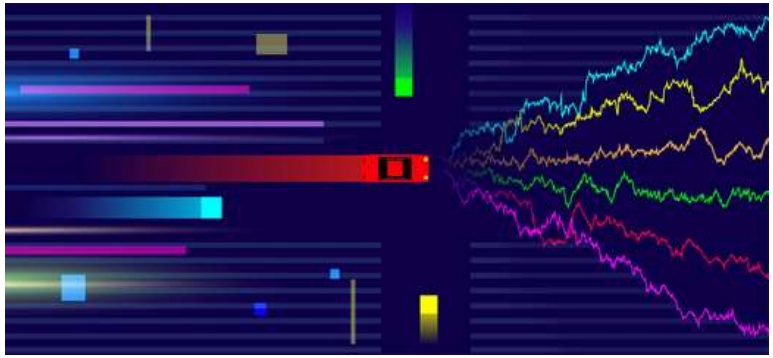
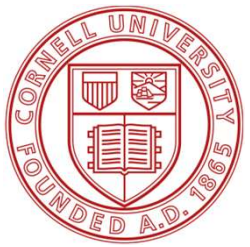
RL for Operations
Day 1: MDP Basics, VI+PI, Deep RL

Sean Sinclair, Sid Banerjee, Christina Yu
Cornell University




CORNELL UNIVERSITY
FOUNDED A.D. 1865

Plan for Today

MDP Basics

- Basic framework for Markov Decision Processes
- Tabular RL Algorithms with policy iteration + value iteration
- DeepRL algorithms (and their "tabular" counterparts)

Simulation Packages

- OpenAI Framework for simulation design
- Existing packages and code-bases for RL algorithm development

Simulation Implementation

- Developed simulator for problem using OpenAI Gym API

Tabular RL Algorithms

- Implement basic tabular RL algorithms to understand key algorithmic design aspects of *value estimates + value iteration*, *policy iteration*

Plan for Today

MDP Basics

- Basic framework for Markov Decision Processes
- Tabular RL Algorithms with policy iteration + value iteration
- DeepRL algorithms (and their "tabular" counterparts)

Simulation Packages

- OpenAI Framework for simulation design
- Existing packages and code-bases for RL algorithm development

Simulation Implementation

- Developed simulator for problem using OpenAI Gym API

Tabular RL Algorithms

- Implement basic tabular RL algorithms to understand key algorithmic design aspects of *value estimates + value iteration*, *policy iteration*

Simulation Packages

Sean Sinclair,
Cornell University

MARO

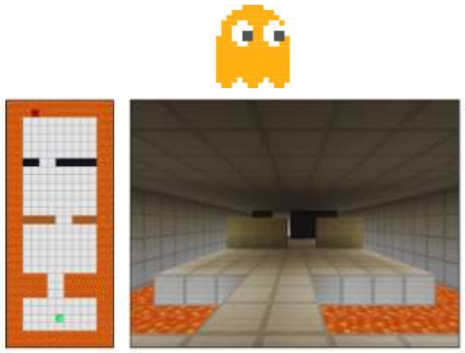
ORSuite

Gym

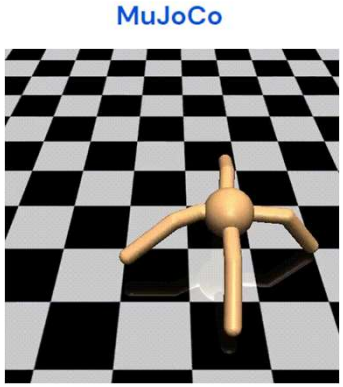
rlberry

MuJoCo

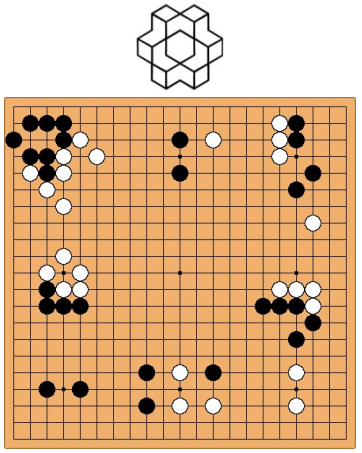
Typical RL Platforms



Games with Image Feedback



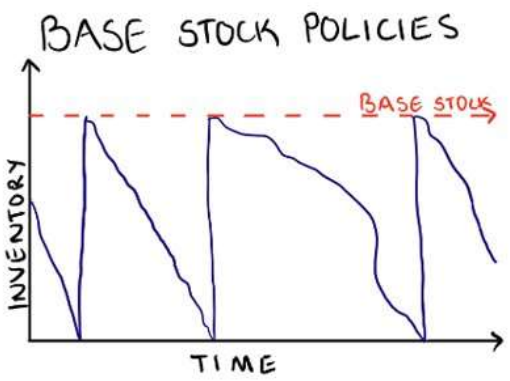
MuJoCo Simulator



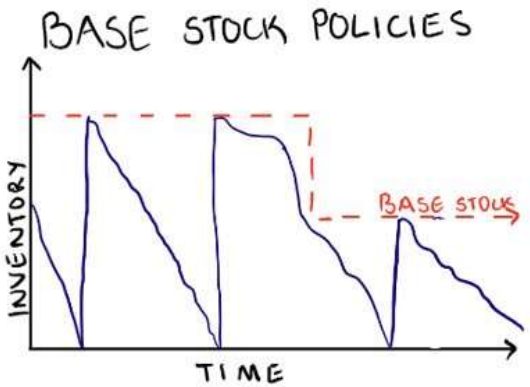
AlphaGo Zero

Focused on game playing and robotics, algorithms use good simulators for training data

Sequential decision making is at the heart of OR



Change base stock level



Make impactful decisions that affect cost and dynamics

Philosophy

Open Source

Publishable and reproducible results

Evaluation

Metrics should match effects and concerns

Traces/Stochastic

Include both traces and stochastic events to avoid overfitting

Design

Agent design / learning should simulate 'real-world'

Range of Difficulties

Produce instances with range of difficulties

Feedback

Environments should simulate feedback effects of actions

Benchmarks

Provide benchmark algorithms, 'MNIST' of RL

Markov Decision Process (MDP)

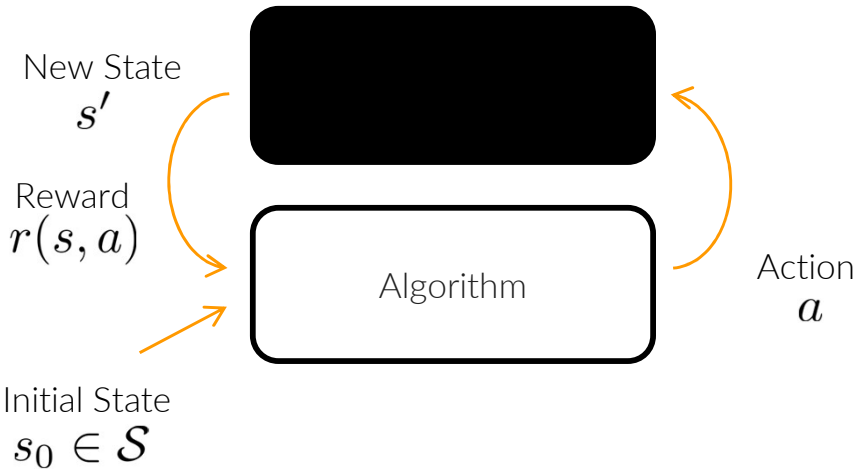
Environment

Gym

Specify Environment

- Immediate reward function
- Conditional transition distribution
- Additional side information available to algorithm
- Standardized with OpenAI Gym API

Agent

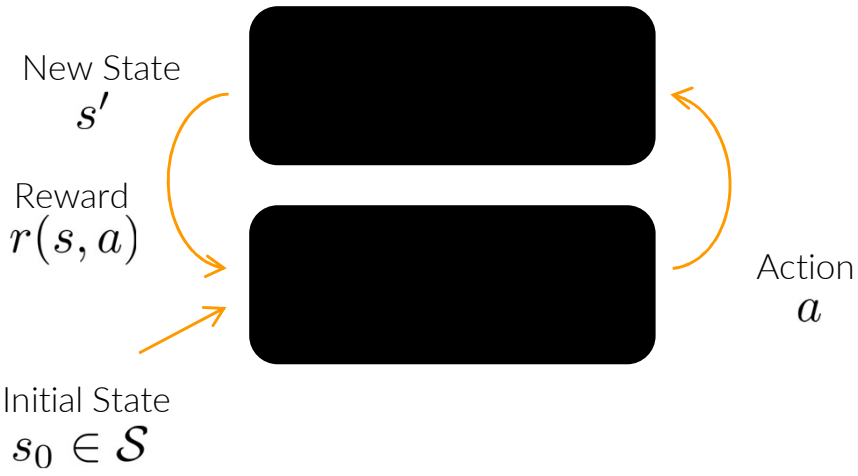


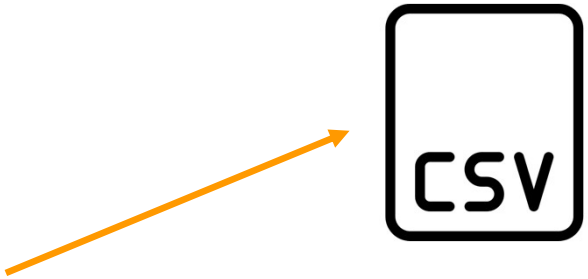

```python
class Agent(object):

    def __init__(self):
        pass

    def update_config(self, env, config):
        ''' Update agent information based on the config_file'''
        pass

    def update_obs(self, obs, action, reward, newObs):
        '''Add observation to records'''
```

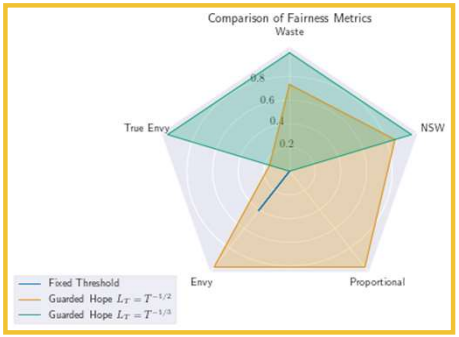
Specify Agent

- Define policy
- Update policy based on observed information from environment
- Custom specification, aimed at mimicking how most RL algorithms are developed

Instrumentation







Specify Instrumentation

- Runs multiple simulations between environment and algorithm
- Calculates dataset based on observed trajectories
- Generates plots based on *domain-specific* metrics
- Not standardized across packages

Comparison

Code Package	Simulators	Algorithms	Instrumentation
MuJoCo	✓	✗	✗
Arcade Learning Environment	✓	✗	✗
RL Berry	✓	✓	✓
ORGym	✓	✓	✗
Park	✓	✗	✗
BSuite	✓	✗	✓
MARO	✓	✓	✓
ORSuite	✓	✓	✓
OpenAI Gym	✓	✗	✗

MuJoCo

Simulators

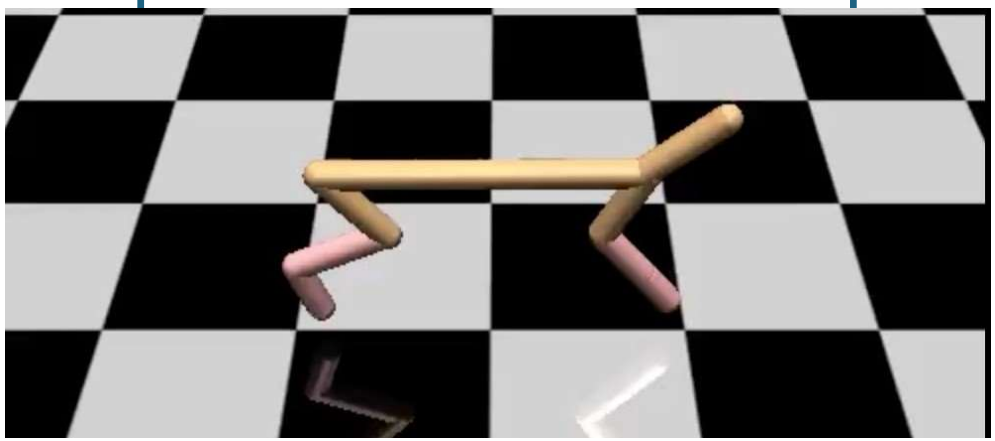
- Physics based robotics tasks
- Originally C/C++ with a Python API

Algorithms

- None included

Instrumentation

- None included



MuJoCo

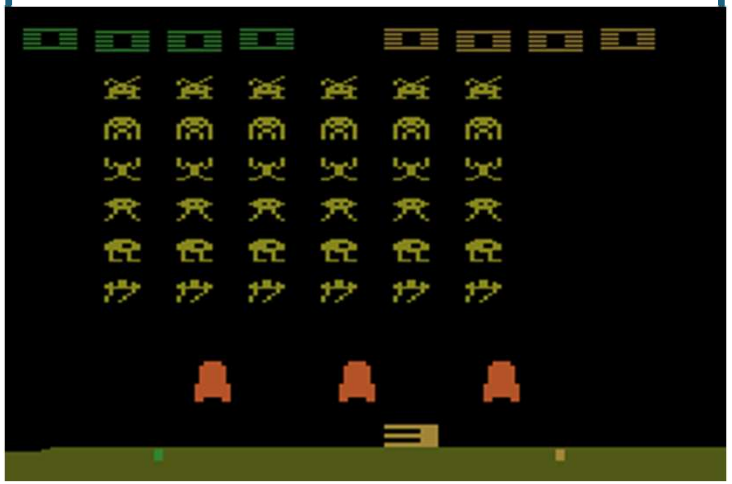
Arcade Learning Environment

Simulators

- 50 games from Atari 2600
- Uses OpenAI Gym API framework

Algorithms

- None included



Instrumentation

- None included

RL Berry

Simulators

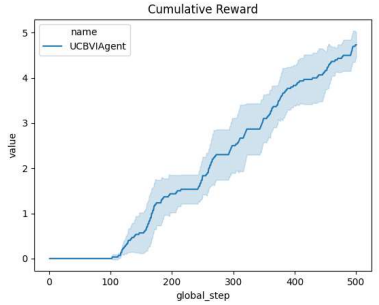
- Simple “theoretical” environments and OpenAI Gym API:
  - Chain Environment
  - GridWorld
  - MountainCar

Algorithms

- Several theoretical algorithms included:
  - Value Iteration
  - UCBVI
  - KernelUCBVI
  - AdaMB / AdaQN
  - A2C
  - DQN
  - REINFORCE

Instrumentation

- Beginner friendly learning framework, *model.fit()*
- Hyperparameter tuning
- Limited metrics + benchmarks



rlberry

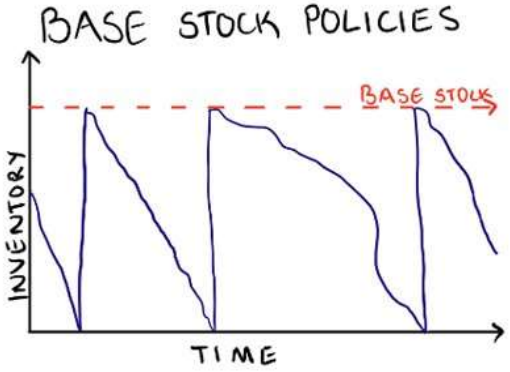
ORGym

Simulators

- Simplified OR Models
- Uses OpenAI Gym API
  - Knapsack
  - BinPacking
  - Newsvendor
  - Inventory Management
  - TSP
  - Portfolio Optimization

Algorithms

- None included



Instrumentation

- None included, examples use RLLib and Ray

Park

Simulators

- Systems models:
- Uses OpenAI Gym API
  - Adaptive video streaming
  - Spark cluster job scheduling
  - SQL Query Optimization
  - Circuit Design
  - Switch Scheduling
  - Server load balancing

Algorithms

- None included

Instrumentation

- None included

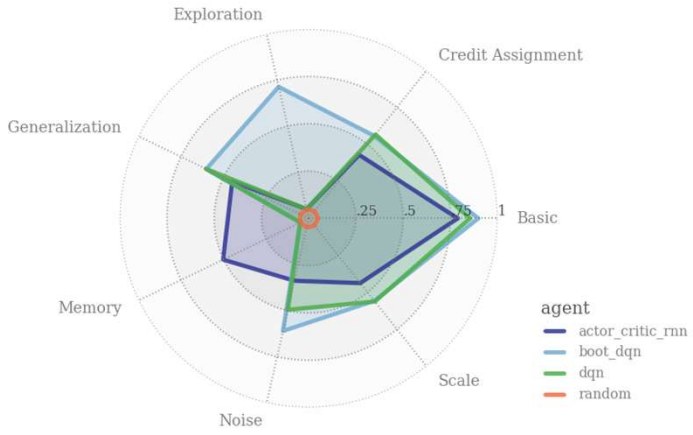
BSuite

Simulators

- Uses OpenAI Gym API
- “Theoretical” algorithms, configurable to test level of difficulty across different metrics

Algorithms

- None included



Instrumentation

- Yes – visualizations and PDF companion documents

MARO

Simulators

- Does NOT use OpenAI Gym API

- CitiBike Management
- VM Allocation
- Container Inventory Management

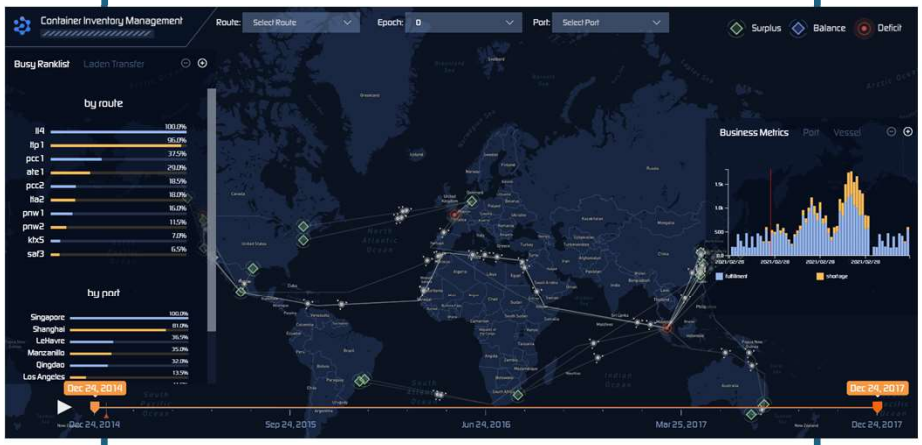
- Uses real-world traces

Algorithms

- Framework provided for implementing DeepRL algorithms

Instrumentation

- Framework provided for running experiments, not automated report generation



MARO

ORSuite

Simulators

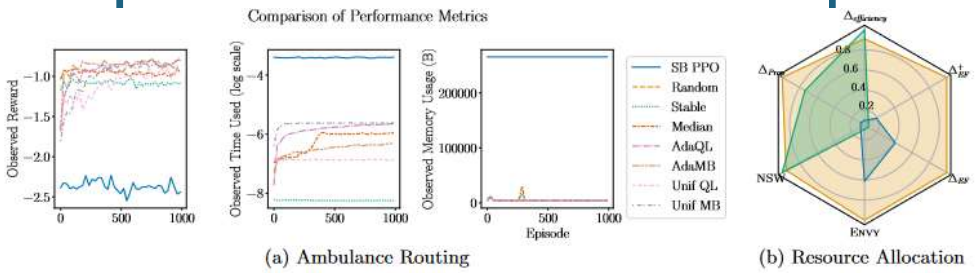
- OR models:
- Uses OpenAI Gym API
  - Inventory Management
  - Ridesharing Systems
  - Ambulance Routing
  - "Oil Problem" (aka continuous grid-world)
  - Vaccine Allocation
  - Revenue Management

Algorithms

- Environment specific algorithms included
- Wrappers for stablebaselines DeepRL algorithms

Instrumentation

- Domain specific metrics
- Ability to run experiments for custom and DeepRL algorithms
- Automatic plots + radar plots



(a) Ambulance Routing (b) Resource Allocation

ORSuite

OpenAI Gym

Simulators

- Develops OpenAI Gym API
- Wrapper for MuJoCo
  - GridWorld
  - Cartpole
  - Mountain Car

Algorithms

- None included
- Compatible with many existing DeepRL algorithmic software
  - RLLib with Ray
  - Stablebaselines
  - Tianshou
  - RLGarage

Instrumentation

- None included

Comparison

Code Package	Simulators	Algorithms	Instrumentation
MuJoCo	✓	✗	✗
Arcade Learning Environment	✓	✗	✗
RL Berry	✓	✓	✓
ORGym	✓	✓	✗
Park	✓	✗	✗
BSuite	✓	✗	✓
MARO	✓	✓	✓
ORSuite	✓	✓	✓
OpenAI Gym	✓	✗	✗

Conclusions

Simulators

- Typically use OpenAI Gym API for Markovian Settings
- MARO: “Business Engine” structure, more common for logged business data

Algorithms

- Compatible with many existing DeepRL software
  - RLLib with Ray
  - Stablebaselines
  - Tianshou
  - RLGarage

(more on this day 3+4)

Instrumentation

- RLLib, ORSuite, BSuite

Simulation Packages

Sean Sinclair,
Cornell University

MARO

ORSuite

Gym

rlberry

MuJoCo

Plan for Today

MDP Basics

- Basic framework for Markov Decision Processes
- Tabular RL Algorithms with policy iteration + value iteration
- DeepRL algorithms (and their "tabular" counterparts)

Simulation Packages

- OpenAI Framework for simulation design
- Existing packages and code-bases for RL algorithm development

Simulation Implementation

- Developed simulator for problem using OpenAI Gym API

Tabular RL Algorithms

- Implement basic tabular RL algorithms to understand key algorithmic design aspects of *value estimates + value iteration*, *policy iteration*

References

[MuJoCo] https://mujoco.org/
[ArcadeLearningEnvironment] https://github.com/mgbellemare/Arcade-Learning-Environment
[RLBerry] https://github.com/rlberry-py/rlberry
[ORGym] https://github.com/hubbs5/or-gym
[Park] https://github.com/park-project/park
[BSuite] https://github.com/deepmind/bsuite
[MARO] https://github.com/microsoft/maro
[ORSuite] https://github.com/cornell-orie/ORSuite
[OpenAI Gym] https://www.gymlibrary.ml/
[Tianshou] https://github.com/thu-ml/tianshou
[RLLib] https://docs.ray.io/en/latest/rllib/index.html
[StableBaselines] https://stable-baselines.readthedocs.io/en/master/
[RLGarage] https://github.com/rlworkgroup/garage