

Lecture 3a: Intro to RL, Q-learning

CS885 Reinforcement Learning

2022-09-16

Complementary readings: [SutBar] Sec. 5.1-5.3, 6.1-6.3, 6.5, [Sze] Sec. 3.1, 4.3, [SigBuf] Sec. 2.1-2.5, [RusNor] Sec. 21.1-21.3

Pascal Poupart
David R. Cheriton School of Computer Science



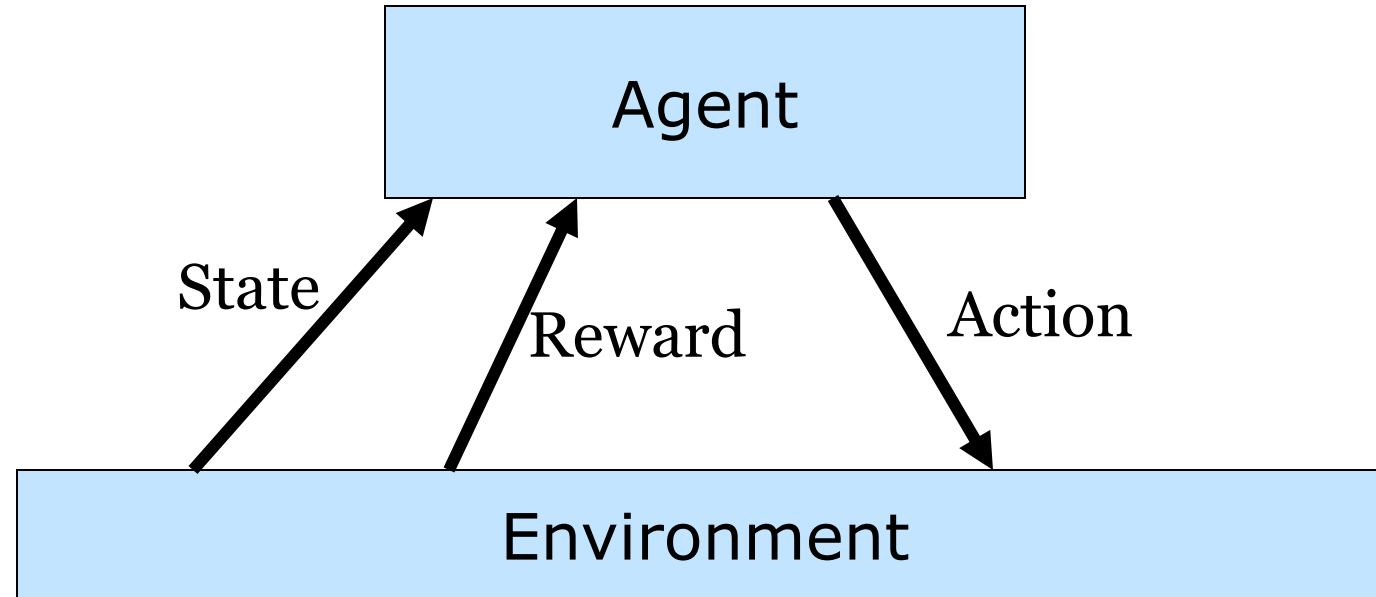
Outline

- Reinforcement Learning
 - Model-based RL, model-free RL
 - Value-based RL, policy-based RL, actor-critic
- Algorithms:
 - Monte-Carlo evaluation
 - Temporal Difference (TD) evaluation
 - Control: Q-learning

Recap: Markov Decision Process

- Formal Definition
 - States: $s \in S$
 - Actions: $a \in A$
 - Rewards: $r \in \mathbb{R}$
 - Transition model: $\Pr(s_t | s_{t-1}, a_{t-1})$
 - Reward model: $\Pr(r_t | s_t, a_t), R(s_t, a_t) = \sum_{r_t} r_t \Pr(r_t | s_t, a_t)$
 - Discount factor: $0 \leq \gamma \leq 1$
 - discounted: $\gamma < 1$ undiscounted: $\gamma = 1$
 - Horizon (i.e., # of time steps): h
 - Finite horizon: $h \in \mathbb{N}$ infinite horizon: $h = \infty$
- Goal: **find optimal policy** such that $\pi^* = \operatorname{argmax}_{\pi} \sum_{t=0}^h \gamma^t E_{\pi}[r_t]$

Reinforcement Learning Problem



Goal: Learn to choose actions that maximize rewards

Reinforcement Learning

- Formal Definition

- States: $s \in S$
- Actions: $a \in A$
- Rewards: $r \in \mathbb{R}$
- Transition model: $\Pr(s_t | s_{t-1}, a_{t-1})$
- Reward model: $\Pr(r_t | s_t, a_t)$
- Discount factor: $0 \leq \gamma \leq 1$
 - discounted: $\gamma < 1$ undiscounted: $\gamma = 1$
- Horizon (i.e., # of time steps): h
 - Finite horizon: $h \in \mathbb{N}$ infinite horizon: $h = \infty$

} unknown model

- Goal: **find optimal policy** such that $\pi^* = \operatorname{argmax}_{\pi} \sum_{t=0}^h \gamma^t E_{\pi}[r_t]$

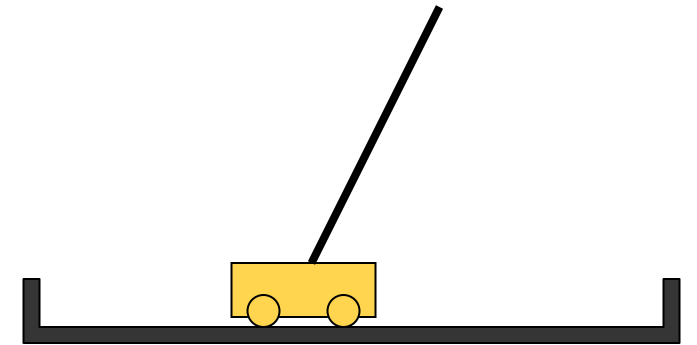
Policy Optimization

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

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



Important Components in RL

RL agents may or may not include the following components:

- **Model:** $\Pr(s' | s, a)$, $\Pr(r | s, a)$
 - Environment dynamics and rewards
- **Policy:** $\pi(s)$
 - Agent action choices
- **Value function:** $V(s)$
 - Expected total rewards of the agent policy

Categorizing RL agents

Value based

- No policy (implicit)
- Value function

Policy based

- Policy
- No value function

Actor critic

- Policy
- Value function

Model based

- Transition and reward model

Model free

- No transition and no reward model (implicit)

Online RL

- Learn by interacting with environment

Offline RL

- No environment
- Learn only from saved data

Toy Maze Example

3	r	r	r	+1
2	u		u	-1
1	u	l	l	l
	1	2	3	4

Start state: (1,1)

Terminal states: (4,2), (4,3)

No discount: $\gamma = 1$

Reward is -0.04 for non-terminal states

Four actions: up (u), left (l), right (r), down (d)

Do not know the transition probabilities

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

Model Free Evaluation

- Given policy π , estimate $V^\pi(s)$ without any transition or reward model

- **Monte Carlo** evaluation

$$V^\pi(s) = E_\pi[\sum_t \gamma^t r_t] \approx \frac{1}{n(s)} \sum_{k=1}^{n(s)} \left[\sum_t \gamma^t r_t^{(k)} \right] \quad (\text{sample approximation})$$

- **Temporal difference (TD)** evaluation

$$V^\pi(s) = E[r|s, \pi(s)] + \gamma \sum_{s'} \Pr(s'|s, \pi(s)) V^\pi(s') \approx r + \gamma V^\pi(s') \quad (\text{one sample approximation})$$

Monte Carlo Evaluation

- Let G_k be a one-trajectory Monte Carlo target

$$G_k = \sum_t \gamma^t r_t^{(k)}$$

- Approximate value function

$$\begin{aligned} V_n^\pi(s) &\approx \frac{1}{n(s)} \sum_{k=1}^{n(s)} G_k \\ &= \frac{1}{n(s)} \left(G_{n(s)} + \sum_{k=1}^{n(s)-1} G_k \right) \\ &= \frac{1}{n(s)} \left(G_{n(s)} + (n(s) - 1) V_{n-1}^\pi(s) \right) \\ &= V_{n-1}^\pi(s) + \frac{1}{n(s)} \left(G_{n(s)} - V_{n-1}^\pi(s) \right) \end{aligned}$$

- **Incremental update**

$$V_n^\pi(s) \leftarrow V_{n-1}^\pi(s) + \alpha_n \left(G_n - V_{n-1}^\pi(s) \right)$$

↙ learning rate $1/n(s)$

Temporal Difference Evaluation

- Approximate value function: $V^\pi(s) \approx r + \gamma V^\pi(s')$

- **Incremental update**

$$V_n^\pi(s) \leftarrow V_{n-1}^\pi(s) + \alpha_n (r + \gamma V_{n-1}^\pi(s') - V_{n-1}^\pi(s))$$

- **Theorem:** If α_n is appropriately decreased with # of times a state is visited then $V_n^\pi(s)$ converges to correct value.
 - **Sufficient conditions** for α_n : (1) $\sum_n \alpha_n \rightarrow \infty$ (2) $\sum_n (\alpha_n)^2 < \infty$
 - Often $\alpha_n(s) = 1/n(s)$ where $n(s) = \#$ of times s is visited

Temporal Difference (TD) Evaluation

$\text{TDEvaluation}(\pi, V^\pi)$

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

- Monte Carlo evaluation:
 - Unbiased estimate
 - High variance
 - Needs many trajectories
- Temporal difference evaluation:
 - Biased estimate
 - Lower variance
 - Needs less trajectories

Model Free Control

- Instead of evaluating the state value function, $V^\pi(s)$, evaluate the state-action value function, $Q^\pi(s, a)$

$Q^\pi(s, a)$: value of executing a followed by π

$$Q^\pi(s, a) = E[r|s, a] + \gamma \sum_{s'} Pr(s'|s, a)V^\pi(s')$$

- Greedy policy π' :

$$\pi'(s) = \operatorname{argmax}_a Q^\pi(s, a)$$

Bellman's Equation

- Optimal state value function $V^*(s)$

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

- Optimal state-action value function $Q^*(s, a)$

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

where $V^*(s) = \max_a Q^*(s, a)$
 $\pi^*(s) = \operatorname{argmax}_a Q^*(s, a)$

Monte Carlo Control

- Let G_k^a be a one-trajectory Monte Carlo target

$$G_k^a = \underbrace{r_0^{(k)}}_a + \underbrace{\sum_{t=1} \gamma^t r_t^{(k)}}_\pi$$

- Alternate between

- **Policy evaluation**

$$Q_n^\pi(s, a) \leftarrow Q_{n-1}^\pi(s, a) + \alpha_n (G_n^a - Q_{n-1}^\pi(s, a))$$

- **Policy improvement**

$$\pi'(s) \leftarrow \operatorname{argmax}_a Q^\pi(s, a)$$

Temporal Difference Control

- Approximate Q-function:

$$Q^*(s, a) = E[r|s, a] + \gamma \sum_{s'} \Pr(s'|s, a) \max_{a'} Q^*(s', a')$$
$$\approx r + \gamma \max_{a'} Q^*(s', a')$$

- **Incremental update**

$$Q_n^*(s, a) \leftarrow Q_{n-1}^*(s, a) + \alpha_n \left(r + \gamma \max_{a'} Q_{n-1}^*(s', a') - Q_{n-1}^*(s, a) \right)$$

Q-Learning

Qlearning(s, Q^*)

Repeat

Select and execute a

Observe s' and r

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

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

Update Q-value:

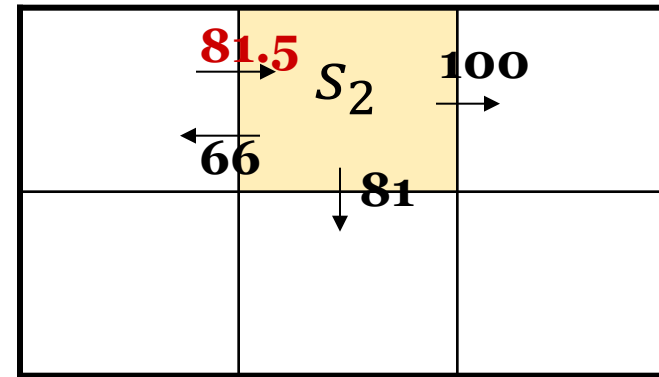
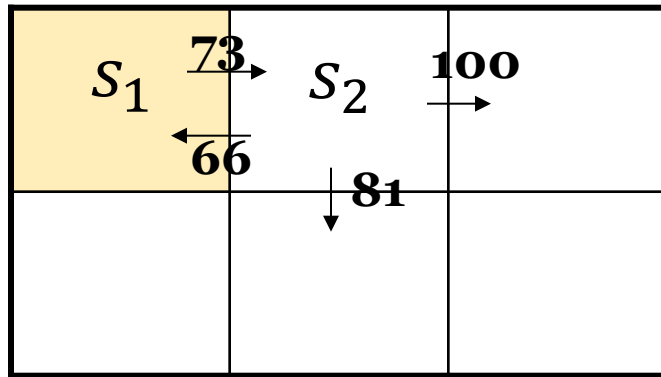
$$Q^*(s, a) \leftarrow Q^*(s, a) + \alpha \left(r + \gamma \max_{a'} Q^*(s', a') - Q^*(s, a) \right)$$

$s \leftarrow s'$

Until convergence of Q^*

Return Q^*

Q-learning Example



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

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

Q-Learning

Qlearning(s, Q^*)

Repeat

Select and execute a

Observe s' and r

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

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

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 agent always chooses action with highest value, then it is **exploiting**
 - The learned model is not accurate
 - Leads to suboptimal results
- By taking random actions (**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 $a^* = \operatorname{argmax}_a Q(s, a)$

- Boltzmann exploration

- Increasing temperature T increases stochasticity

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

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 (sufficient conditions for α):

$$(1) \sum_n \alpha_n \rightarrow \infty \quad (2) \sum_n (\alpha_n)^2 < \infty$$

Summary

- We can optimize a policy by RL when the transition and reward functions are unknown
- **Model free, value based agent:**
 - Monte Carlo learning (unbiased, but needs lots of data)
 - Temporal difference learning (low variance, less data)
- Active learning:
 - Exploration/exploitation dilemma