Dynamic Big Data in the Cloud with Kubernetes

Frank Conrad
Architect

apomaya.com, voicea.ai

scalable efficient low latency processing

frank@fc-tb.de, frank@apomaya.com, frankc@voicea.ai

issues that we will be talking about today

- Process the data faster!
  - Deliver those reports earlier!
- lots of different job profiles (cpu bound or memory bound)
  - cluster not fully utilized
- job runtime is not deterministic
  - interfering with other pods
- my job doesn’t run well, because some pods get delayed

Copyright © 2019 Frank Conrad

agenda

- why kubernetes?
- getting started
- cluster autoscaler / node pools
- tips and tricks for autoscaling
- batch jobs on kubernetes
- final words

Copyright © 2019 Frank Conrad

kubernetes (k8s)

- just another cluster?
- container, deployment, orchestration
- supports good CI/CD
- built-in real dynamic cluster (cluster autoscaler)
- good cloud support
- provider, vendor agnostic API / usage
- leverages the cloud better by resource/utilization based billing
- simpler version handling / migration for apps
- help leverage scale of map-reduce pattern in cloud

Copyright © 2019 Frank Conrad

cloud

- allow dynamic resource changes (instances,…)
- makes operations easier by providing same management for us
- provider for managed kubernetes (backplane)
  - google / gke
  - aws / eks
  - azure / aks
  - digitalocean,….

Copyright © 2019 Frank Conrad

agenda

- why kubernetes?
- getting started
- cluster autoscaler / node pools
- tips and tricks for autoscaling
- batch jobs on kubernetes
- final words

Copyright © 2019 Frank Conrad

Tools for deployment
my recommendation

- to set up a cluster / infrastructure
  - cloud provider tools (gcloud, eksctl,…)
  - terraform
- to manage and update k8s cluster/apps
  - helm / helmfile
  - kubectl
  - operators
- starting point to search for tools https://github.com/ramitsurana/awesome-kubernetes

Copyright © 2019 Frank Conrad

helm / helmfile

- helm
  - the package manger for k8s (https://hub.helm.sh/, https://github.com/helm/charts)
  - chart
    - templating
    - values
  - chart repository is only optional (but very helpful in larger orgs)
- helmfile
  - templating on values
- helm-tiller

Copyright © 2019 Frank Conrad

operator

- k8s operator is a pattern to put app / SRE knowledge into code, to automate
- can handle special operational jobs, which are not handled by standard k8s functions
  - can run cassandra, postgres, prometheus-grafana, airflow, spark,…
  - backups, updates, scaling
- runs on top of k8s API

Copyright © 2019 Frank Conrad

agenda

- why kubernetes?
- getting started
- cluster autoscaler / node pools
- tips and tricks for autoscaling
- batch jobs on kubernetes
- final words

Copyright © 2019 Frank Conrad

cluster autoscaler

- scales cluster based on the demand
  - up
    - if scheduler fails to find free slot for pod
    - find the best fitting node pool and add nodes
  - down
    - nodes not utilised
    - under-utilised nodes, pods get evicted, if possible
    - normal k8s don’t reschedule pods

Copyright © 2019 Frank Conrad

node pools

- group of nodes with same spec and behavior
- can be static or dynamic (cluster autoscaler)
- how to target pods to nodes (node pools):
  node affinity / node selector
- how to limit pods going to nodes:
  taints
- based on ASG, instance groups,… depend on provider

Copyright © 2019 Frank Conrad

dedicated node pool

- by default k8s tries to use any node for any pod that fits
- node spec must fit your job requirements
- job components can run on different node pools
- can have many node pools
- each node pool can scale to zero

Copyright © 2019 Frank Conrad

target dedicated node pool

- done at pod schedule time
- only allow specific pod via taints
  - dedicated=special01:NoSchedule
    preemptive=true:NoSchedule

```
tolerations:
- effect: NoSchedule
  key: "dedicated"
  operator: Equal
  value: "special01"
- effect: NoSchedule
  key: "preemptive"
  operator: Equal
  value: "true"
```

Copyright © 2019 Frank Conrad

target dedicated node pool

- target pods to a pool via affinity / nodeselector
  - dedicated=special01 (more intuitive, better grafana selection)
  - special01=dedicated (add for better use in node selector)

```
affinity:
  nodeAffinity:
    requiredDuringSchedulingIgnoredDuringExecution:
      nodeSelectorTerms:
      - matchExpressions:
        - key: "dedicated"
          operator: In
          values:
          - "special01"
```

Copyright © 2019 Frank Conrad

hint for managing node pools

- don’t use default labels / direct pool name
  - use schema above, to make node pool changes without change pod settings
    - pool name: special01-b or special01-g
  - allow simpler changes
    - node pools are typical for many settings (labels, taints) immutable
    - create new pools with update specs
    - fallback pools
  - smooth migration (cordon old pool, make pool static)

Copyright © 2019 Frank Conrad

node pool fallback

- when limiting pods to specific pools
  - what to do if the pool hits limits or fails? e.g.
    - no available spot / preemptive instances
    - max’ed out available nodes in pool
    - provider run out of capacity (rare, but it happens!)
  - behavior of cluster autoscaler
  - via affinity

Copyright © 2019 Frank Conrad

node pool fallback via affinity

```
affinity:
  nodeAffinity:
    requiredDuringSchedulingIgnoredDuringExecution:
    …
    preferredDuringSchedulingIgnoredDuringExecution:
    - preference:
      - matchExpressions:
        - key: "dedicated"
          operator: In
          values:
          - "special01"
      weight: 100
    - preference:
      - matchExpressions:
        - key: "dedicated"
          operator: In
          values:
          - "fallback01"
      weight: 10
```

Copyright © 2019 Frank Conrad

node pool network optimization

- lower network latency inside zone
- you pay for cross-zone network traffic
- any zone failure will end up causing recomputation of affected jobs
- constrain node pool definition to a single zone
- define via affinity weights similar fallback

Copyright © 2019 Frank Conrad

agenda

- why kubernetes?
- getting started
- cluster autoscaler / node pools
- tips and tricks for autoscaling
- batch jobs on kubernetes
- final words

Copyright © 2019 Frank Conrad

cluster autoscaler aware pod

- if you use local store (emtpyDir,…), to allow eviction
  - add label: "cluster-autoscaler.kubernetes.io/safe-to-evict": "true"
- use PodDisruptionBudget (PDB) to tell the autoscaler about eviction rules
- if no eviction allowed: "cluster-autoscaler.kubernetes.io/safe-to-evict": „false"
  - for spark use this
- see FAQ: https://github.com/kubernetes/autoscaler/blob/master/cluster-autoscaler/FAQ.md

Copyright © 2019 Frank Conrad

over provision

- drive cluster autoscaler via:
  - dummy pods requesting resource with lower scheduling priority (PriorityClass). Get evicted if real load starts
    - via deployment, get const amount of pods requesting resources to do over provision
    - via separate pod started, one time
      should have max lifetime (sleep X)
  - dummy pods which force scale via pod anti affinity

Copyright © 2019 Frank Conrad

agenda

- why kubernetes?
- getting started
- cluster autoscaler / node pools
- tips and tricks for autoscaling
- batch jobs on kubernetes
- final words

Copyright © 2019 Frank Conrad

spark operator

- https://github.com/radanalyticsio/spark-operator
  - managing the Spark clusters in Kubernetes and OpenShift
- https://github.com/GoogleCloudPlatform/spark-on-k8s-operator
  - the most sophisticated operator for spark apps
  - give currently best k8s integration
    - affinity, taints,…

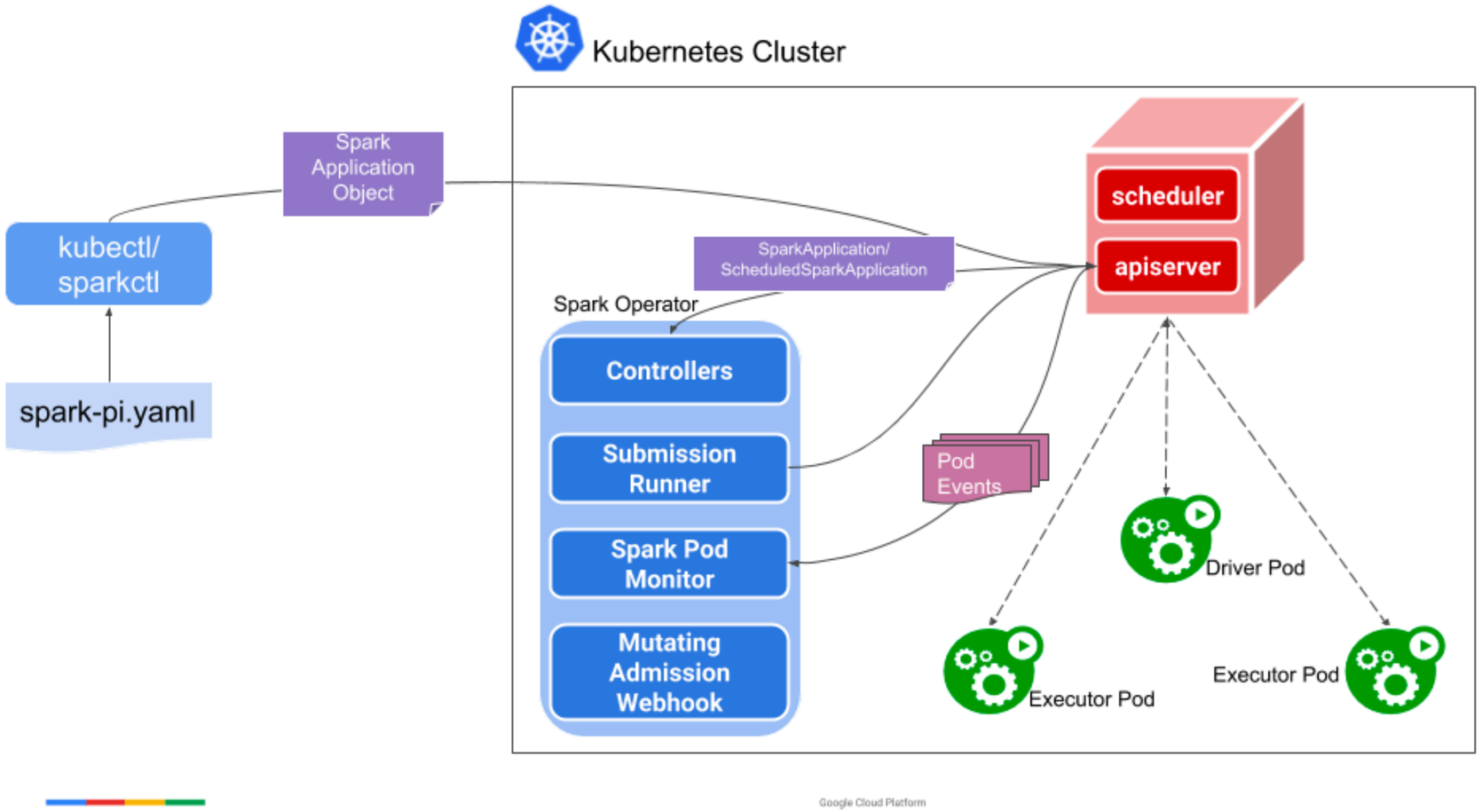
Copyright © 2019 Frank Conrad

spark-on-k8s-operator

- spark 2.3 and up
- translate SparkApplication into spark-submit (cluster per job)
- cron support
- sparkctl tool to simplify usage
- helm to install incubator/sparkoperator
- enhanced failure handling
- support metric exporting to prometheus
- support driver UI access and ingress

Copyright © 2019 Frank Conrad

spark-on-k8s-operator



source: https://github.com/GoogleCloudPlatform/spark-on-k8s-operator/blob/master/docs/architecture-diagram.png

Copyright © 2019 Frank Conrad

spark-pi.yaml

```
apiVersion: "sparkoperator.k8s.io/v1beta1"
kind: SparkApplication
metadata:
  name: spark-pi
  namespace: spark-operator
spec:
  type: Scala
  mode: cluster
  image: "gcr.io/spark-operator/spark:v2.4.0"
  imagePullPolicy: Always
  mainClass: org.apache.spark.examples.SparkPi
  mainApplicationFile: "local:///opt/spark/examples/jars/spark-examples_2.11-2.4.0.jar"
  sparkVersion: "2.4.0"
…
  driver:
    cores: 0.1
    coreLimit: "200m"
    memory: "512m"


…
  executor:
    cores: 1
    instances: 1
    memory: "512m"
…
    affinity:
      nodeAffinity:
        requiredDuringSchedulingIgnoredDuringExecution:
…
    tolerations:
    - effect: NoSchedule
      key: "dedicated"
      operator: Equal
      value: "special01"
```

Copyright © 2019 Frank Conrad

spark podtemplate

- allows you to customise pods with anything (driver/executor)
  - affinity
  - tainted
- https://issues.apache.org/jira/browse/SPARK-24434
- is in master but not released

Copyright © 2019 Frank Conrad

history server

- install via helm stable/spark-history-server
- works together with spark-on-k8s-operator
- NFS can use GCP filestore

Copyright © 2019 Frank Conrad

agenda

- why kubernetes?
- getting started
- cluster autoscaler / node pools
- tips and tricks for autoscaling
- batch jobs on kubernetes
- final words

Copyright © 2019 Frank Conrad

interesting stuff

- evict pods when spot instance terminates
  - https://github.com/pusher/k8s-spot-termination-handler
  - on aws 3 min - enough time to run checkpoint , gke only 30 sec - careful!
- move pods to spot instances
  - https://github.com/pusher/k8s-spot-rescheduler

Copyright © 2019 Frank Conrad

hints

- don’t write to image
- use emptyDir or (local) volumes
  - emptyDir: memory can be an accelerator
  - always specify size

Copyright © 2019 Frank Conrad

large images

- when images get to multiple GB in size…
  - first, try to optimise image and download
  - run docker pull in init container
  - for parts like ML models,…
    - use NFS server (gcp filestore,…)
    - mount RO volumes

Copyright © 2019 Frank Conrad

summary

- using kubernetes and cloud, it is now possible to run batch jobs on demand
- size your resources for the specific job using node pools
- leverage scale to get results faster, at a similar cost
- run resources only when you use them

Copyright © 2019 Frank Conrad

Think different

Thank you

Questions?

Thanks to Adam Shepard for supporting this talk.

Copyright © 2019 Frank Conrad

k8s scheduler

- kubernetes allows for a custom scheduler or extender
- there are schedulers to addresses batch problems (all or nothing)
  - https://github.com/kubernetes-sigs/kube-batch
  - https://github.com/palantir/k8s-spark-scheduler
  - https://github.com/atlassian/escalator

Copyright © 2019 Frank Conrad

flink operator

- https://github.com/iSofiane/flink-on-k8s-operator
  - looks dead
- https://github.com/lightbend/flink-k8s-operator
  - young project, can see the future of running flink jobs
  - has helm chart to deploy
  - support prometheus
- Zalando has good suggestions to run flink on k8s

Copyright © 2019 Frank Conrad

airflow on k8s

- now supports k8s executor
  - run a task directly as a pod
- k8s airflow operator https://github.com/GoogleCloudPlatform/airflow-operator
- alternative helm chart stable/airflow
- good example of how in the future tasks could be handled for build systems…

Copyright © 2019 Frank Conrad