
KubeCon
CloudNativeCon
North America 2024

KubeCon | CloudNativeCon

North America 2024

SIG Scheduling Intro & Updates

Aldo Culquicondor
Software Engineer
Google

Kensei Nakada
Software Engineer
Tetrate.io

SIG Scheduling

Responsible for the components that make Pod placement decisions

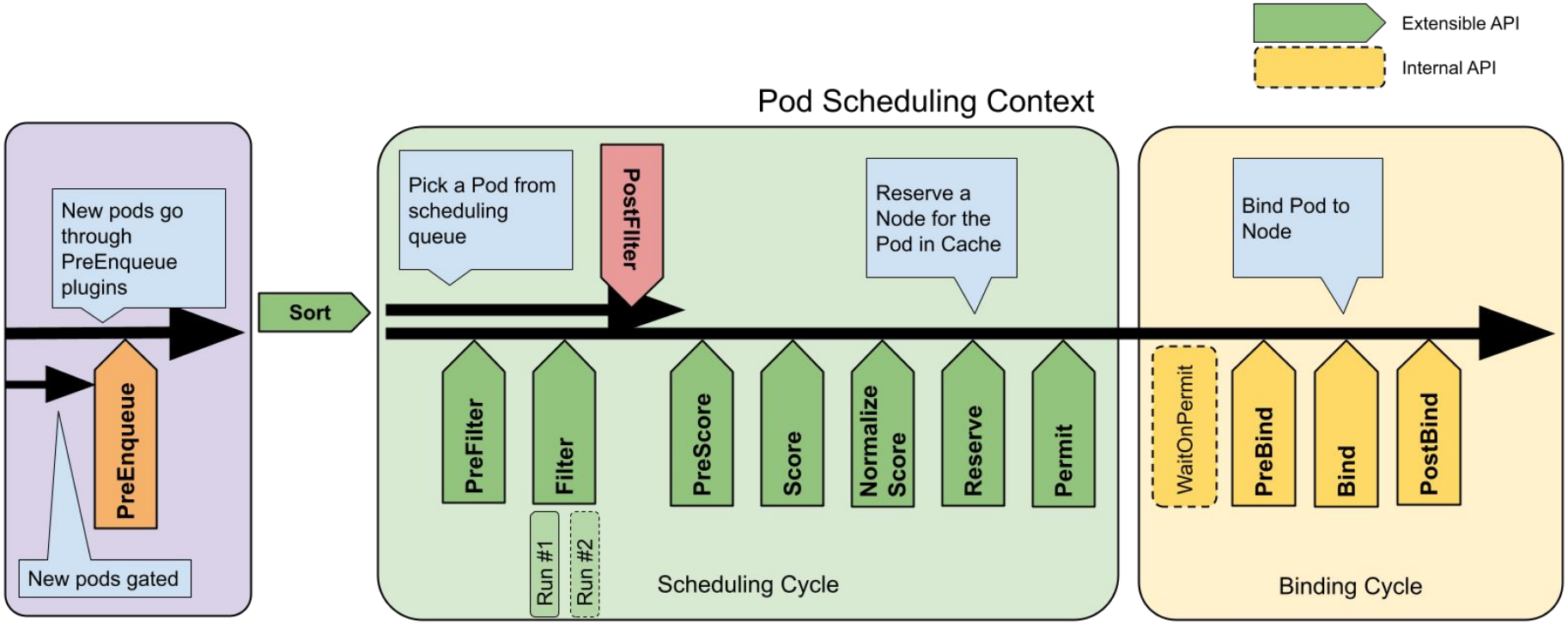
kube-scheduler

kueue

kwok

kube-scheduler-simulator

kube-scheduler

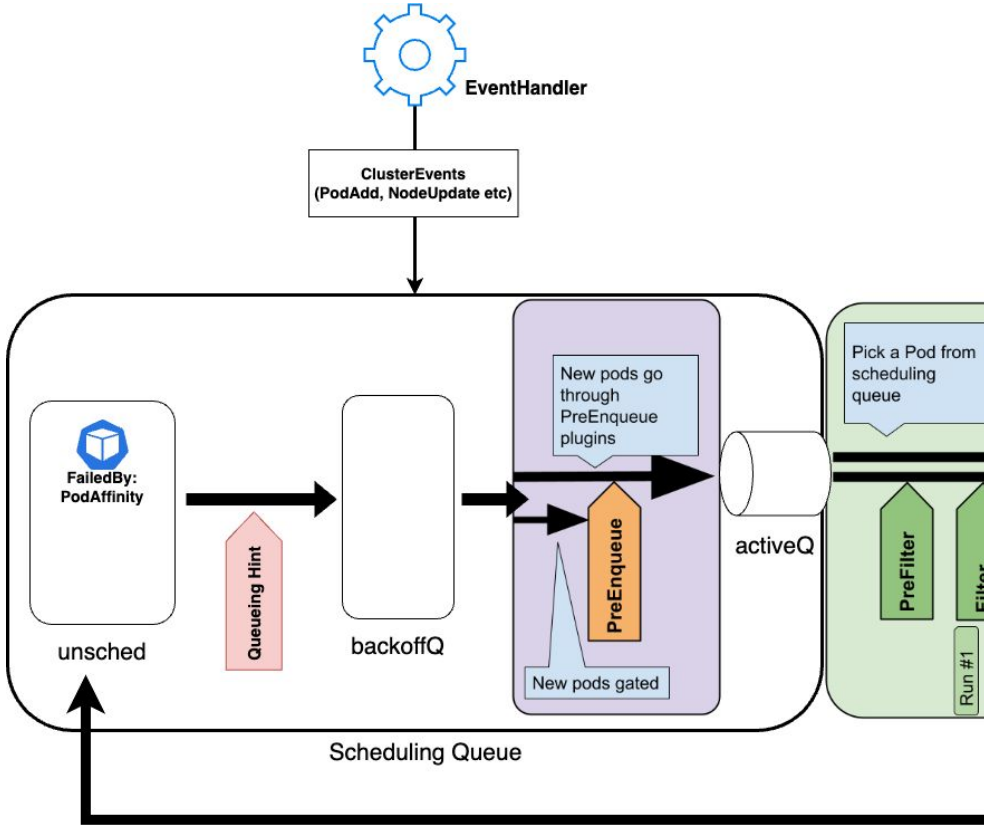


kube-scheduler: Queueing Hints (v1.32: beta)

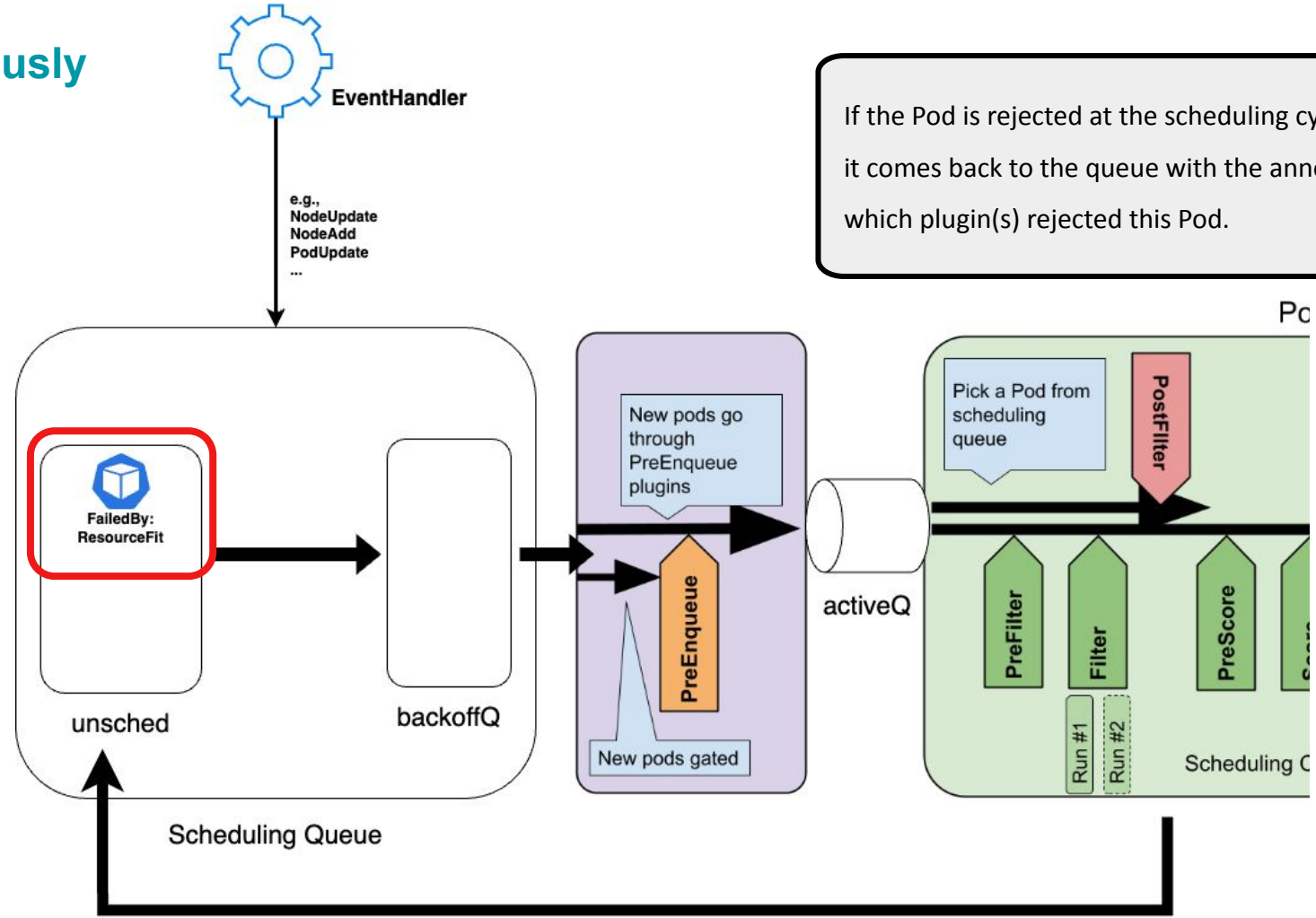
💬 Motivation

Cluster events can make previously unschedulable Pods schedulable. How to decide whether it's worth retrying a particular Pod?

Existing mechanism was too coarse and not extensible for out-of-tree plugins.

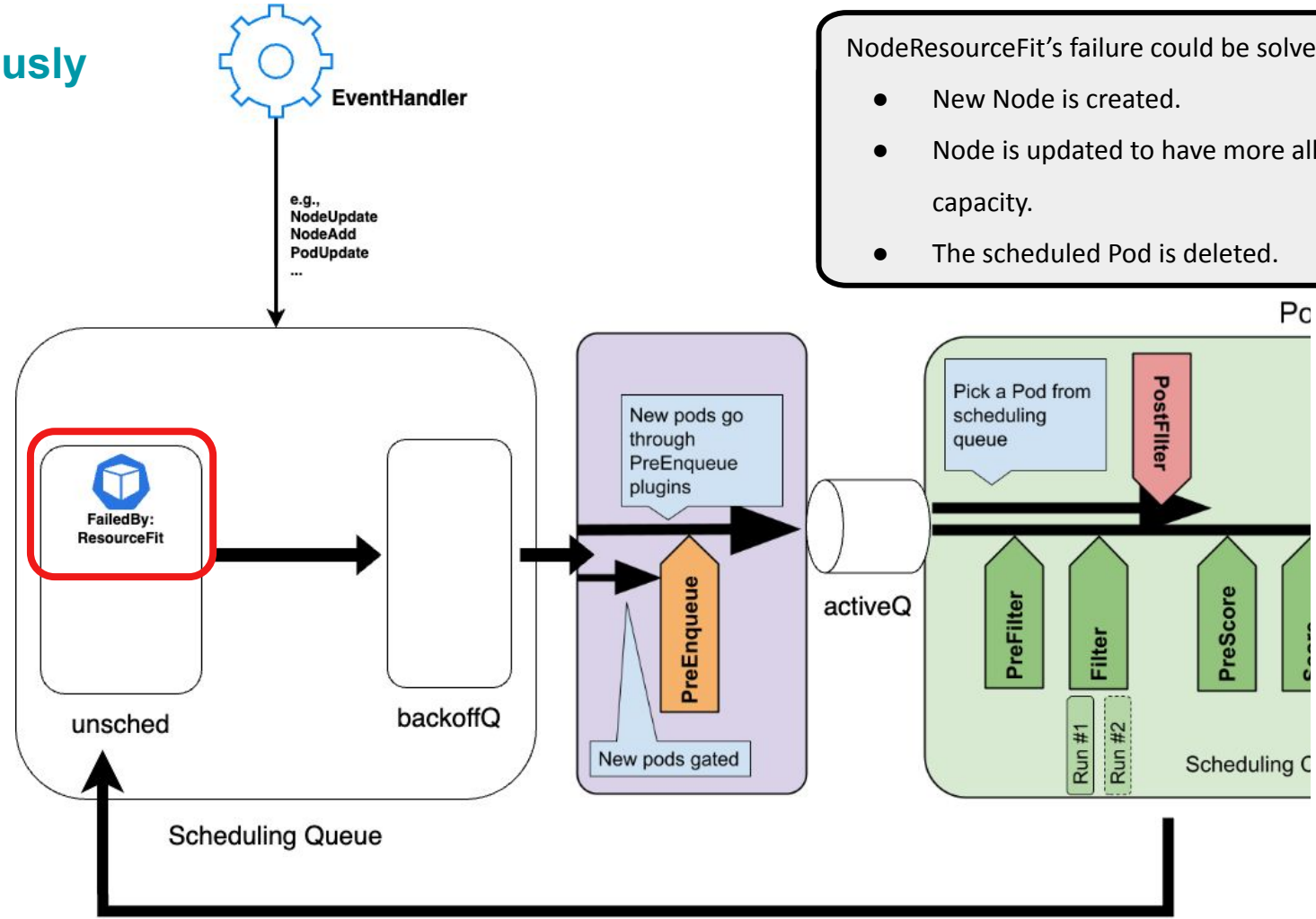


Previously



If the Pod is rejected at the scheduling cycle,
it comes back to the queue with the annotation
which plugin(s) rejected this Pod.

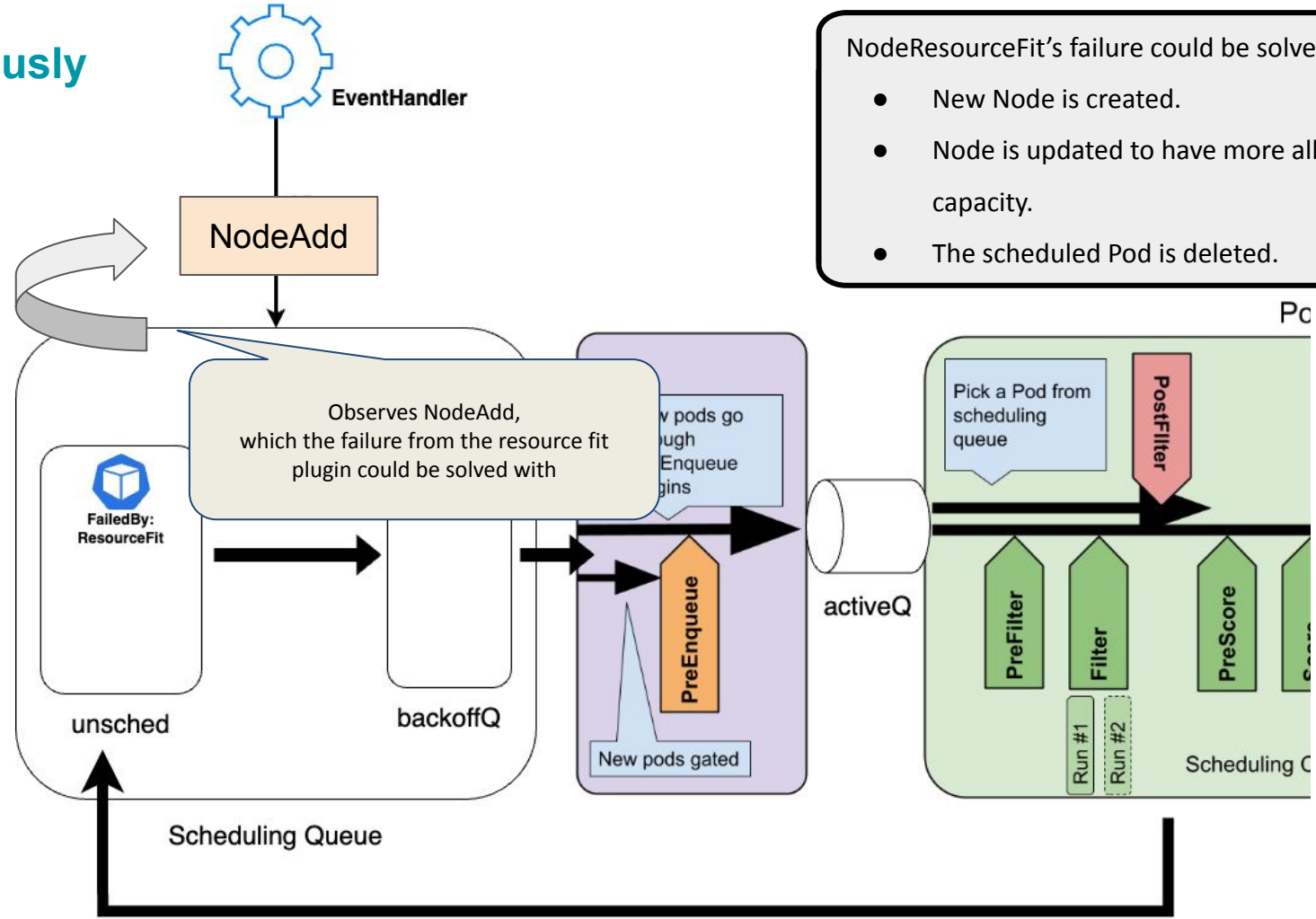
Previously



NodeResourceFit's failure could be solved by:

- New Node is created.
- Node is updated to have more allocatable capacity.
- The scheduled Pod is deleted.

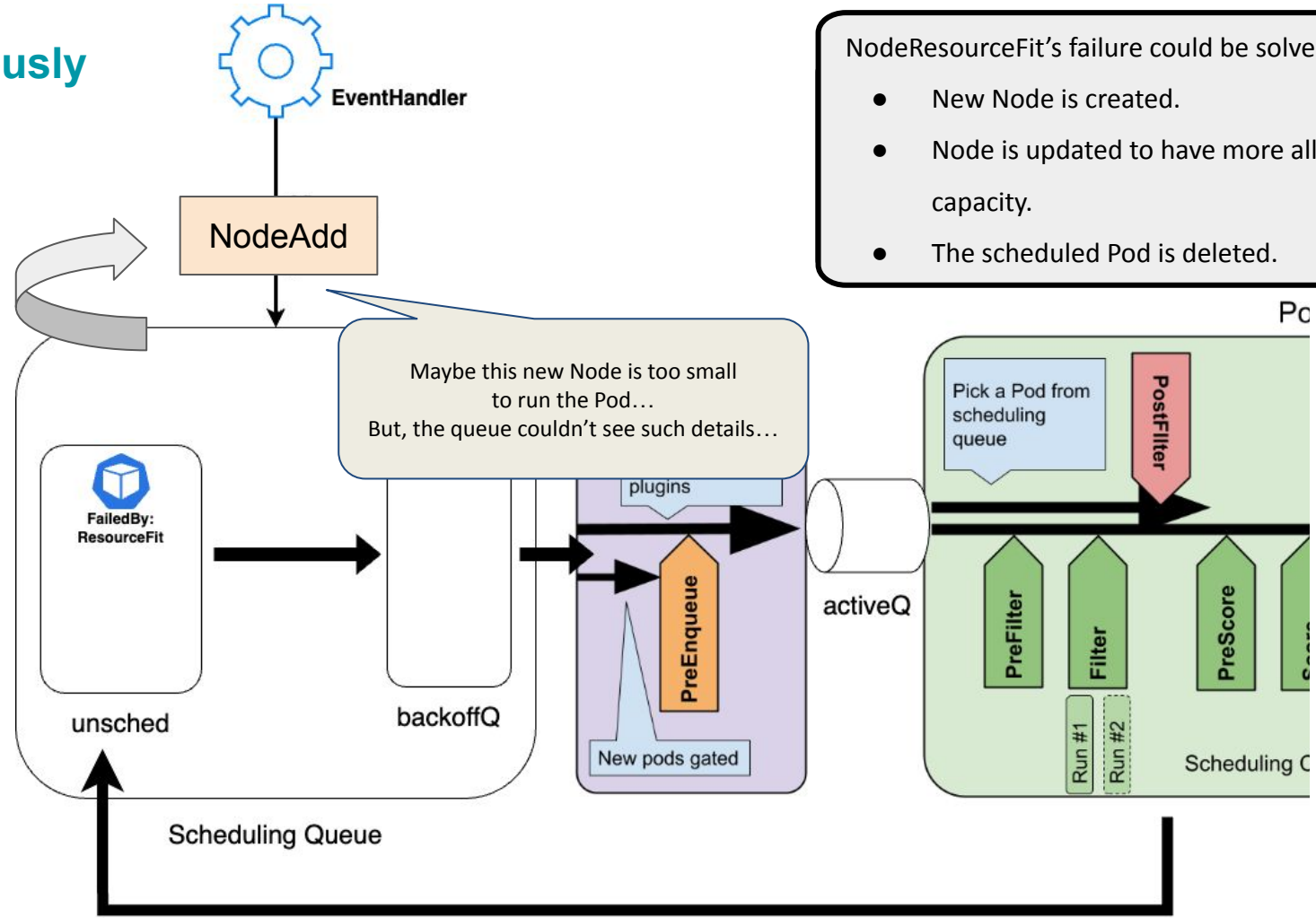
Previously



NodeResourceFit's failure could be solved by:

- New Node is created.
- Node is updated to have more allocatable capacity.
- The scheduled Pod is deleted.

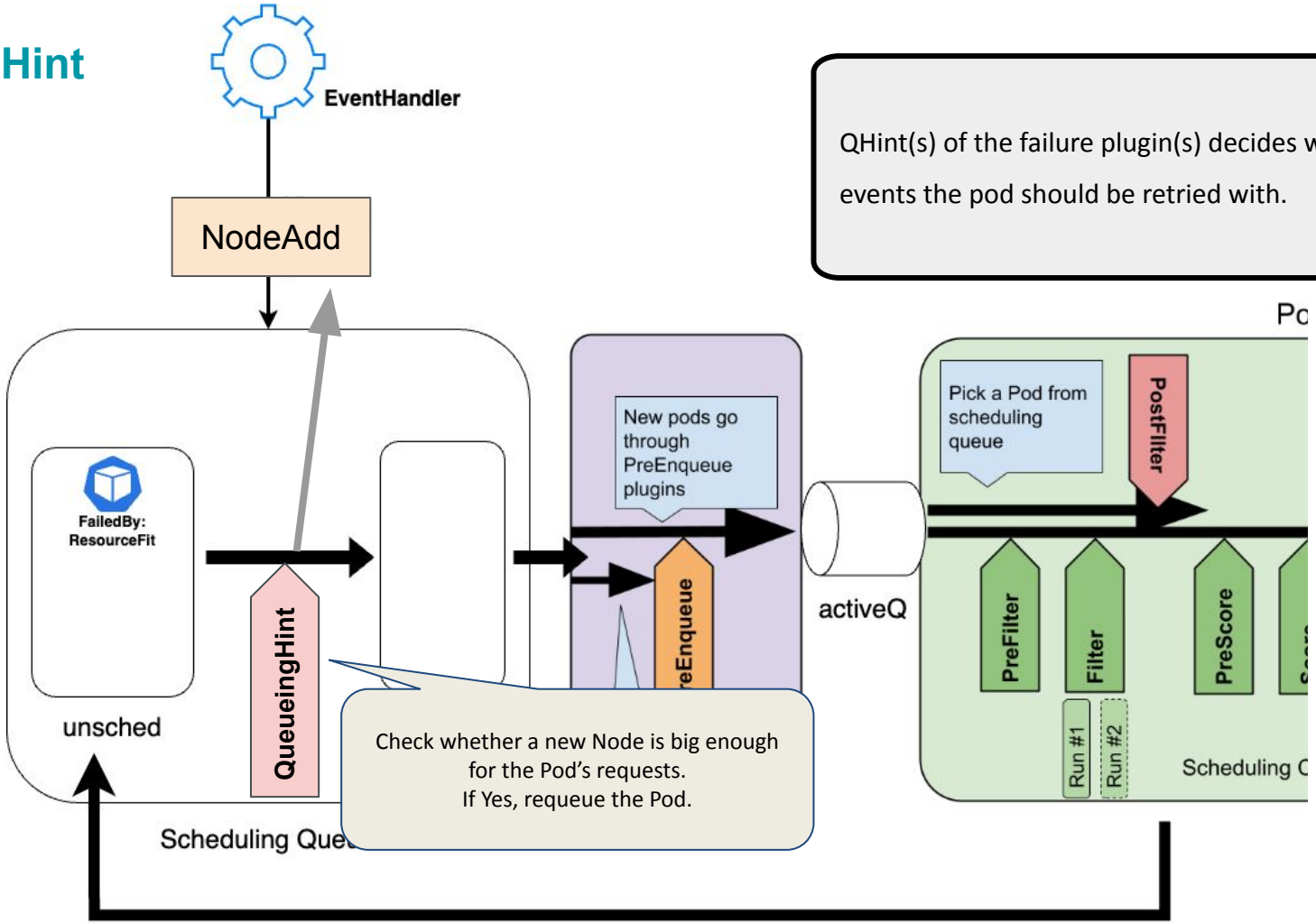
Previously



NodeResourceFit's failure could be solved by:

- New Node is created.
- Node is updated to have more allocatable capacity.
- The scheduled Pod is deleted.

After QHint



QHint(s) of the failure plugin(s) decides which events the pod should be retried with.

History of Queueing Hints

- v1.28:
  - The feature is released as Beta (enabled by default) with a few QHint implementation (DRA etc)
  - 🔥 Users observed a memory leak issue.
  - We disabled the feature by default in a patch release.
- v1.29-v1.31: We kept working on expanding the QHints and doing bug fixes.
- v1.32:
  - We finished implementing QHints in all plugins.
  - We identified and fixed the last major memory leak, finally!
  - We increased the integration test/performance test coverage... a lot.
  - 🚀 We enabled the feature by default.

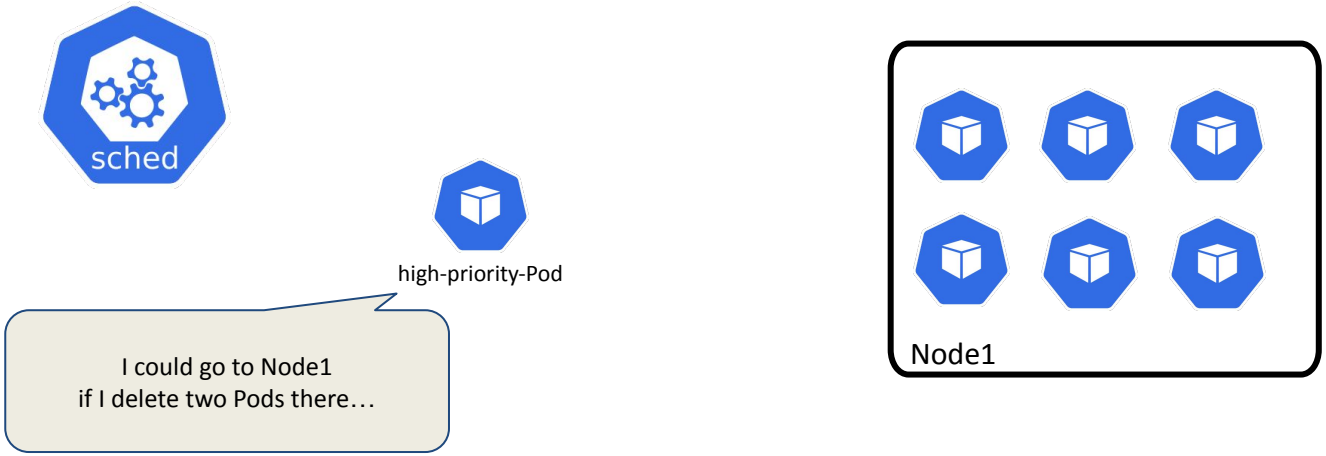
kube-scheduler: Async preemption (v1.32: alpha)

💬 Motivation

If your pods get unschedulable, they go through **the preemption** before going back to the queue.

When the preemption happens, the scheduling cycle takes time to complete

because it has to make some API calls. It impacts the whole scheduling latency.



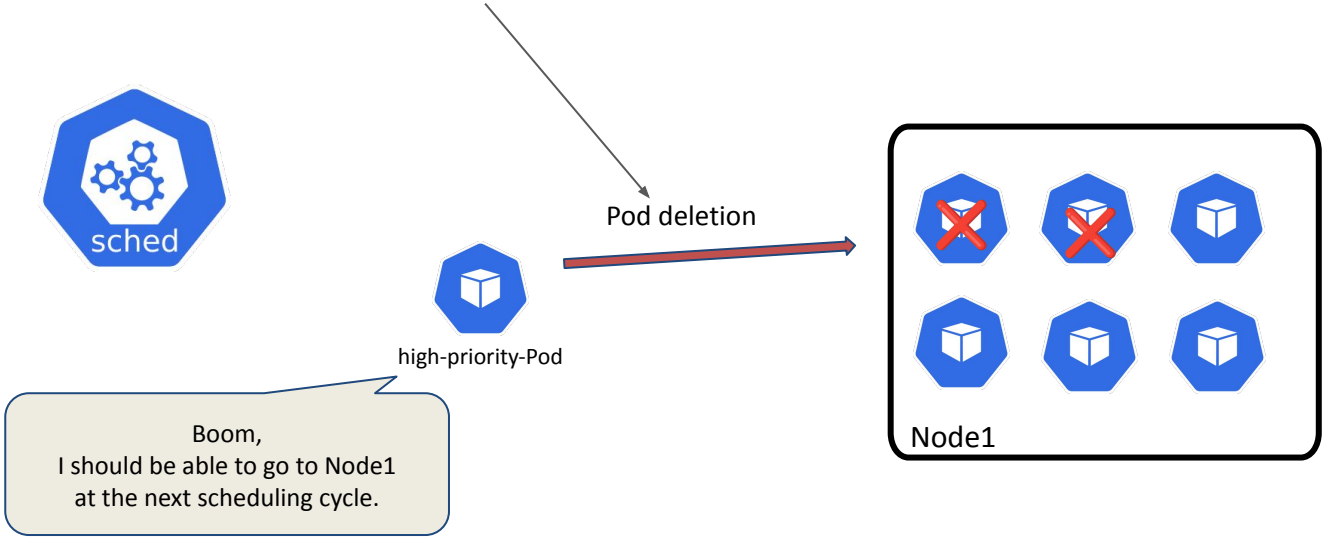
kube-scheduler: Async preemption (v1.32: alpha)

💬 Motivation

If your pods get unschedulable, they go through **the preemption** before going back to the queue.

When the preemption happens, the scheduling cycle takes time to complete,

because it has to make **some API calls**. It impacts the whole scheduling latency.



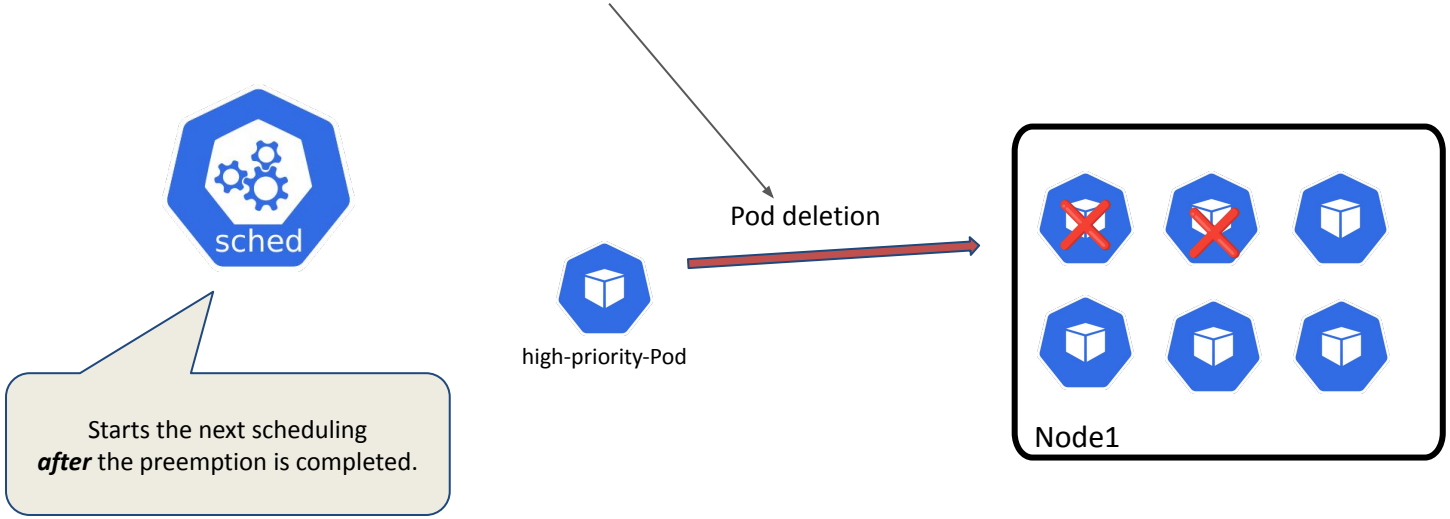
kube-scheduler: Async preemption (v1.32: alpha)

💬 Motivation

If your pods get unschedulable, they go through **the preemption** before going back to the queue.

When the preemption happens, the scheduling cycle takes time to complete,

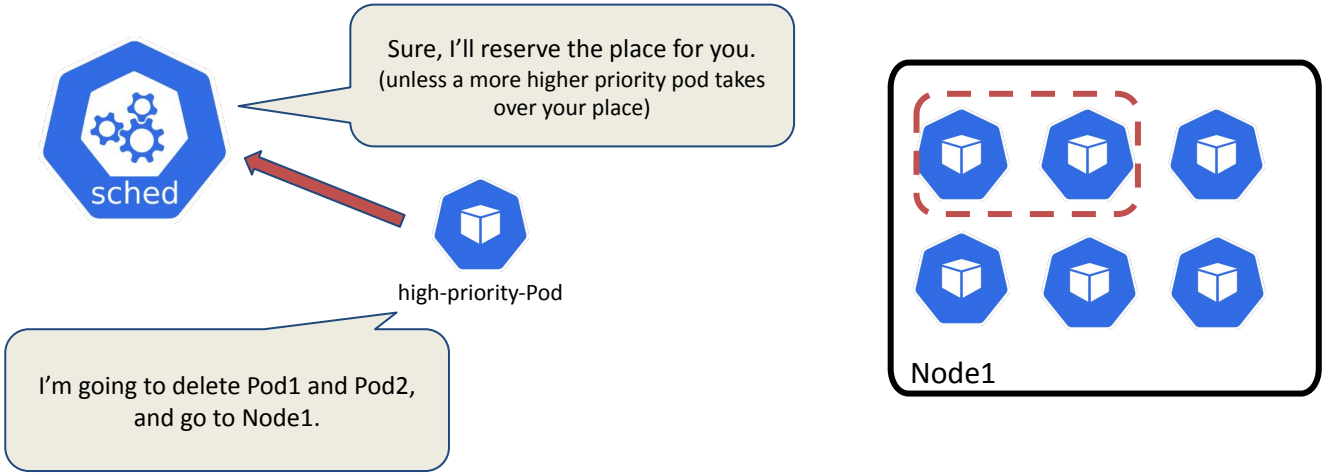
because it has to make **some API calls**. It impacts the whole scheduling latency.



kube-scheduler: Async preemption (v1.32: alpha)

💡 Proposal

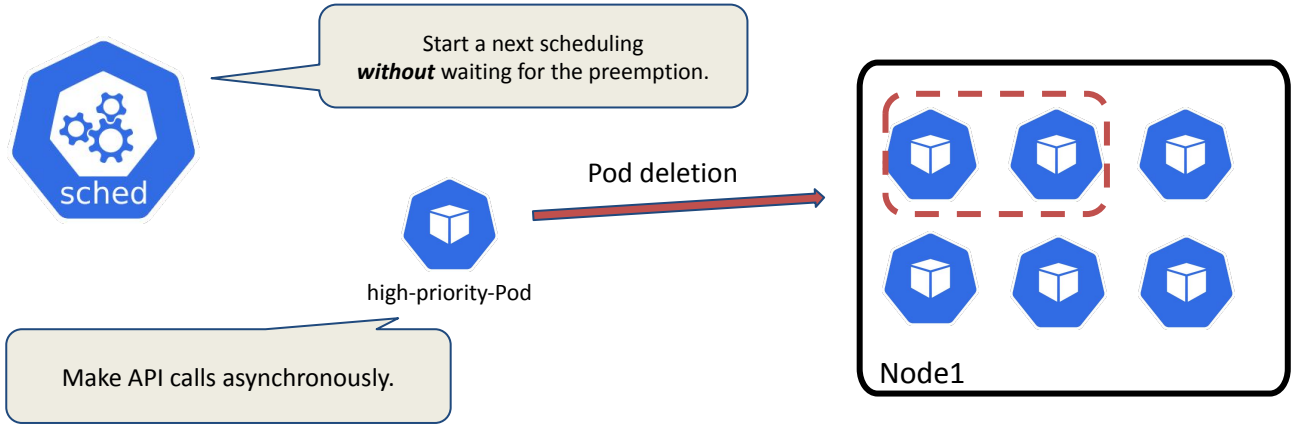
Once deciding which Pod(s) to delete, **making the API calls asynchronously**,
and starting the next scheduling cycle **without waiting for the preemption** to be completed.
The preemption asks the scheduler to reserve the preemptor's place on the node
before starting the next scheduling cycle
so that the next scheduling cycle takes the preemption into consideration.



kube-scheduler: Async preemption (v1.32: alpha)

💡 Proposal

Once deciding which Pod(s) to delete, **making the API calls asynchronously**,
and starting the next scheduling cycle **without waiting for the preemption** to be completed.
The preemption asks the scheduler to reserve the preemptor's place on the node
before starting the next scheduling cycle
so that the next scheduling cycle takes the preemption into consideration.

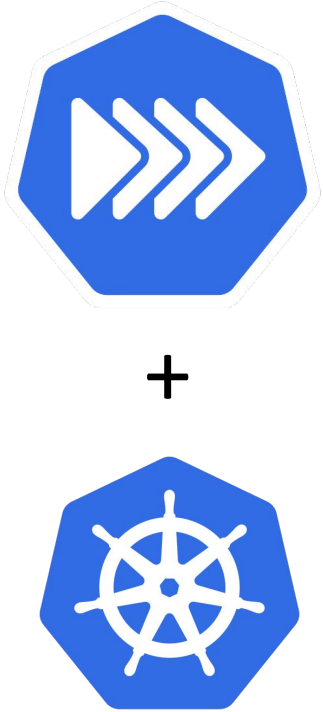


kube-scheduler: Other Updates

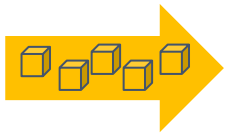
- v1.30:
  - Pod Scheduling readiness gates graduated to GA
  - minDomains for Pod topology spreading graduated to GA
- v1.31:
  - matchLabelKeys in Pod affinity and anti-affinity graduated to Beta
- v1.32:
  - DRA structured parameters graduated to Beta
  - Classic DRA was removed

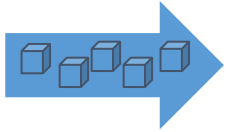
Kueue

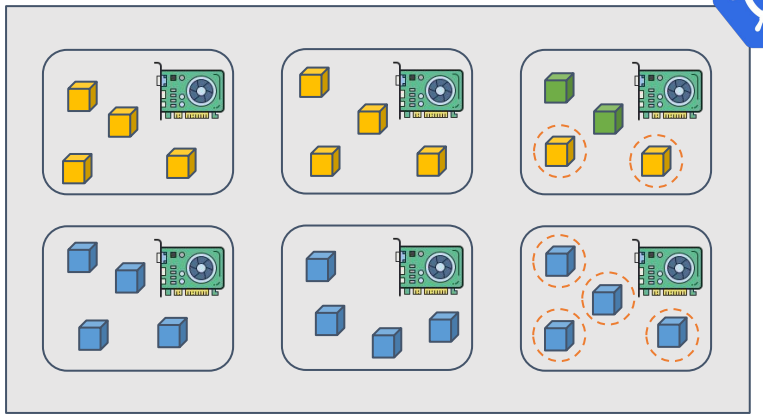
- Kueue interacts with kube-scheduler and cluster-autoscaler to provide a full batch /training system in Kubernetes.
- Kueue determines whether workloads should wait for resources or run, based on:
  - Per-tenant quotas
  - Borrowing and lending limits
  - Fair sharing rules New in v0.7
  - The hierarchy of the organization New in v0.9
- Kueue integrates with Pods, Job, JobSet, Kubeflow, KubeRay and has extension mechanisms.



Kueue: Fair sharing







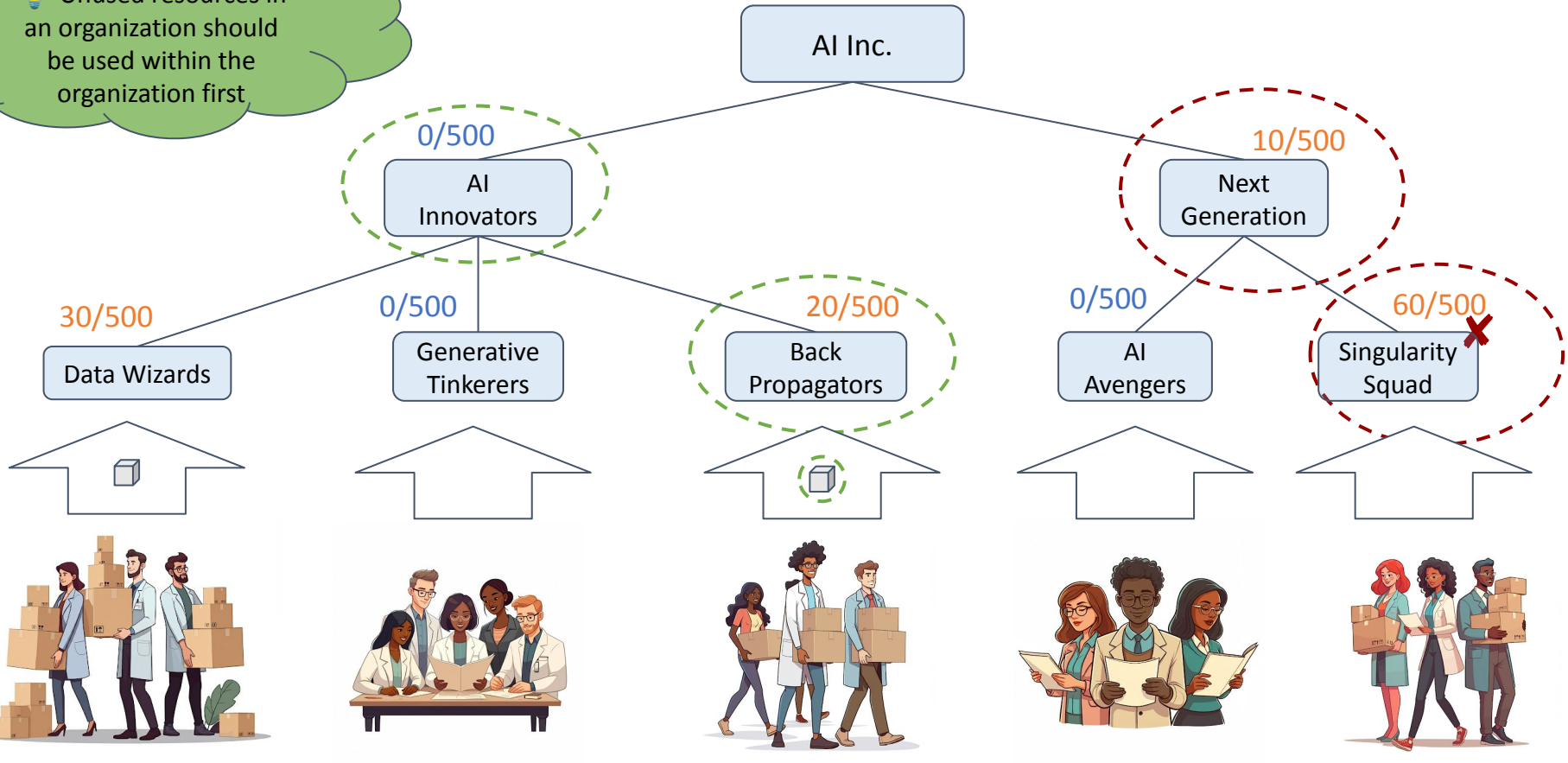
```yaml
kind: ClusterQueue
metadata:
  name: "bob-queue"
spec:
  cohort: "lab"
  resourceGroups:
  - coveredResources: ["acme.com/gpu"]
    flavors:
    - name: "default-flavor"
      resources:
      - name: "acme.com/gpu"
        nominalQuota: 8
  preemption:
    reclaimWithinCohort: "Any"
  fairSharing:
    weight: 1
```

```yaml
kind: ClusterQueue
metadata:
  name: "alice-queue"
spec:
  cohort: "lab"
  resourceGroups:
  - coveredResources: ["acme.com/gpu"]
    flavors:
    - name: "default-flavor"
      resources:
      - name: "acme.com/gpu"
        nominalQuota: 8
  preemption:
    reclaimWithinCohort: "Any"
  fairSharing:
    weight: 2
```

sched.co/1izqO

Kueue: Hierarchical cohorts

Unused resources in an organization should be used within the organization first



And it works in multiple clusters!

sched.co/1iCOV

Kwok

***K**ubernetes **W**ith**O**ut **K**ubelet*

The toolkit to set up a cluster of thousands of Nodes for scheduling simulations… in seconds!

What's new?

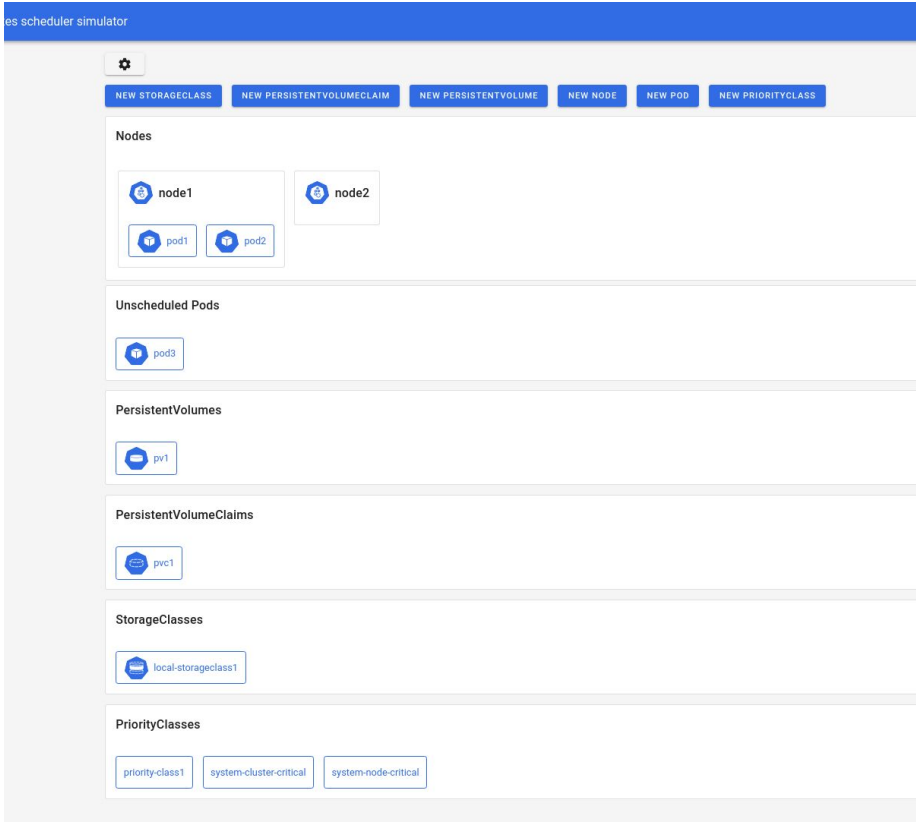
- Optimized Stage to allow easier simulation of normal/abnormal behavior of resources
- Optimized the experience of CPU and MEM simulation on metrics
- Helm chart installation available
- kwokctl
  - Support for MacOS and Windows
  - More runtimes such as lima and finch
- Roadmap:
  - Plan API v1alpha2
  - Enhance the experience of using kwok on existing clusters without using kwokctl
  - Extend simulation capabilities to include GPU usage and other custom metrics
  - Simulation for Volume Provisioner

kube-scheduler-simulator

Kube-scheduler-simulator offers the debuggability for your scheduler.

- The scheduler is composed of the plugins.
- The simulator visualizes all the internal decisions for each pod's scheduling.

You can create resources on Web UI

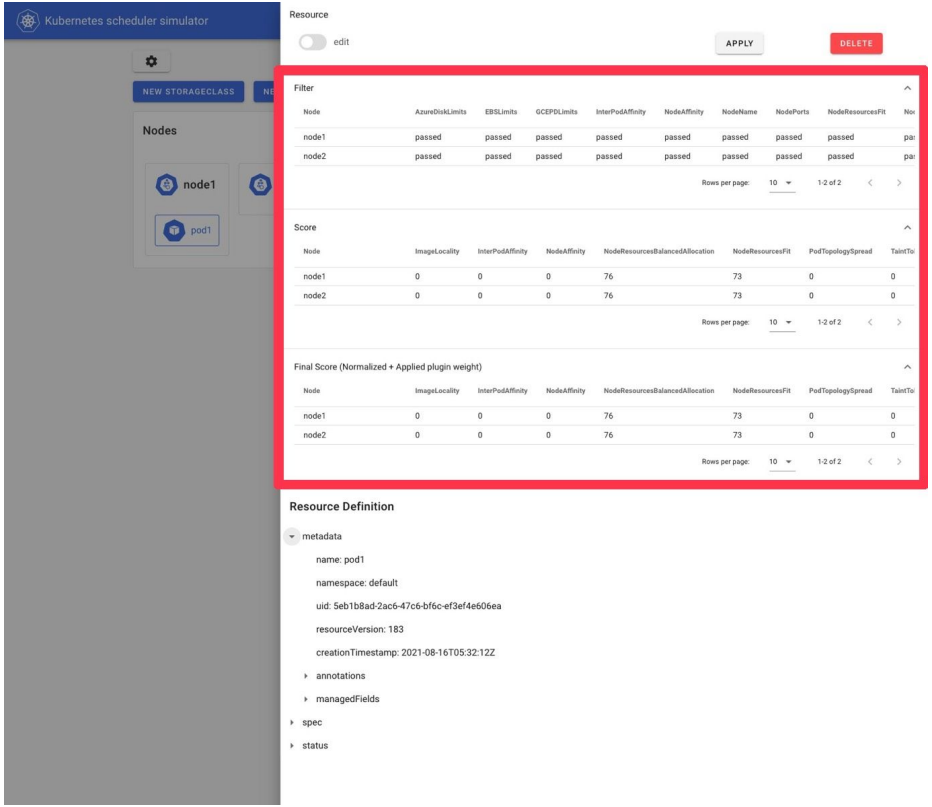


kube-scheduler-simulator

Kube-scheduler-simulator offers the debuggability for your scheduler.

- The scheduler is composed of the plugins.
- The simulator visualizes all the internal decisions for each pod's scheduling.

It shows how your scheduler decides the Node for your Pod.



kube-scheduler-simulator: What's new?

We've been adding new features while refactoring the simulator to be more maintainable.

- v0.2.0: Migrate the internal Kube-apiserver and controllers to Kwok.
- v0.3.0: Introduce the syncing feature:
  - *Testing a scheduler is hard* => Run scheduling simulator using production pods.
  - The syncing feature keeps importing the resources from your prod cluster to simulate your prod environment at the simulator, with potentially different configuration.
- v0.4.0: The standalone debuggable scheduler:
  - The debuggable scheduler schedules pods like the normal scheduler but also outputs all the scheduling internal decisions as pod annotations.
  - It now works standalone: you can use it in the dev cluster instead of the normal scheduler so you can debug low-level scheduling decisions.

Feedback

sched.co/1hovV