
dapr
HELM
TiKV
KEDA
KubeCon
CloudNativeCon
China 2025

KubeCon CloudNativeCon China 2025

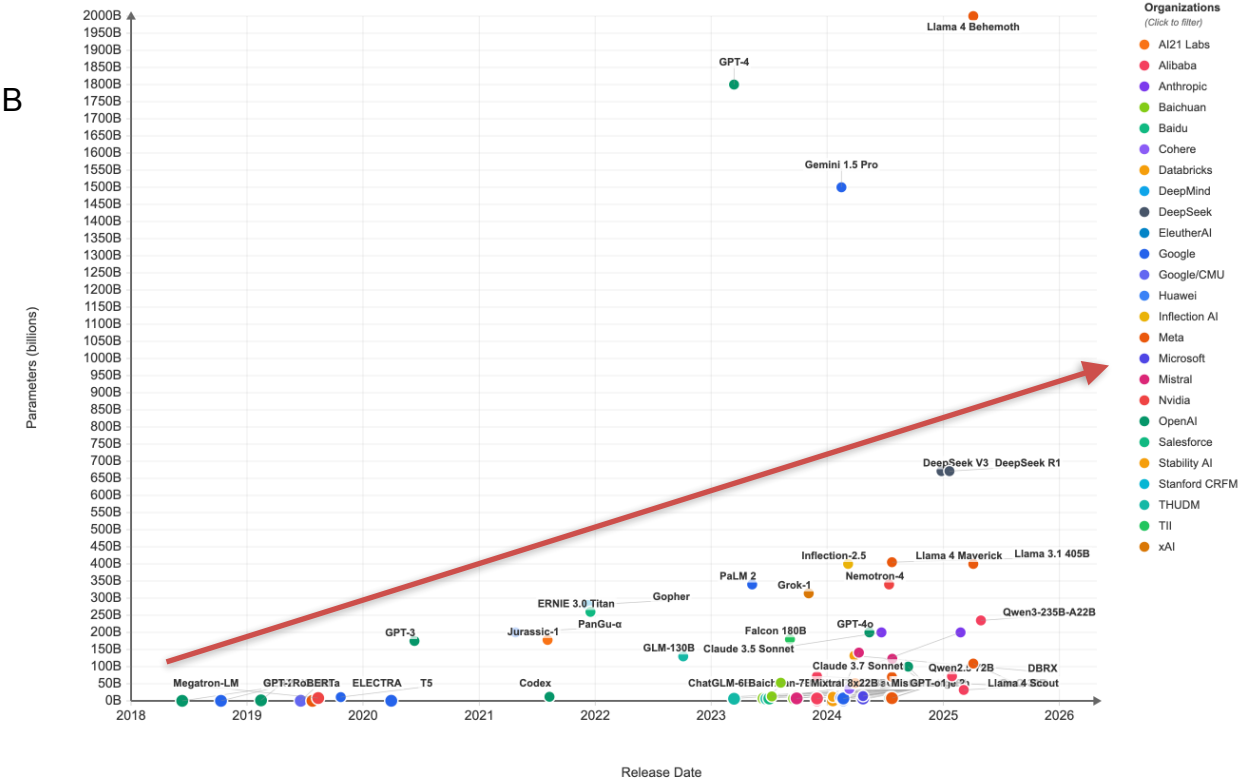
New pattern for sailing multi-host LLM Inference

Kante Yin
@kerthcet
https://ky.dev
LWS maintainer, InftyAI farmer

Why Multi-Host Inference

Why Multi-Host Inference

- **Growing model size**
  - DeepSeek R1 FP8: 671GB
  - KVCache: 400GB
  - Needs 16 * H100 80GB



from ModelSpec.ai

Why Multi-Host Inference

- **Growing model size**
  - DeepSeek R1 FP8: 671GB
  - KVCache: 400GB
  - Needs 16 * H100 80GB
- **High Throughput**
  - xPyD paradigm
  - Mitigate interference between P & D
  - Resource coupling

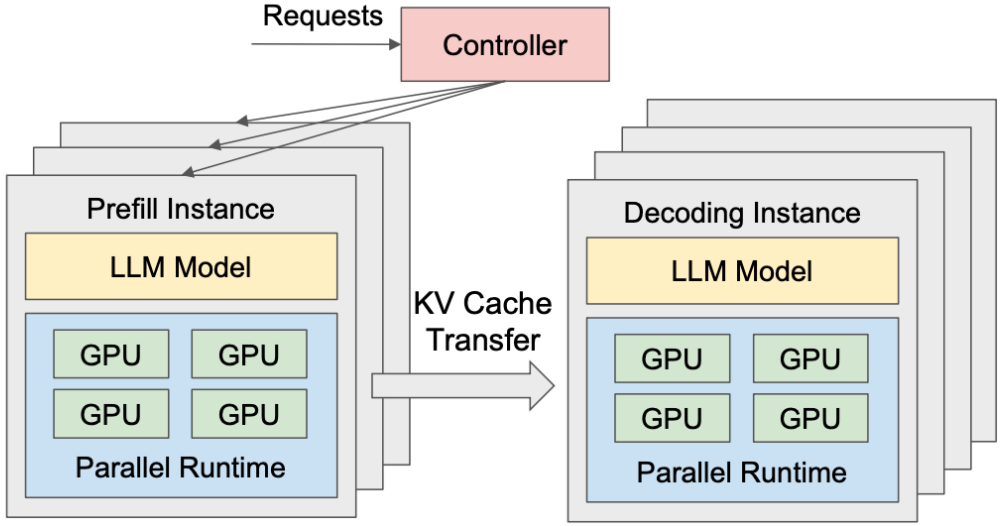
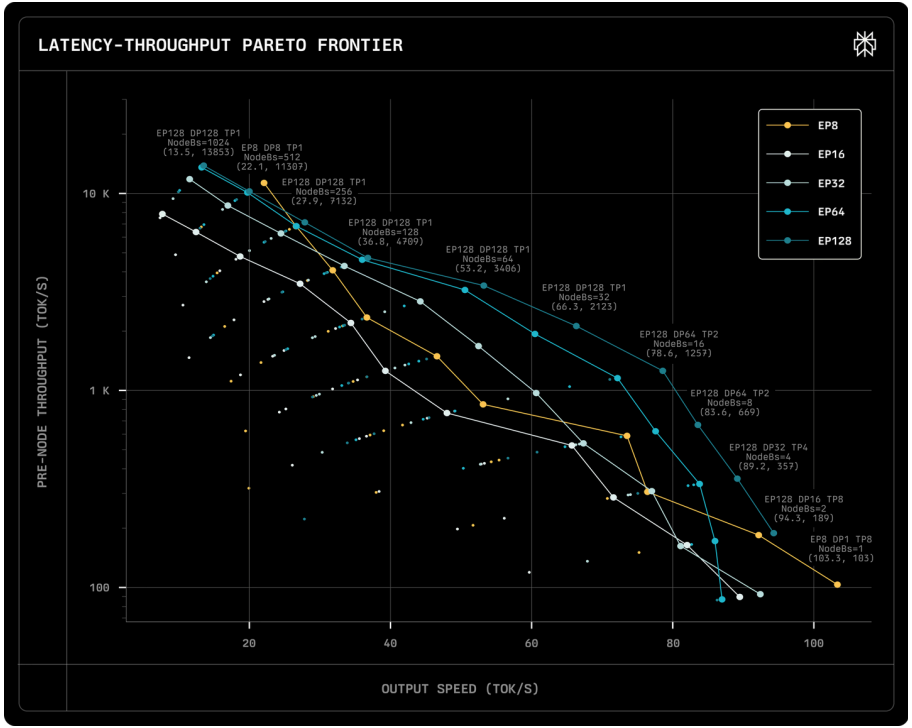


Figure 6: DistServe Runtime System Architecture

from DistServe

Why Multi-Host Inference

- **Growing model size**
  - DeepSeek R1 FP8: 671GB
  - KVCache: 400GB
  - Needs 16 * H100 80GB
- **High Throughput**
  - xPyD paradigm
  - Mitigate interference between P & D
  - Resource coupling



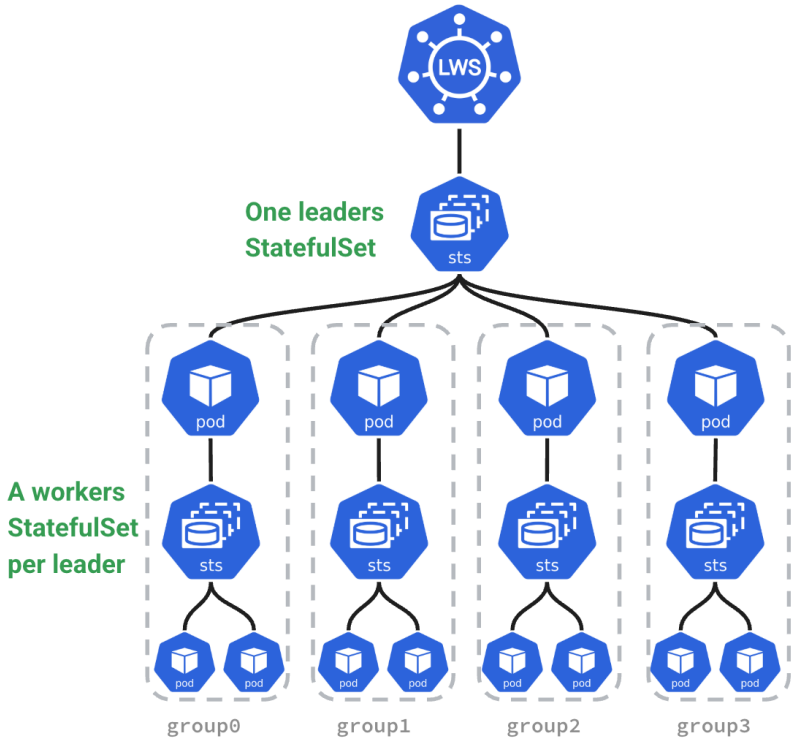
from [Perplexity]

What's LWS

LeaderWorkerSet (LWS) is an API for deploying **a group of pods as a unit** of replication, acting like **Statefulset on Statefulset**.

Design principles:

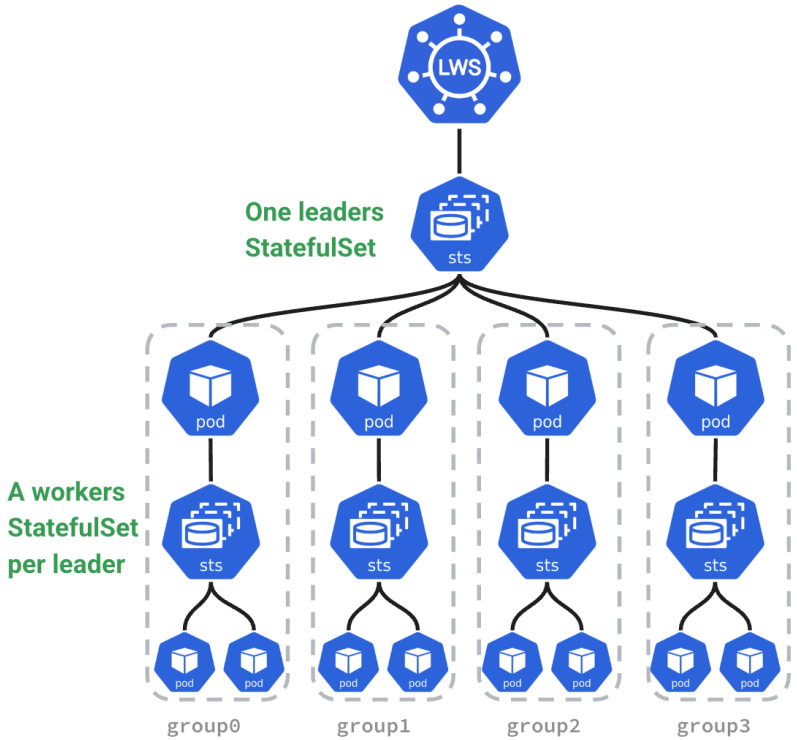
- Kubernetes capacity reuse
- 1 leader + n workers as a group (superpod), leader as the proxy
- The superpod should behave as an unit, e.g. lifecycle, rolling update.
- Each Pod should have an unique index because we're sharding, that's why we choose Statefulset



How it Works

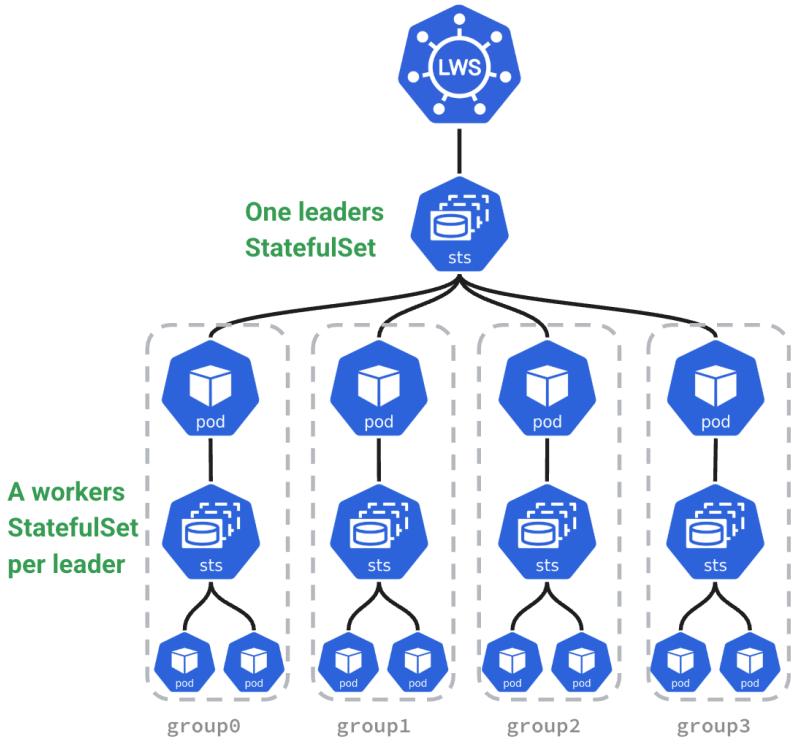
Workflow

- Create a leaderWorkerSet
- Create a leader Statefulset with replicas=4
- Each leader Pod creates a worker Statefulset with replicas= 2.
- The leader Pod and worker Pods are grouped as the superpod



What it Looks Like

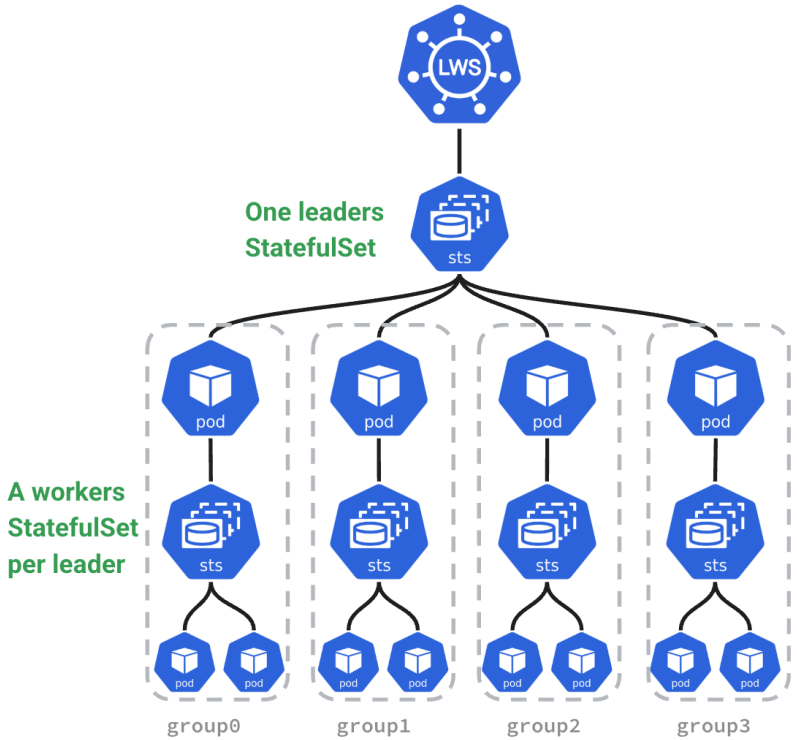

```yaml
apiVersion: leaderworkerset.x-k8s.io/v1
kind: LeaderWorkerSet
metadata:
  name: lws-ample
spec:
  replicas: 4
  leaderWorkerTemplate:
    size: 3
    leaderTemplate:
      spec:
        …
    workerTemplate:
      spec:
        …
  rolloutStrategy:
    type: RollingUpdate
    rollingUpdateConfiguration:
      maxUnavailable: 2
      maxSurge: 2
  restartPolicy: RecreateGroupOnPodRestart
```



Features

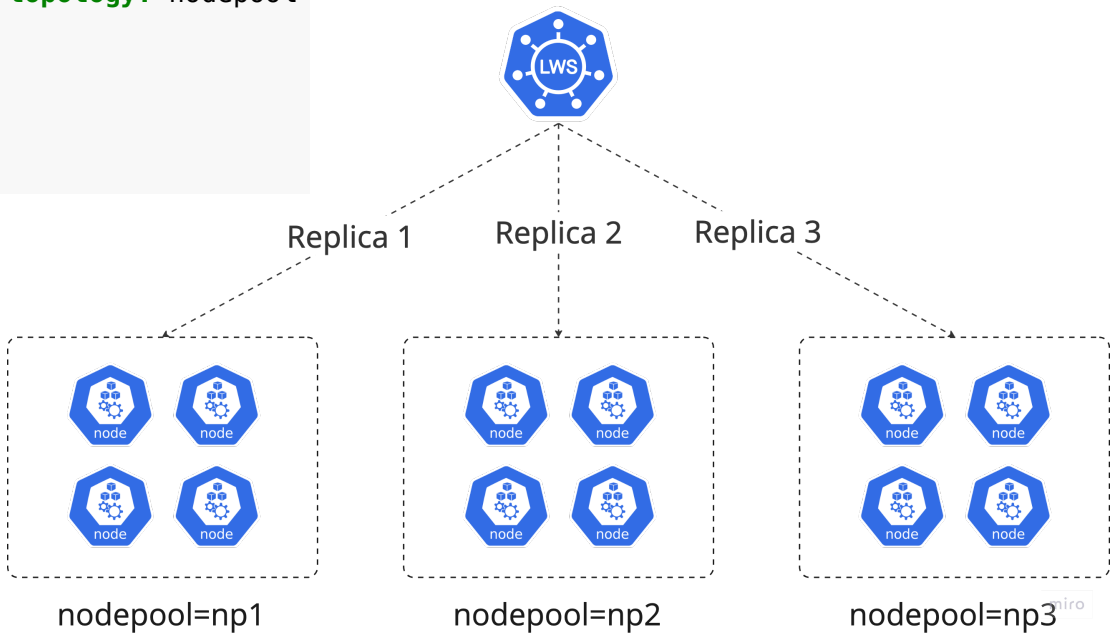
Features - HPA


```yaml
apiVersion: autoscaling/v2
kind: HorizontalPodAutoscaler
metadata:
  name: lws-hpa
spec:
  minReplicas: 3
  maxReplicas: 5
  metrics:
  - type: Resource
    resource:
      name: cpu
      target:
        type: Utilization
        averageUtilization: 50
  scaleTargetRef:
    apiVersion: leaderworkerset.x-k8s.io/v1
    kind: LeaderWorkerSet
    name: lws-sample
```



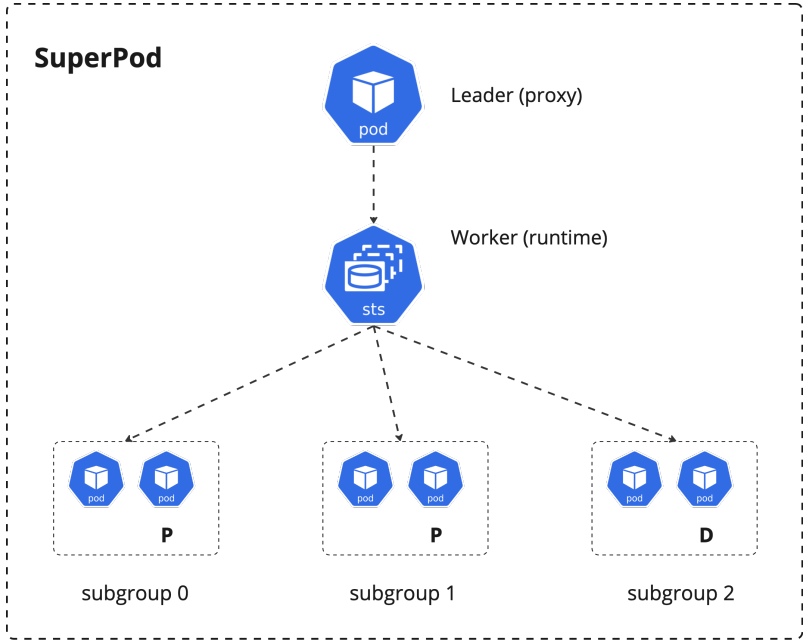
Feature - TopologyAwarePlacement


```yaml
apiVersion: leaderworkerset.x-k8s.io/v1
kind: LeaderWorkerSet
metadata:
  name: lws-sample
  annotations:
    leaderworkerset.sigs.k8s.io/exclusive-topology: nodepool
spec:
  replicas: 3
  leaderWorkerTemplate:
    size: 9
  …
```



Feature - SubGroup


```yaml
apiVersion: leaderworkerset.x-k8s.io/v1
kind: LeaderWorkerSet
metadata:
  name: lws-sample
spec:
  replicas: 3
  leaderWorkerTemplate:
    size: 7
    subGroupPolicy:
      type: LeaderExcluded
      subGroupSize: 2
  …
```



One Replica of LWS (P:D=2:1)

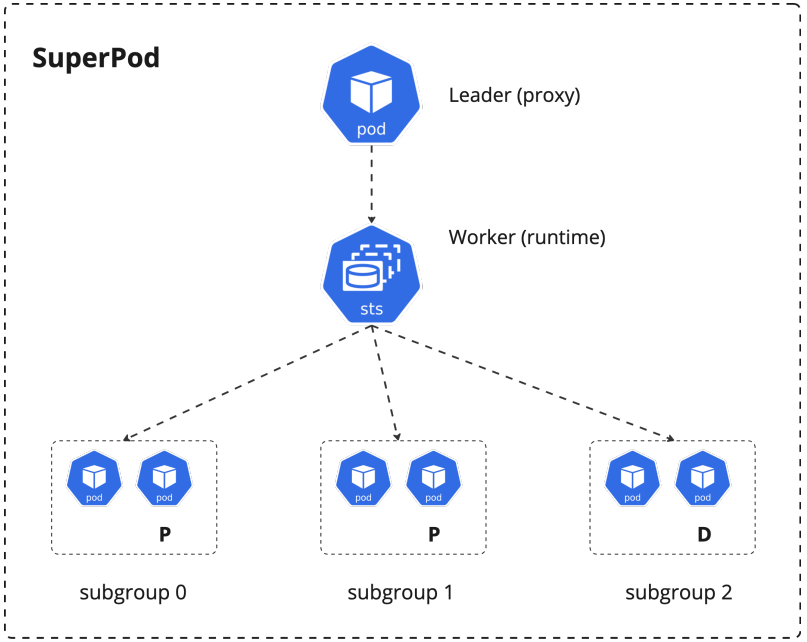
Ongoing Functions

KEPs

- Gang scheduling support, #407 — Volcano community
- ResourceClaimTemplate integration, #444
- Fine-gained system metrics, like rolling update duration, #87
- Make spec.leaderWorkerTemplate.size mutable, #552 — MistralAI
- Partitioned update support, #511 — Alibaba
- In-place rolling update for image, #376 — OpenKruise community
- Unique node selector and toleration per replica, #223

Mostly planned for v0.7.0, see issues.

Disaggregated Serving



One Replica of LWS (P:D=2:1)

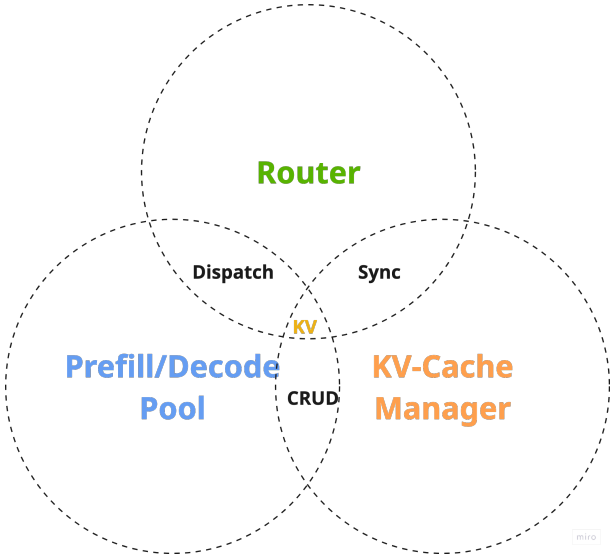
Homogeneous Disaggregated Serving with LWS:

- Proxy can't scale independently
- Static PD ratio, scaling PD as a whole
- With identical Pod template, no separate resource pools for P & D

Disaggregated Serving

Heterogeneous Disaggregated Serving:

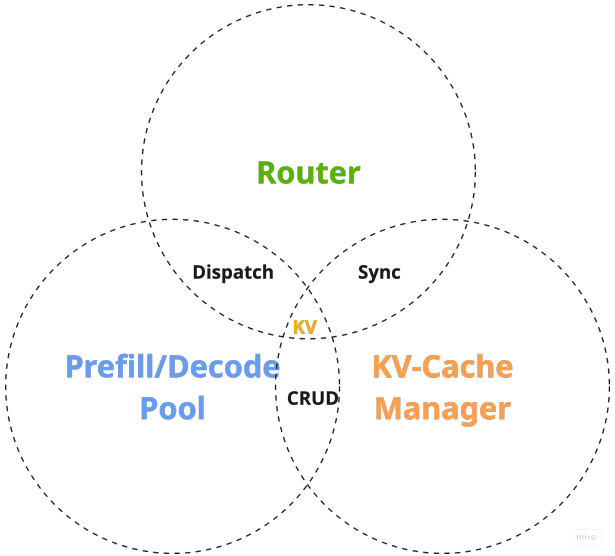
- Multiple role templates, e.g. Proxy, Prefill, Decode
- Rolling update strategy based on P-D ratio
- Independent scaling capacity



Disaggregated Serving

Heterogeneous Disaggregated Serving:

- Multiple role templates, e.g. Proxy, Prefill, Decode
- Rolling update strategy based on P-D ratio
- Independent scaling capacity



We're looking to build a new orchestration on top of LWS.

Adopters & Integrations

Adopters:

- AWS
- DaoCloud
- Google Cloud
- Nvidia
- More than we know …

Integrations:

- Nvidia Dynamo
- llmaz
- vLLM
- SGLang

Learn more details at our website. Please join the list if you use LWS as well.

Join Us

Github: https://github.com/kubernetes-sigs/lws

Website: https://lws.sigs.k8s.io/

Slack: we're under the guidance of wg-serving

💙 Thanks to all the contributors!

Join Us

Github: https://github.com/kubernetes-sigs/lws

Website: https://lws.sigs.k8s.io/

Slack: we're under the guidance of wg-serving

🌟 Give us a star !

Join Us

Github: https://github.com/kubernetes-sigs/lws

Website: https://lws.sigs.k8s.io/

Slack: we're under the guidance of wg-serving

🏃 Join us !


dapr
HELM
TiKV
KEDA
KubeCon
CloudNativeCon
China 2025
Thanks!