Allocating Kubernetes Idle Per-Instance

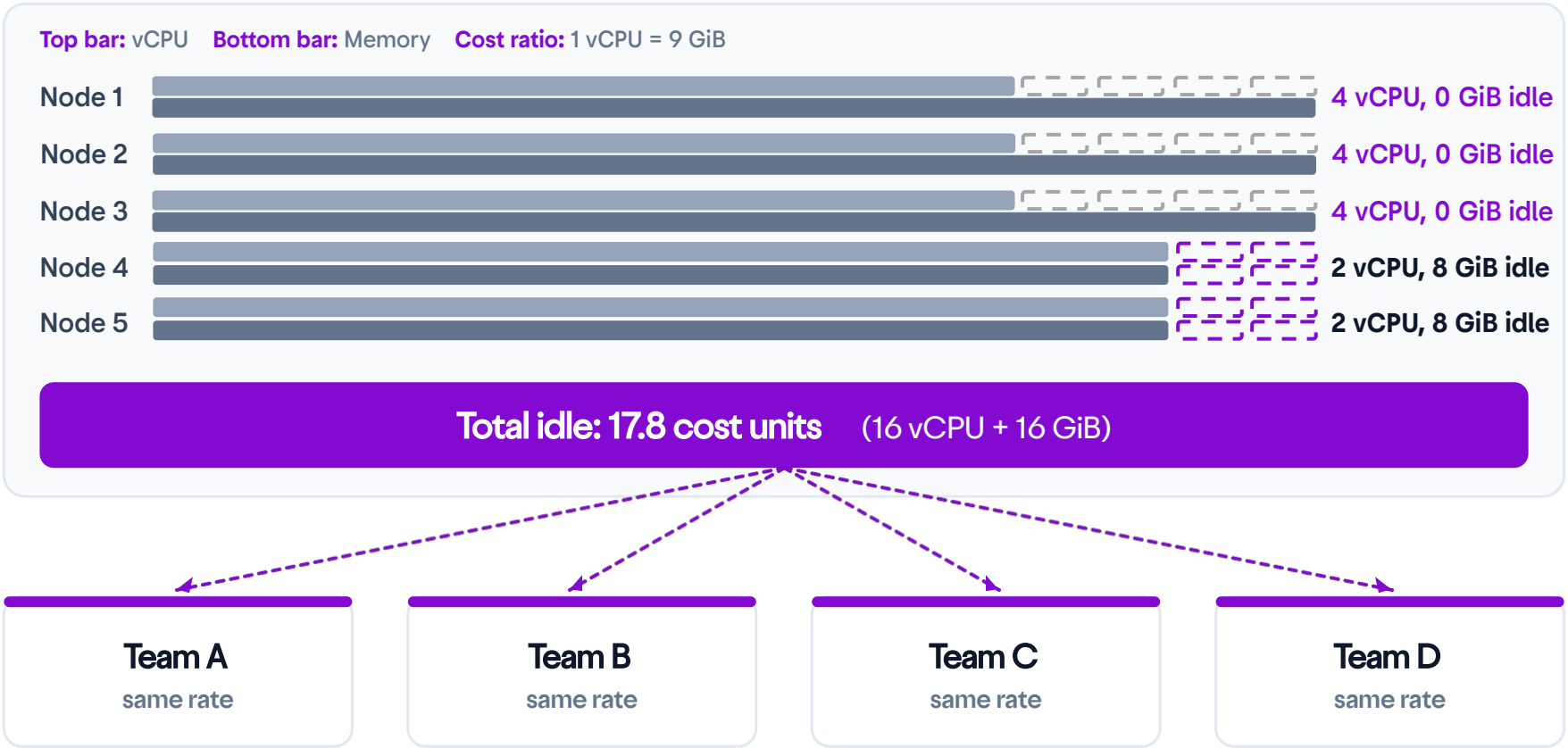
Letting Scale Do What Fairness Rules Couldn't

Vinicius Krischke | Nubank

KubeCon + CloudNativeCon North America 2026

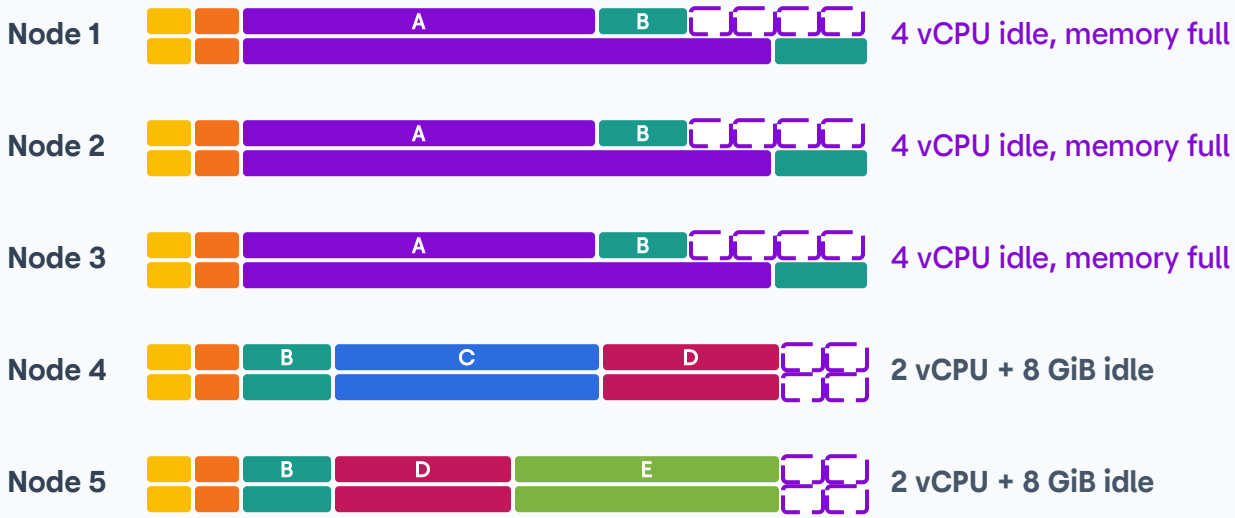
Idle is a tax with no signal

Well-fitted or not, everyone pays the same idle rate.



Same five nodes, now with the workloads

Each node: 16 vCPU / 64 GiB. Top bar: vCPU. Bottom bar: memory.



Total Idle Cost

17.8

cost units

16 vCPU idle

16 GiB idle

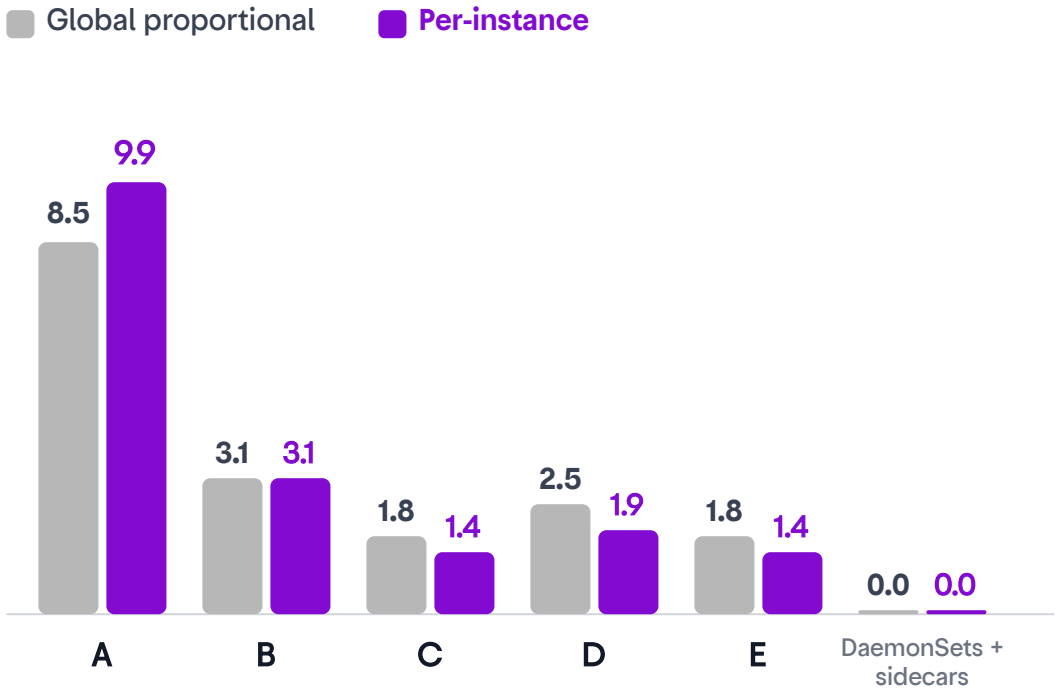
DaemonSets · Sidecars · A (memory-heavy) · B · C · D · E · Idle

DaemonSet: 1 pod per node. Sidecar: helper container inside an app pod.

illustrative numbers

Two ways to split the same 17.8 idle cost units

Per-instance moves idle toward the services whose shape creates it.



+16%

Shift to Service A

Increases from 8.5 to 9.9 cost units

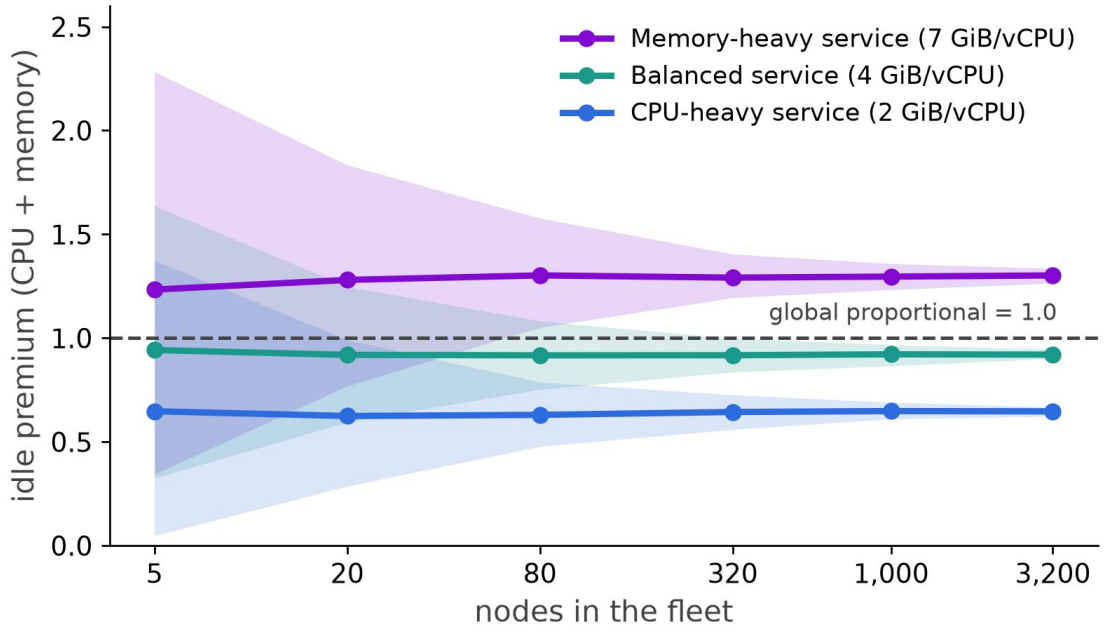
Key Takeaways:

- **Service A** pays more: its memory-heavy pods strand CPU on its nodes.
- **Services C, D, E** pay about 20% less: they only run on nodes with little idle.
- Service B, spread across every node, barely moves (3.1 to 3.1).

illustrative numbers. Idle in cost units (1 vCPU = 9 GiB). DaemonSets and sidecars receive no idle under either method.

Attribute locally, let scale handle fairness

Idle premium = share of idle / share of requests. 1.0 = the old global method.



5 nodes

noisy: right about 6 times in 10

80 nodes

right 98% of the time

3,200 nodes

bands about 25x tighter

"right" = the memory-heavy service pays more idle than the balanced one

synthetic fleet; idle in cost units (1 vCPU = 9 GiB); illustrative

Thank you

Vinicius Krischke

Staff Business Analyst, Nubank

Questions? Reach me directly on LinkedIn.

