Preparation

Please go to **https://github.com/ray-project/tutorial**

And click on the “Binder” link and wait a couple minutes.

Binder not working? Backup:

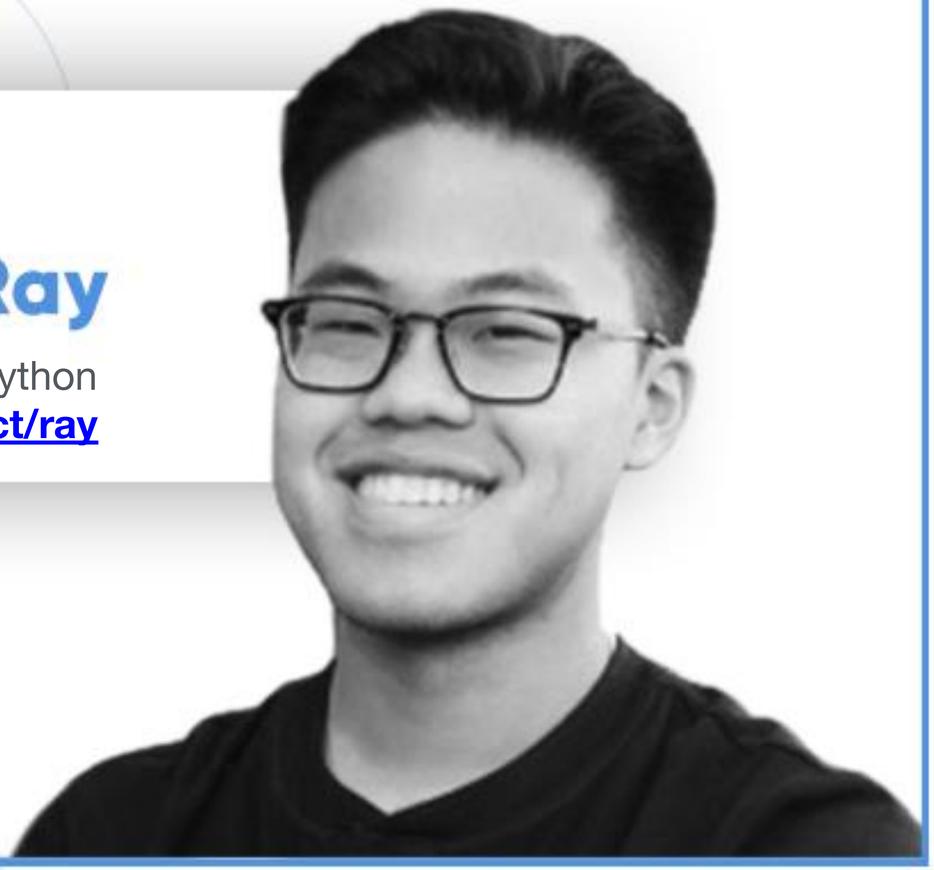
- run locally with "pip install ray[rllib] tensorflow jupyterlab"
- sign in to Google CoLab and import the tutorial notebooks

#ODSC

BOSTON
APR 30 - MAY 3

Scaling AI Applications with Ray

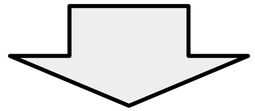
A general-purpose system for parallel and distributed Python
https://github.com/ray-project/ray



Richard Liaw
AI Researcher,
RISELab at UC Berkeley

The Big Picture

Distributed System	Distributed System	Distributed System	Distributed System	Distributed System	Distributed System
Distributed Training	Model Serving	Streaming	Distributed RL	Data Processing	Hyperparameter Search
Horovod, Distributed TF, Parameter Server	Clipper, TensorFlow Serving	Flink, many others	Baselines, RLlab, ELF, Coach, TensorForce, ChainerRL	MapReduce, Hadoop, Spark	Vizier, many internal systems at companies



Libraries

Distributed Training	Model Serving	Streaming	Distributed RL	Data Processing	Hyperparameter Search

Distributed System (Ray)

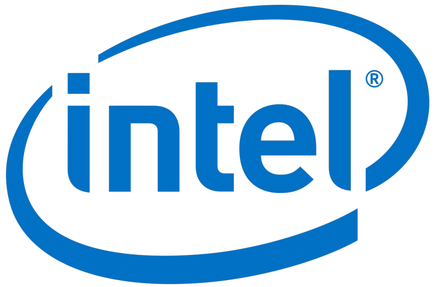
A Growing Number of Use Cases

蚂蚁金服
ANT FINANCIAL

Berkeley
UNIVERSITY OF CALIFORNIA

BONSAI AI

ERICSSON



Alibaba.com™

facebook

J.P.Morgan

Morgan Stanley

PRIMER

Microsoft

Agenda

Part 1 (2:00-2:30): Ray as a unifying distributed system for machine learning
- Ray API, Architecture
- Exercises

Part 2 (2:30-3:00): Model Tuning/Training
- Overview of Tune
- Tune Exercise

Part 3 (3:00-3:30): Applied Reinforcement Learning
- Overview of RLlib
- RLlib Exercise

Ray API

Functions -> Tasks

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


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


Ray API

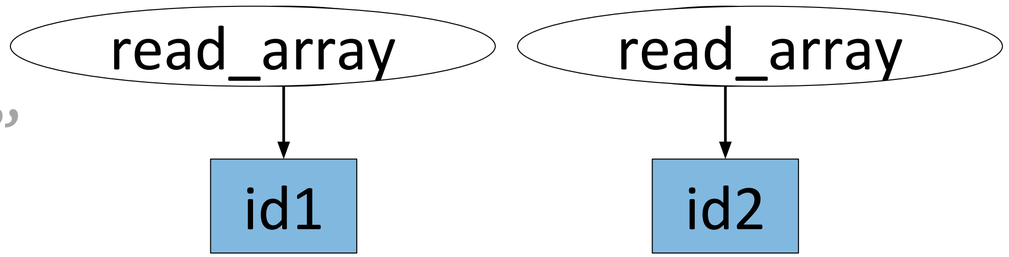
Functions -> Tasks

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

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


Ray API

Functions -> Tasks

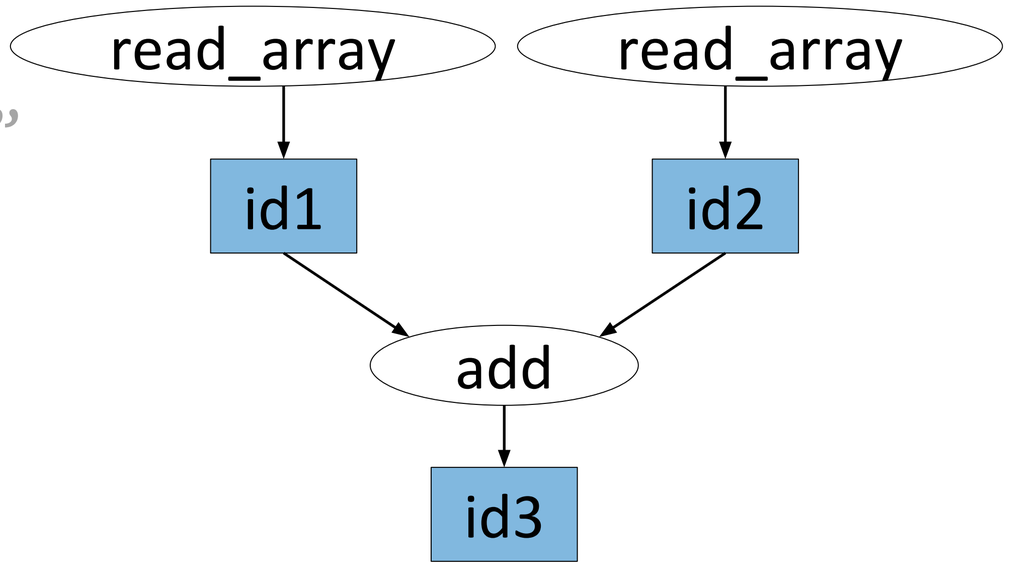
```
@ray.remote
def read_array(file):
    # read array “a” from “file”
    return a

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

id1 = read_array.remote([5, 5])
id2 = read_array.remote([5, 5])
```



Ray API

Functions -> Tasks

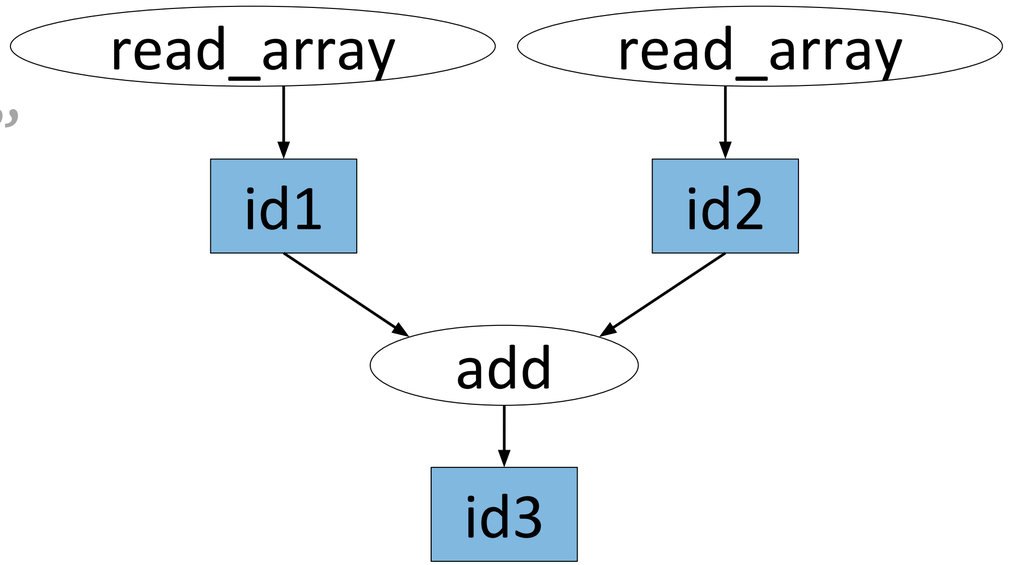
```
@ray.remote
def read_array(file):
    # read array “a” from “file”
    return a

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

id1 = read_array.remote([5, 5])
id2 = read_array.remote([5, 5])
id3 = add.remote(id1, id2)
```



Ray API

Functions -> Tasks

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

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

id1 = read_array.remote([5, 5])
id2 = read_array.remote([5, 5])
id3 = add.remote(id1, id2)
ray.get(id3)
```



Ray API

Functions -> Tasks

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

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

id1 = read_array.remote([5, 5])
id2 = read_array.remote([5, 5])
id3 = add.remote(id1, id2)
ray.get(id3)
```

Classes -> Actors

Ray API

Functions -> Tasks

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

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

id1 = read_array.remote([5, 5])
id2 = read_array.remote([5, 5])
id3 = add.remote(id1, id2)
ray.get(id3)
```

Classes -> Actors

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


Ray API

Functions -> Tasks

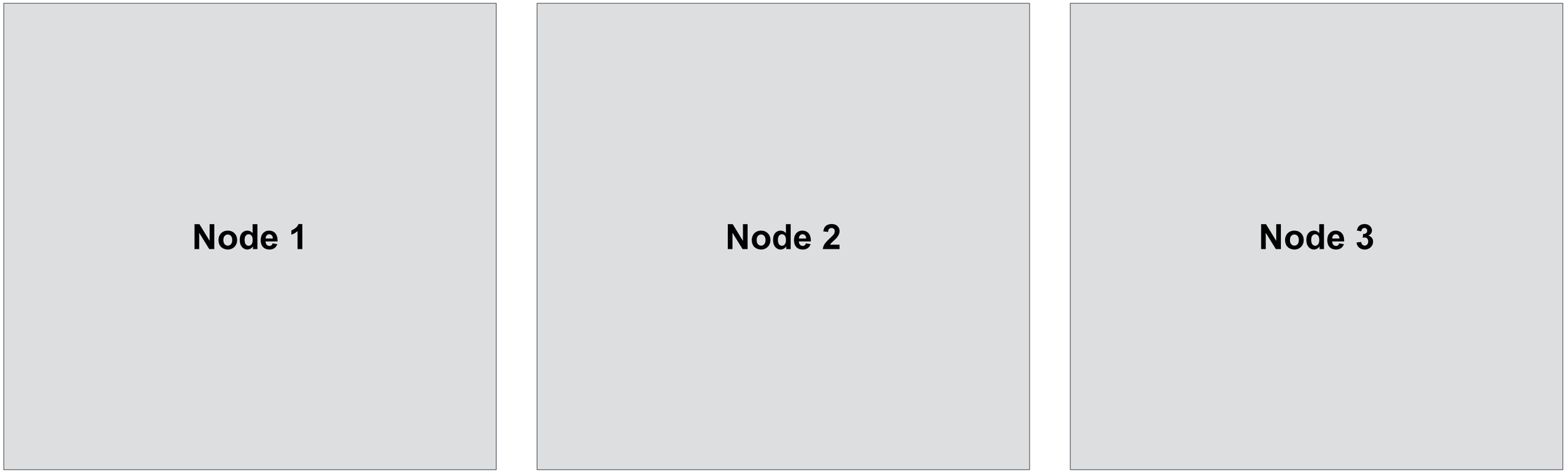
```
@ray.remote
def read_array(file):
    # read array “a” from “file”
    return a

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

id1 = read_array.remote([5, 5])
id2 = read_array.remote([5, 5])
id3 = add.remote(id1, id2)
ray.get(id3)
```

Classes -> Actors

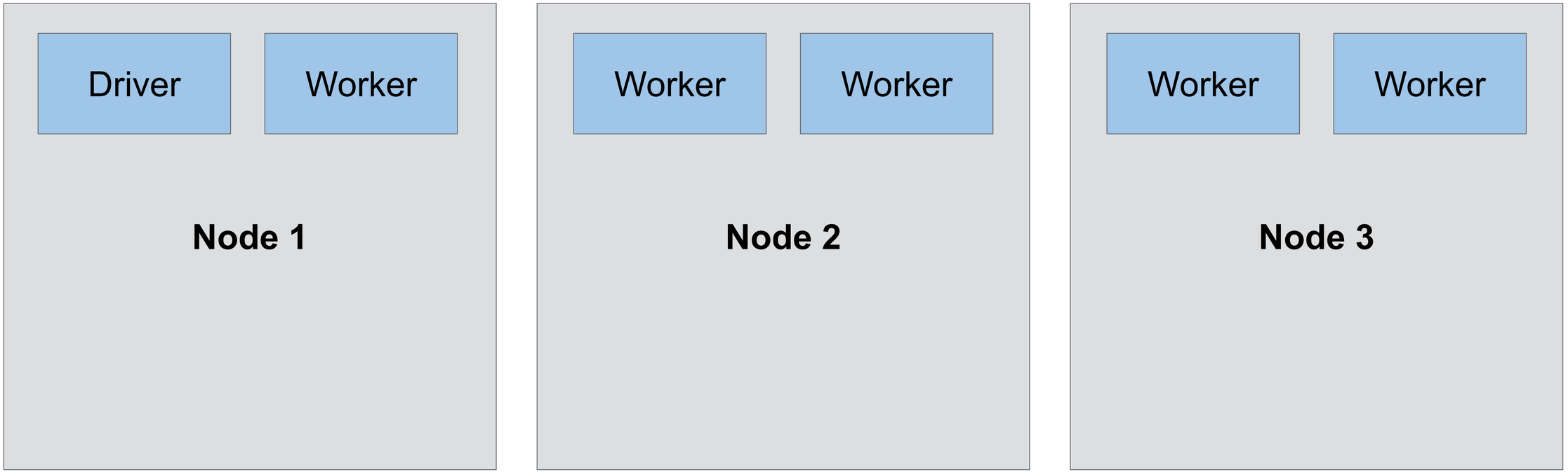
```
@ray.remote
class Counter(object):
    def __init__(self):
        self.value = 0
    def inc(self):
        self.value += 1
        return self.value

c = Counter.remote()
id4 = c.inc.remote()
id5 = c.inc.remote()
ray.get([id4, id5])
```

Quick Overview of **Ray Architecture**

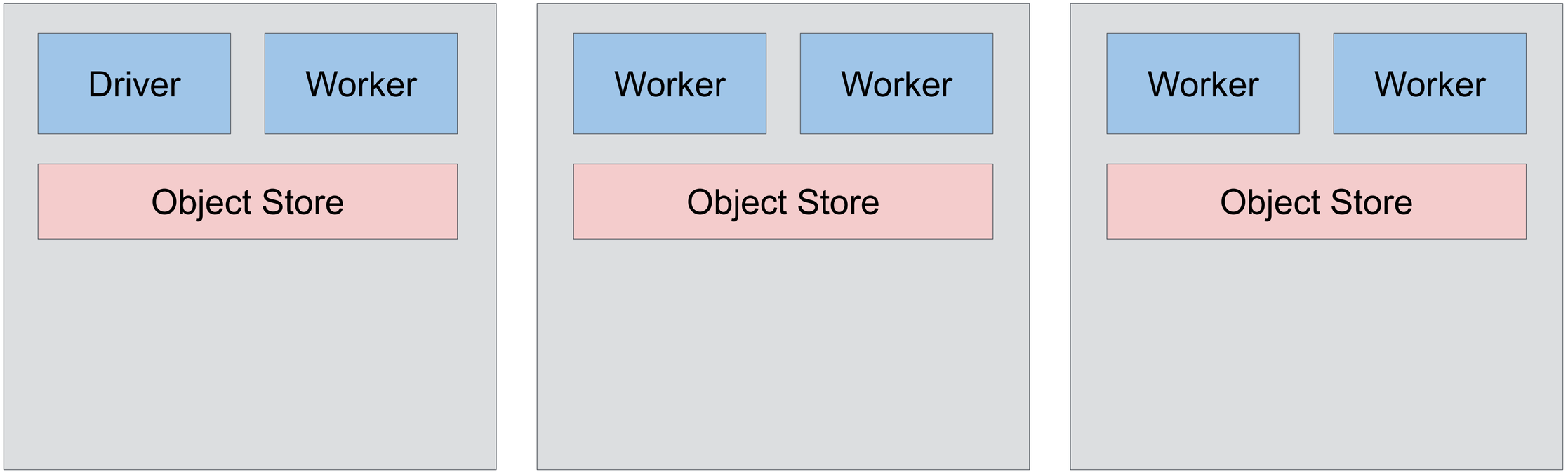
Ray Architecture



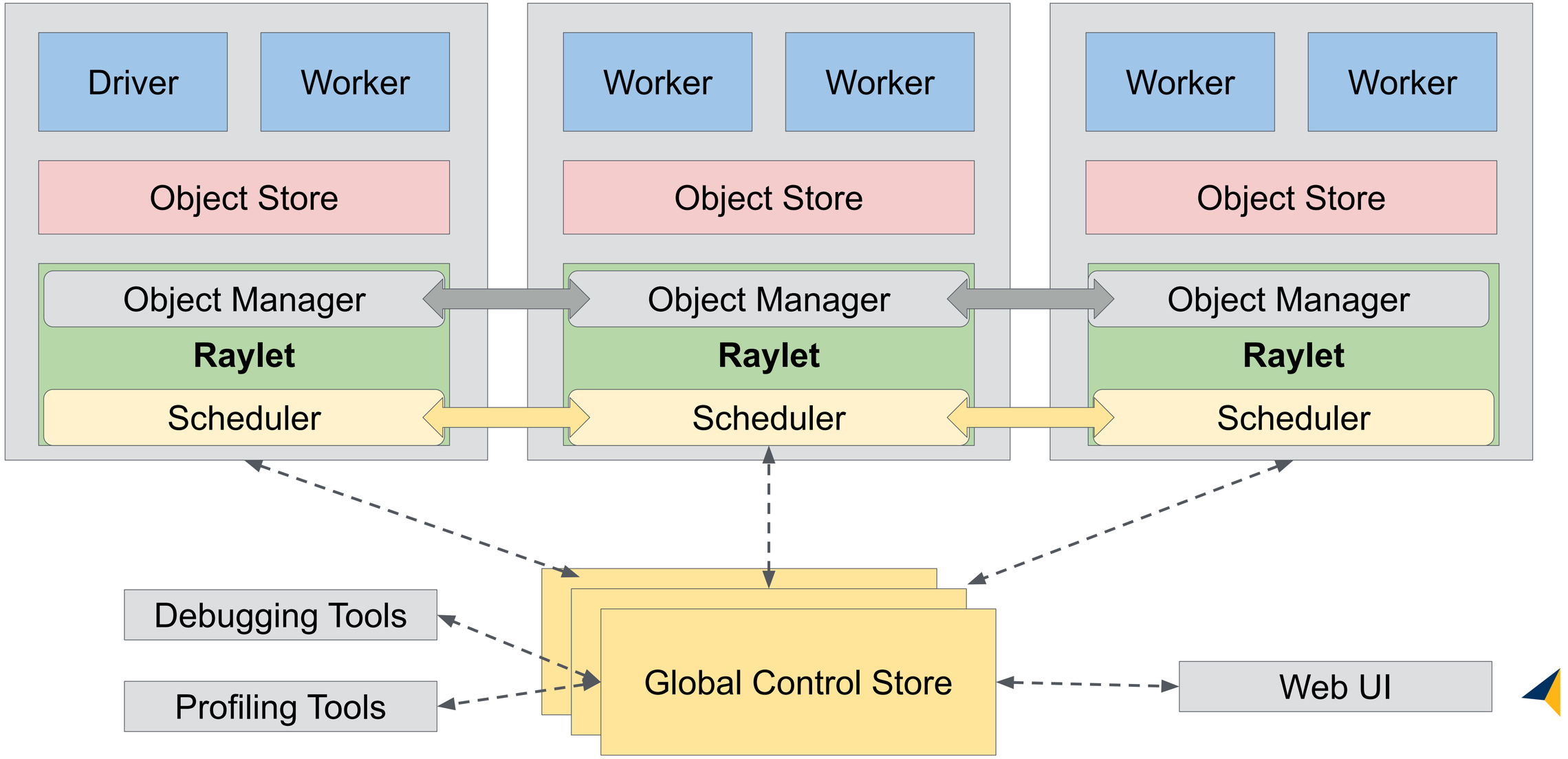
Ray Architecture



Ray Architecture



Ray Architecture



Exercises

Content available at github.com/ray-project/tutorial/.

Part 1: Ray as a unifying distributed system for machine learning

- exercises/ - 1, 2, (3 optional)

Click on the “Binder” link and wait a couple minutes.

Binder not working? Backup:

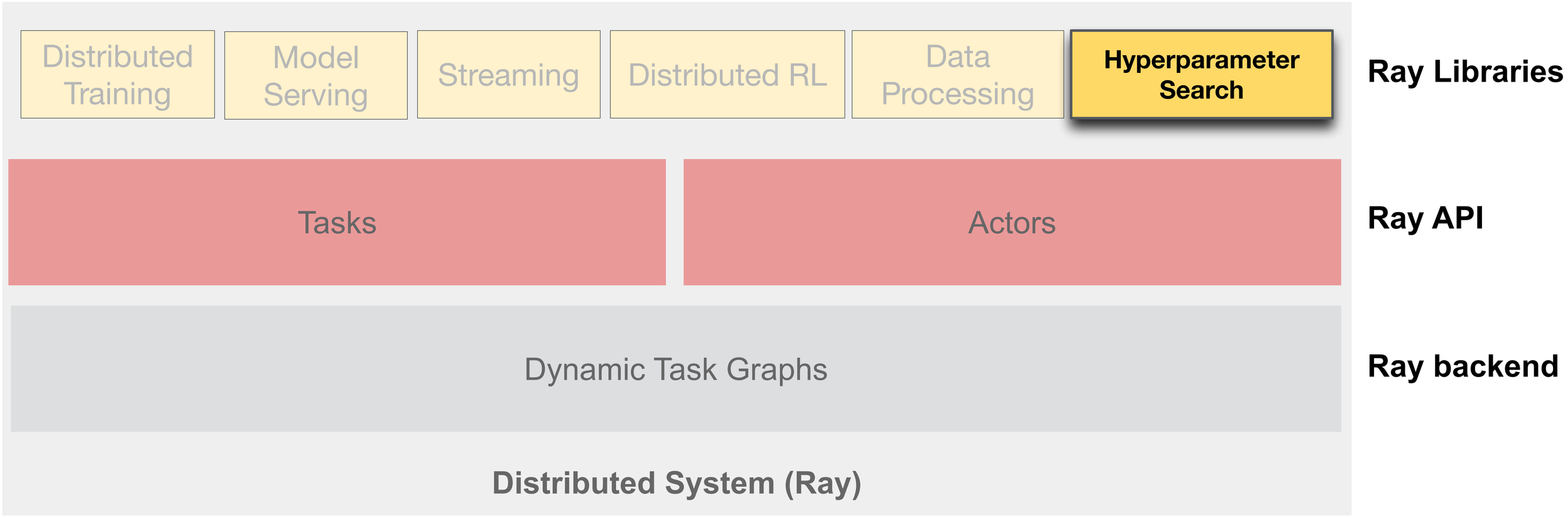
- run locally with "pip install ray[rllib] tensorflow jupyterlab"

Distributed Hyperparameter Search on Ray

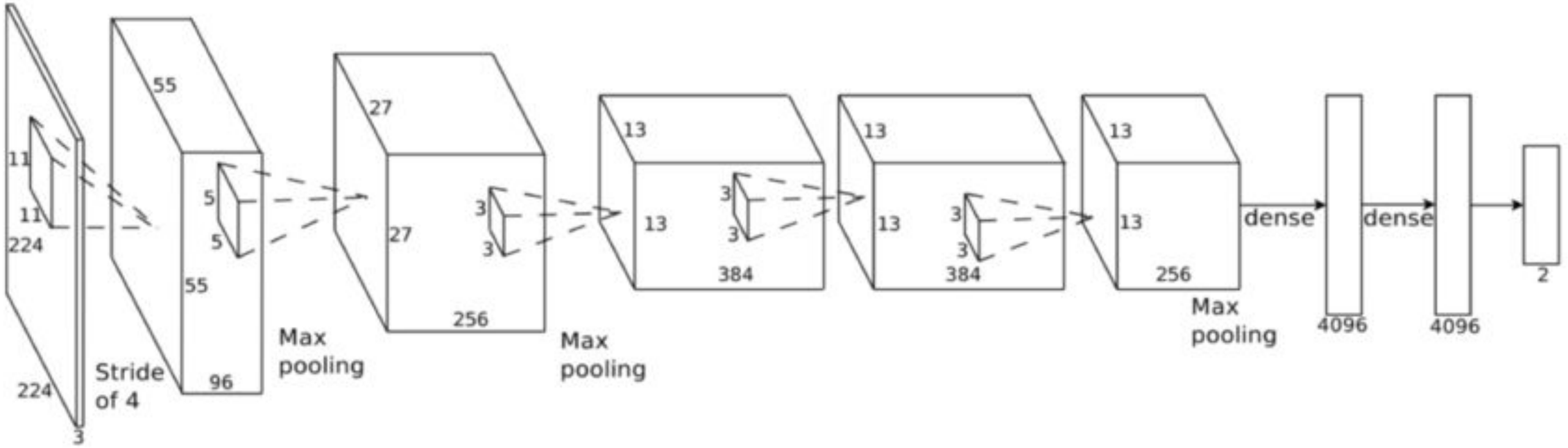
riselab
UC Berkeley

ray.readthedocs.io/en/latest/tune.html

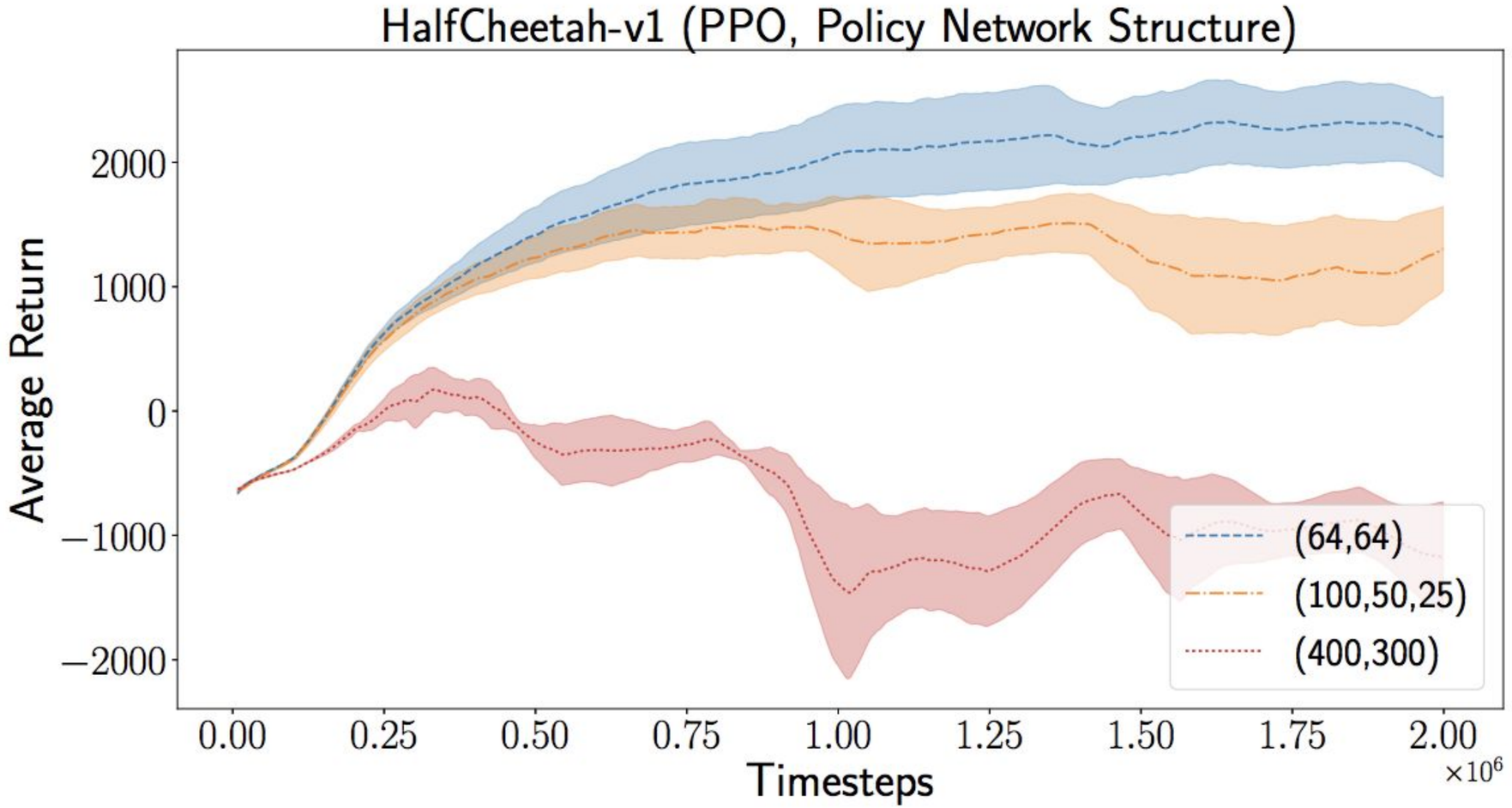
What is Tune?


Distributed Training
Model Serving
Streaming
Distributed RL
Data Processing
Hyperparameter Search
Ray Libraries
Tasks
Actors
Ray API
Dynamic Task Graphs
Ray backend
Distributed System (Ray)

Hyperparameters?



Are hyperparameters actually that important?

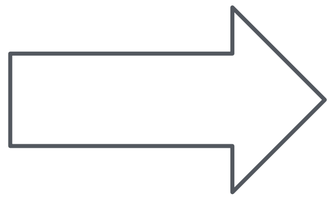


Why a framework for tuning hyperparameters?

We want the best model

Resources are expensive

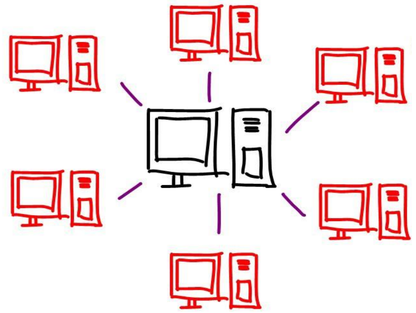
Model training is time-consuming



Tune is built with Deep Learning as a priority.

Resource Aware Scheduling

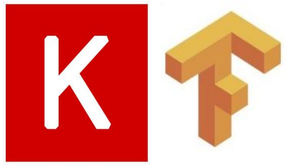
Seamless Distributed Execution



Simple API for new algorithms

```
class TrialScheduler:
    def on_result(self, trial, result): ...
    def choose_trial_to_run(self): ...
```

Framework Agnostic




ray.readthedocs.io/en/latest/tune.html

Tune is simple to use.

```
# Function-based API
def train():
  for _ in range(N):
    reporter(...)
```

```
# Class-based API
class MyModel(Trainable):
  def _setup(); def _train();
  def _save(); def _restore();
```

Two simple APIs for model training

Tune API

Bayesian Optimization | Grid Search | ... | HyperBand | Population Based Training

Pair **search algorithms** and **trial schedulers** to guide your distributed optimization.

ray.readthedocs.io/en/latest/tune.html

Exercises

Content available at github.com/ray-project/tutorial/.

Part 2: Hyperparameter Tuning with Tune

- tune_exercises/ - Tutorial.ipynb

Click on the “Binder” link and wait a couple minutes.

Binder not working? Backup:

- run locally with "pip install ray[rllib] tensorflow jupyterlab"

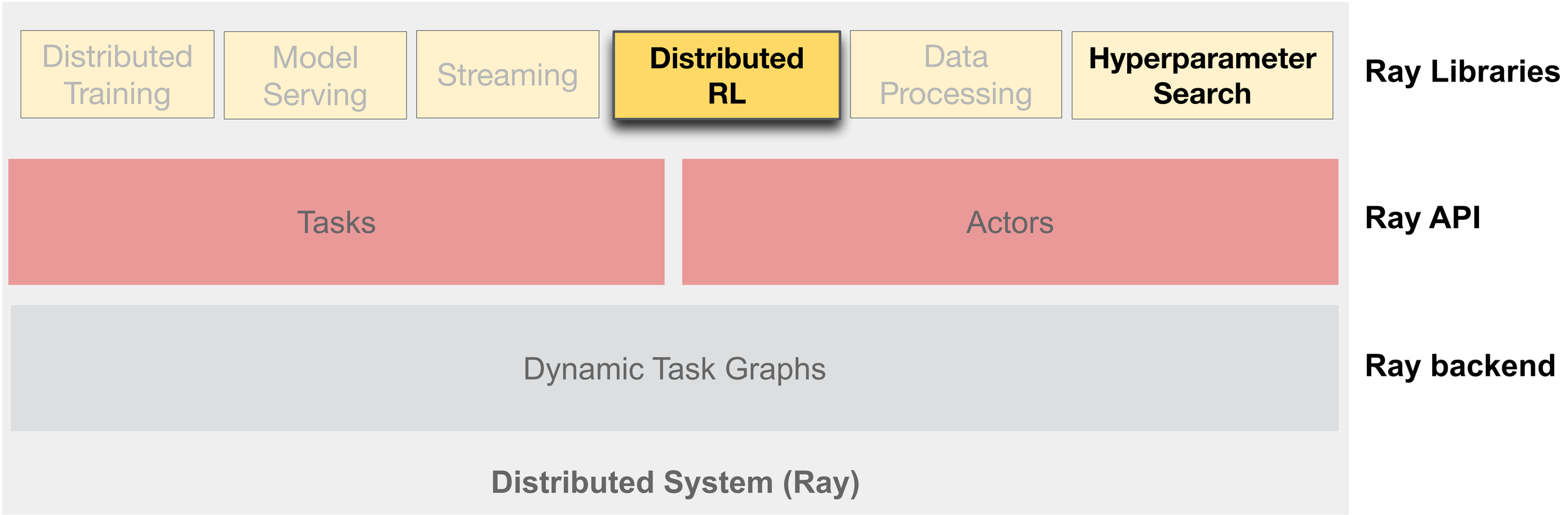
RLlib

A scalable and unified library for reinforcement learning

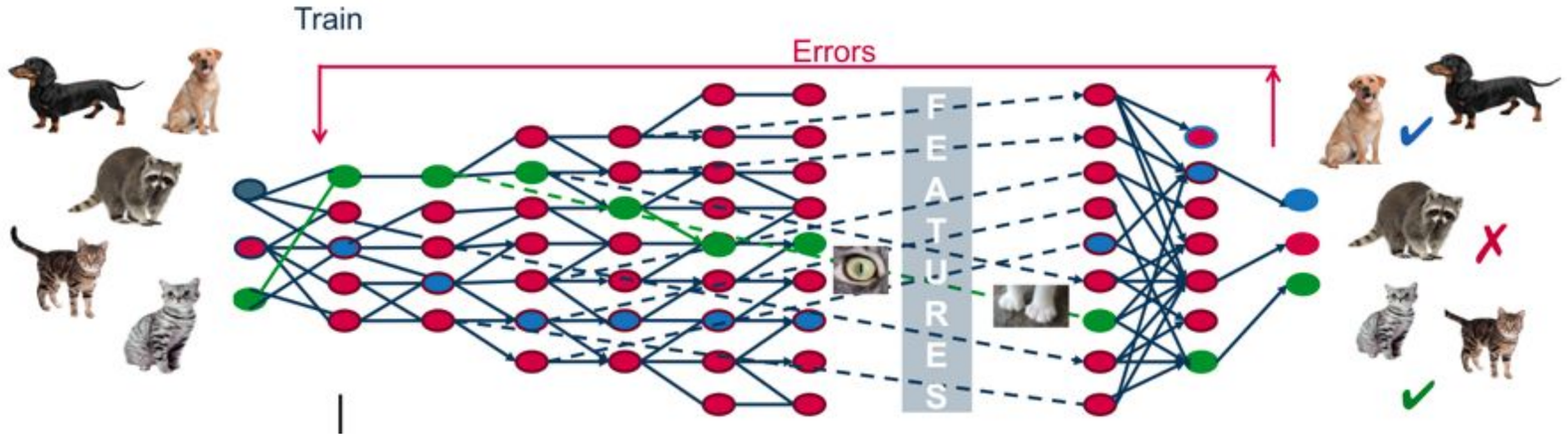
https://rllib.io

riselab
UC Berkeley

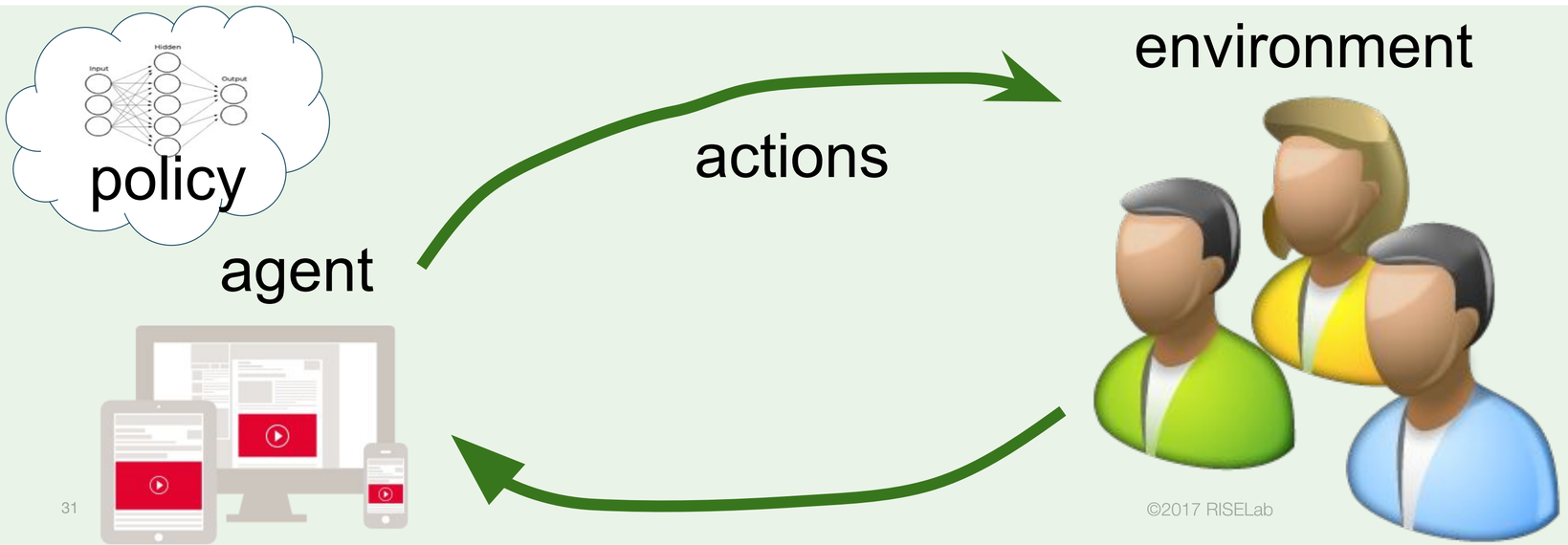
What is RLlib?



Background: What is reinforcement learning?



Supervised Learning



Reinforcement Learning

Growing number of RL applications

Robotics

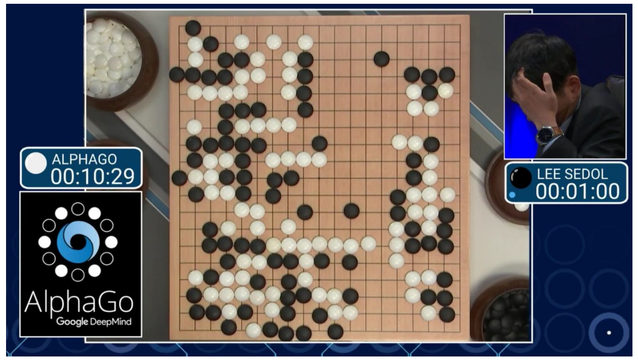
Industrial Control

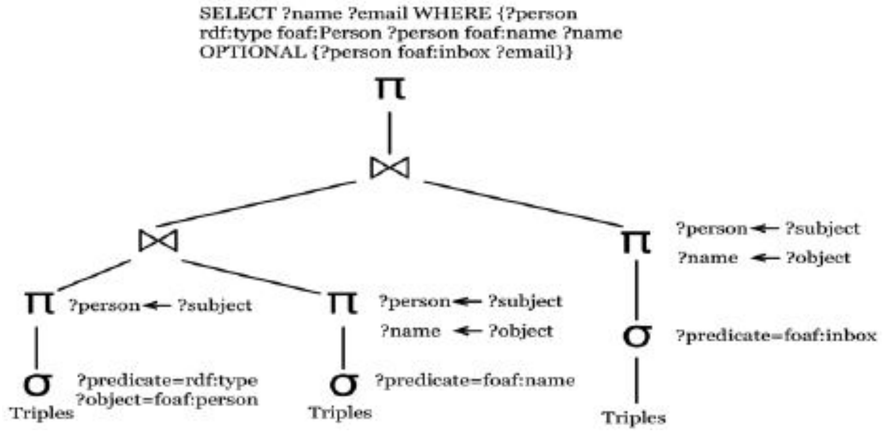
Advertising

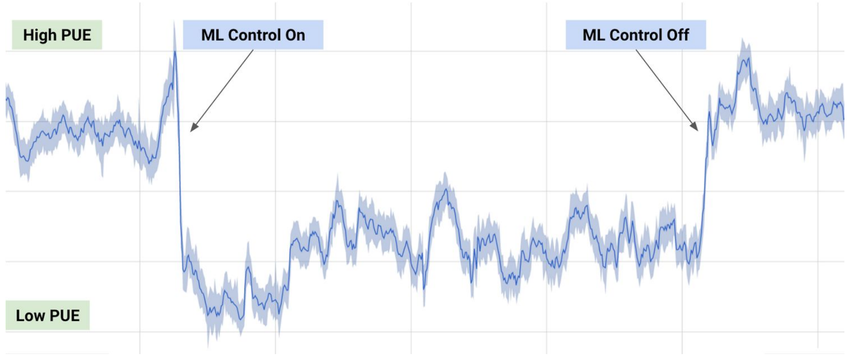
System Optimization

Finance

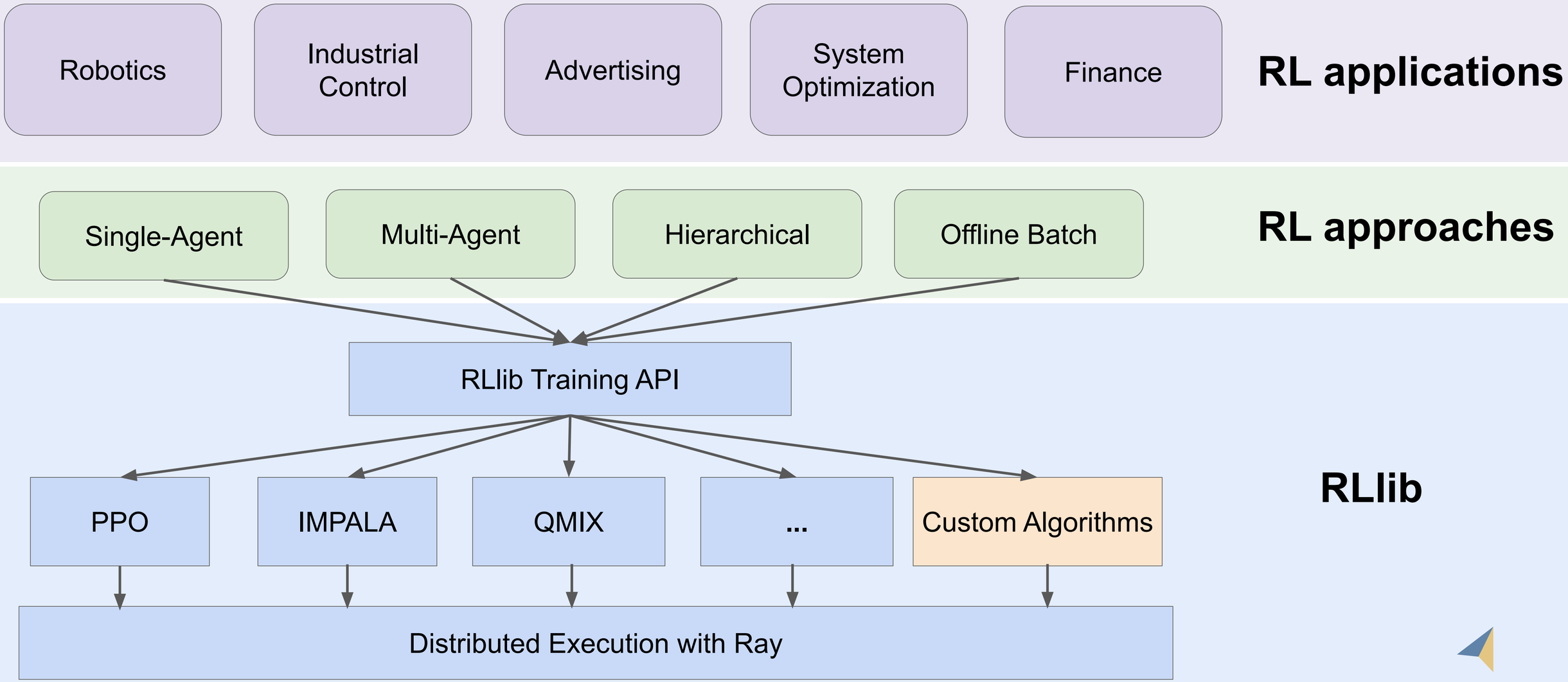
RL applications





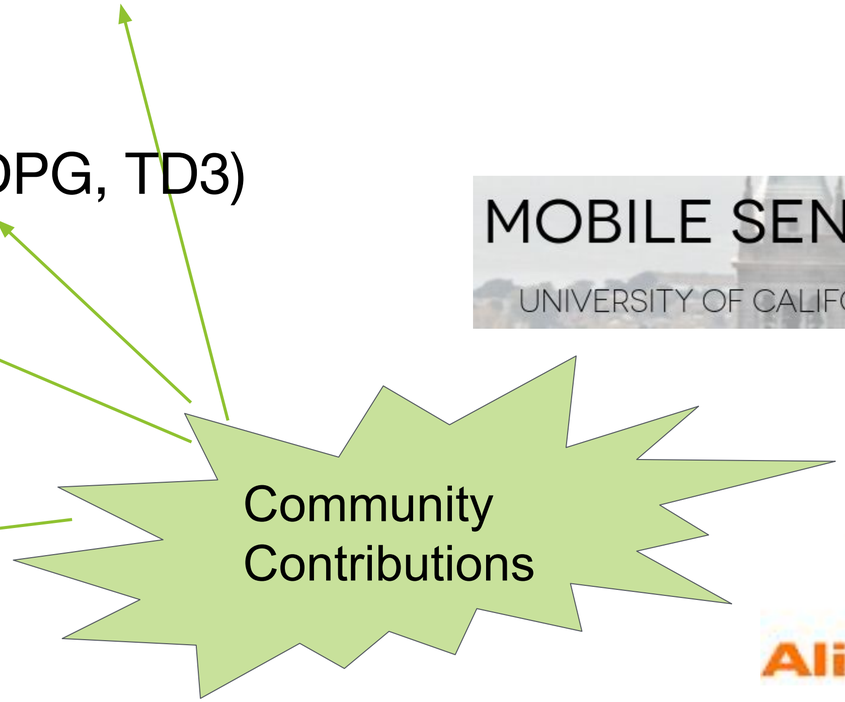


A scalable, unified library for reinforcement learning



Reference Algorithms

- **High-throughput architectures**
  - Distributed Prioritized Experience Replay (Ape-X)
  - Importance Weighted Actor-Learner Architecture (IMPALA)
- **Gradient-based**
  - Advantage Actor-Critic (A2C, A3C)
  - Deep Deterministic Policy Gradients (DDPG, TD3)
  - Deep Q Networks (DQN, Rainbow)
  - Policy Gradients
  - Proximal Policy Optimization (PPO)
- **Derivative-free**
  - Augmented Random Search (ARS)
  - Evolution Strategies



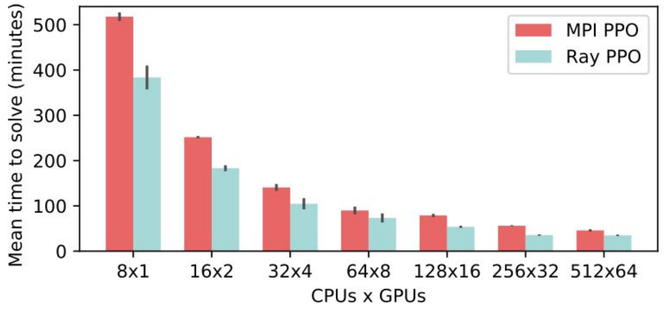
MOBILE SENSING LAB

UNIVERSITY OF CALIFORNIA, BERKELEY

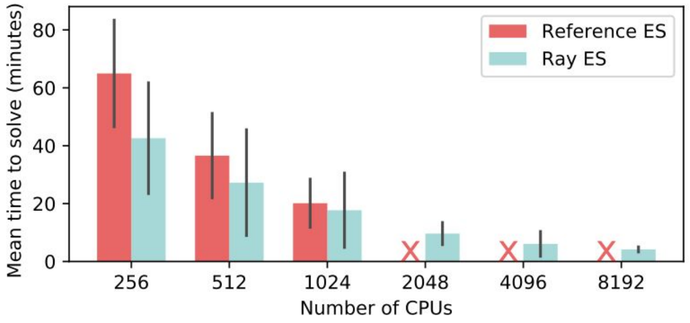
Alibaba Group

Performance

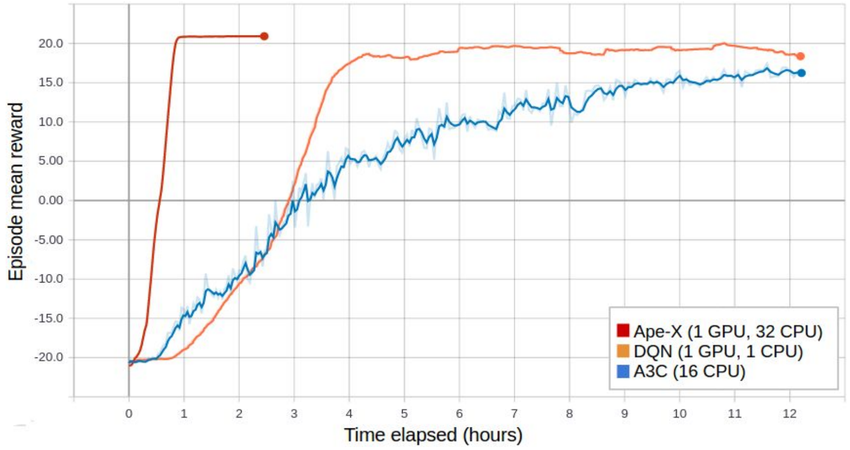
Distributed PPO
(vs OpenMPI)



Evolution Strategies
(vs Redis-based)



Ape-X Distributed DQN, DDPG



Amazon SageMaker RL

End-to-end examples for classic RL and real-world RL applications

Robotics | Industrial Control | HVAC | Autonomous Vehicles | Operations | Finance | Games | NLP

RL Environments to model real-world problems

- **AWS Simulation Environments**: Amazon Sumerian | AWS RoboMaker
- **Open Source Environments**: EnergyPlus | RoboSchool | PyBullet | ...
- **Custom Environments**: Bring Your Own
- **Commercial simulators**: MATLAB & Simulink

Open AI Gym

RL Toolkits that provide RL agent algorithm implementations

- **RL-Coach**: DQN | PPO | HER | Rainbow | ...
- **RL-Ray RLLib**: APEX | ES | IMPALA | A3C | ...
- **Open AI Baselines**: TRPO | GAIL | ...
- ...

SageMaker Deep Learning Frameworks

TensorFlow | MxNet | PyTorch | Chainer

Training Options

Single Machine / Distributed | Local / Remote simulation | CPU / GPU Hardware

SageMaker supported | Customer BYO

Exercises

Content available at github.com/ray-project/tutorial/.

Part 3: Applied Reinforcement Learning

- rllib_exercises/ - exercise01, exercise02

Click on the “Binder” link and wait a couple minutes.

Binder not working? Backup:

- run locally with "pip install ray[rllib] tensorflow jupyterlab"

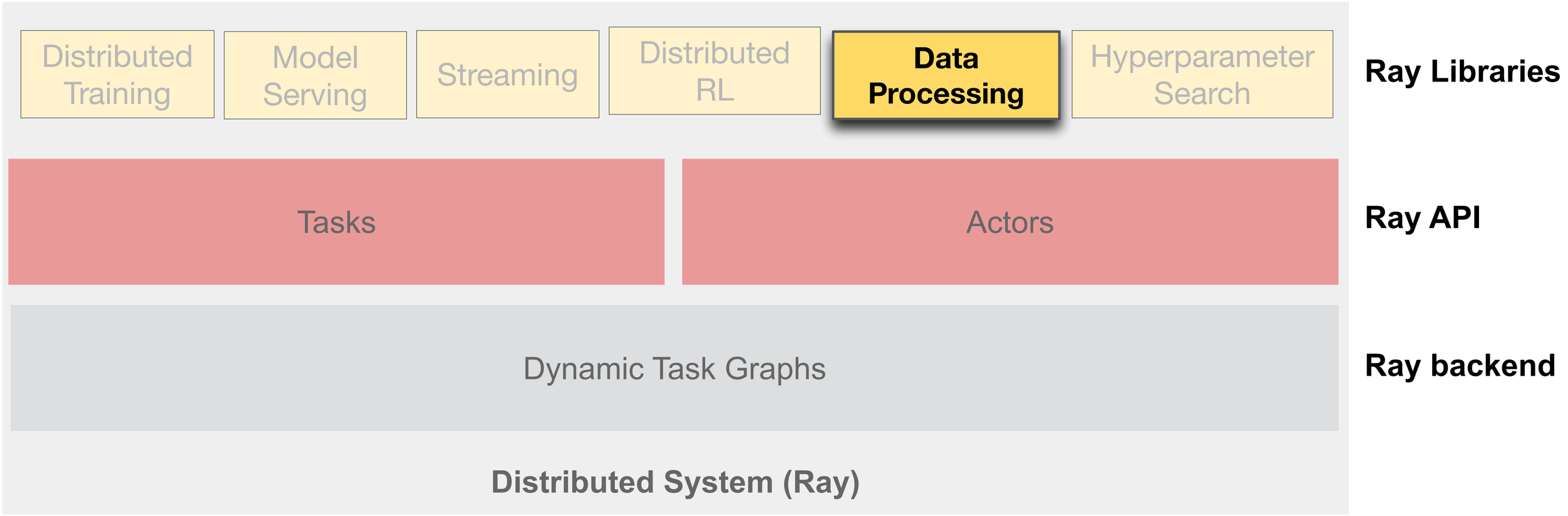
MODIN

Accelerate your Pandas workflows by changing a single line of code

riselab
UC Berkeley

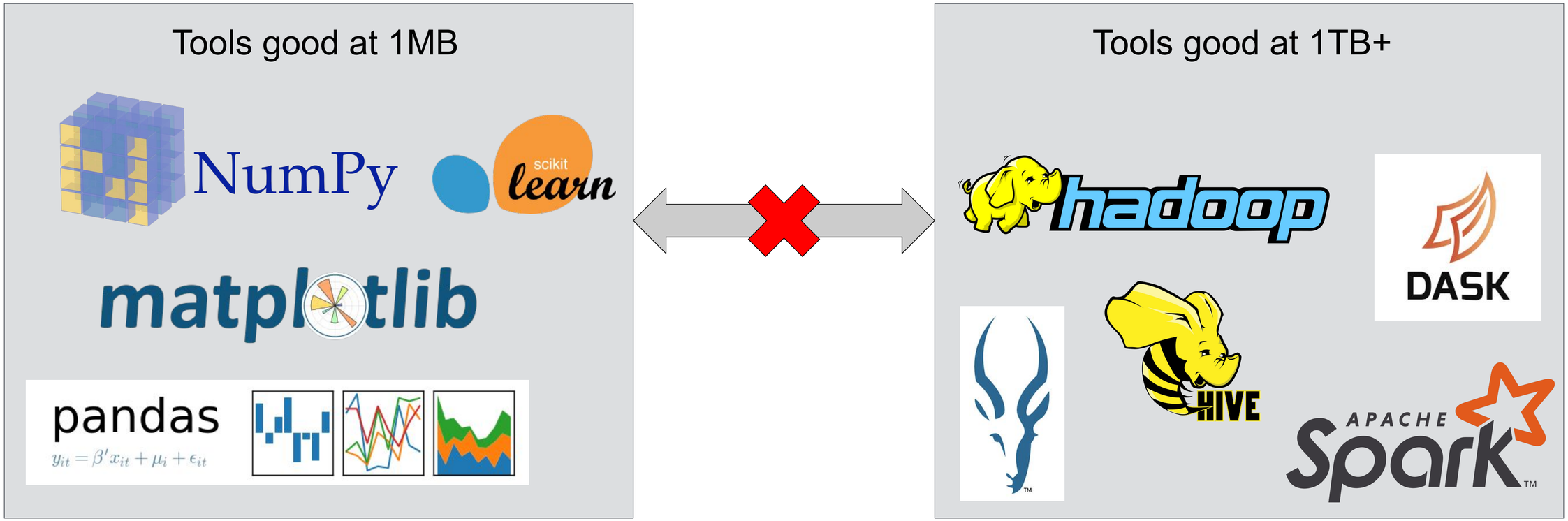
ray.readthedocs.io/en/latest/tune.html

What is Modin?



Modin: Pandas on Ray

Accelerate your pandas workloads by changing one line of code



Modin: Pandas on Ray

Accelerate your pandas workloads by changing one line of code

To use Modin, replace the pandas import:

```
# import pandas as pd
import modin.pandas as pd
```

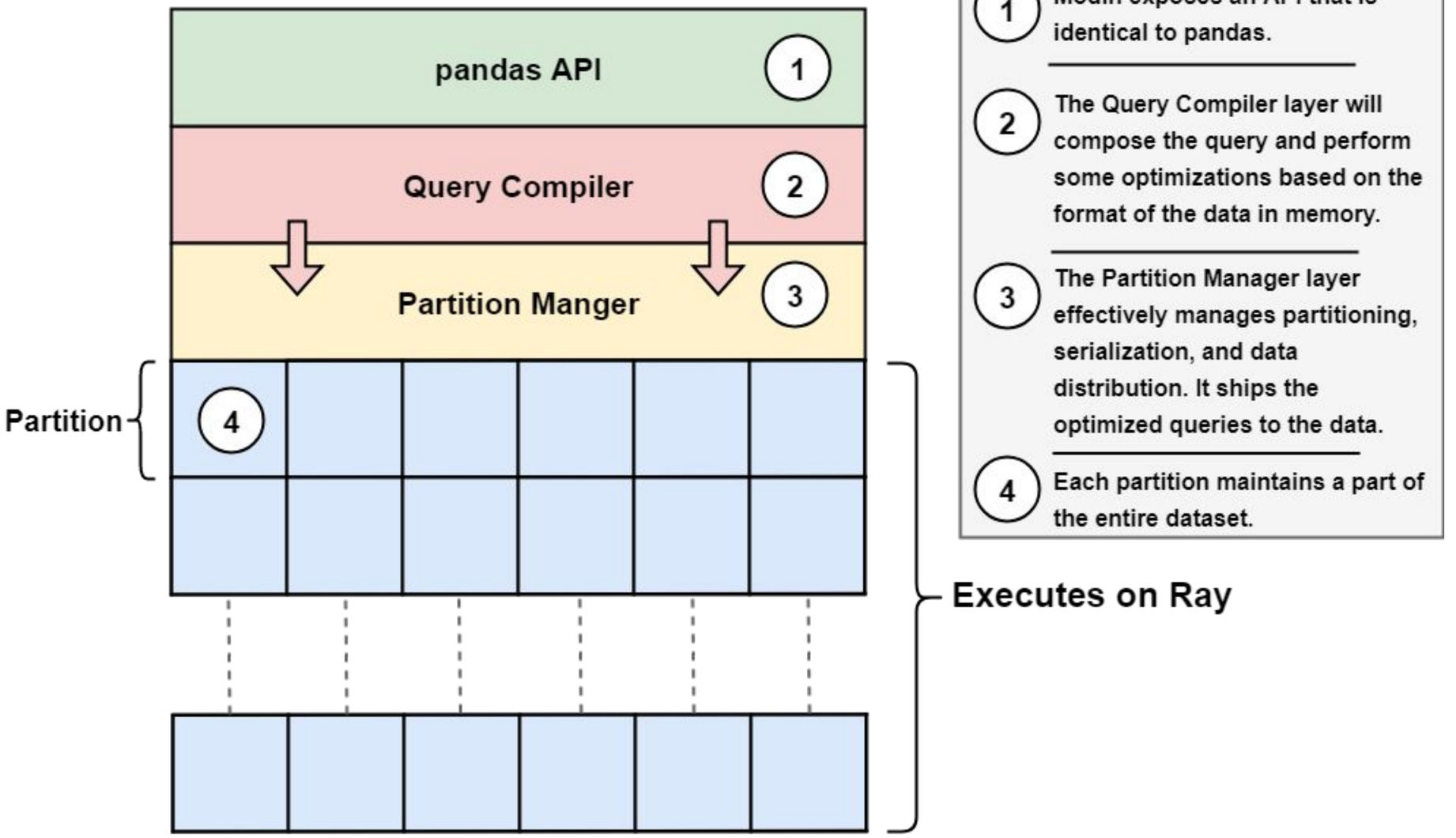
Installation

Modin can be installed from PyPI:

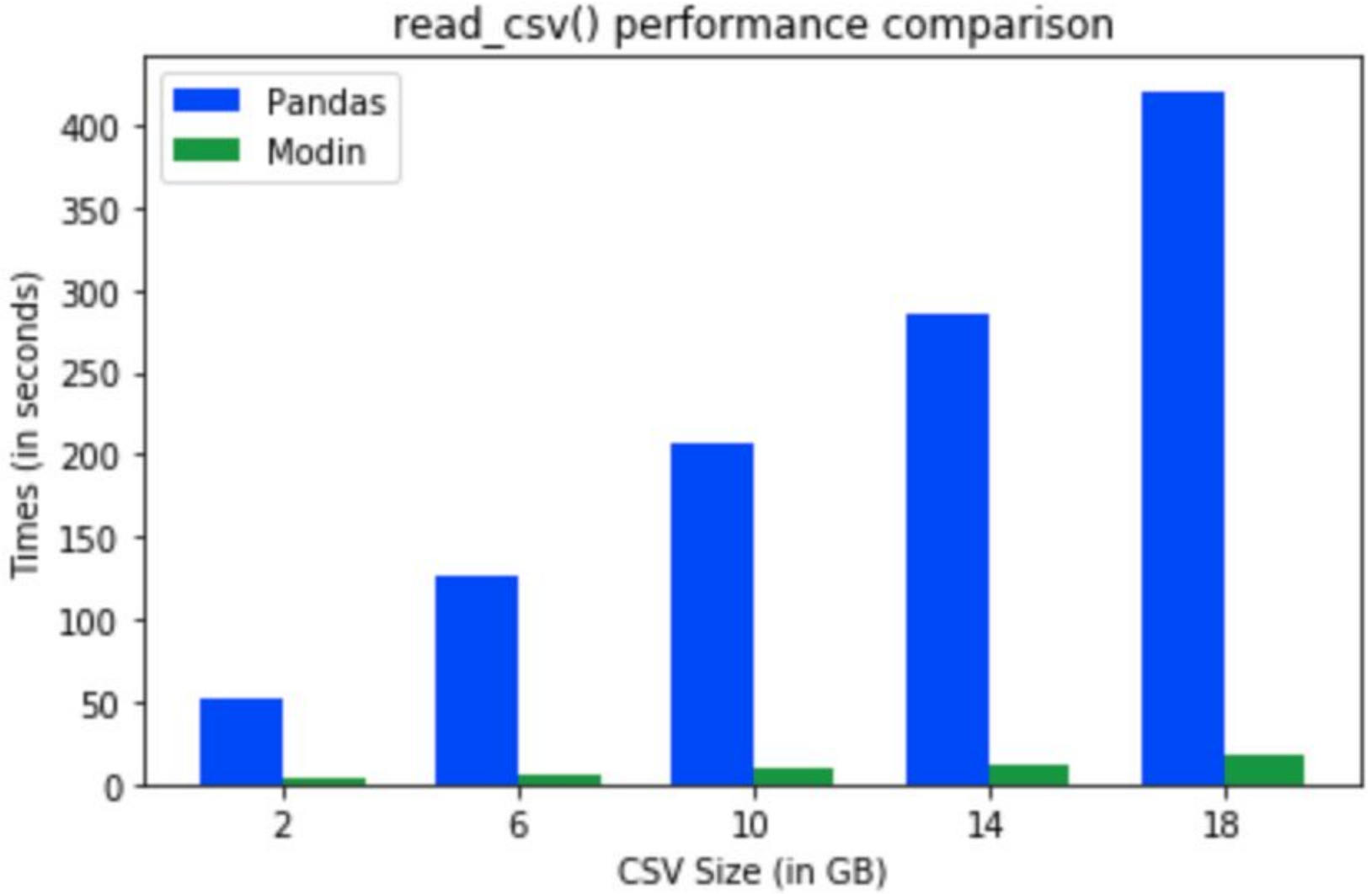
```
pip install modin
```


Why Modin?

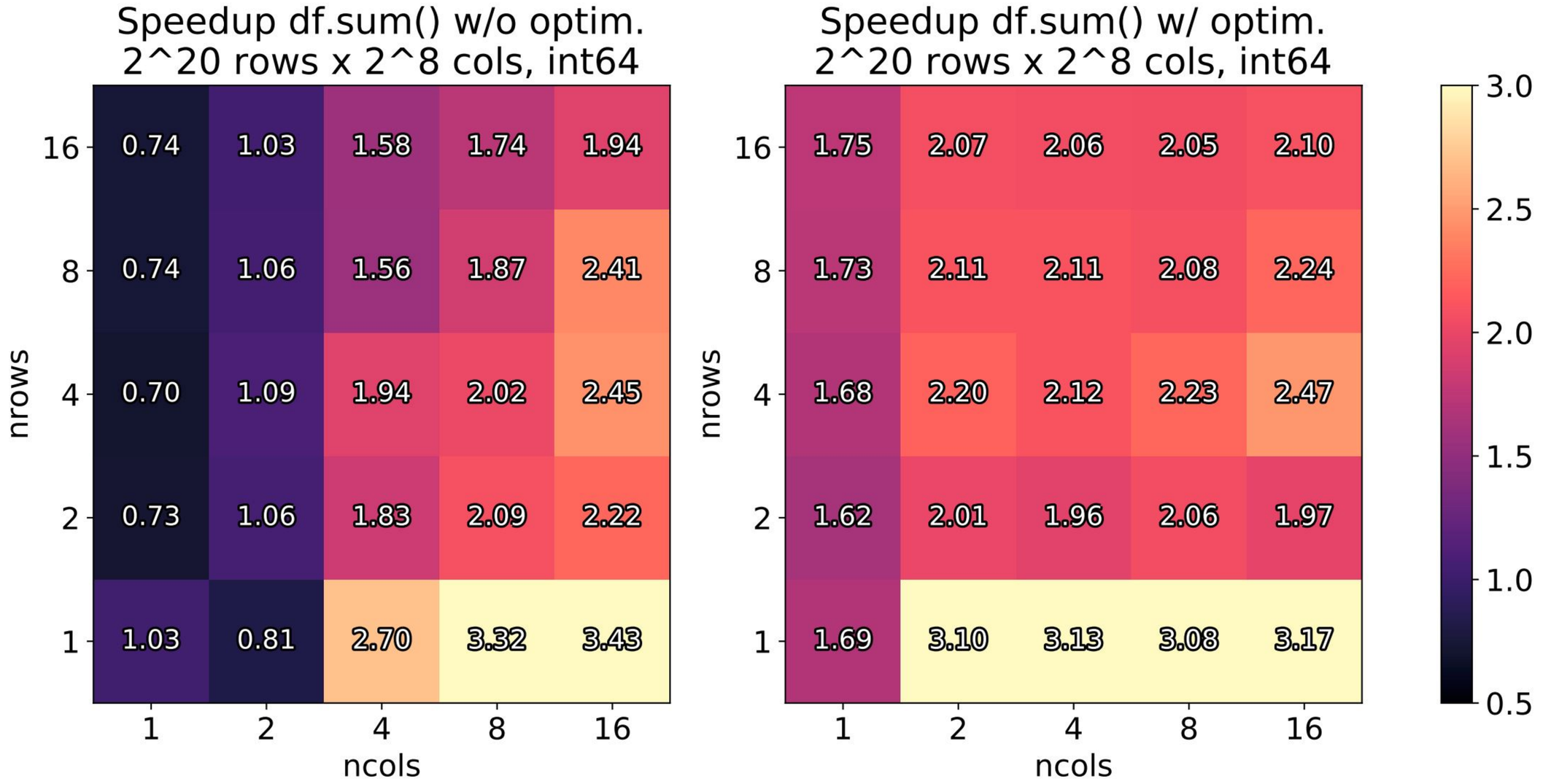
- Faster pandas, even on your laptop
  - Up to 4x speed improvement over pandas on 4 physical cores
- Cluster support -- experimental!
- A DataFrame library aimed at bridging the gap between MB-scale and TB-scale data



Performance

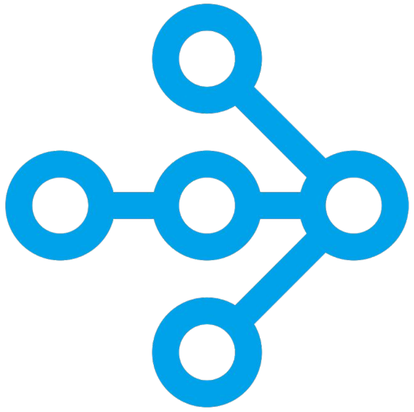


Performance



Conclusion

- Ray is an open source project for distributed computing
- **special-purpose** distributed systems -> **general-purpose** distributed system
- Support for the full ML lifecycle (data collection, training, simulation, serving)



RAY

github.com/ray-project/ray