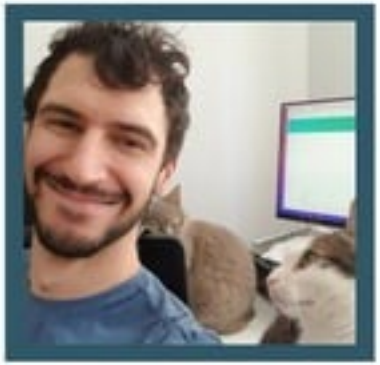
Introduction to Kafka Cruise Control

Viktor Somogyi-Vass



Viktor Somogyi-Vass

viktor.somogyi@cloudera.com

Operating Kafka clusters are hard.

1

Surprises

There can be data center outages, maintenance can go wrong and upgrades can go wrong.

1

Surprises

There can be data center outages, maintenance can go wrong and upgrades can go wrong.

2

Scaling

New brokers won't get populated. What to put on them? Where to put partitions when decommissioning brokers?

1

Surprises

There can be data center outages, maintenance can go wrong and upgrades can go wrong.

2

Scaling

New brokers won't get populated. What to put on them? Where to put partitions when decommissioning brokers?

3

Performance

Brokers can be overloaded or even underloaded. Rebalances can ruin your cluster.

How do we solve these problems?

SOME HIGHLIGHTS

SOME HIGHLIGHTS

SELF-HEALING

By rebalances

Detect—failures and anomalies are detected automatically

Rebalance—healing will be done via rebalancing partitions to healthy brokers

SOME HIGHLIGHTS

SELF-HEALING
By rebalances

Detect—failures and anomalies are detected automatically

Rebalance—healing will be done via rebalancing partitions to healthy brokers

ADMIN
Scaling and maintenance

Upscaling—brokers can be added to Cruise Control and populated with partitions

Downscaling—brokers can be removed from the cluster and partitions will be reassigned optimally

SOME HIGHLIGHTS

SELF-HEALING
By rebalances

Detect—failures and anomalies are detected automatically

Rebalance—healing will be done via rebalancing partitions to healthy brokers

ADMIN
Scaling and maintenance

Upscaling—brokers can be added to Cruise Control and populated with partitions

Downscaling—brokers can be removed from the cluster and partitions will be reassigned optimally

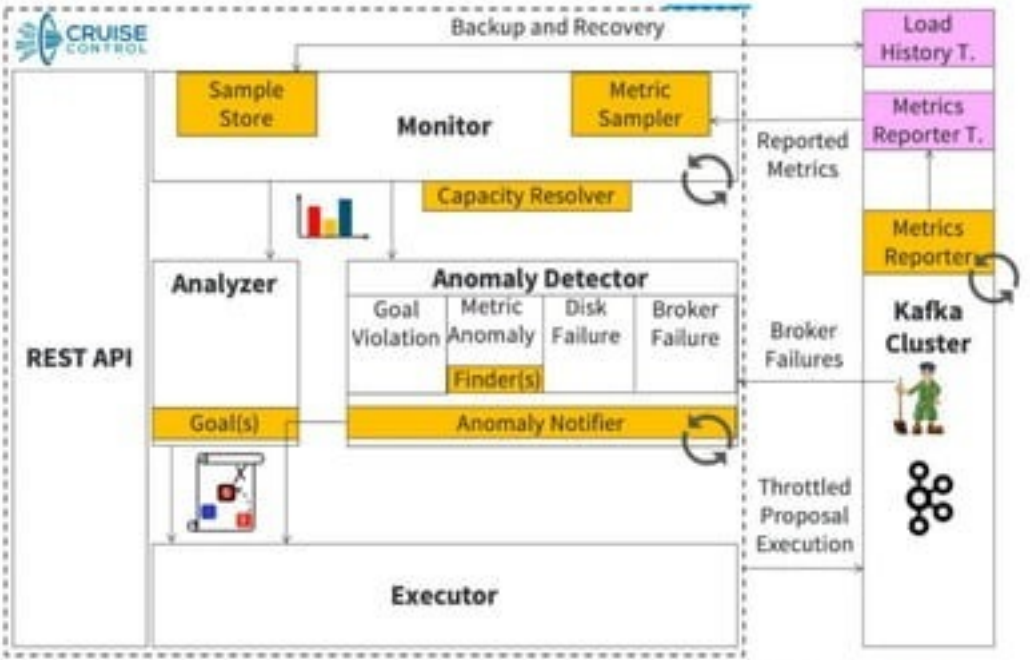
MONITORING
Even load everywhere

Metrics—are collected periodically to ensure an up-to-date view of the cluster

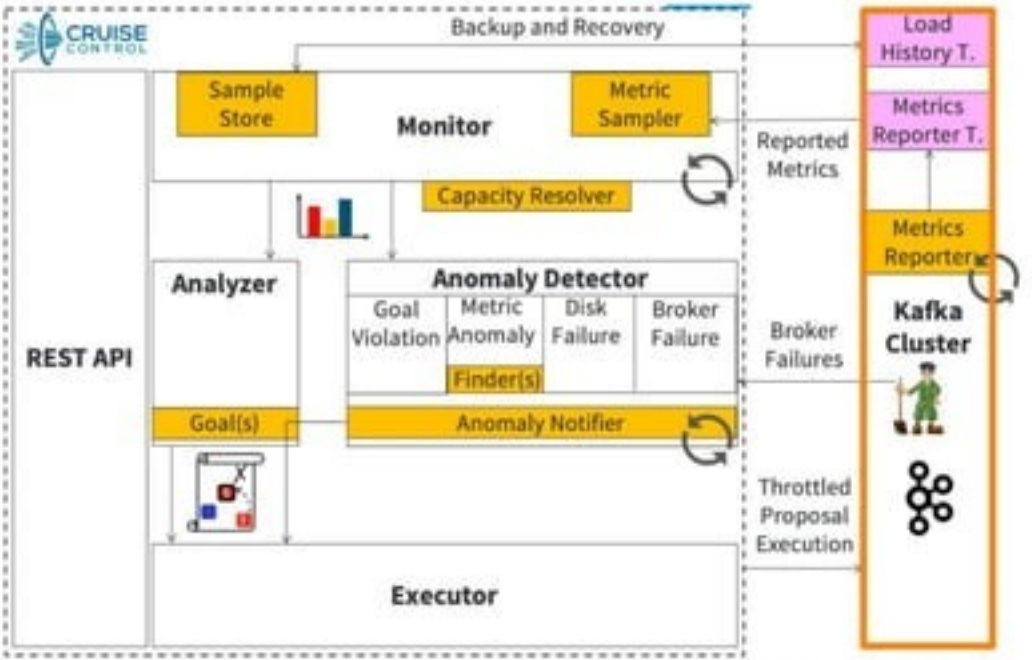
Estimation—partitions level utilization is estimated with a linear model

How does it work?

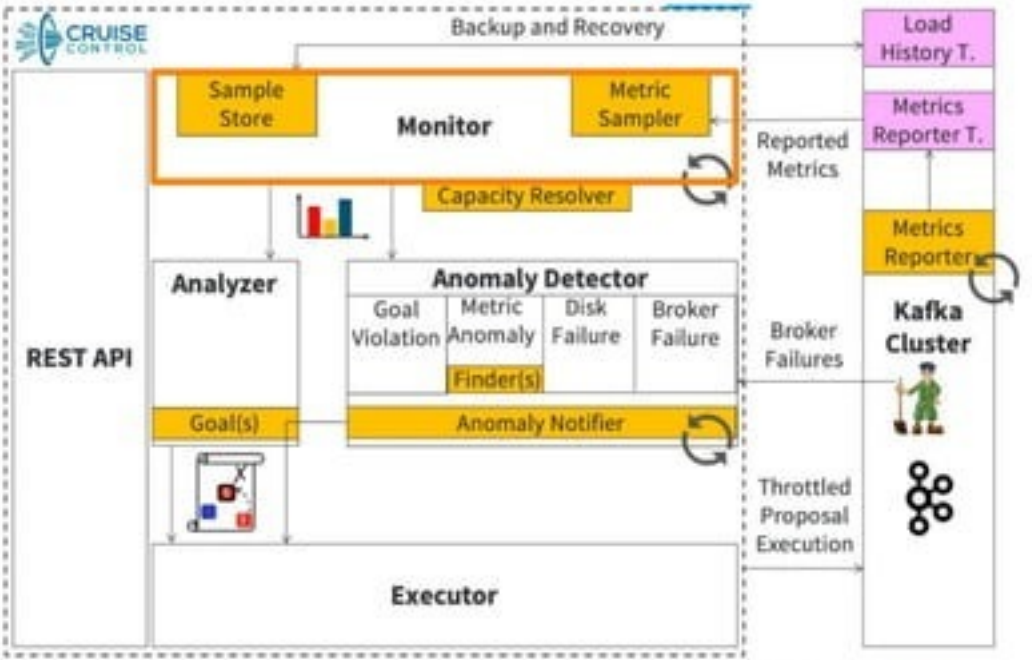
ARCHITECTURE



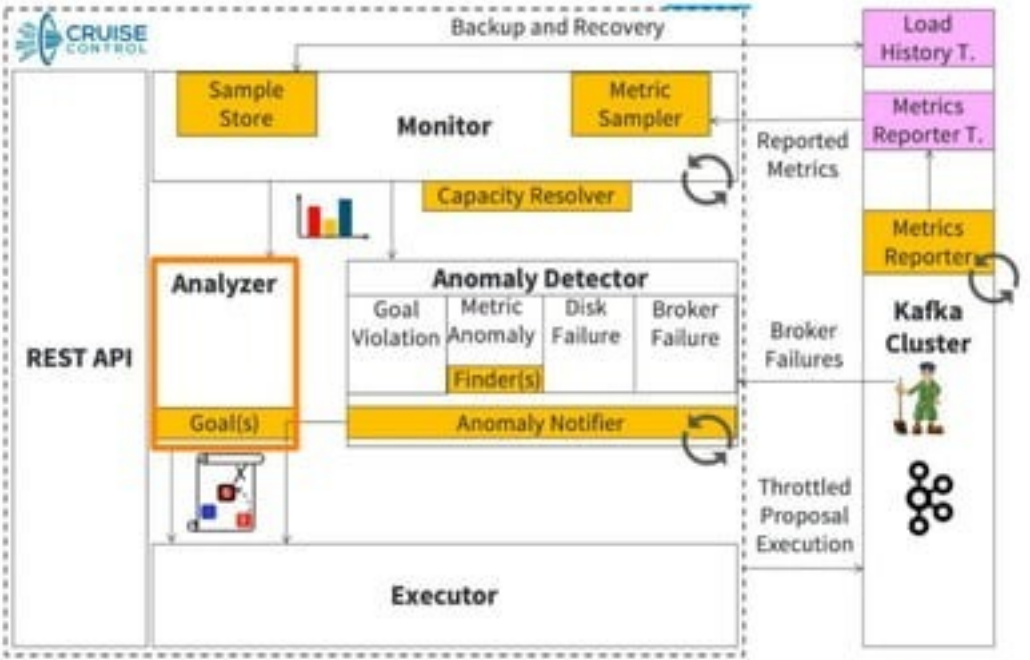
ARCHITECTURE



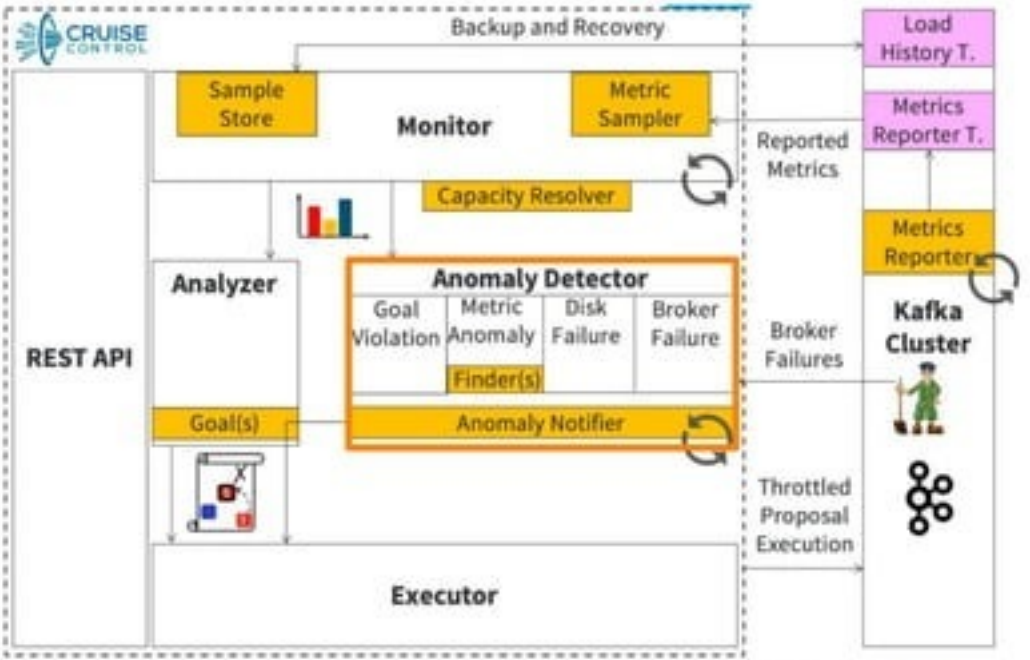
ARCHITECTURE



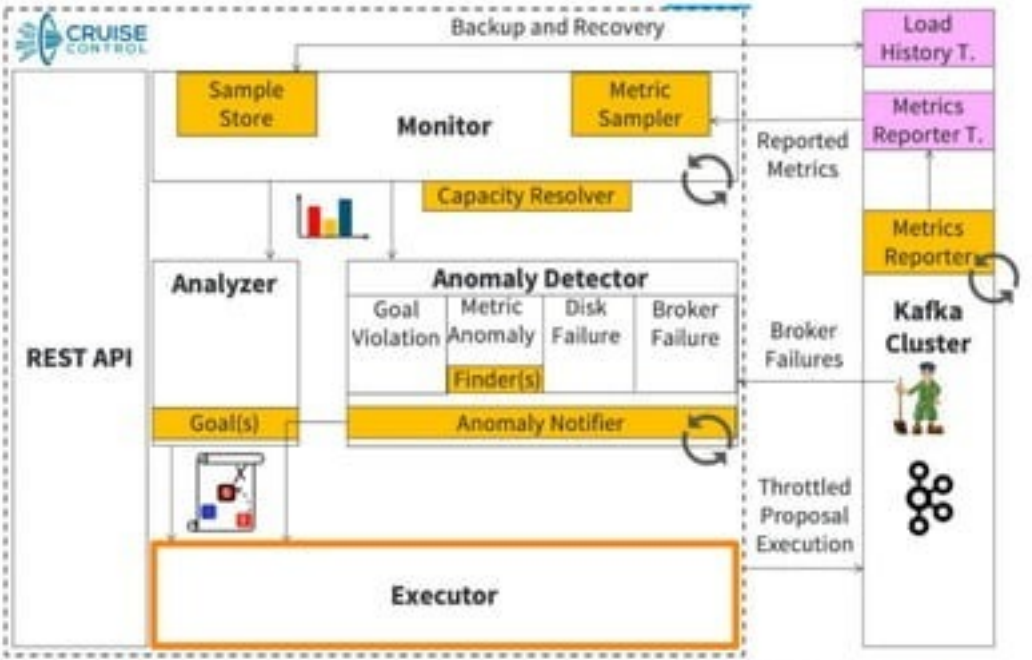
ARCHITECTURE



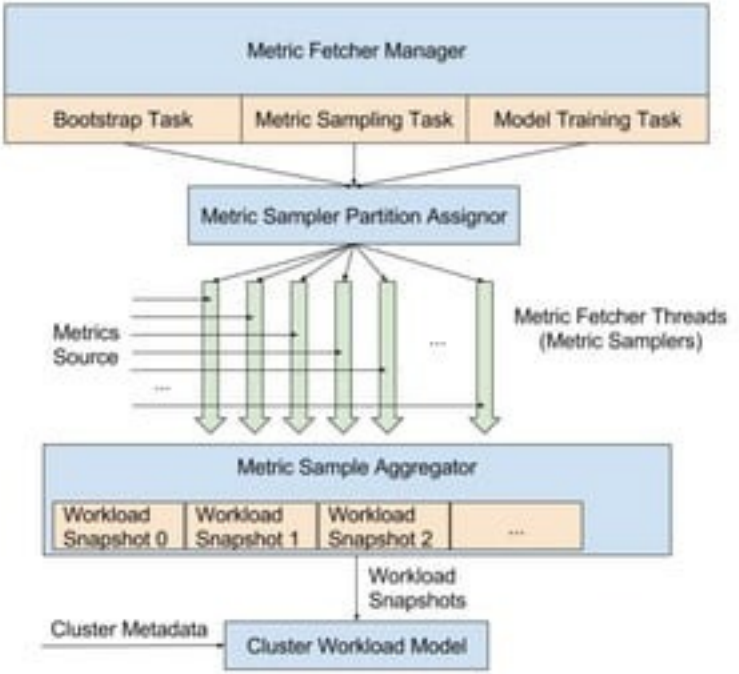
ARCHITECTURE



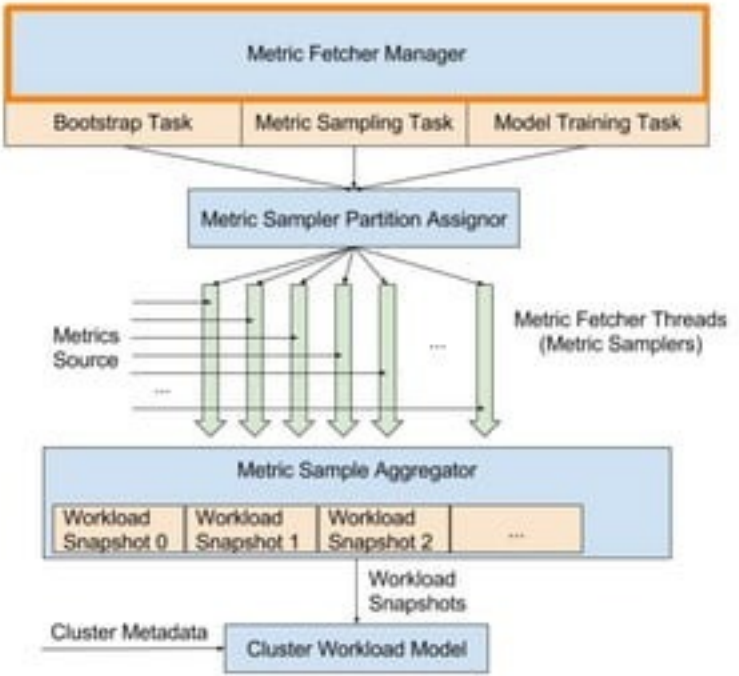
ARCHITECTURE



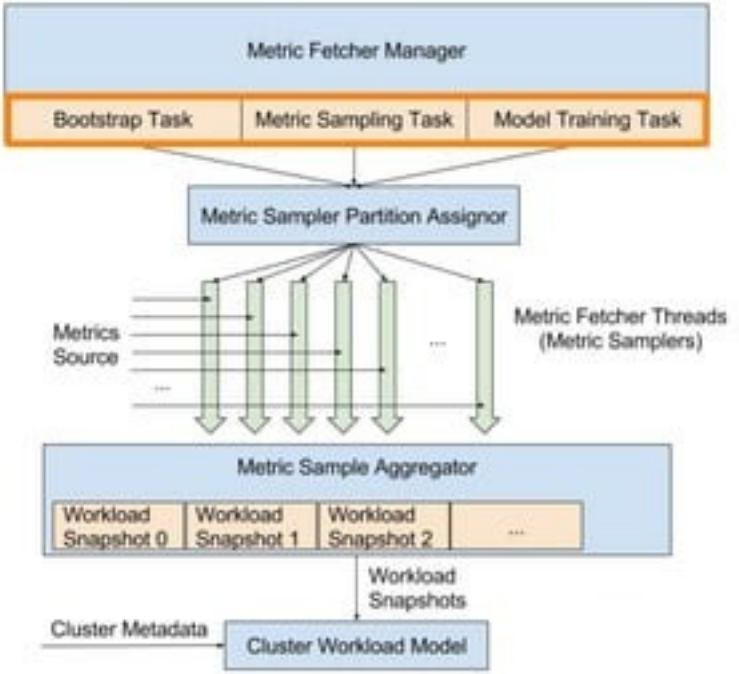
THE LOAD MONITOR



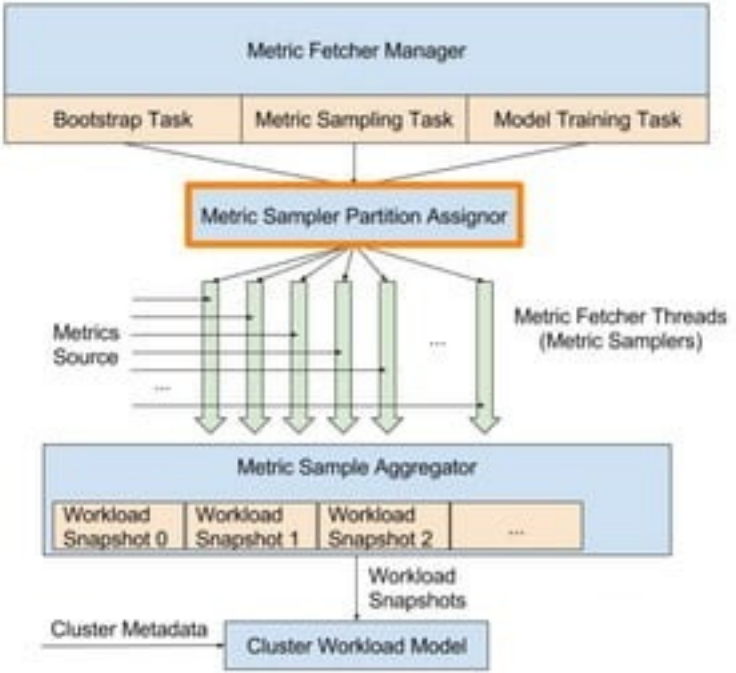
THE LOAD MONITOR



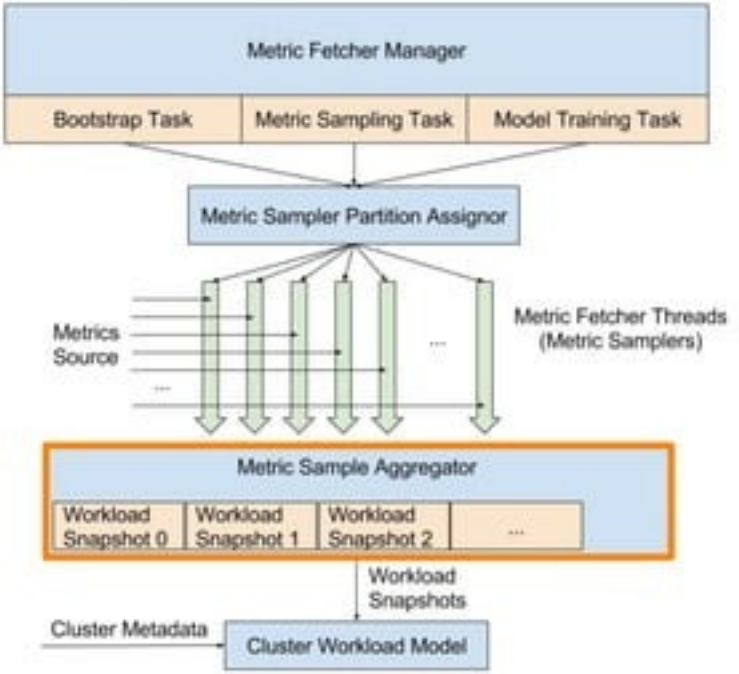
THE LOAD MONITOR



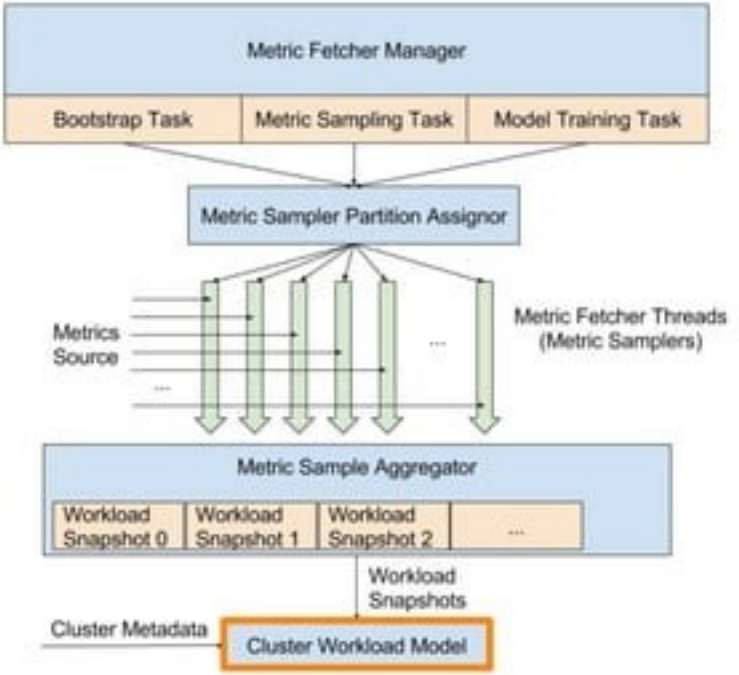
THE LOAD MONITOR



THE LOAD MONITOR



THE LOAD MONITOR



THE ANALYZER

The brain

THE ANALYZER

The brain

Goals

Define the characteristics of an optimal aspect

THE ANALYZER

The brain

Goals

Define the characteristics of an optimal aspect

Heuristic model

An iterative model defines how it reaches the optimal load

THE ANALYZER

The brain

Goals

Define the characteristics of an optimal aspect

Heuristic model

An iterative model defines how it reaches the optimal load

```
For each goal g in the goal list ordered by priority {
 For each broker b {
   while b does not meet g's requirement {
     For each replica r on b sorted by the resource utilization density {
       Move r (or the leadership of r) to another eligible broker b' so b'
still satisfies g and all the satisfied goals
       Finish the optimization for b once g is satisfied.
     }
     Fail the optimization if g is a hard goal and is not satisfied for b
   }
 }
 Add g to the satisfied goals
 }
```

THE ANOMALY DETECTOR

THE ANOMALY DETECTOR

Broker failure

A non-empty broker crashes. Results in under-replication.

THE ANOMALY DETECTOR

Broker failure

A non-empty broker crashes. Results in under-replication.

Disk failure

A non-empty disk dies. Some replicas might get lost.

THE ANOMALY DETECTOR

Broker failure

A non-empty broker crashes. Results in under-replication.

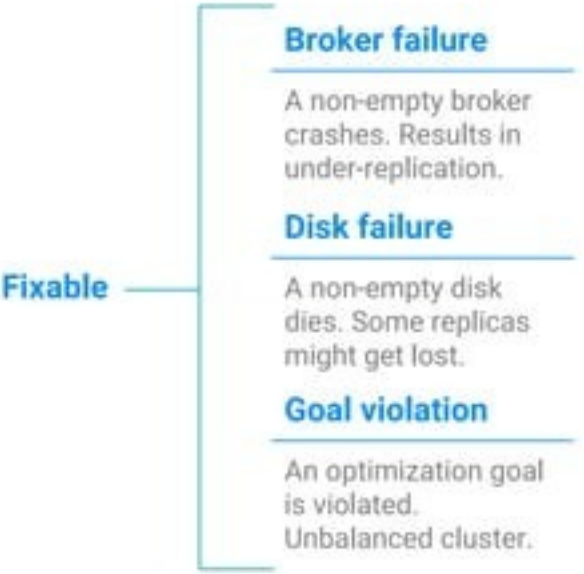
Disk failure

A non-empty disk dies. Some replicas might get lost.

Goal violation

An optimization goal is violated. Unbalanced cluster.

THE ANOMALY DETECTOR



THE ANOMALY DETECTOR

Fixable

Broker failure

A non-empty broker crashes. Results in under-replication.

Disk failure

A non-empty disk dies. Some replicas might get lost.

Goal violation

An optimization goal is violated. Unbalanced cluster.

Metric anomaly

One of the collected metric observes an unexpected value.

THE ANOMALY DETECTOR

THE ANOMALY DETECTOR

EXECUTOR

EXECUTOR

Long running

Depending on the size of the executed task.

EXECUTOR

Long running

Depending on the size of the executed task.

Interruptible

At any point it can be stopped safely. No "undo" functionality.

EXECUTOR

Long running

Depending on the size of the executed task.

Interruptible

At any point it can be stopped safely. No "undo" functionality.

Reassignment

Optimization is done by reassigning partitions between disks, brokers and adjusting leadership

EXECUTOR

Long running

Depending on the size of the executed task.

Interruptible

At any point it can be stopped safely. No "undo" functionality.

Reassignment

Optimization is done by reassigning partitions between disks, brokers and adjusting leadership

Concurrent

Reassignment is concurrent, limits are set either manually or by the concurrency adjuster

So how do we use it?

What else can it do?

SLOW BROKER FINDER

SLOW BROKER FINDER

DETECTION
With metrics

Log flush—broker log flush 99.9th percentile is used, both absolute and relative to incoming traffic

Scoring—brokers have score, every anomaly increases it and every normal behavior decreases it

SLOW BROKER FINDER

DETECTION
With metrics

Log flush—broker log flush 99.9th percentile is used, both absolute and relative to incoming traffic

Scoring—brokers have score, every anomaly increases it and every normal behavior decreases it

REMEDIATION
Demoting and rebalance

Demoting—brokers will be demoted as a first attempt

Removal—slow brokers can be removed from the cluster if the problem persists after demotion

PROVISIONER

PROVISIONER

RIGHTSIZING
With API

Underprovisioned—the cluster lacks certain kind of resources

Overprovisioned—by removing resources and rebalancing it can achieve some cost saving

PROVISIONER

RIGHTSIZING
With API

Underprovisioned—the cluster lacks certain kind of resources

Overprovisioned—by removing resources and rebalancing it can achieve some cost saving

RECOMMENDATION
With goals

Resource—a goal can request a resource, like broker, disk to be added or removed

Constraints—every goal can give constraints, e.g. racks for which brokers should not be added

PROVISIONER

RIGHTSIZING
With API

Underprovisioned—the cluster lacks certain kind of resources

Overprovisioned—by removing resources and rebalancing it can achieve some cost saving

RECOMMENDATION
With goals

Resource—a goal can request a resource, like broker, disk to be added or removed

Constraints—every goal can give constraints, e.g. racks for which brokers should not be added

IMPLEMENTATION
Provider dependent

API—there is an API that can be used as a basis for your implementation

Providers—for all providers like AWS or Azure you need to add glue code to acquire or release resources

CONCURRENCY ADJUSTER

Inter-broker replica reassignment limits aren't easy

CONCURRENCY ADJUSTER

Inter-broker replica reassignment limits aren't easy

MANUAL LIMITS

Challenging

Destructive—if the manually set limit is too high, then it can be fast but also overwhelm the cluster

Slow—If the limit is too small then rebalances can last for a long time

CONCURRENCY ADJUSTER

Inter-broker replica reassignment limits aren't easy

MANUAL LIMITS
Challenging

Destructive—if the manually set limit is too high, then it can be fast but also overwhelm the cluster

Slow—If the limit is too small then rebalances can last for a long time

FEEDBACK LOOP
Adaptive limit

Metrics—metrics are collected to assess whether rebalance concurrency needs adjustments

Limits—if metrics are over a threshold then concurrency will adjust with AIMD algorithm

CONCURRENCY ADJUSTER

Inter-broker replica reassignment limits aren't easy

MANUAL LIMITS
Challenging

Destructive—if the manually set limit is too high, then it can be fast but also overwhelm the cluster

Slow—If the limit is too small then rebalances can last for a long time

FEEDBACK LOOP
Adaptive limit

Metrics—metrics are collected to assess whether rebalance concurrency needs adjustments

Limits—if metrics are over a threshold then concurrency will adjust with AIMD algorithm

RESULTS
AIMD

Automated—no need for constant oversight for setting concurrency in a fluctuating traffic environment

Fast—rebalances will complete faster due to the optimal limit

Stable—clients won't experience any side effects caused by rebalances

Security matters

SECURITY IN CRUISE CONTROL

SECURITY IN CRUISE CONTROL

Basic Auth

The simple basic authentication that sends username and password

SECURITY IN CRUISE CONTROL

Basic Auth

The simple basic authentication that sends username and password

SPNEGO

Kerberos over HTTP. Tokens are negotiated via the SPNEGO protocol.

SECURITY IN CRUISE CONTROL

Basic Auth

The simple basic authentication that sends username and password

SPNEGO

Kerberos over HTTP. Tokens are negotiated via the SPNEGO protocol.

Trusted Proxy

If a process acts on behalf of a user, the doAs header will forward the username.

SECURITY IN CRUISE CONTROL

Basic Auth

The simple basic authentication that sends username and password

TLS/HTTPS

Communication is secured via HTTPS to the server and TLS to Zookeeper or Kafka

SPNEGO

Kerberos over HTTP. Tokens are negotiated via the SPNEGO protocol.

Trusted Proxy

If a process acts on behalf of a user, the doAs header will forward the username.

SECURITY IN CRUISE CONTROL

Basic Auth

The simple basic authentication that sends username and password

TLS/HTTPS

Communication is secured via HTTPS to the server and TLS to Zookeeper or Kafka

Authorization

Simple model with viewer, user and admin roles.

SPNEGO

Kerberos over HTTP. Tokens are negotiated via the SPNEGO protocol.

Trusted Proxy

If a process acts on behalf of a user, the doAs query param will forward the username.

THANK YOU

CLOUDERA