Ray - Scalability from a Laptop to a Cluster

Dean Wampler - July 1, 2020
dean@anyscale.com
@deanwampler
ray.io
anyscale.com

Outline

- Why Ray?
- ML/AI Ray Libraries
- Ray for Microservices
- Adopting Ray

RAY

Why Ray??

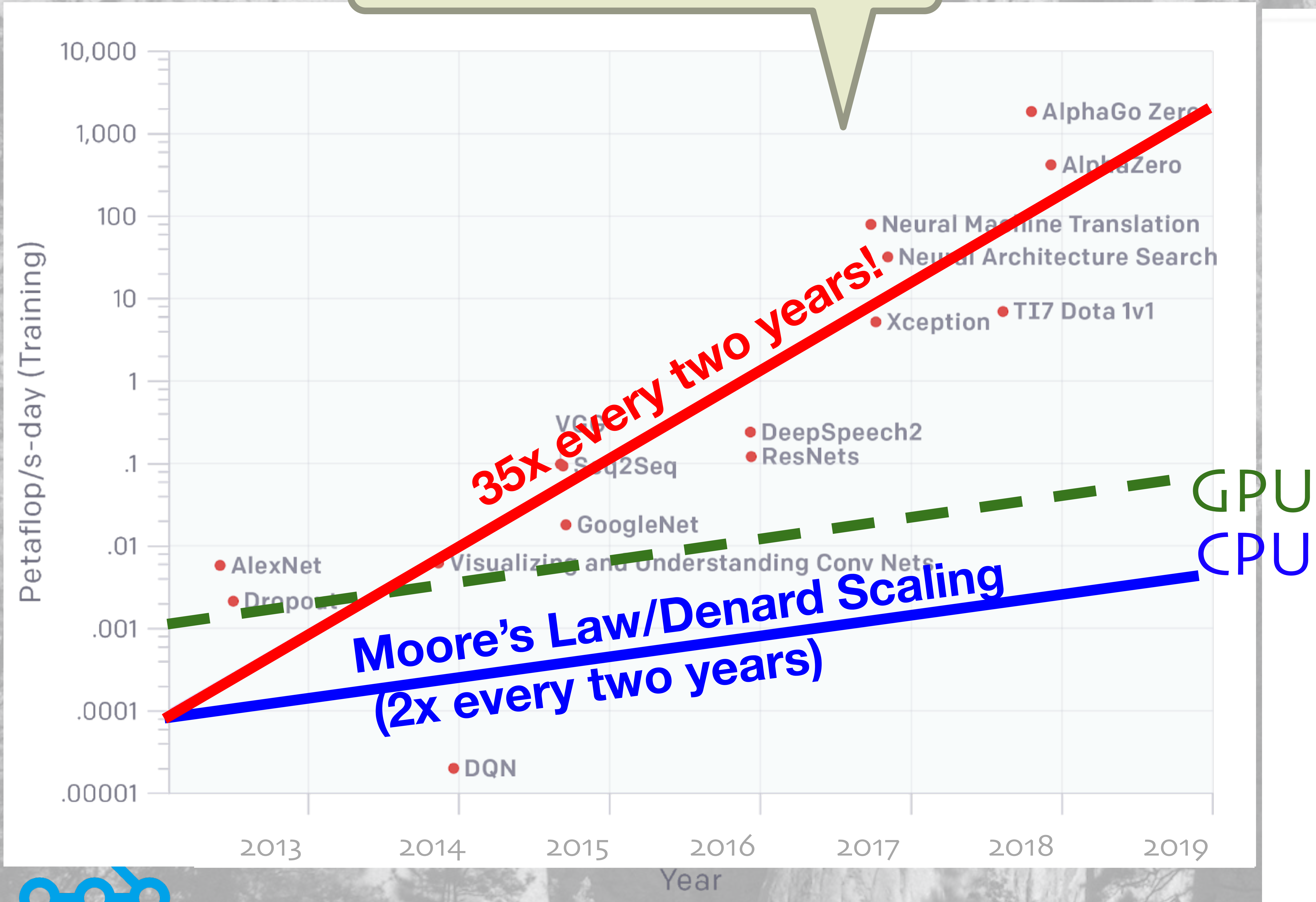

@deanwampler

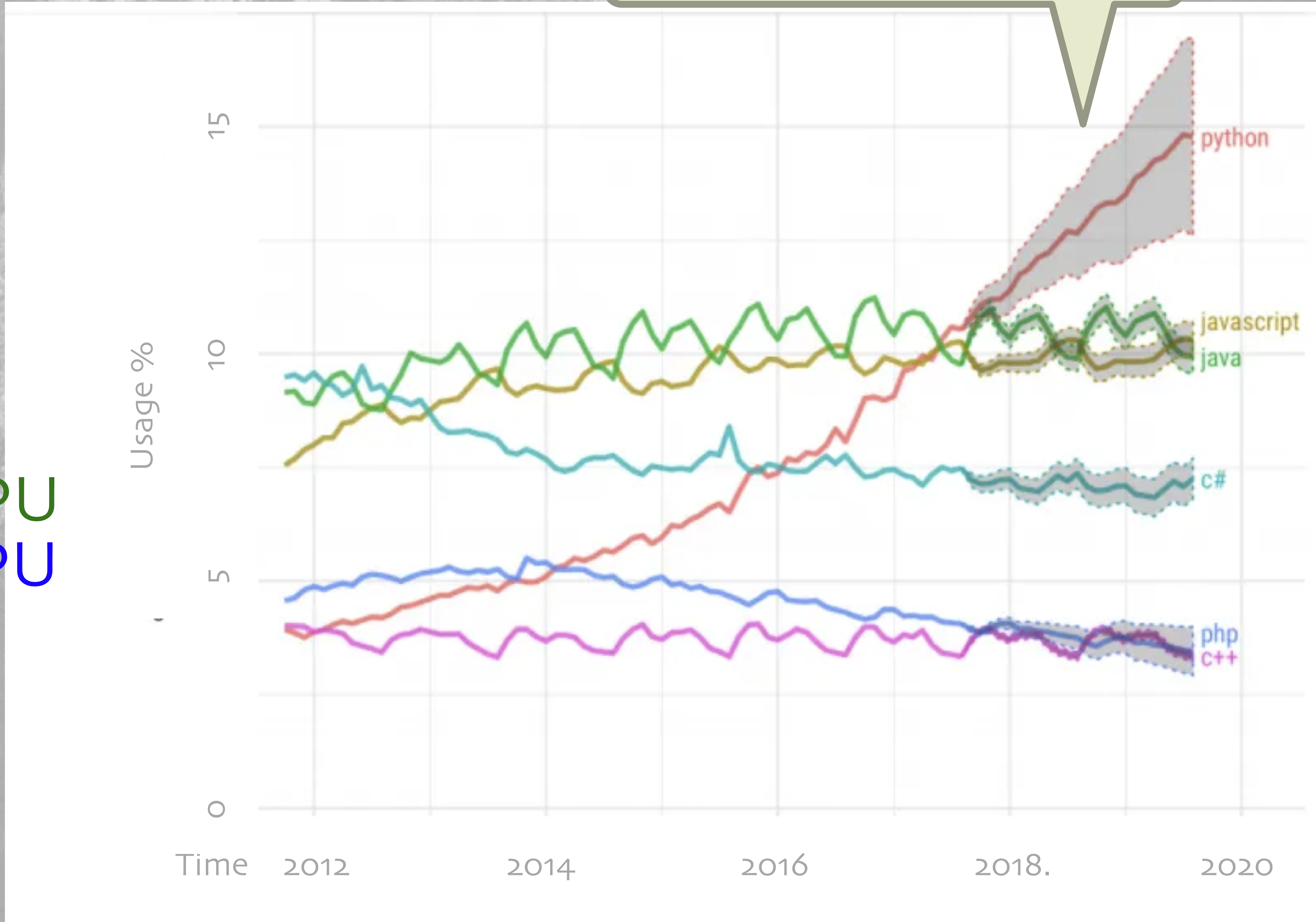
Two Major Trends

Model sizes and therefore compute requirements outstripping Moore's Law

Hence, there is a pressing need for robust, easy to use solutions for distributed Python

Python growth driven by ML/AI and other data science workloads





https://openai.com/blog/ai-and-compute/

@deanwampler

The ML Landscape Today

All require distributed implementations to scale

Featurization	Streaming	Hyperparam Tuning	Training	Simulation	Model Serving
Apache Spark, hadoop	Apache Spark, kafka	SIGOPT, talos AUTONOMY	HOROVOD, PyTorch, scikit learn	MPI, akka, Orleans	Clipper

The Ray Vision: Sharing a Common Framework

Domain-specific libraries for each subsystem

tune

raysgd

rllib

RAY SERVE

Featurization

Streaming

Hyperparam Tuning

Training

Simulation

Model Serving

Framework for distributed Python (and other languages…)

RAY

More libraries coming soon

API - Designed to Be Intuitive and Concise

Functions -> Tasks

```
def make_array(…):
    a = … # Construct a NumPy array
    return a


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

The Python you already know…

API - Designed to Be Intuitive and Concise

Functions -> Tasks

```
@ray.remote
def make_array(…):
    a = … # Construct a NumPy array
    return a

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

Now these functions are remote “tasks"

For completeness, add these first:

```
import ray
import numpy as np
ray.init()
```

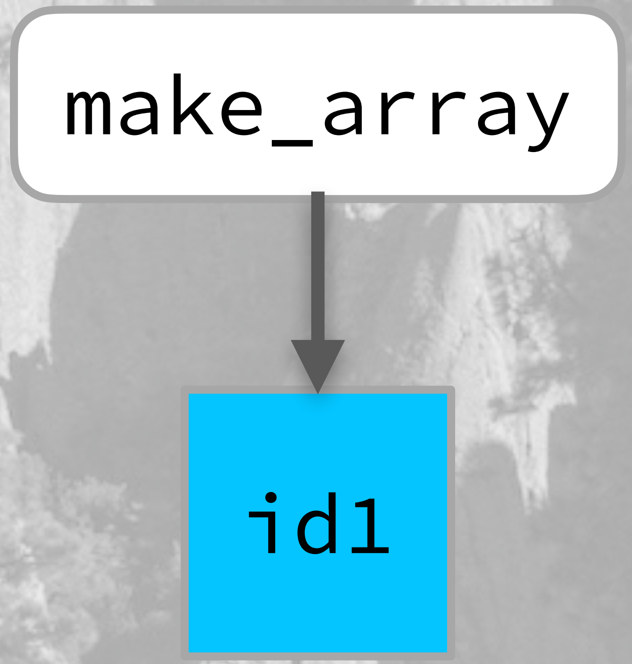
API - Designed to Be Intuitive and Concise

Functions -> Tasks

```python
@ray.remote
def make_array(…):
    a = … # Construct a NumPy array
    return a

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

id1 = make_array.remote(…)
```



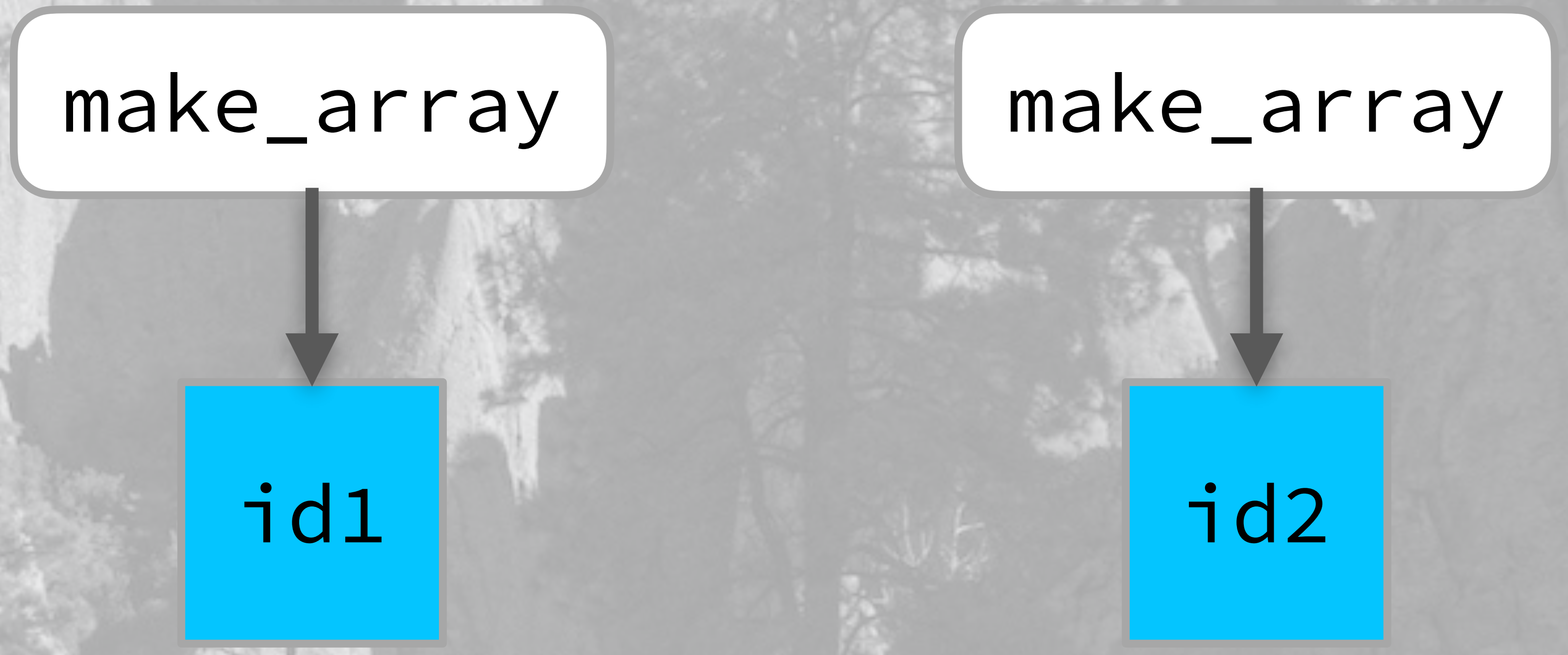
API - Designed to Be Intuitive and Concise

Functions -> Tasks

```python
@ray.remote
def make_array(…):
    a = … # Construct a NumPy array
    return a

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

id1 = make_array.remote(…)
id2 = make_array.remote(…)
```



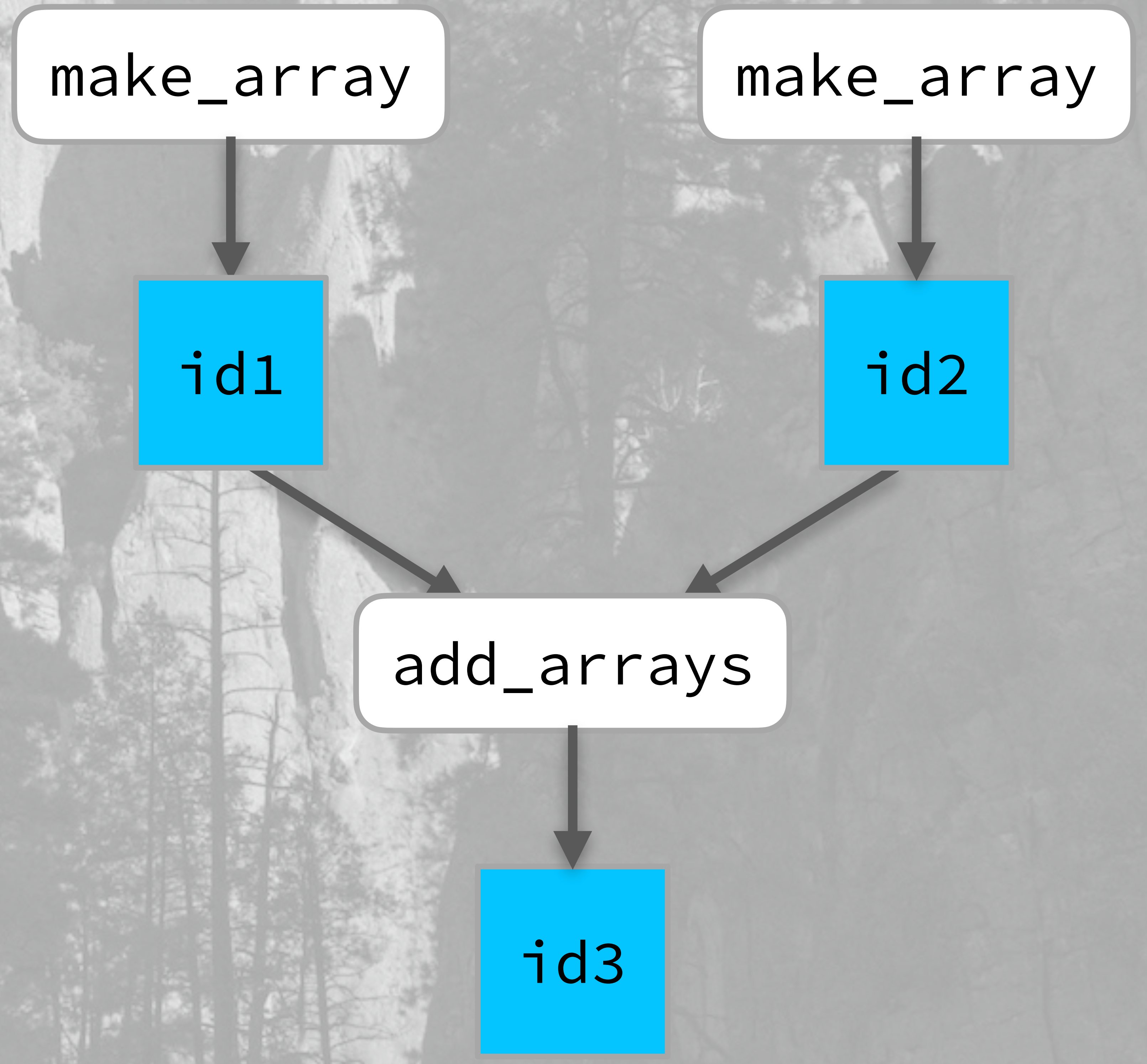
API - Designed to Be Intuitive and Concise

Functions -> Tasks

```
@ray.remote
def make_array(…):
    a = … # Construct a NumPy array
    return a

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

id1 = make_array.remote(…)
id2 = make_array.remote(…)
id3 = add_arrays.remote(id1, id2)
```



API - Designed to Be Intuitive and Concise

Functions -> Tasks

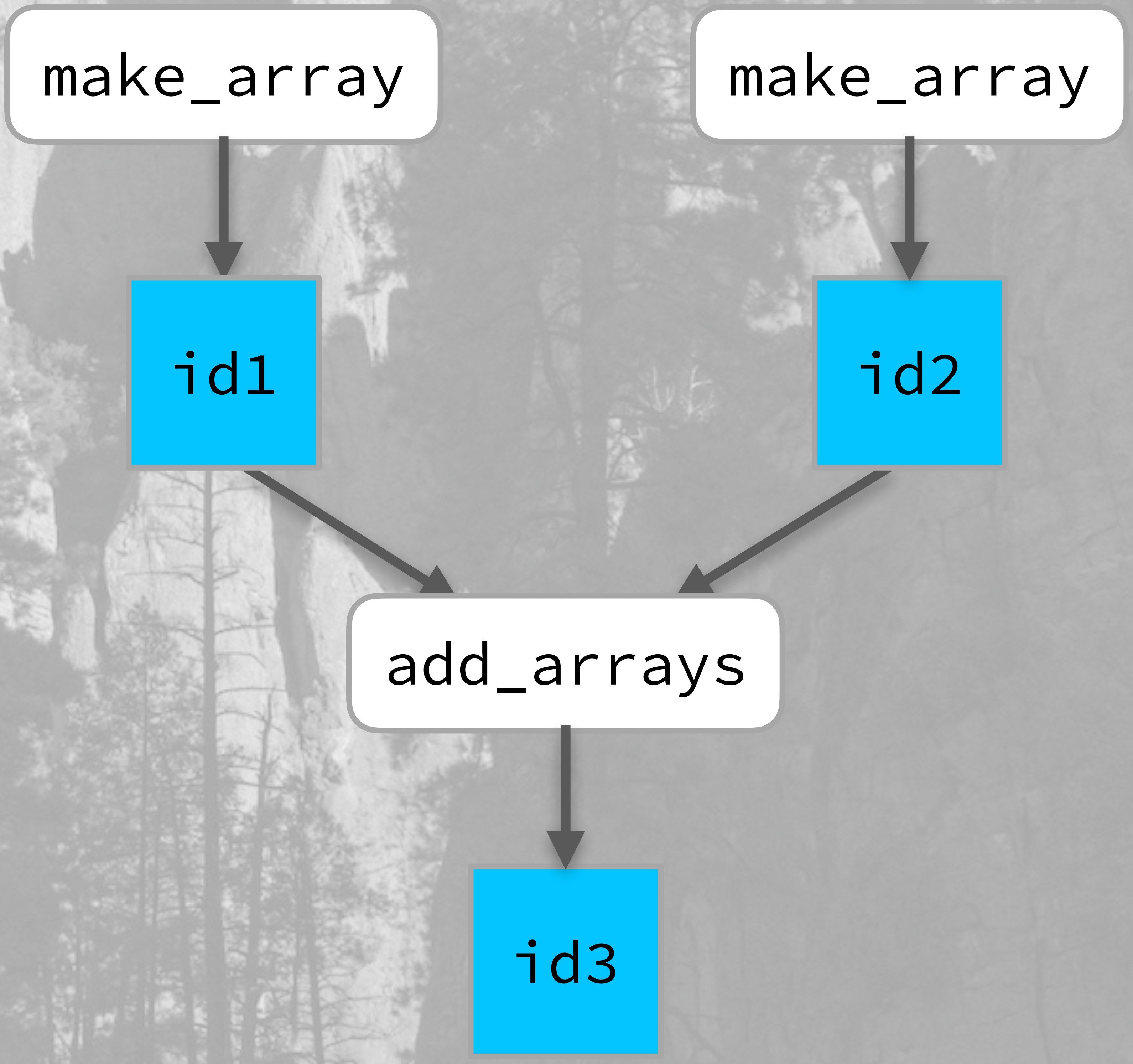
```
@ray.remote
def make_array(…):
    a = … # Construct a NumPy array
    return a

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

id1 = make_array.remote(…)
id2 = make_array.remote(…)
id3 = add_arrays.remote(id1, id2)
ray.get(id3)
```

Ray handles extracting the arrays from the object ids

Ray handles sequencing of async dependencies



API - Designed to Be Intuitive and Concise

What about distributed state?

Functions -> Tasks

```
@ray.remote
def make_array(…):
    a = … # Construct a NumPy array
    return a

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

id1 = make_array.remote(…)
id2 = make_array.remote(…)
id3 = add_arrays.remote(id1, id2)
ray.get(id3)
```

API - Designed to Be Intuitive and Concise

Functions -> Tasks

```
@ray.remote
def make_array(…):
    a = … # Construct a NumPy array
    return a

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

id1 = make_array.remote(…)
id2 = make_array.remote(…)
id3 = add_arrays.remote(id1, id2)
ray.get(id3)
```

The Python classes you love…

Classes -> Actors

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

API - Designed to Be Intuitive and Concise

Functions -> Tasks

```
@ray.remote
def make_array(…):
    a = … # Construct a NumPy array
    return a

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

id1 = make_array.remote(…)
id2 = make_array.remote(…)
id3 = add_arrays.remote(id1, id2)
ray.get(id3)
```

Classes -> Actors

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

… now a remote "actor"

You need a "getter" method to read the state.

API - Designed to Be Intuitive and Concise

Functions -> Tasks

```
@ray.remote
def make_array(…):
    a = … # Construct a NumPy array
    return a

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

id1 = make_array.remote(…)
id2 = make_array.remote(…)
id3 = add_arrays.remote(id1, id2)
ray.get(id3)
```

Classes -> Actors

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

c = Counter.remote()
id4 = c.increment.remote()
id5 = c.increment.remote()
ray.get([id4, id5]) # [1, 2]
```

API - Designed to Be Intuitive and Concise

Functions -> Tasks

```python
@ray.remote
def make_array(…):
    a = … # Construct a NumPy array
    return a

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

id1 = make_array.remote(…)
id2 = make_array.remote(…)
id3 = add_arrays.remote(id1, id2)
ray.get(id3)
```

Classes -> Actors

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

c = Counter.remote()
id4 = c.increment.remote()
id5 = c.increment.remote()
ray.get([id4, id5]) # [1, 2]
```

Optional configuration specifications

RAY

Machine Learning with Ray-based Libraries

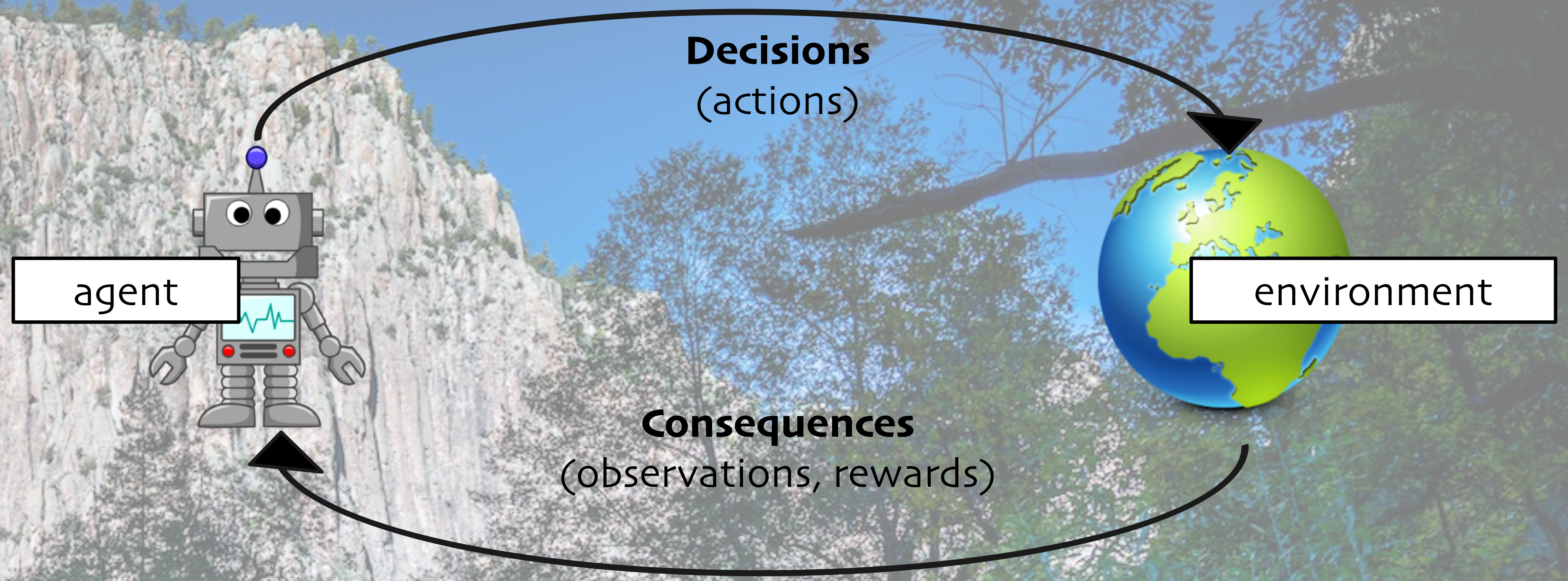

@deanwampler

Ray Libraries

Reinforcement Learning - Ray RLlib

rllib.io

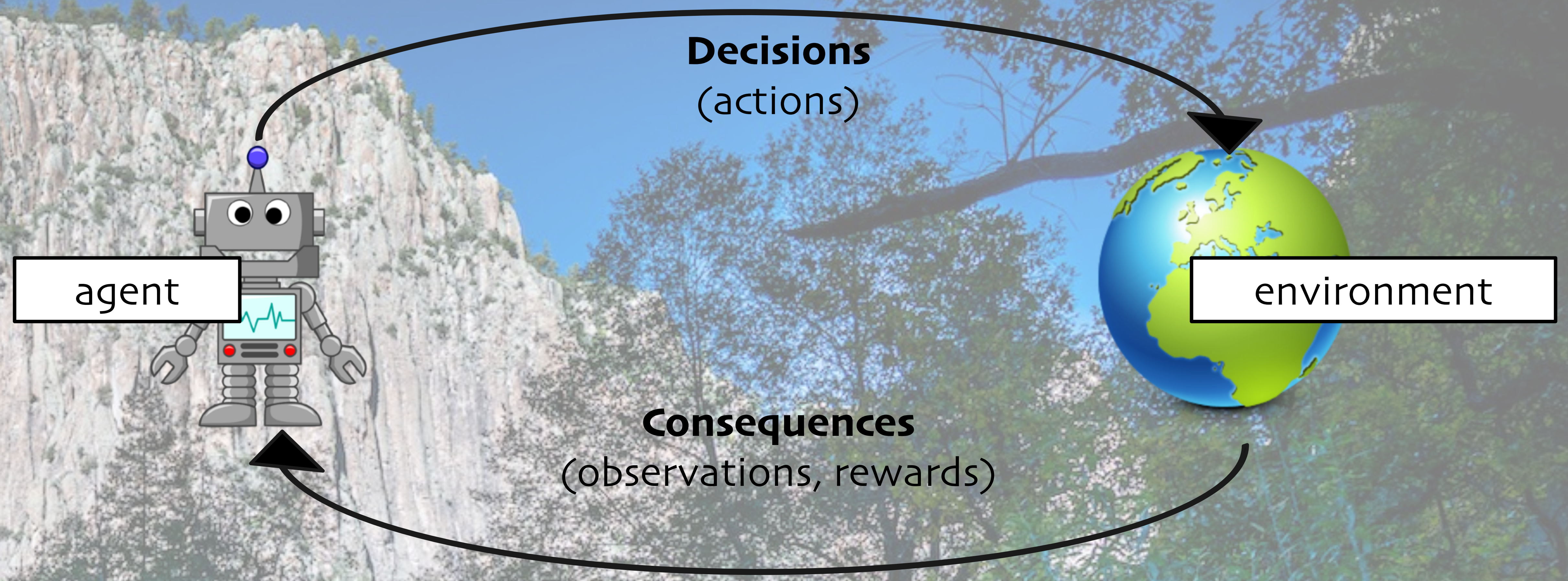
Reinforcement Learning



Games | Robotics, Autonomous Vehicles | Industrial Processes | System Optimization | Advertising, Recommendations | Finance

RL applications

Reinforcement Learning



Games

Robotics, Autonomous Vehicles

Industrial Processes

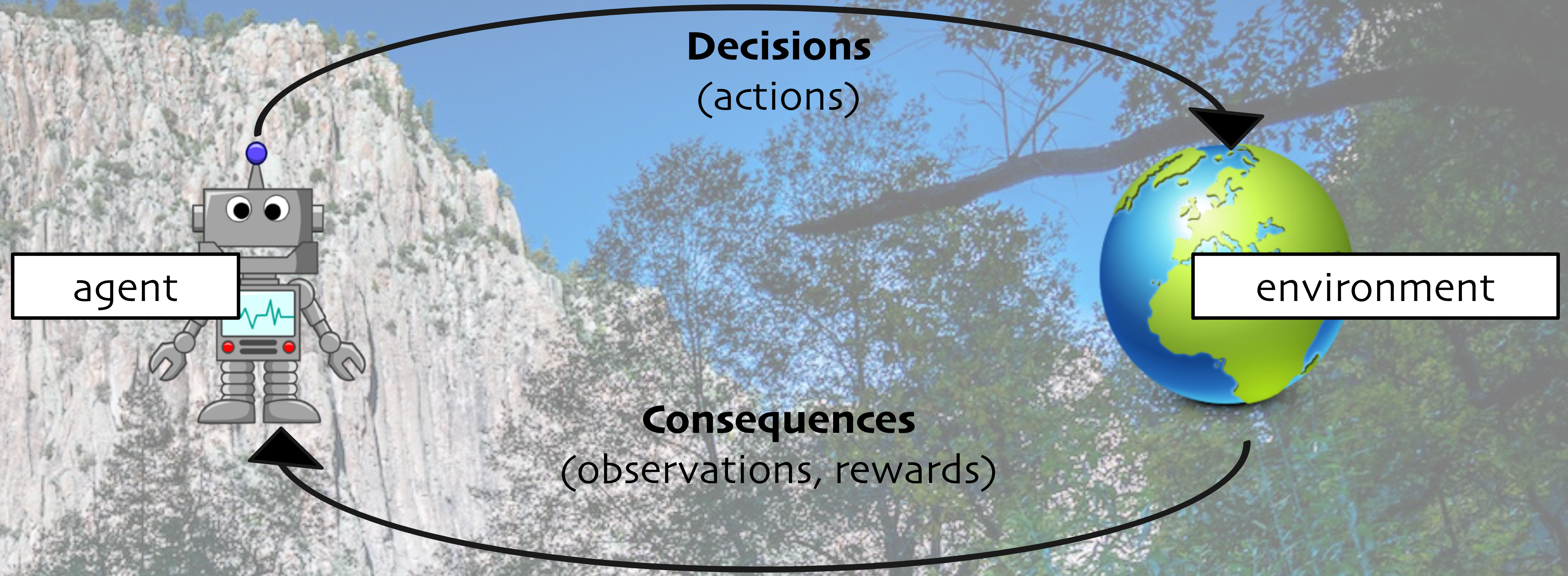
System Optimization

Advertising, Recommendations

Finance

RL applications

Reinforcement Learning



Games

Robotics, Autonomous Vehicles

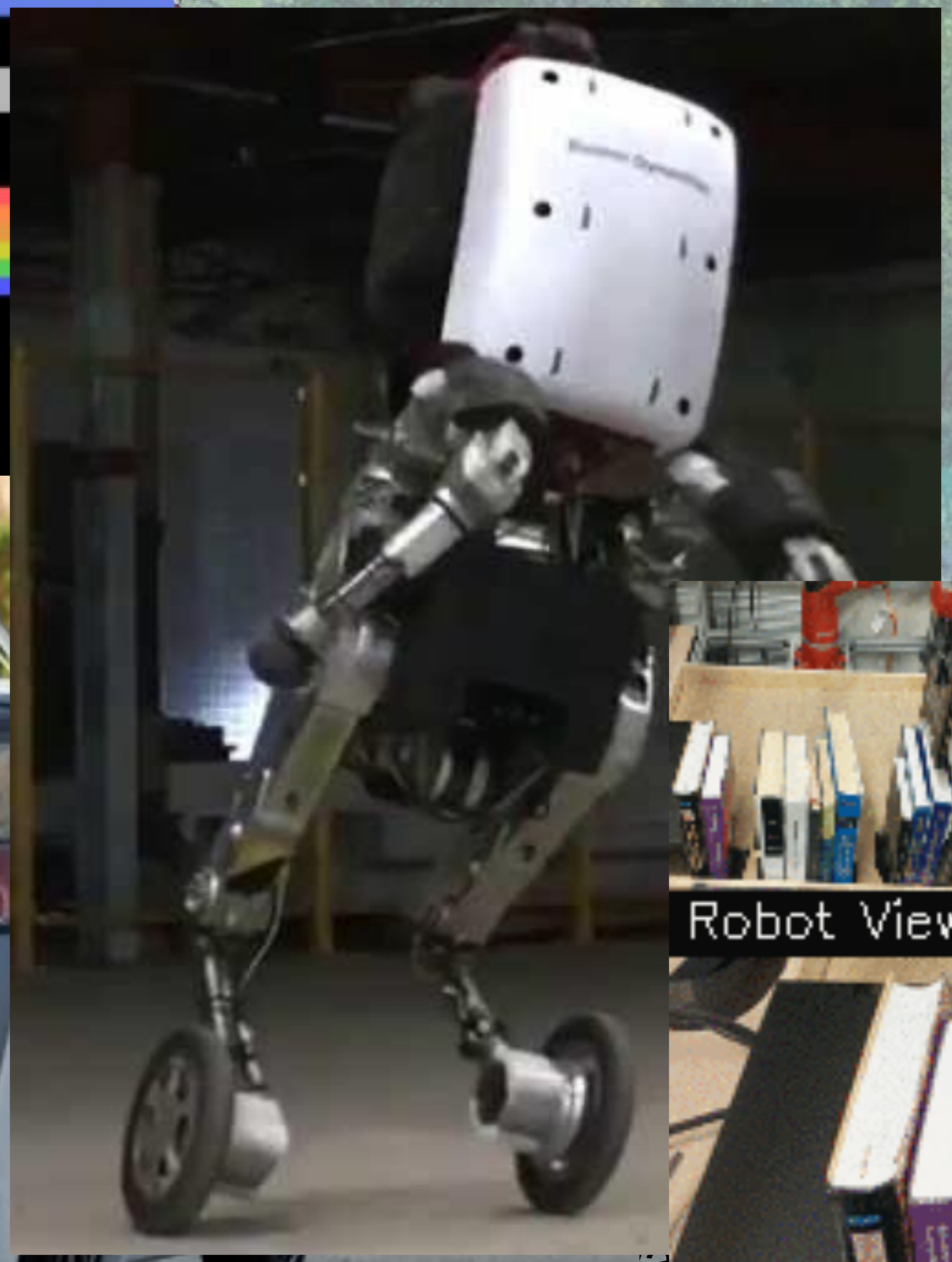
Industrial Processes

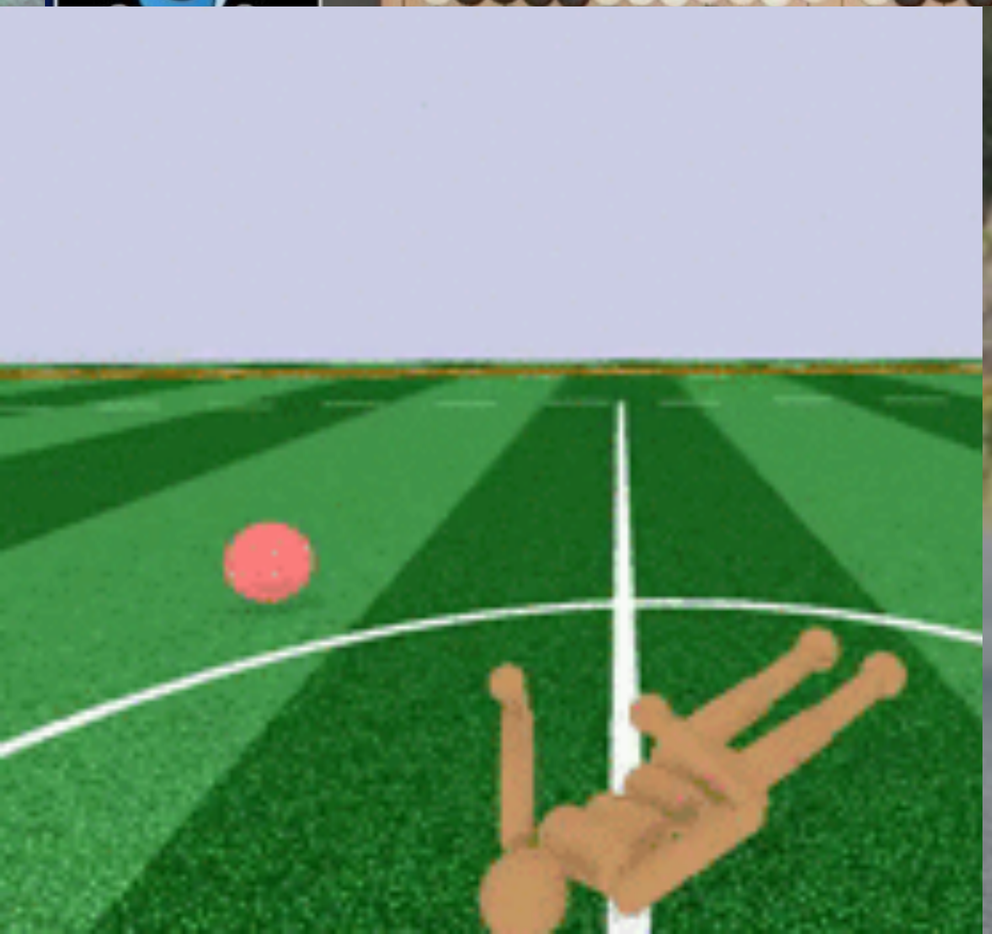
System Optimization

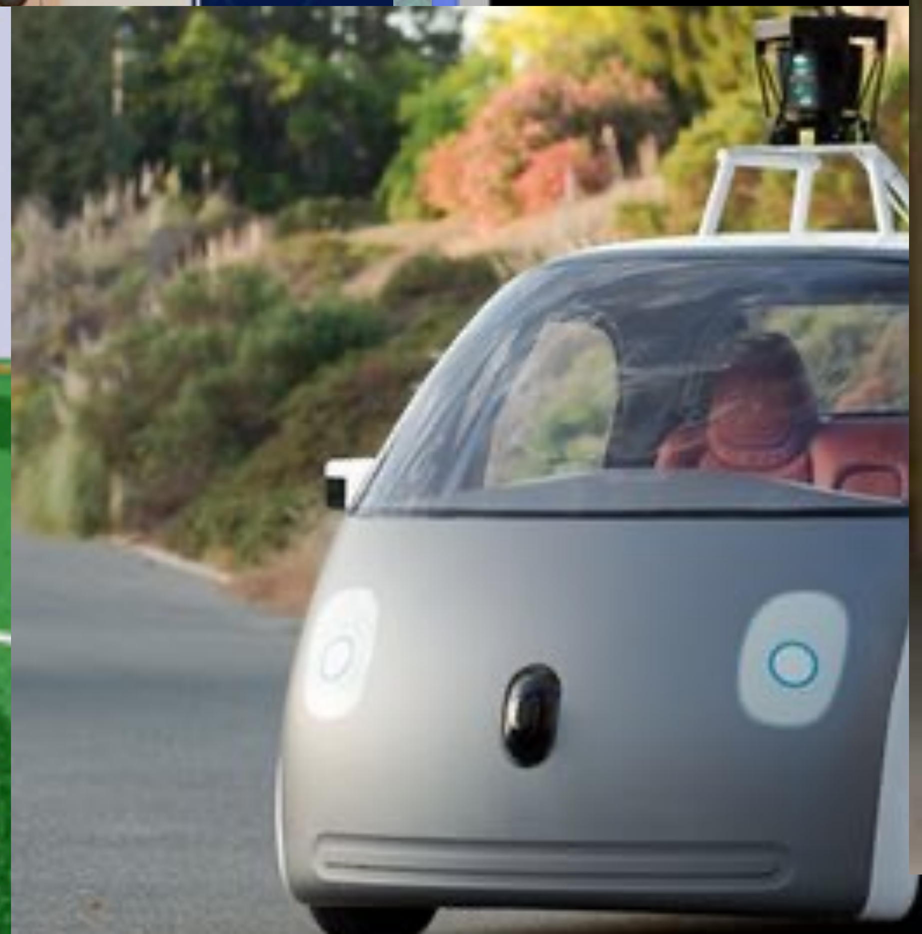
Advertising, Recommendations

Finance

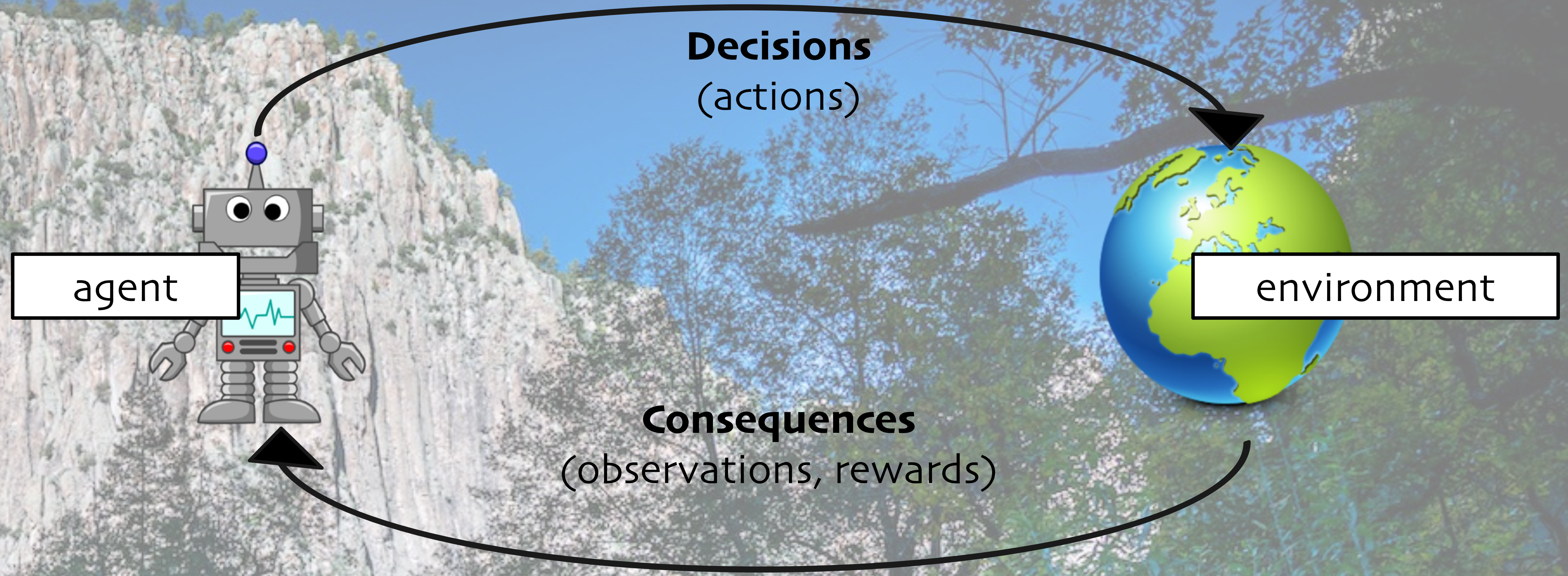
RL applications







Reinforcement Learning



Games

Robotics, Autonomous Vehicles

Industrial Processes

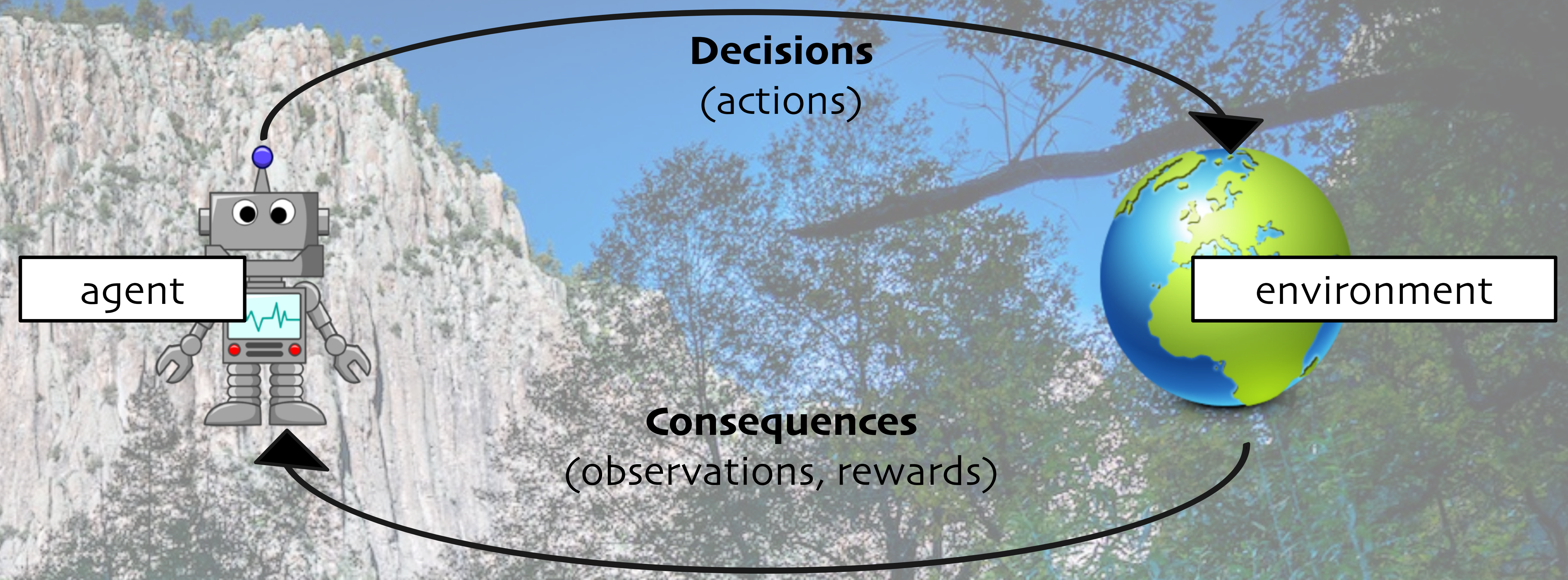
System Optimization

Advertising, Recommendations

Finance

RL applications

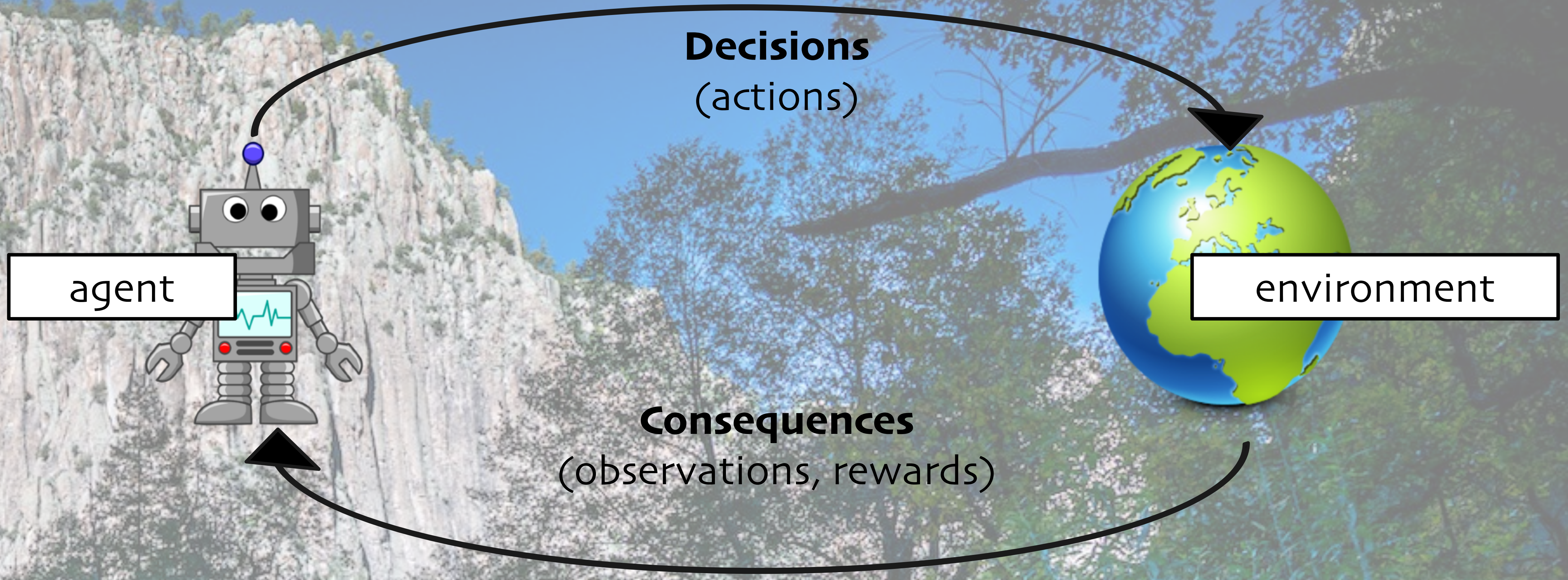
Reinforcement Learning



- Games
- Robotics, Autonomous Vehicles
- Industrial Processes
- System Optimization
- Advertising, Recommendations
- Finance

RL applications

Reinforcement Learning



Games

Robotics, Autonomous Vehicles

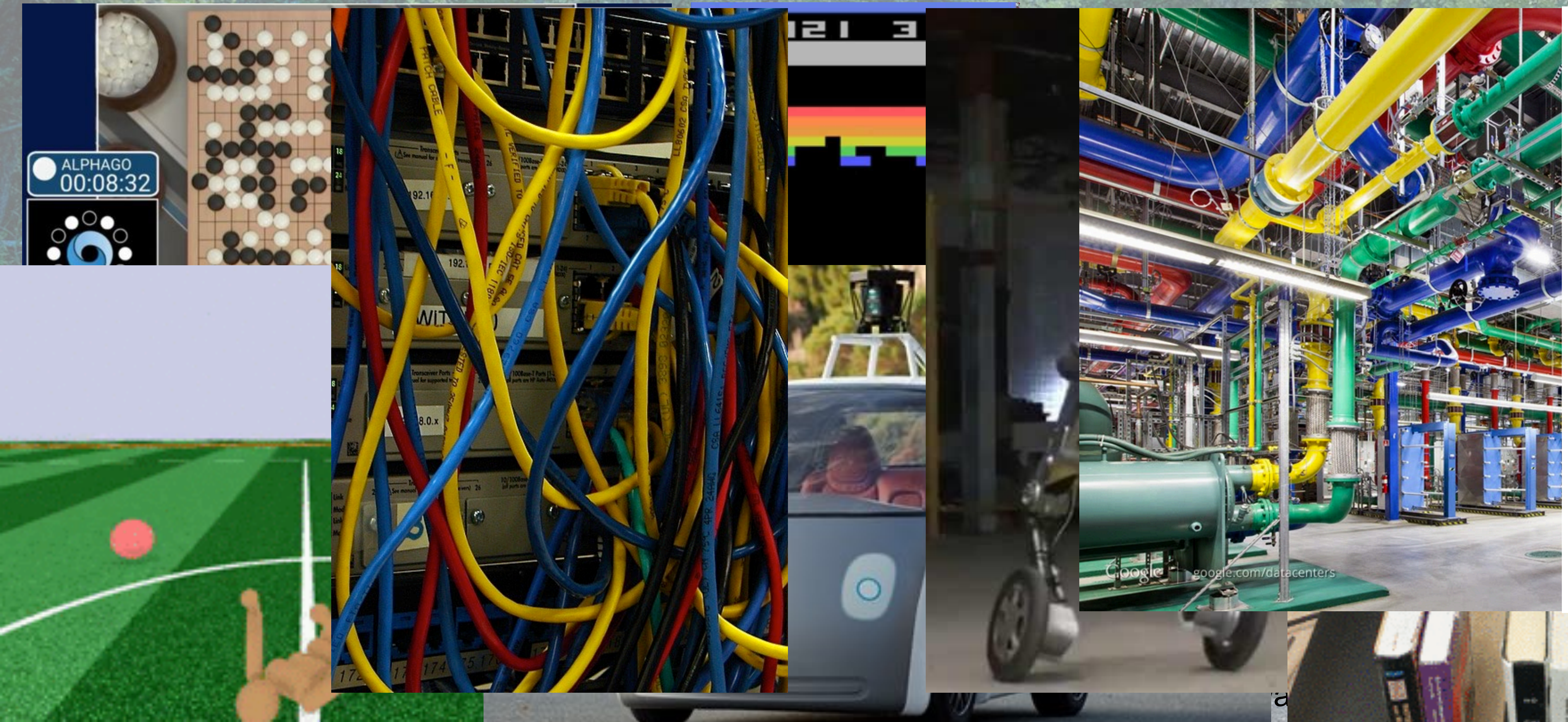
Industrial Processes

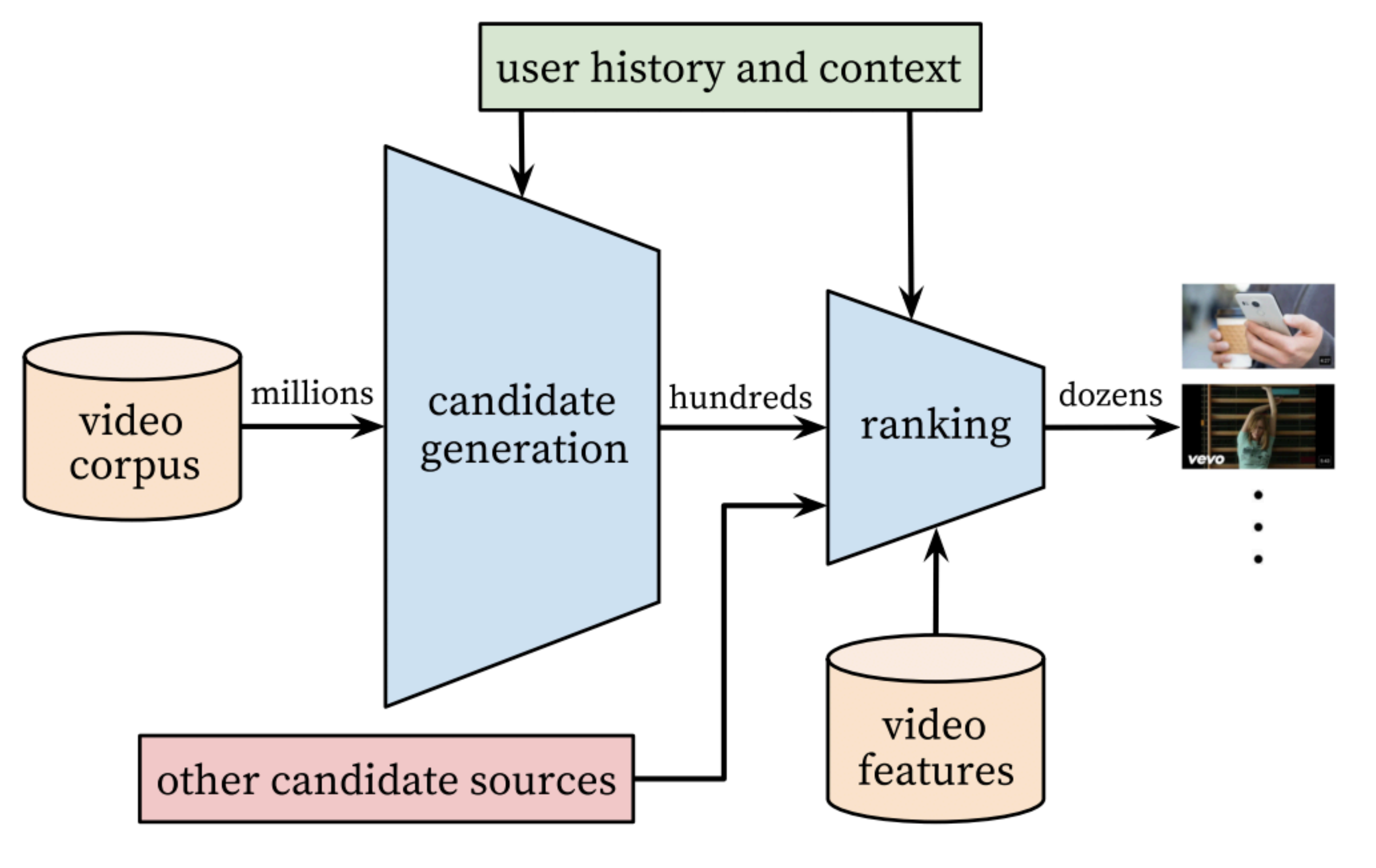
System Optimization

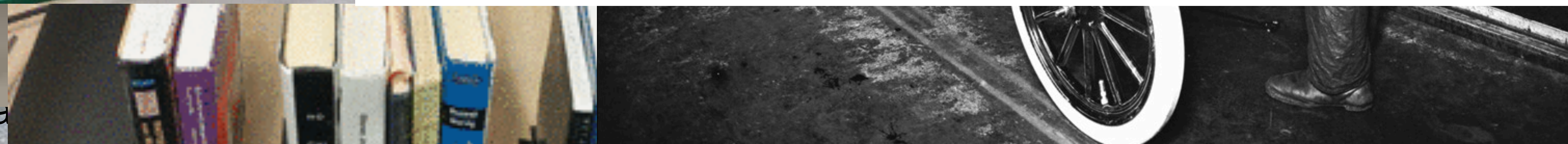
Advertising, Recommendations

Finance

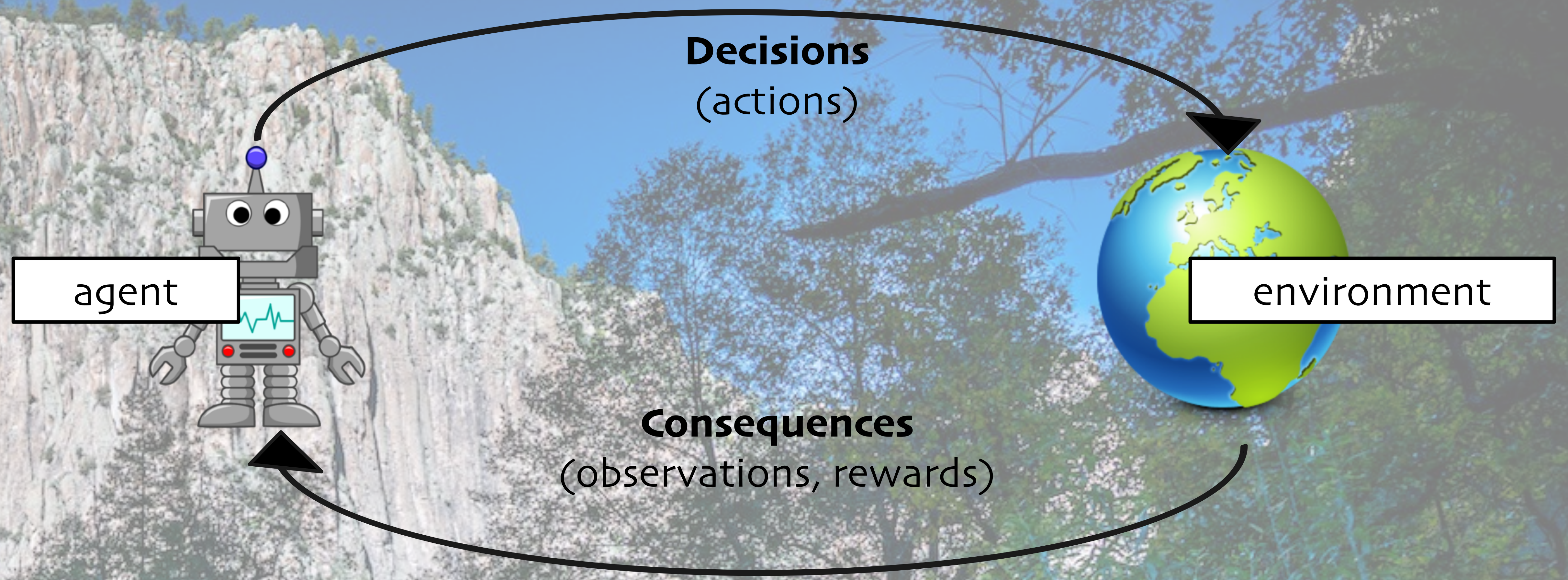
RL applications







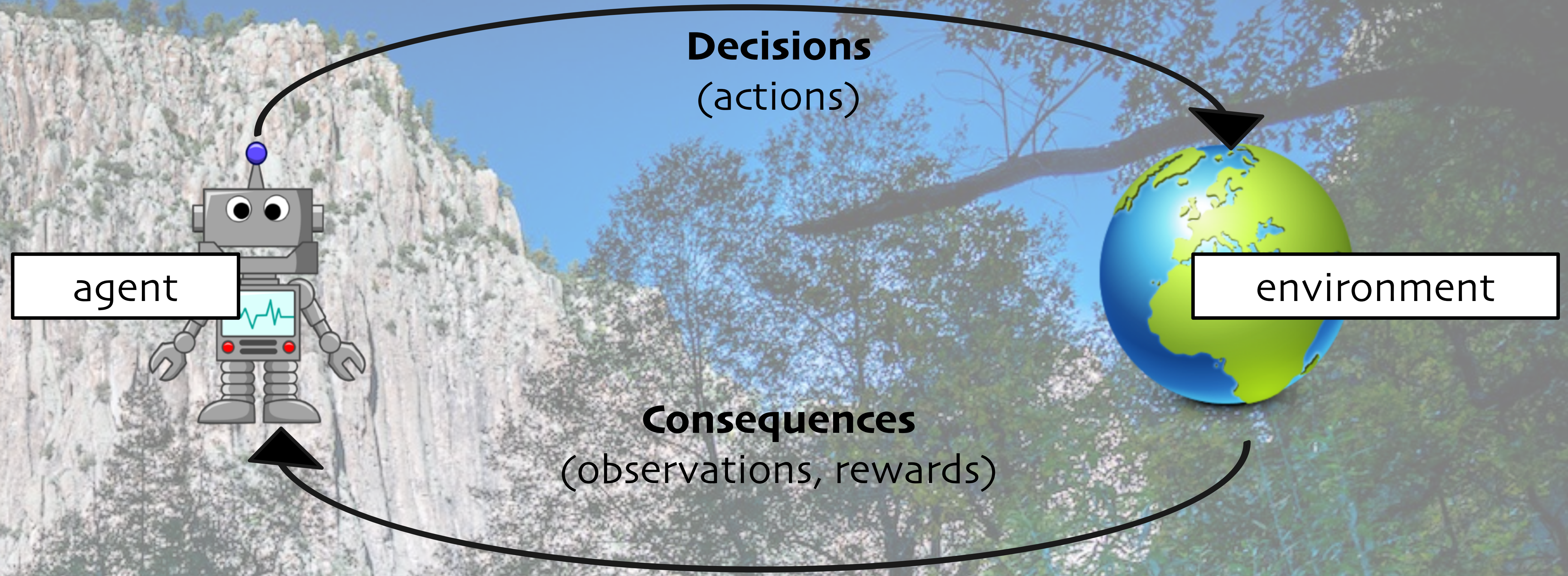
Reinforcement Learning



Games | Robotics, Autonomous Vehicles | Industrial Processes | System Optimization | Advertising, Recommendations | Finance

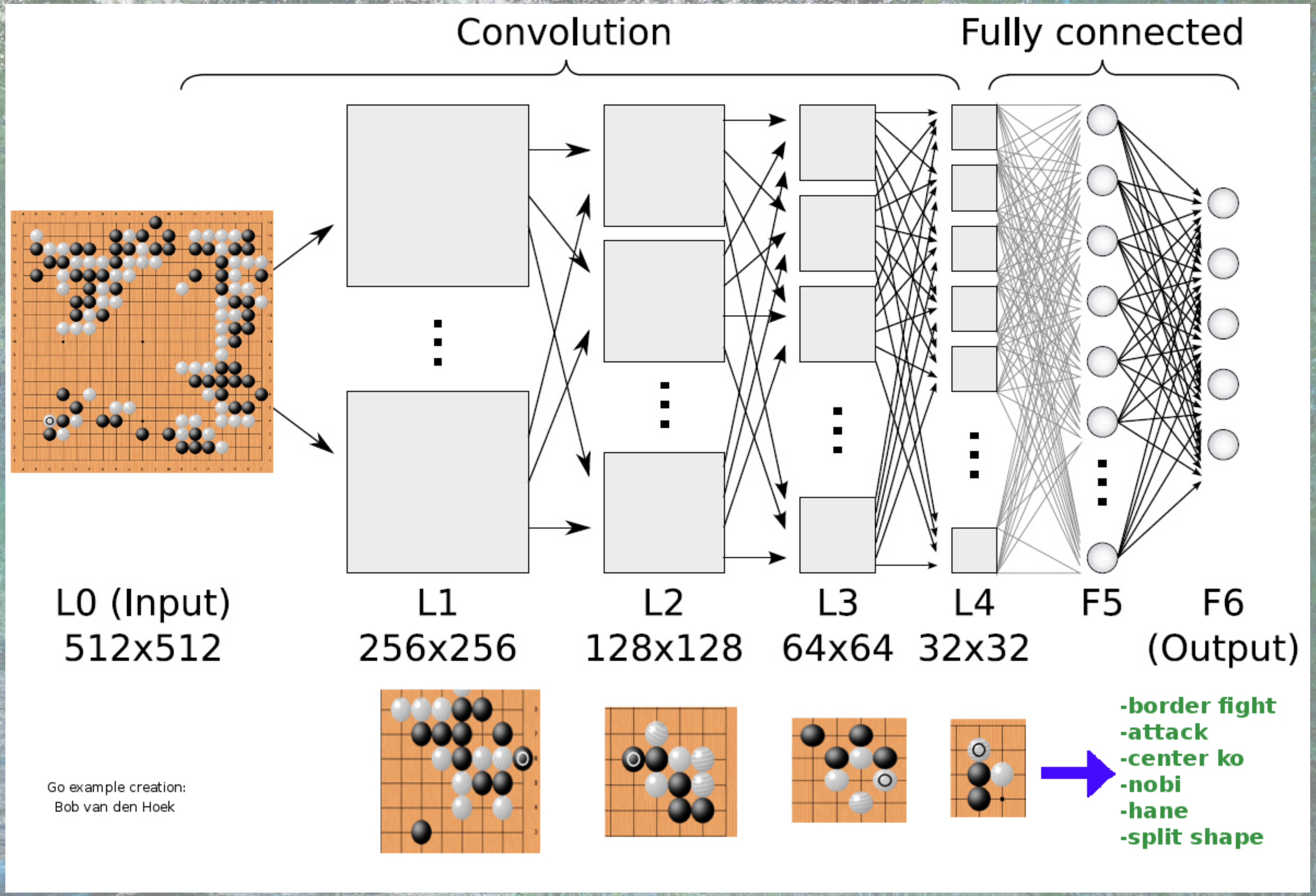
RL applications

Go as a Reinforcement Learning Problem



AlphaGo (Silver et al. 2016)

- **Observations**:
  - board state
- **Actions**:
  - where to place the stones
- **Rewards**:
  - 1 if win
  - 0 otherwise



@deanwampler

RLlib: A Scalable, Unified Library for RL

Games | Robotics, Autonomous Vehicles | Industrial Processes | System Optimization | Advertising, Recommendations | Finance

RL applications

OpenAI Gym | Multi-agent/ Hierarchical | Policy Serving | Offline Data

} (1) Application Support

Custom Algorithms | RLlib Algorithms

} (2) Abstractions for RL

RLlib Abstractions

Ray Tasks and Actors

} (3) Distributed Execution

A Broad Range of Popular Algorithms

- High-throughput architectures
  - Distributed Prioritized Experience Replay (Ape-X)
  - Importance Weighted Actor-Learner Architecture (IMPALA)
  - Asynchronous Proximal Policy Optimization (APPO)

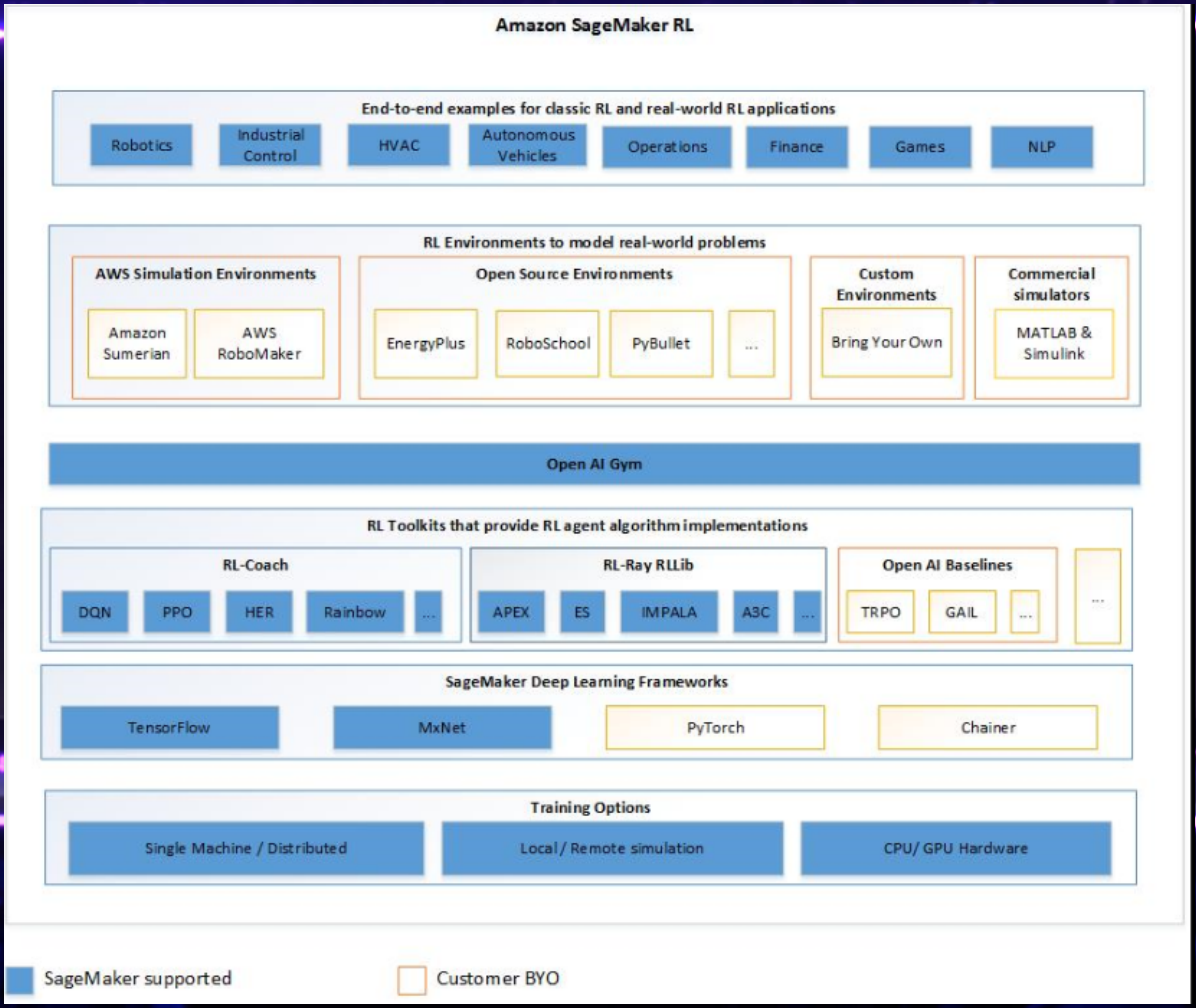
- Gradient-based
  - Soft Actor-Critic (SAC)
  - Advantage Actor-Critic (A2C, A3C)
  - Deep Deterministic Policy Gradients (DDPG, TD3)
  - Deep Q Networks (DQN, Rainbow, Parametric DQN)
  - Policy Gradients
  - Proximal Policy Optimization (PPO)

- gradient-free
  - Augmented Random Search (ARS)
  - Evolution Strategies

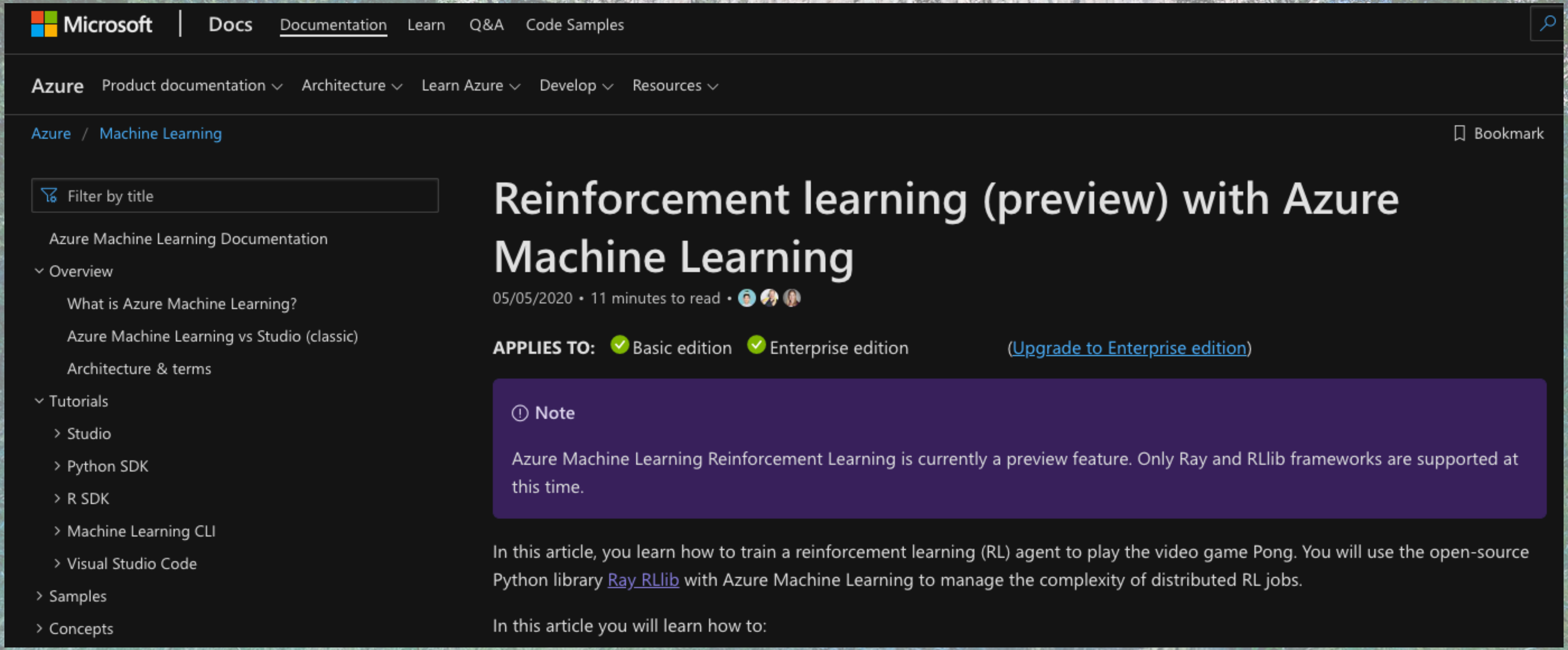
- Multi-agent specific
  - QMIX Monotonic Value Factorisation (QMIX, VDN, IQN)

- Offline
  - Advantage Re-Weighted Imitation Learning (MARWIL)


Amazon
SageMaker RL
Reinforcement learning for every


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

Now in Azure



Reinforcement learning (preview) with Azure Machine Learning

05/05/2020 • 11 minutes to read •

APPLIES TO: Basic edition Enterprise edition (Upgrade to Enterprise edition)

Note

Azure Machine Learning Reinforcement Learning is currently a preview feature. Only Ray and RLlib frameworks are supported at this time.

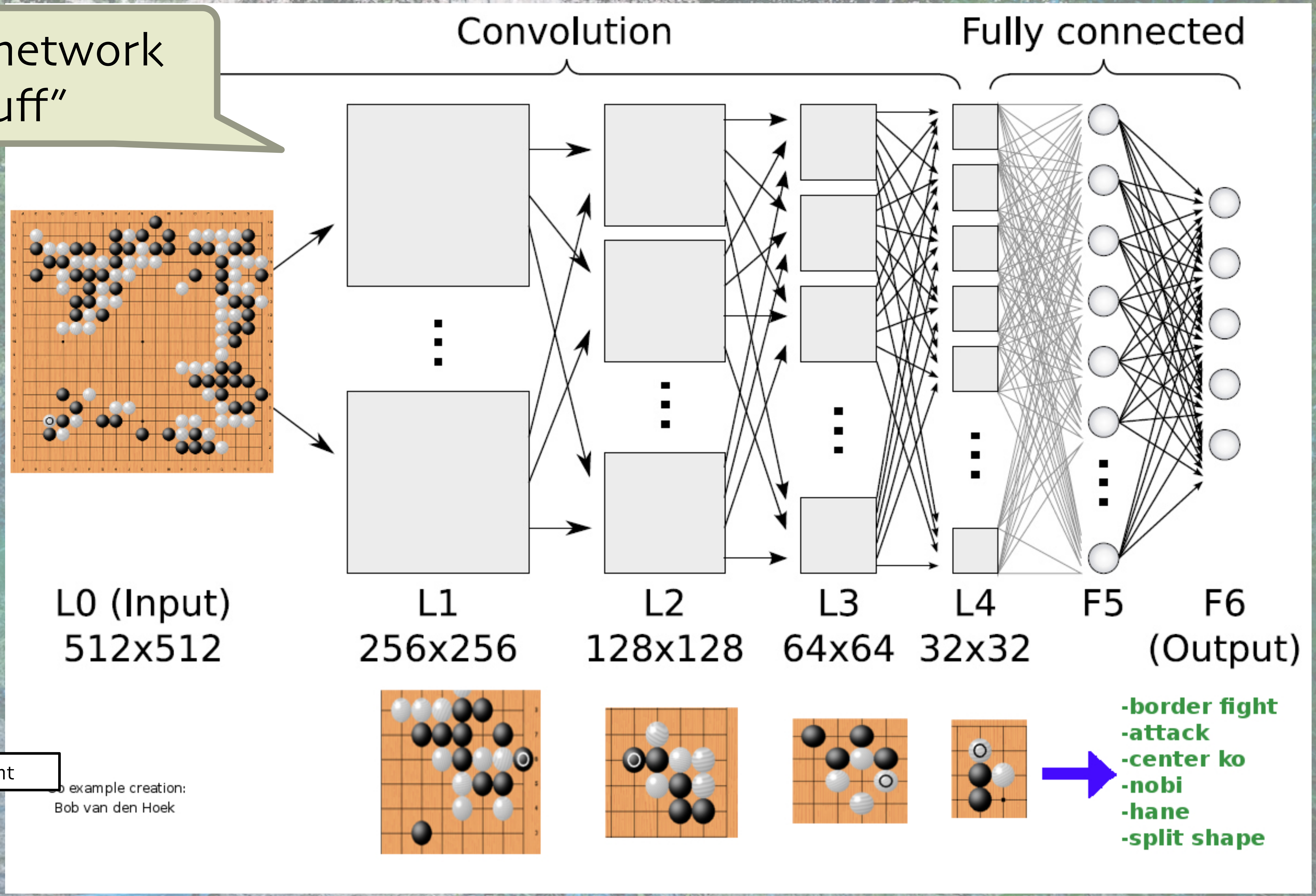
In this article, you learn how to train a reinforcement learning (RL) agent to play the video game Pong. You will use the open-source Python library Ray RLlib with Azure Machine Learning to manage the complexity of distributed RL jobs.

In this article you will learn how to:

Diverse Compute Requirements Motivated Creation of Ray!

And repeated play, over and over again, to train for achieving the best reward

Neural network "stuff"

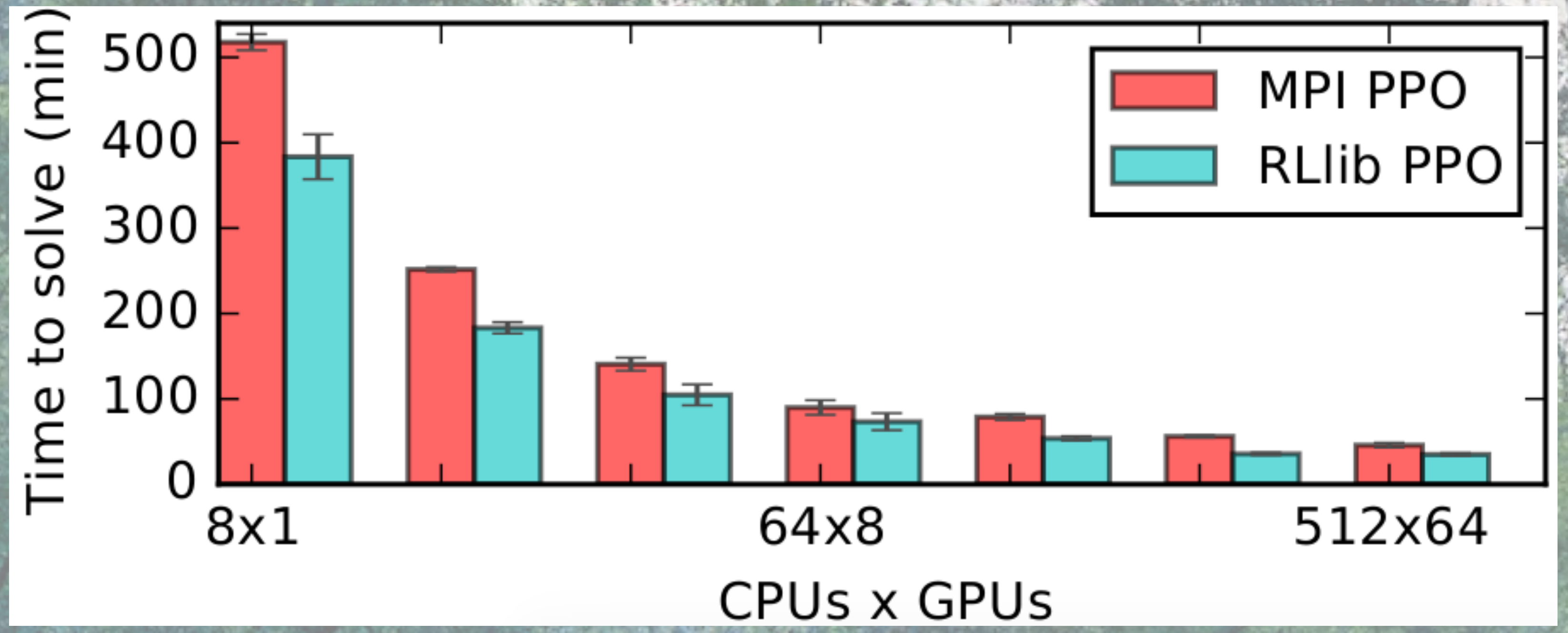


Simulator (game engine, robot sim, factory floor sim…)

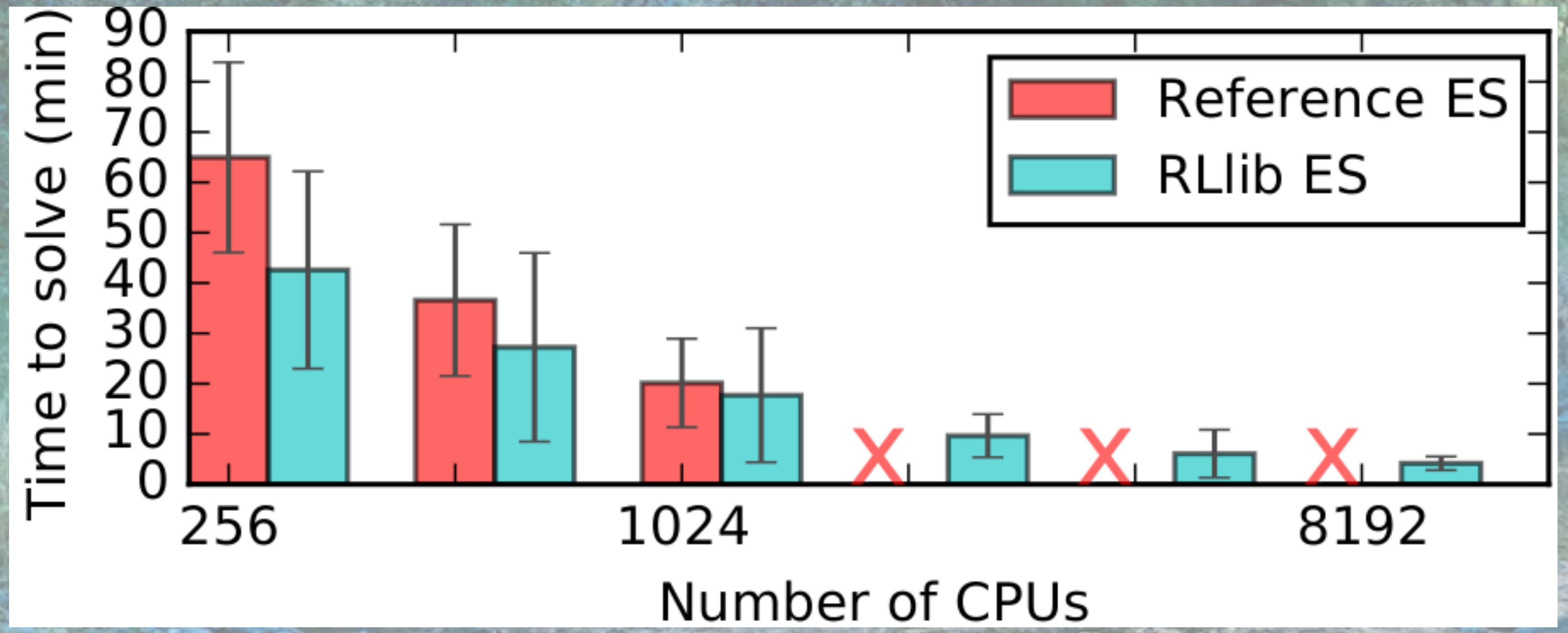
Complex agent?

RLlib Provides a Unified Framework for Scalable RL that Doesn't Compromise on Performance

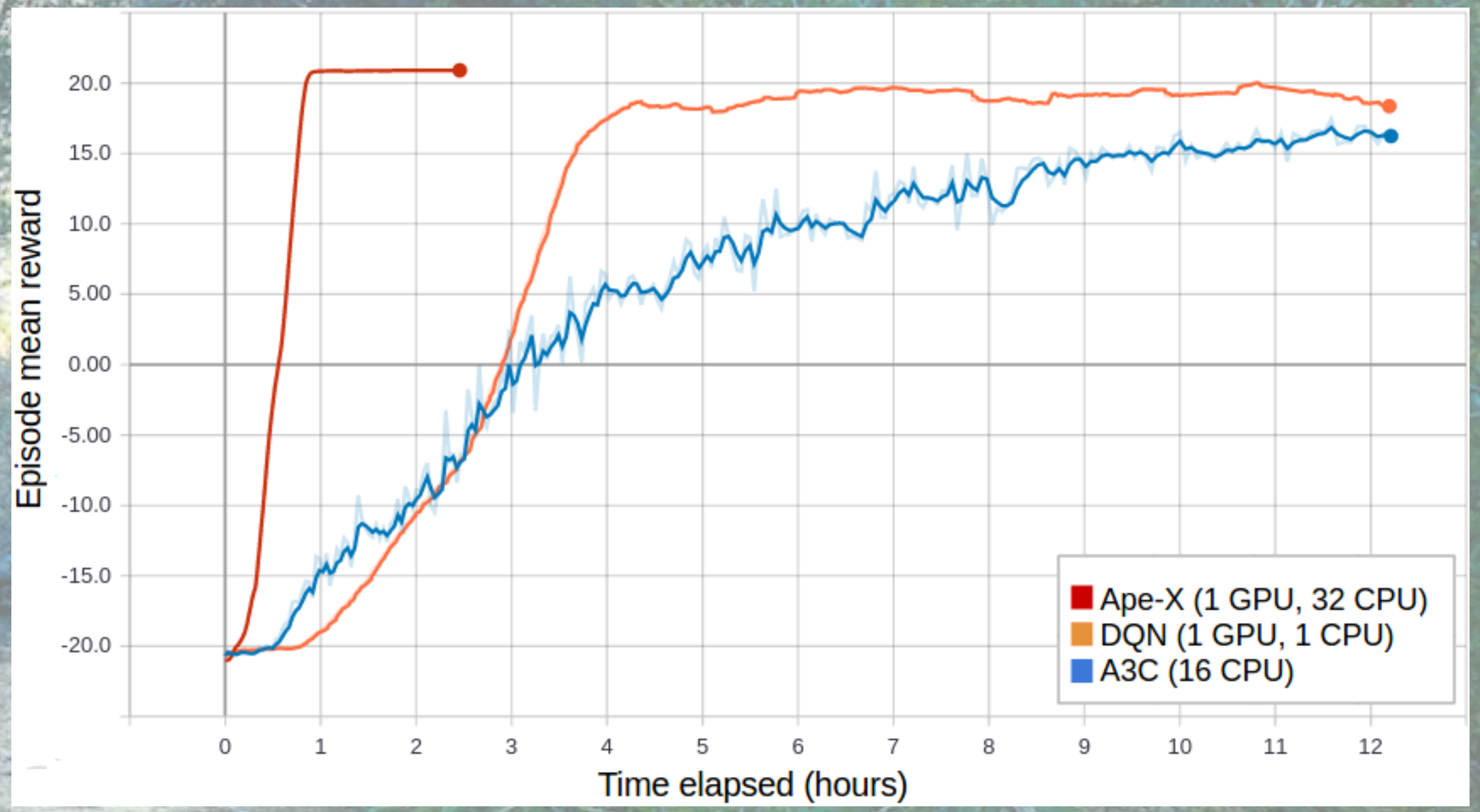
Distributed PPO



Evolution Strategies



Ape-X Distributed DQN, DDPG



@deanwampler


@deanwampler

Hyperparameter Tuning - Ray Tune

tune

raysgd

rllib

RAY SERVE

Featurization

Streaming

Hyperparam Tuning

Training

Simulation

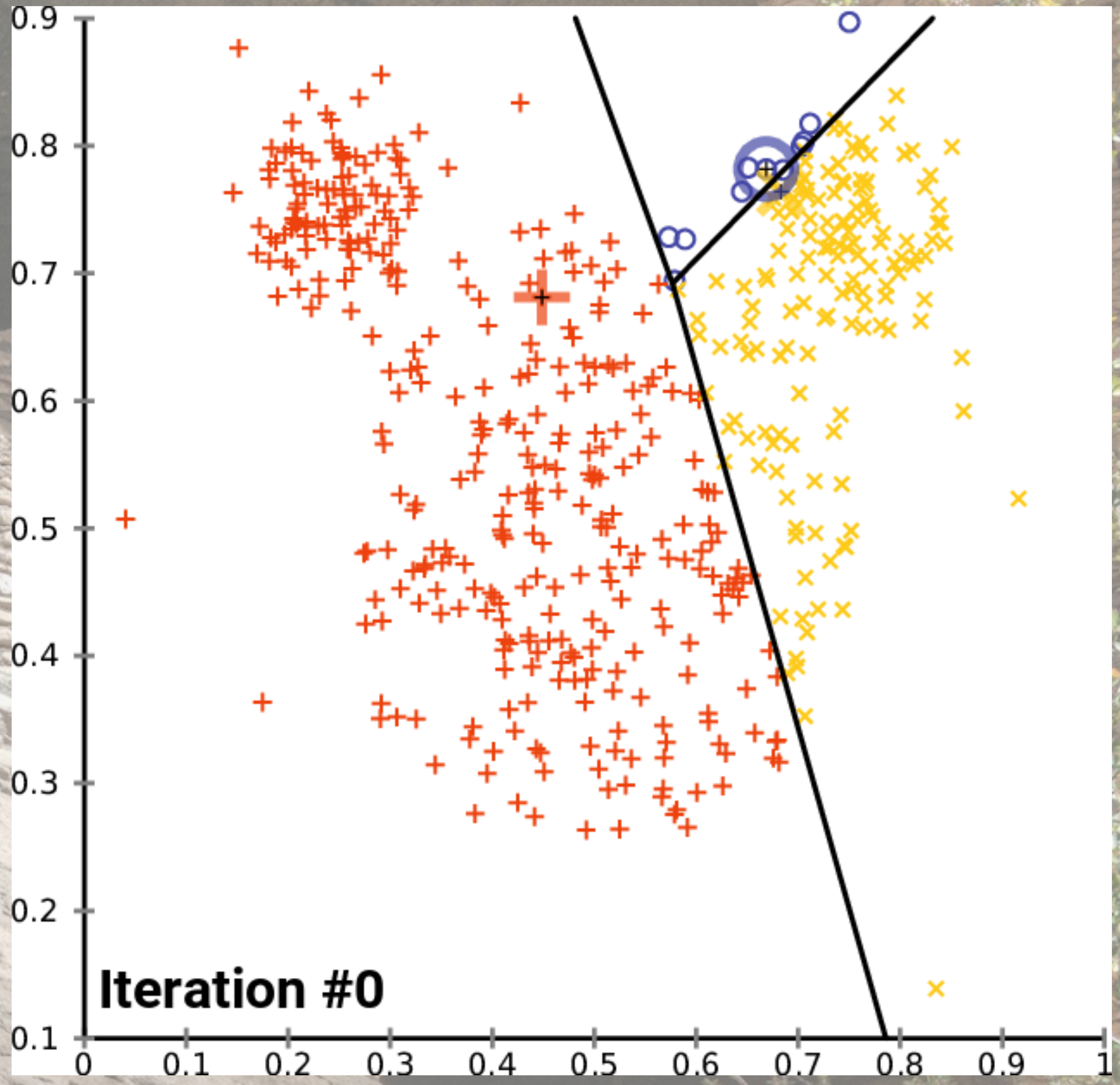
Model Serving

RAY

What Is Hyperparameter Tuning?

Trivial example:

- What's the best value for "k" in k-means??
  - k is a "hyperparameter"
    - The resulting clusters are defined by "parameters"

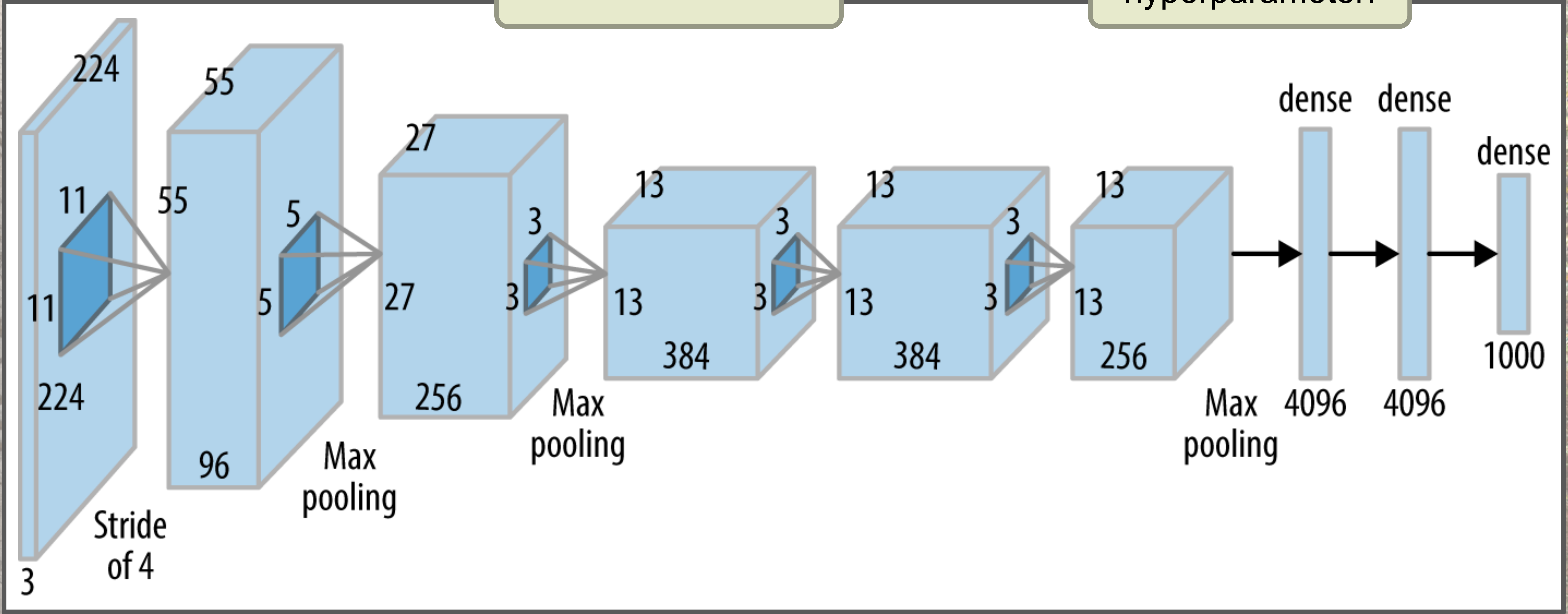


credit: https://commons.wikimedia.org/wiki/File:K-means_convergence.gif

Nontrivial Example - Neural Networks

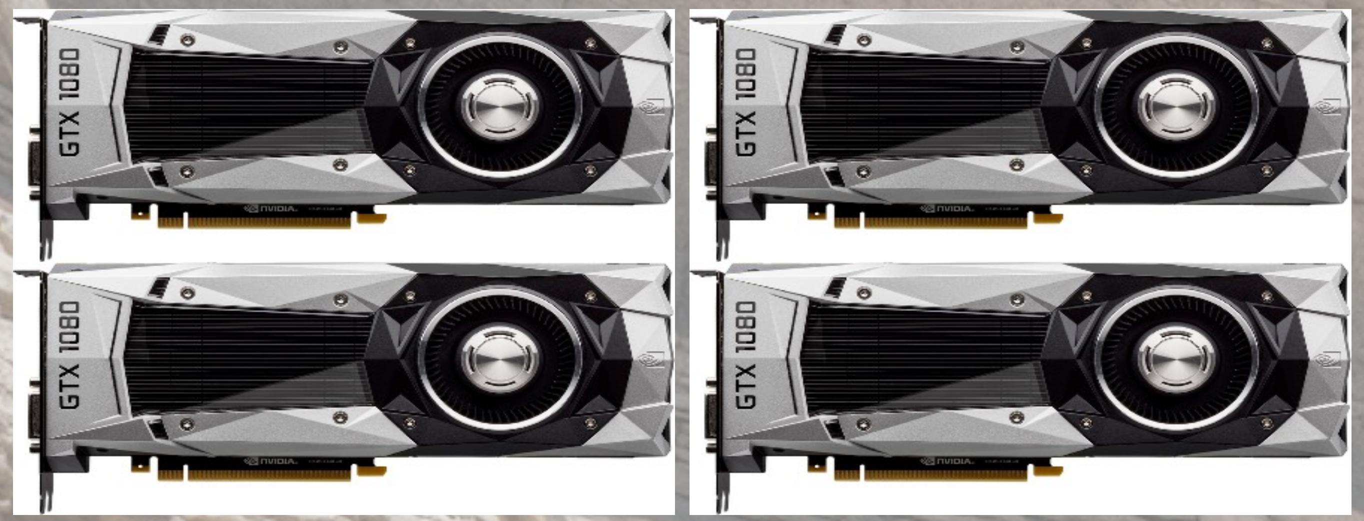
How many layers?
What kinds of layers?

Every number shown is a hyperparameter!

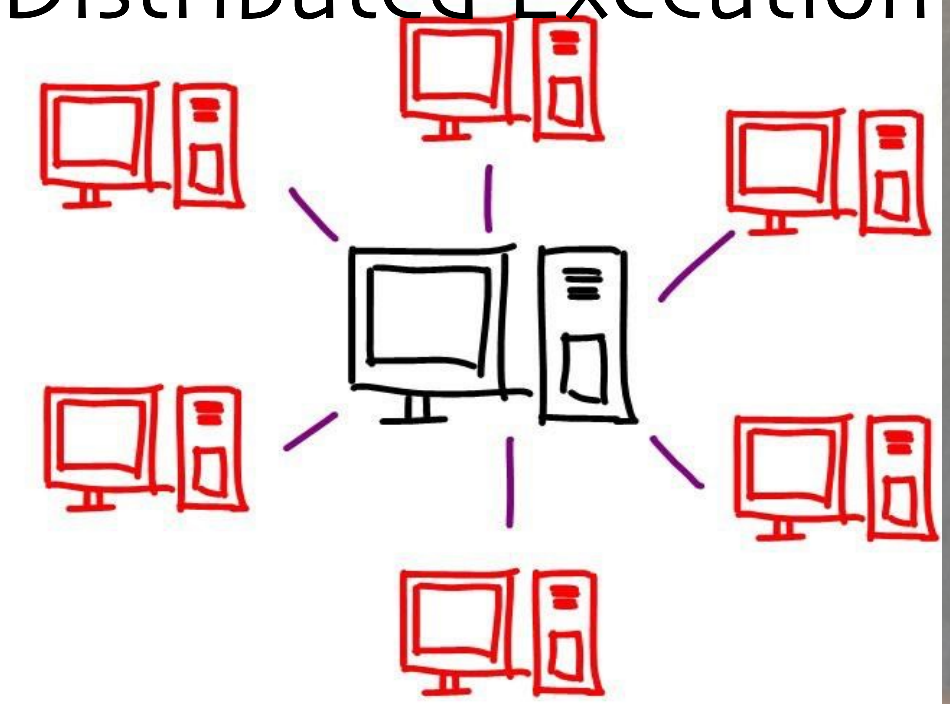


Tune is Built with Deep Learning as a Priority

Resource Aware Scheduling



Seamless Distributed Execution



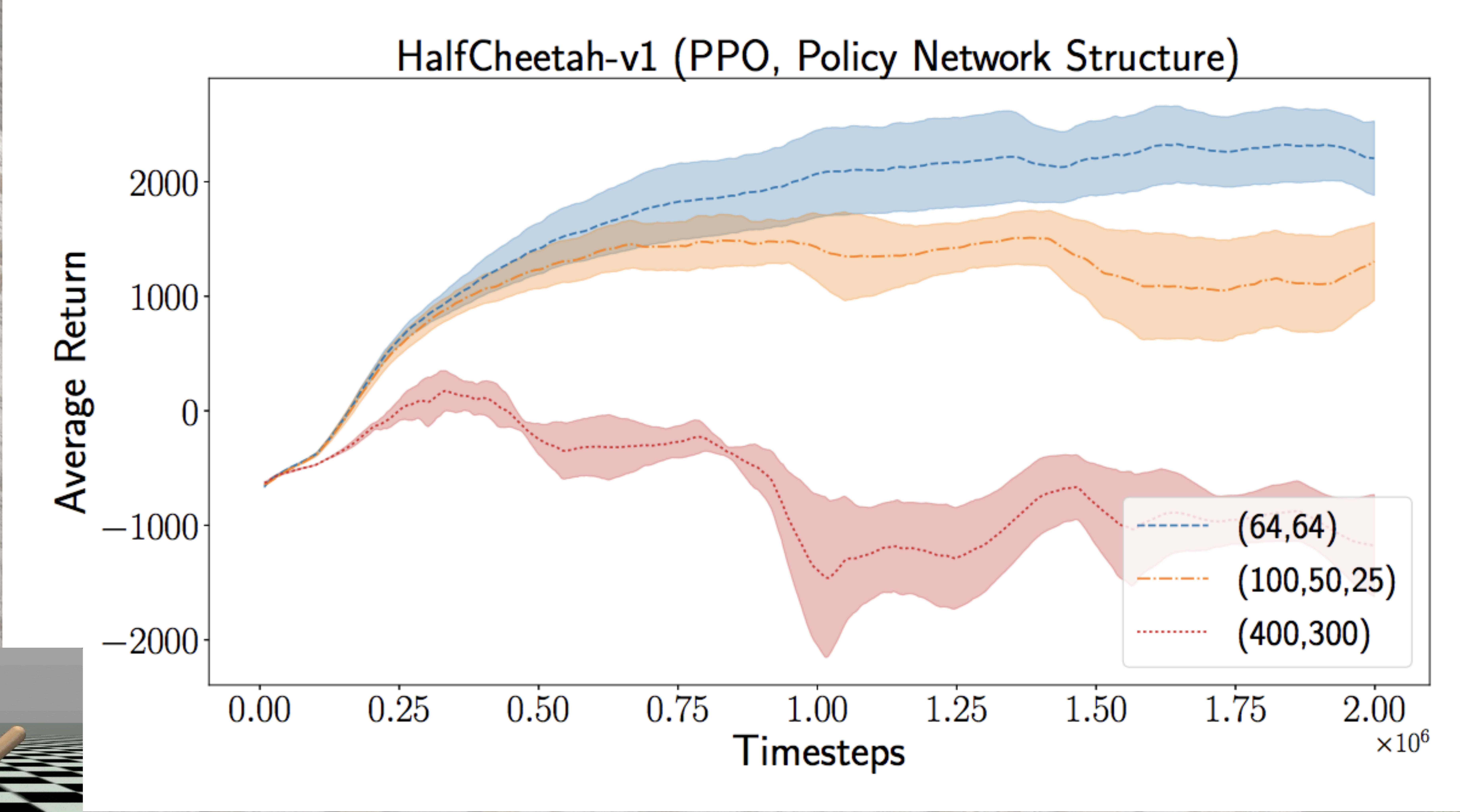
Simple API for new algorithms

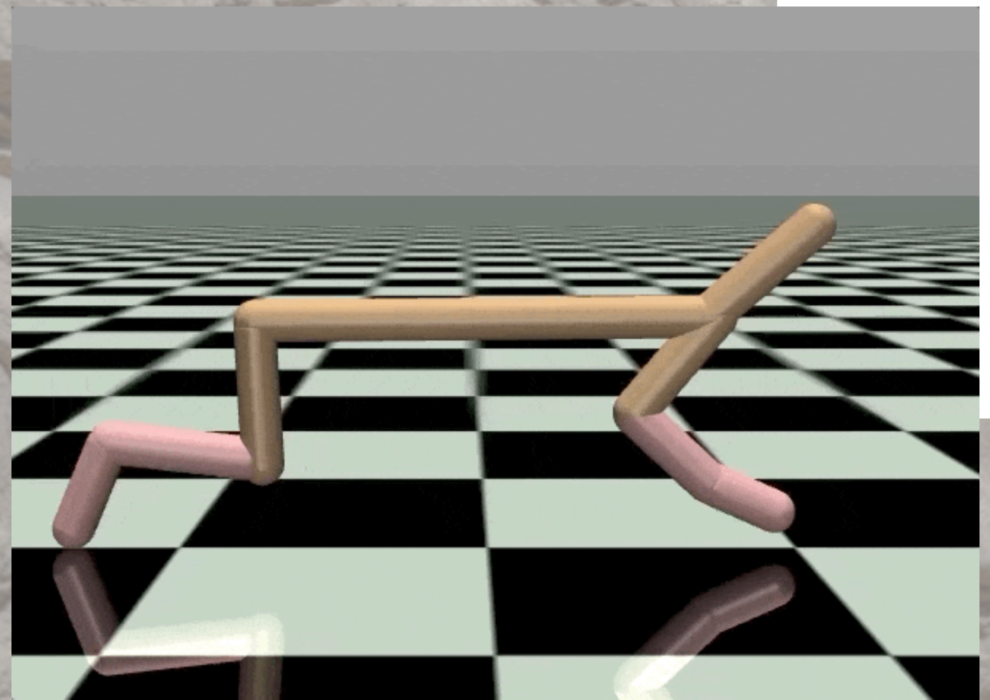
```
class TrialScheduler:
    def on_result(self, trial, result): ...
    def choose_trial_to_run(self): ...
```

Framework Agnostic

tune.io

Hyperparameters Are Important for Performance





Why We Need a Framework for Tuning Hyperparameters

We want the best model

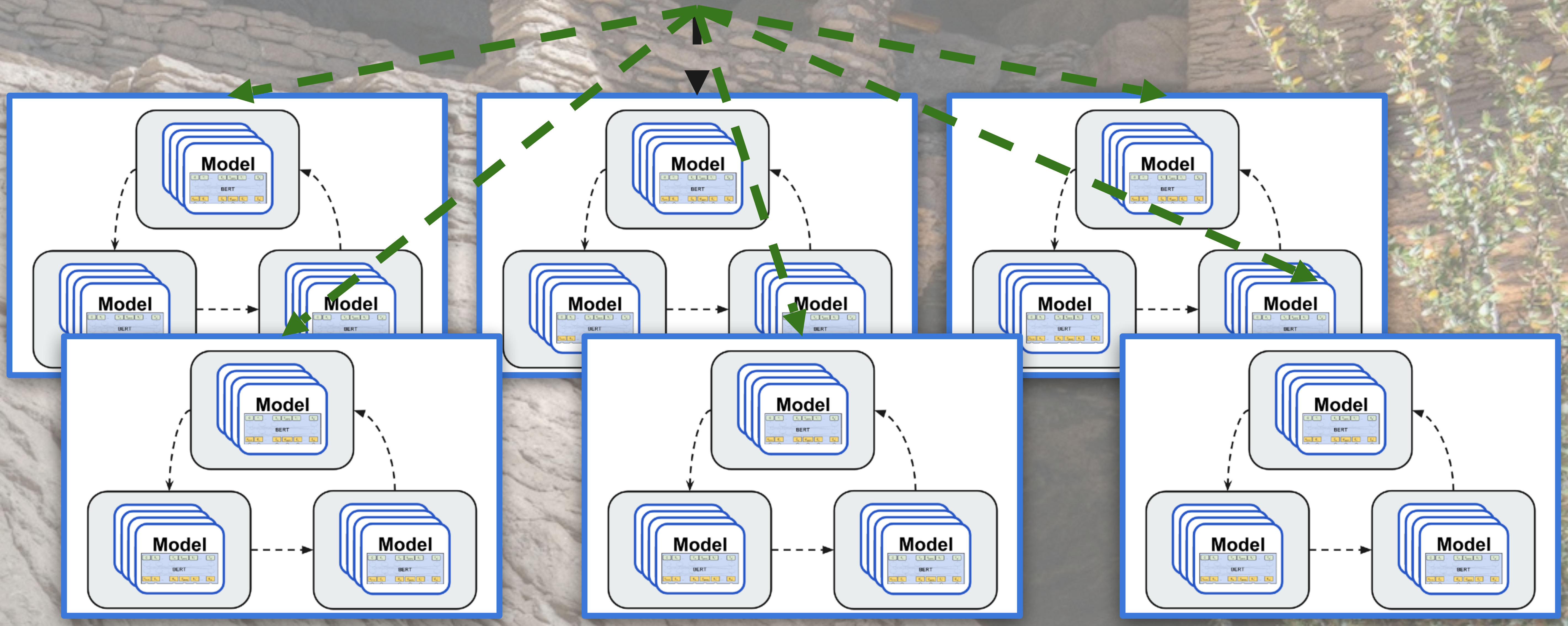
Resources are expensive

Model training is time-consuming

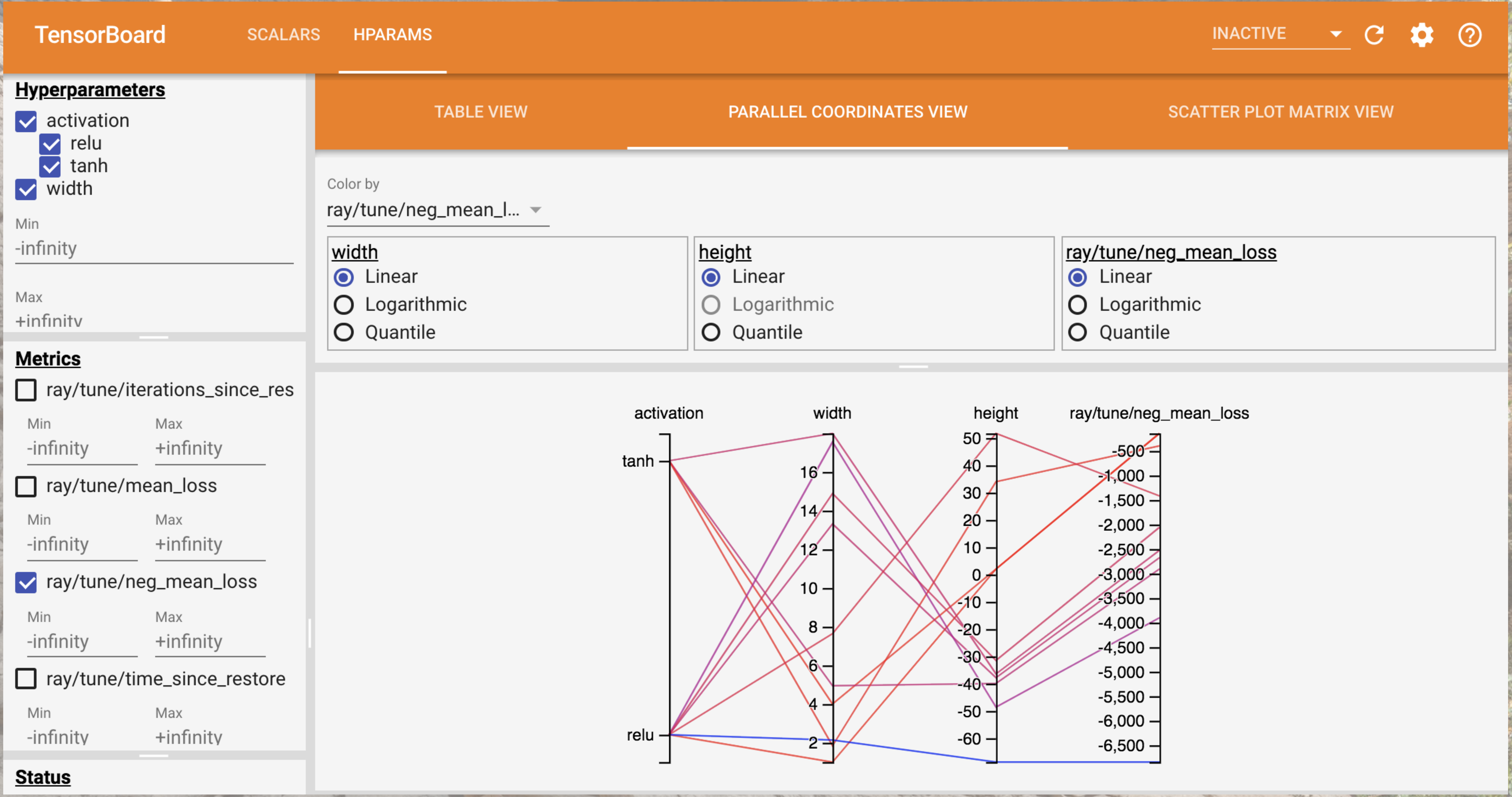
tune

Tuning + Distributed Training

```
tune.run(PytorchTrainable,
      config={
         "model_creator": PretrainBERT,
         "data_creator": create_data_loader,
         "use_gpu": True,
         "num_replicas": 8,
         "lr": tune.uniform(0.001, 0.1)
      },
      num_samples=100,
      search_alg=BayesianOptimization()
)
```



Native Integration with TensorBoard HParams



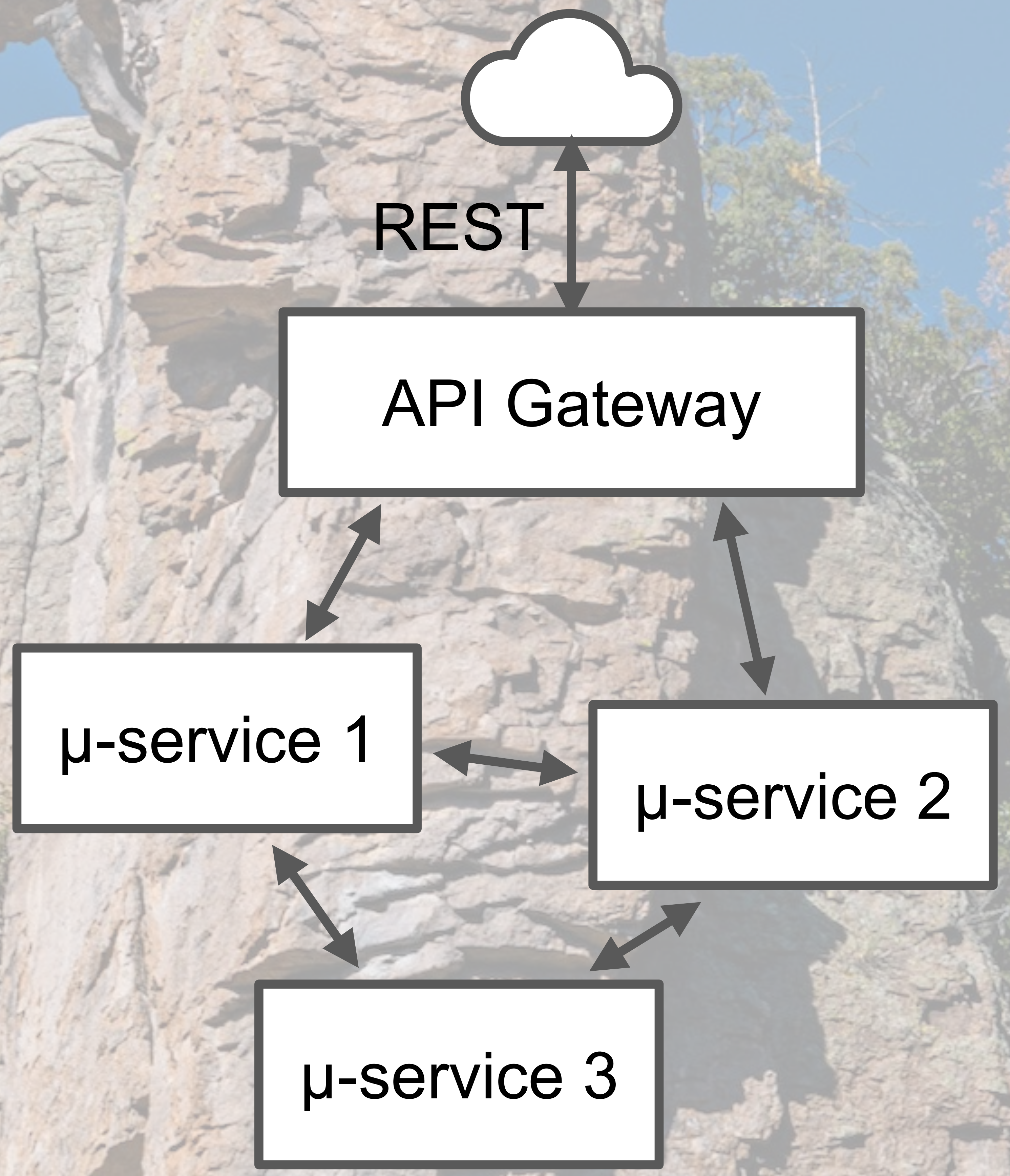
RAY

What about Ray for Microservices?


@deanwampler

What Are Microservices?

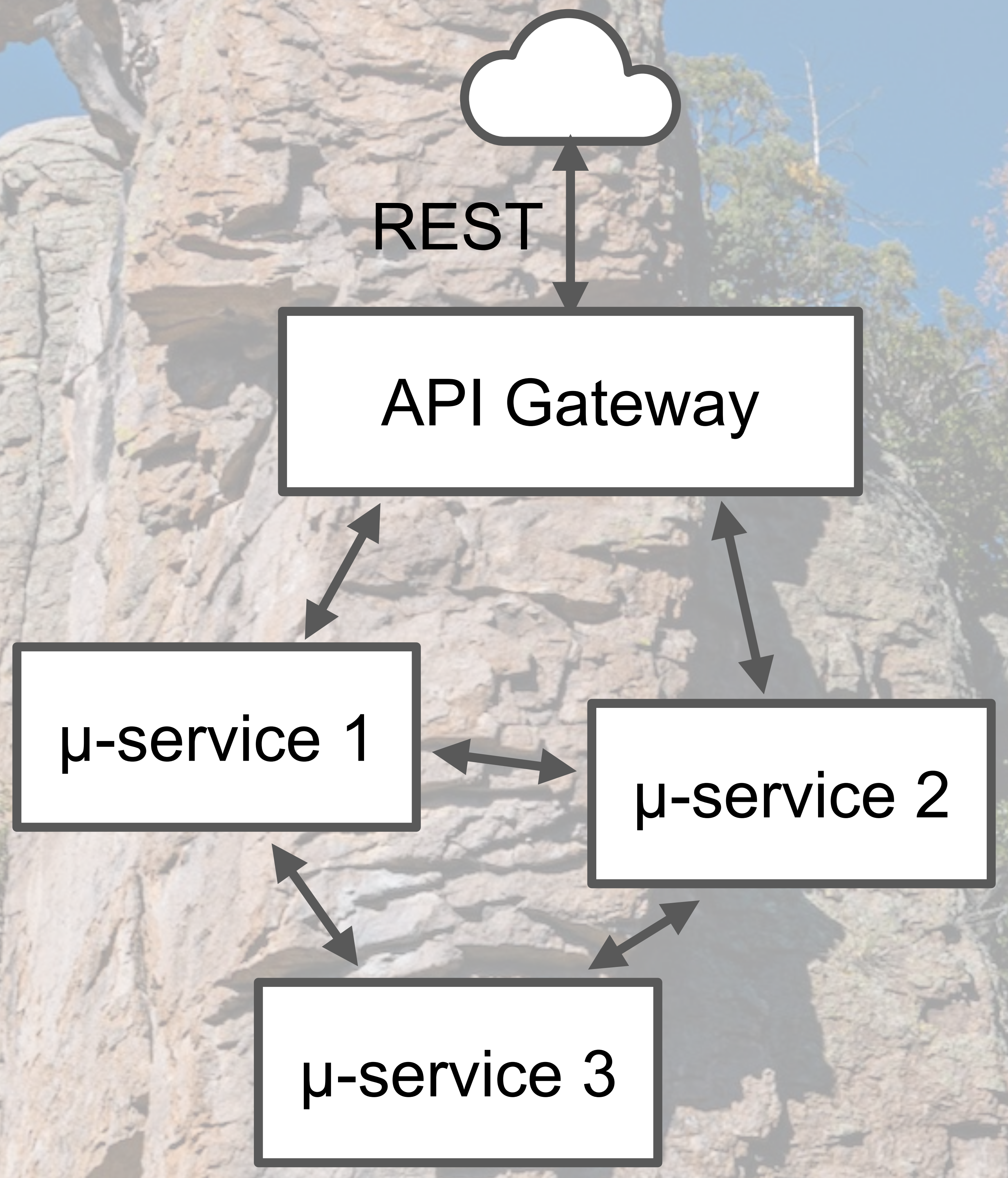
- They partition the domain
    - Conway's Law - Embraced
    - Separate responsibilities
- Separate management



What Are Microservices?

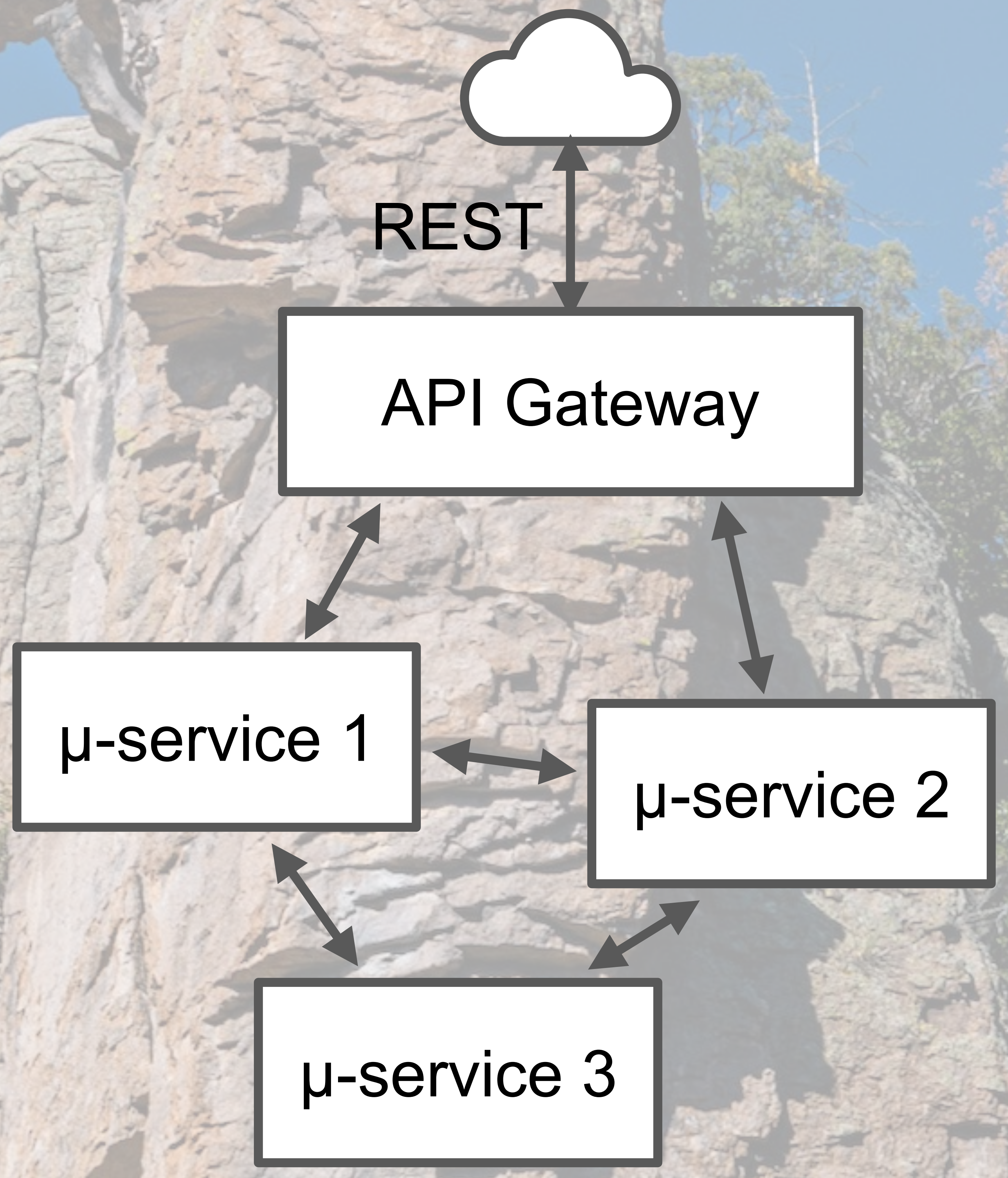
- They partition the domain
- Conway's Law - Embraced
- Separate responsibilities
- Separate management

What we mostly care about for today's talk, the "Ops in DevOps"



Conway's Law - Embraced

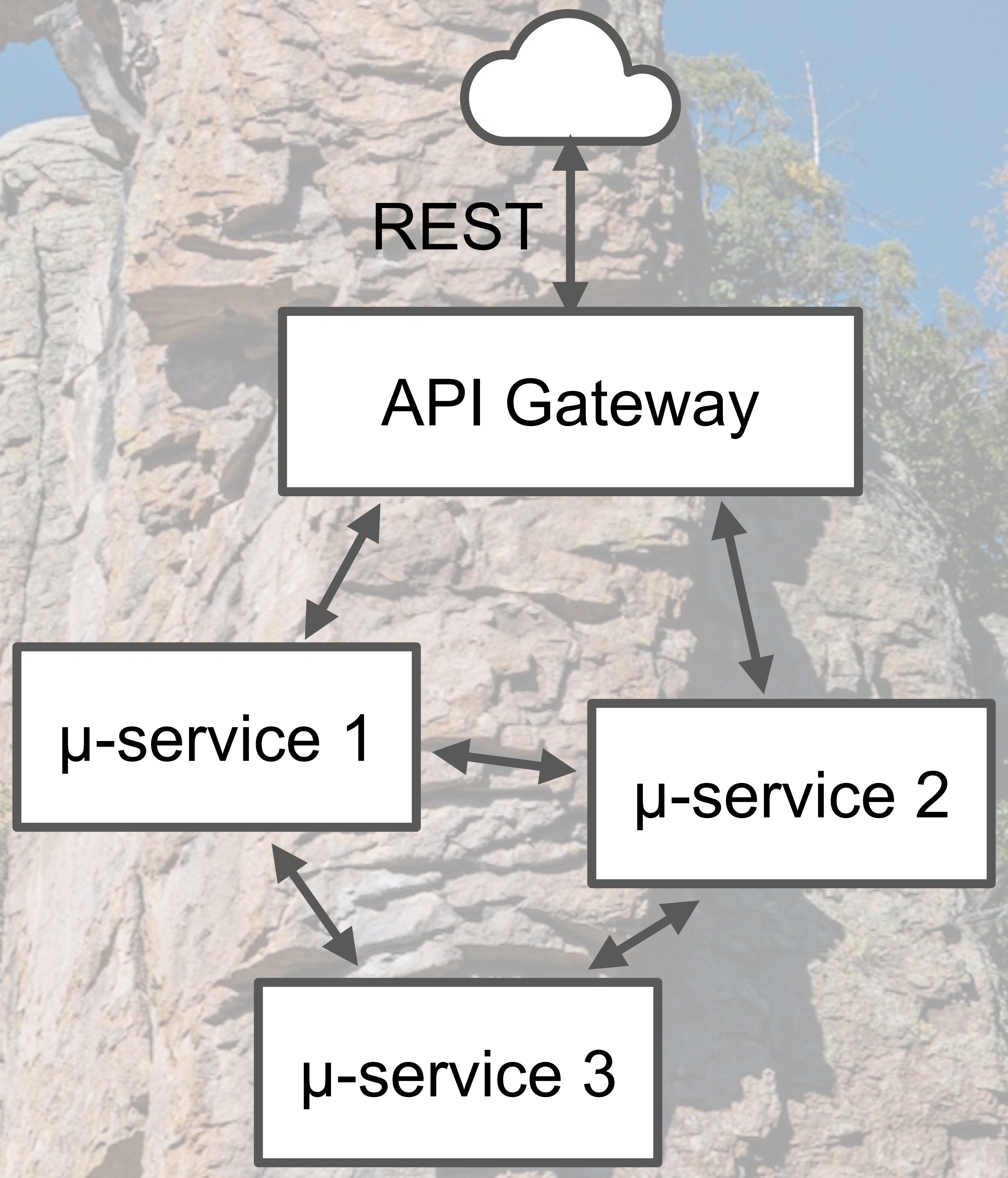
- "Any organization that designs a system will produce a design whose structure is a copy of the organization's communication structure"
- Let each team own and manage the services for its part of the domain



en.wikipedia.org/wiki/Conway's_law

Separate Responsibilities

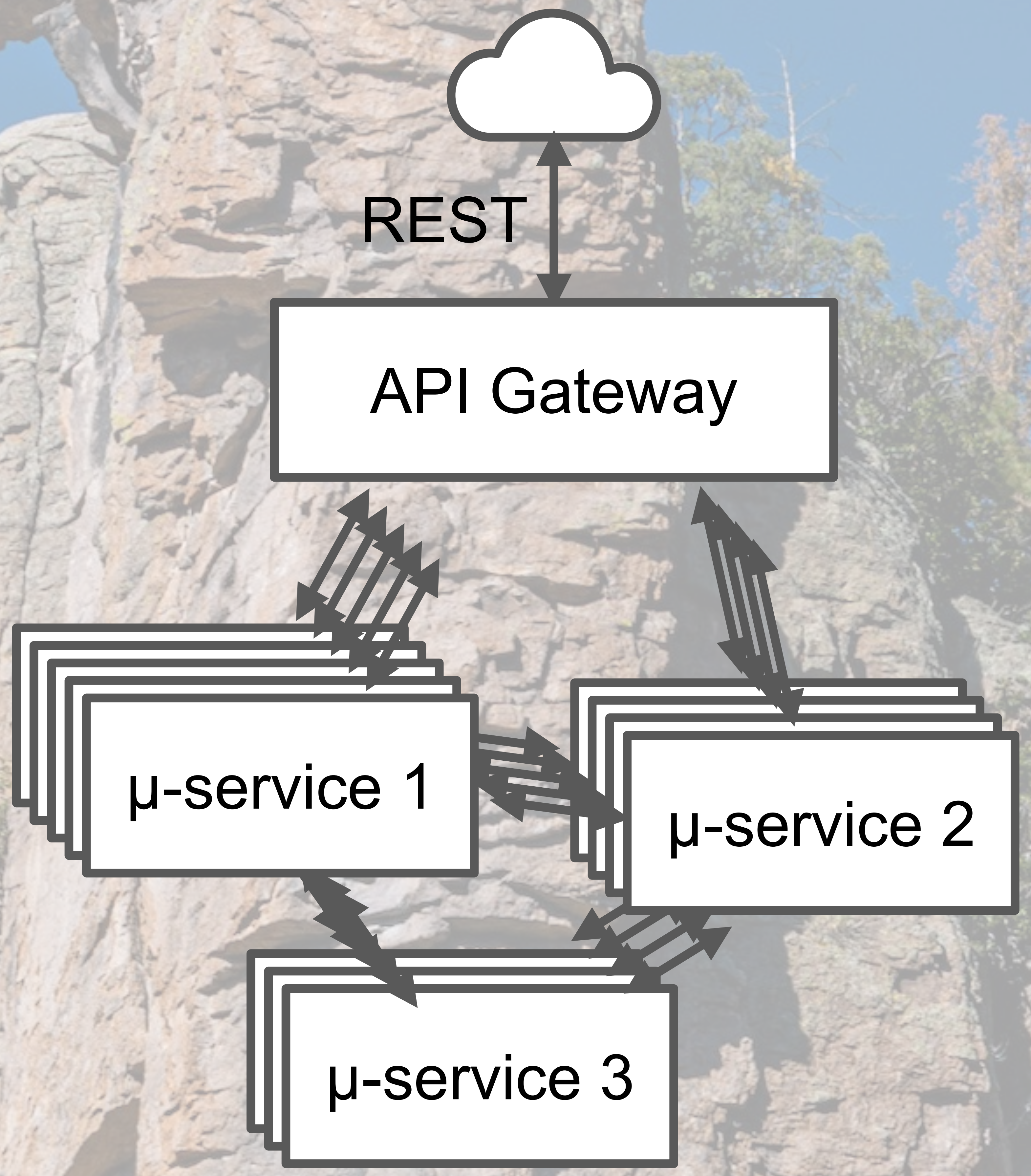
- Each microservice does “one thing”, a single responsibility with minimal coupling to the other microservices
- (Like, hopefully, the teams are organized, too…)



wikipedia.org/wiki/Single-responsibility_principle

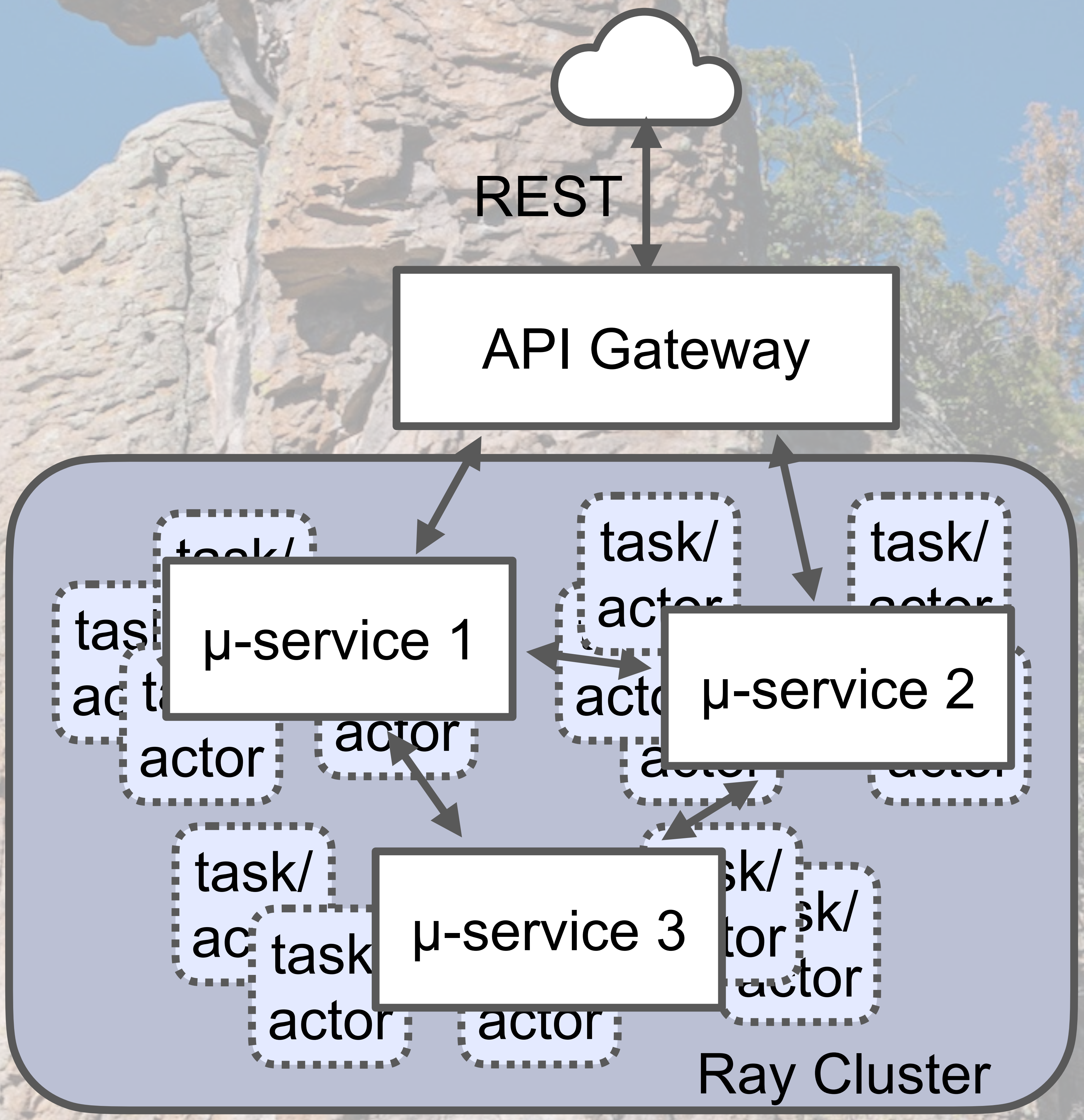
Separate Management

- Each team manages its own instances
- Each microservice has a different number of instances for scalability and resiliency
- But they have to be managed **explicitly**



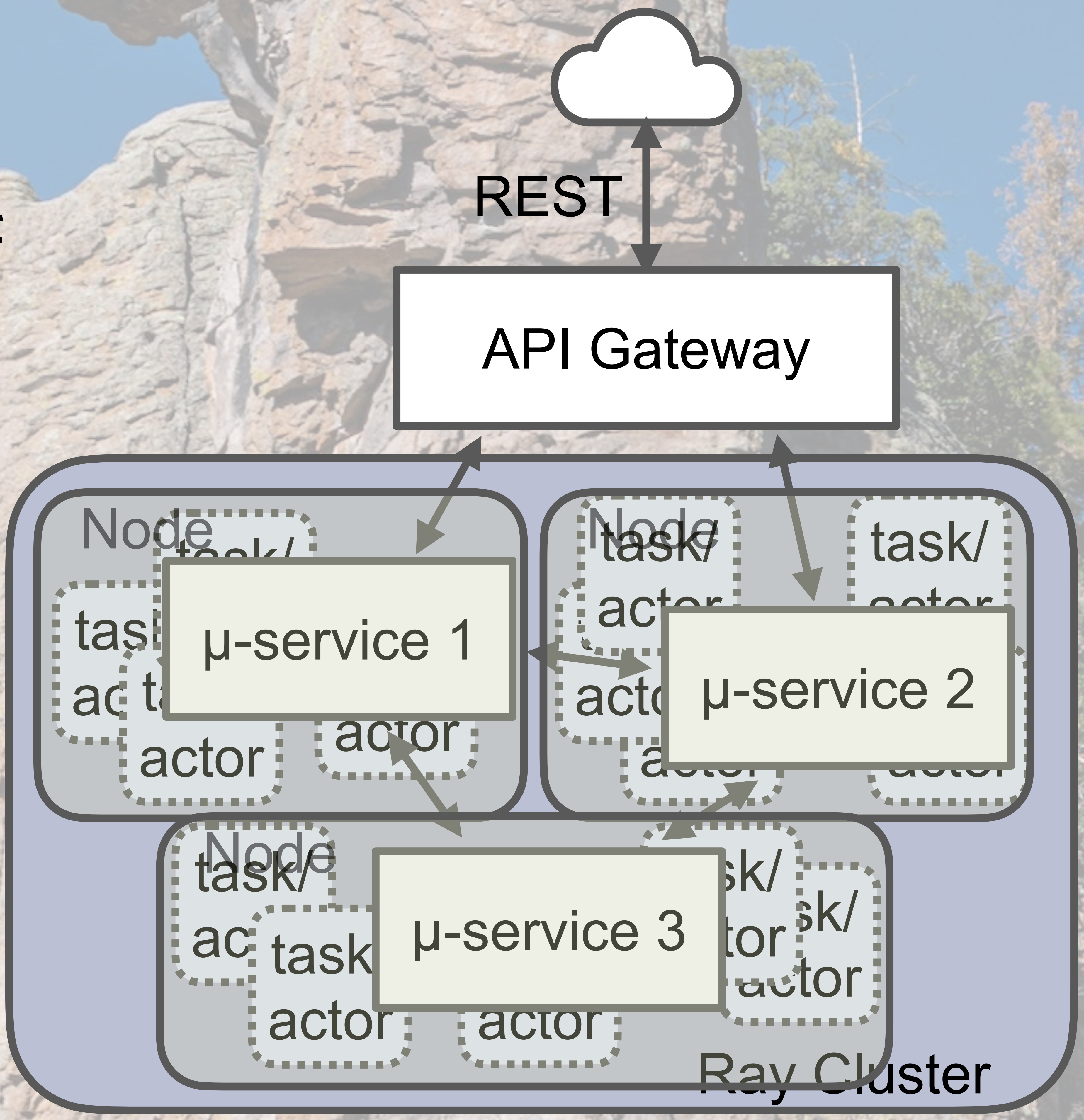
Management - Simplified

- With Ray, you have one "logical" instance to manage and Ray does the cluster-wide scaling for you.



What about Kubernetes (and others…)?

- Ray scaling is very fine grained.
- It operates within the “nodes” of coarse-grained managers
  - Containers, pods, VMs, or physical machines



RAY

Adopting Ray and the Ray community


@deanwampler

If you're already using...

- joblib
- multiprocessing.Pool

For example, from this:

```
from multiprocessing.pool import Pool
```

To this:

```
from ray.util.multiprocessing.pool import Pool
```

- Use Ray's implementations
  - Drop-in replacements
  - Change import statements
  - Break the one-node limitation!

- ... And Ray is integrated with asyncio

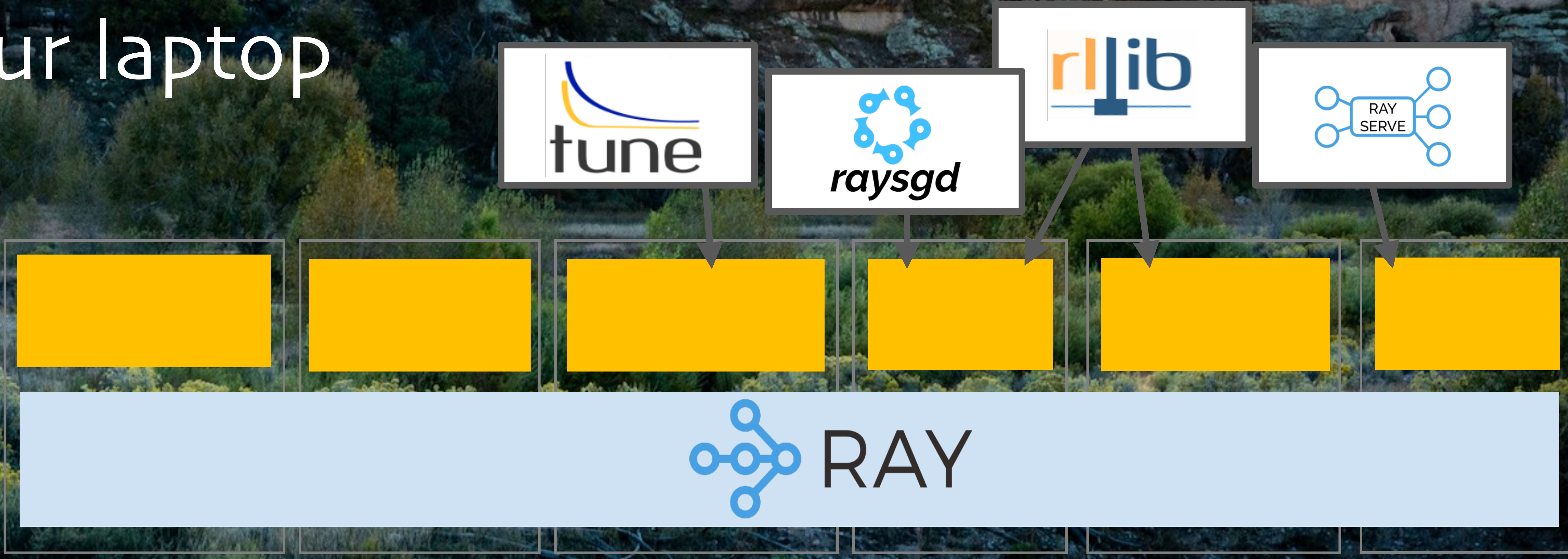
See these blog posts:
https://medium.com/distributed-computing-with-ray/how-to-scale-python-multiprocessing-to-a-cluster-with-one-line-of-code-d19f242f60ff
https://medium.com/distributed-computing-with-ray/easy-distributed-scikit-learn-training-with-ray-54ff8b643b33

Ray Community and Resources

- ray.io
- Tutorials (free): anyscale.com/academy
- Need help?
  - Ray Slack: ray-distributed.slack.com
  - ray-dev Google group

Conclusion

- Ray is the new state-of-the-art for distributed computing
  - The shortest path from your laptop to the cloud
  - Run complex distributed tasks on large clusters from simple code on your laptop



https://anyscale.com/events
https://anyscale.com/blog

RAY SUMMIT
CONNECT

Anyscale Academy

RAY

anyscale

RAY SUMMIT

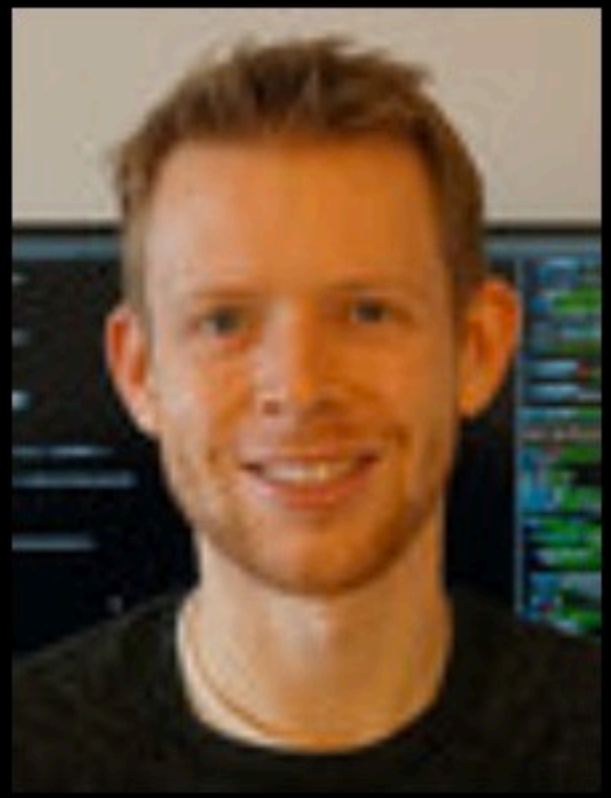
Presented by Anyscale

Sneak Peak!!

Sept 30 - Oct 1, 2020

raysummit.org

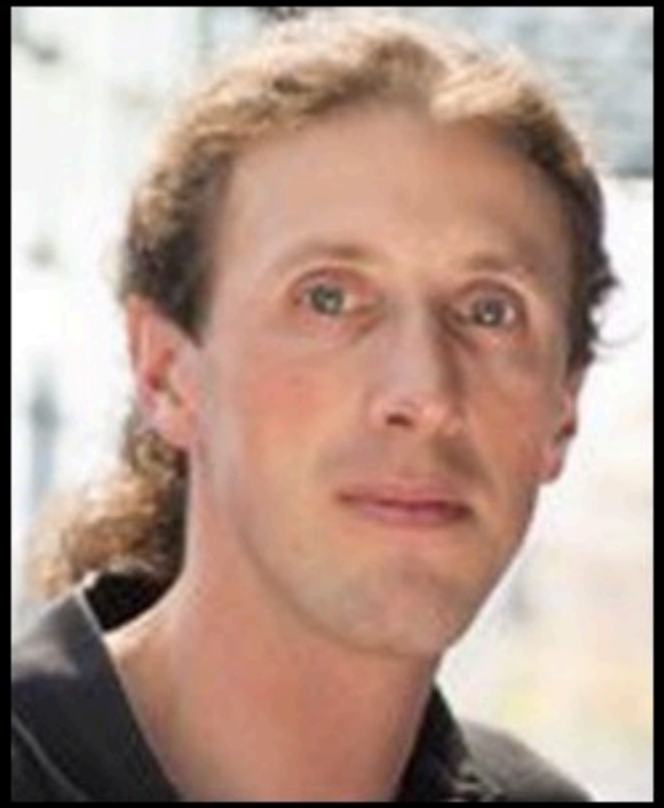
Michael Jordan
U.C. Berkeley



Wes McKinney
Ursa Labs

Azalia Mirhoseini
Google

Ion Stoica
U.C. Berkeley



Gaël Varoquaux
Inria

Manuela Veloso
J. P. Morgan AI Research

RAY

anyscale

ray.io

anyscale.com

dean@anyscale.com

@deanwampler

https://anyscale.com/events
https://anyscale.com/blog