# Kubernetes

## Kubernetes

- Docker provides individual containers on a local machine
- Kubernetes manages collections of running containers across a cluster/datacenter
- also provides networking, storage, monitoring, service discovery

## The Cluster

A cluster with 6 CPU nodes and 8 GPU nodes (running on Google GCE).

In [1]:

```
# make sure we have a running cluster
kubectl get nodes
```

```
NAME                                          STATUS   ROLES    AGE   VERSION
gke-tmb-cluster-default-pool-7a7b0dec-71mc   Ready    <none>   11h   v1.13.11-gke.14
gke-tmb-cluster-default-pool-7a7b0dec-8xm3   Ready    <none>   11h   v1.13.11-gke.14
gke-tmb-cluster-default-pool-7a7b0dec-fdkw   Ready    <none>   11h   v1.13.11-gke.14
gke-tmb-cluster-default-pool-7a7b0dec-m2b7   Ready    <none>   11h   v1.13.11-gke.14
gke-tmb-cluster-default-pool-7a7b0dec-whjj   Ready    <none>   11h   v1.13.11-gke.14
gke-tmb-cluster-default-pool-7a7b0dec-wzq1   Ready    <none>   11h   v1.13.11-gke.14
gke-tmb-cluster-gpus-6c20b4bb-9w8f           Ready    <none>   11h   v1.13.11-gke.14
gke-tmb-cluster-gpus-6c20b4bb-bm56           Ready    <none>   11h   v1.13.11-gke.14
gke-tmb-cluster-gpus-6c20b4bb-mfrs           Ready    <none>   11h   v1.13.11-gke.14
gke-tmb-cluster-gpus-6c20b4bb-pzm4           Ready    <none>   11h   v1.13.11-gke.14
gke-tmb-cluster-gpus-6c20b4bb-r7tc           Ready    <none>   11h   v1.13.11-gke.14
gke-tmb-cluster-gpus-6c20b4bb-t4r9           Ready    <none>   11h   v1.13.11-gke.14
gke-tmb-cluster-gpus-6c20b4bb-tx26           Ready    <none>   11h   v1.13.11-gke.14
gke-tmb-cluster-gpus-6c20b4bb-zjp1           Ready    <none>   11h   v1.13.11-gke.14
```

In [2]:

```
kubectl delete jobs --all
kubectl delete pods --all
```

```
No resources found
pod "myjob-z6f49" deleted
```

## Pods

- Kubernetes groups containers into *pods*
- (Docker container = whale, Pod = group of whales)
- specifications are written in YAML or JSON

In [5]:

```
kubectl delete pod/mypod || true
kubectl apply -f - <<'EOF'
apiVersion: v1
kind: Pod
metadata:
  name: mypod
spec:
  containers:
  - name: mypod
    image: gcr.io/research-191823/bigdata19
    command: ["nvidia-smi"]
    resources:
      limits:
        nvidia.com/gpu: "1"
  restartPolicy: Never
EOF
```

```
Error from server (NotFound): pods "mypod" not found
pod/mypod created
```

# Pod Status and Logs

The Kubernetes runtime keeps track of pod status and logs.

In [6]:

```
kubectl get pods
```

```
NAME    READY   STATUS              RESTARTS   AGE
mypod   0/1     ContainerCreating   0          0s
```

In [7]:

```
sleep 15
```

In [8]:

```
kubectl logs pod/mypod
```

```
Mon Dec  9 16:33:29 2019
+-----------------------------------------------------------------------------+
| NVIDIA-SMI 418.67       Driver Version: 418.67       CUDA Version: 10.1     |
|-------------------------------+----------------------+----------------------+
| GPU  Name        Persistence-M| Bus-Id        Disp.A | Volatile Uncorr. ECC |
| Fan  Temp  Perf  Pwr:Usage/Cap|         Memory-Usage | GPU-Util  Compute M. |
|===============================+======================+======================|
|   0  Tesla T4            Off  | 00000000:00:04.0 Off |                    0 |
| N/A   40C    P8    10W /  70W |      0MiB / 15079MiB |      0%      Default |
+-------------------------------+----------------------+----------------------+

+-----------------------------------------------------------------------------+
| Processes:                                                       GPU Memory |
|  GPU       PID   Type   Process name                             Usage      |
|=============================================================================|
|  No running processes found                                                 |
+-----------------------------------------------------------------------------+
```

# Debugging Pod Startup Problems

Sometimes pods don't get scheduled (never start running). Here are some tricks to debug this.

In [9]:

```
# sometimes pods don't schedule; there is tons of info
kubectl describe pod/mypod | sed 10q
kubectl describe pod/mypod | echo ... $(wc -l) ...
```

```
Name:               mypod
Namespace:          default
Priority:           0
PriorityClassName:  <none>
Node:               gke-tmb-cluster-gpus-6c20b4bb-9w8f/10.138.0.75
Start Time:         Mon, 09 Dec 2019 08:33:28 -0800
Labels:             <none>
Annotations:        kubectl.kubernetes.io/last-applied-configuration:
                      {"apiVersion":"v1","kind":"Pod","metadata":{"annotations":{},"name":"mypod","n
amespace":"default"},"spec":{"containers":[{"command":["nvid...
                    kubernetes.io/limit-ranger: LimitRanger plugin set: cpu request for container my
pod
... 60 ...
```

In [10]:

```
# the Events: section usually tells you why a job didn't get assigned to a node

kubectl describe pod/mypod | grep -A100 Events:
```

```
Events:
  Type    Reason     Age   From                                          Message
  ----    ------     ----  ----                                          -------
  Normal  Scheduled  17s   default-scheduler                             Successfully assigned defaul
t/mypod to gke-tmb-cluster-gpus-6c20b4bb-9w8f
  Normal  Pulling    16s   kubelet, gke-tmb-cluster-gpus-6c20b4bb-9w8f  pulling image "gcr.io/resear
ch-191823/bigdata19"
  Normal  Pulled     16s   kubelet, gke-tmb-cluster-gpus-6c20b4bb-9w8f  Successfully pulled image "g
cr.io/research-191823/bigdata19"
  Normal  Created    16s   kubelet, gke-tmb-cluster-gpus-6c20b4bb-9w8f  Created container
  Normal  Started    16s   kubelet, gke-tmb-cluster-gpus-6c20b4bb-9w8f  Started container
```

In [11]:

```
# nodes also have descriptions (even longer)

node=$(kubectl get nodes | awk '/gpus/{print $1; exit}')
kubectl describe node/$node | sed 10q
kubectl describe node/$node | echo ... $(wc -l) ...
```

```
Name:               gke-tmb-cluster-gpus-6c20b4bb-9w8f
Roles:              <none>
Labels:             beta.kubernetes.io/arch=amd64
                    beta.kubernetes.io/fluentd-ds-ready=true
                    beta.kubernetes.io/instance-type=n1-standard-16
                    beta.kubernetes.io/os=linux
                    cloud.google.com/gke-accelerator=nvidia-tesla-t4
                    cloud.google.com/gke-nodepool=gpus
                    cloud.google.com/gke-os-distribution=cos
                    failure-domain.beta.kubernetes.io/region=us-west1
... 85 ...
```

In [12]:

```
# you want to make sure that nodes have the right allocatable resources
kubectl describe node/$node | grep -A10 Allocatable:
```

```
Allocatable:
 attachable-volumes-gce-pd:  127
 cpu:                        15890m
 ephemeral-storage:          47093746742
 hugepages-2Mi:              0
 memory:                     56288600Ki
 nvidia.com/gpu:             1
 pods:                       110
System Info:
 Machine ID:                 265593ea8efdb186402965bc1163ba81
 System UUID:                265593EA-8EFD-B186-4029-65BC1163BA81
```

In [13]:

```
# also make sure there are resources available
kubectl describe node/$node | grep -A10 Allocated
```

```
Allocated resources:
  (Total limits may be over 100 percent, i.e., overcommitted.)
  Resource                   Requests    Limits
  --------                   --------    ------
  cpu                        400m (2%)   1050m (6%)
  memory                     210Mi (0%)  510Mi (0%)
  ephemeral-storage          0 (0%)      0 (0%)
  attachable-volumes-gce-pd  0           0
  nvidia.com/gpu             0           0
Events:                      <none>
```

In [15]:

```
# nodes can be prevented from scheduling jobs by "taints"
kubectl describe node/$node | grep -A2 Taints:
```

```
Taints:             nvidia.com/gpu=present:NoSchedule
Unschedulable:      false
Conditions:
```

In [14]:

```
# only pods that tolerate the taints are scheduled
kubectl describe pod/mypod | grep -A5 Tolerations:
```

```
Tolerations:     node.kubernetes.io/not-ready:NoExecute for 300s
                 node.kubernetes.io/unreachable:NoExecute for 300s
                 nvidia.com/gpu:NoSchedule
Events:
  Type    Reason     Age   From                                             Message
  ----    ------     ----  ----                                             -------
```

In [16]:

```
kubectl delete pods --all
```

```
pod "mypod" deleted
```

# Jobs

Jobs are like batch queuing. Job specs are a wrapper around pod specs.

In [17]:

```
kubectl delete job/myjob || true
kubectl apply -f - <<'EOF'
apiVersion: batch/v1
kind: Job
metadata:
  name: myjob
  labels:
    app: bigdata19
spec:
  backoffLimit: 0
  template:
    # below is a regular Pod spec
    spec:
      containers:
        - name: myjob
          image: gcr.io/research-191823/bigdata19
          command:
            - "/bin/bash"
            - "-c"
            - |
              nvidia-smi
          stdin: true
          tty: true
          resources:
            limits:
              nvidia.com/gpu: "1"
      restartPolicy: Never
EOF
```

```
Error from server (NotFound): jobs.batch "myjob" not found
job.batch/myjob created
```

In [18]:

```
sleep 15
```

In [19]:

```
kubectl logs job.batch/myjob
```

```
Mon Dec  9 16:33:53 2019
+-----------------------------------------------------------------------------+
| NVIDIA-SMI 418.67       Driver Version: 418.67       CUDA Version: 10.2     |
|-------------------------------+----------------------+----------------------+
| GPU  Name        Persistence-M| Bus-Id        Disp.A | Volatile Uncorr. ECC |
| Fan  Temp  Perf  Pwr:Usage/Cap|         Memory-Usage | GPU-Util  Compute M. |
|===============================+======================+======================|
|   0  Tesla T4            Off  | 00000000:00:04.0 Off |                    0 |
| N/A   43C    P0    25W /  70W |      0MiB / 15079MiB |      5%      Default |
+-------------------------------+----------------------+----------------------+

+-----------------------------------------------------------------------------+
| Processes:                                                       GPU Memory |
|  GPU       PID   Type   Process name                             Usage      |
|=============================================================================|
|  No running processes found                                                 |
+-----------------------------------------------------------------------------+
```

In [20]:

```
kubectl get jobs
```

```
NAME    COMPLETIONS   DURATION   AGE
myjob   1/1           3s         16s
```

In [21]:

```
kubectl delete jobs --all
```

```
job.batch "myjob" deleted
```

# Configmaps

In [22]:

```
# configmaps are little mountable file systems, for config information and scripts
# we put our Python scripts there
kubectl delete configmap files || true
kubectl create configmap files \
--from-file=training.py=training.py \
--from-file=helpers.py=helpers.py
```

```
configmap "files" deleted
configmap/files created
```

# Running a Training Job

In [23]:

```
# with the scripts transferred, let's run actual training
# note the use of multi-line quoting for the shell script
kubectl delete job/myjob || true
kubectl apply -f - <<'EOF'
apiVersion: batch/v1
kind: Job
metadata:
  name: myjob
  labels:
    app: bigdata19
spec:
  backoffLimit: 0
  template:
    spec:
      containers:
        - name: myjob
          image: gcr.io/research-191823/bigdata19
          command:
            - "/bin/bash"
            - "-c"
            - |
              cp /files/*.py .
              python3 training.py
          stdin: true
          tty: true
          resources:
            limits:
              nvidia.com/gpu: "1"
          volumeMounts:
            - mountPath: /files
              name: files
      restartPolicy: Never
      volumes:
        - configMap:
            name: files
          name: files
EOF
```

```
Error from server (NotFound): jobs.batch "myjob" not found
job.batch/myjob created
```

# Training Job

In [24]:

```
kubectl get jobs
```

```
NAME    COMPLETIONS   DURATION   AGE
myjob   0/1           1s         1s
```

In [25]:

```
sleep 30
```

In [26]:

```
kubectl logs job/myjob
```

```
/opt/conda/lib/python3.6/site-packages/torchvision/io/_video_opt.py:17: UserWarning: video reader ba
sed on ffmpeg c++ ops not available
  warnings.warn("video reader based on ffmpeg c++ ops not available")
Mon Dec  9 16:34:14 UTC 2019; myjob-fgflt; root; /workspace; GPU 0: Tesla T4 (UUID: GPU-7ffd1122-5cc
e-b8a1-db96-99d9e51ebbc8);
creating resnet50
        0 bs   128 per sample loss 5.57e-02 loading 8.92e-03 training 1.51e-02
      896 bs   128 per sample loss 5.56e-02 loading 5.56e-03 training 8.87e-03
     1792 bs   128 per sample loss 5.54e-02 loading 3.94e-03 training 5.89e-03
```

In [27]:

```
kubectl delete jobs --all || true
kubectl delete pods --all || true
```

```
job.batch "myjob" deleted
pod "myjob-fgflt" deleted
```

# Kubernetes

- a way of running services and jobs on a cluster of machines
- configurations are given as JSON or YAML files (or via APIs)
- both CPUs and GPUs supported

In [ ]:

```

```