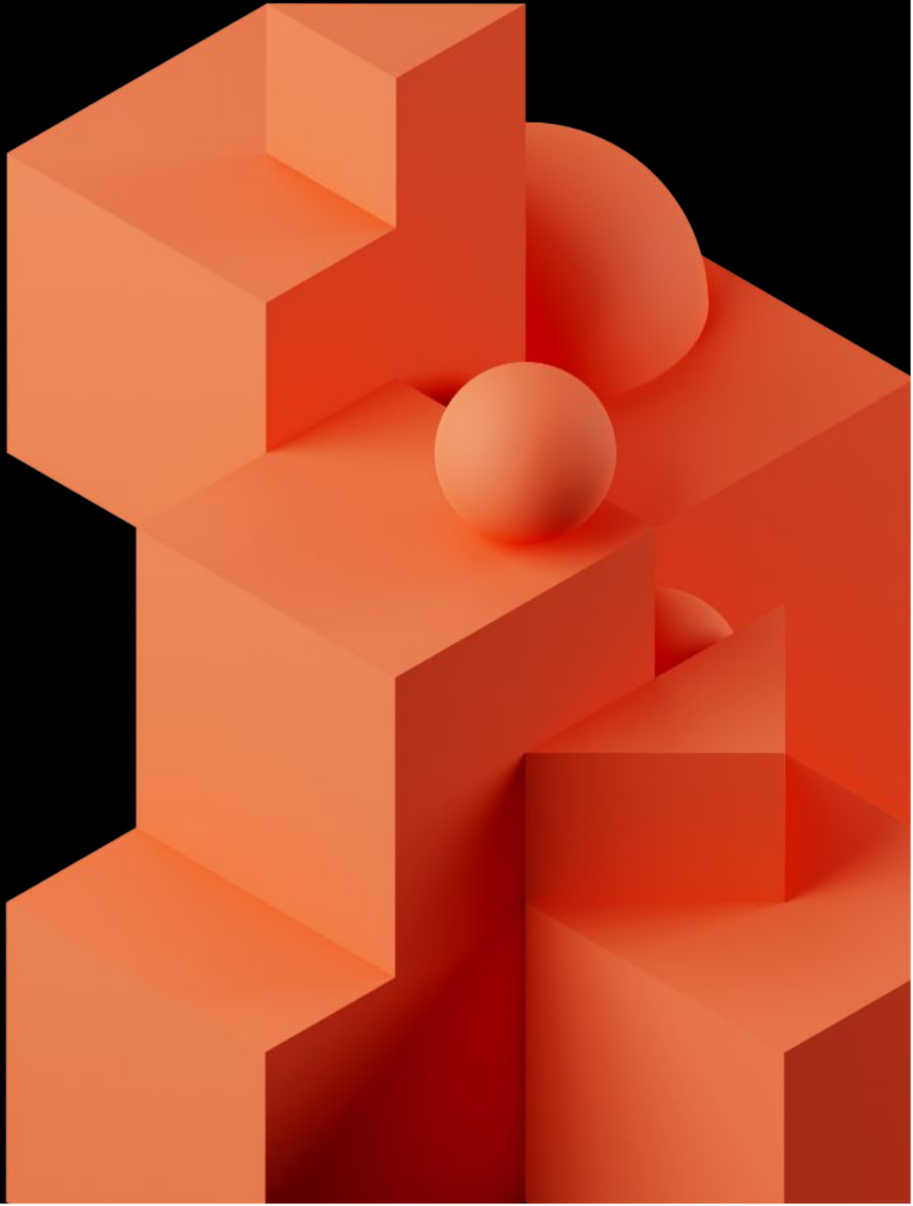
DATA+AI SUMMIT
BY databricks

Ray on Spark

Ben Wilson, Databricks

Jiajun Yao, Anyscale



Databricks
2023

Who we are …

Ben Wilson

- Works with ML open source software at Databricks
- MLflow maintainer

Jiajun Yao

- Software engineer at Anyscale
- Ray committer

Agenda

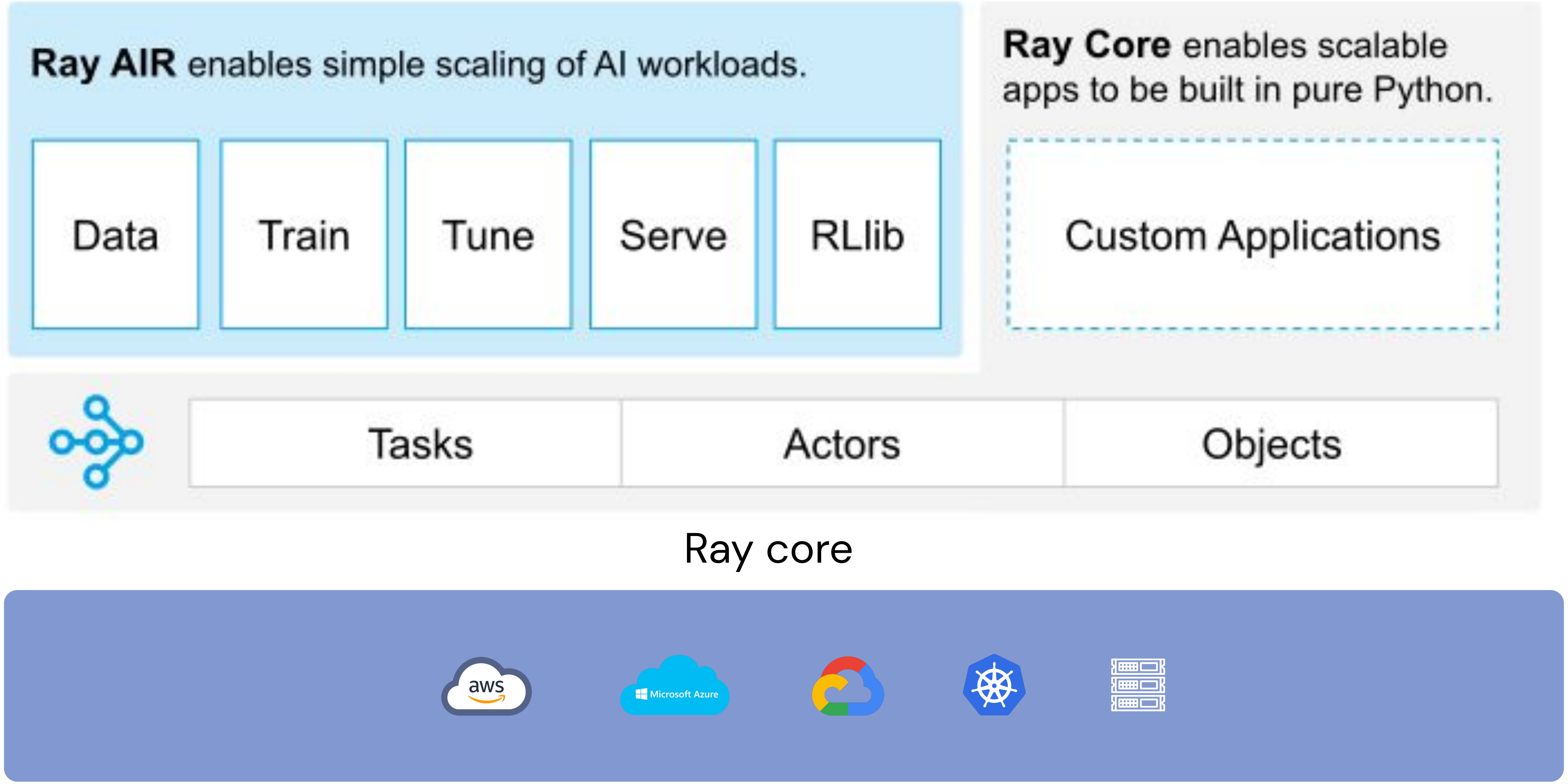
- What is Ray
- What is Ray-on-Spark
- Why Ray-on-Spark
- How to use Ray-on-Spark
- Demos
- How does Ray-on-Spark work
- Future work

What is Ray

What is Ray

- An open-source unified **distributed** framework that makes it easy to scale **AI** and **Python** applications.
- An ecosystem of Python libraries (for scaling ML and more).
- Makes distributed computing easy and accessible to everyone.
- Runs on laptop, public cloud, K8s, on-premise.

What is Ray



Ray AIR enables simple scaling of AI workloads.

Data

Train

Tune

Serve

RLlib

Ray Core enables scalable apps to be built in pure Python.

Custom Applications

Tasks

Actors

Objects

Ray core

Library + app ecosystem

General-purpose framework for distributed computing

Run anywhere

What is Ray

Function

```
def read_array(file):
    # read ndarray “a”
    # from “file”
    return a

def add(a, b):
    return np.add(a, b)

a = read_array(file1)
b = read_array(file2)
sum = add(a, b)
```

Class

```
class Counter(object):
    def __init__(self):
        self.value = 0
    def inc(self):
        self.value += 1
        return self.value

c = Counter()
c.inc()
c.inc()
```

What is Ray

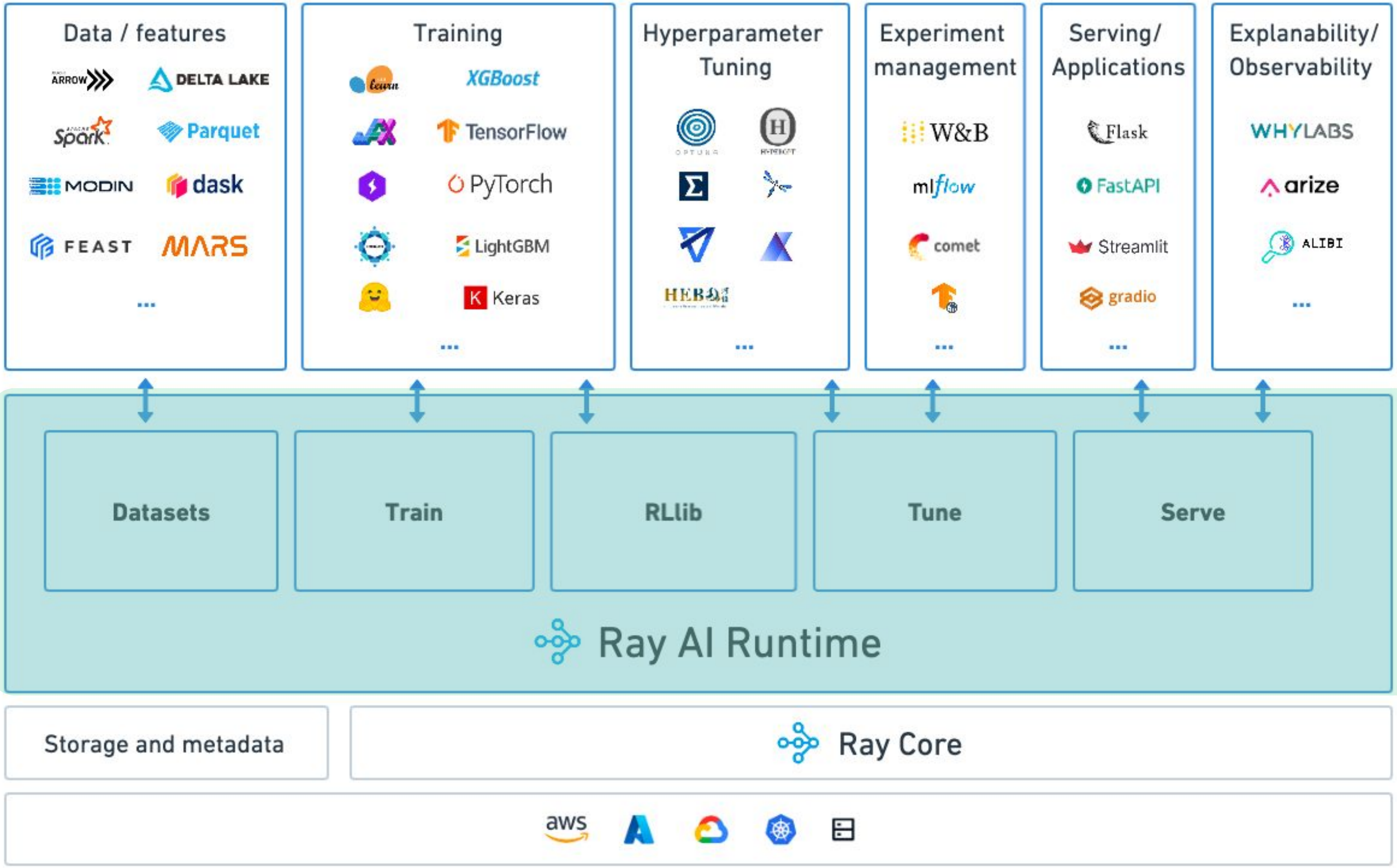
Function -> **Task**

```
@ray.remote
def read_array(file):
    # read ndarray “a”
    # from “file”
    return a

@ray.remote
def add(a, b):
    return np.add(a, b)

a_ref =
read_array.remote(file1)
b_ref =
read_array.remote(file2)
sum_ref = add.remote(a, b)
sum = ray.get(sum_ref)
```

Class -> **Actor**

```
@ray.remote
class Counter(object):
    def __init__(self):
        self.value = 0
    def inc(self):
        self.value += 1
        return self.value

c = Counter.remote()
c.inc.remote()
c.inc.remote()
```

What is Ray



Data / features

Training

Hyperparameter Tuning

Experiment management

Serving/ Applications

Explanability/ Observability

Datasets

Train

RLlib

Tune

Serve

Ray AI Runtime

Storage and metadata

Ray Core

High-level libraries that make scaling easy for both data scientists and ML engineers.

What is Ray

amazon | McKinsey & Company | OAK RIDGE National Laboratory | ERICSSON | WILD LIFE | cruise

Morgan Stanley | rbi restaurant brands international | shopify | J.P.Morgan | KOCH | RIOT GAMES

intel | Microsoft | 蚂蚁金服 ANT FINANCIAL | Uber | NetEase Games | RICARDO

OpenAI | ByteDance | lyft | instacart | verizon | Spotify

25,000+
GitHub
stars

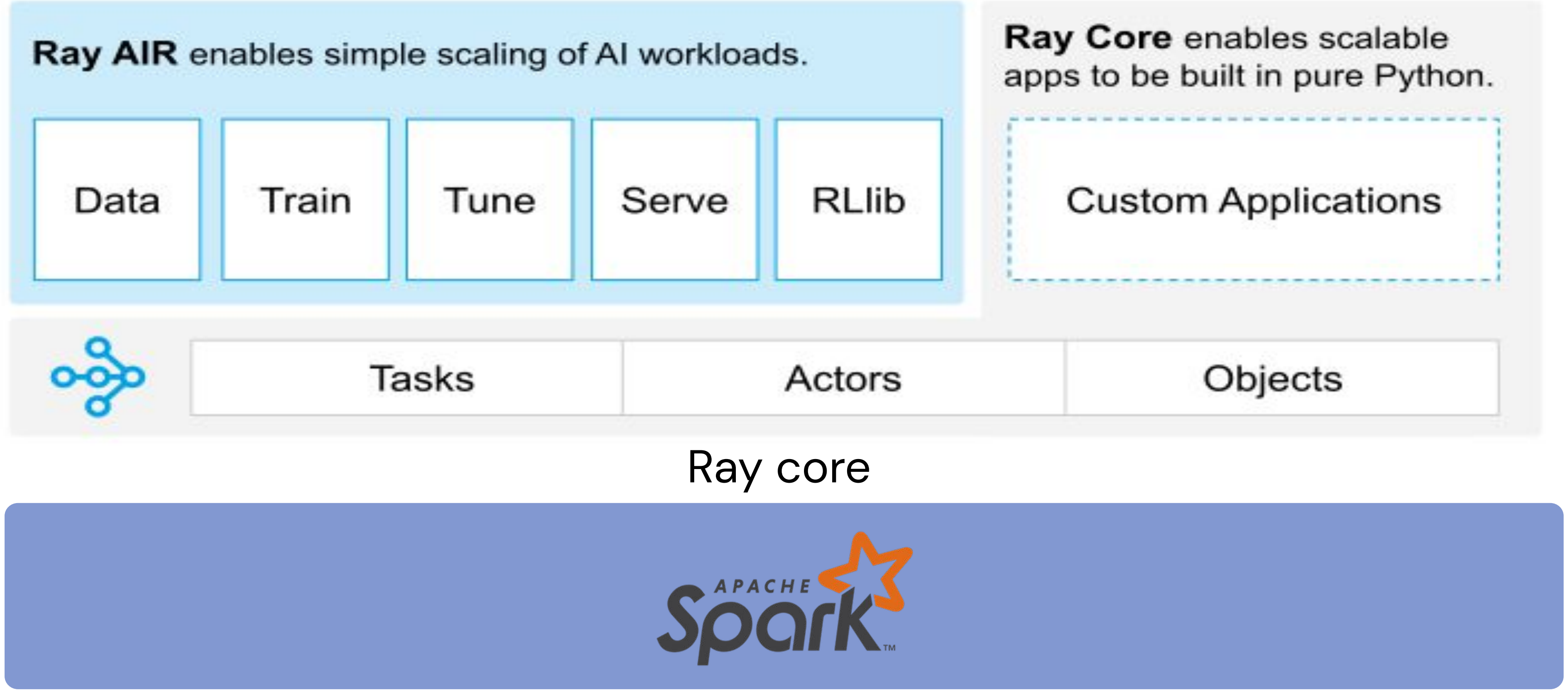
820+
Community
Contributors

5,000+
Repositories
Depend on Ray

1,000+
Organizations
Using Ray

What is Ray-on-Spark

- A library to deploy Ray clusters on Spark and run Ray applications.


Ray AIR enables simple scaling of AI workloads.
Data
Train
Tune
Serve
RLlib
Ray Core enables scalable apps to be built in pure Python.
Custom Applications
Tasks
Actors
Objects
Ray core
APACHE Spark

Why Ray-on-Spark

- User asks
  - Spark users want to use both Spark MLlib and Ray ML libraries (e.g. RLLib).
- Cost
  - Share the same physical cluster between Ray and Spark applications.
- Easy to manage
  - No need to manage two separate physical clusters.

How to use Ray-on-Spark

- Install Ray

```
% pip install ray[all]>=2.3.0
```

- Start a Ray cluster

```
import ray

ray.util.spark.setup_ray_cluster(num_worker_nodes=5)
```

- Run Ray applications

```
ray.init() # Connect to the previously created Ray cluster

... # Your Ray application code

print(ray.nodes())
```

- Stop the Ray cluster

```
ray.util.spark.shutdown_ray_cluster()
```

Getting started

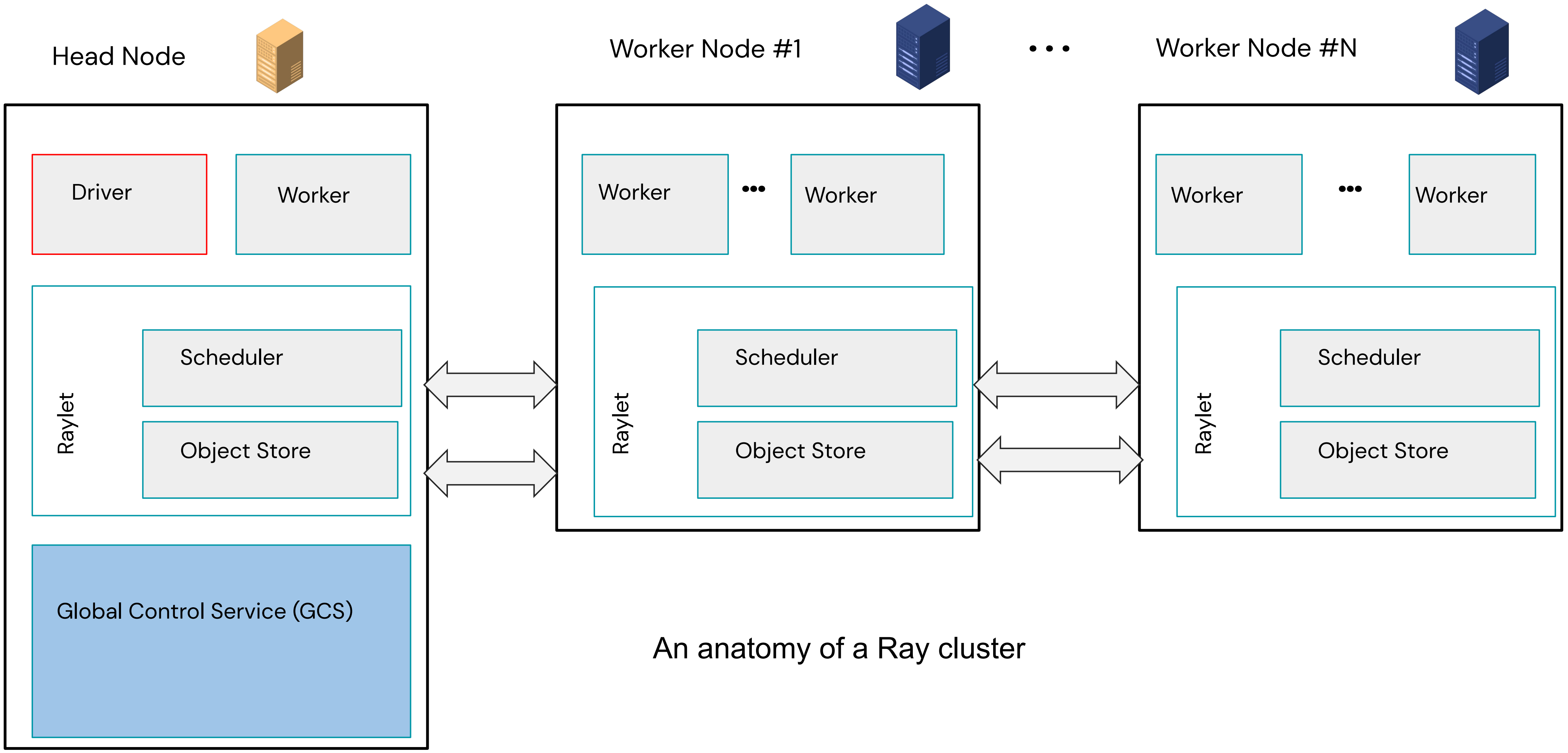
The Ray Dashboard

Validation

Parallel processing

Distributed Hyperparameter tuning

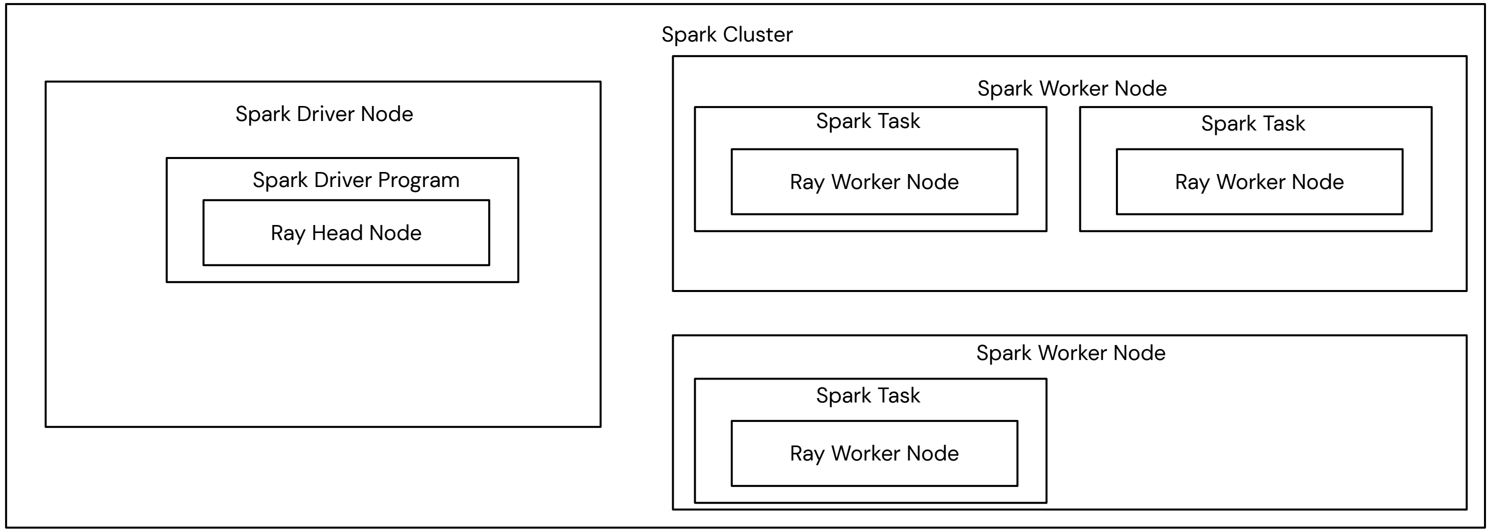
How does Ray-on-Spark work



An anatomy of a Ray cluster

How does Ray-on-Spark work

- Ray head node runs on the Spark driver node.
- Ray worker nodes are started by a long-running Spark job.
- Each long-running Spark task starts a Ray worker node and allocates to the node the full set of resources available to it.



Future work

- Autoscaling support
- Delta data source support in Ray Data

Conclusion

- Ray-on-Spark is in Public Preview for Ray >= 2.3 & (Spark >= 3.3 | Databricks Runtime >= 12.0)

Try out Ray-on-Spark on Spark standalone clusters

Try out Ray-on-Spark on Databricks clusters

Learn more about Ray