



KubeCon



CloudNativeCon

Europe 2026

#KubeCon #CloudNativeCon

Breaking the Monolith

**Decomposing and Governing Giant LLM Jobs
Across Clusters**

@kevin-wangzefeng



Background

Multi-Cluster in Enterprise AI

- **Clear Advantages**
 - Unified Resource Management
 - Flexible Multi-Cloud GPU Integration
 - Hardware Abstraction
- **Defining Cluster Boundaries**
 - By Geography, Capacity, or Hardware Specs
- **Complexity from AI Cluster Evolution**
 - SuperPOD / HyperNodes: New Cluster for New Generation

Trade-off: Utilization vs. Performance

- **Traditional Strategy**
 - Monolithic Single-Cluster Job Scheduling
- **Resource Fragmentation**
 - Mixed Workloads Deplete Contiguous GPUs
 - Bin-Packing & Preemption Aren't Silver Bullets
- **The Cost**
 - Monolithic Jobs Generate Massive "Resource Bubbles"

Our Goal & Where We Are

The Irony: We Build Boundaries to Cross Them.

Volcano-global v0.2

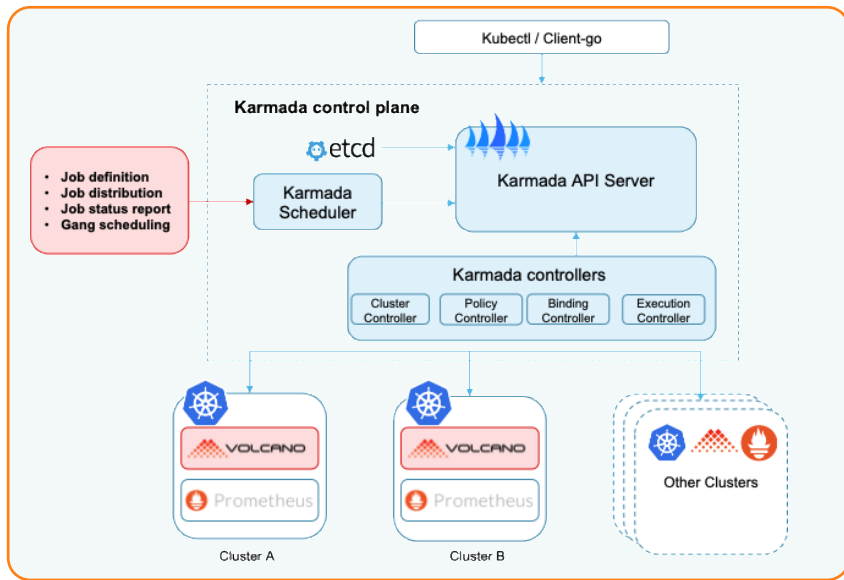
- Queue Capacity support

Karmada v1.14 - v1.16

- Improved cross-cluster
Failover and Eviction

Volcano-global v0.1

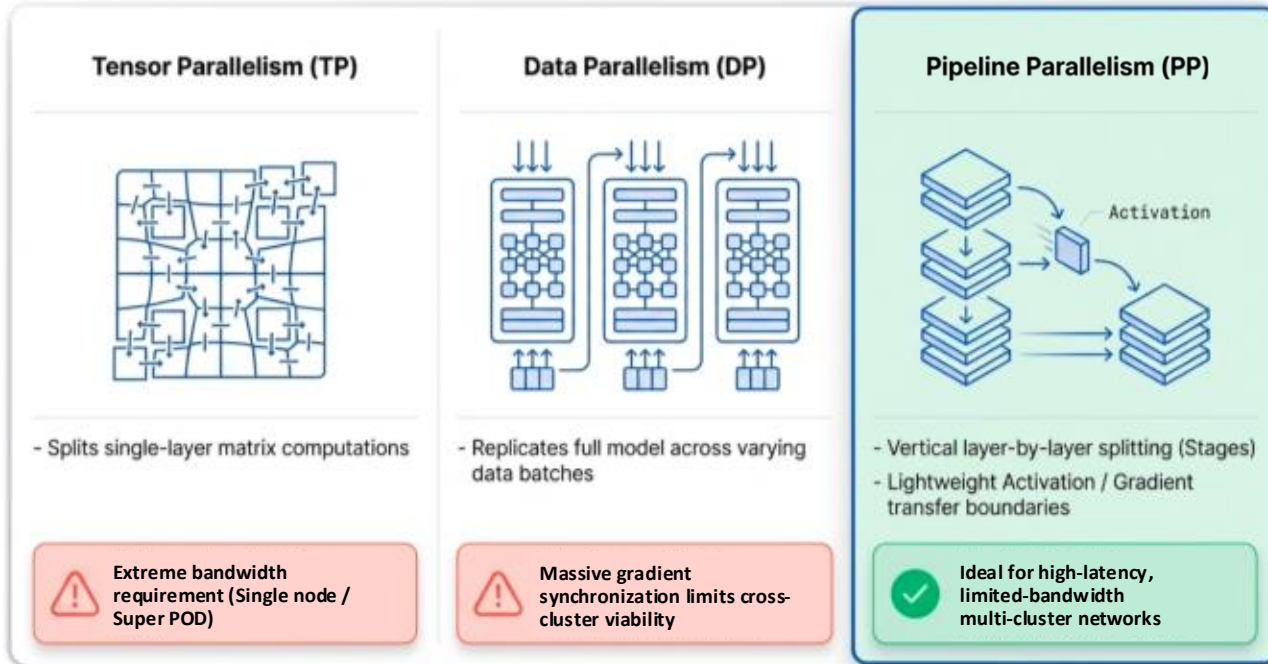
- Queue Priority support
- Integration with Karmada



The Missing Piece: Splitting Giant LLMs

- Federation-Native API Abstraction
- Extensible Architecture for Job Splitting and scheduling
- Precise resource monitoring, usage estimation& allocation

How shall we split the training job then?

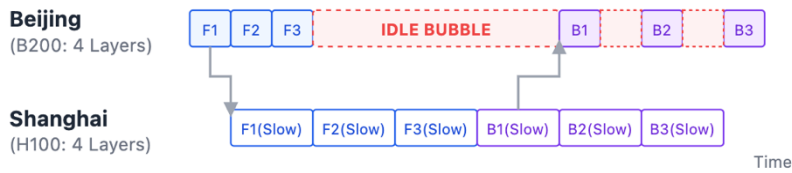


Pipeline Parallelism (PP) Wins for most multi-cluster networks

Asymmetric Pipeline Parallelism

Symmetric Partitioning (The Bottleneck)

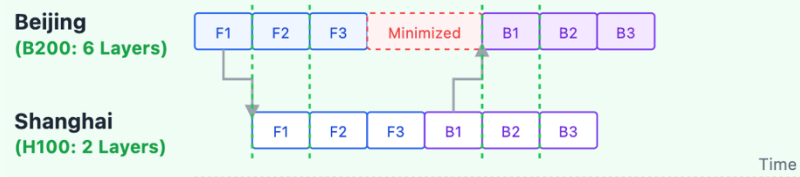
Fast clusters stall while waiting for slow clusters to complete Forward & Backward passes.



- Fast Clusters waiting for Slow Clusters ==> resource bubbles
- Asymmetric layer assignment ==> balanced Micro-batch execution
- Aligning execution timelines ==> eliminate idle intervals

Asymmetric Layout (Optimized)

Throughput (T_{step}) alignment synchronizes the timeline and compresses the wait bubble.



- Orchestrating heterogenous clusters via ReplicatedJobs
- Custom worker counts per stage
- Asymmetric hardware targeting

Asymmetric Pipeline Parallelism with HyperJob

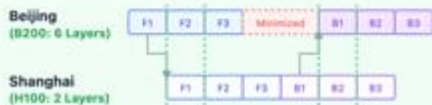
Symmetric Partitioning (The Bottleneck)

Fast clusters stall while waiting for slow clusters to complete Forward & Backward passes.



Asymmetric Layout (Optimized)

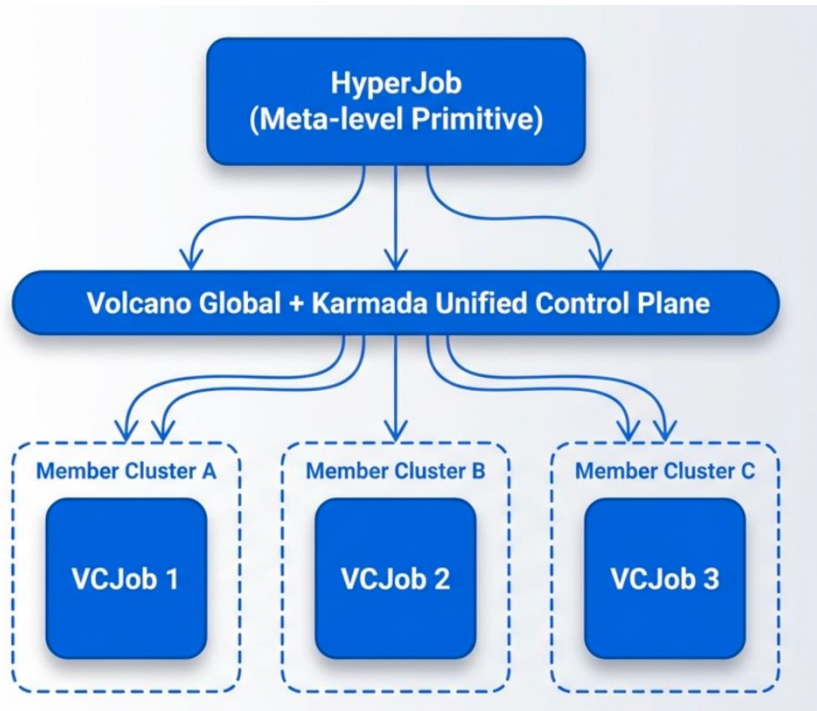
Throughput (T_{step}) alignment synchronizes the timeline and compresses the wait bubble.



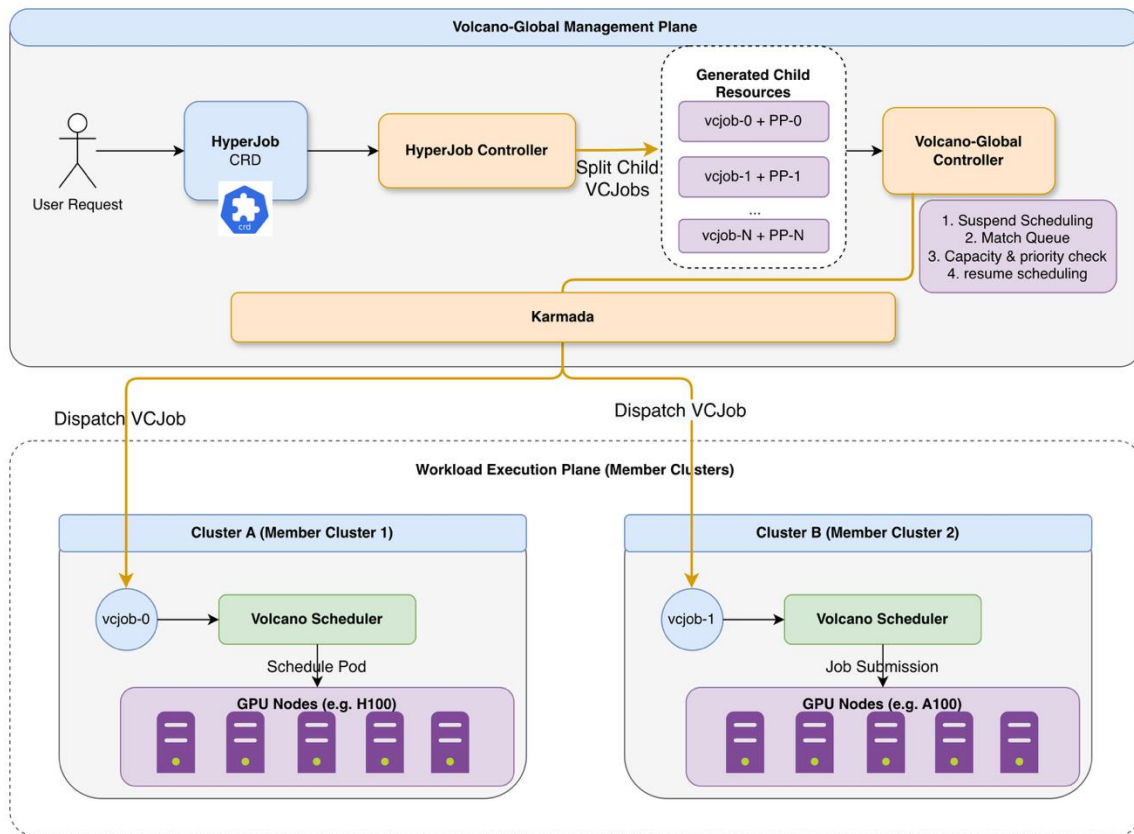
```
apiVersion: training.volcano.sh/v1alpha1
kind: HyperJob
metadata:
  name: heterogeneous-training
spec:
  replicatedJobs:
    - name: trainer
      clusterNames: ["cluster-gpu-b200"]
      replicas: 6
      templateSpec:
        # Detailed Volcano Job Config...
    - name: trainer
      clusterNames: ["cluster-gpu-h100"]
      replicas: 2
      templateSpec:
        # Detailed Volcano Job Config...
```

HyperJob: The Federated-Native Primitive

- **Meta-Level Orchestration**
 - Manages ReplicatedJobs, not individual Pods
 - Preserves full VolcanoJob autonomy within member clusters
- **Declarative Splitting**
 - Translates a unified AI Job into physically decoupled Job Deployments
- **Unified State Aggregation**
 - Abstracts fragmented distributed execution into a single, global terminal state (Completed / Failed)

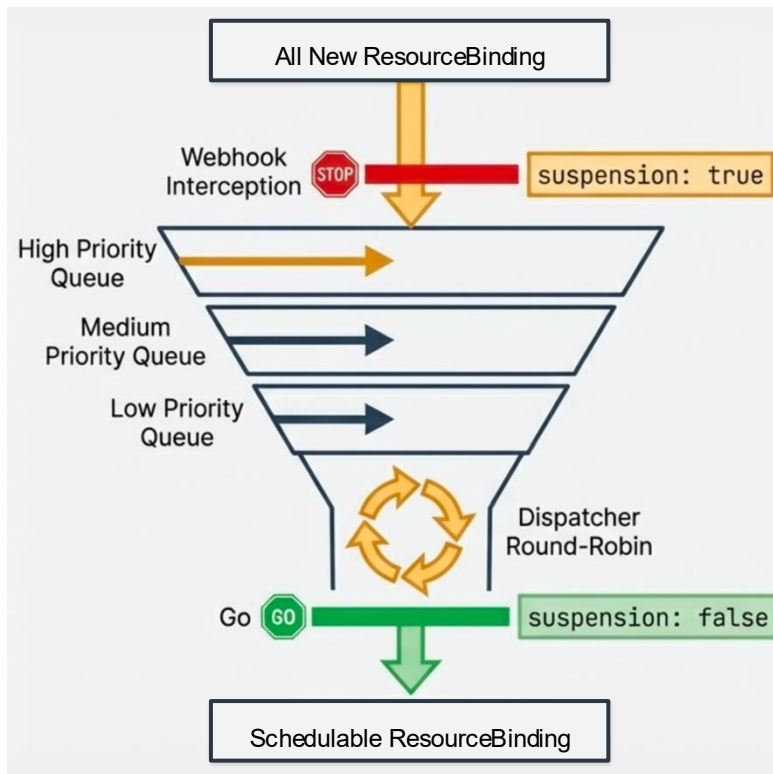


HyperJob API workflow with Volcano-global and Karmada

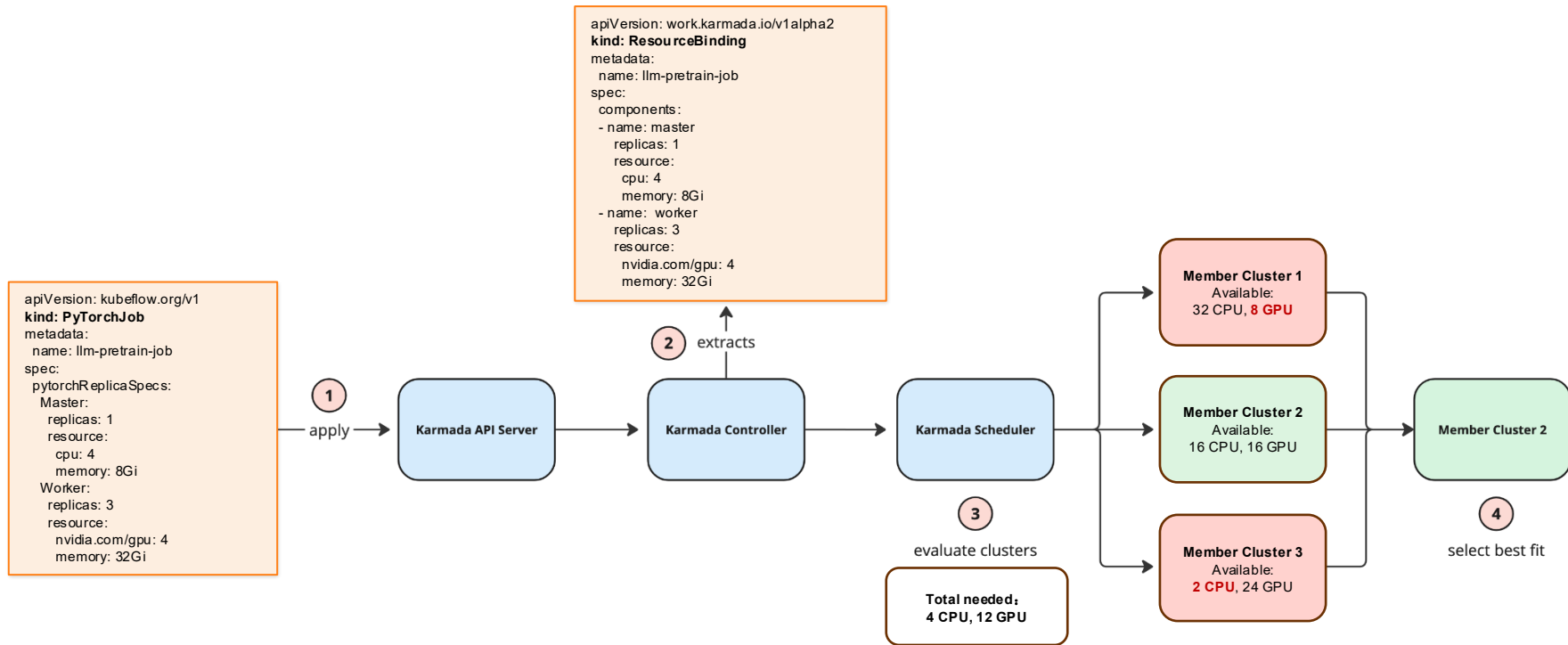


Global Control for Queue Priority and Capacity

- **Intercept & Suspend (The Webhook)**
 - Mutating Webhook intercepts ResourceBinding creation
 - Injects `spec.suspension.scheduling: true` to halt immediate scheduling
- **Sort & Queue (The Dispatcher)**
 - Orders queues by capacity and jobs by priority.
 - Applies Round-Robin dispatching to prevent queue starvation
- **Release to Execute (UnSuspend)**
 - Patches `suspension.scheduling: false`
 - Hands control back to Karmada Scheduler for cluster binding



Karmada Scheduler Enhancement: Multi-Components Awareness



Future Plan: Smarter SplitPolicy

- **More Resource Awareness & Dynamic Weighting**
 - Inter-cluster network latency, real-time resource utilization awareness
- **Pluggable Algorithm Adapters**
 - Decoupled interfaces for model-specific splitting logic
 - PP: Automated compute profiling for Transformer layer partitioning
 - DP: Exploring Hierarchical All-Reduce to minimize cross-cluster network overhead
- **Adaptive Fallback**
 - Multi-option splitting profiles for graceful scheduling degradation

Recap & Takeaways

- **The Paradigm Shift**

- From Single-Cluster Monoliths to Federation-Native Execution

- **HyperJob: The Federated-Native Primitive**

- consists of multiple VcJob
- Each VcJob scheduled to a cluster
- Decomposing Giant AI Jobs into Distributed VcJobs

- **The Unified Control plane: Volcano-global + Karmada**

- Global Queues & Priority
 - Fair Allocation Prevents Shared Pool Overload
- Karmada Enhancement
 - Multi-Template Aware Scheduling

- **A Practical Example**

- Symmetric splits across heterogeneous hardware creates massive pipeline bubbles
- Asymmetric PP: Aligning Compute Times Across Heterogeneous Hardware

- **Future Plan**

- Resource Awareness
- Dynamic Weighting
- Pluggable Algorithm Adapters
- DP Hierarchical All-Reduce Exploration
- Multi-option splitting profiles

Join the Community and stay tuned!



<https://karmada.io>



<https://github.com/karmada-io/karmada>



<https://slack.cncf.io> (#karmada)



<https://volcano.sh>



<https://github.com/volcano-sh/volcano>
<https://github.com/volcano-sh/volcano-global>



<https://slack.cncf.io> (#volcano)



KubeCon



CloudNativeCon

Europe 2026

Thank You

