Module 11
Introduction to Reinforcement Learning

CS 886 Sequential Decision Making and Reinforcement Learning

University of Waterloo

Machine Learning

- Supervised Learning
  - Teacher tells learner what to remember
- Reinforcement Learning
  - Environment provides hints to learner
- Unsupervised Learning
  - Learner discovers on its own

Animal Psychology

- Negative reinforcements:
  - Pain and hunger
- Positive reinforcements:
  - Pleasure and food
- Reinforcements used to train animals

- Let's do the same with computers!

RL Examples

- Game playing (backgammon, solitaire)
- Operations research (pricing, vehicule routing)
- Elevator scheduling
- Helicopter control
- Spoken dialog systems

Reinforcement Learning

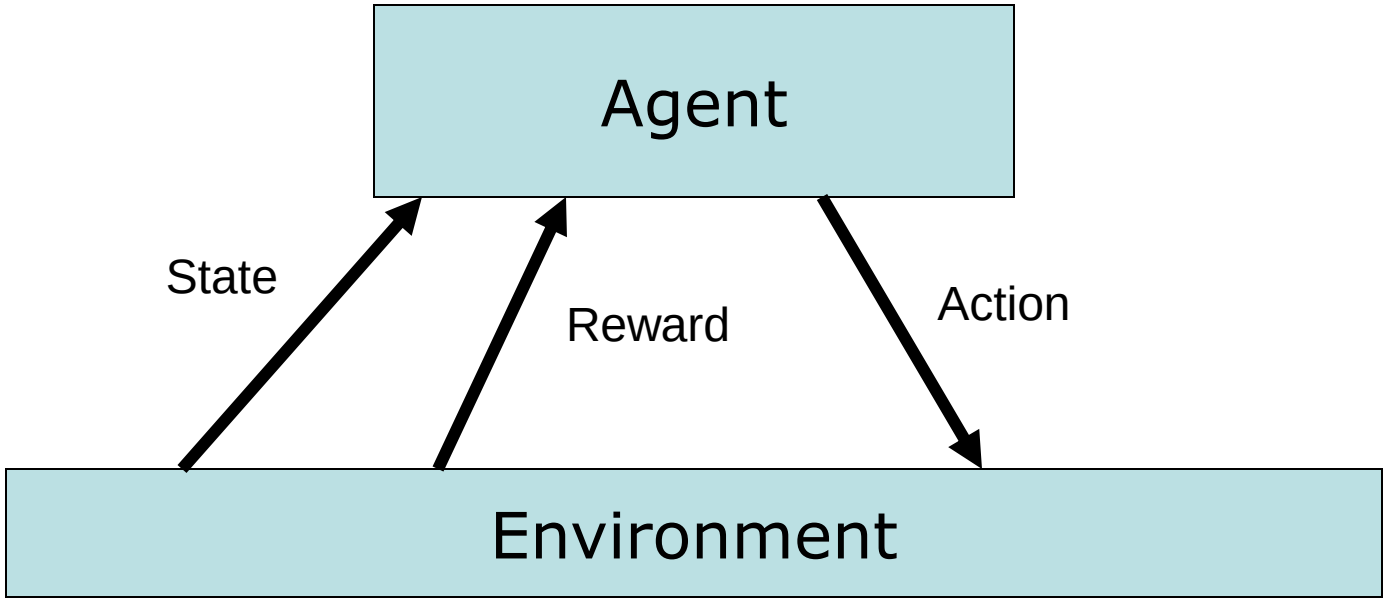
- Definition:
  - Markov decision process with unknown transition and reward models

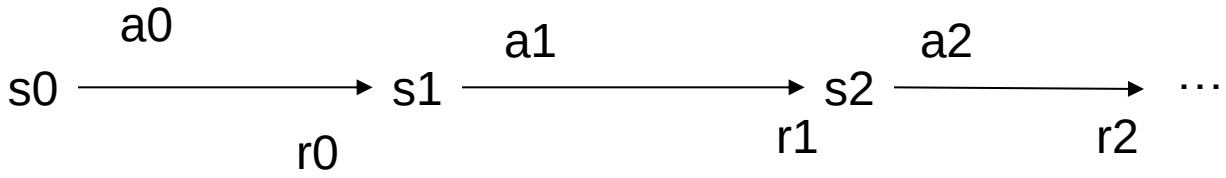
- Set of states S
- Set of actions A
  - Actions may be stochastic
- Set of reinforcement signals (rewards)
  - Rewards may be delayed

Policy optimization

- Markov Decision Process:
  - Find optimal policy given transition and reward model
  - Execute policy found
- Reinforcement learning:
  - Learn an optimal policy while interacting with the environment

Reinforcement Learning Problem



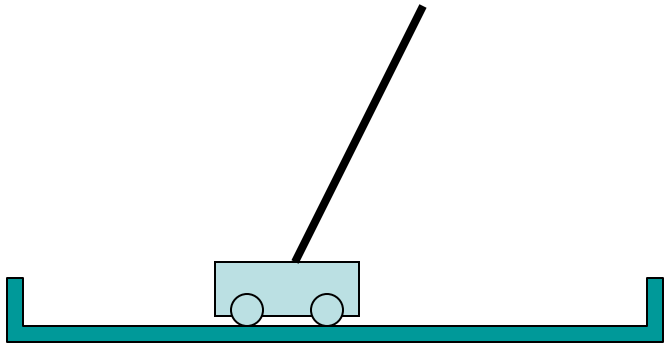


Goal: Learn to choose actions that maximize

$$r_0 + \gamma r_1 + \gamma^2 r_2 + \gamma^3 r_3 + \cdots$$

Example: Inverted Pendulum

- State: $x(t), x'(t), \theta(t), \theta'(t)$
- Action: Force F
- Reward: 1 for any step where pole balanced



Problem: Find $\pi: S \rightarrow A$ that maximizes rewards

Types of RL

- Passive vs Active learning
  - Passive learning: the agent executes a fixed policy and tries to evaluate it
  - Active learning: the agent updates its policy as it learns

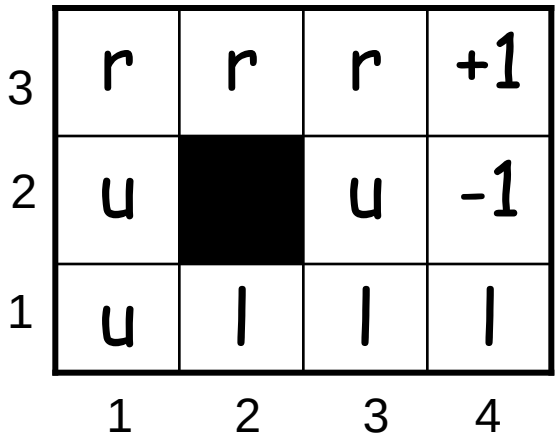
- Model based vs model free
  - Model-based: learn transition and reward model and use it to determine optimal policy
  - Model free: derive optimal policy without learning the model

Passive Learning

- Transition and reward model known:
  - Evaluate π:
    $V^{\pi}(s) = R\big(s, \pi(s)\big) + \gamma \sum_{s'} \Pr(s'|s, \pi(s))\, V^{\pi}(s') \;\; \forall s$

- Transition and reward model unknown:
  - Estimate value of policy as agent executes policy:
    $V^{\pi}(s) = E_{\pi}[\sum_t \gamma^t R(s_t, \pi(s_t))]$
  - Model based vs model free

Passive learning



$\gamma = 1$

Reward is -0.04 for non-terminal states

Do not know the transition probabilities

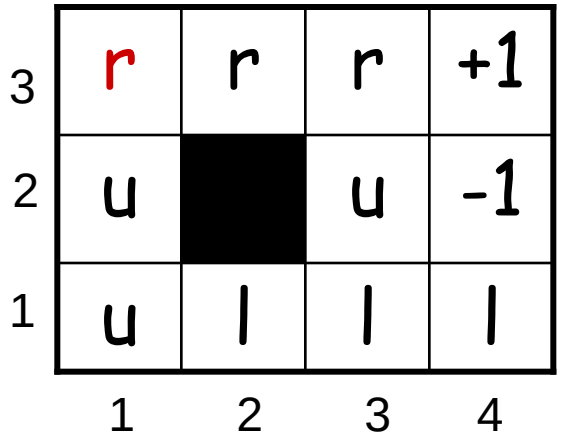
(1,1)→ (1,2)→ (1,3)→ (1,2)→ (1,3)→ (2,3)→ (3,3)→ $(4,3)_{+1}$
(1,1)→ (1,2)→ (1,3)→ (2,3)→ (3,3)→ (3,2)→ (3,3)→ $(4,3)_{+1}$
(1,1)→ (2,1)→ (3,1)→ (3,2)→ $(4,2)_{-1}$

What is the value $V(s)$ of being in state s?

Passive ADP

- Adaptive dynamic programming (ADP)
  - Model-based
  - Learn transition probabilities and rewards from observations
  - Then update the values of the states

ADP Example



$\gamma = 1$

Reward is -0.04 for non-terminal states

We need to learn all the transition probabilities!

(1,1)→ (1,2)→ **(1,3)**→ (1,2)→ **(1,3)**→ (2,3)→ (3,3)→ $(4,3)_{+1}$
(1,1)→ (1,2)→ **(1,3)**→ (2,3)→ (3,3)→ (3,2)→ (3,3)→ $(4,3)_{+1}$
(1,1)→ (2,1)→ (3,1)→ (3,2)→ $(4,2)_{-1}$

P((2,3)|(1,3),r) =2/3
P((1,2)|(1,3),r) =1/3

} Use this information in

$$V^{\pi}(s) = R\big(s,\pi(s)\big) + \gamma \sum_{s'} \Pr(s'|s,\pi(s))\, V^{\pi}(s')$$

Passive ADP

PassiveADP(π)

Repeat

Execute $\pi(s)$

Observe s' and r

Update counts: $n(s) \leftarrow n(s) + 1,\ n(s,s') \leftarrow n(s,s') + 1$

Update transition: $\Pr(s'|s,\pi(s)) \leftarrow \frac{n(s,s')}{n(s)}\ \forall s'$

Update reward: $R(s,\pi(s)) \leftarrow \frac{r + (n(s)-1)R(s,\pi(s))}{n(s)}$

Solve: $V^{\pi}(s) = R(s,\pi(s)) + \gamma \sum_{s'} \Pr(s'|s,\pi(s))\, V^{\pi}(s')\ \forall s$

$s \leftarrow s'$

Until convergence of V^{π}

Return V^{π}

Passive TD

- Temporal difference (TD)
  - Model free

- At each time step
  - Observe: s, a, s', r
  - Update $V^{\pi}(s)$ after each move
  - $V^{\pi}(s) = V^{\pi}(s) + \alpha\,(R(s,\pi(s)) + \gamma\, V^{\pi}(s') - V^{\pi}(s))$

Learning rate

Temporal difference

TD Convergence

Theorem: If α is appropriately decreased with number of times a state is visited then $V^{\pi}(s)$ converges to correct value

- α must satisfy:
  - $\sum_t \alpha_t \rightarrow \infty$
  - $\sum_t (\alpha_t)^2 < \infty$
- Often $\alpha(s) = 1/n(s)$
  - Where $n(s)$ = # of times s is visited

Passive TD

PassiveTD(π, V^{π})

Repeat

Execute $\pi(s)$

Observe s' and r

Update counts: $n(s) \leftarrow n(s) + 1$

Learning rate: $\alpha \leftarrow 1/n(s)$

Update value: $V^{\pi}(s) \leftarrow V^{\pi}(s) + \alpha(r + \gamma V^{\pi}(s') - V^{\pi}(s))$

$s \leftarrow s'$

Until convergence of V^{π}

Return V^{π}

Comparison

- Model free approach:
  - Less computation per time step
- Model based approach:
  - Fewer time steps before convergence

Active Learning

- Ultimately, we are interested in improving π
- Transition and reward model known:

$$V^*(s) = \max_a R(s,a) + \gamma \sum_{s'} \Pr(s'|s,a)\, V^*(s')$$

- Transition and reward model unknown:
  - Improve policy as agent executes policy
  - Model based vs model free

Q-learning (aka active temporal difference)

- Q-function: $Q: S \times A \rightarrow \Re$
  - Value of state-action pair
  - Policy $\pi(s) = argmax_a\, Q(s,a)$ is the greedy policy w.r.t. Q

- Bellman's equation:

$$Q^*(s,a) = R(s,a) + \gamma \sum_{s'} \Pr(s'|s,a) \max_{a'} Q^*(s',a')$$

Q-Learning

Qlearning(s, Q^*)

Repeat

Select and execute a

Observe s' and r

Update counts: $n(s) \leftarrow n(s) + 1$

Learning rate: $\alpha \leftarrow 1/n(s)$

Update Q-value:

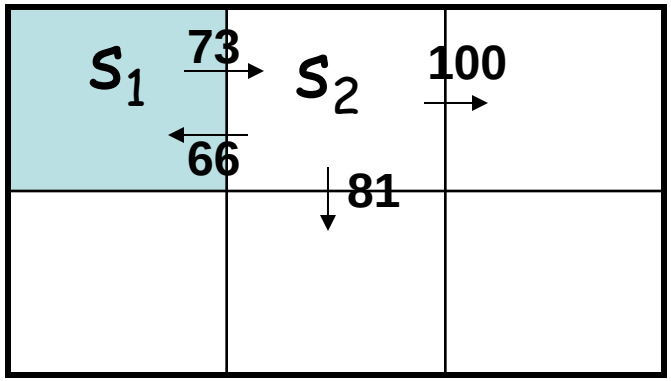
$$Q^*(s,a) \leftarrow Q^*(s,a) + \alpha\left(r + \gamma \max_{a'} Q^*(s',a') - Q^*(s,a)\right)$$

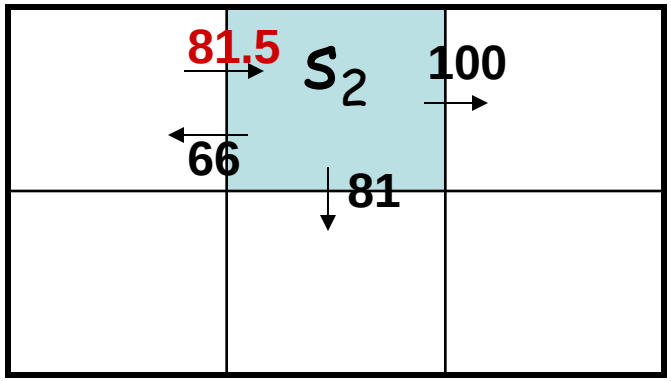
$s \leftarrow s'$

Until convergence of Q

Return Q

Q-learning example





$\gamma = 0.9, \quad \alpha = 0.5, \quad r = 0$ for non-terminal states

$$Q(s_1, right) = Q(s_1, right) + \alpha\left(r + \gamma \max_{a'} Q(s_2, a') - Q(s_1, right)\right)$$
$$= 73 + 0.5(0 + 0.9 \max\{66, 81, 100\} - 73)$$
$$= 73 + 0.5(17)$$
$$= 81.5$$

Q-Learning

Qlearning(s, Q^*)

Repeat

Select and execute a

Observe s' and r

Update counts: $n(s) \leftarrow n(s) + 1$

Learning rate: $\alpha \leftarrow 1/n(s)$

Update Q-value:

$$Q^*(s,a) \leftarrow Q^*(s,a) + \alpha\left(r + \gamma \max_{a'} Q^*(s',a') - Q^*(s,a)\right)$$

$s \leftarrow s'$

Until convergence of Q^*

Return Q^*

Exploration vs Exploitation

- If an agent always chooses the action with the highest value then it is exploiting
  - The learned model is not the real model
  - Leads to suboptimal results
- By taking random actions (pure exploration) an agent may learn the model
  - But what is the use of learning a complete model if parts of it are never used?
- Need a balance between exploitation and exploration

Common exploration methods

- ε-greedy:
  - With probability ϵ execute random action
  - Otherwise execute best action a^*

$$a^* = argmax_a \, Q(s,a)$$

- Boltzmann exploration

$$\Pr(a) = \frac{e^{\frac{Q(s,a)}{T}}}{\sum_a e^{\frac{Q(s,a)}{T}}}$$

Exploration and Q-learning

- Q-learning converges to optimal Q-values if
  - Every state is visited infinitely often (due to exploration)
  - The action selection becomes greedy as time approaches infinity
  - The learning rate α is decreased fast enough, but not too fast

Model-based Active RL

- Idea: at each step
  - Execute action
  - Observe resulting state and reward
  - Update model
  - Update policy π

Model-based Active RL

ModelBasedActiveRL(s)

Repeat

Select and execute a

Observe s' and r

Update counts: $n(s,a) \leftarrow n(s,a) + 1$,

$$n(s,a,s') \leftarrow n(s,a,s') + 1$$

Update transition: $\Pr(s'|s,\pi(s)) \leftarrow \frac{n(s,a,s')}{n(s,a)} \; \forall s'$

Update reward: $R(s,\pi(s)) \leftarrow \frac{r + (n(s,a)-1)R(s,a)}{n(s,a)}$

Solve: $V^*(s) = \max_a R(s,a) + \gamma \sum_{s'} \Pr(s'|s,a)\, V^*(s') \; \forall s$

$s \leftarrow s'$

Until convergence of V^*

Return V^*

Summary

- We can optimize a policy by RL when the transition and reward functions are unknown
- Comparison:
  - Model free: computationally cheaper
  - Model-based: faster convergence
- Active learning:
  - Exploration/exploitation dilemma