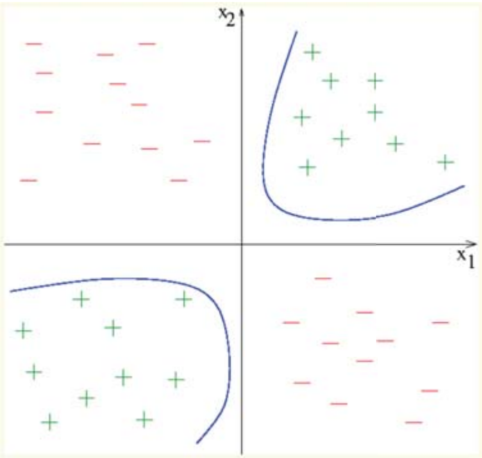
Reinforcement Learning for Electronic Design Automation: Successes and Opportunities

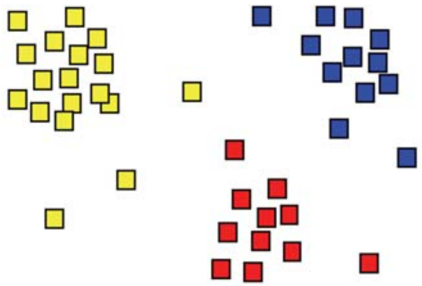
Professor Matthew E. Taylor (Matt)
March 22, 2021

Machine Learning (ML)

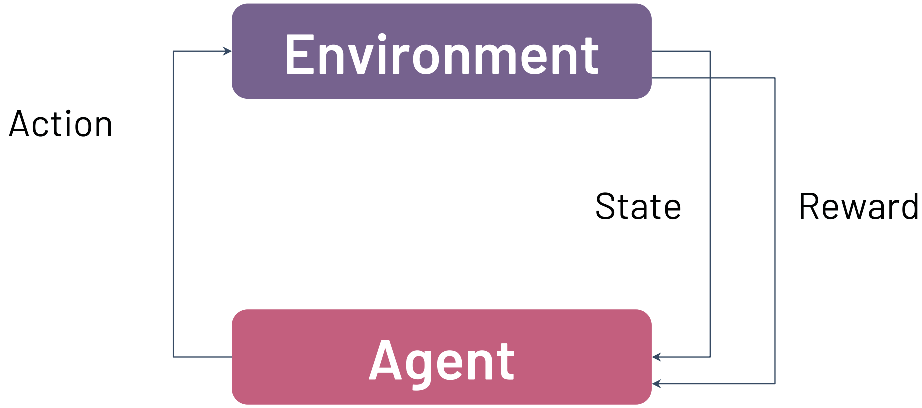
Supervised

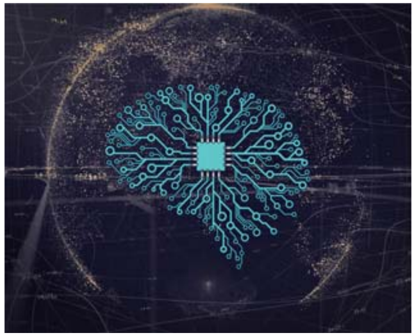


Unsupervised



Reinforcement Learning





AI Index 2019 Report

1998 – 2018: # of peer-reviewed AI papers grown by 300%
2014 - 2018: North America accounts for >60% of global AI patent activity

AI is the new Electricity
Data is the new Oil

Started in 1950s… Why now?

More compute
Better algos
Wider adoption & business understanding
Available talent & more accessible approaches

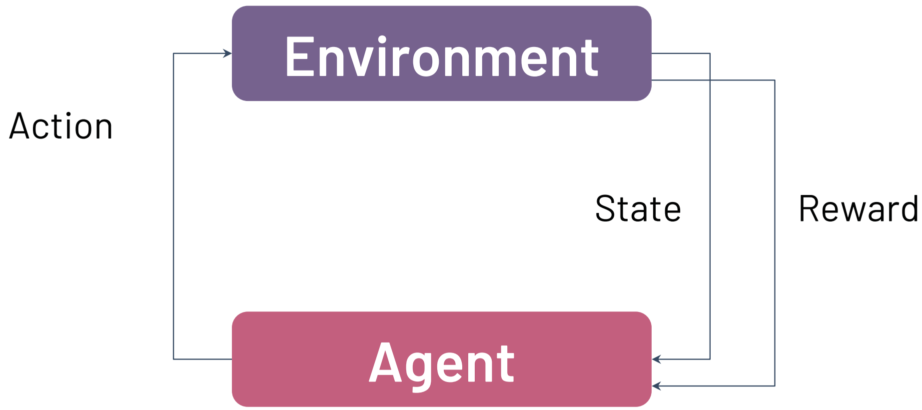
→ Automation
→ Optimization
→ New business processes

Reinforcement Learning (RL)

No labels: agent never told right or wrong

Agent interacts with environment (simulator or real world)

Typically can gather data, possibly at cost, by interacting with environment

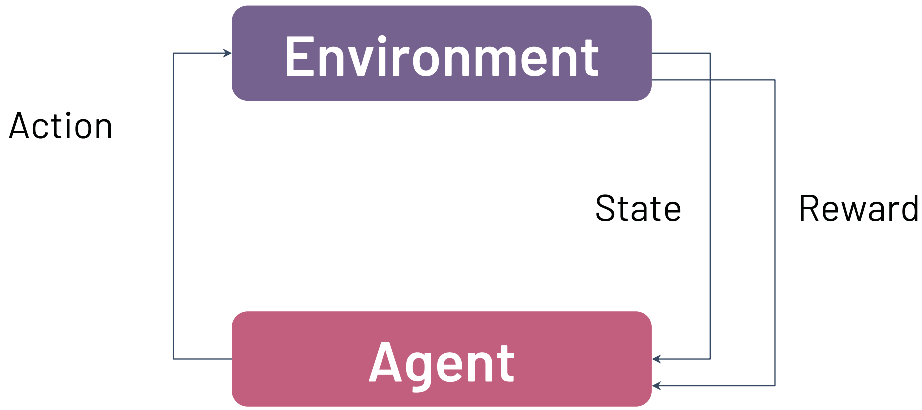


Reinforcement Learning (RL)

The agent typically learns via exploring vs. exploiting

Possible goals include

- Automation
- Improvement
- Enabling novel processes



RL Applications

(Un)Supervised learning performs well for many real-world applications

Dota

Robotics

American Options Exercise Policy

Stock Trading

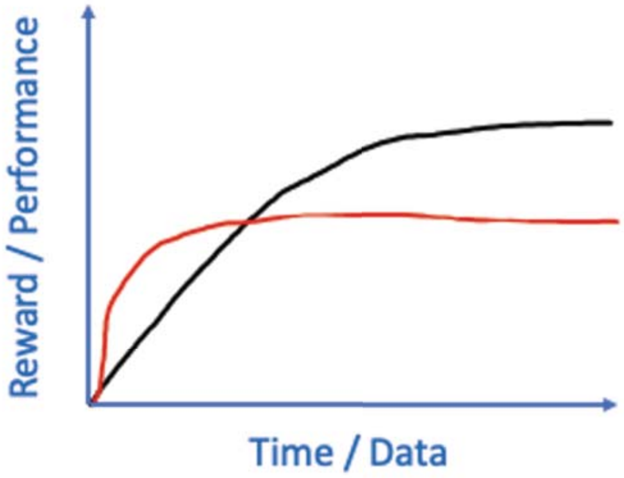
AlphaGO

Data Center Cooling

RL Goals

Learn to maximize real-valued reward signal (ideally)

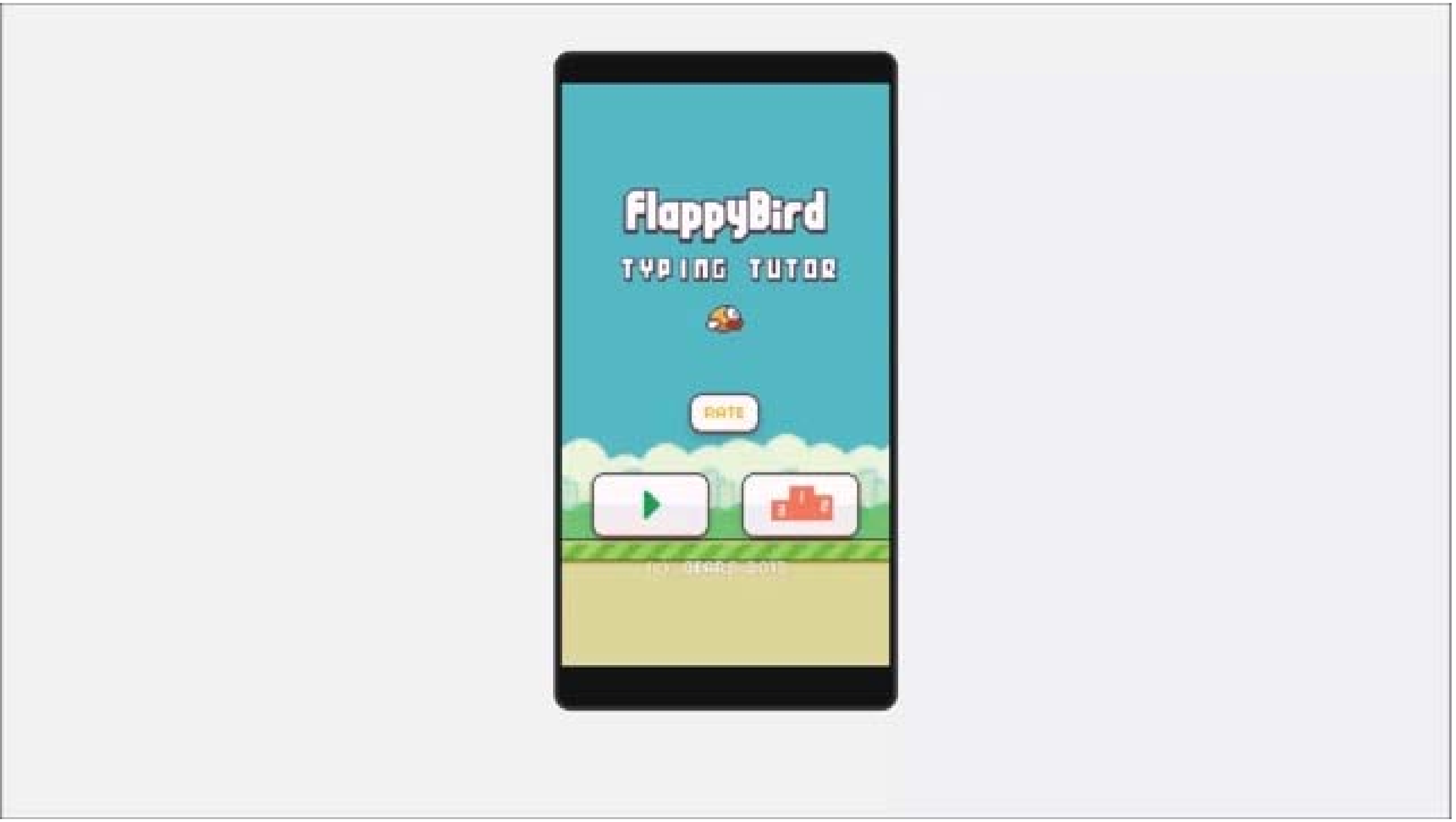
- With maximal final performance
- With little data
- Reducing human effort
- Discovering novel solutions
- Handling non-stationary environments



Outline

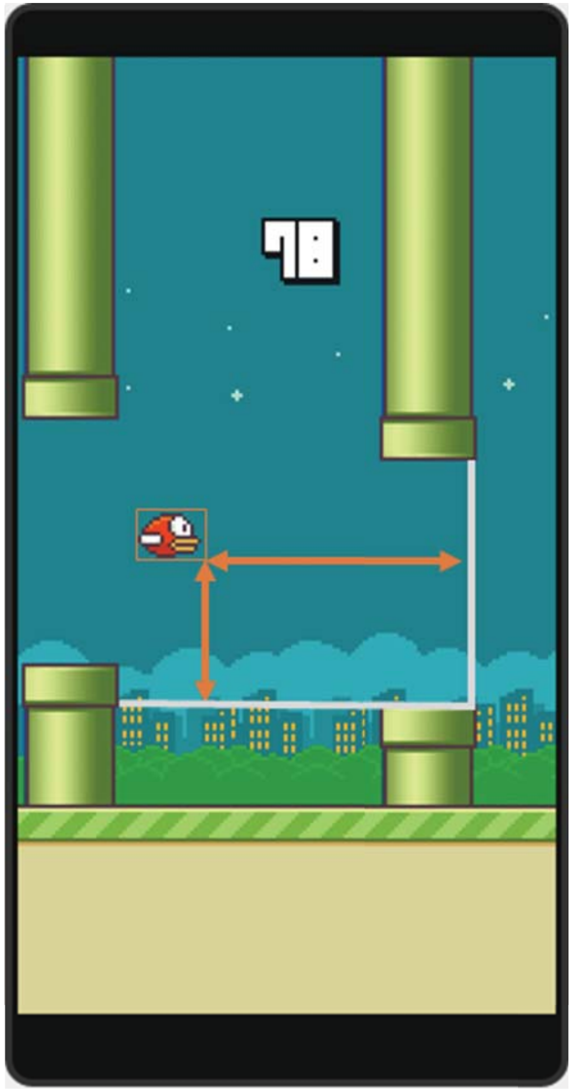
1. Background on RL
2. Examples of RL
3. Next Steps

https://www.flickr.com/photos/



- 2-3 days of development
- 2013 release, 2014 reportedly making $50k/day
- Then, removed because “too addictive”

https://www.youtube.com/watch?v=0Jw4HTWvGdY http://sarvagyavaish.github.io/FlappyBirdRL/



Example 1: Flappy Bird

Transition function: controlled by game

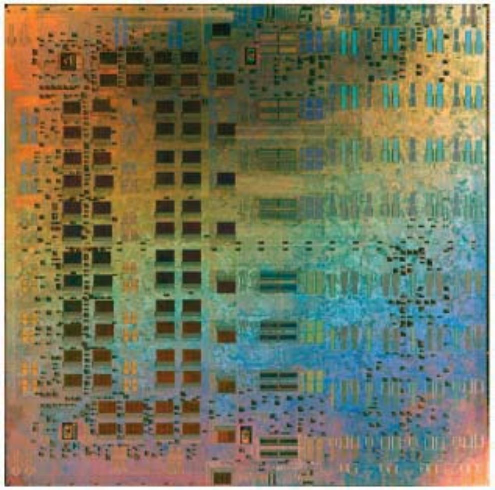
Action?

Reward?

State representation?

Example 2: Compiler Optimization

PACT-08

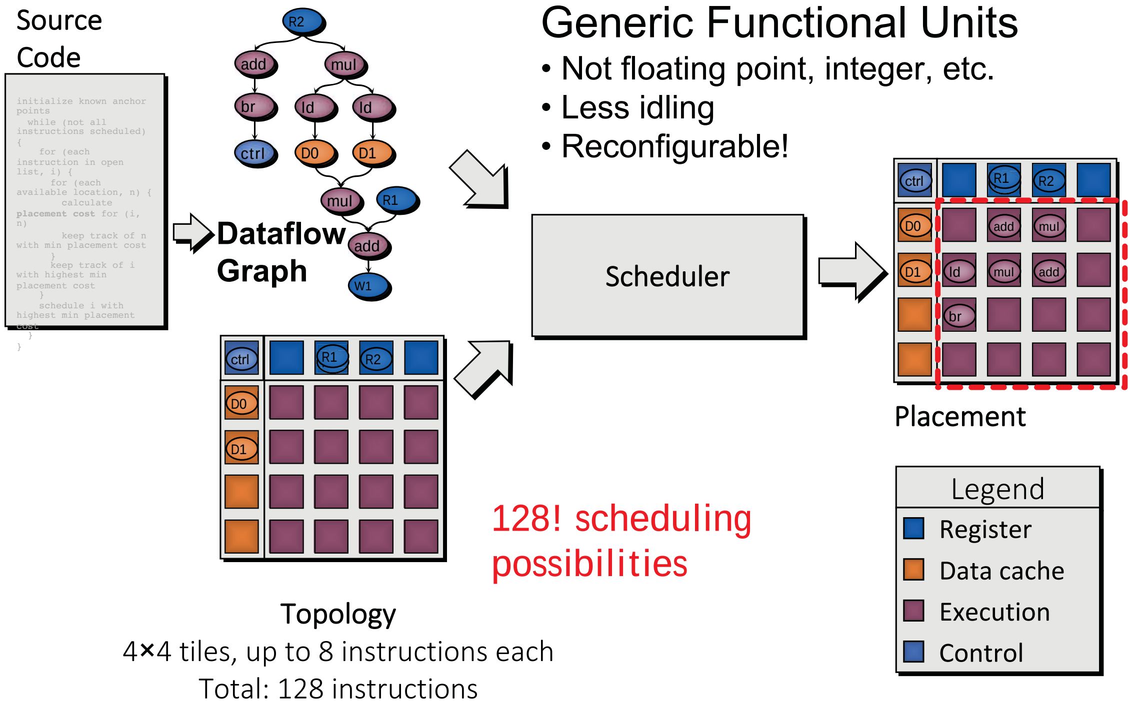


TRIPS: Tera-op, Reliable, Intelligently adaptive Processing System

SPS scheduler: 2006

UT-Austin:
Kathryn S. McKinley & Doug Berger

Scheduling Overview

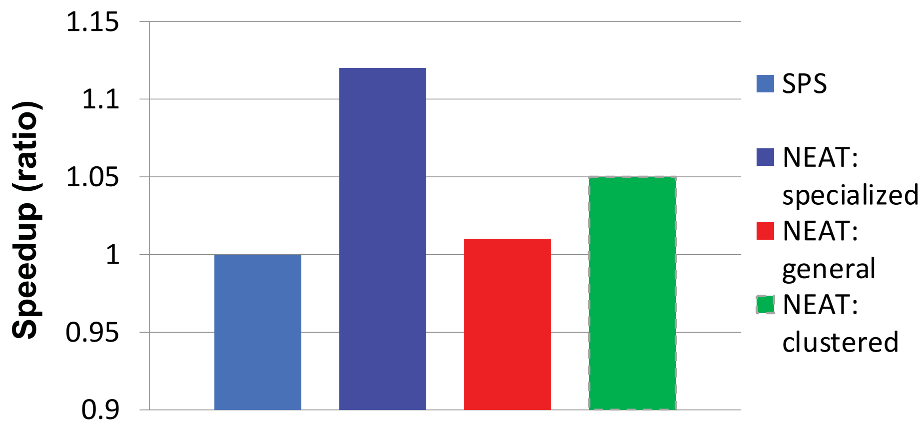


Example 2: Compiler Optimization

PACT-08

State: 11 features based on current instruction & already placed
Action: Place an instruction
Reward: 0 until all instructions placed, then, what's the speedup?

Heuristics → Learned scheduler heuristics
Per benchmark or general
47 small benchmarks



Example 3: Water Treatment

Developed @ UofA

ISL Adapt, UofA, and Amii

No ground truth
Raw water from North Saskatchewan River
State: Sensors added to filtration plant
Actions: Changes like chemicals, backwash cleaning, etc.
Reward: Environmental and fiscal benefits

bit.ly/3ouscLO

Water Treatment: Drayton Valley

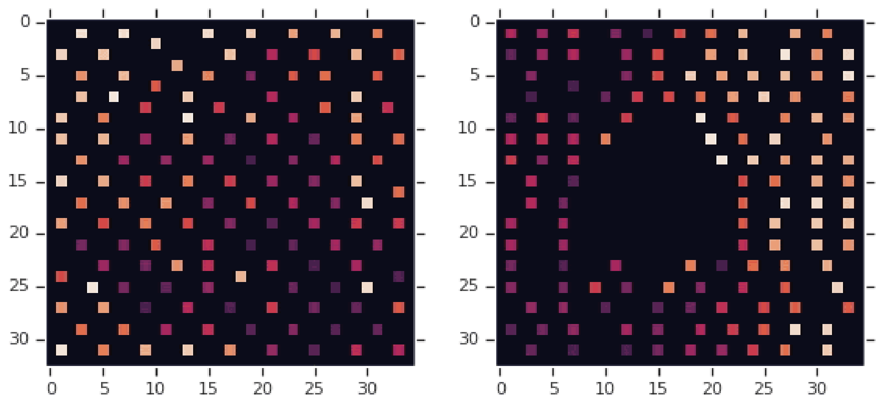
Example 4: Google's Chip Placement with Deep RL

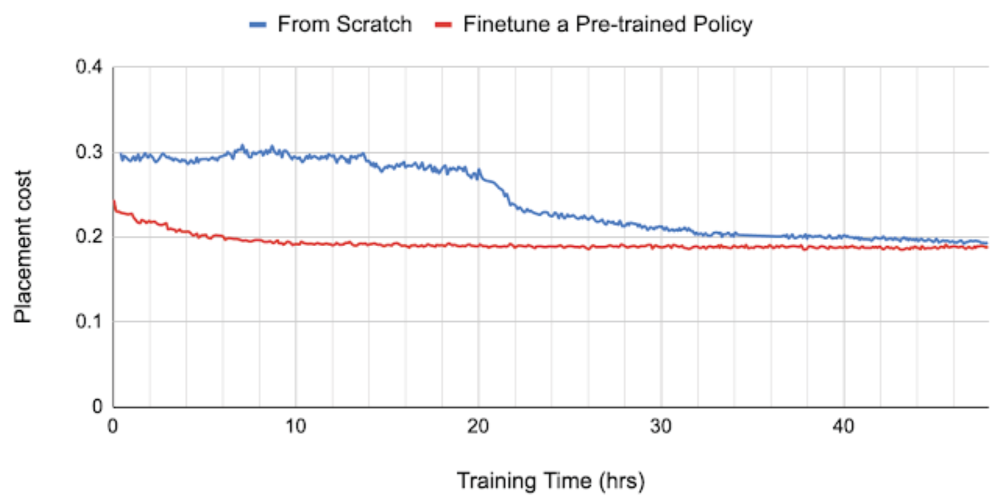
States: Every possible partial placement of netlist onto chip canvas
Actions: Place current macro at any location on discrete canvas space

- Don't violate hard constraints

Reward: 0 for all actions except last action

- Negative weighted sum of proxy wirelength & congestion





Outline

1. Background on RL
2. Examples of RL
3. Next Steps

RL Strengths

Agent can autonomously learn to maximize rewards

Programmer just specifies goals

Often much less work than directly programming

Can achieve superhuman performance

Can handle unanticipated changes in the environment

RL Weaknesses

Agent maximizes rewards whether it's what you actually wanted or not!

- Example: agent collects points in a game, rather than completing level

Can require lots of computation and/or interaction with the real world

- Interacting with world can have cost in time, money, wear, etc.

Solutions are often black box: explainability is not well understood (yet)

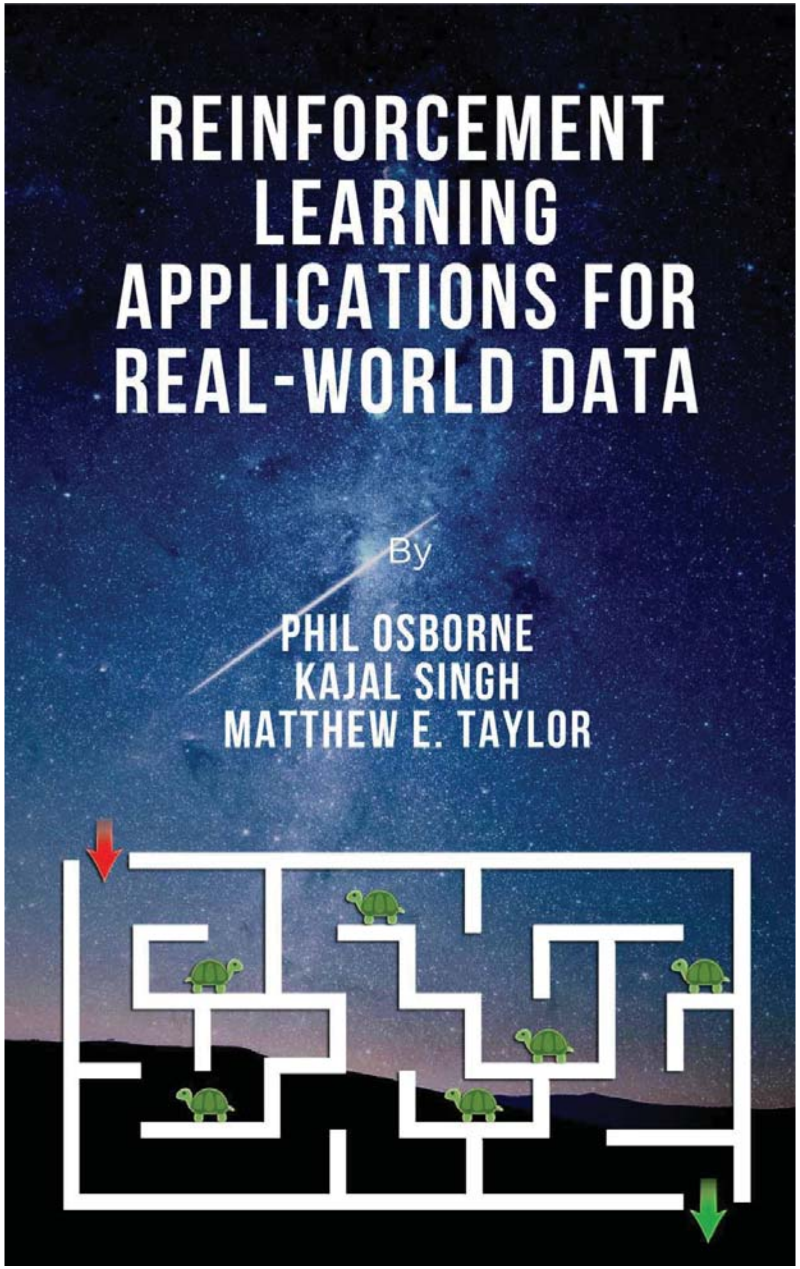
Initial performance could be very poor

Real World RL: What’s a *Good Problem*?

Sequential experimentation / process. Other methods can’t work
Full model isn’t known (dynamic programming/planning/optimization) or is too large

How costly is exploration?
How big is the state and action space?
Is the reward “obvious”? Is it dense?
Can you see the true state?

Do you have, or can you build a simulator?
Can you bootstrap off something/someone else?


REINFORCEMENT LEARNING APPLICATIONS FOR REAL-WORLD DATA
By
PHIL OSBORNE
KAJAL SINGH
MATTHEW E. TAYLOR

Thank you! Questions?

Matt Taylor:
IRLL.ca
Amii.ca

RL
- Coursera RL specialization from U Alberta
    - https://www.coursera.org/specializations/reinforcement-learning
- Udacity class from Georgia Tech
    - https://www.udacity.com/course/reinforcement-learning--ud600
- THE book on RL (Sutton & Barto, 2018)
    - http://www.incompleteideas.net/book/the-book-2nd.html
- Csaba Szepesvári: Algorithms in Reinforcement Learning
    - https://sites.ualberta.ca/~szepesva/rlbook.html

Deep RL
- Class on YouTube from UCL/Deepmind
    - https://www.youtube.com/playlist?list=PLqYmG7hTraZDNJre23vqCGIVpfZ_K2RZs
- OpenAI Spinning Up in Deep RL
    - https://spinningup.openai.com/en/latest/