GOTC

全球开源技术峰会

THE GLOBAL OPENSOURCE TECHNOLOGY CONFERENCE

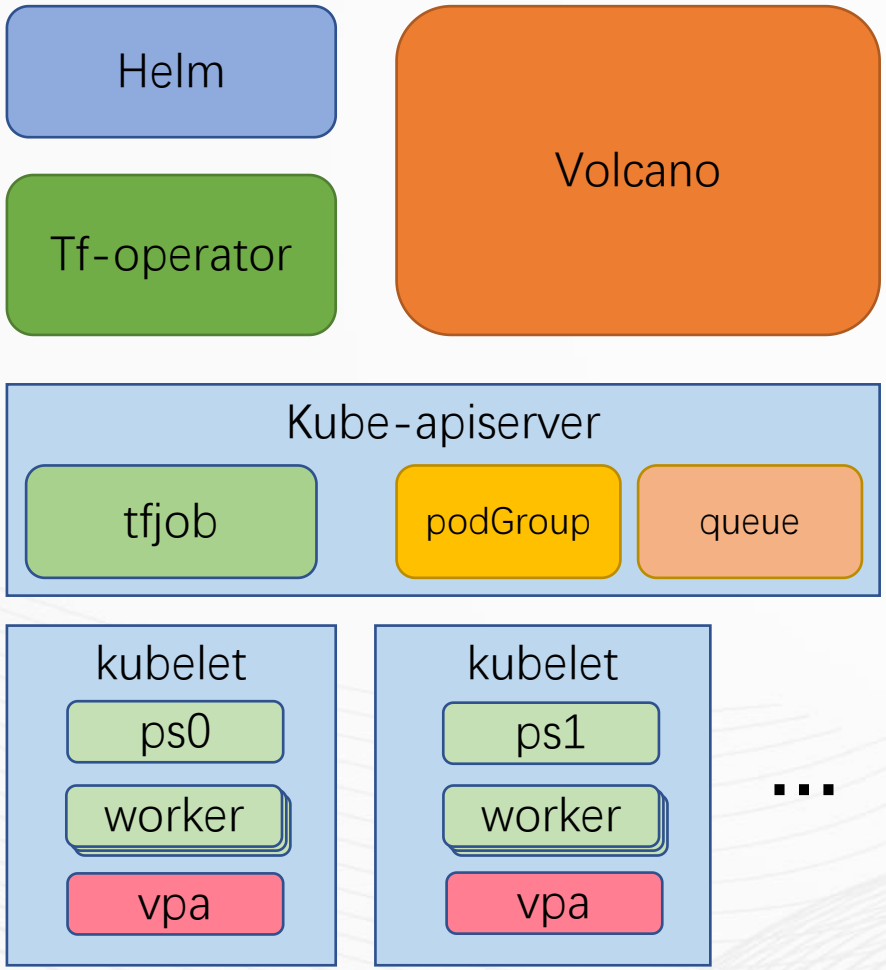
OPEN SOURCE , OPEN WORLD

CNCF分论坛专场

唯品会如何基于Volcano与AI训练场景提高集群利用率

何颖鹏 2021年07月10日

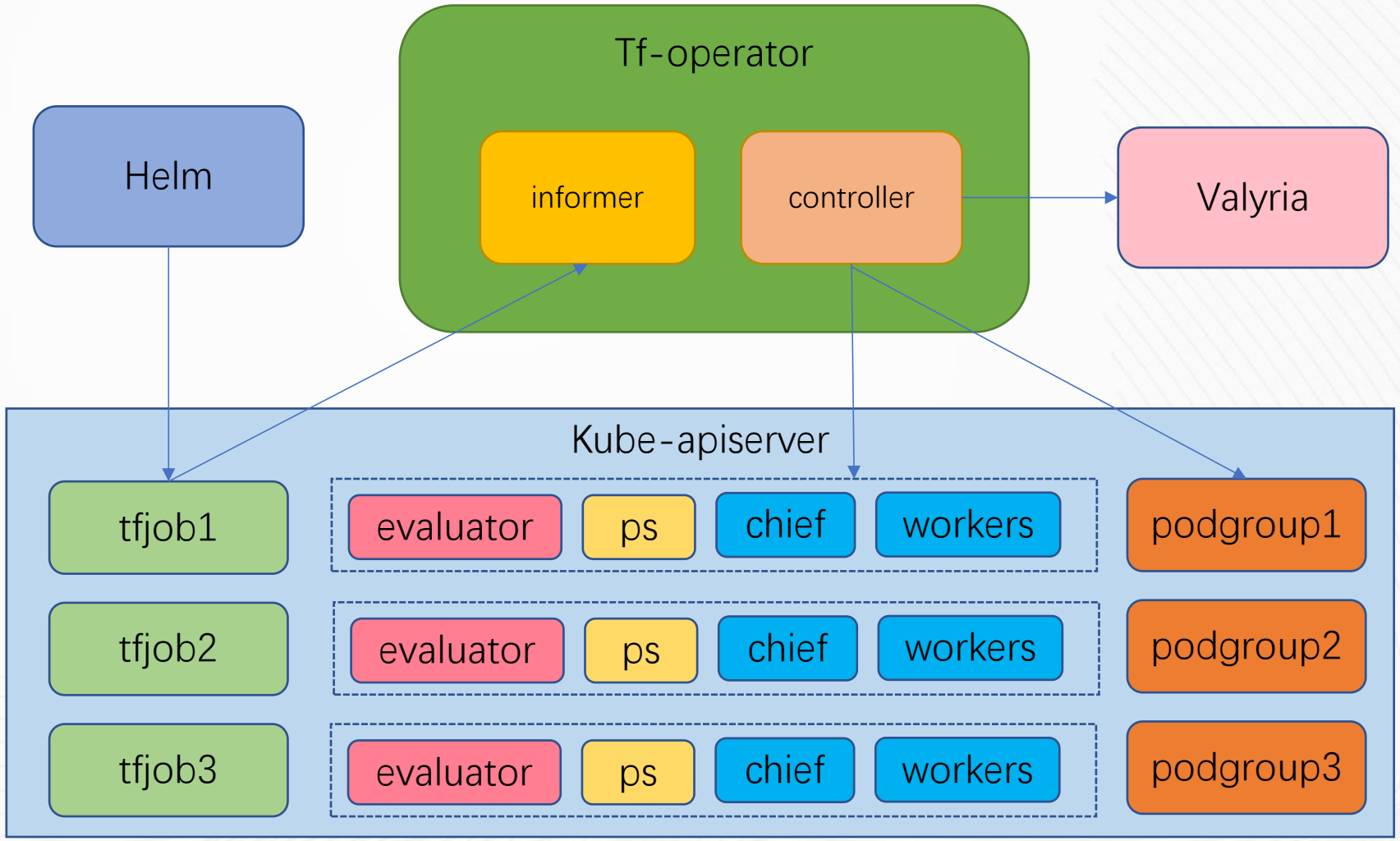
概况



1. 使用helm发布tfjobs，由tf-operator管理训练任务流程
2. 使用volcano对tfjobs的任务容器进行批量调度
3. 对tf-operator进行改造，按业务划分volcano队列，并支持任务优先级
4. 改造kubelet实现资源超卖，提高容器部署密度
5. 通过自研的vpa，实现在容器运行时动态调整资源

Tf-operator部署与使用

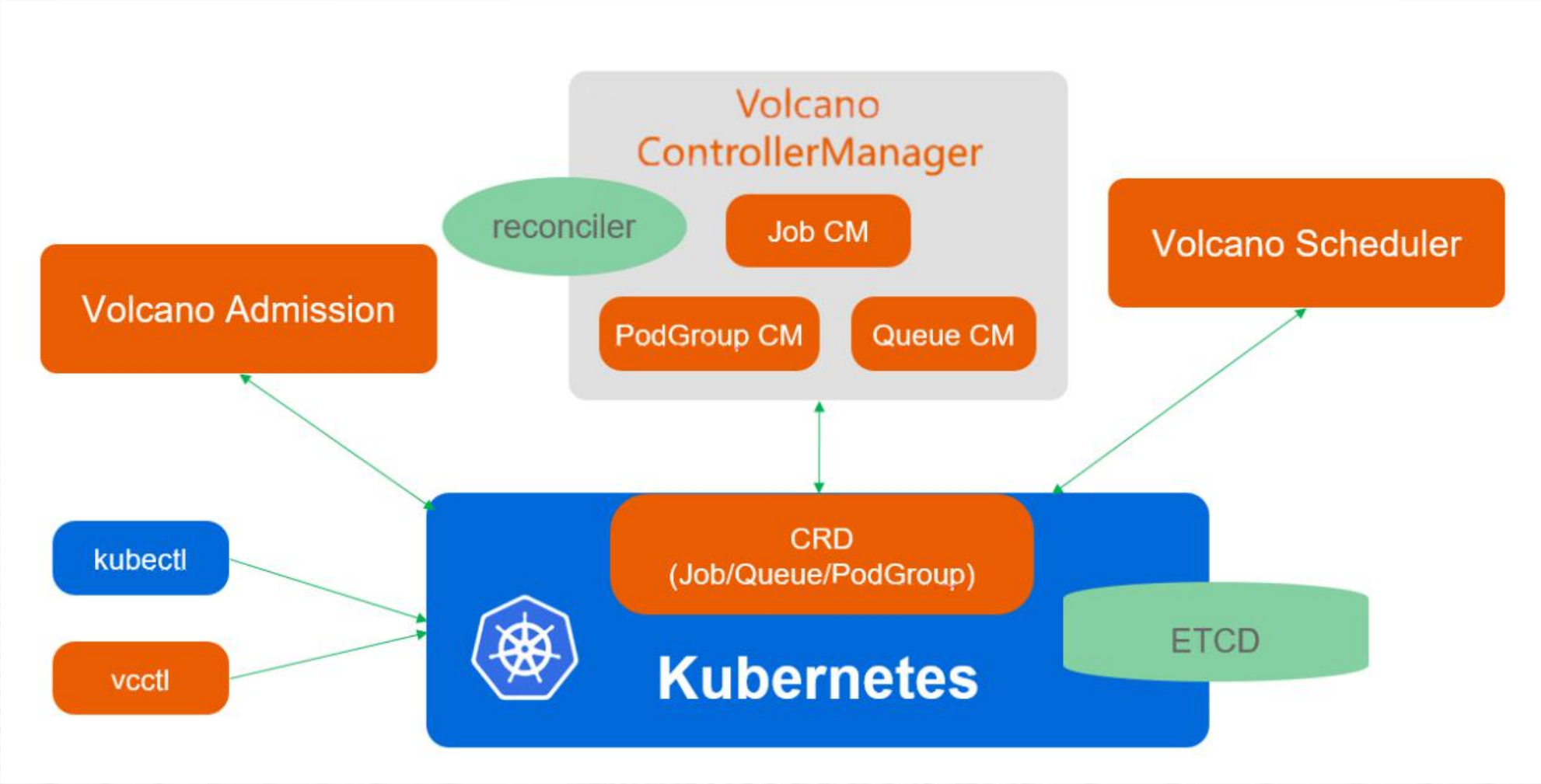
```
apiVersion: "kubeflow.org/v1"
kind: "TFJob"
metadata:
  namespace: "__NAMESPACE__"
  name: "__NAME__“
spec:
  runPolicy:
    schedulingPolicy:
      queue: <queueName>
      priorityClass: <priority>
  tfReplicaSpecs:
    Chief:
      replicas: 1
      template:
          ...
    Evaluator:
      replicas: 1
      template:
          ...
    PS:
      replicas: 1
      template:
          ...
    Worker:
      replicas: 1
      template:
          ...
```



1. tf-operator管理训练任务流程，同步tfjob状态到我们的valyria管理平台
2. 参考社区实现，修改tf-operator，支持创建podgroup时指定queue和priorityClass

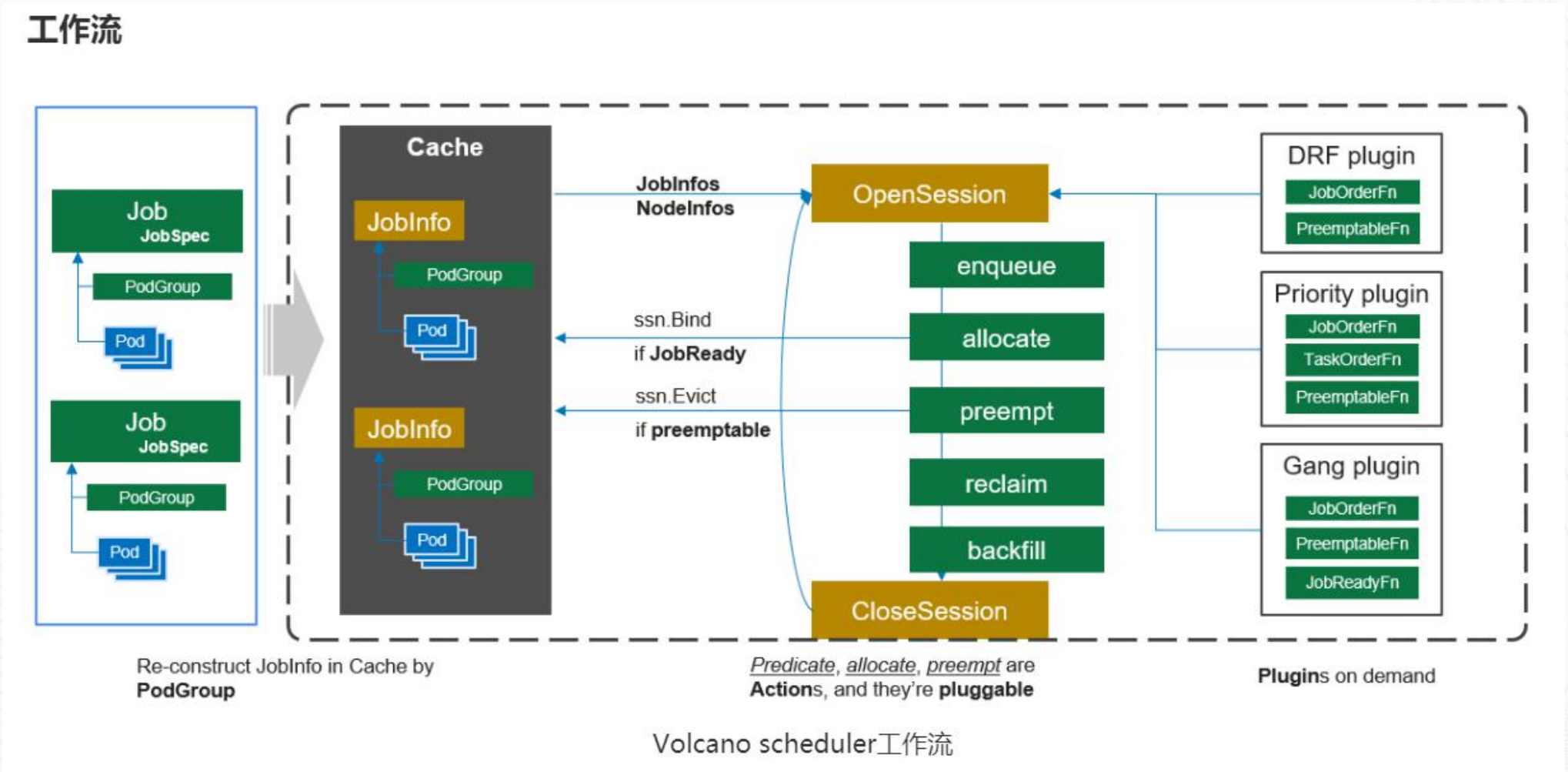
Volcano原理与部署

Volcano架构



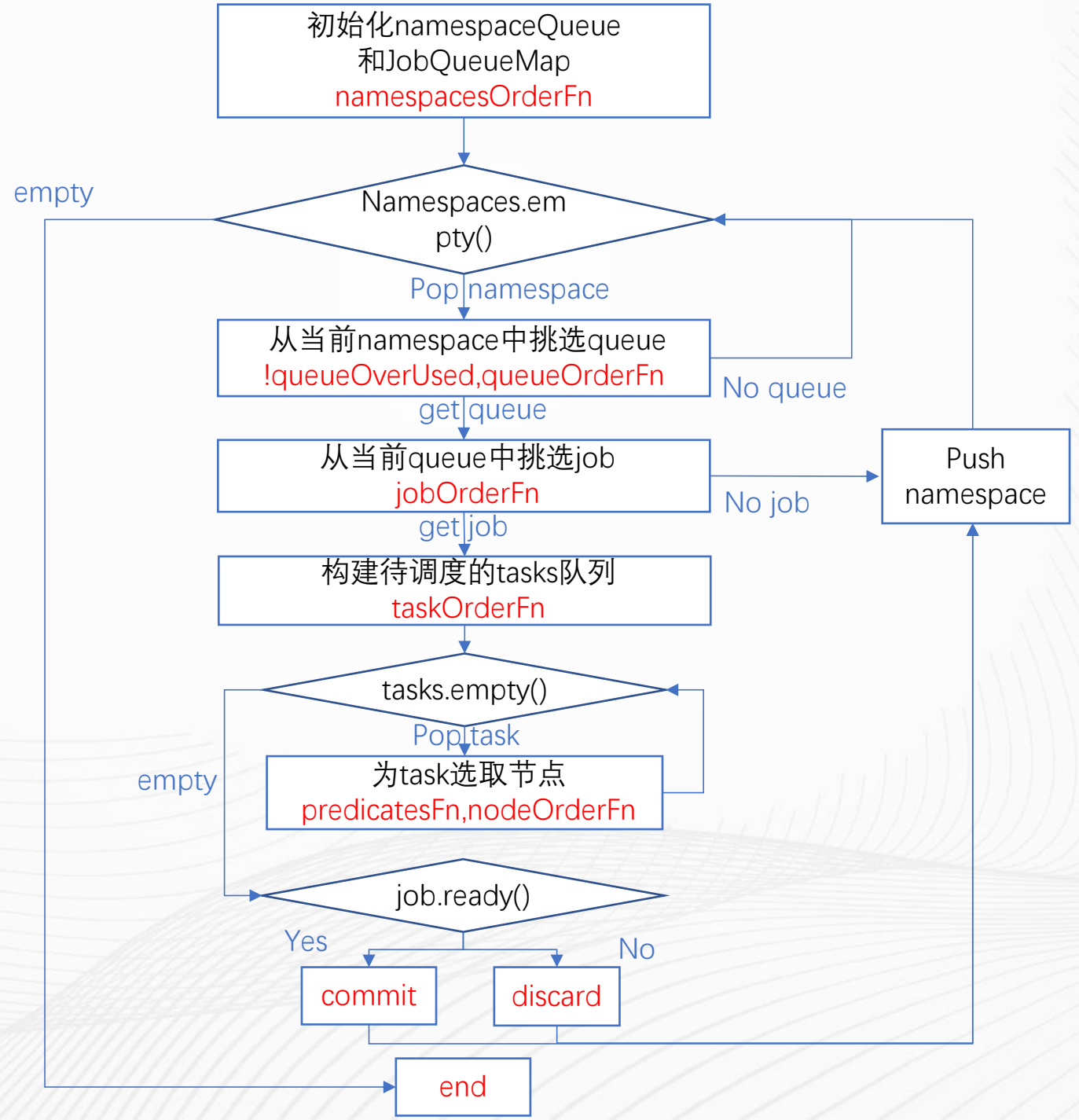
https://volcano.sh/en/docs/architecture/

使用Volcano实现对tfjobs容器的批调度



https://volcano.sh/en/docs/schduler_introduction/

Allocate具体流程



任务队列

GOTC

tfjob

```
apiVersion: "kubeflow.org/v1"
kind: "TFJob"
metadata:
    ...
spec:
  runPolicy:
    schedulingPolicy:
      queue: <queueName>
      priorityClass: <priority>
  tfReplicaSpecs:
    ...
```

queue

```
apiVersion: scheduling.volcano.sh/v1beta1
kind: Queue
metadata:
  name: rec-queue
spec:
  weight: 400
```

```
apiVersion: scheduling.volcano.sh/v1beta1
kind: Queue
metadata:
  name: search-queue
spec:
  weight: 200
```

configmap

```
actions: "enqueue, allocate, backfill"
tiers:
- plugins:
  - name: drf
  - name: priority
  - name: gang
  - name: conformance
- plugins:
  - name: proportion
- plugins:
  - name: nodeorder
  - name: predicates
```

按业务组拆分队列

1.当单一业务组训练任务较多时，可以使用整个集群的所有资源

2.当多个业务组训练任务较多时，按queue的权重分配资源，保证资源不被单一业务组抢占

Queue Overused：derserved < allocated,

其中desearved=min((totalResource*queueWeight)/totalWeight, request)

request = sum(pod resources)

allocated = sum(allocated pod resources)

任务优先级

GOTC

tfjob

```
apiVersion: "kubeflow.org/v1"
kind: "TFJob"
metadata:
    ...
spec:
  runPolicy:
    schedulingPolicy:
      queue: <queueName>
      priorityClass: <priority>
  tfReplicaSpecs:
    ...
```

queue

```
apiVersion: scheduling.k8s.io/v1
kind: PriorityClass
metadata:
  name: p0-pri
value: 1000
```

```
apiVersion: scheduling.k8s.io/v1
kind: PriorityClass
metadata:
  name: p1-pri
value: 100
```

configmap

```
actions: "enqueue, allocate, backfill"
tiers:
- plugins:
  - name: drf
  - name: priority
  - name: gang
  - name: conformance
- plugins:
  - name: proportion
- plugins:
  - name: nodeorder
  - name: predicates
```

优先级低

↓

优先级高

1.Gang插件优先级最高，优先调度未ready的批任务；

2. priority其次，保证优先级较高的任务先调度；

3. drf最后，在同等优先级下，需要资源较少的任务优先调度

如何提高集群利用率

通过使用Volcano的批调度功能，实现了tfjob容器的全量执行，避免了任务碎片化的问题

问题

1.对训练任务进行批调度，需要等待足够资源来调度容器，可能会导致部分宿主机空闲

2.业务对训练任务的计算量估算不准确，申请过多资源导致宿主机空闲

解决方案

1.改造kubelet，对宿主机CPU资源进行超卖，提高集群的容器部署密度

2.实现VPA，监控宿主机上容器的资源利用率，实时动态调整容器资源

集群资源超卖

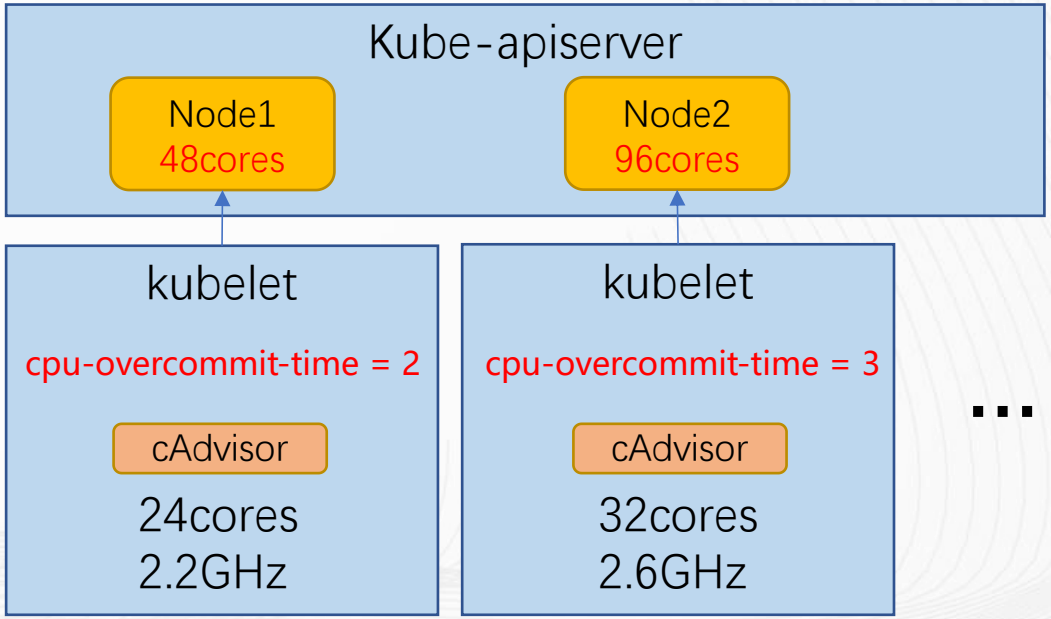
GOTC

通过定制kubelet，增加超卖参数(cpu-overcommit-time),
修改上报到kube-apiserver的宿主机cpu数量，实现CPU超卖

通过CPU超卖，我们可以实现：

1. 抹平不同CPU的性能带来的异构问题

2. 提高容器部署密度，因此提升CPU使用率



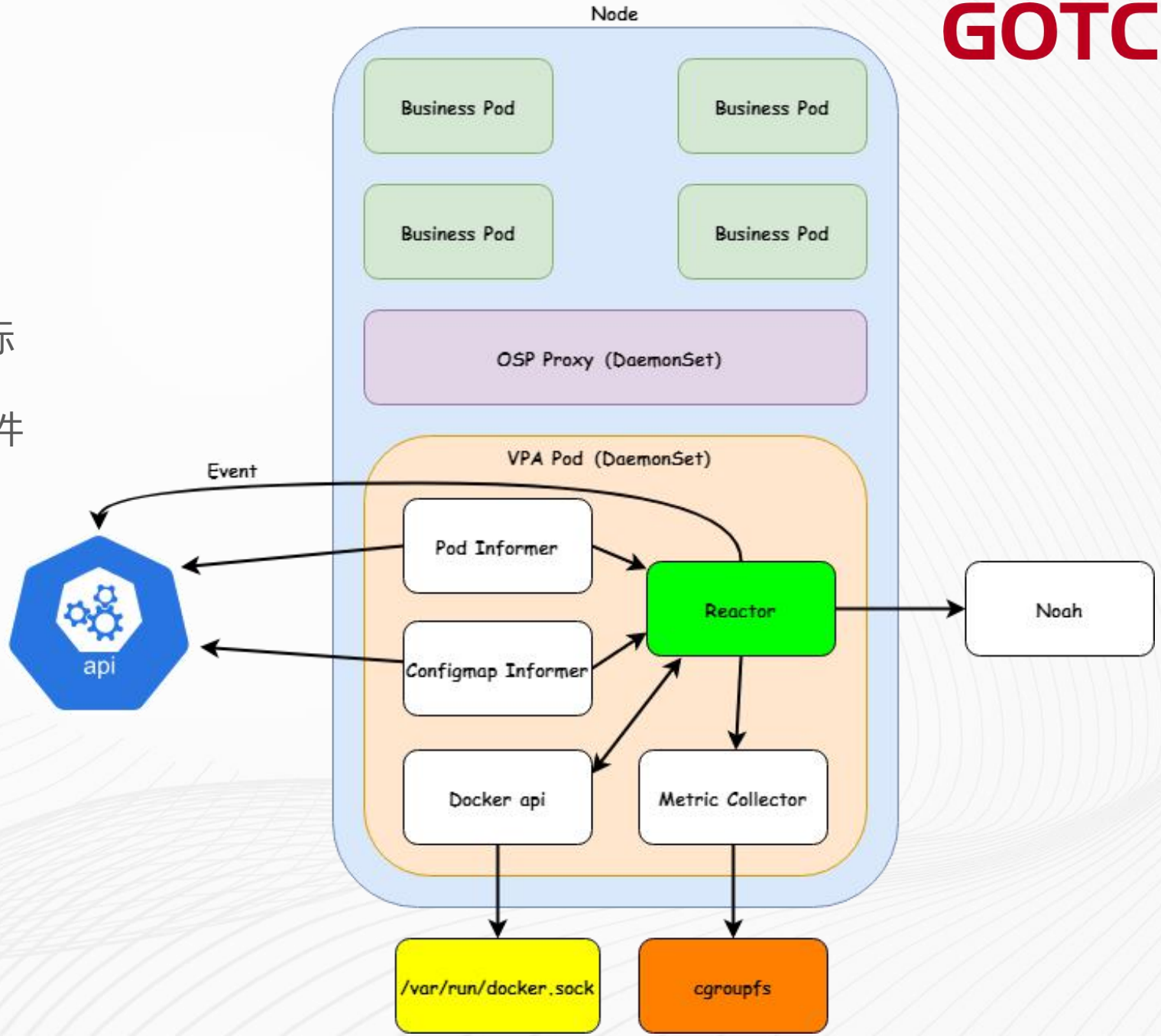
问题

资源超卖导致CPU热点

VPA介绍

VPA设计方案

1. 以Daemonset形式部署，去中心化
2. 实时监控本宿主机容器的cgroup秒级指标
3. 可定制化的handler处理容器和宿主机事件
4. 通过configmap动态配置策略参数
5. 调用docker api实时调整容器资源，无需重启容器



VPA介绍

GOTC

VPA对训练任务容器的处理策略

当宿主机的CPU使用率过高时，对容器的CPU limit进行缩容压制

操作	说明
获取当前宿主机真实cpu使用率和目标cpu使用率	当前宿主机48c，真实cpu使用率95%，过载线90%，警戒线80%，即目标使用率是85%
计算从真实使用率压制到目标使用率需要压制的核数	（95%-85%）* 48c = 4.8c
将宿主机上的容器按训练优先级（P2|P1|P0）及真实cpu使用率倒序排序	podA(P0) 20c，podB(P1) 10c，podC(P2) 6c, podD(P2) 4c 排序后：podC -> podD -> podB -> podA
按排序后按Max(目标压制核数,50%*容器真实cpu使用率)的标准选择需要压制的容器	先选取podC进行压制，其cpu使用量50%为3c，则压制3c，还需要压制1.8c，因此需要将podC的cpu limit调为3c 再选取podD进行压制，其cpu使用量50%为2c，但只需压制1.8c即可，因此需要将podD的cpu limit调为2.2c
修改容器cpu limit并且添加annotation标志容器被压制	每个容器只进行一次压制

VPA介绍

GOTC

VPA对于在线流量容器的操作

情况	操作
宿主机有足够资源时，某个容器的资源使用率到达阈值	对容器的资源进行暂时扩容，当容器的资源使用率下降后回收
宿主机资源到达告警阈值时	不再对容器进行临时扩容
宿主机CPU/网络使用率达到热点阈值时	调用上游LB接口，按优先级降低容器流量
宿主机内存使用率到达热点阈值时	Cordon宿主机

现状与未来工作方向

GOTC

现状

1. 每日运行约200个训练任务，运行时间平均为75分钟，资源等待时间平均为40秒

2. 宿主机CPU使用率平均为50%，90分位到达75%

工作方向

1. 训练任务容器资源监控加强 – 记录各角色容器资源使用率，以此建议用户优化资源申请量

2. Volcano支持按照label选取宿主机 – 生产集群按照label划分宿主机的用途，volcano应该只考虑用于训练任务的宿主机资源

3. VPA功能增强 – 在宿主机资源充足的情况下，对需要资源的容器进行临时扩容，提升资源使用率

GOTC

THANKS

全球开源技术峰会

THE GLOBAL OPENSOURCE TECHNOLOGY CONFERENCE