CSCI E-82a
Probabilistic Programming and AI
Lecture 10
Introduction to Reinforcement Learning

Steve Elston

HARVARD
Extension School

Introduction to Reinforcement Learning

- Why is reinforcement learning exciting?
- What is reinforcement learning?
- Reward functions

Why is Reinforcement Learning Exciting?

- Difficult robotics tasks
  - Walking robot
  - Drone flight control
  - Navigation
- Complex control problems
  - Control smart power grids
  - Allocate server resources
  - Optimize elevator availability
- Play games at super-human level
  - Backgammon
  - Go
  - Atari
- Google Translate???? – see Wu, et. al., 2016 https://arxiv.org/pdf/1609.08144.pdf%20(7.pdf
- Many more…….

Why is Reinforcement Learning Exciting?

Long history of research

- Theseus, Claud Shannon, 1952
- Analog reinforcement learning, Marvin Minsky, 1954
- Dynamic programming, Richard Bellman, 1957
- MENACE for tic-tac-toe, Donald Michie, 1961, 1962
- Generalized Reinforcement Learning, Harry Klopf, 1972
- Learning with critic, Bernard Widrow, et.al., 1973
- Q-learning, Chris Watkins, 1989
- TD Gammon, Gerald Tesauro, 1992

Why is Reinforcement Learning Exciting?

- Rapid advances in algorithms
  - Deep Q-Networks (DQN) only since 2013
- But there are **pitfalls**:
  - Learning can be slow
  - Gaining experience can be expensive
  - Unintended behaviors occur
- Many recent improvements in learning rate, reduce required experience – **improved data efficiency**
- Multiple agent methods – **complex tasks**

Why is Reinforcement Learning Exciting?

How useful is Reinforcement Learning in the real world?

- Playing games is relatively easy
  - Games have rules and no unexpected behavior
  - Can play simulated game many times
- Walking robot trained with RL, using simulation for experience
  - https://m.youtube.com/watch?v=_YrlR1iNVcQ
  - https://m.youtube.com/watch?v=yQMrrCiOZUQ
- But can an RL agent learn to open a door?
  - Learning mechanisms is clearly not like human
  - https://m.youtube.com/watch?v=ZhsEKTo7V04

Topic Overview

We will cover the following reinforcement learning topics

- Bandit models
- Monte Carlo RL
- Time difference algorithms
- Q-learning algorithms
- Function approximation and deep RL
- Actor-critic methods (time permitting)

What is Reinforcement Learning?

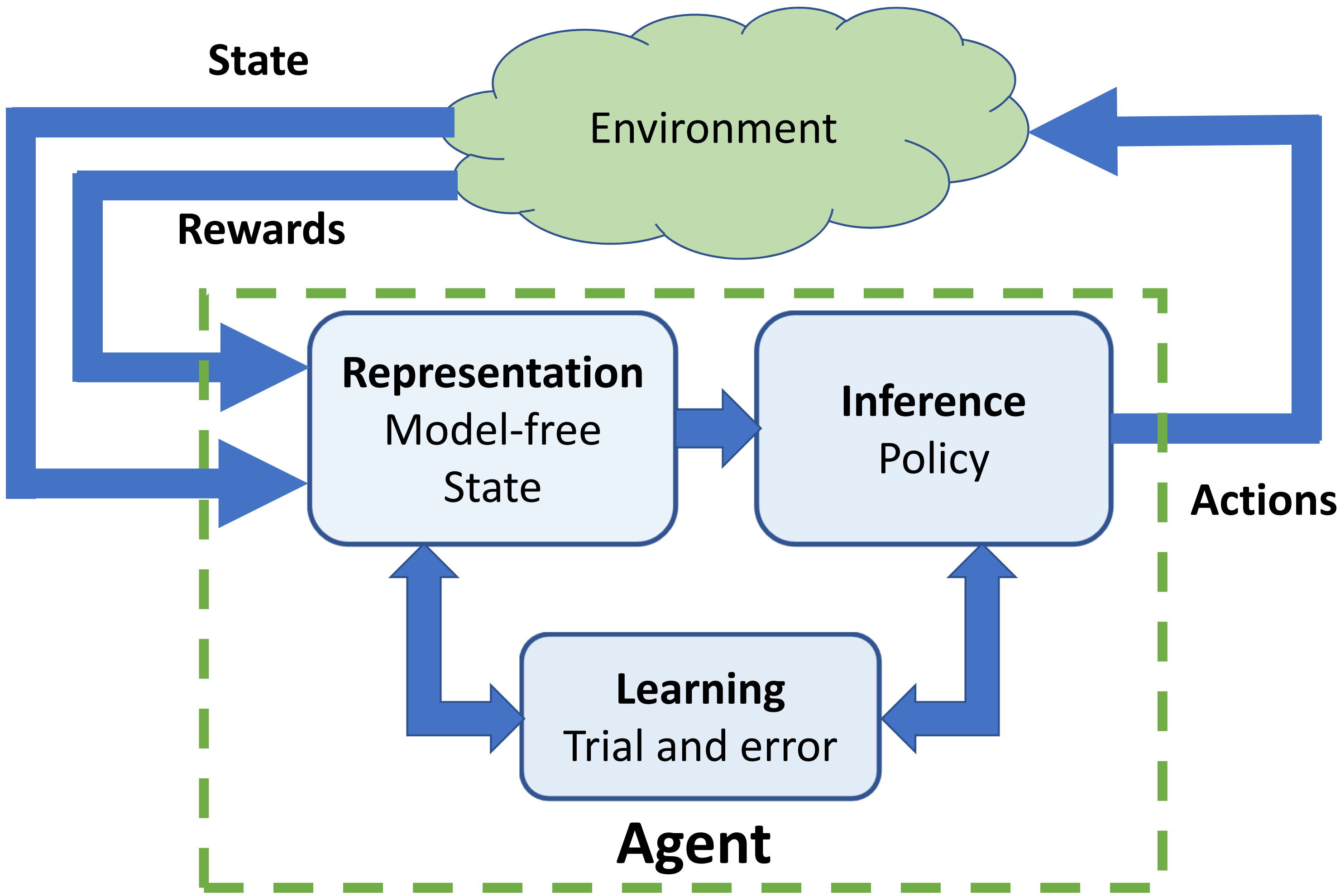
Model Type	Labeled Cases	Purpose	Metric
Supervised Machine Learning	Yes	Make Predictions	Error
Unsupervised Machine Learning	No	Find Structure	Error
Reinforcement Learning	No	Learn policy	Cumulative reward

What is Reinforcement Learning?

Key differences with other ML methods:

- RL agent learns by **trial and error**!
- RL agent has no supervisor, only **reward signal**
- Cumulative reward feedback is **delayed**
- Agent **learns policy** for a given **task**
- Policy determines **actions**, given state
- Optimal **policy maximizes cumulative reward** or **utility**
- Time matters; **sequential, non-iid data**

The Reinforcement Learning Agent



What is Reinforcement Learning?

- Reinforcement learning **agent operates sequentially** over time steps:
  - From **state**, s_t
  - Executes **action**, a_t
  - Receives scalar **reward**, r_t
  - Receives **observations**, o_t, and **updates state**, s_{t+1}
- In response, the **environment**:
  - Receives and executes **action**, a_t
  - Emits **observations**, o_t
  - Emits reward, r_t

What is Reinforcement Learning?

- Agent **learns from experience**
- **State** is the history of the actions, rewards, observations

  $S_t = (a_{t-n}, r_{t-n}, o_{t-n},\ldots., a_{t-1}, r_{t-1}, o_{t-1}, a_t, r_t, o_t,)$
- Agent's **actions affect subsequent data**
- Time matters; **sequential process, non-iid data**
- State is affected by actions

Reward Functions

- A **good reward function** is key to success
- Reward function must be specific to a **task**
- Good reward function must reflect the goal
- Good reward function should be understandable and simple
- Poor reward function can lead to unexpected results

Reward Functions

Properties of reward functions, R_t:

- Reward is a **scalar feedback signal**
- Reward depends on agent's action
- Measures agent's progress at time t
- Time maters, **agent executes actions sequentially**
- **Non-instantaneous feedback**: non-zero reward may be delayed

Reward Functions

- Reward function examples
- Agent plays a game:

  R(t) = +1 for win; -1 for loss

  Delayed reward; only at end of game

  No path penalty

Reward Functions

- Reward function examples
- Agent navigates robot to goal by shortest path:

  R(t) = -1 for step; +10 for goal

  Penalize for extra steps
- Poor reward function:

  R(t) = +10 for goal

  No penalty for long path

Reward Functions

- Reward function examples
- Agent directs walking robot:

  R(t) = +1 for step; -10 for falling

  Discourages falling
- Poor reward function:

  R(t) = +1 for step; +10 for getting up

  Falling increases cumulative reward!