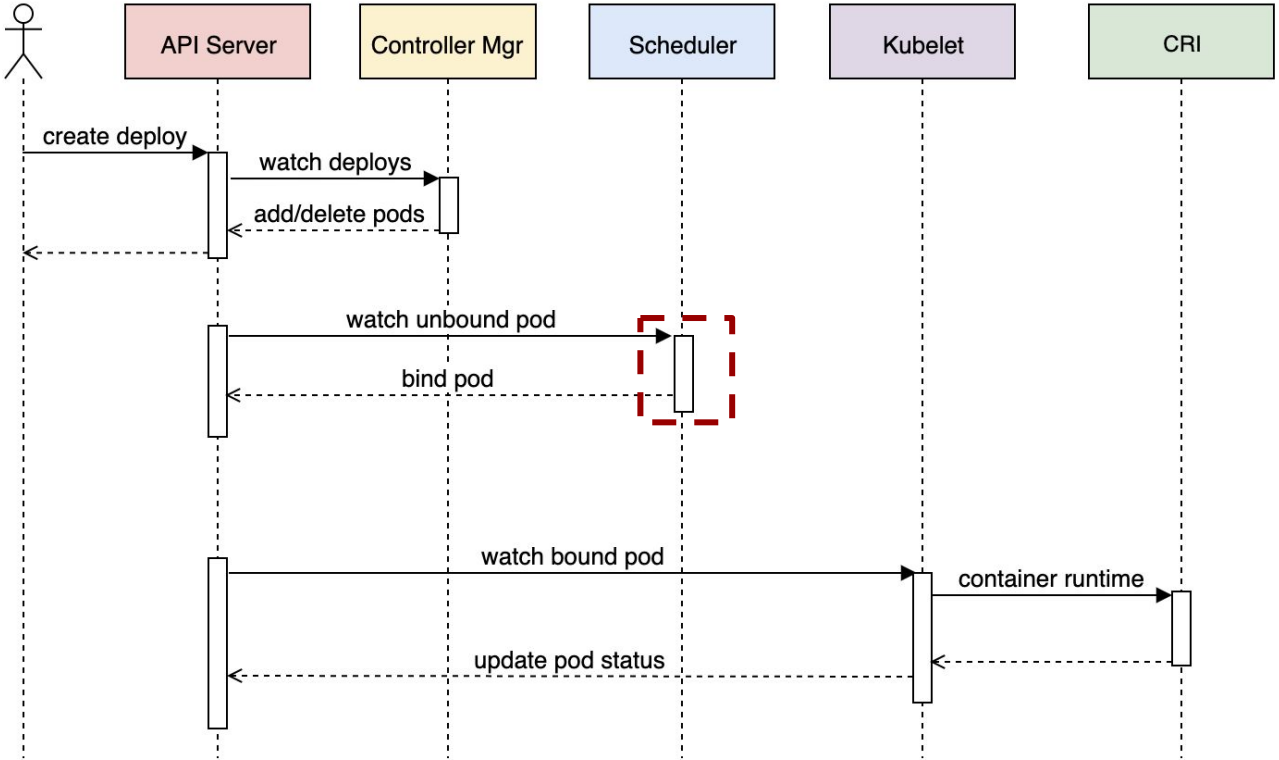
SIG-Scheduling Intro

Wei Huang (IBM, @Huang-Wei)
Ravi Santosh Gudimetla (Red Hat, @ravisantoshgudimetla)

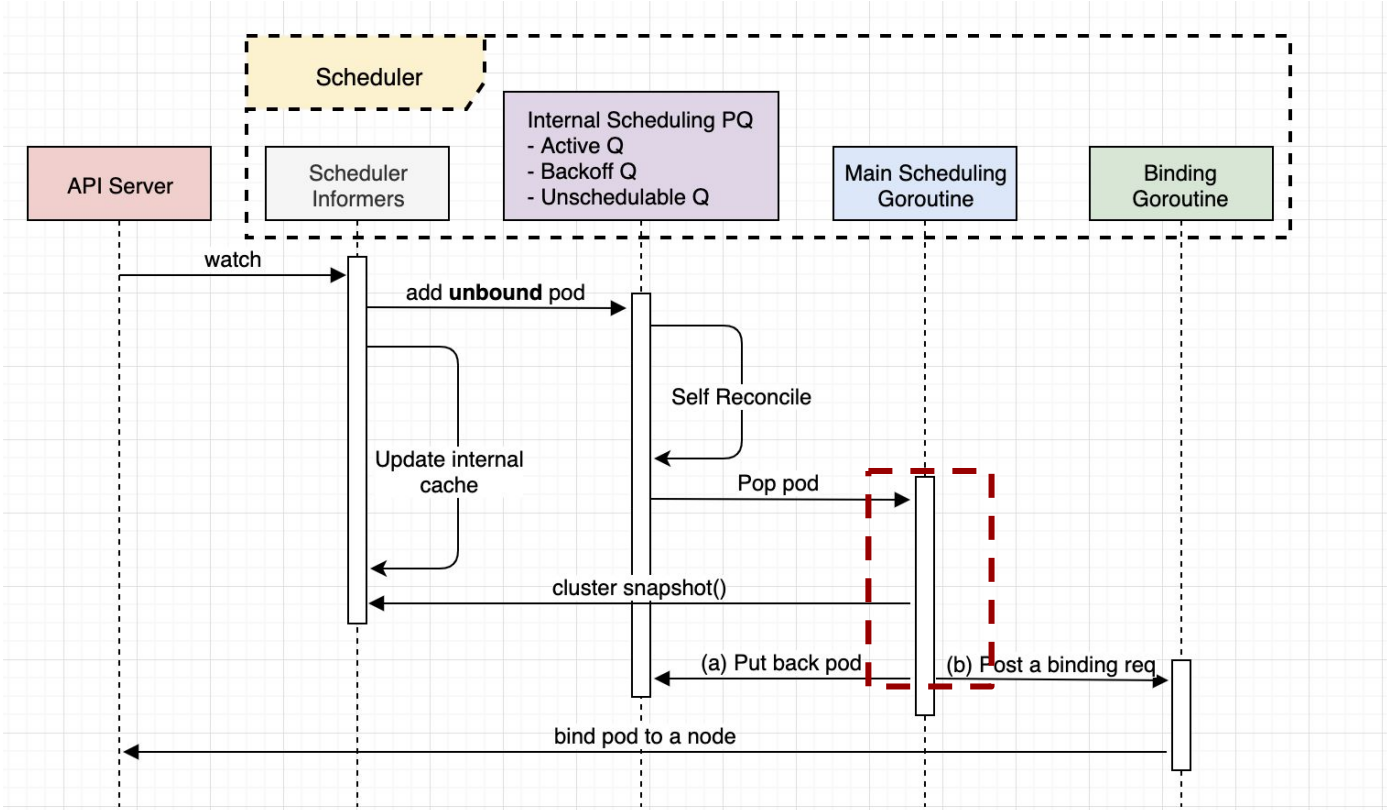
kubernetes

Scope of Scheduler



kubernetes

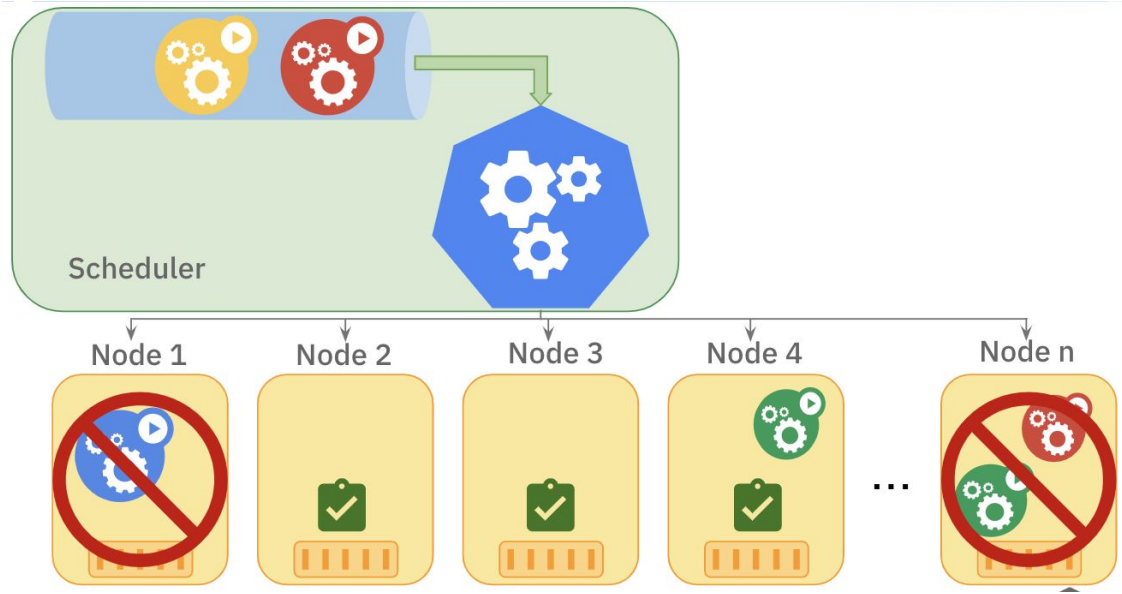
Detailed Scheduling Flow



kubernetes

Phase 1: Predicates (a.k.a Filtering)

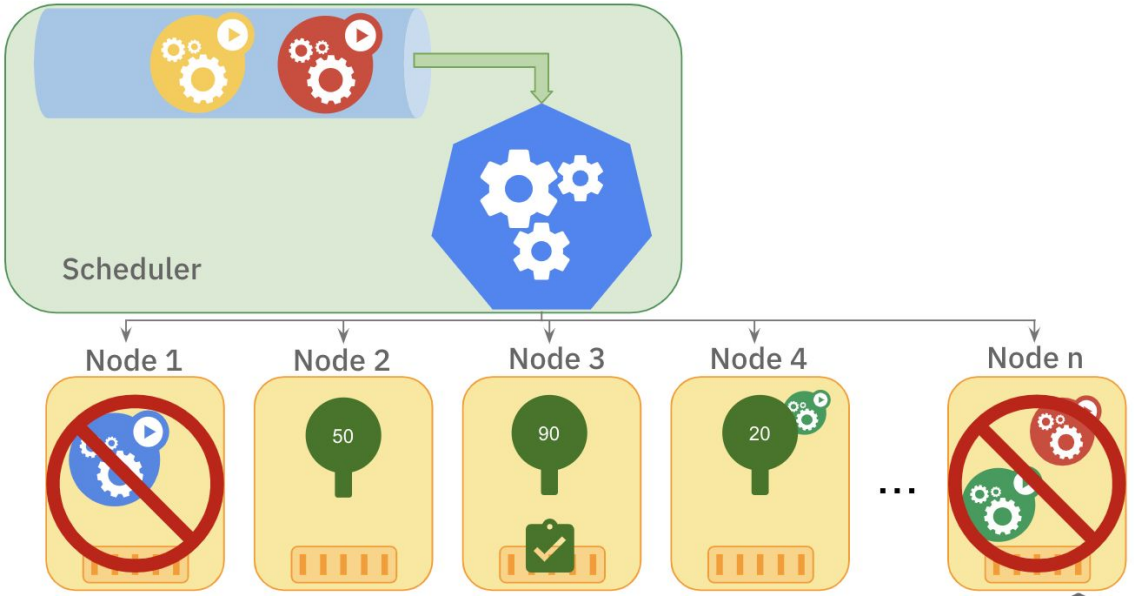
1. Find qualified nodes which pass all Predicates.
2. If none is qualified, see if preempting low-priority Pods helps.



kubernetes

Phase 2: Priorities (a.k.a. Scoring)

1. For each “filtered” node, score it according based on Priorities.
2. The node with highest score will be chosen as the running node.



Recent Developments in default scheduler

Recent Changes:

GA in 1.14: Priority and Preemption

GA in 1.17: ResourceQuotaScopeSelectors

Major on-going features:

- Scheduler Framework - Alpha in 1.15; Code Migration Phase 1 completed in 1.17
- EvenPodsSpread - Alpha in 1.16
- Extended RequestedToCapacityRatio Priority - Alpha in 1.16

Design Rationale of Scheduler

1. The scheduler is **NOT** responsible for managing lifecycle of Pods.
2. The minimum scheduling unit is **POD** (tried EquivalenceCache, but not good as supposed to be)
3. Schedule **one pod** at a time
4. **Best Fit** vs. First Fit
5. **Predicates** and **Priorities**
6. **Configurable** (schedule config file)
7. **Plugable** (new scheduler framework, scheduler extender, multiple schedulers)

Descheduler

Trigger Of Pod Movement/Migration

Eviction -> Creation -> Re-schedule

Use Cases of Descheduler

- Some nodes are **under or over utilized**.
- The original scheduling decision does not hold true any more, **as taints or labels are added to or removed from nodes, pod/node affinity** requirements are not satisfied any more.
- **New nodes** are added to clusters.

kubernetes

Descheduler strategies

- RemoveDuplicates
- LowNodeUtilization
- RemovePodsViolatingInterPodAntiAffinity
- RemovePodsViolatingNodeAffinity

Pod Eviction Restriction

- Critical pods (with annotations scheduler.alpha.kubernetes.io/critical-pod and evict) are never evicted.
- Pods (static or mirrored pods or stand alone pods) not part of an RC, RS, Deployment or Jobs are never evicted because these pods won't be recreated.
- Pods associated with DaemonSets are never evicted.
- Pods with local storage are never evicted.
- Best efforts pods are evicted before Burstable and Guaranteed pods.
- Pod are never evicted if its PDB is violated.

Overview of kube-batch

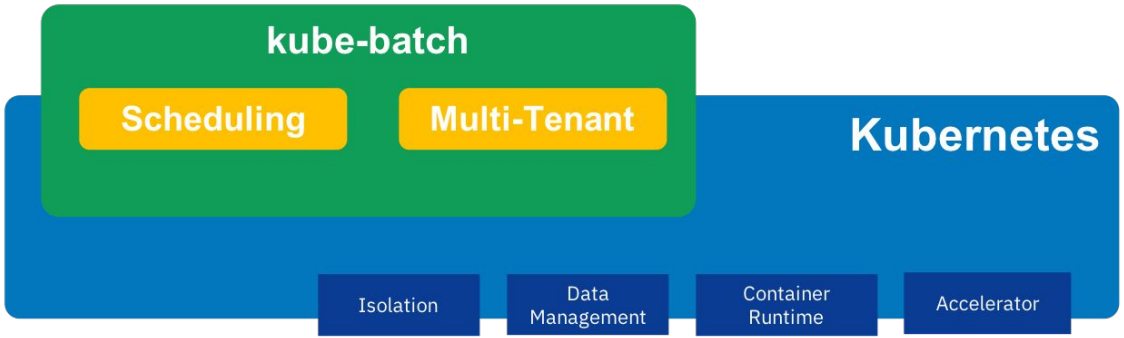
PaddlePaddle

Spark

Caffe2

serverless

Infra



kube-batch focus on:

- "Batch" scheduling
- Resource sharing between multi-tenant

kube-batch **NOT** support:

- Data Management
- Accelerator (Kubelet), e.g. GPU
- Isolation for multi-tenant
- Job Management
- New container runtime, e.g. Singularity, CharlieCloud

kubernetes

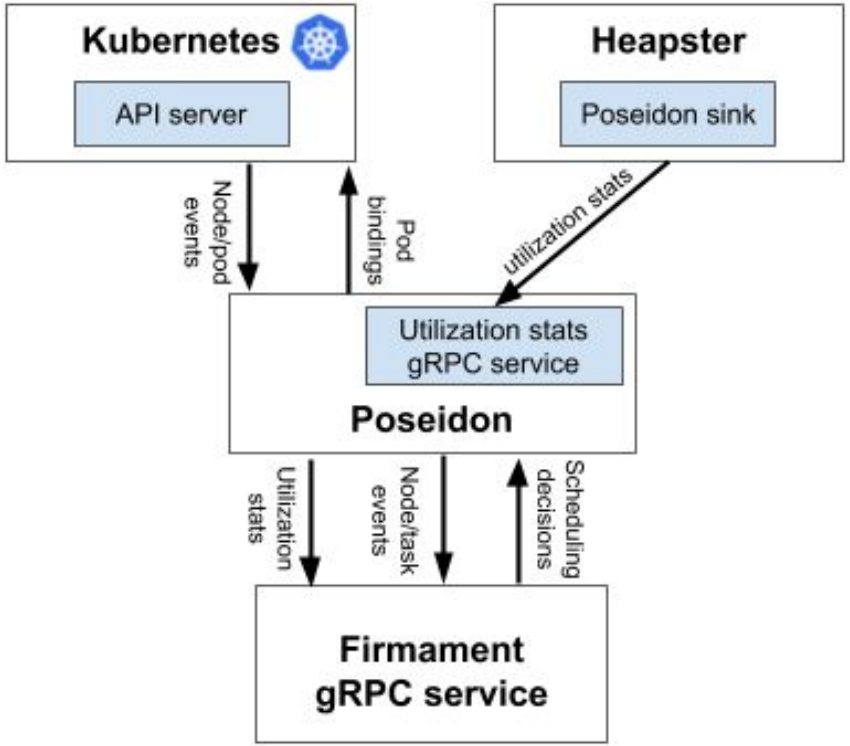
Features of kube-batch

- Co-scheduling
- “Fair-sharing” (job/queue)
- Preemption/Reclaim
- Task Priority within Job
- Predicates
- Queue
- Backfill (partially)
- Dynamic configuration

Batch Capability into Kubernetes (#68357)

kubernetes

Poseidon



Poseidon/Firmament scheduler augments the current Kubernetes scheduling capabilities by incorporating a new novel flow network graph based scheduling capabilities alongside the default Kubernetes Scheduler.

Firmament models workloads on a cluster as flow networks and runs min-cost flow optimizations over these networks to make scheduling decisions.

kubernetes

Features of Poseidon

1. Node level Affinity and Anti-Affinity
2. Pod level Affinity and Anti-Affinity
3. Taints & Tolerations
4. Gang Scheduling

kubernetes

How those schedulers **work together ???**

kubernetes

Sorry, I don-t know :(

Contact Us

Chairs

- @ahg-g
- @k82cn

Home page: https://github.com/kubernetes/community/tree/master/sig-scheduling

Slack channel: https://kubernetes.slack.com/messages/sig-scheduling

Mail list: https://kubernetes.slack.com/messages/sig-scheduling

Google doc: https://docs.google.com/document/d/13mwye7nvrmV11q9_Eg77z-1w3X7Q1GTbslpml4J7F3A/view

Thanks!

Q & A