QUOTA ISN'T ENOUGH: TOPOLOGY AWARE SCHEDULING ON MODERN GPU CLUSTERS

JJ Jordan, Member of Technical Staff, Cohere
Amy Chen, Software Engineer, CoreWeave

01

THE PROBLEM

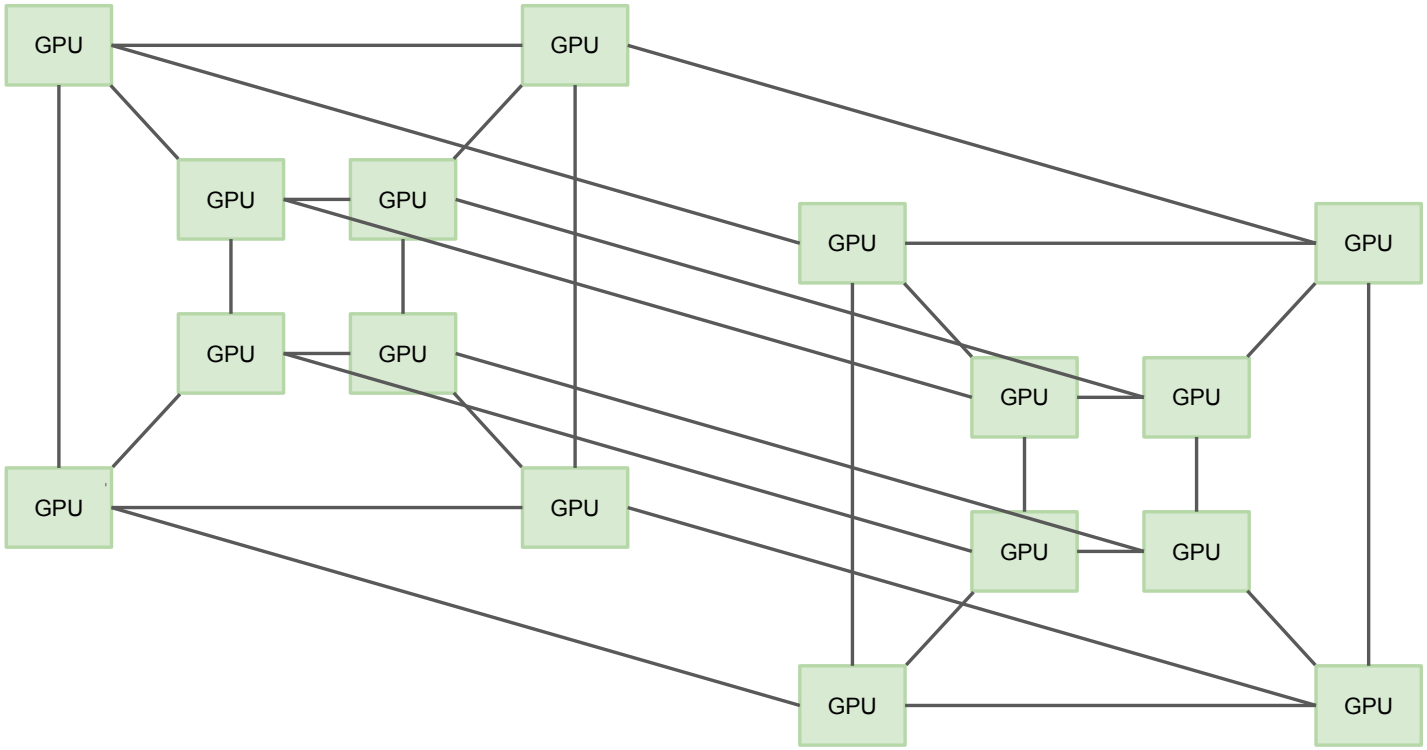
JJ

NVL72 GRACE/BLACKWELL COMMUNICATION HIERARCHY

Tier	Unidirectional Bandwidth (bytes/s)	Relative Speed	Scheduler Context
HBM3e Memory	8,000 GB/s	160x	On-Package GPU
NVLink 5	**900 GB/s**	**18x**	**Intra-Rack Copper Backplane**
NVLink-C2C	450 GB/s	9x	Host CPU/GPU Comms
PCIe Gen5 P2P	64 GB/s	1.3x	Intra-host if no NVLink
InfiniBand	**50 GB/s**	**1x**	**Inter-Rack Fabric**

JJ

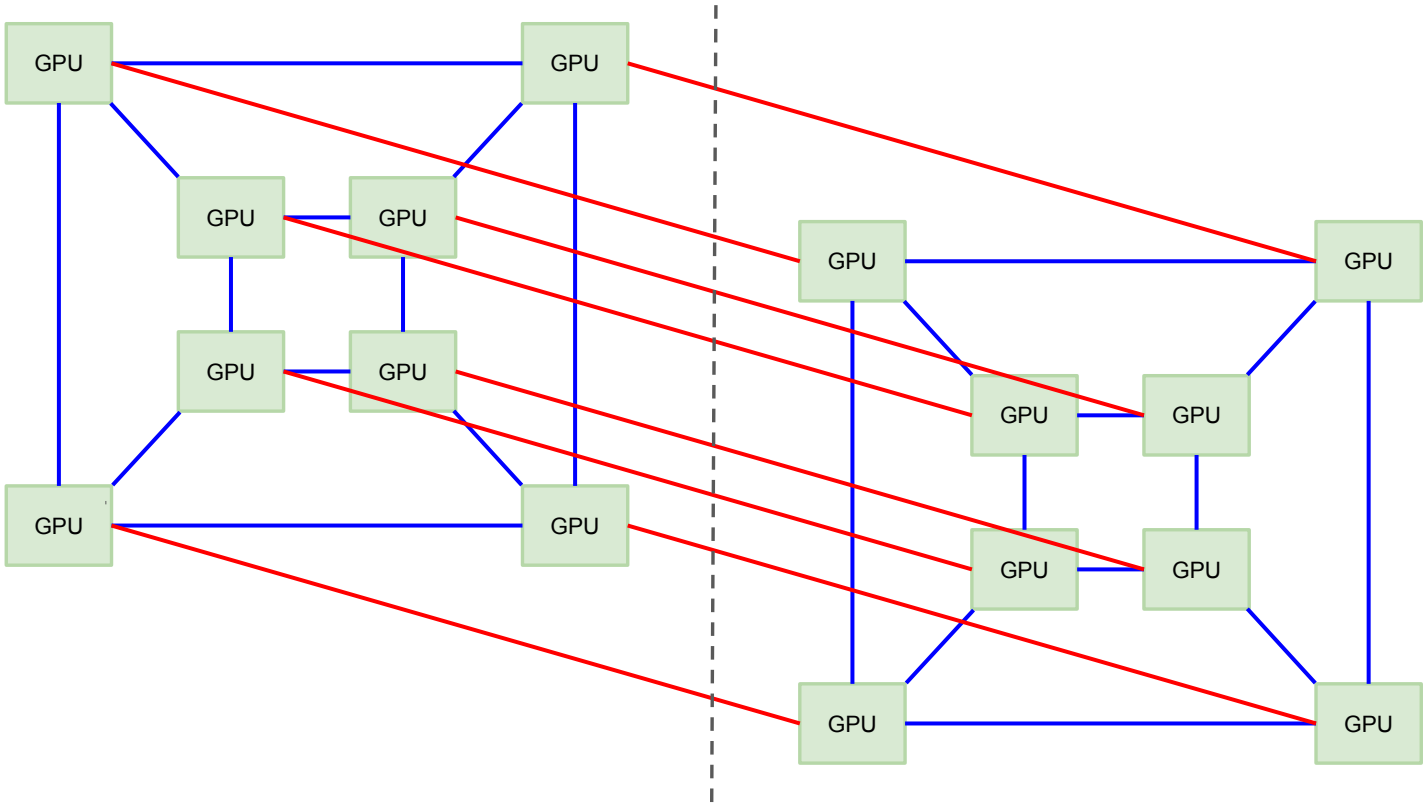
COLLECTIVE OPERATIONS: ALL-REDUCE



16 GPUs.
Each line is a pair that exchanges data.

JJ

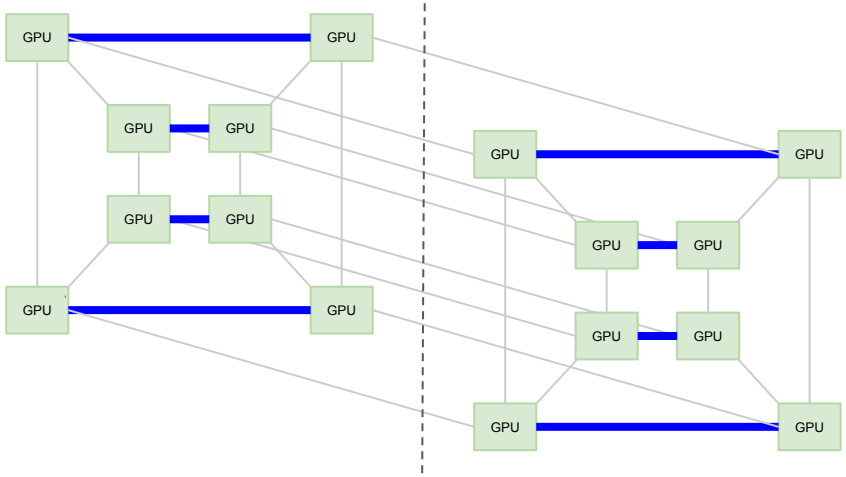
COLLECTIVE ACROSS TWO RACKS

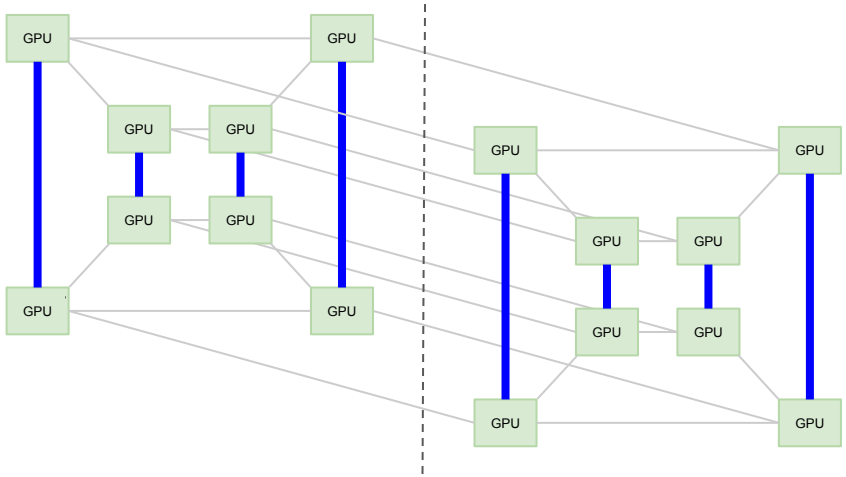


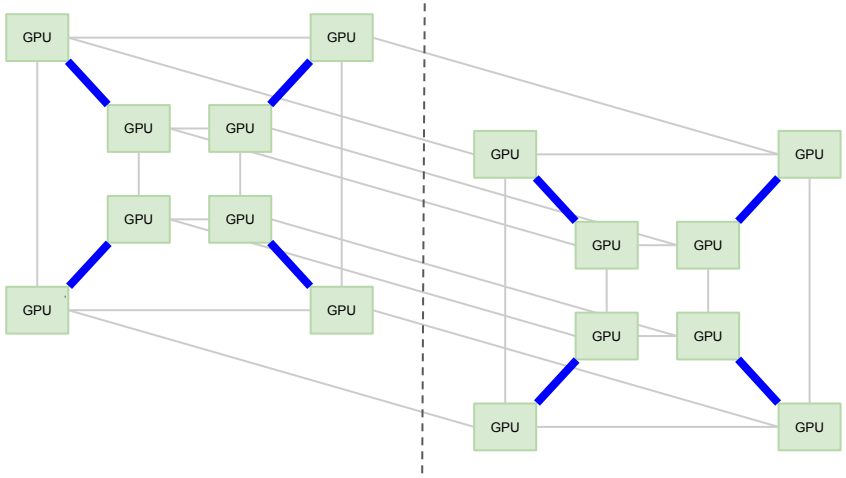
Left of the line is one NVLink domain. Right is another. Any pair that crosses it is on the network.

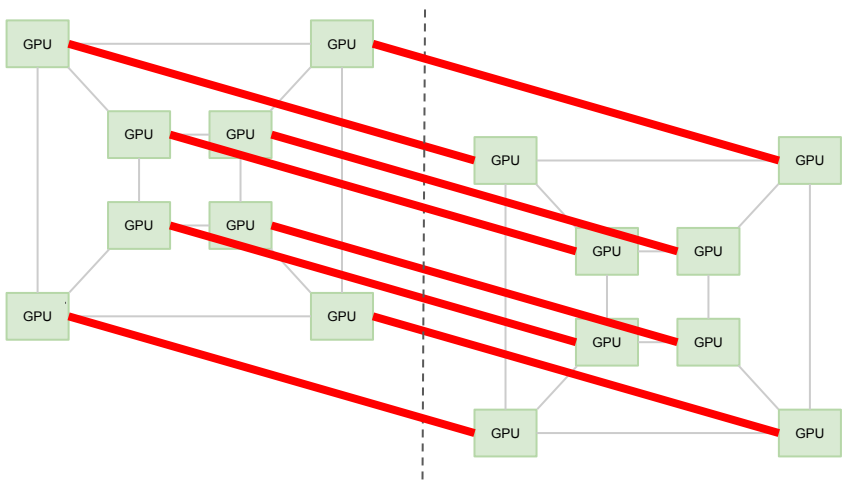
JJ

BALANCED COLLECTIVE ROUNDS





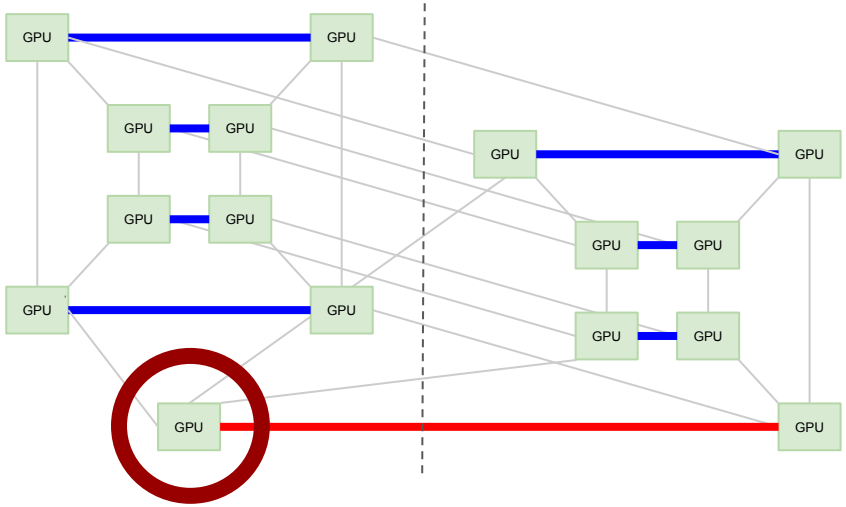


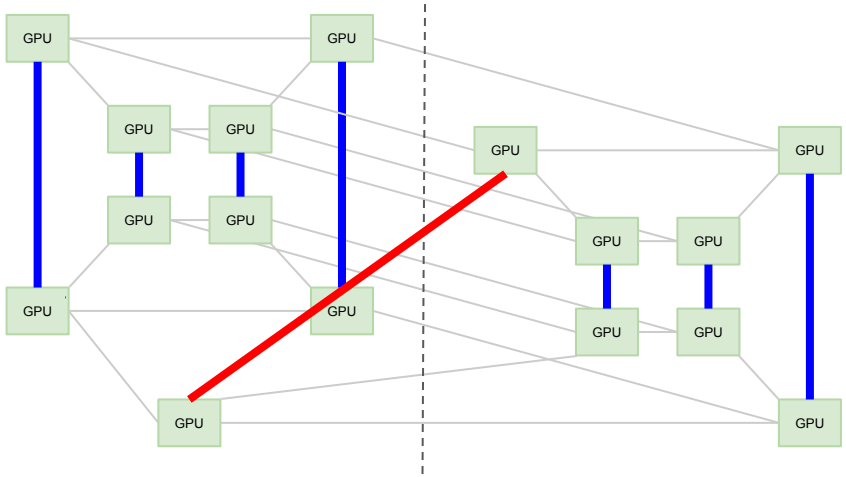


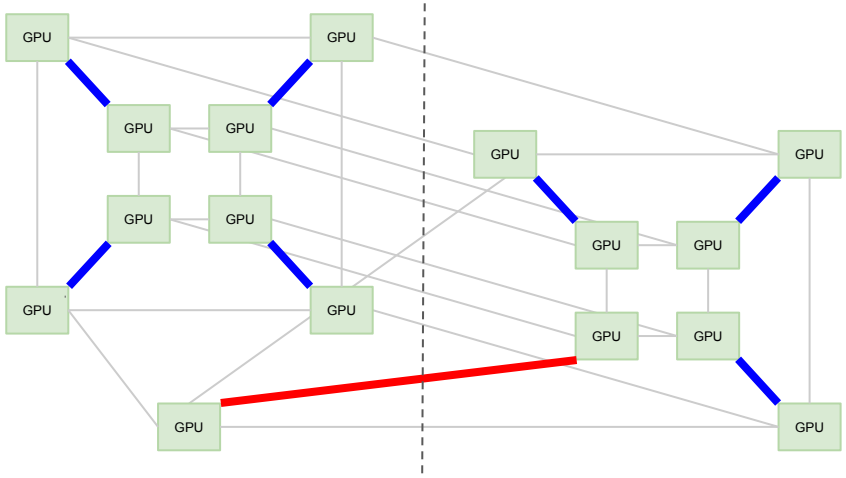
Rounds 1-3 all on NVLink. Only round 4 crosses racks. One slow round out of four.

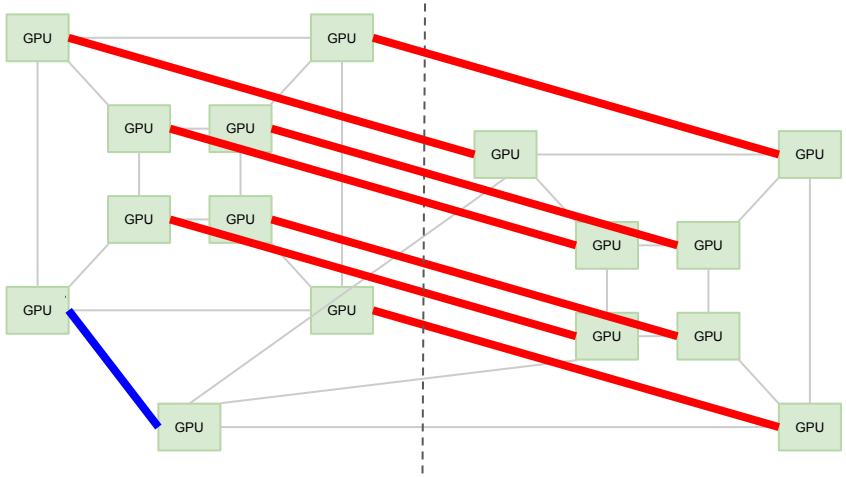
JJ

UNBALANCED COLLECTIVE ROUNDS









One GPU on the wrong rack. Now, every round must cross the network.

AMY

THIS CLASS OF HARDWARE IS DIFFERENT

72

GPUs in one NVLink domain. The rack is the unit of compute

$$$

Expensive per-rack cost for GB200 on-demand for 24 hours. **NOT reserved.**

18×

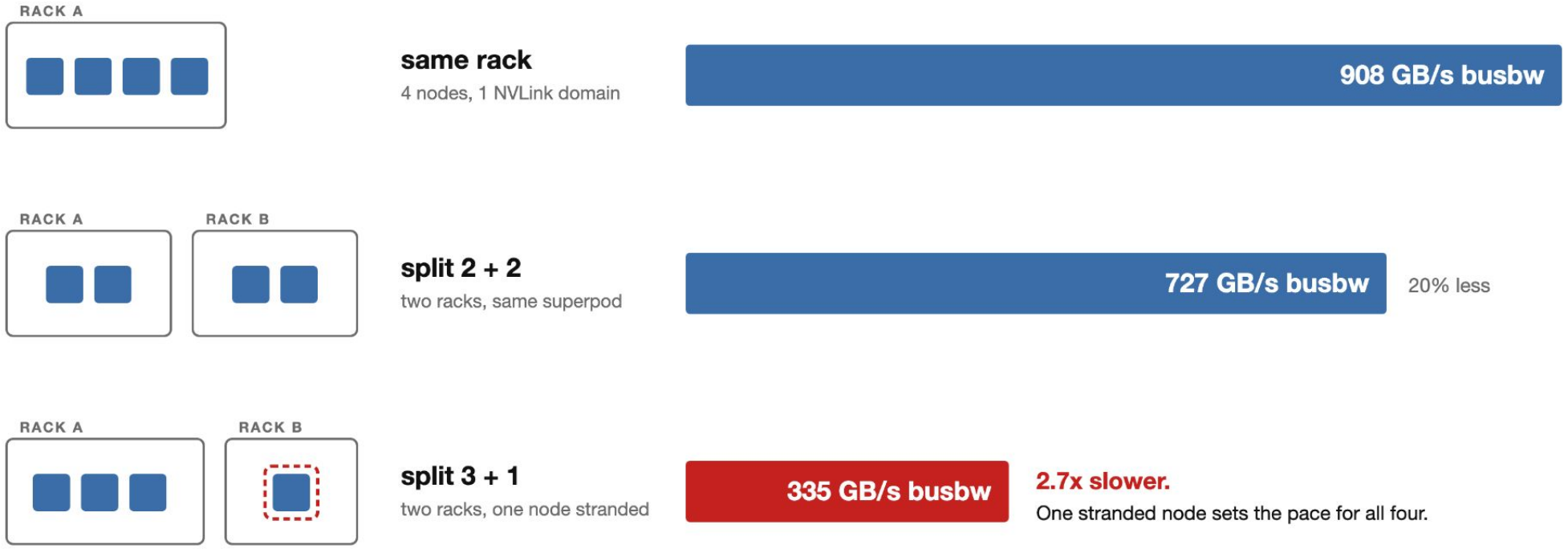
NVLink vs. InfiniBand bandwidth per GPU. 900 GB/s vs. 50 GB/s. In comparison, H100 network cliff is 4.5x

MOTIVATION

JJ

PLACEMENT IS WORTH REAL MONEY

SAME FOUR NODES, SAME NCCL ALL-REDUCE TEST. ONLY PLACEMENT CHANGED.



NCCL all-reduce bus bandwidth at 8 GB message size. GB200 NVL72, 4 GPUs per node. Rack boundary crossed over InfiniBand.

JJ

THE AWKWARD ARITHMETIC

2^n vs. 72

- ML jobs want powers of two (2, 4, 8, 16, 32, 64); the rack gives you 72
- 64 becomes the practical unit for large jobs
- 8-GPU left over in every rack

MOTIVATION

TAS IN 90 SECONDS

JOB

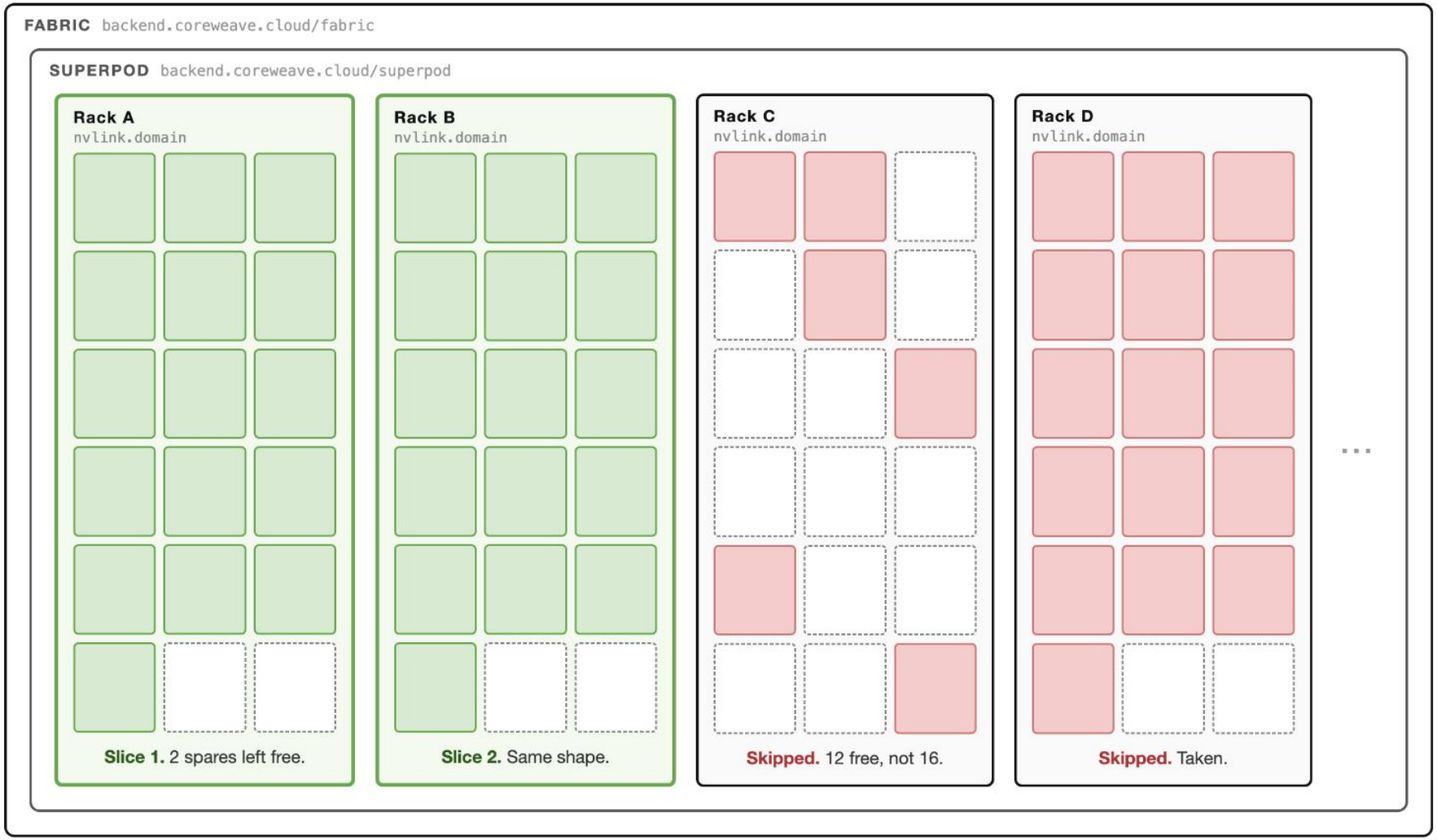
128 GPUs, 32 nodes
stay inside the **NVLink domain**
in **slices of 64**

Slice 1	Slice 2
16 nodes	16 nodes

↓

KUEUE ASKS, AT ADMISSION

Enough quota? Yes.
Two racks with 16 free nodes? Yes.
Pin every pod to a rack and a node.



this job · other jobs · free

One square = one node, 4 GPUs. One rack = 18 nodes = one NVLink domain. All 32 nodes decided at once, before any pod exists.

AMY

TAS IN 90 SECONDS

ADMIN
set up once per cluster

TOPOLOGY the shape of the hardware

```
kind: Topology
metadata.name: multinode-nvlink-ib
spec.levels:  # coarse to fine
  - nodeLabel: backend.coreweave.cloud/fabric
  - nodeLabel: backend.coreweave.cloud/superpod
  - nodeLabel: backend.coreweave.cloud/leafgroup
  - nodeLabel: ds.coreweave.com/nvlink.domain
  - nodeLabel: kubernetes.io/hostname
```

→

RESOURCEFLAVOR which nodes, which shape

```
kind: ResourceFlavor
metadata.name: gb200
spec.nodeLabels:
  node.kubernetes.io/instance-type:
    gb200-4x
spec.topologyName: multinode-nvlink-ib
```

→

CLUSTERQUEUE how much

```
kind: ClusterQueue
spec.resourceGroups[0].flavors:
  - name: gb200
    resources:
      - name: nvidia.com/gpu
        nominalQuota: 1024
        # 16 racks x 64
```

USER
one annotation on the pod template

JOB what the job needs from the shape

```
spec.template.metadata.annotations:
  kueue.x-k8s.io/podset-required-topology:
    ds.coreweave.com/nvlink.domain

# that's it. no nodeSelector, no affinity rules.
```

THE KNOBS

```
podset-required-topology      one domain, or wait
podset-preferred-topology     one domain if you can, spill if not
podset-unconstrained-topology  anywhere, fill the gaps
podset-slice-required-topology
  + podset-slice-size          64 and 64, not 72 and 56
```

KUEUE
at admission, before any pod exists

1. FITS QUOTA?
Does the ClusterQueue have 128 GPUs of gb200 to give?

→

2. FITS THE SHAPE?
Walk the topology tree. Which racks have room for 64? Which nodes inside them?

→

3. PIN IT
Write the rack and node assignment into the Workload. Pods get a nodeSelector. kube-scheduler just obeys.

Quota answers "how much." Topology answers "where." Both are decided at admission, for the whole job at once.

MOTIVATION

AMY

TAS IN 90 SECONDS

ADMIN
set up once per cluster

TOPOLOGY the shape of the hardware

```
kind: Topology
metadata.name: multinode-nvlink-ib
spec.levels:  # coarse to fine
  - nodeLabel: backend.coreweave.cloud/fabric
  - nodeLabel: backend.coreweave.cloud/superpod
  - nodeLabel: backend.coreweave.cloud/leafgroup
  - nodeLabel: ds.coreweave.com/nvlink.domain
  - nodeLabel: kubernetes.io/hostname
```

→

RESOURCEFLAVOR which nodes, which shape

```
kind: ResourceFlavor
metadata.name: gb200
spec.nodeLabels:
  node.kubernetes.io/instance-type:
    gb200-4x
spec.topologyName: multinode-nvlink-ib
```

→

CLUSTERQUEUE how much

```
kind: ClusterQueue
spec.resourceGroups[0].flavors:
  - name: gb200
    resources:
      - name: nvidia.com/gpu
        nominalQuota: 1024
        # 16 racks x 64
```

USER
one annotation on the pod template

JOB what the job needs from the shape

```
spec.template.metadata.annotations:
  kueue.x-k8s.io/podset-required-topology:
    ds.coreweave.com/nvlink.domain

# that's it. no nodeSelector, no affinity rules.
```

THE KNOBS

```
podset-required-topology      one domain, or wait
podset-preferred-topology     one domain if you can, spill if not
podset-unconstrained-topology  anywhere, fill the gaps
podset-slice-required-topology
  + podset-slice-size          64 and 64, not 72 and 56
```

KUEUE
at admission, before any pod exists

1. FITS QUOTA?
Does the ClusterQueue have 128 GPUs of gb200 to give?

→

2. FITS THE SHAPE?
Walk the topology tree. Which racks have room for 64? Which nodes inside them?

→

3. PIN IT
Write the rack and node assignment into the Workload. Pods get a nodeSelector. kube-scheduler just obeys.

Quota answers "how much." Topology answers "where." Both are decided at admission, for the whole job at once.

MOTIVATION

AMY

TAS IN 90 SECONDS

ADMIN
set up once per cluster

TOPOLOGY the shape of the hardware

```
kind: Topology
metadata.name: multinode-nvlink-ib
spec.levels:  # coarse to fine
  - nodeLabel: backend.coreweave.cloud/fabric
  - nodeLabel: backend.coreweave.cloud/superpod
  - nodeLabel: backend.coreweave.cloud/leafgroup
  - nodeLabel: ds.coreweave.com/nvlink.domain
  - nodeLabel: kubernetes.io/hostname
```

→

RESOURCEFLAVOR which nodes, which shape

```
kind: ResourceFlavor
metadata.name: gb200
spec.nodeLabels:
  node.kubernetes.io/instance-type:
    gb200-4x
spec.topologyName: multinode-nvlink-ib
```

→

CLUSTERQUEUE how much

```
kind: ClusterQueue
spec.resourceGroups[0].flavors:
  - name: gb200
    resources:
      - name: nvidia.com/gpu
        nominalQuota: 1024
        # 16 racks x 64
```

USER
one annotation on the pod template

JOB what the job needs from the shape

```
spec.template.metadata.annotations:
  kueue.x-k8s.io/podset-required-topology:
    ds.coreweave.com/nvlink.domain

# that's it. no nodeSelector, no affinity rules.
```

THE KNOBS

```
podset-required-topology      one domain, or wait
podset-preferred-topology     one domain if you can, spill if not
podset-unconstrained-topology  anywhere, fill the gaps
podset-slice-required-topology
  + podset-slice-size          64 and 64, not 72 and 56
```

KUEUE
at admission, before any pod exists

1. FITS QUOTA?
Does the ClusterQueue have 128 GPUs of gb200 to give?

→

2. FITS THE SHAPE?
Walk the topology tree. Which racks have room for 64? Which nodes inside them?

→

3. PIN IT
Write the rack and node assignment into the Workload. Pods get a nodeSelector. kube-scheduler just obeys.

Quota answers "how much." Topology answers "where." Both are decided at admission, for the whole job at once.

02

IN PRODUCTION

AMY

1. TEST POD PLACEMENT WITHOUT BURNING GPUS

HPC-V TEST POD ON NODE-17

1 no `queue-name` label

2 annotation `podset-unconstrained-topology: "true"`

3 PriorityClass **-1**

WHAT KUEUE TAS SEES

node-17

4 of 4 GPU free

free | free | free | free

Looks like a TAS pod (knob 2), but no Workload (knob 1).
Its GPUs vanish from Kueue capacity accounting.

WHAT KUBE-SCHEDULER SEES

node-17

4 of 4 GPU used by one hpc-v pod, priority -1

hpc-v test pod
1 pod, all 4 GPUs

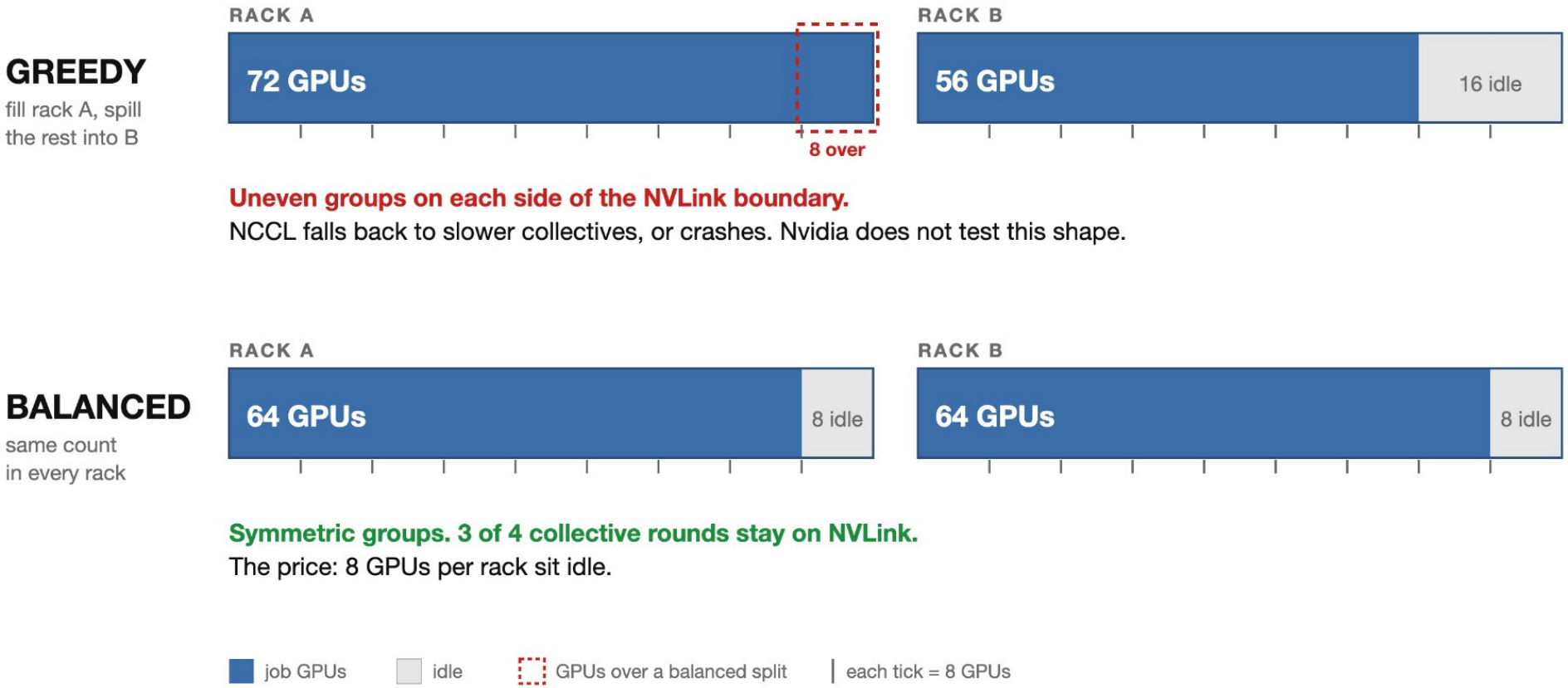
Real usage, lowest possible priority (knob 3).
Any customer pod placed here preempts it.

Kueue admits customer jobs onto node-17 as if it were empty.
kube-scheduler evicts the hpc-v pod. Kueue never learns it was there.

JJ

2. BALANCED vs. GREEDY PLACEMENT

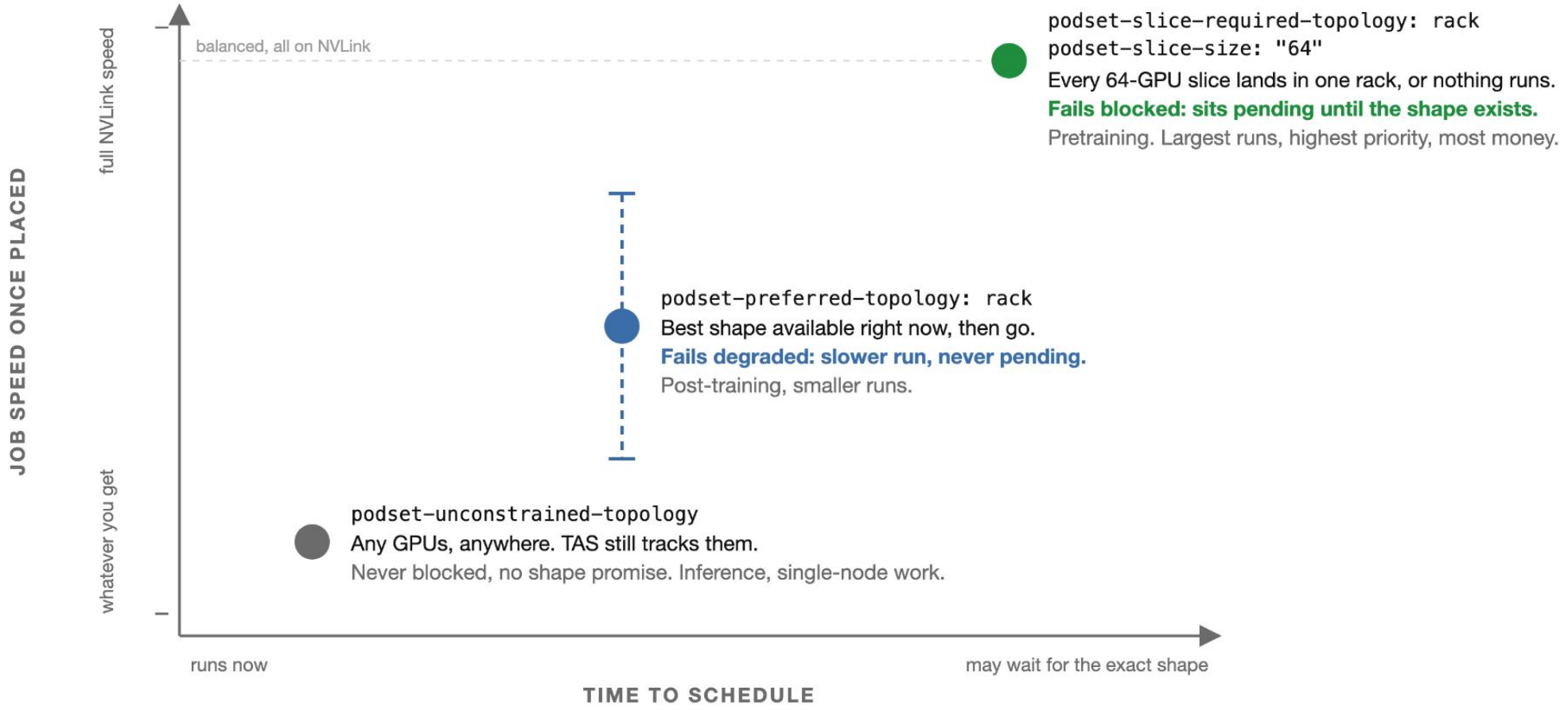
SAME 128-GPU JOB, TWO NVL72 RACKS, TWO WAYS TO PLACE IT



AMY

2. NETWORK SPEED IS A SCHEDULING REQUEST

THREE WAYS TO ASK KUEUE FOR A SHAPE, AND WHAT EACH ONE TRADES

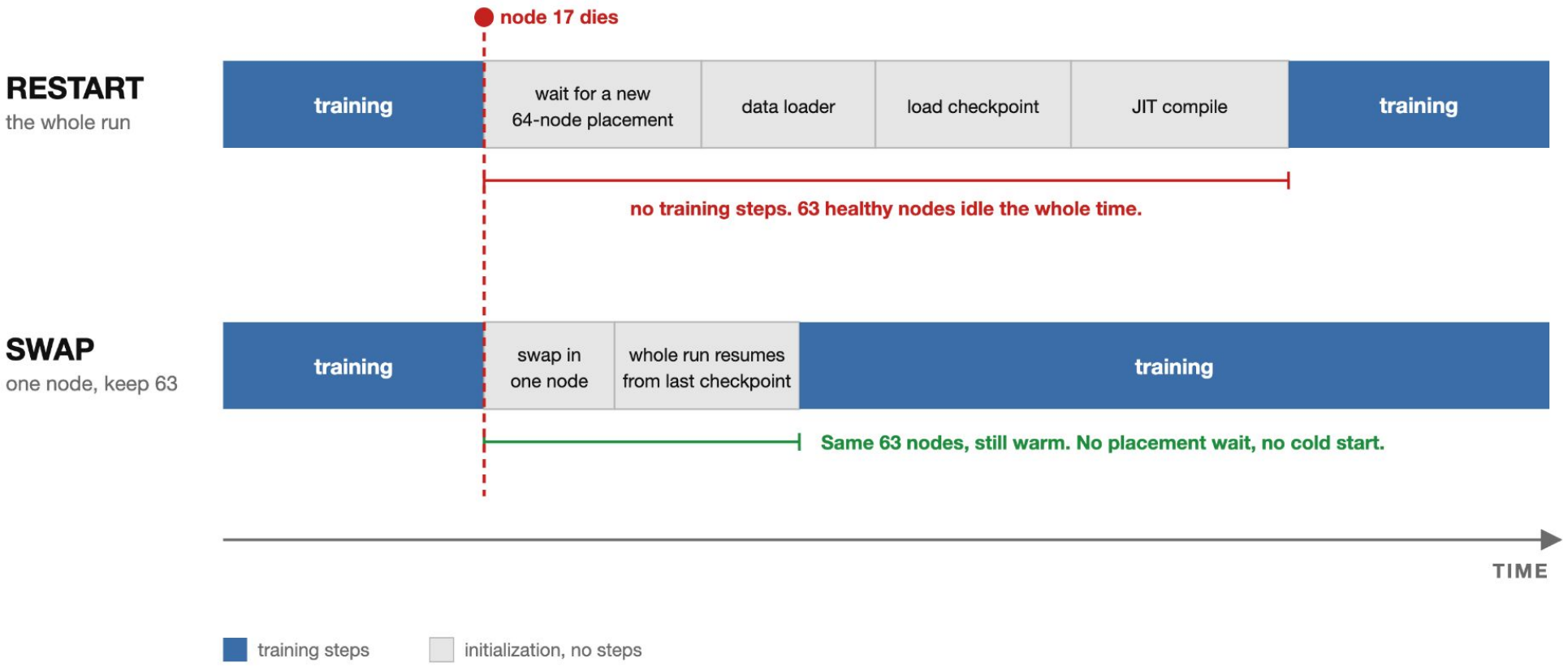


The real choice is the failure mode: degraded now, or blocked until it is right.

JJ

3. TAS × CHECKPOINTING AND NODE HOT SWAPS

A 64-NODE RUN. ONE NODE DIES. TWO WAYS BACK TO TRAINING.



Cordoning the bad node must not take the other 63 with it.

AMY

3. WHAT KUEUE DOES ON NODE HOTSWAP

THE WORKLOAD'S TOPOLOGY ASSIGNMENT WHEN NODE 17 FAILS

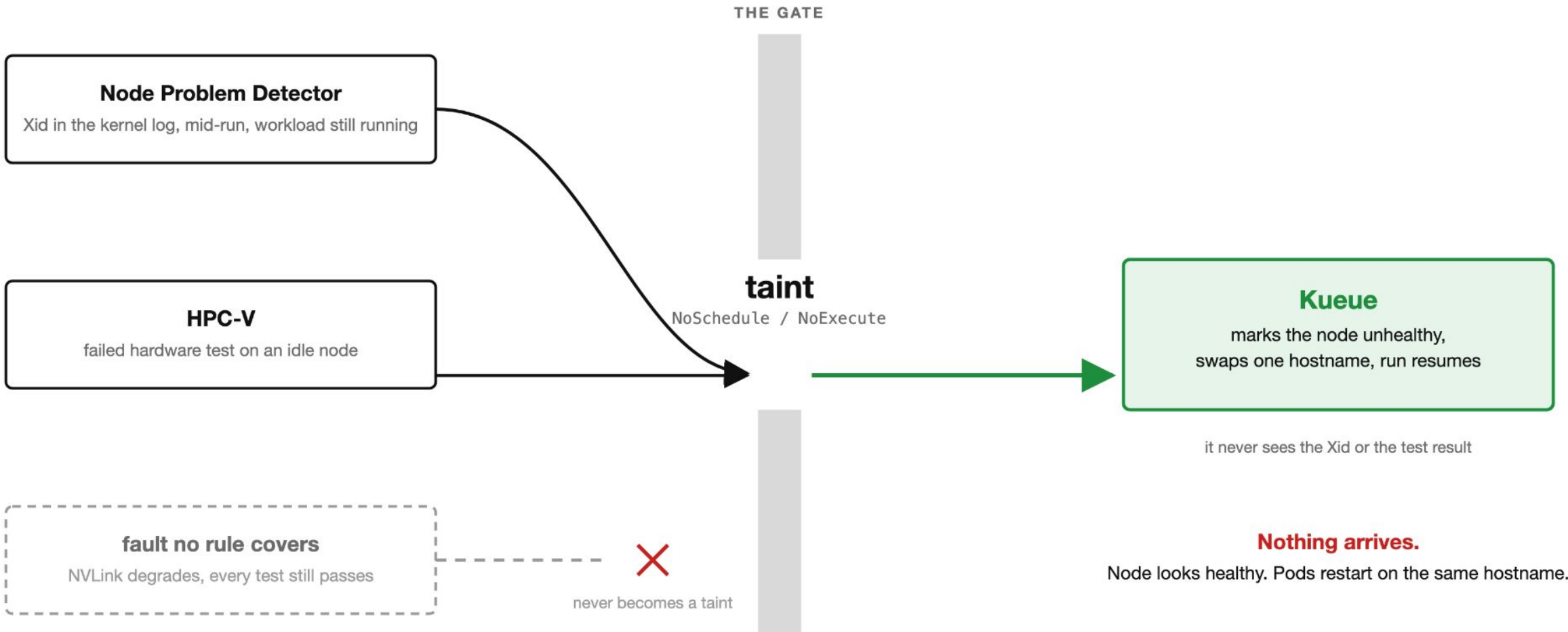
```
topologyAssignment.domains:

  gpu-node-01
  gpu-node-02
  gpu-node-03
- gpu-node-17      NotReady, cordoned
+ gpu-node-41      spare in the same rack
  gpu-node-18
  gpu-node-19
  ... 58 more
```

unchanged lines keep their pods exactly where they are

Kueue rewrites one entry. The other 63 stay put.

KUEUE LISTENS FOR ONE SIGNAL: THE TAINT



Every detector we run exists to produce this one taint. If a fault misses it, Kueue cannot help.

JJ

4. “WHY WON’T IT PREEMPT?”

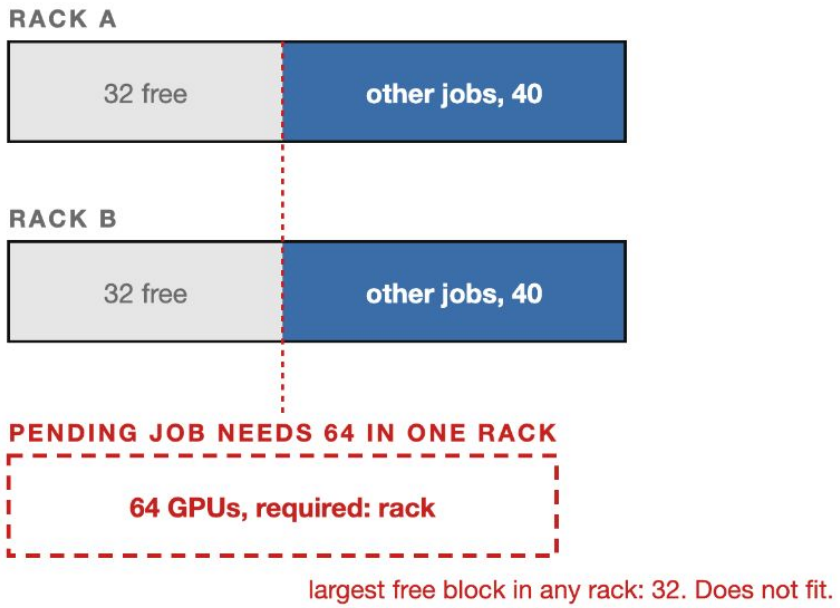
A 64-GPU JOB IS PENDING. TWO 32-GPU LOW-PRIORITY JOBS ARE RUNNING. WHY NO PREEMPTION?

WHAT THE MLE SEES

32 + 32
= 64

Two lower-priority jobs, together enough.
Preempt both, admit mine.

WHAT KUEUE SEES, AFTER EVICTING BOTH



largest free block in any rack: 32. Does not fit.

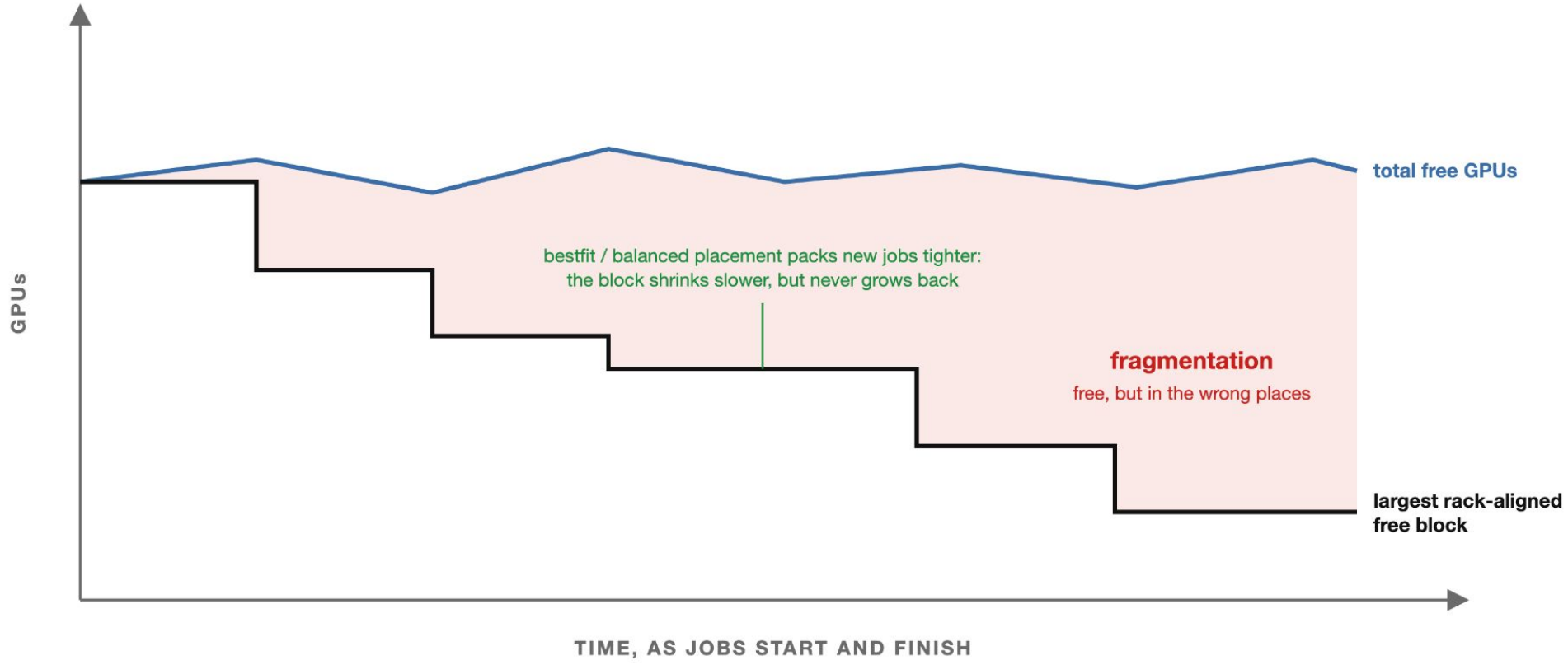
Preemption checks shape, not just count.

Kueue will not evict two jobs to end up exactly where it started.

AMY

5. FRAGMENTATION: MOSTLY UNSOLVED (for now)

FREE GPUS STAY HIGH. USABLE RACK-ALIGNED BLOCKS SHRINK.



Entropy only goes up. Every job that starts or finishes scatters free GPUs a little more.

Fewer racks can take a 64. More "why won't it preempt" tickets.

FRAGMENTATION IS A FUNCTION OF TIME, NOT OF THE FIRST PLACEMENT

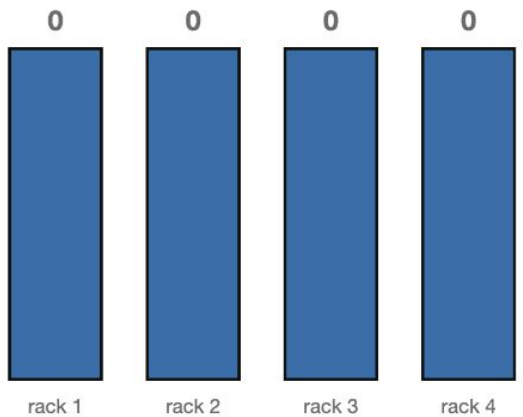
AMY

T0

Perfectly packed

four racks, every node busy

FREE GPUs PER RACK

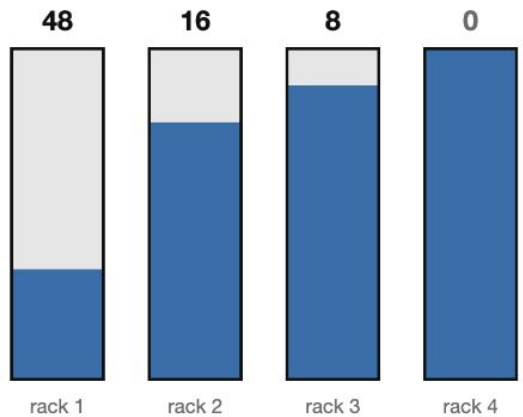


largest free block: **0** · total free: **0**

T1

Jobs finish, not in order

runs end when they end



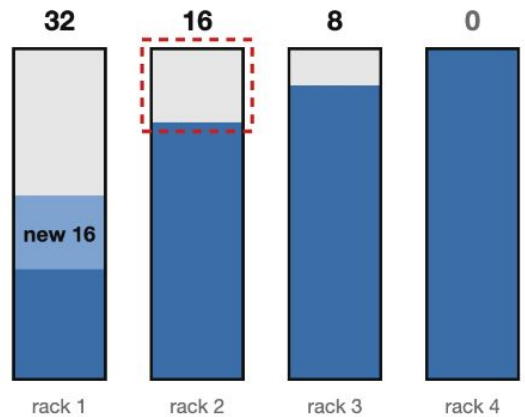
largest free block: **48** · total free: **72**

72 free, and no rack has 64.

T2

A 16-node pretraining job arrives

required rack, BestFit: emptiest rack first



largest free block: **32** · total free: **56**

Rack 2 fit it exactly. BestFit took rack 1 anyway.

preferred (post-training) would take the 16; inference fills the 8

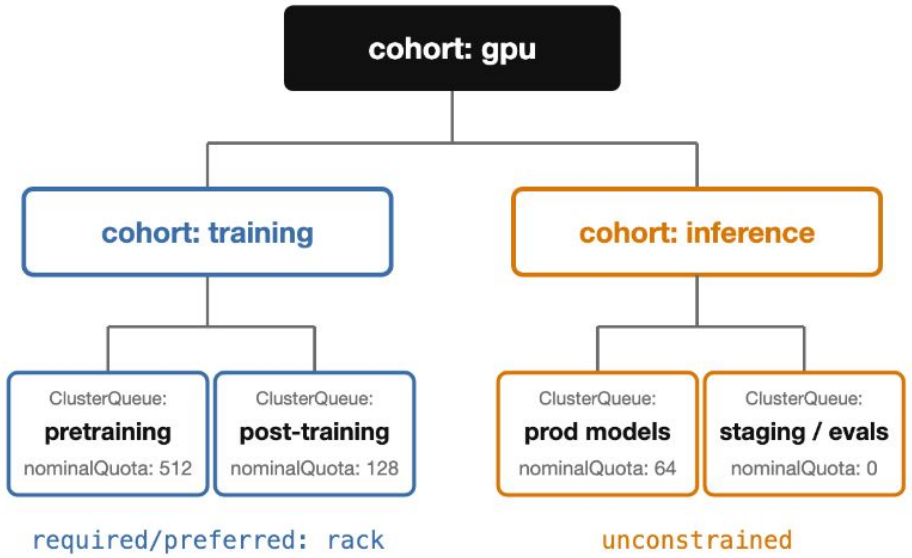
Total free never dropped below 56. A 64-GPU job has not fit since t1.

Departures are uneven. Required arrivals go to the emptiest rack. The exact-fit holes are what is left over.

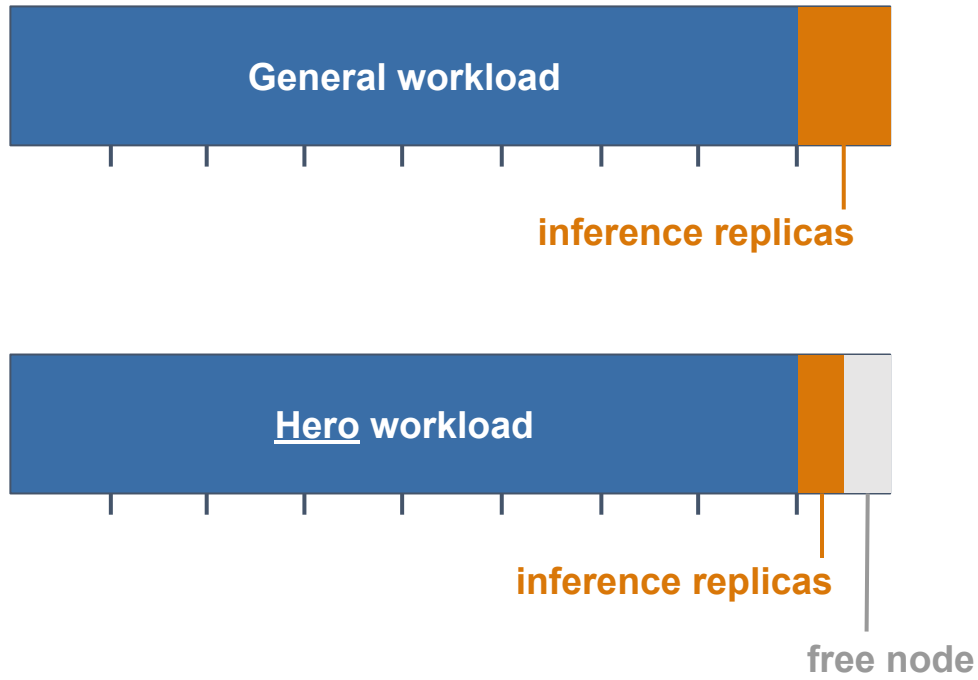
JJ

TRAINING AND INFERENCE IN ONE CLUSTER

QUOTA VIEW: HIERARCHICAL COHORTS



HARDWARE VIEW: ONE NVL72 RACK



Inference fills the gaps training cannot use.

HERO WORKLOAD SCHEDULING TODAY

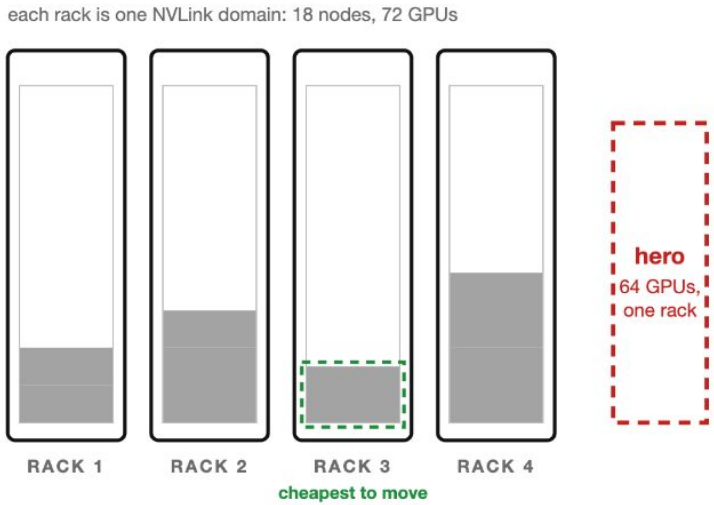
AMY

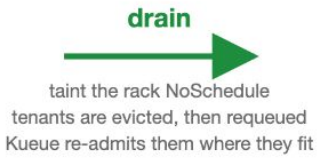
HERO WORKLOADS

A HERO NEEDS ONE WHOLE RACK. EVERYONE ELSE IS INSIDE THEIR QUOTA.

BEFORE

Quota says yes. Topology says no.





AFTER

Hero running. Tenants moved, not lost.



Kueue will not preempt inside another ClusterQueue's nominalQuota, even when topology demands it.

CONFIGURABLE PREEMPTION

- Configurable Preemptions
- kubernetes-sigs/kueue #13396
  - Preempt for topology, not just quota.
  - Bigger jobs may move smaller ones.
  - Defrag happens when a job needs it, not on a schedule.

CLOSING

TAKEAWAYS

1 **NVL72 Hardware is more sensitive to topology than previous generations.**
Across racks, traffic falls off an 18x bandwidth cliff.

2 **The scheduler must know the topology.**
TAS makes placement something you ask for: required, preferred, sliced.

3 **Good placement is worth 2.7x on the same hardware.**
Fragmentation destroys good placement over time.

THANK YOU.