Ray - Scalability from a Laptop to a Cluster

Dean Wampler - EuroPython 2020
dean@anyscale.com
@deanwampler
ray.io
anyscale.com

What We’ll Talk About:

- Ray demo
  - We’ll get into the mechanics of using the Ray API
- Why Ray Is Needed
- ML/AI Ray Libraries
- Ray for Microservices
- Adopting Ray and the Ray community

RAY

Demo

- Adapted from forthcoming tutorials:
  - github.com/anyscale/academy
- Contact Dean for details:
  - dean@anyscale.com

RAY

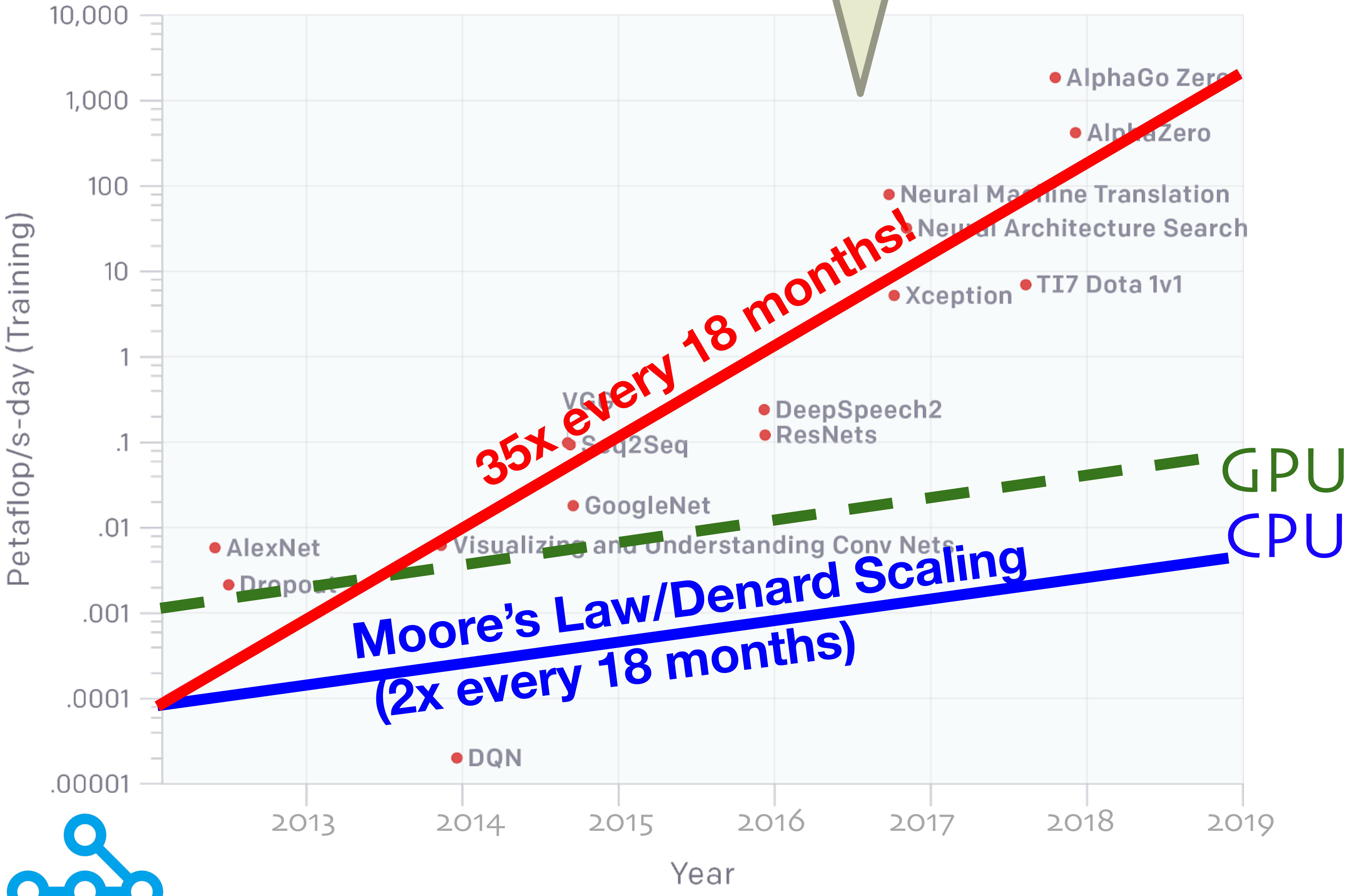
Why Ray?

Two Major Trends

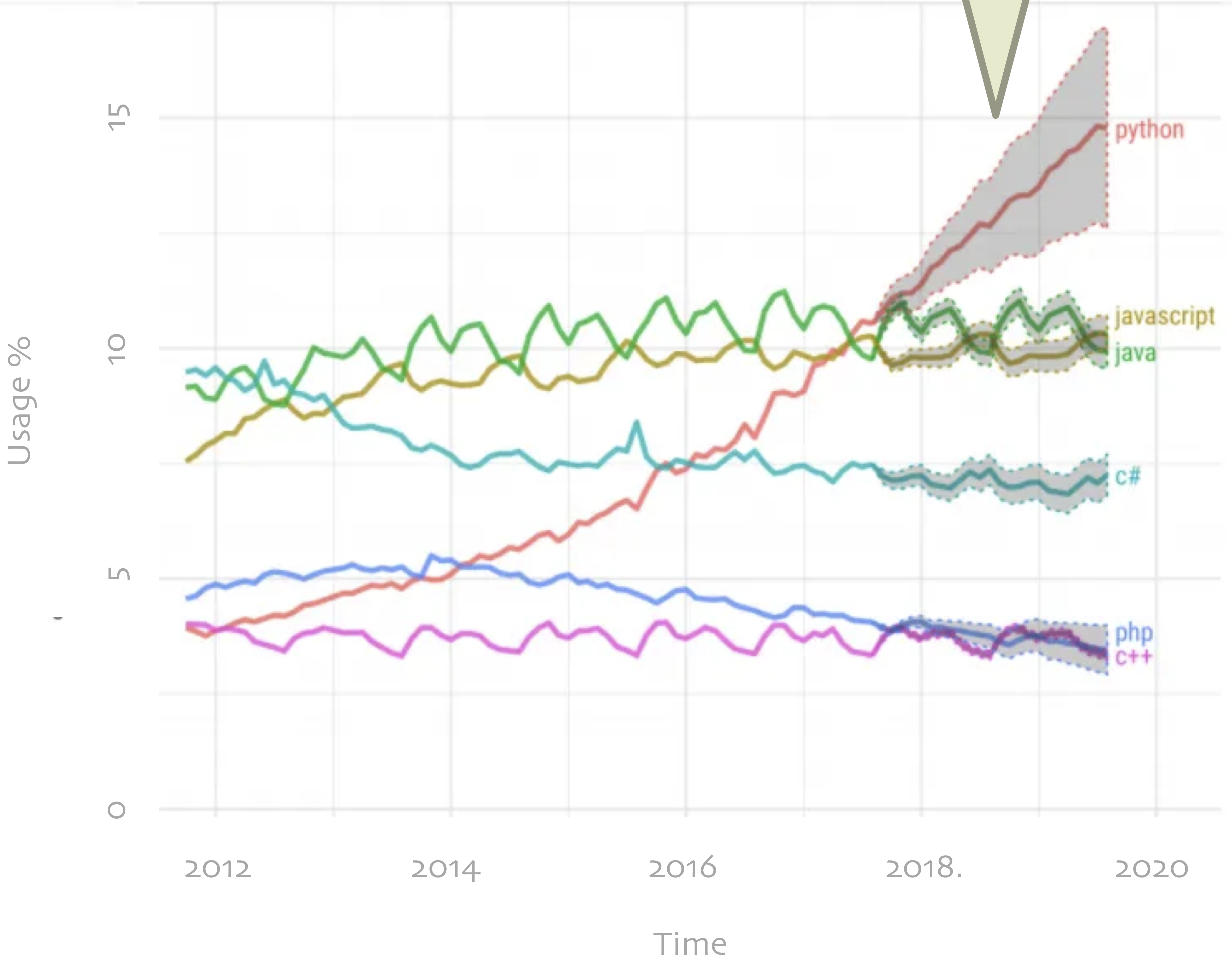
Model sizes and therefore compute requirements outstripping Moore's Law

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

APACHE Spark

hadoop

Streaming

APACHE Spark

kafka

Hyperparam Tuning

SIGOPT

talos AUTONOMY

Training

HOROVOD

PyTorch

scikit learn

Simulation

MPI

akka

Orleans

Model Serving

Clipper

The Ray Vision: Sharing a Common Framework

Domain-specific libraries for each subsystem

Featurization

Streaming

Hyperparam Tuning

Training

Simulation

Model Serving

Framework for distributed Python (and other languages…)

tune
raysgd
rllib
RAY SERVE
RAY

More libraries coming soon

RAY

Machine Learning with Ray-based Libraries

Ray Libraries

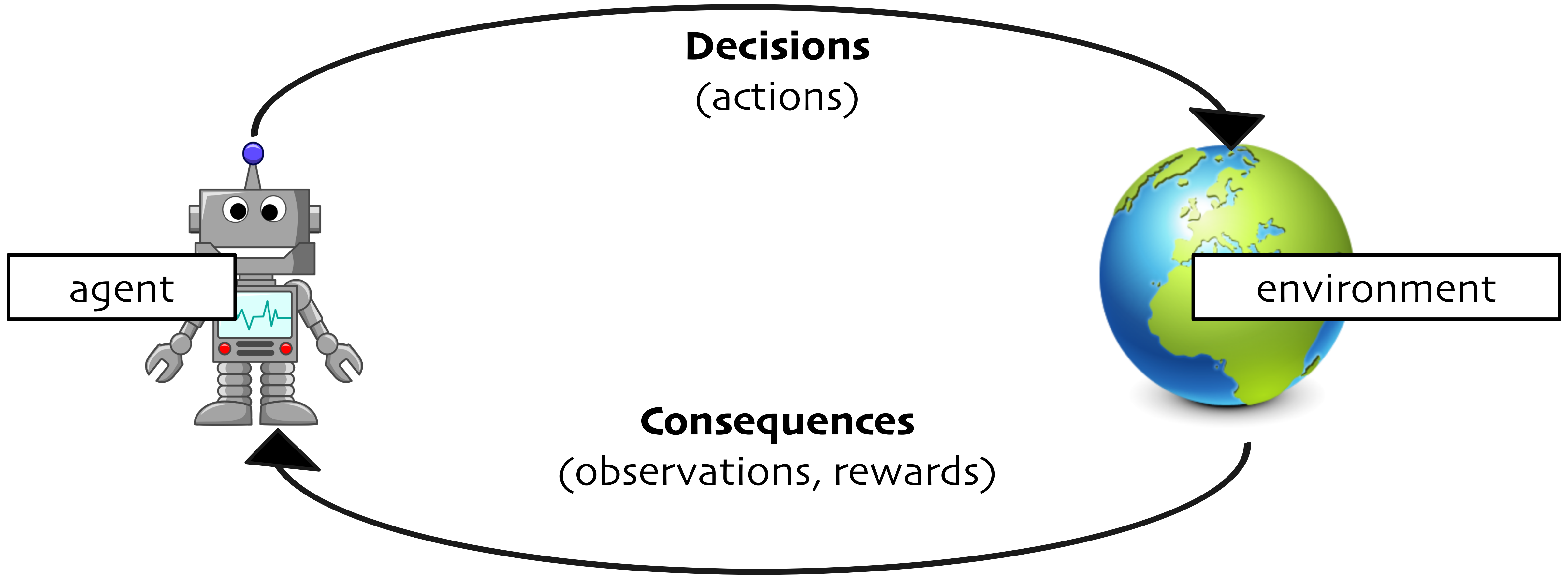
tune
raysgd
rllib
RAY SERVE
Featurization
Streaming
Hyperparam Tuning
Training
Simulation
Model Serving
RAY

Reinforcement Learning - Ray RLlib

tune
raysgd
rllib
RAY SERVE
Featurization
Streaming
Hyperparam Tuning
Training
Simulation
Model Serving
RAY

rllib.io

Reinforcement Learning



Games

Robotics, Autonomous Vehicles

Industrial Processes

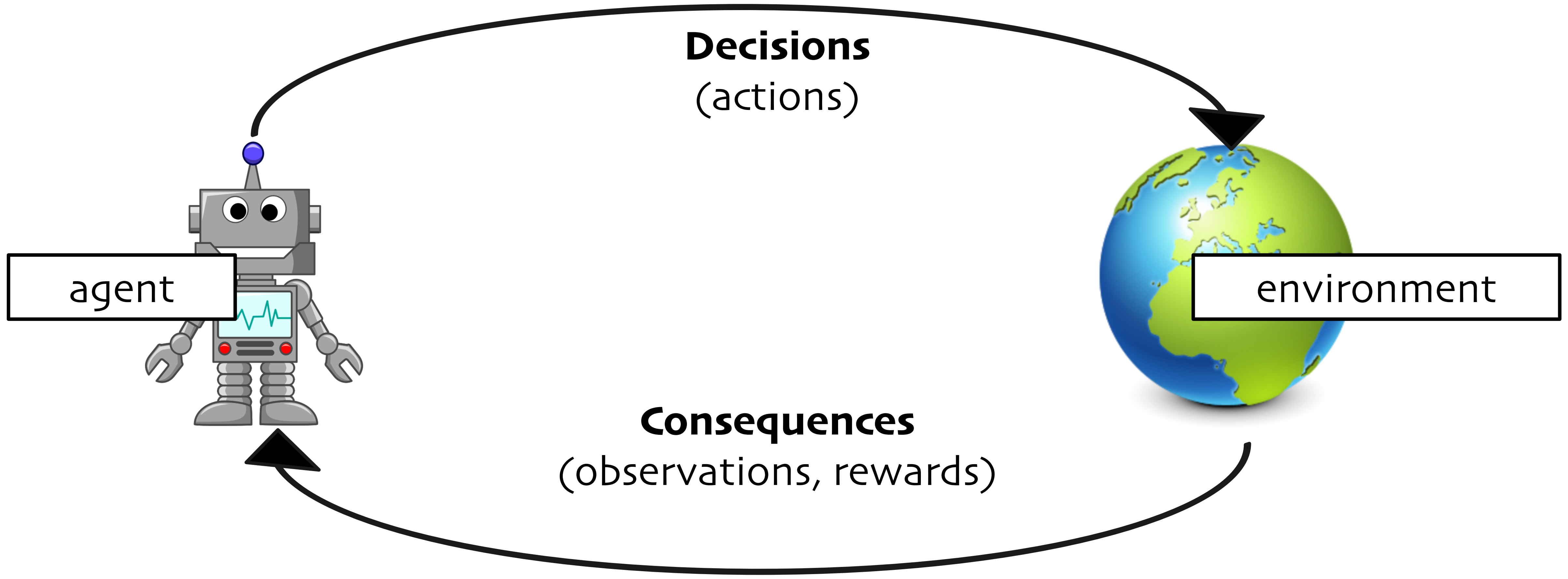
System Optimization

Advertising, Recommendations

Finance

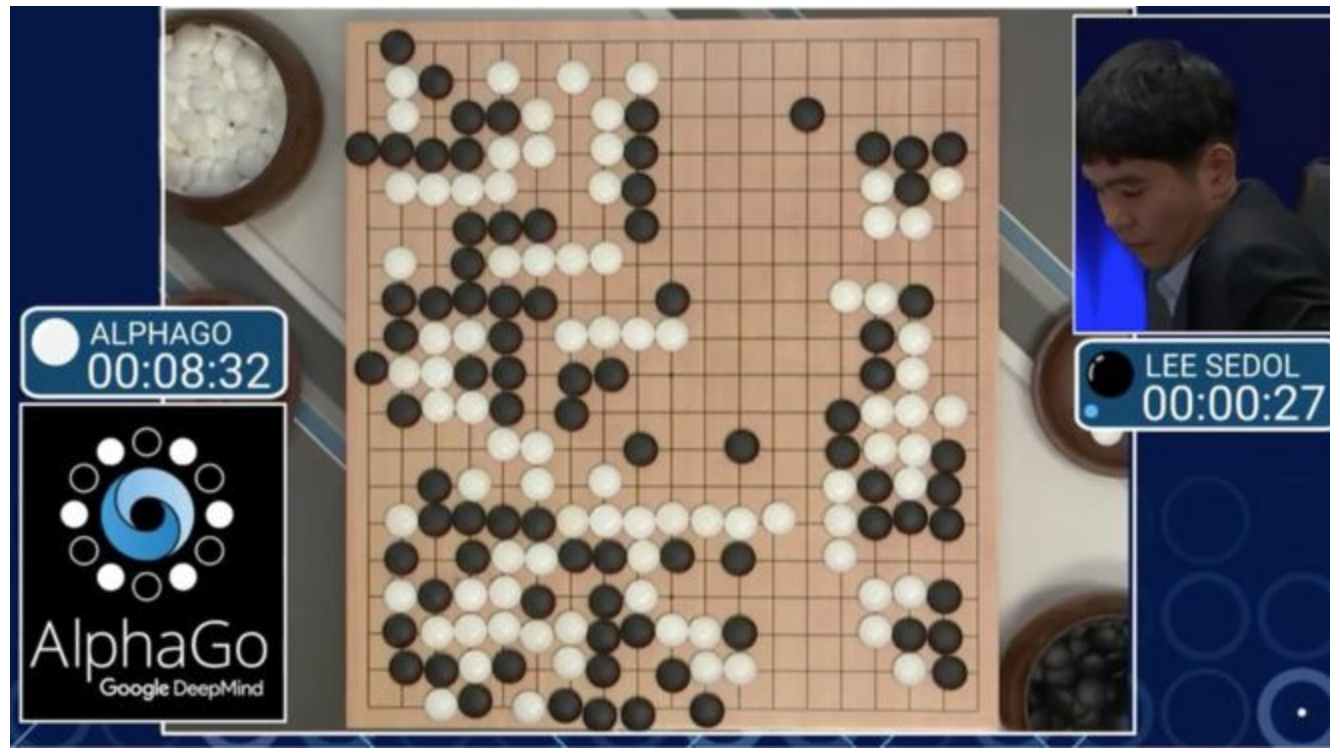
RL applications

Reinforcement Learning

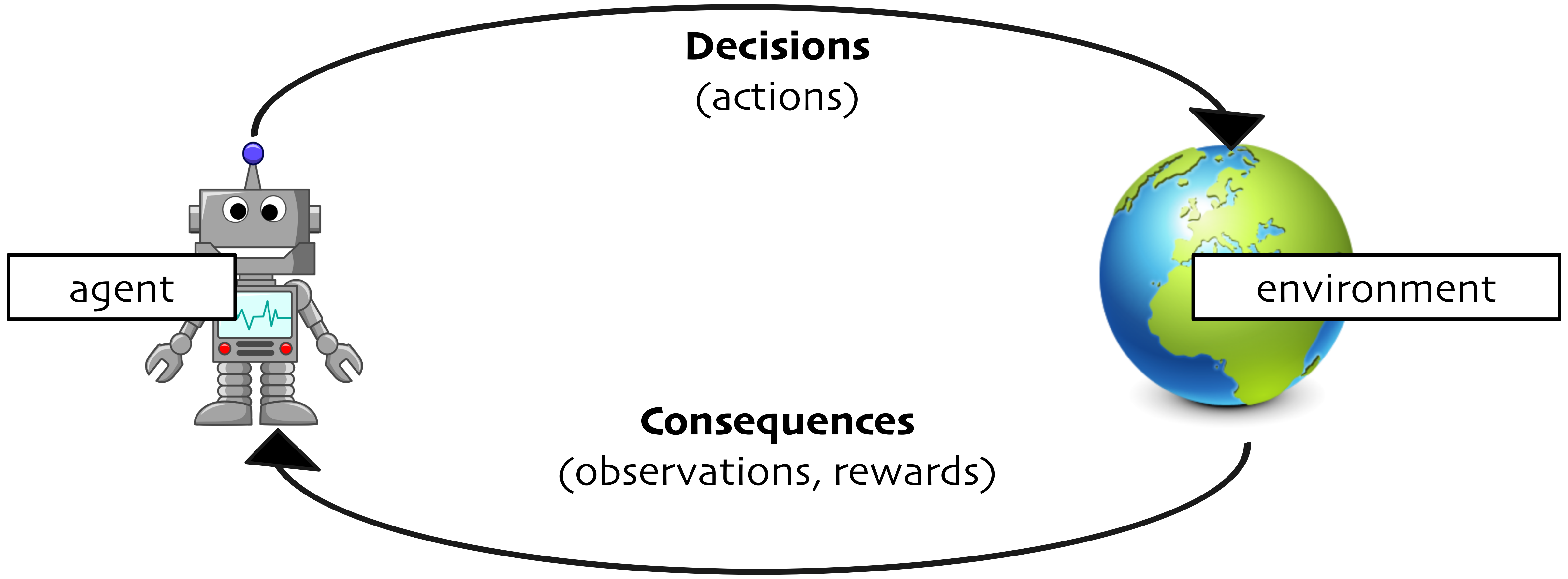


Games | Robotics, Autonomous Vehicles | Industrial Processes | System Optimization | Advertising, Recommendations | Finance

RL applications



Reinforcement Learning



Games

Robotics, Autonomous Vehicles

Industrial Processes

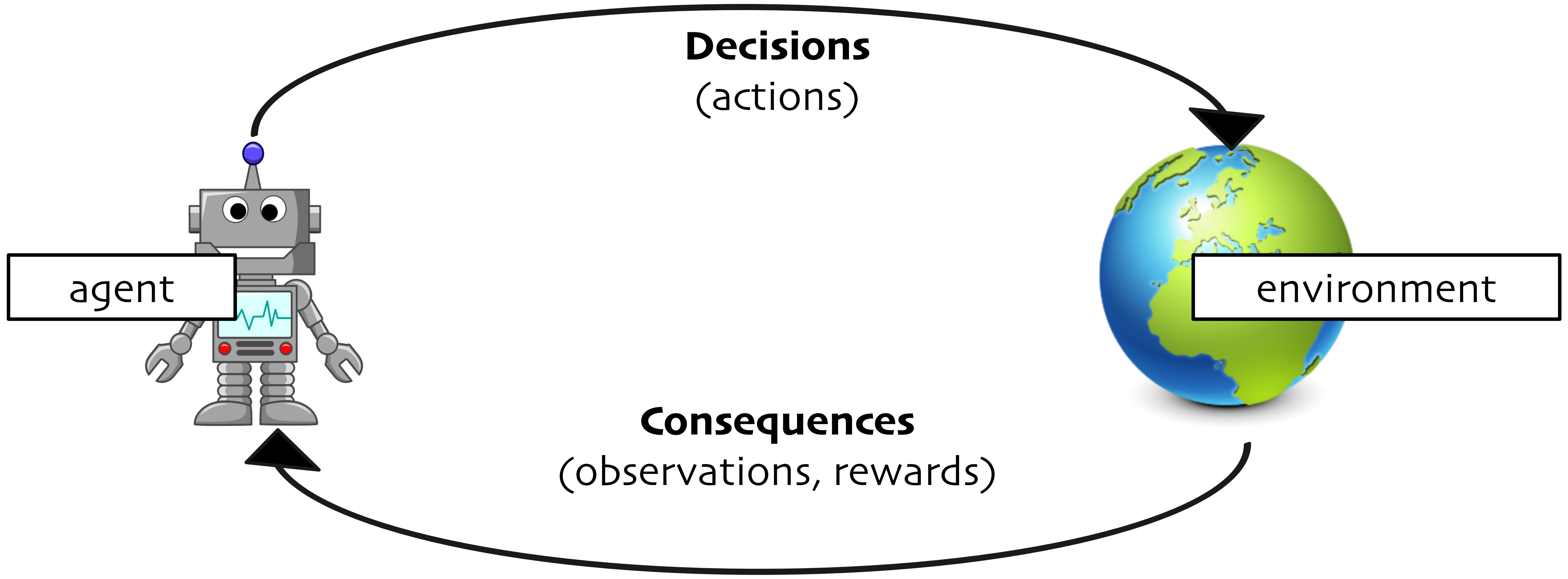
System Optimization

Advertising, Recommendations

Finance

RL applications

Reinforcement Learning



Games

Robotics, Autonomous Vehicles

Industrial Processes

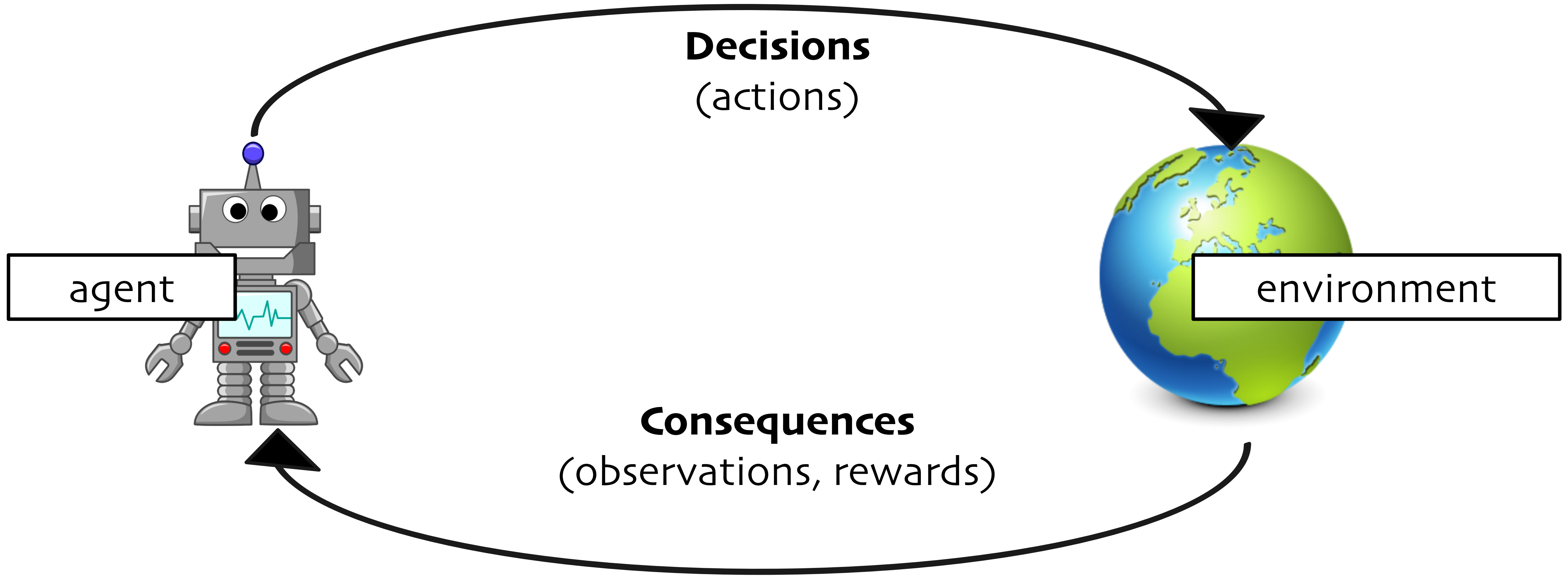
System Optimization

Advertising, Recommendations

Finance

RL applications

Reinforcement Learning



Decisions
(actions)

agent

environment

Consequences
(observations, rewards)

Games

Robotics, Autonomous Vehicles

Industrial Processes

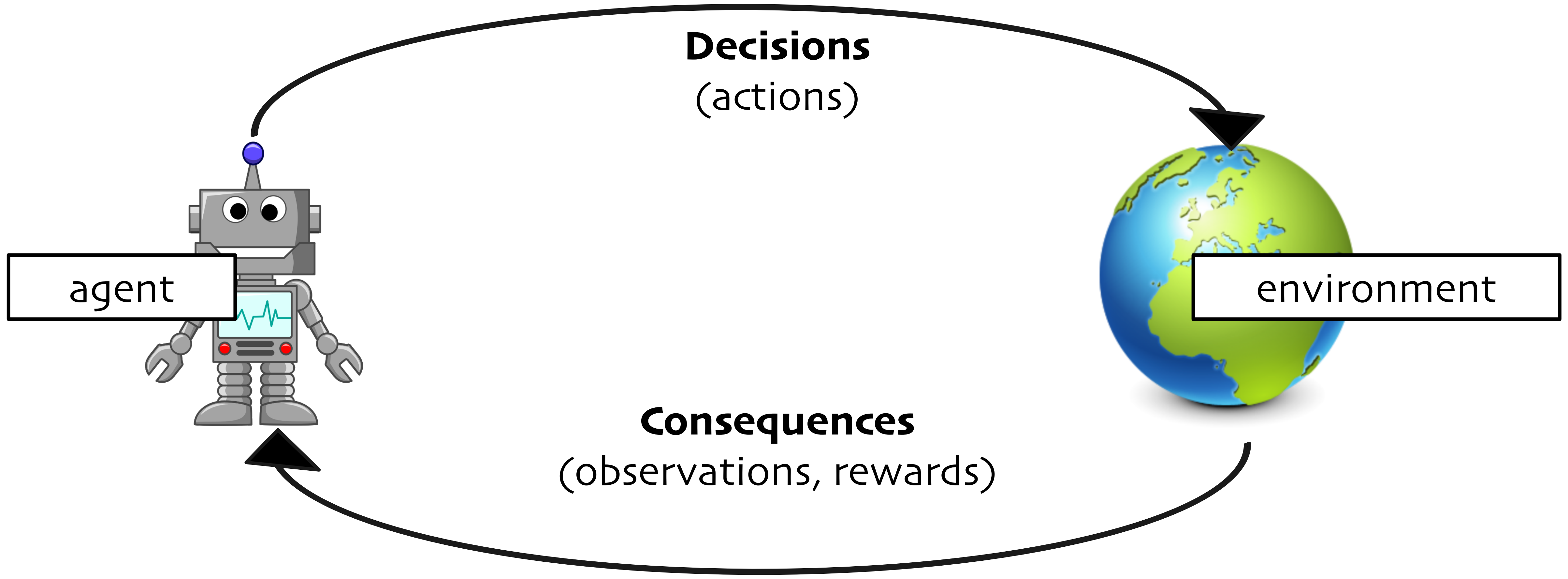
System Optimization

Advertising, Recommendations

Finance

RL applications

Reinforcement Learning



Games

Robotics, Autonomous Vehicles

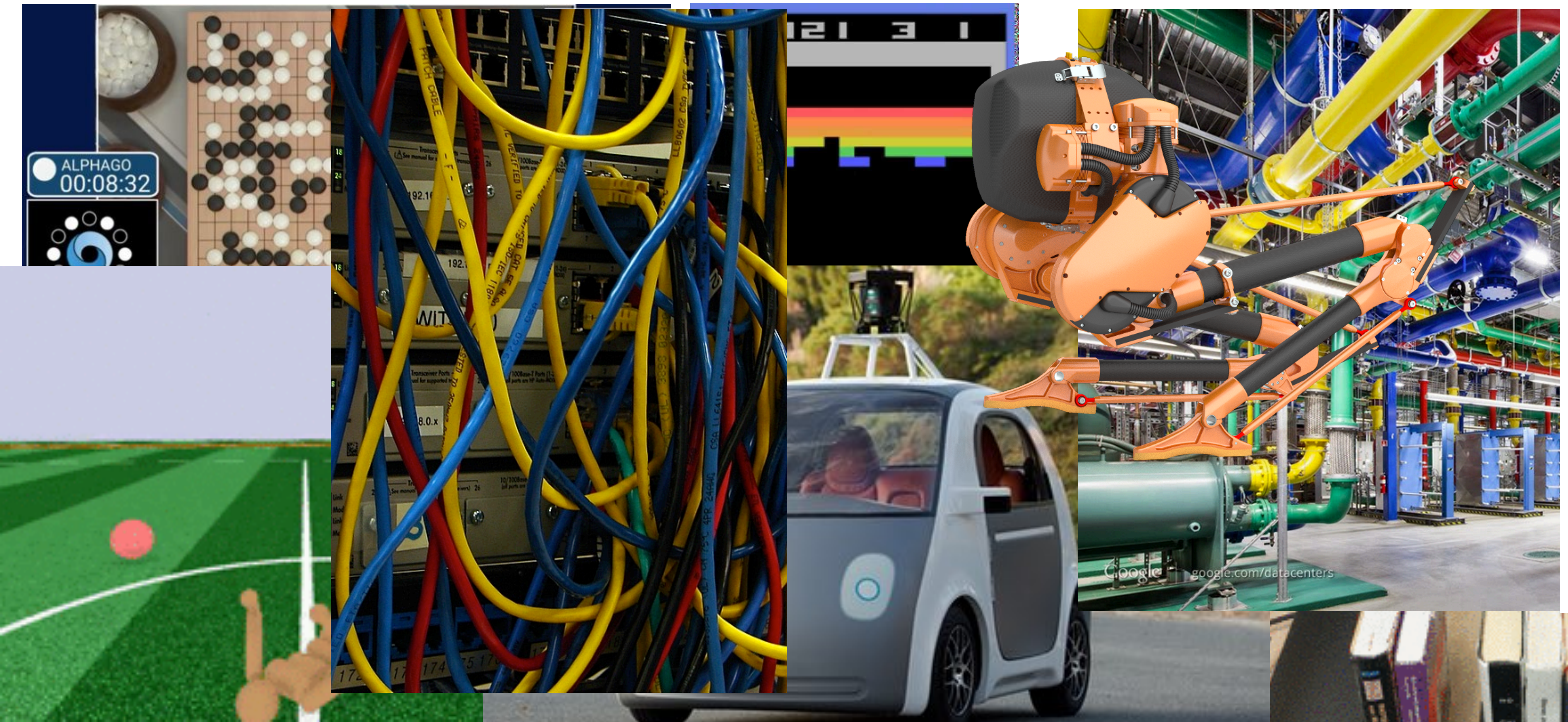
Industrial Processes

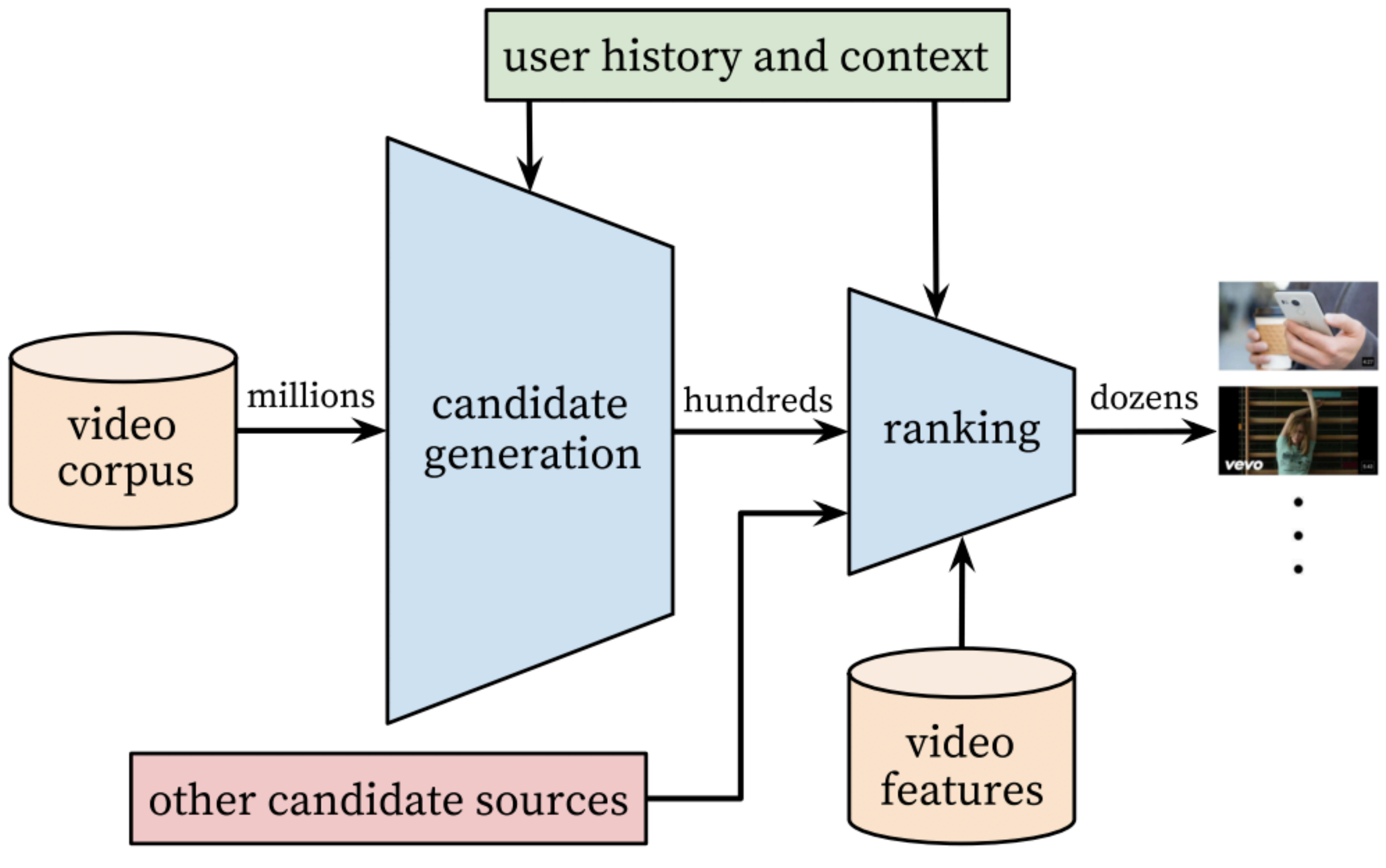
System Optimization

Advertising, Recommendations

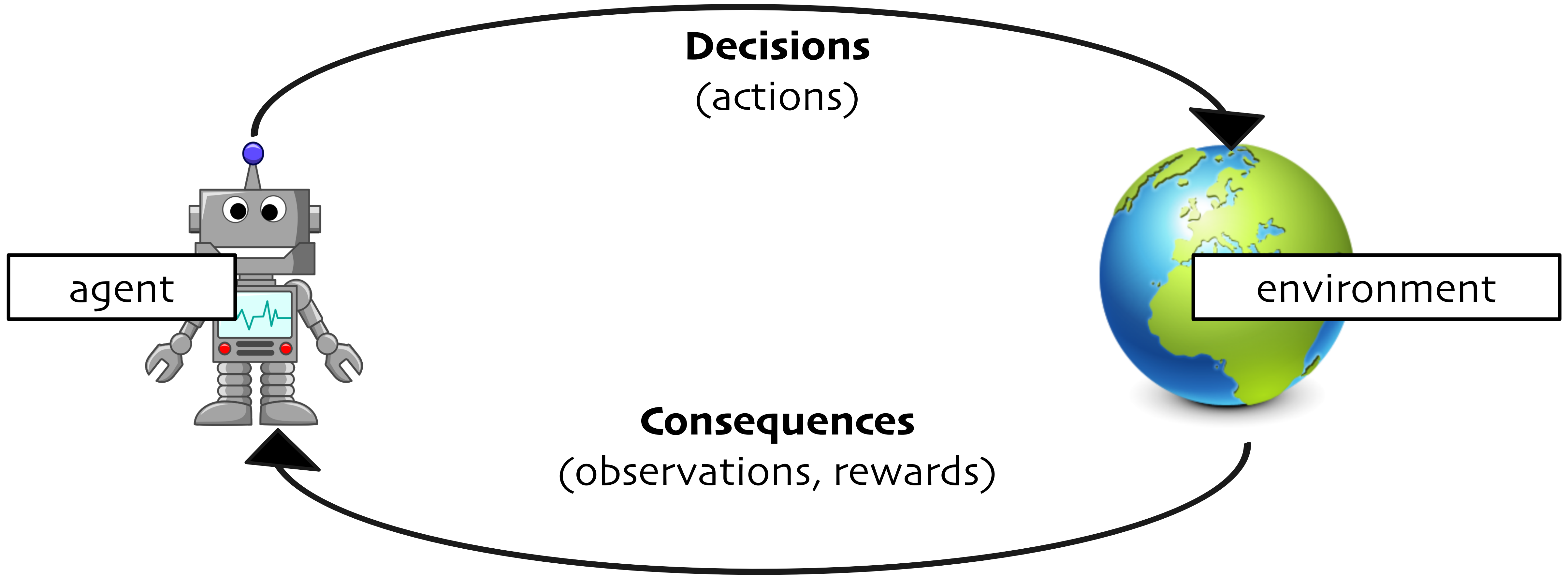
Finance

RL applications





Reinforcement Learning



Decisions
(actions)

agent

environment

Consequences
(observations, rewards)

Games

Robotics, Autonomous Vehicles

Industrial Processes

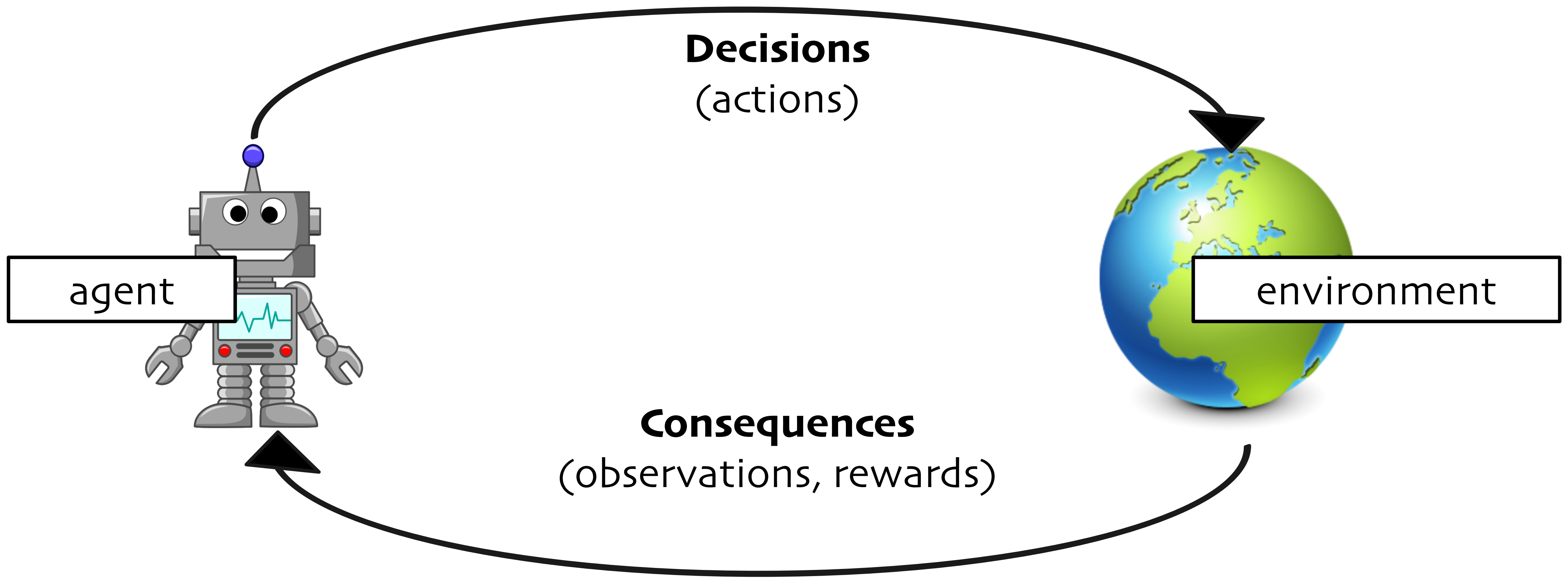
System Optimization

Advertising, Recommendations

Finance

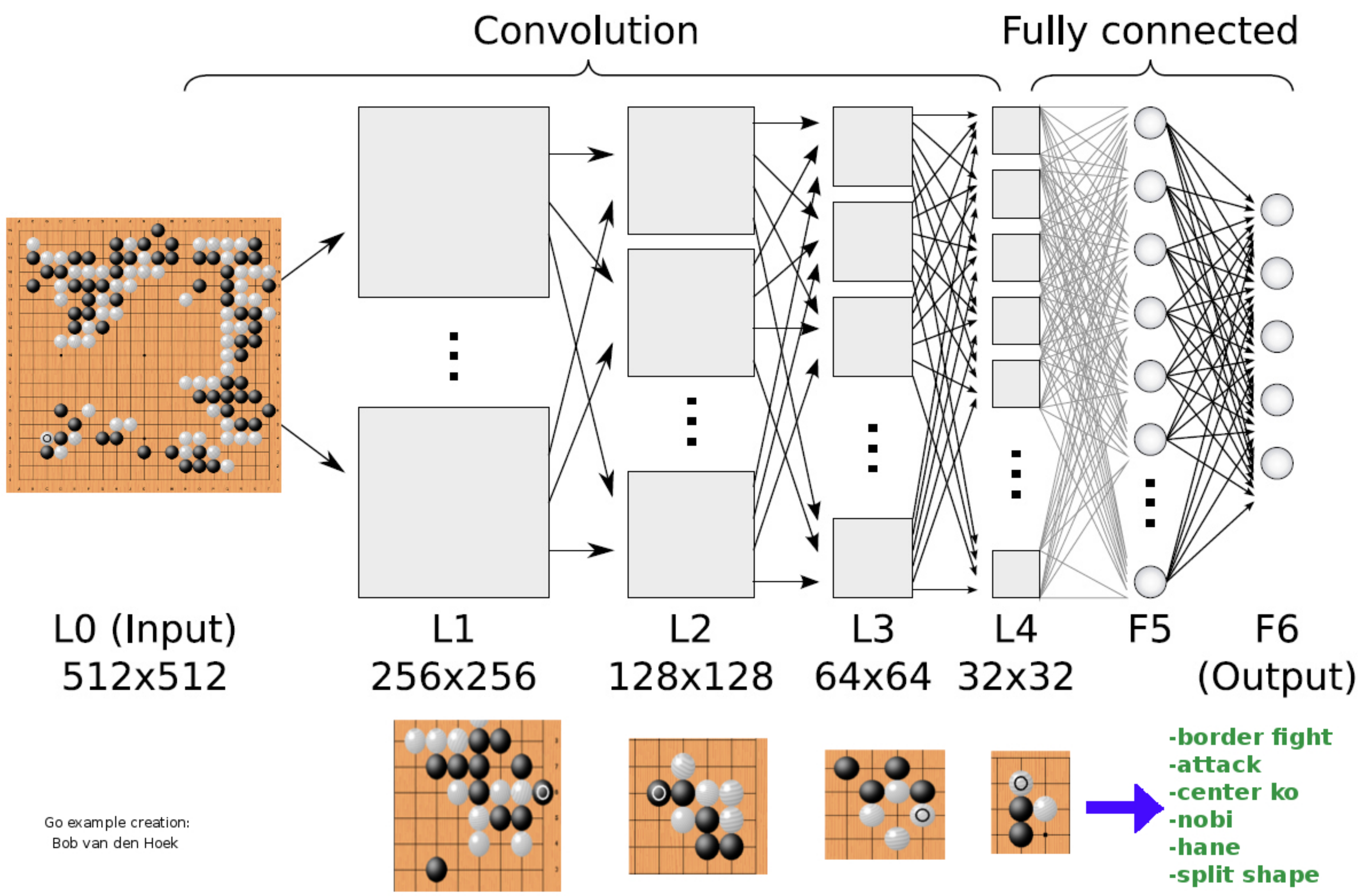
RL applications

Go as a Reinforcement Learning Problem



AlphaGo (Silver et al. 2016)

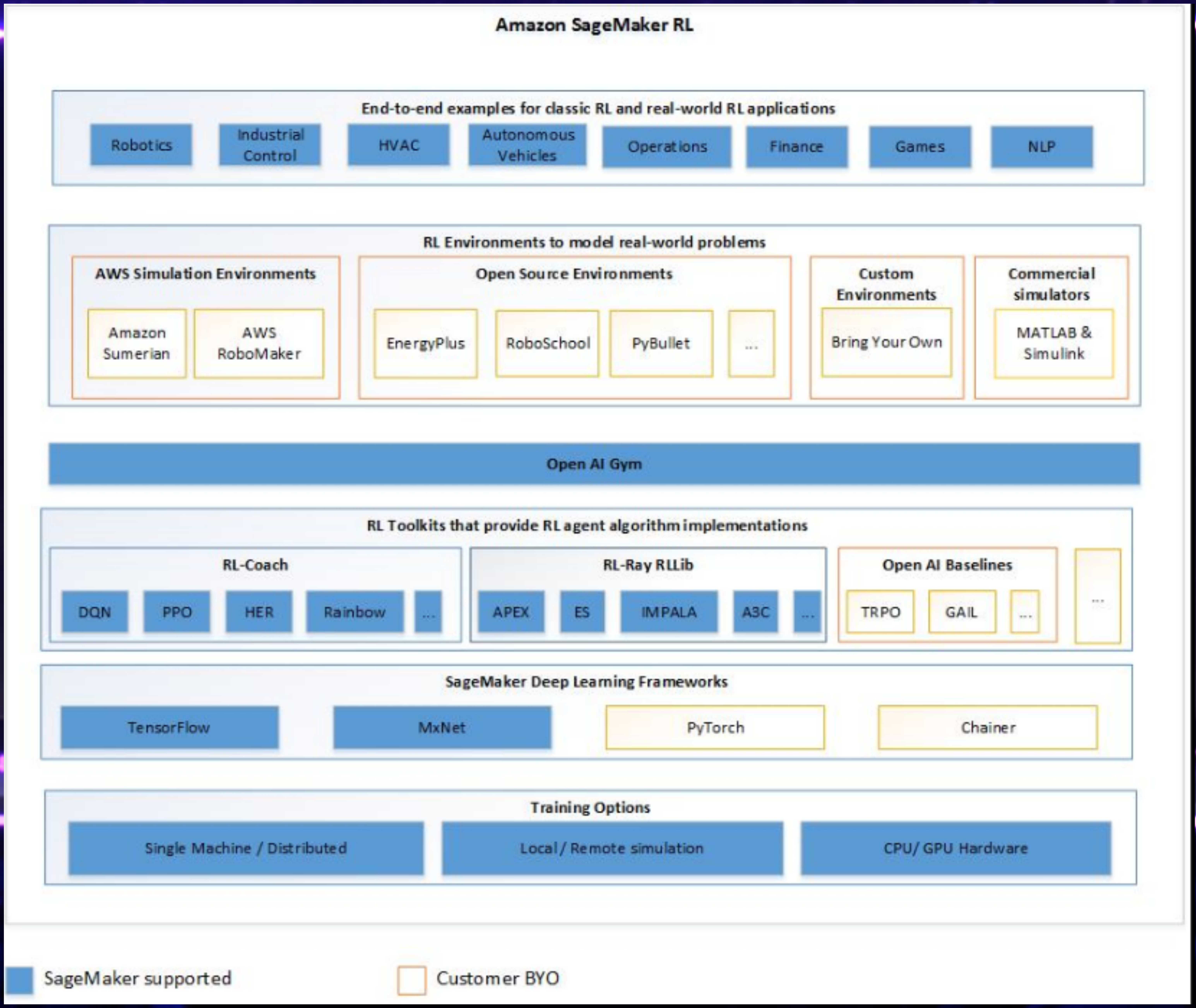
- **Observations**:
  - board state
- **Actions**:
  - where to place the stones
- **Rewards**:
  - 1 if win
  - 0 otherwise



Go example creation:
Bob van den Hoek

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
RLlib Abstractions
} (2) Abstractions for RL
Ray Tasks and Actors
} (3) Distributed Execution



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

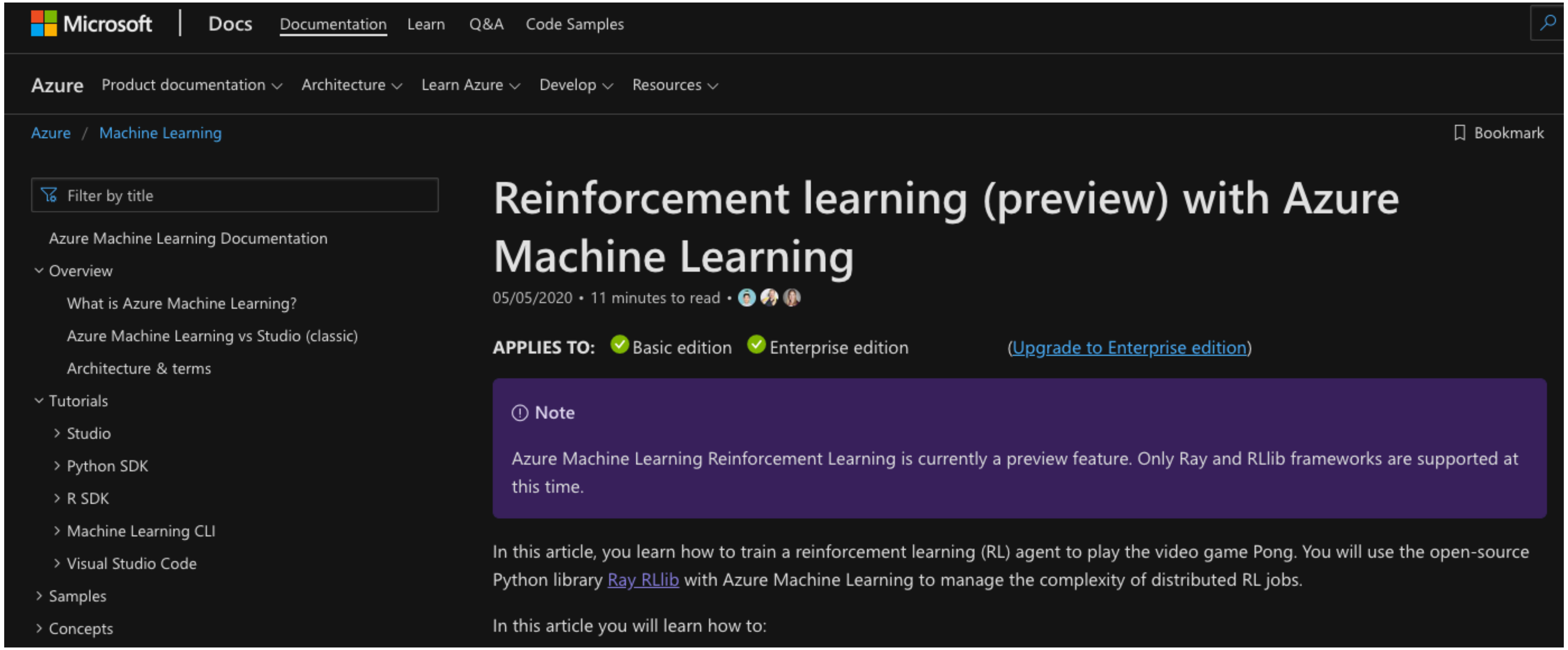
TensorFlow | MxNet | PyTorch | Chainer

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:

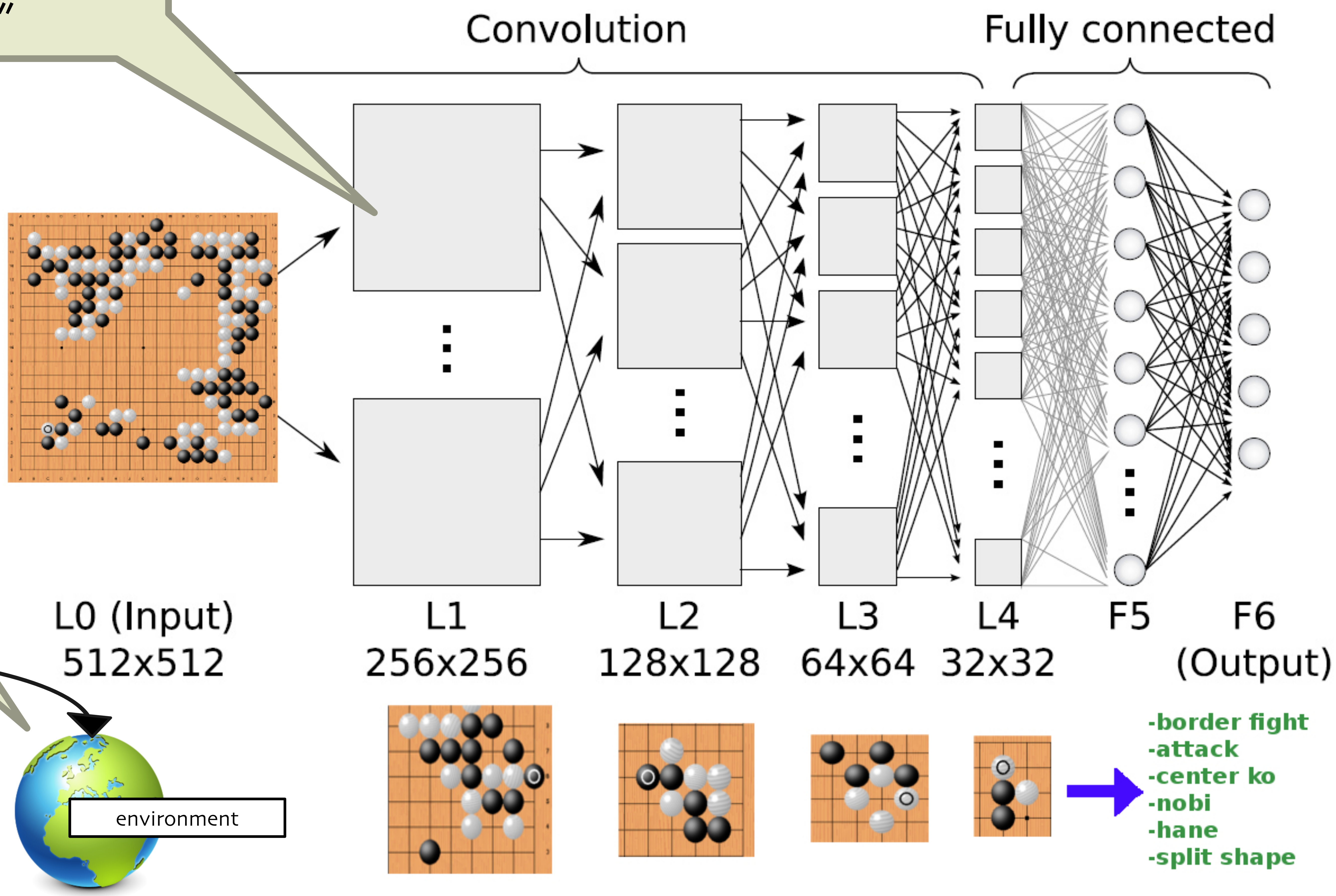
Diverse Compute Requirements Motivated Creation of Ray!

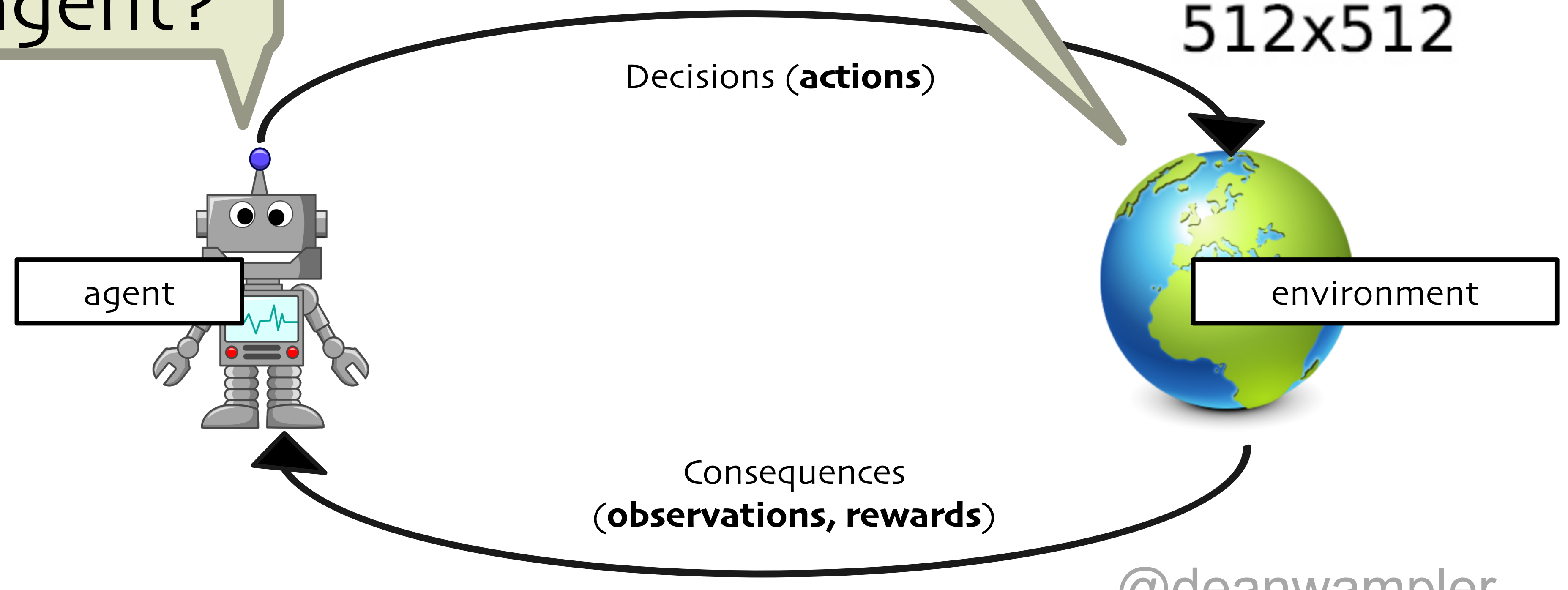
And repeated play, over and over again, to train for achieving the best reward

Neural network "stuff"

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

Complex agent?





Hyperparameter Tuning - Ray Tune

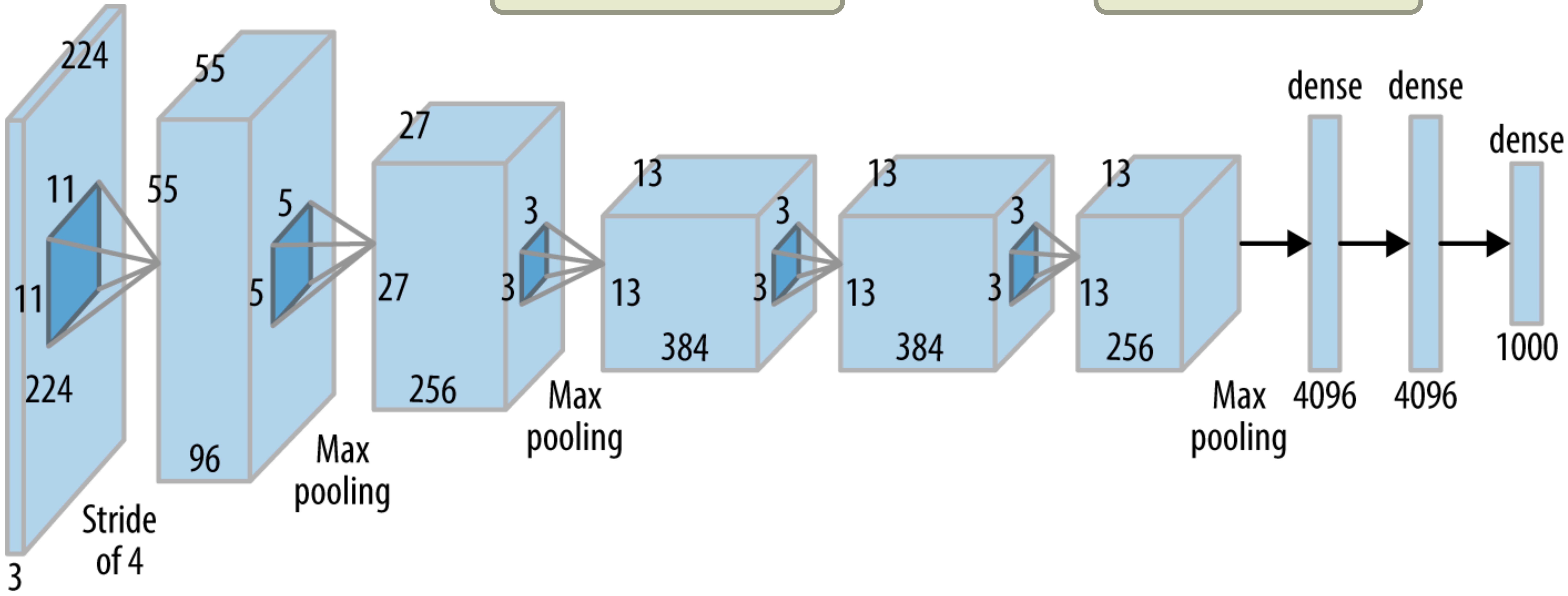
tune
raysgd
rllib
RAY SERVE
Featurization
Streaming
Hyperparam Tuning
Training
Simulation
Model Serving
RAY

tune.io

Nontrivial Example - Neural Networks

How many layers?
What kinds of layers?

Every number shown is a hyperparameter!

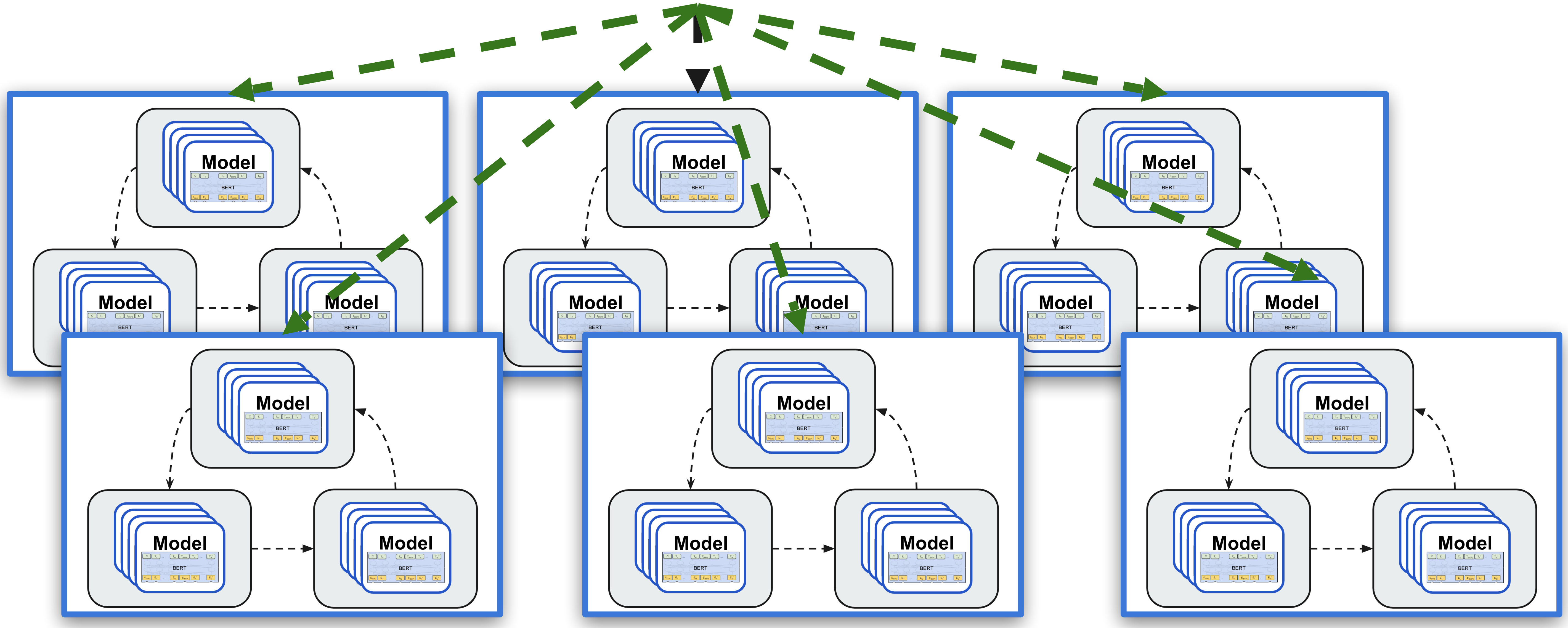


@deanwampler

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()
)
```

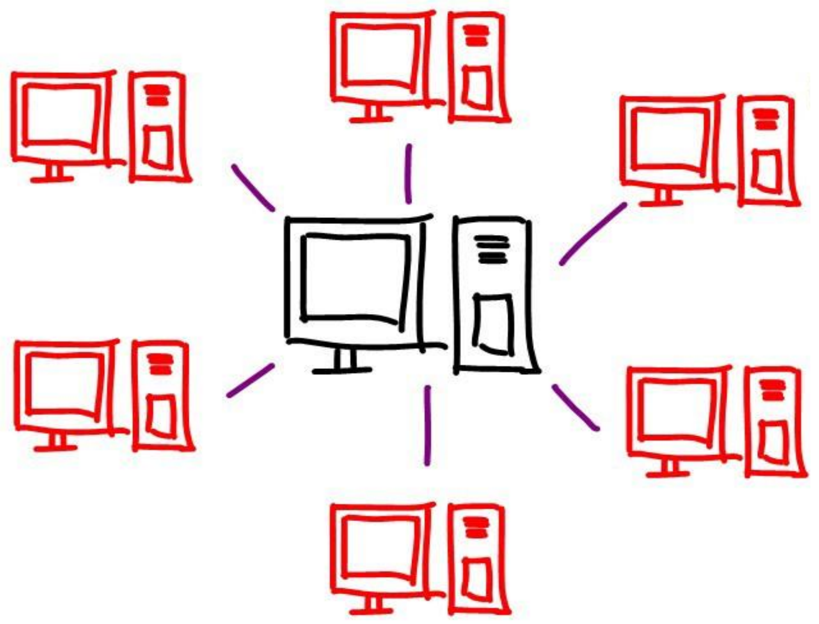
tune



Tune is Built with Deep Learning as a Priority

Resource Aware Scheduling

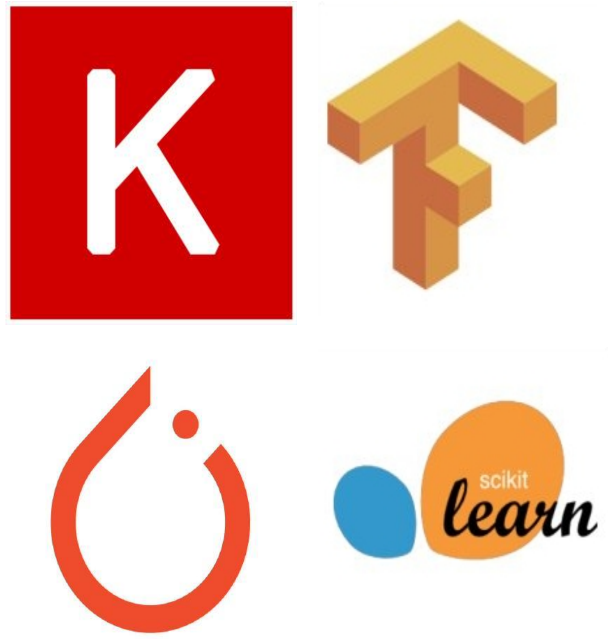
Seamless Distributed Execution



Simple API for new algorithms

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

Framework Agnostic



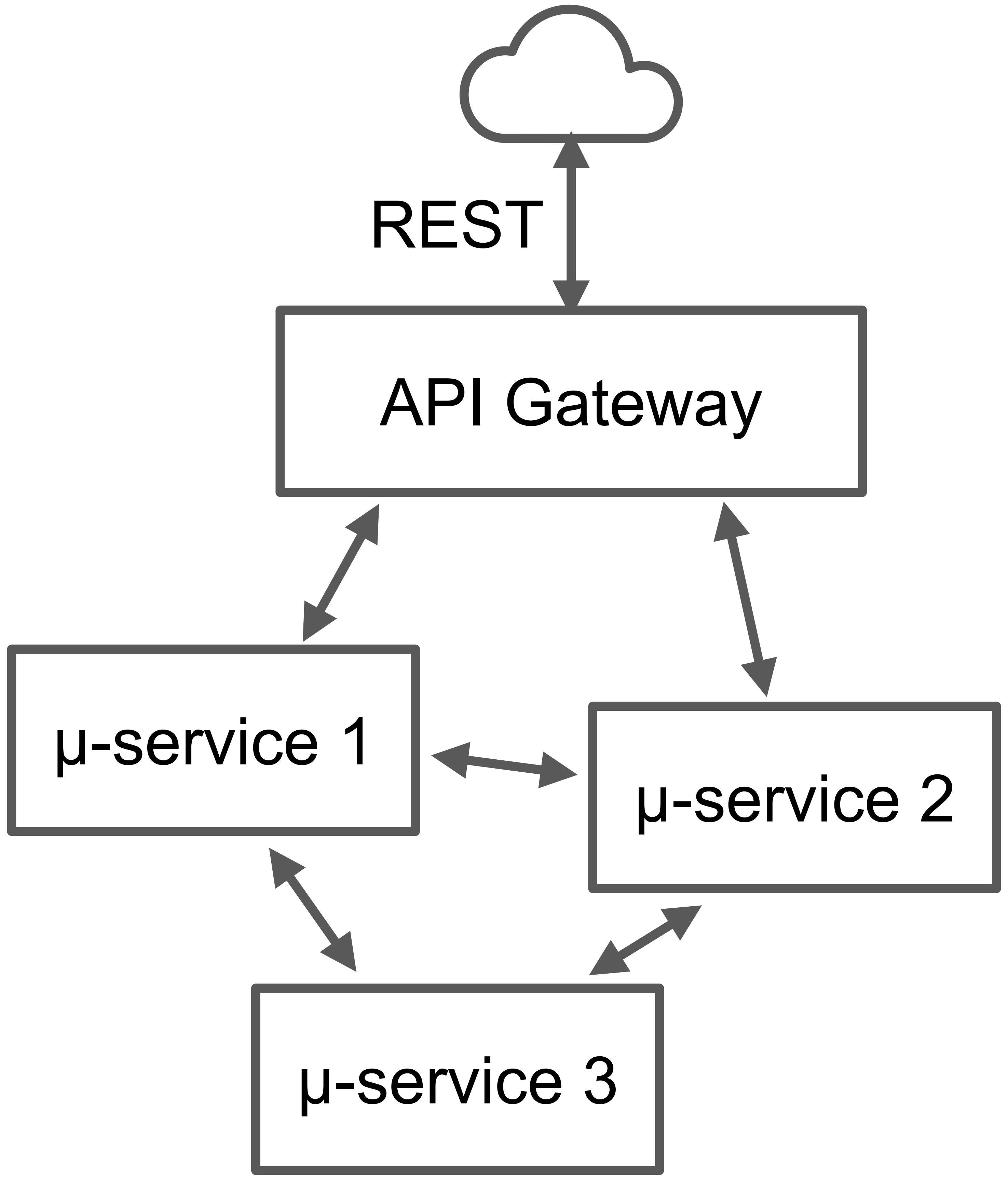
tune.io

RAY

What about Ray for Microservices?

What Are Microservices?

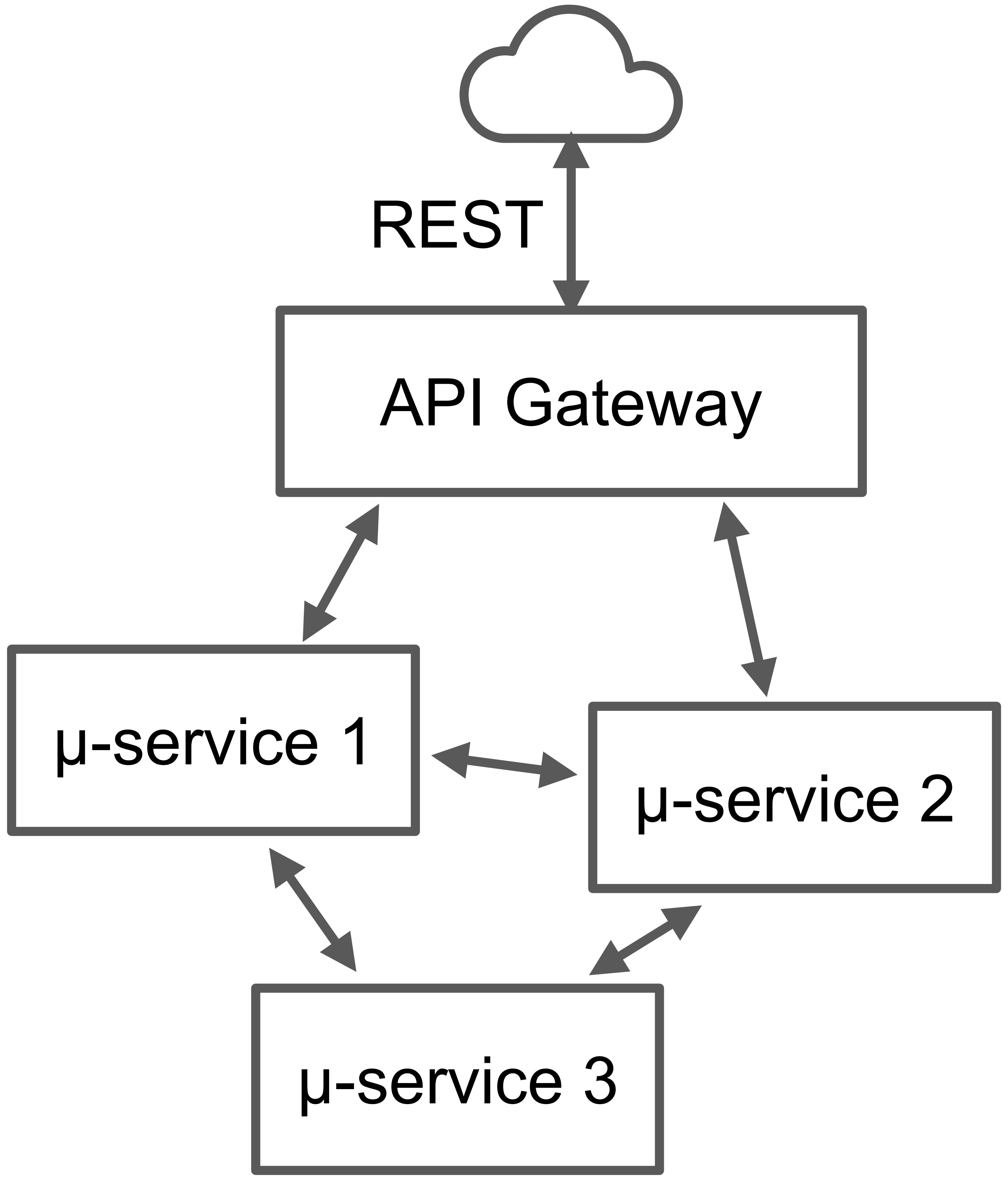
- They partition the domain
  - Conway's Law - Embraced
  - Separate responsibilities
- Separate management



What Are Microservices?

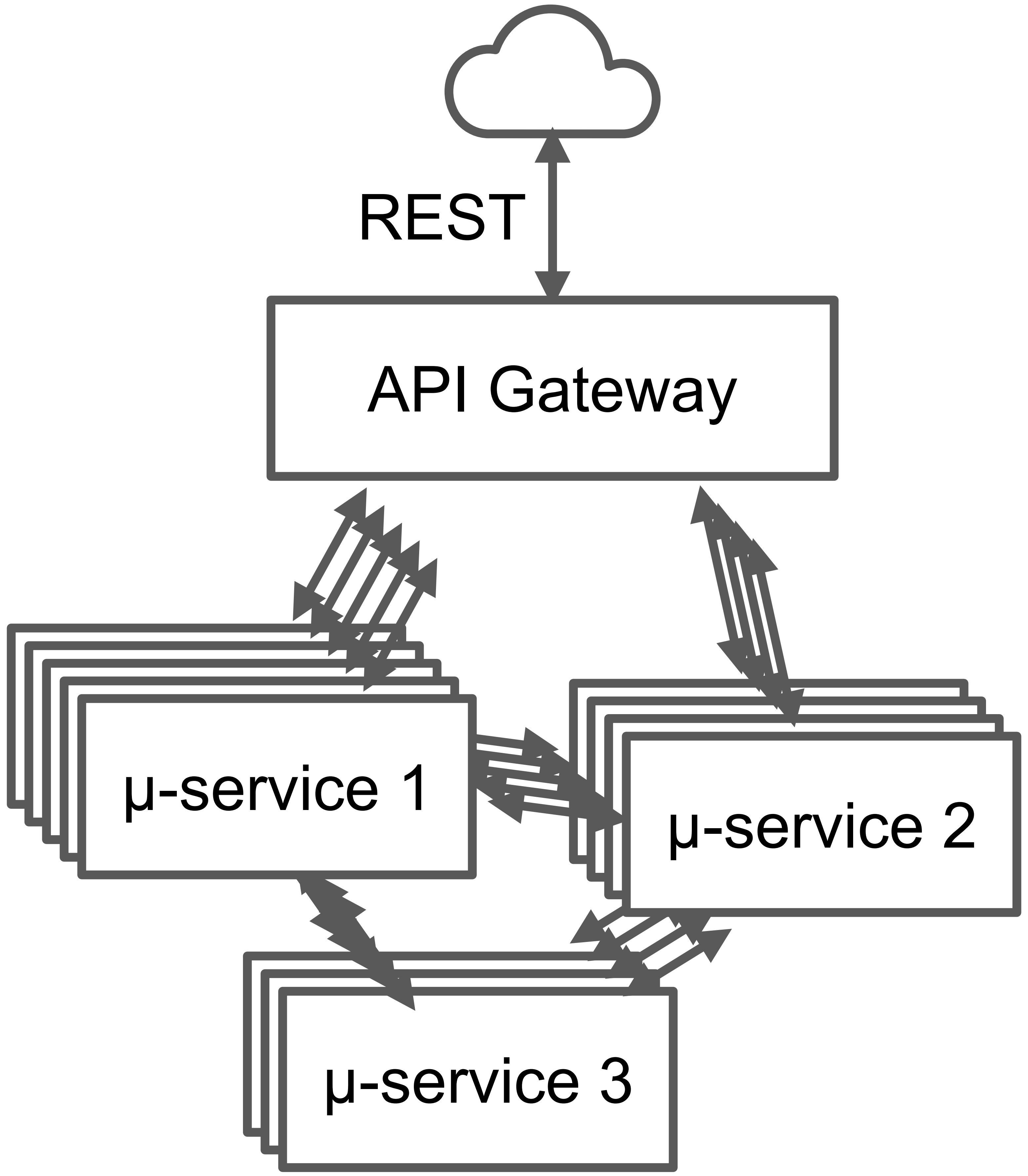
- They partition the domain
  - Conway's Law - Embraced
  - Separate responsibilities
- Separate management

What's most interesting for our purposes today.



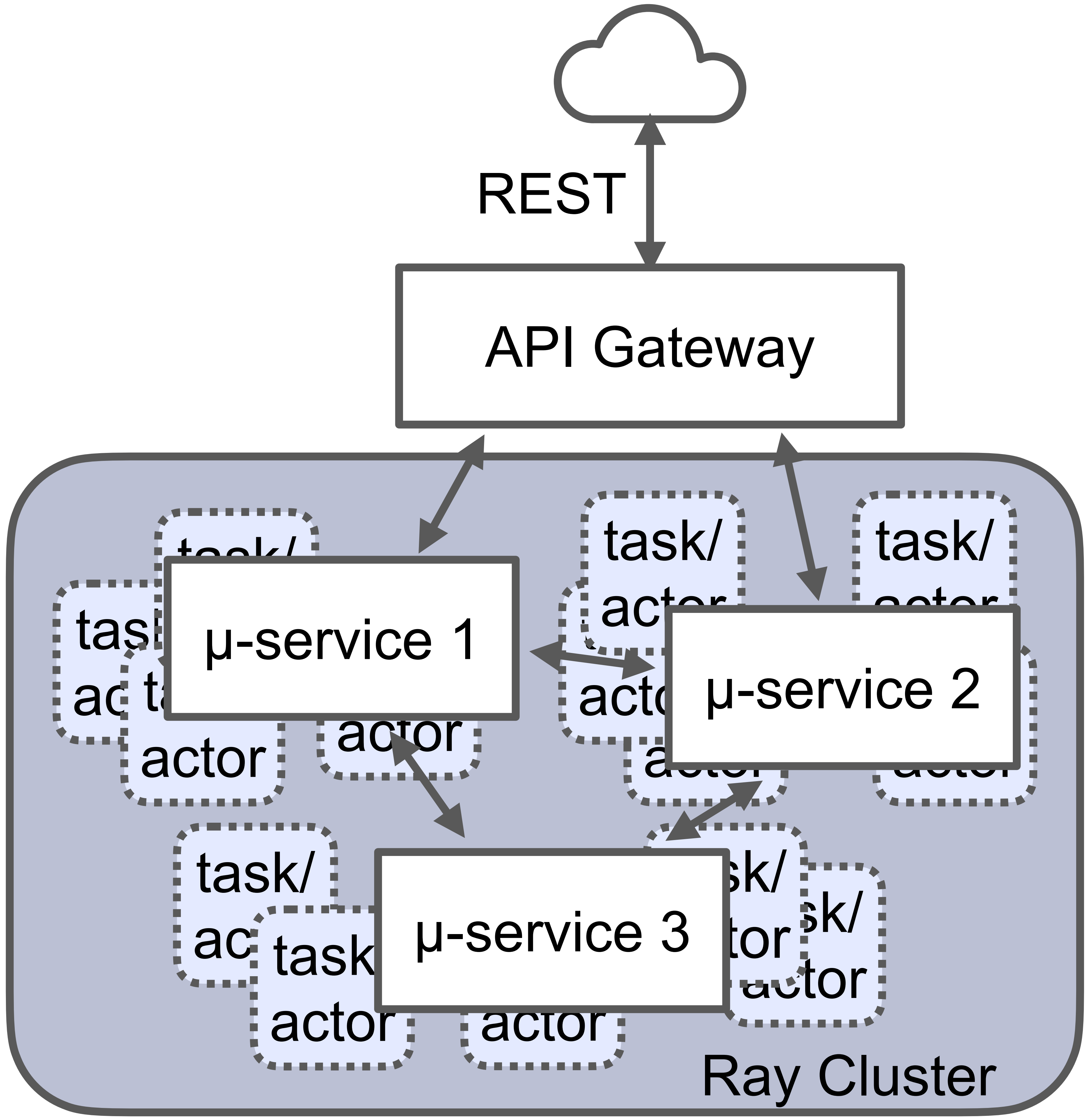
Separate Management

- 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**



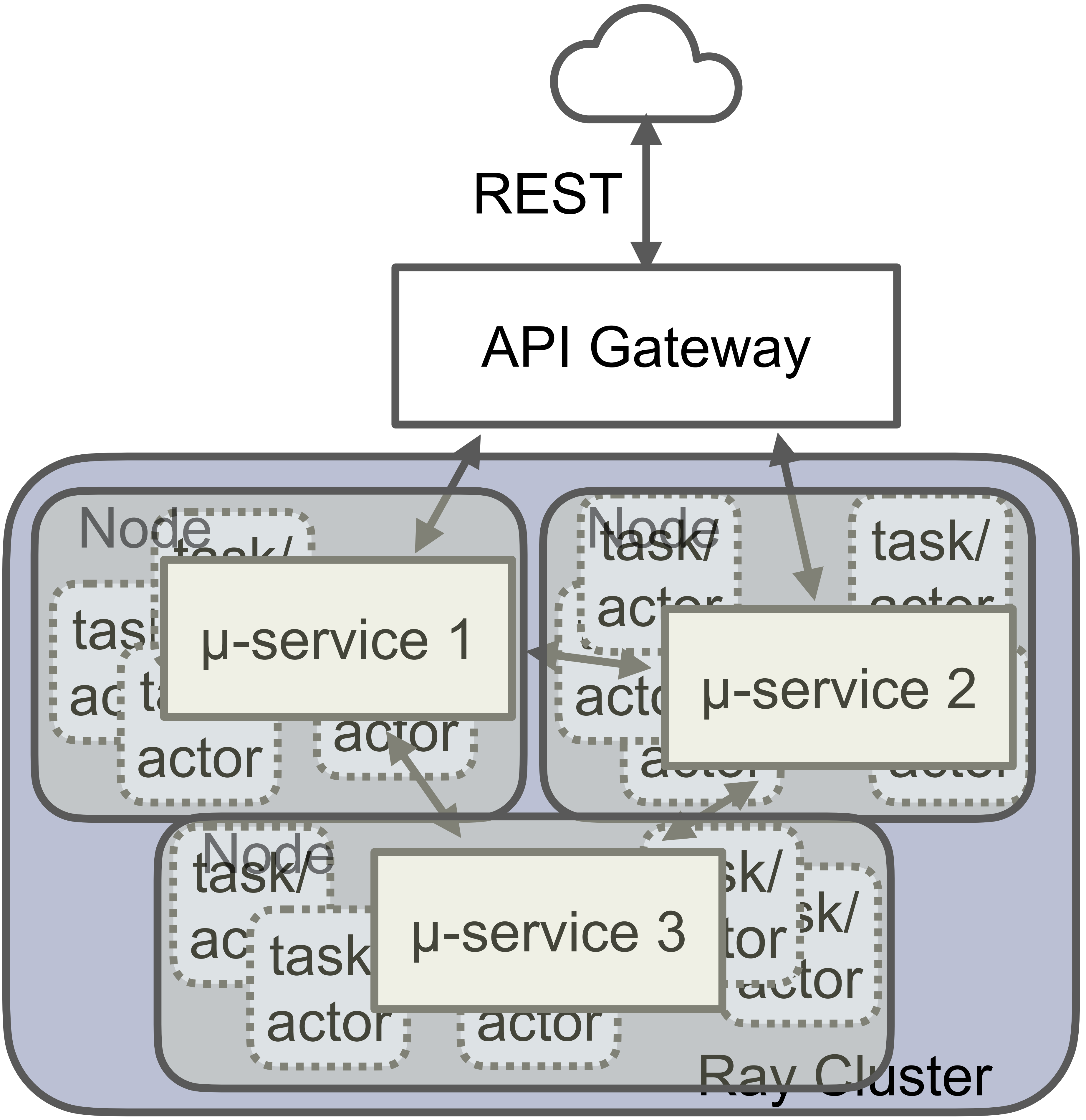
Management - Simplified

- 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

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

RAY SUMMIT

Presented by Anyscale

Scalable machine learning, scalable Python, for everyone

September 30 – October 1

FEATURED SPEAKERS

Michael Jordan
Distinguished Professor, University of California, Berkeley

Azalia Mirhoseini
Senior Research Scientist, Google Brain

Wes McKinney
Founder, Ursa Labs

Manuela Veloso
Head of J.P. Morgan AI Research

Gaël Varoquaux
Tenured Research Director, Inria

Ion Stoica
Professor, University of California, Berkeley

Zoubin Ghahramani
Chief Scientist & VP, Artificial Intelligence, Uber Technologies

raysummit.org

anyscale

Thanks for Listening

ray.io

anyscale.com - We're Hiring!

dean@anyscale.com

@deanwampler

RAY SUMMIT
Presented by Anyscale
September 30 – October 1, 2020 | Virtual Experience

https://anyscale.com/events