheduling policy such as **cluster load balancing**, **binpacking**, **cluster affinity** to make full use of resources.
- Use Volcano job to support Tensorflow, Pytorch, MPI workload uniformly

Volcano Community

Users

Top Contributors



Data from https://volcano.devstats.cncf.io

Join us

Website: https://volcano.sh/en/

Github: https://github.com/volcano-sh/volcano

Slack Channel: https://volcano-sh.slack.com/

KubeCon | CloudNativeCon
Europe 2023

Please scan the QR Code above
to leave feedback on this session