
KubeCon
CloudNativeCon
North America 2023

KubeCon | CloudNativeCon

North America 2023

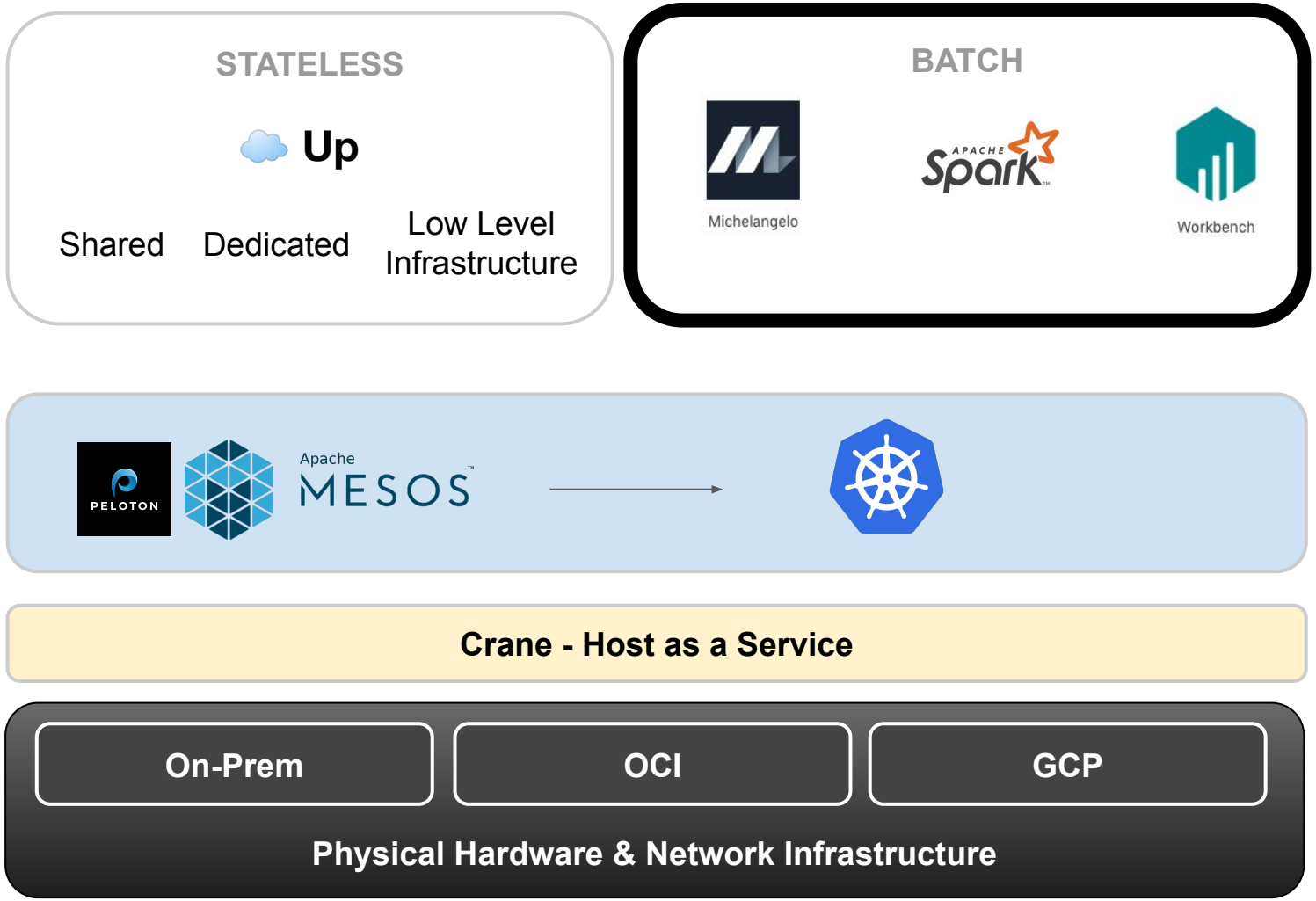
Efficient Resource Utilization for Batch Compute on Kubernetes

Amit Kumar
Kevin Xu

Outline

- Batch Compute at Uber
- Importance of Resource Sharing
- Challenges
- Solutions
  - Regional Resource Management & Federation
  - Specialized Hardware Efficiency
- Future Work
- Q & A

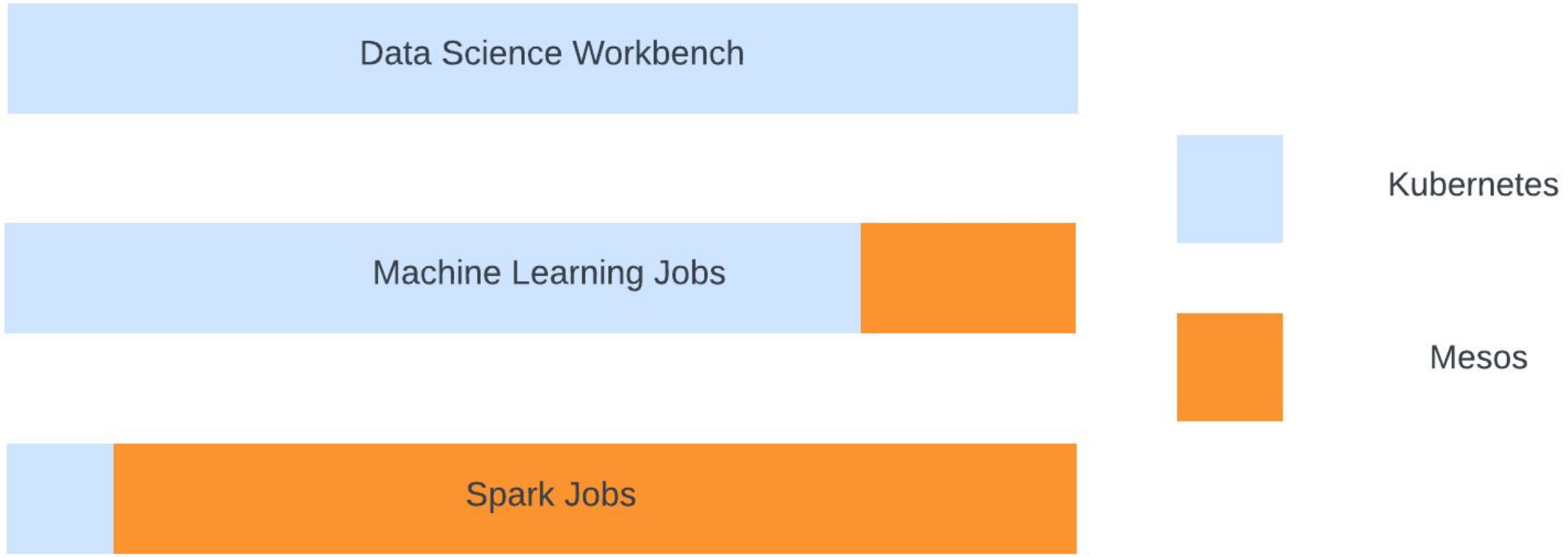
Compute Team @ Uber



Batch Compute at Uber: **Workloads & Use Cases**

Workload Types	Use Cases
APACHE Spark RAY Kubernetes Job	End User Use Cases: ● Rider Pricing Intelligence ● ETA Estimation ● Destination Suggestion Platform Use Cases: ● AI Model Training ● Data Science Notebooks

Batch Compute at Uber: **Workloads & Use Cases**



Batch Compute at Uber: **Scale**

~30k
#hosts

~1M+
#cores

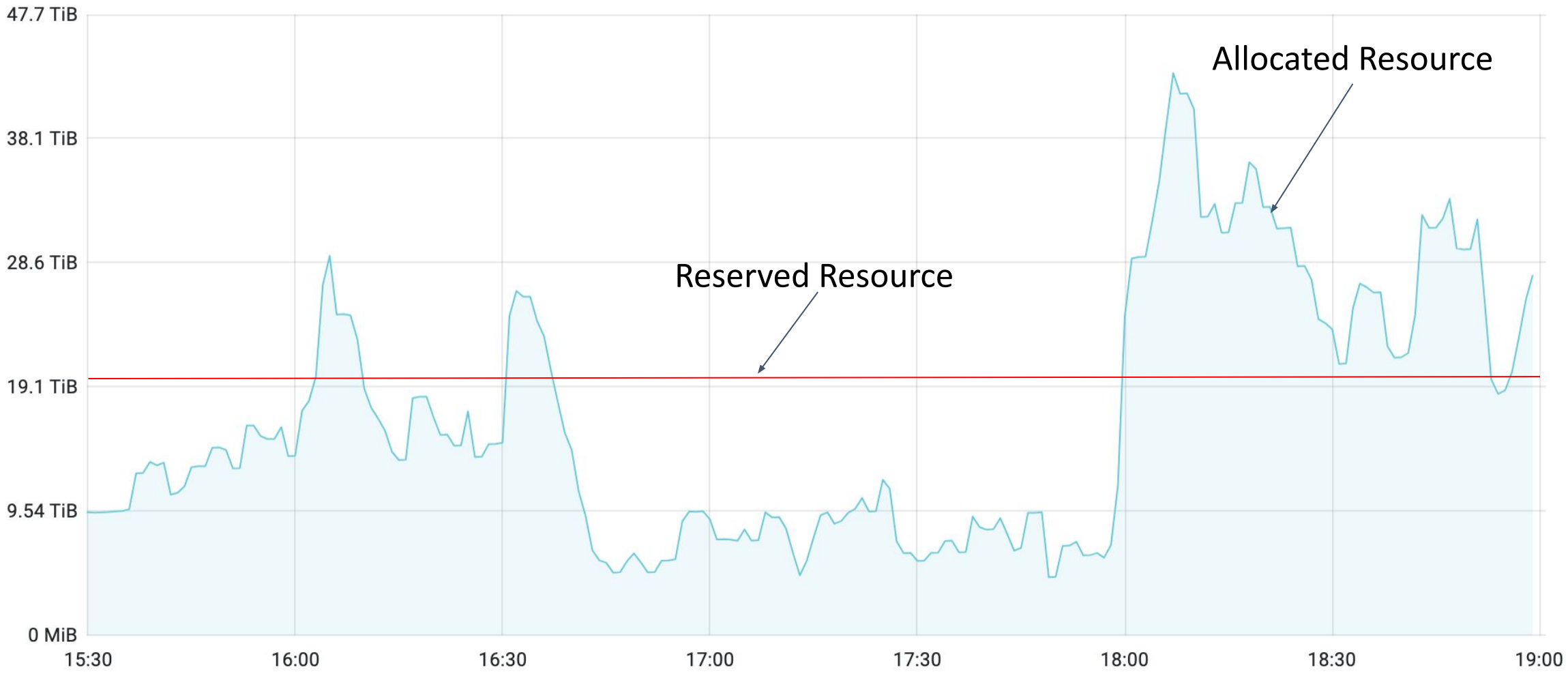
~4000
#gpus

~3M
#containers/day

~500
#pods/s

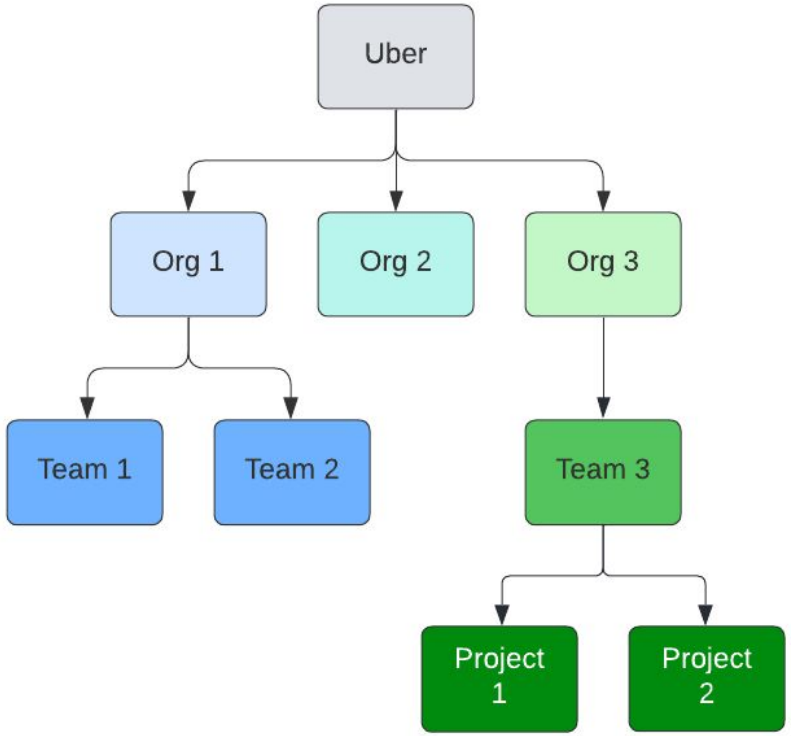
Importance of Resource Sharing

Importance of Resource Sharing: **Team Resource Allocation**

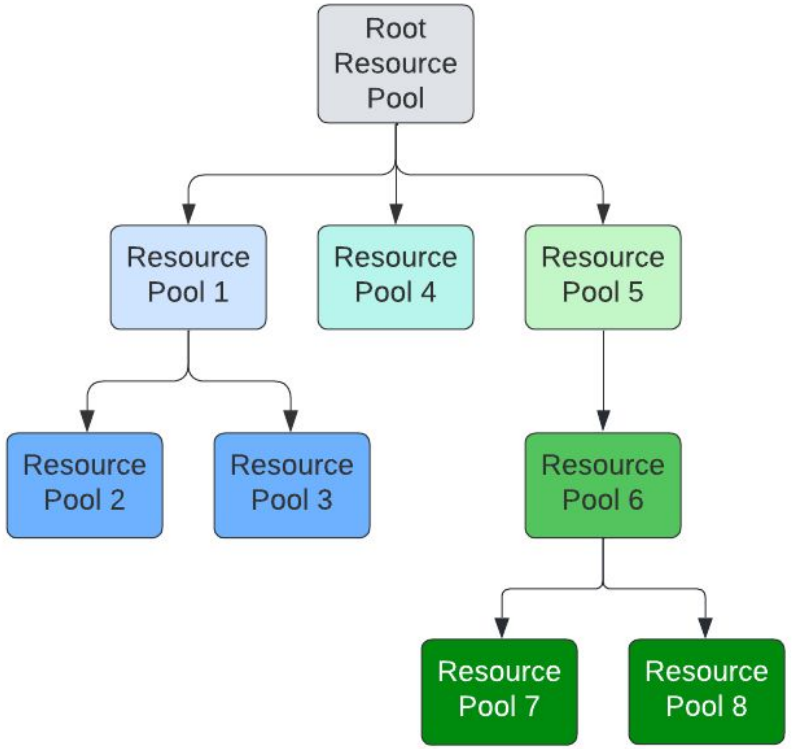


Importance of Resource Sharing: **Total Resource Allocation**

Resource Pools: **Organizational Capacity Management**



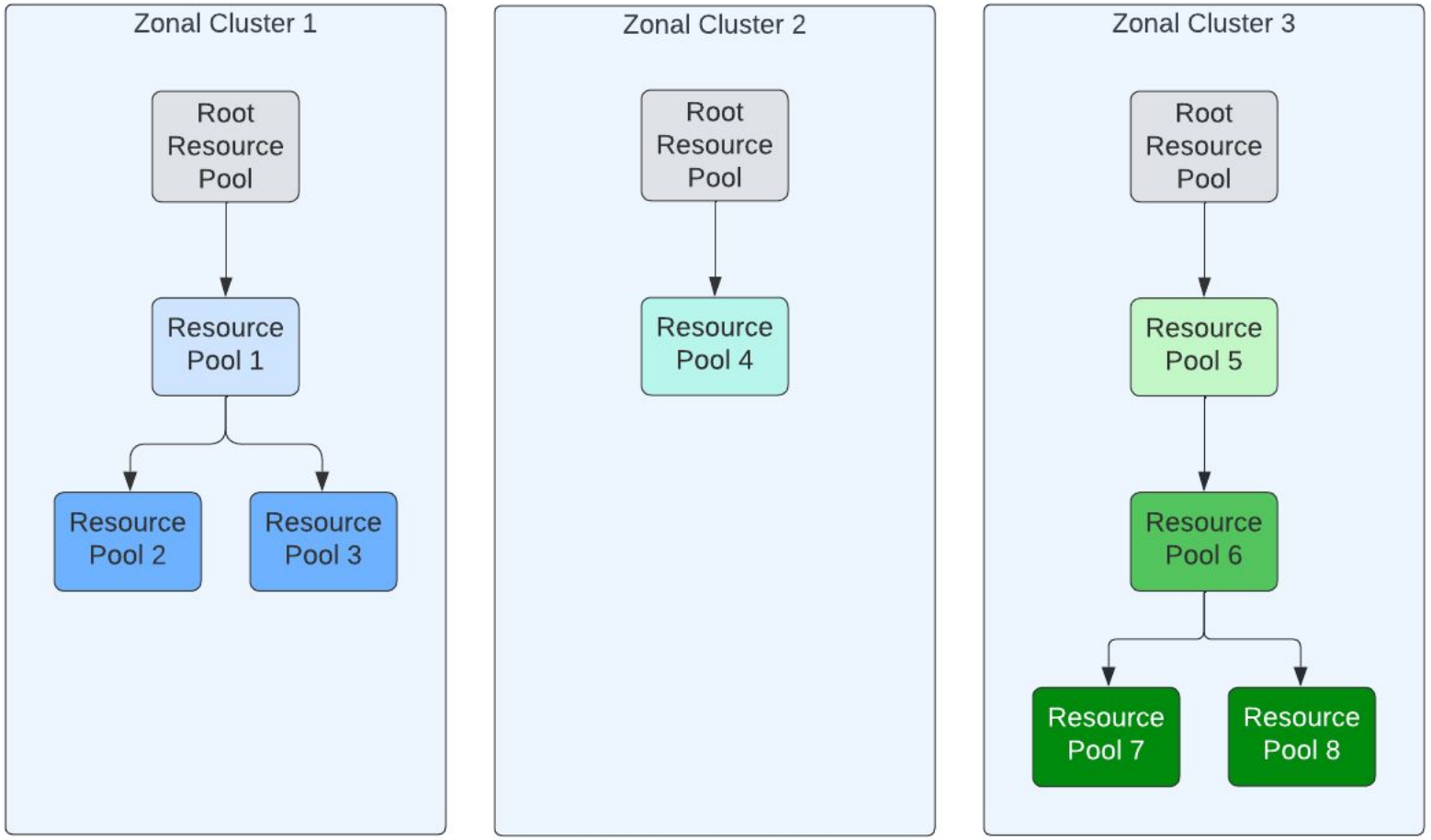
Organization Structure



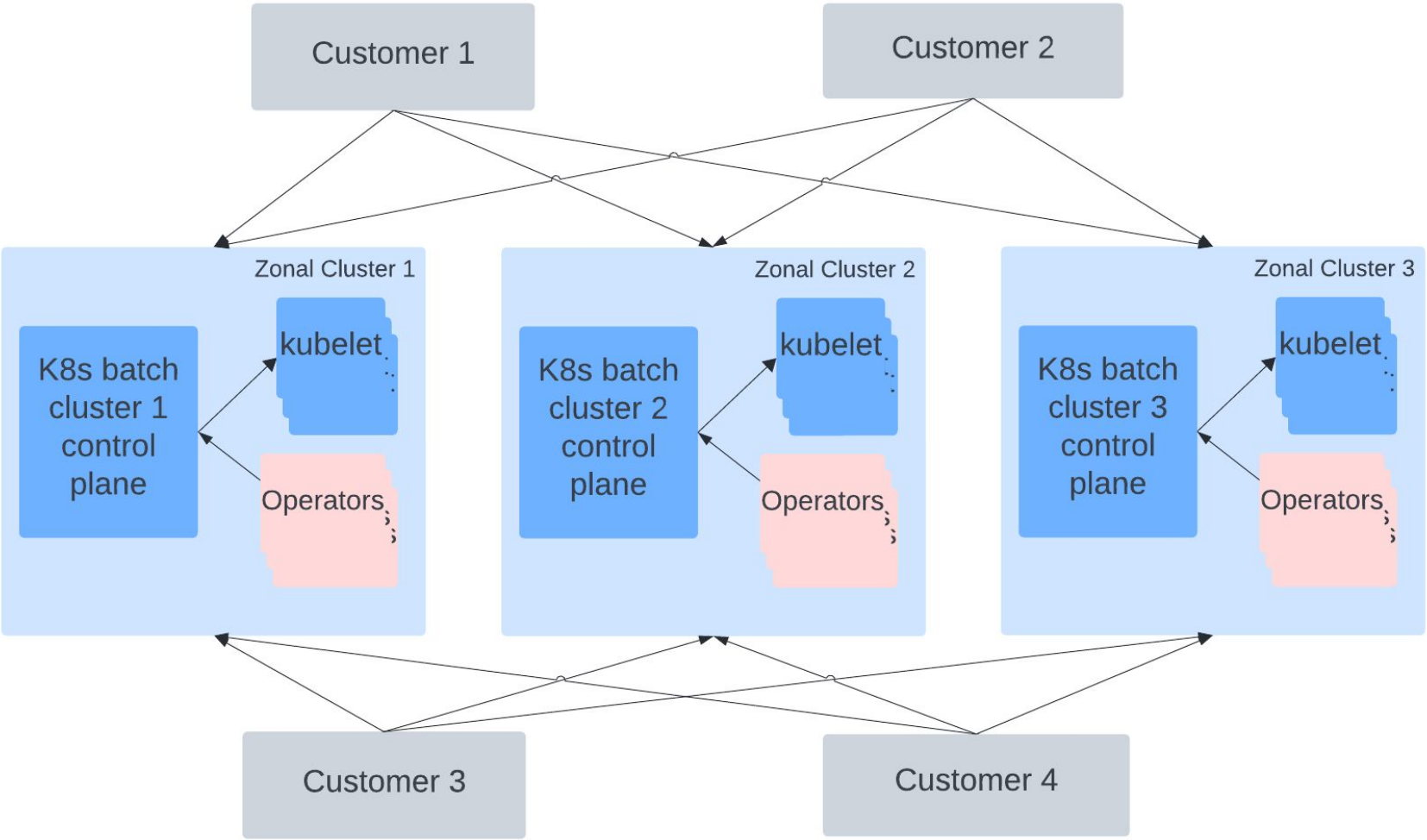
Resource Pool Structure

v1 Architecture

v1 Architecture: **Cluster Local Resource Management**



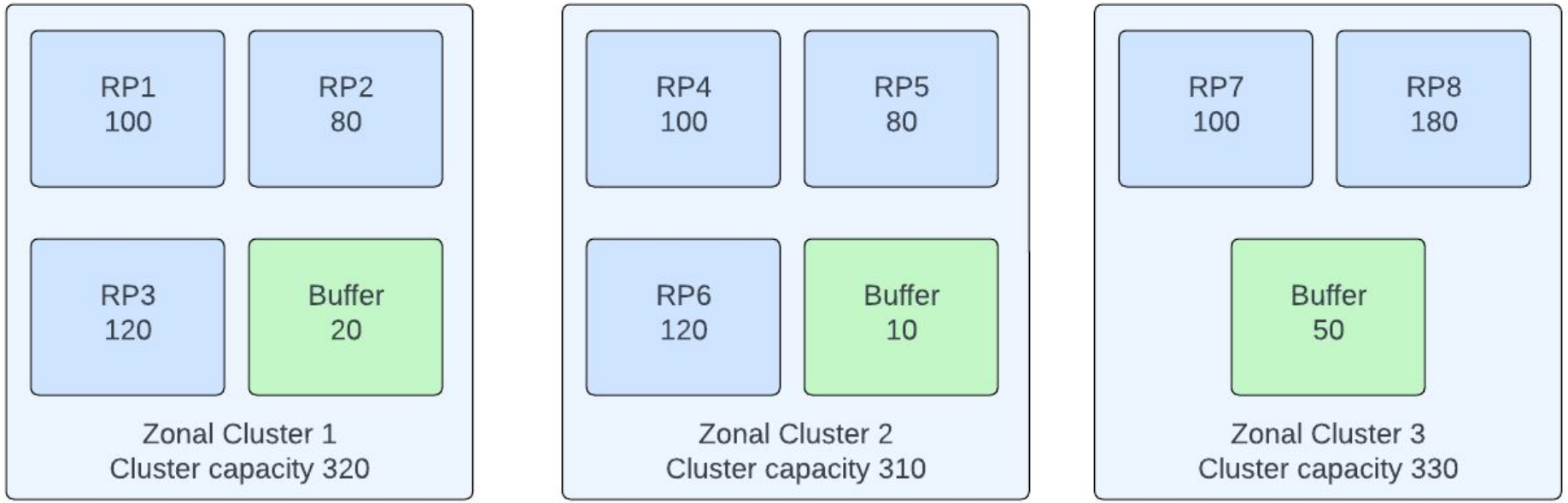
v1 Architecture

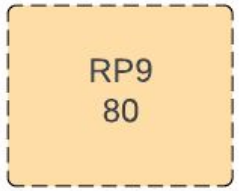


Challenges

1. Fragmentation
2. Non-Uniform Cluster Usage
3. Zonal Availability of Resource Pools
4. Cluster Management and Operations

Challenge #1: **Fragmentation**

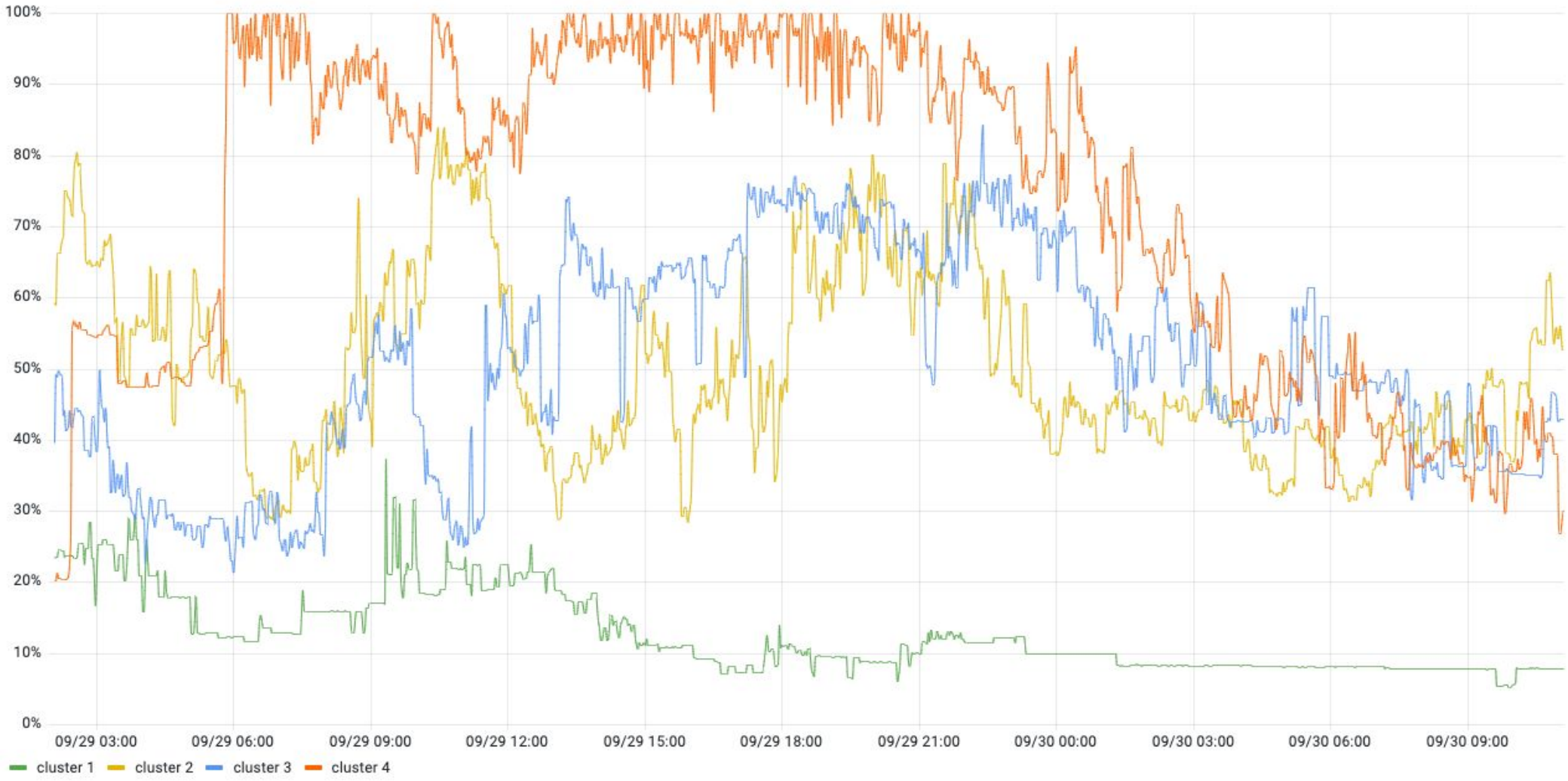




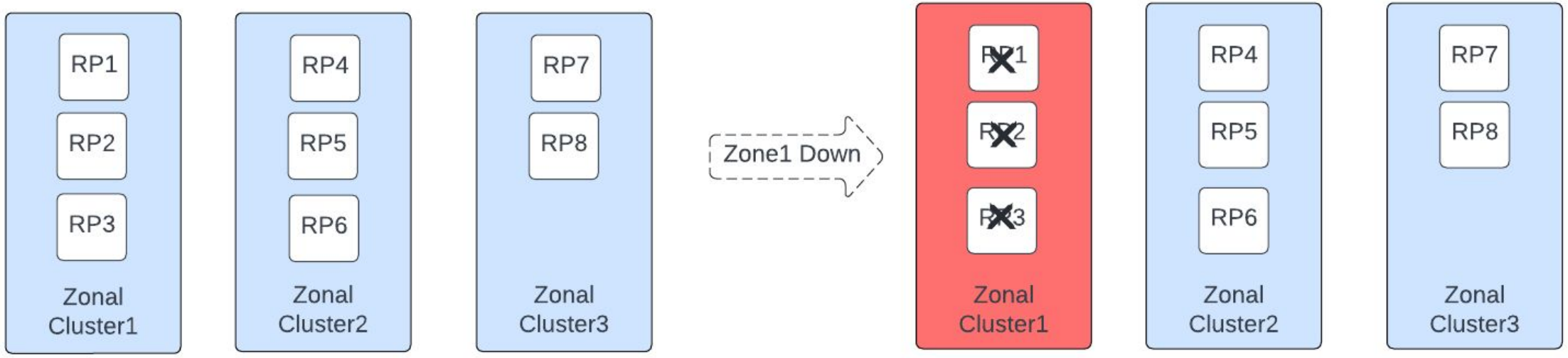
Total available capacity = 960
Total fragmented capacity = 20 + 10 +50 = 80

Incoming resource pool RP9 with capacity requirement of 80 can not be created

Challenge #2: **Non-Uniform Cluster Usage**



Challenge #3: **Zonal Availability of Resource Pools**



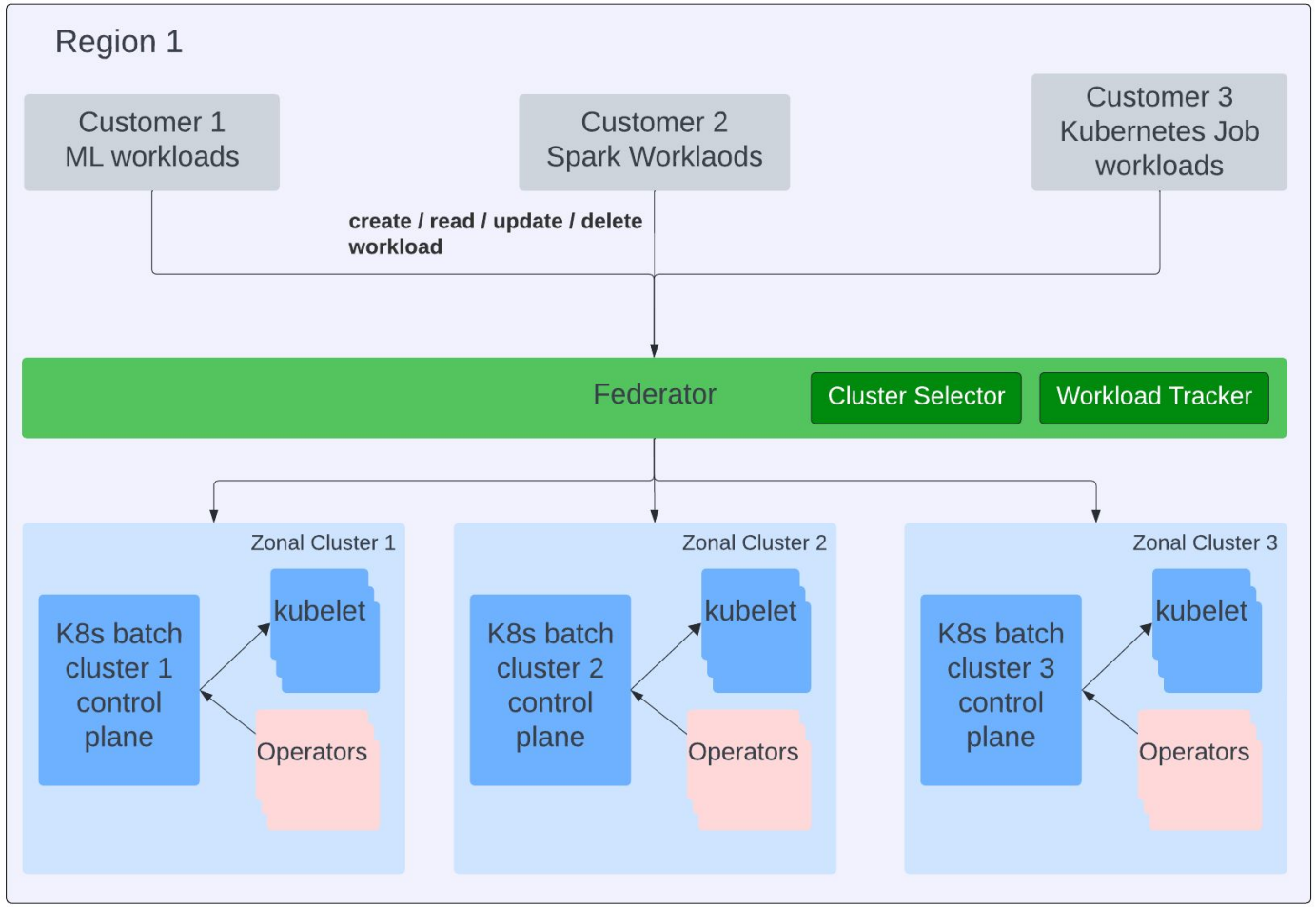
Zone1 goes down
Workloads of RP1, RP2 and RP3 can not run

Challenge #4: **Cluster Management and Operations**

- Coordination required to turn (up/down) cluster
- Cluster Selection
- Releases and Upgrades

Solution: Federation for Batch Compute

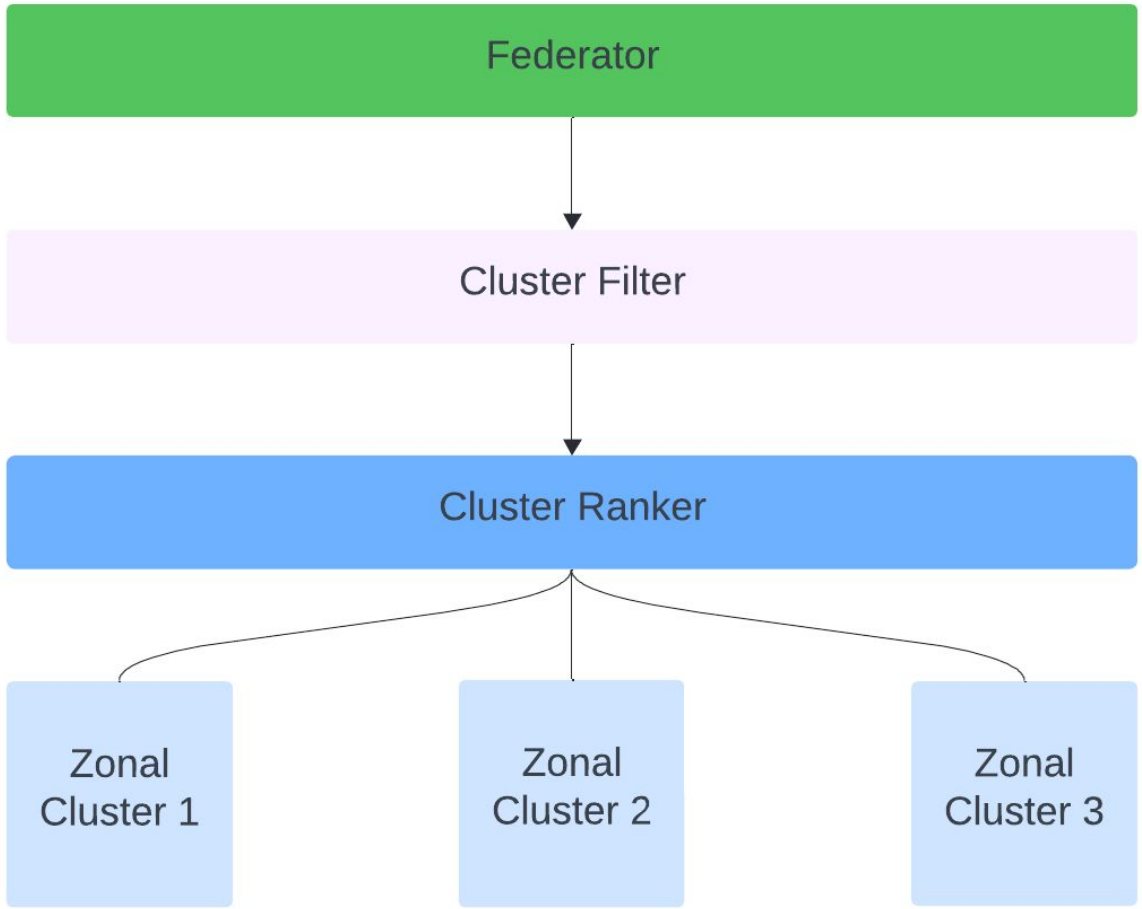
v2 Architecture



v2 Architecture: **Cluster Selection**

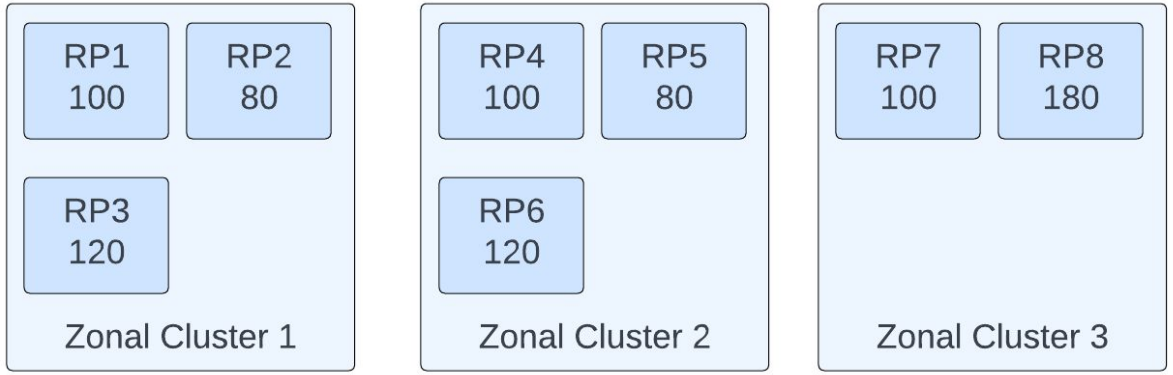
Filter out ineligible clusters

Rank clusters

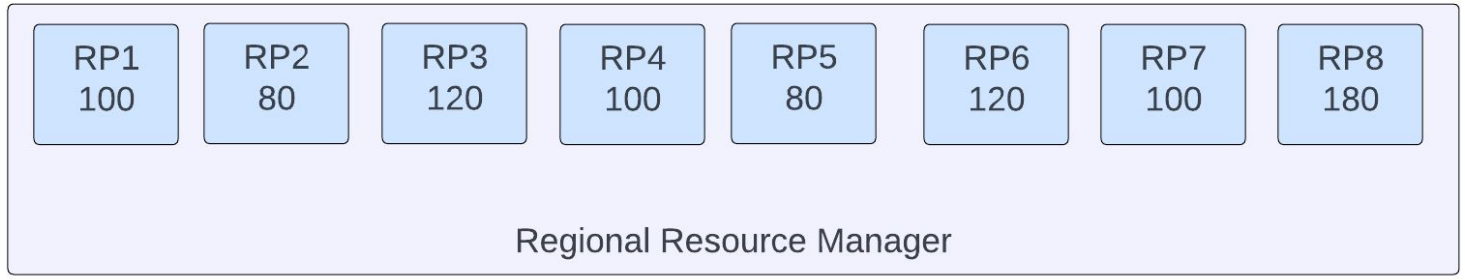


v2 Architecture: **Regional Resource Management**

Zonal



Regional

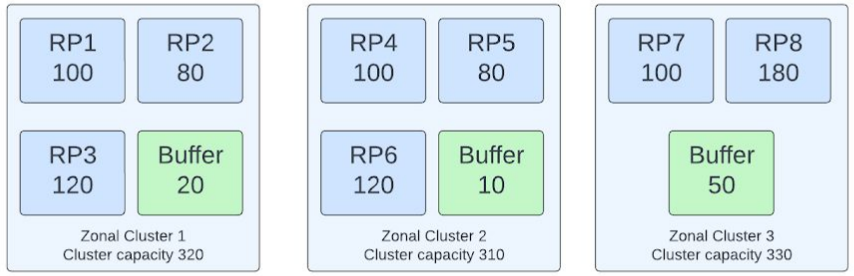


Efficient Resource Utilization for Batch Compute on Kubernetes

- Federation & Regional Resource Management
  - #1 No More Fragmentation
  - #2 Uniform Cluster Usage
  - #3 Regional Availability
  - #4 Cluster Management
- Elastic Resource Sharing
- Specialized Hardware Efficiency
- Future Work

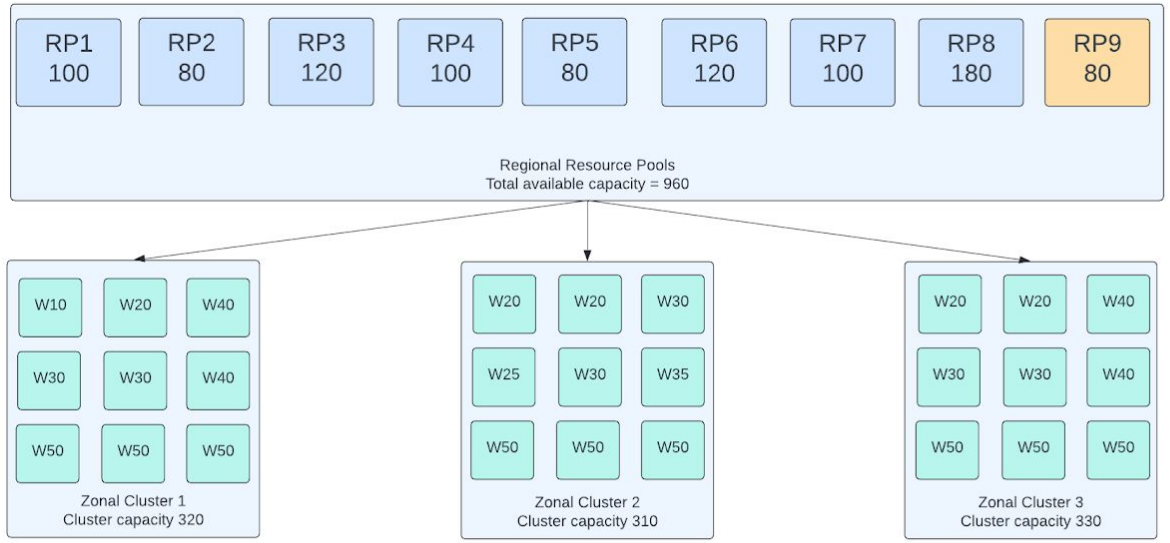
Regional Resource Management: **#1 No More Fragmentation**

v1



Total available capacity = 960
Total fragmented capacity = 20 + 10 +50 = 80

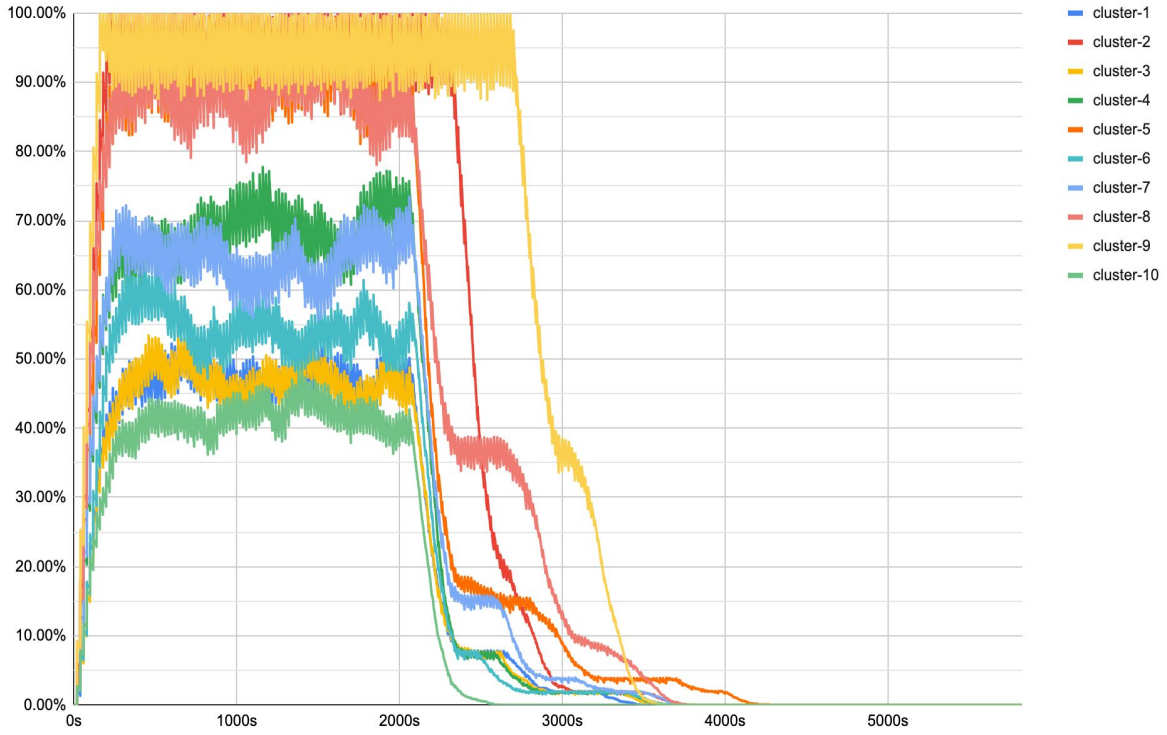
v2



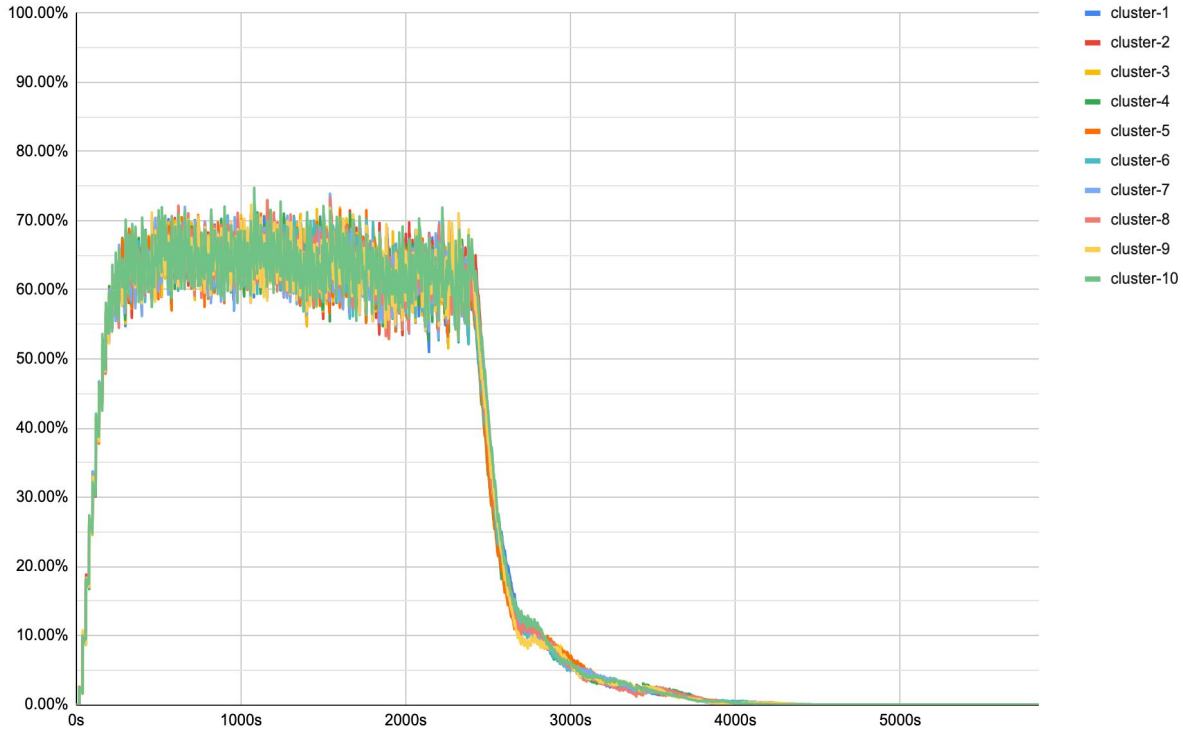
Total fragmented capacity = 0
Sum of reservation in respools = sum of cluster capacity

RP = Resource Pool
W = Workload

Federation: **#2 Uniform Cluster Usage**



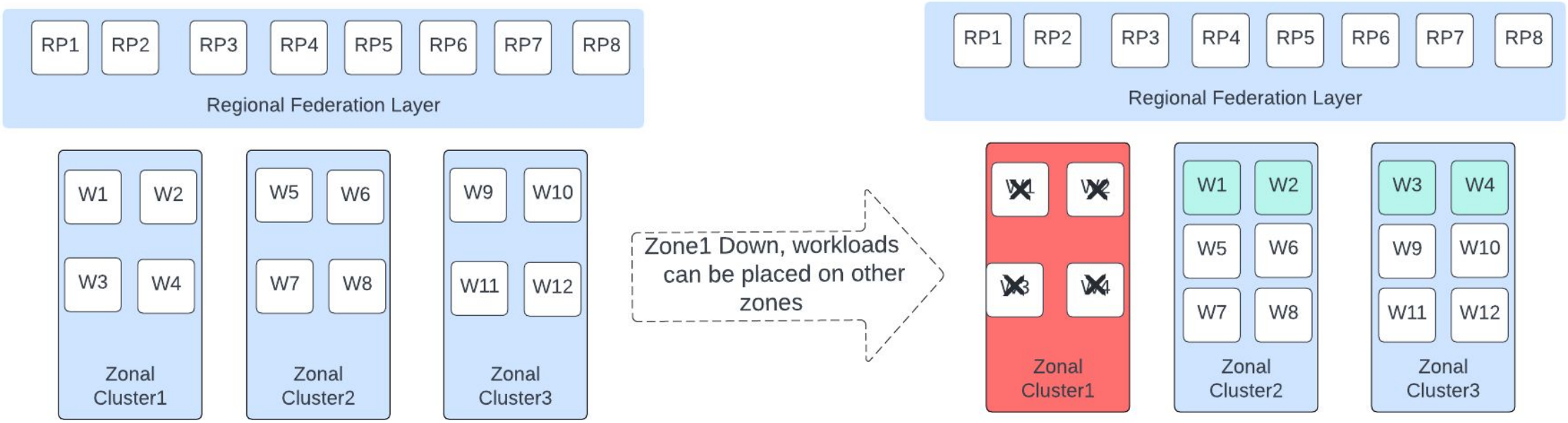
Cluster usage without Federation over time



Cluster usage with Federation over time

- Room to reduce overall cluster size
- 3x reduction in P95 task scheduling time

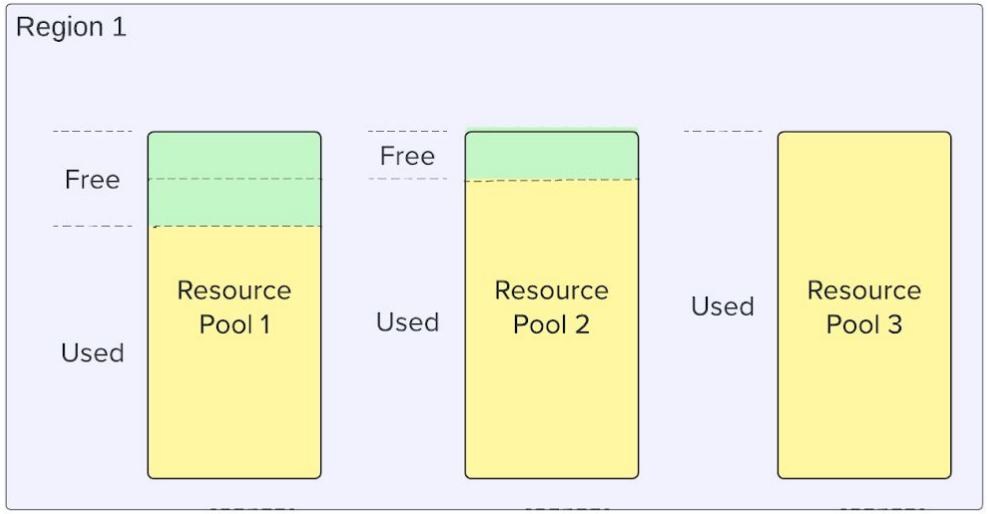
Regional Resource Management: **#3 Regional Availability**



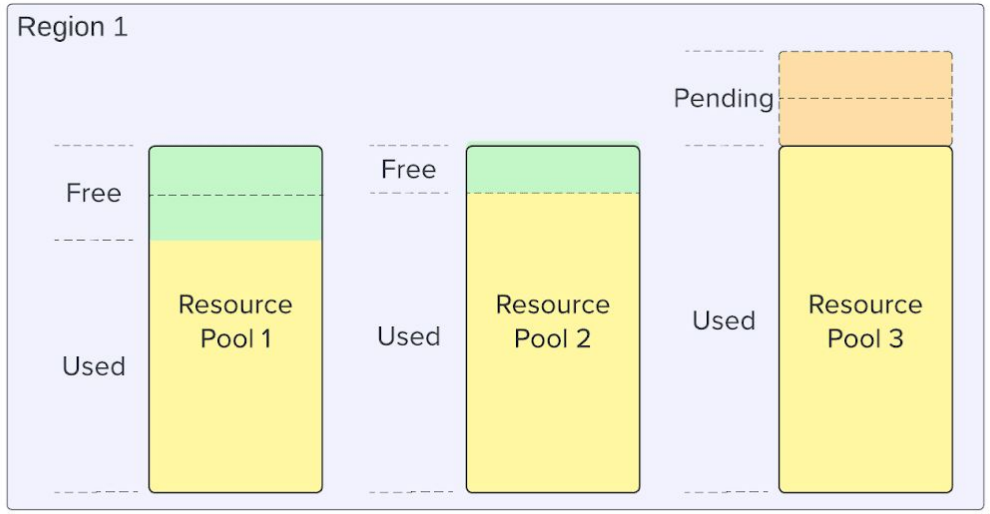
Federation: **#4 Cluster Management**

- No more coordination required to turn (up/down) cluster
- No more cluster selection on the client side
- Easier to release and upgrade

Elastic Resource Sharing: **Resource Borrowing**

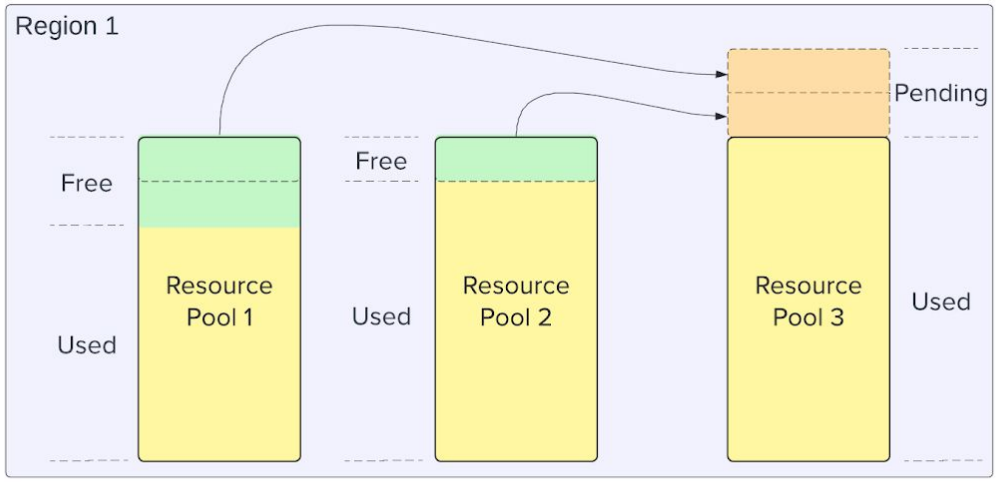


Initial state

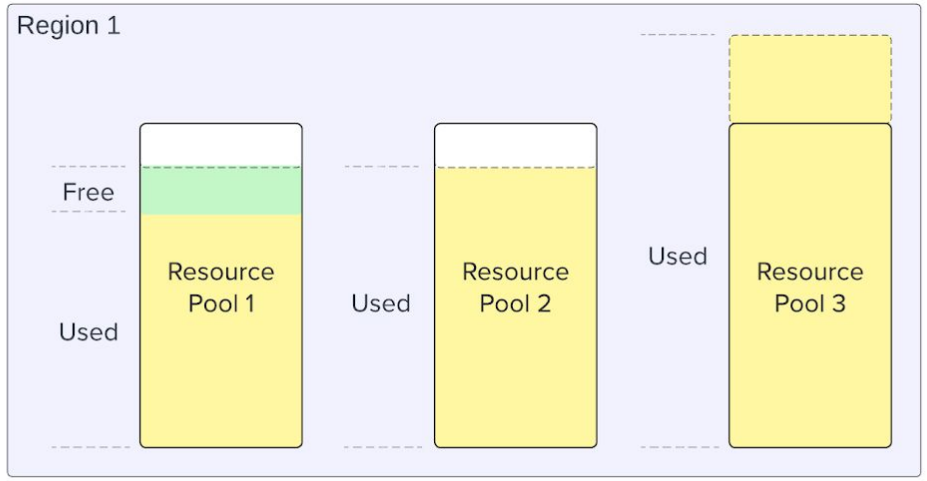


Workload pending on resource pool 3

Elastic Resource Sharing: **Resource Borrowing**

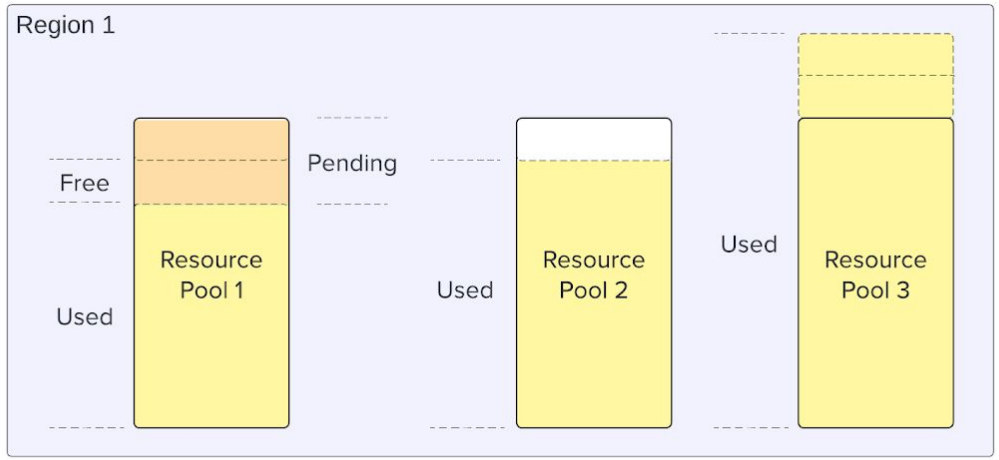


Borrow resources from 1 and 2

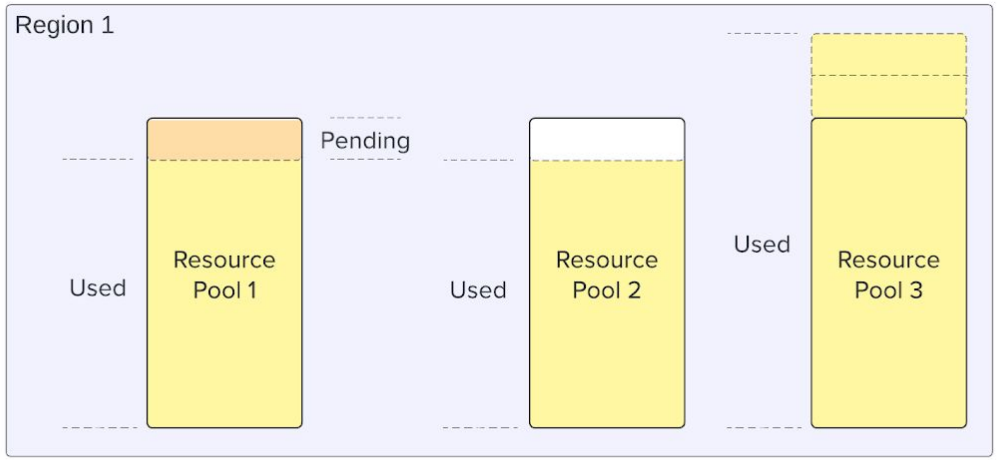


Workload running using borrowed resources

Elastic Resource Sharing: **Preemption**

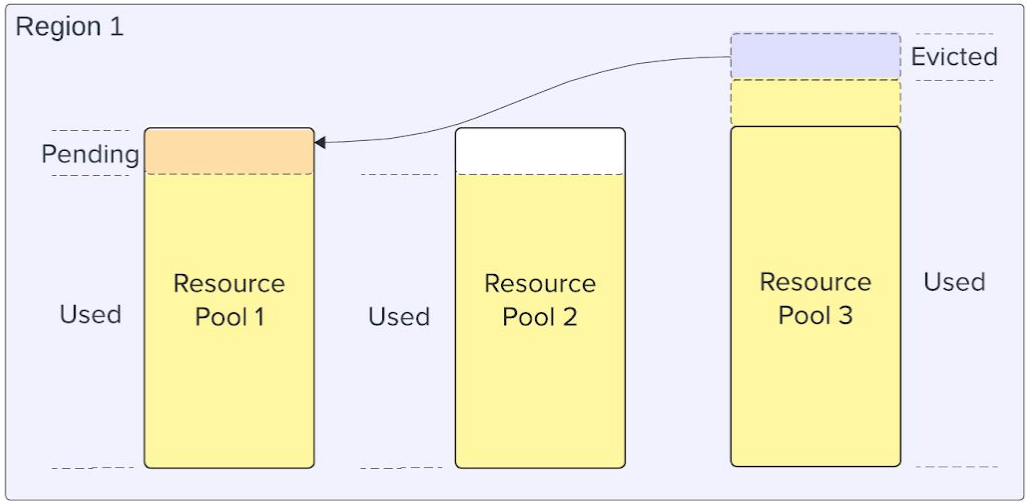


Workload pending on resource pool 1

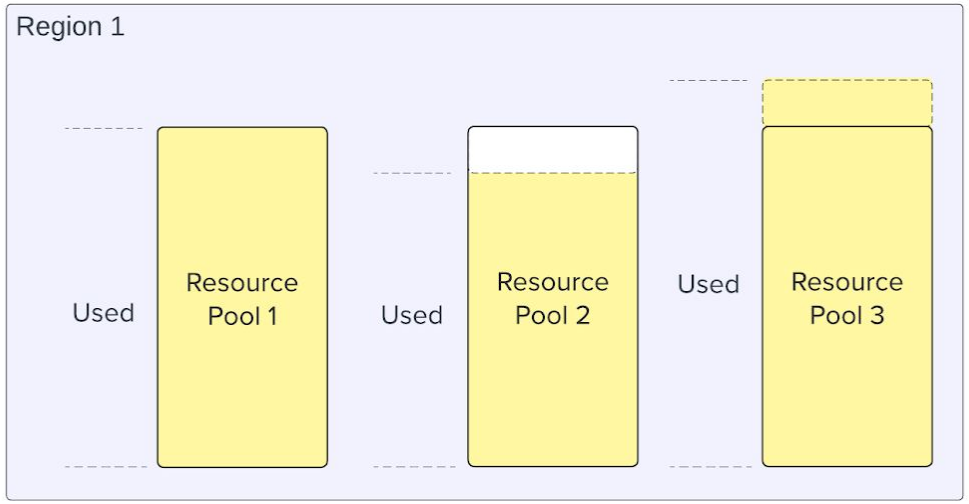


Some workload start to run on resource pool 1

Elastic Resource Sharing: **Preemption**

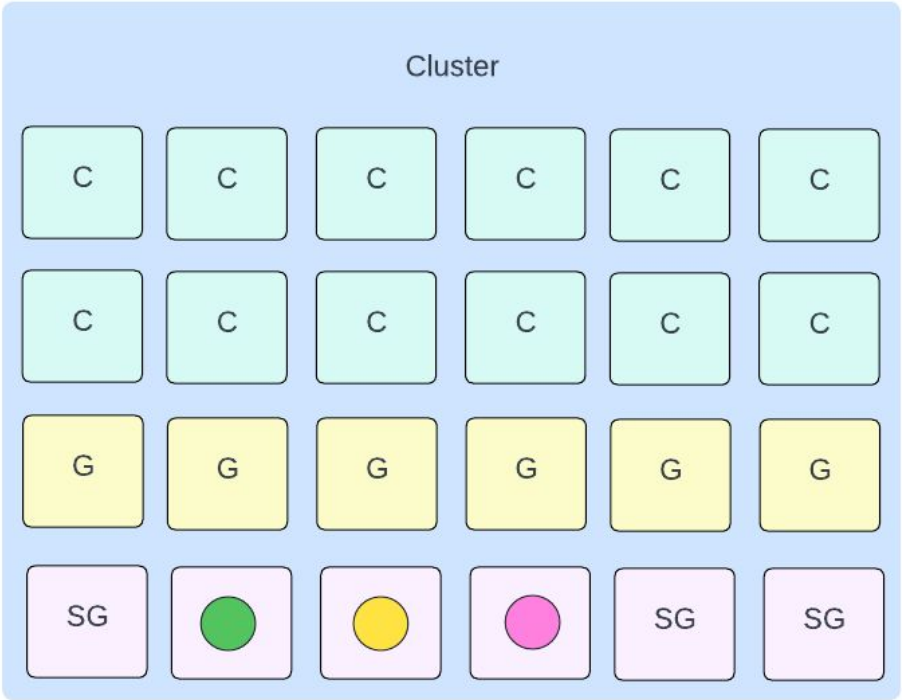


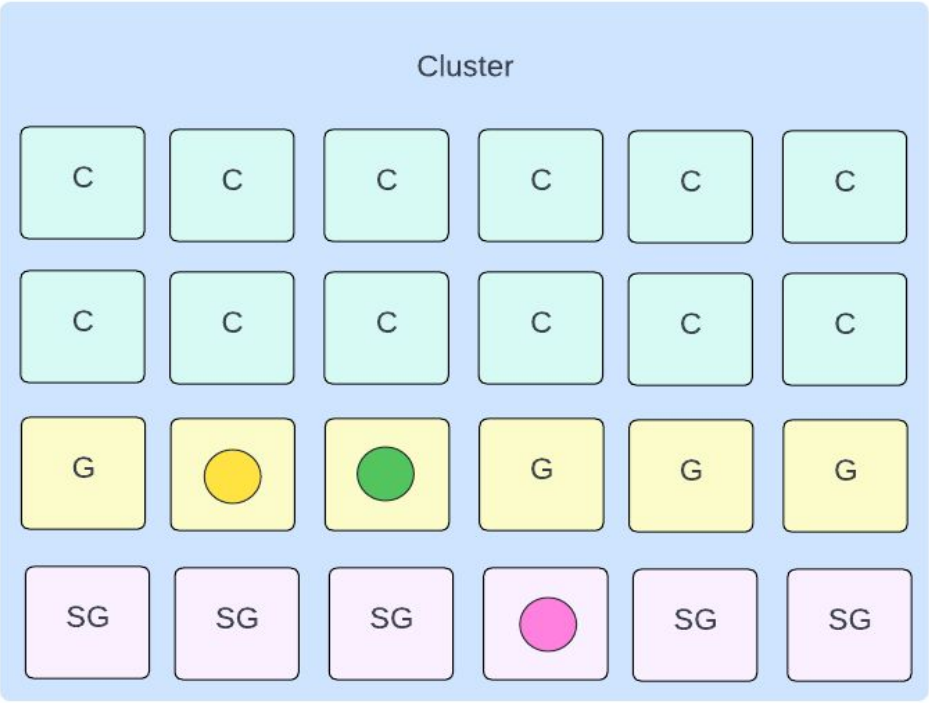
Workload evicted in resource pool 3



Workload running in resource pool 1

Specialized Hardware Efficiency





Example 1:

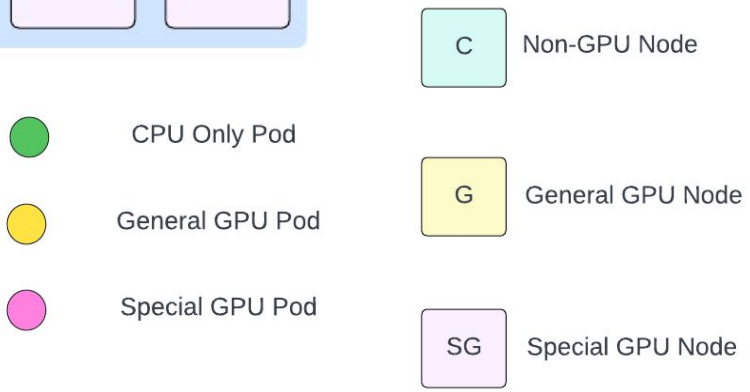
- CPU only Pod placed on Special GPU node
- General GPU Pod placed on Special GPU node

Not a Correct Behaviour

Example 2:

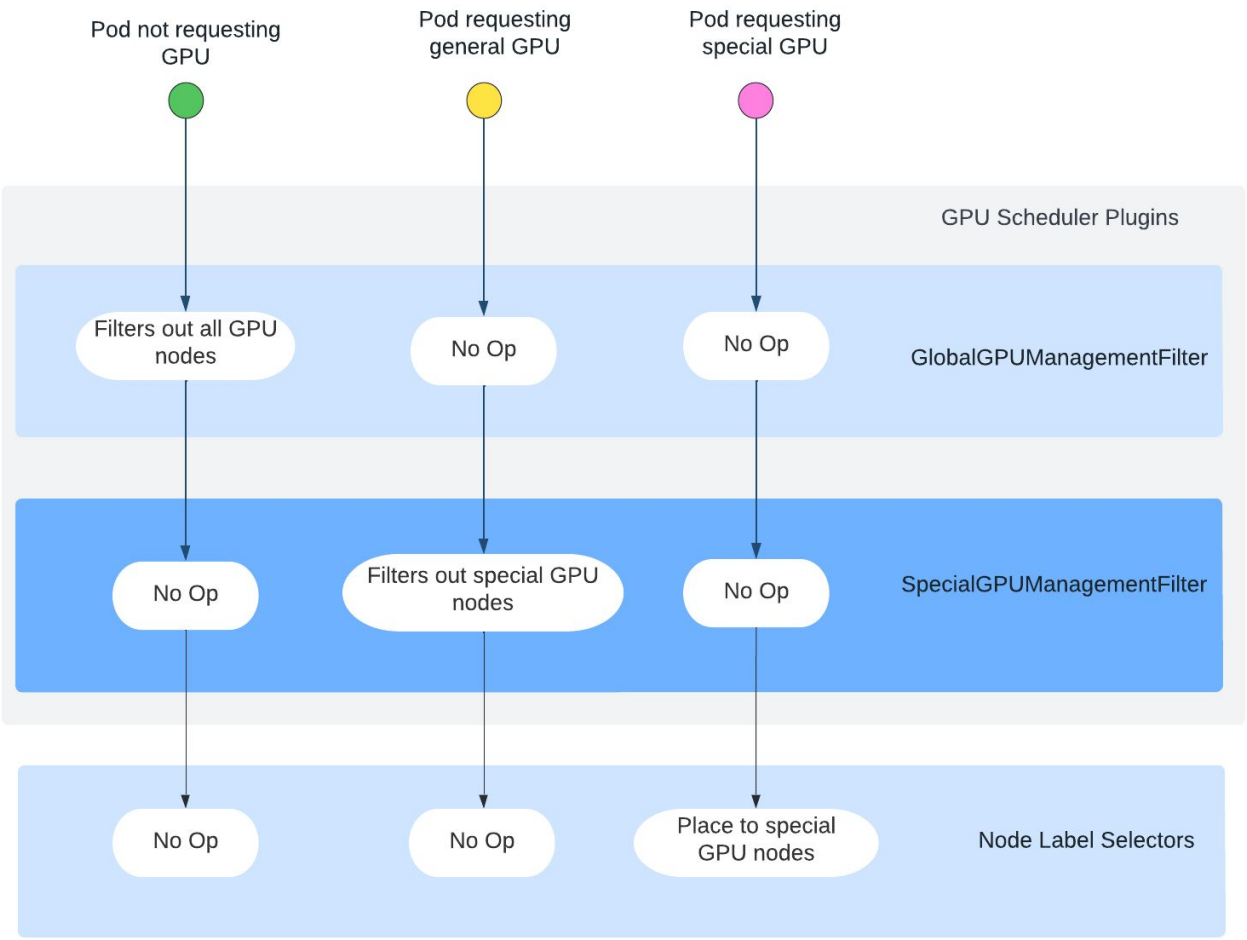
- CPU only Pod placed on General GPU node

Not a Correct Behaviour

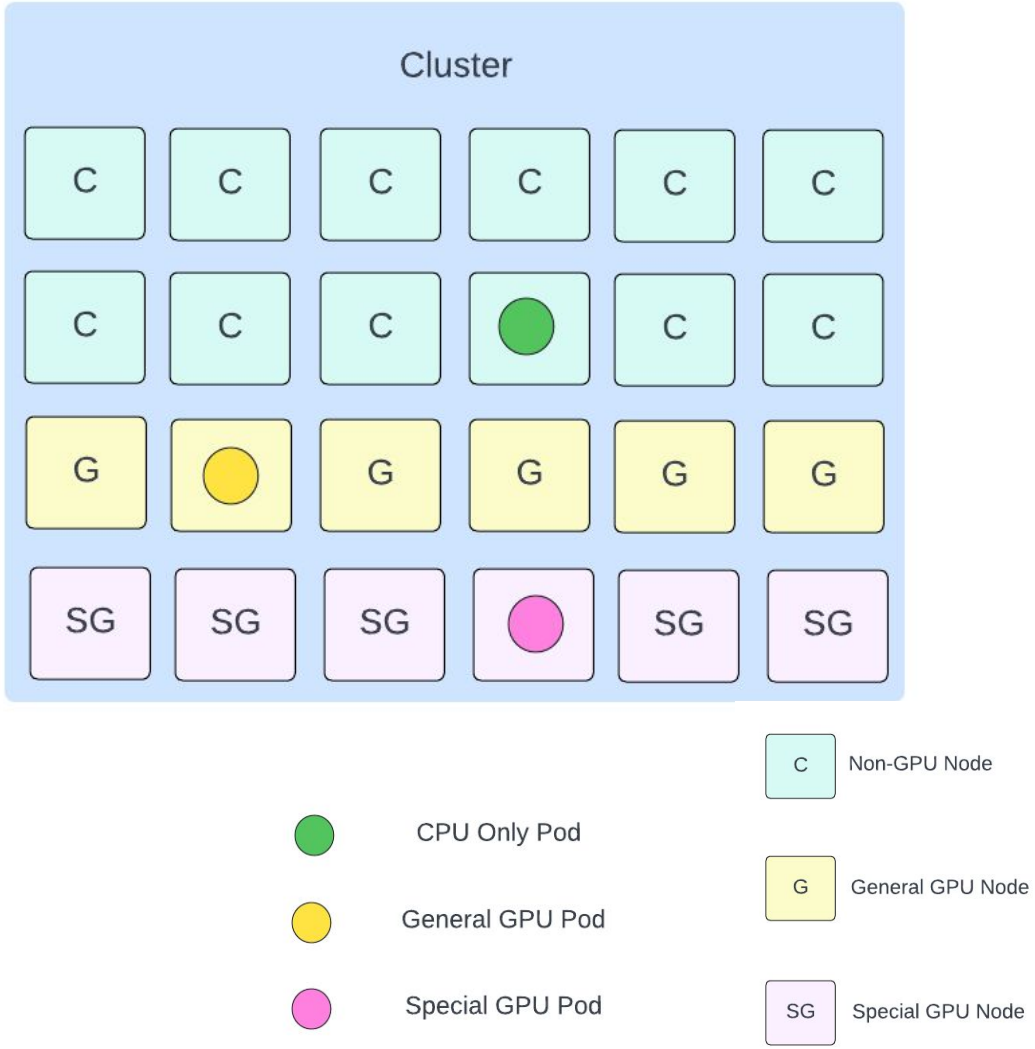


Efficient Resource Utilization for Batch Compute on Kubernetes

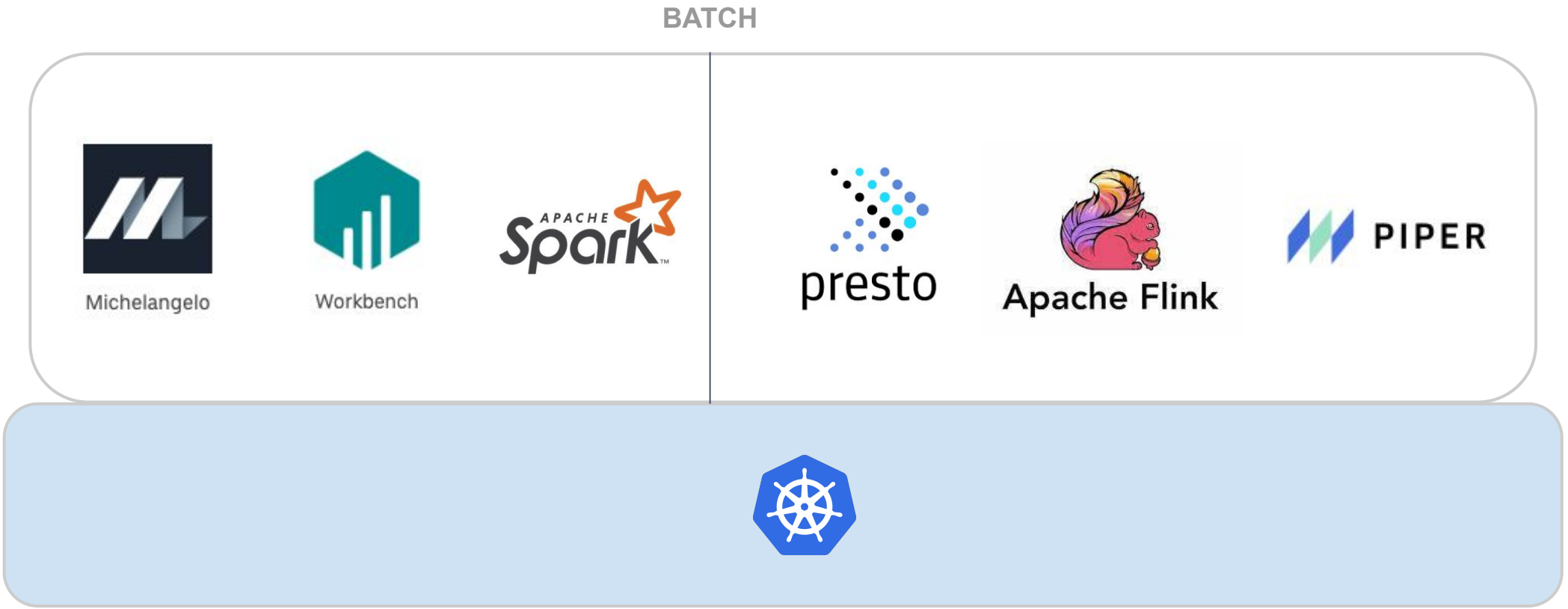
Specialized Hardware Efficiency



Desired Behaviour



Future Work


BATCH
Michelangelo
Workbench
APACHE Spark
presto
Apache Flink
PIPER

Thank You !

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