Introduction to Reinforcement Learning

서울시립대학교
UNIVERSITY OF SEOUL

Presentation: Baekryun Seong

2024.05.20

Table of Contents

In the last presentation, we studied the concept of value function:

State Value Function v^{π} is an expected return of a state.

$$v^{\pi}(s) = \mathbb{E}_{a\sim\pi}[G_t|s] = \mathbb{E}_{a\sim\pi}[R_{t+1} + \gamma G_{t+1}|s]$$

Action Value Function q^{π} is an expected return about an action.

$$q^{\pi}(s,a) = \mathbb{E}_{a\sim\pi}[G_t|s,a] = \mathbb{E}_{a\sim\pi}[R_{t+1} + \gamma G_{t+1}|s,a]$$

So, why should we define it?

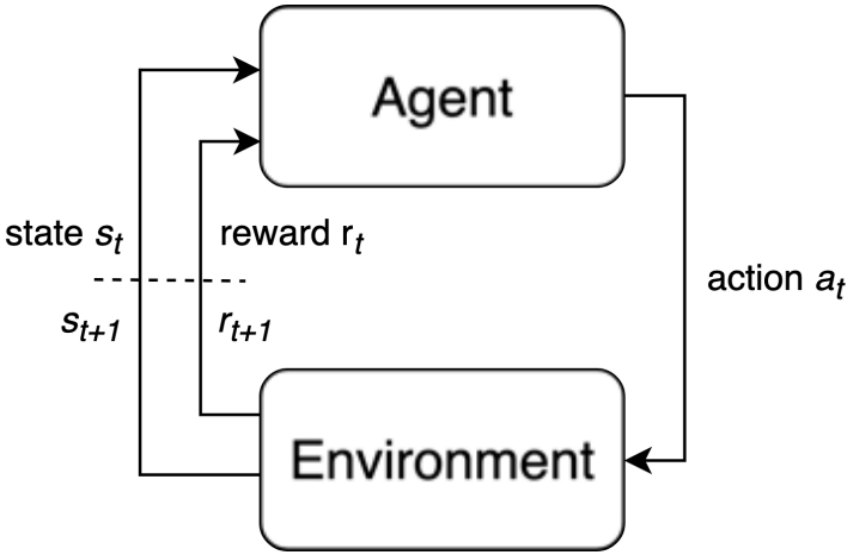


Figure 1.2 The reinforcement learning control loop

- RL Problems have an **objective**, which is the **sum of rewards** received by an agent.
- An agent uses the reward signals it receives to **reinforce good actions.**
- Remark: **Sum of rewards is expected return,** by definition of return. So expected return can be described in value function of a state or action.
- To summarize, **we use value functions to reinforce good actions.**

Graesser, L., & Keng, W. L. (2019). *Foundations of deep reinforcement learning: theory and practice in Python*. Addison-Wesley Professional.

Ch 1.3 State Value Function Example

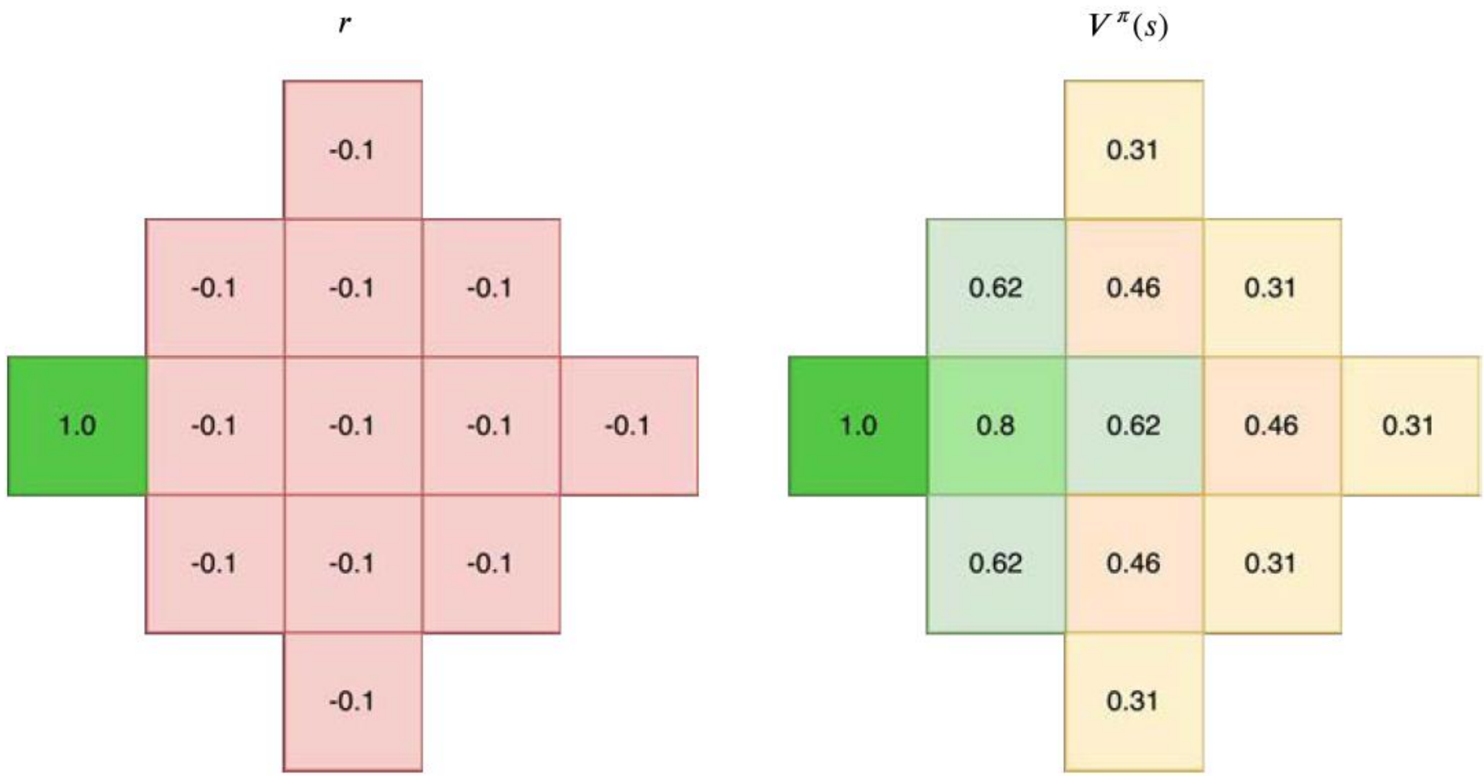


Figure 1.4 Rewards r and values $V^{\pi}(s)$ for each state s in a simple grid-world environment. The value of a state is calculated from the rewards using Equation 1.10 with γ = 0.9 while using a policy π that always takes the shortest path to the goal state with $r = +1$.

Graesser, L., & Keng, W. L. (2019). *Foundations of deep reinforcement learning: theory and practice in Python*. Addison-Wesley Professional.

Ch 1.3 State Value Function Example

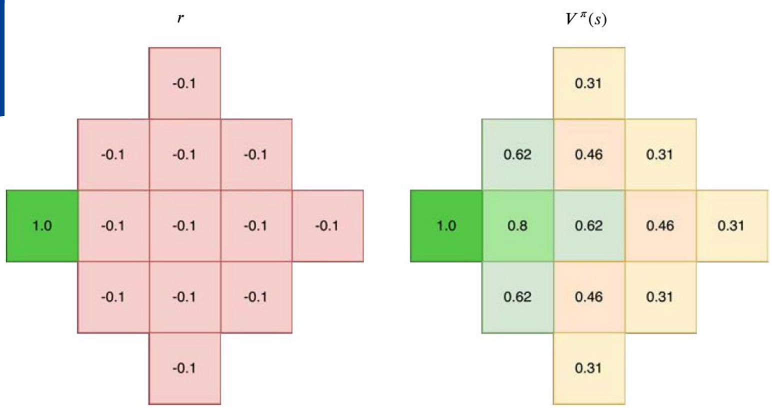


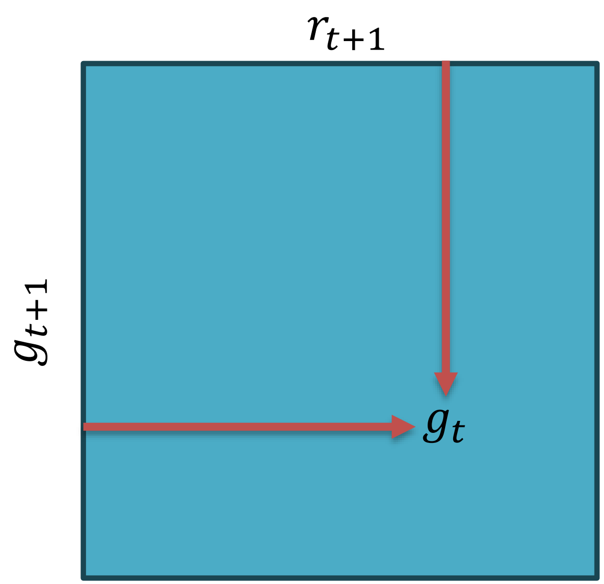
Figure 1.4 Rewards r and values $V^{\pi}(s)$ for each state s in a simple grid-world environment. The value of a state is calculated from the rewards using Equation 1.10 with γ = 0.9 while using a policy π that always takes the shortest path to the goal state with $r = +1$.

Let's check the value 0.8 of figure.

$$v_\pi\big((3,2)\big) = \mathbb{E}[G_t | S_t = (3,2), \pi]$$

$$= \sum_{g_t} P(G_t = g_t | S_t = (3,2), \pi) g_t$$

$$= \sum_{r_{t+1}, g_{t+1}} P(r_{t+1}, g_{t+1} | S_t = (3,2), \pi)(r_{t+1} + \gamma g_{t+1})$$



g_t is decomposable with r_{t+1}, g_{t+1} :

$$g_t = r_{t+1} + g_{t+1}$$

When g_t exists, there must exist r_{t+1}, g_{t+1} and vice versa.

Graesser, L., & Keng, W. L. (2019). *Foundations of deep reinforcement learning: theory and practice in Python*. Addison-Wesley Professional.

Ch 1.3 State Value Function Example Continued…

But we don't know $P(G_t)$, so we will use $P(s', r|s, a)$.

$$\sum_{r_{t+1}, g_{t+1}} P(R_{t+1} = r_{t+1}, G_{t+1} = g_{t+1} | S_t = (3,2), \pi)(r_{t+1} + \gamma g_{t+1})$$

$$= \sum_{r_{t+1}, g_{t+1}} [P(r_{t+1}, g_{t+1} | (3,2), \pi) r_{t+1} + \gamma P(r_{t+1}, g_{t+1} | (3,2), \pi) g_{t+1}]$$

$$= \sum_{r_{t+1}} P(r_{t+1} | (3,2), \pi) r_{t+1} + \gamma \sum_{g_{t+1}} P(g_{t+1} | (3,2), \pi) g_{t+1} \because Marginalization$$

$$= \sum_{a_t, s_{t+1}, r_{t+1}} P(r_{t+1} | S_{t+1} = s_{t+1}, A_t = a_t, (3,2), \pi) P(S_{t+1} = s_{t+1} | A_t = a_t, (3,2), \pi) P(a | (3,2), \pi) r_{t+1}$$

$$+ \gamma \sum_{a_t, s_{t+1}, g_{t+1}} P(g_{t+1} | S_{t+1} = s_{t+1}, A_t = a_t, (3,2), \pi) P(S_{t+1} = s_{t+1} | A_t = a_t, (3,2), \pi) P(a_t | (3,2), \pi) g_{t+1}$$

$$= \sum_{a_t, s_{t+1}, r_{t+1}} P(r_{t+1} | S_{t+1} = s_{t+1}, A_t = a_t, \pi) P(S_{t+1} = s_{t+1} | A_t = a_t, (3,2), \pi) P(a | (3,2), \pi) r_{t+1}$$

$$+ \gamma \sum_{a_t, s_{t+1}, g_{t+1}} [g_{t+1} P(g_{t+1} | S_{t+1} = s_{t+1}, A_t = a_t, \pi)] P(S_{t+1} = s_{t+1} | A_t = a_t, (3,2), \pi) P(a_t | (3,2), \pi)$$

$\because Markov\ Assumption$

Graesser, L., & Keng, W. L. (2019). *Foundations of deep reinforcement learning: theory and practice in Python*. Addison-Wesley Professional.

$$\sum_{a_t, s_{t+1}, r_{t+1}} P(r_{t+1}|S_{t+1} = s_{t+1}, A_t = a_t, \pi) P(S_{t+1} = s_{t+1}|A_t = a_t, (3,2), \pi) P(a|(3,2), \pi) r_{t+1}$$

$$+ \gamma \sum_{a_t, s_{t+1}, g_{t+1}} [g_{t+1} P(g_{t+1}|S_{t+1} = s_{t+1}, A_t = a_t, \pi)] P(S_{t+1} = s_{t+1}|A_t = a_t, (3,2), \pi) P(a_t|(3,2), \pi)$$

Now we remove $\sum_{a_t}, P(a| \cdot)$, as we know that π always select the shortest path to the goal, left.

$$= \sum_{s_{t+1}, r_{t+1}} P(r_{t+1}|S_{t+1} = s_{t+1}, left, \pi) P(S_{t+1} = s_{t+1}|left, (3,2), \pi) r_{t+1}$$

$$+ \gamma \sum_{s_{t+1}, g_{t+1}} [g_{t+1} P(g_{t+1}|S_{t+1} = s_{t+1}, left, \pi)] P(S_{t+1} = s_{t+1}|left, (3,2), \pi)$$

And remove $P(S_{t+1}|left, (3,2))$, because it is always 1, and substitute s_{t+1} as (3,1).

$$= \sum_{r_{t+1}} P(r_{t+1}|S_{t+1} = (3,1), \pi) r_{t+1} + \gamma \sum_{g_{t+1}} [g_{t+1} P(g_{t+1}|S_{t+1} = (3,1), \pi)]$$

Reward of (3,1) is always 1.0 and return of (3,1) is always 0 because the exploration is terminated.

$= (1 - 0.1) = 0.9$

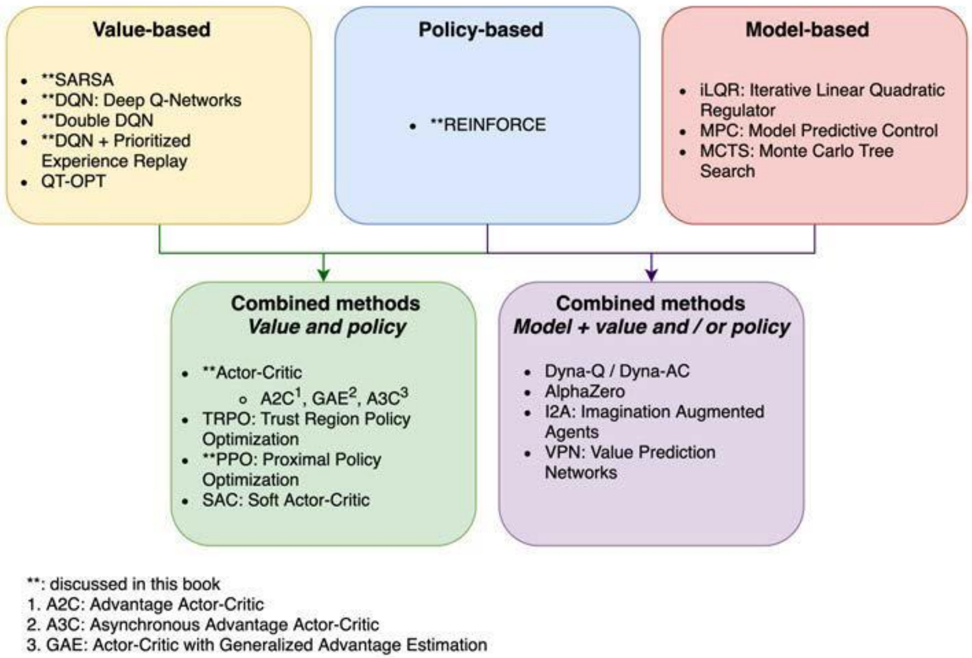
But why it is not 0.8? There is hidden terminal state in the figure, but not described.

$-0.1 + 0.9 \cdot (1) = 0.8$

Graesser, L., & Keng, W. L. (2019). *Foundations of deep reinforcement learning: theory and practice in Python*. Addison-Wesley Professional.

Ch 1.4
Deep Reinforcement Learning Algorithms

- Reinforcement learning is a problem of determining policy over a given environment.
- So, how can we get the good policy?



Graesser, L., & Keng, W. L. (2019). *Foundations of deep reinforcement learning: theory and practice in Python*. Addison-Wesley Professional.

- We can just try to learn a policy directly from the given environment.
- In this case, we learn probability distribution $\pi(a|s)$ directly.

$$J(\tau) = \mathbb{E}_{\tau\sim\pi}\left[\sum_{t=0}^{T} \gamma^t r_t\right]$$

- REINFORCE is the most famous policy-based algorithm.
- Policies always converge to local minima, by Policy Gradient Theorem.
- Policy-based methods have high variance and are sample-efficient.

Graesser, L., & Keng, W. L. (2019). *Foundations of deep reinforcement learning: theory and practice in Python*. Addison-Wesley Professional.

$$MSE(x) = \sigma^2 + \left(h^*(x) - h_{avg}(x)\right)^2 + Var\left(h_S(x)\right)$$

- x is a sample drawn from an environment and σ is a noise variance over x.
- h^* is a ground truth model of environment.
- h_S is a model trained over a dataset S and **trained by a stochastic algorithm.**
- h_{avg} is the mean of outputs $\mathbb{E}_S[h_S(x)]$.
- $\left(h^*(x) - h_{avg}(x)\right)$ is a bias, which means that "How much far away the models' outputs mean from the ground truth."
- $Var\left(h_S(x)\right)$ is a variance, which means that "How much the models' outputs are scattered."
- There is always tradeoff between bias and variance.

Graesser, L., & Keng, W. L. (2019). *Foundations of deep reinforcement learning: theory and practice in Python*. Addison-Wesley Professional.

- We can assume that target environment has Markov property, which means that MDP can be applied.
- By assuming MDP, we can use value functions: v_π, q_π.
- SARSA is one of the oldest RL algorithm: it learns q_π. But it is not used because of high variance and sample inefficiency.
- Deep Q-Networks and its descendants are much more popular these days. (Ch 4, 5)
- Sample-efficient and lower variance, but no guarantee of converge.

Graesser, L., & Keng, W. L. (2019). *Foundations of deep reinforcement learning: theory and practice in Python*. Addison-Wesley Professional.

- If we have the dynamics of the target environment, we can utilize it. E.g. Games
- Monte Carlo Tree Search (MCTS) is a well-known model-based method that can be applied to problems with deterministic discrete state spaces. E.g. Go programs.
- Linear Quadratic Regulators (iLQR) [79] or Model Predictive Control (MPC), involve learning the transition dynamics.
- In robotics. Compared to policy-based or value-based methods, these algorithms also tend to require many fewer samples of data to learn good policies.
- However, for most problems, models are hard to come by. Many environments are stochastic, and their transition dynamics are not known.
- The distinction between model-based and model-free is also used to classify reinforcement learning algorithms.

Graesser, L., & Keng, W. L. (2019). *Foundations of deep reinforcement learning: theory and practice in Python*. Addison-Wesley Professional.

- We can combine policy-based and value-based. It is called Actor-Critic.
- Actor-critic algorithms' area is under development.
- Trust Region Policy Optimization, Proximal Policy Optimization are the examples of actor-critic.

- We can also combine model-based and others.
- AlphaGo combines MCTS, state value and policy-based.
- Dyna-Q iteratively learns the environment and train action-value function.

Graesser, L., & Keng, W. L. (2019). *Foundations of deep reinforcement learning: theory and practice in Python*. Addison-Wesley Professional.

- On-policy algorithms only utilized data generated from the current policy.
  - E.g. REINFORCE, SARSA, actor-critic, PPO
- Off-policy algorithms additionally utilize data not generated from the current policy. It is sample efficient.
  - E.g. DQN and its extensions.

Graesser, L., & Keng, W. L. (2019). *Foundations of deep reinforcement learning: theory and practice in Python*. Addison-Wesley Professional.

Ch 1.5
Deep Learning for Reinforcement Learning

- Deep learning can approximate the function.
- Any input and label pair are given in advance: the neural network interacts with environment.
- Previous input affects later output: it makes hard to apply gradient descent.

Graesser, L., & Keng, W. L. (2019). *Foundations of deep reinforcement learning: theory and practice in Python*. Addison-Wesley Professional.

Ch 1.6
Reinforcement Learning and Supervised Learning

- There are many differences between supervised learning (SL) and reinforcement learning.
  - Lack of an oracle
  - Sparsity of feedback
  - Data generated during training

Graesser, L., & Keng, W. L. (2019). *Foundations of deep reinforcement learning: theory and practice in Python*. Addison-Wesley Professional.

- We do not know the correct answer in RL.
- In SL, labels convey a lot of information. (usually 1 bit per class.)
- In RL, we just receive how good or bad the action was. It does not tell us the action was correct.

- Often, the reward function of environment is very sparse.
- We don't know whether our previous actions were good or not.
- In fact, even if we are given the reward, we cannot specify the actions that evokes positive/negative rewards.

- The combination of lack of oracle and sparse feedback makes RL less sample-efficient.

Graesser, L., & Keng, W. L. (2019). *Foundations of deep reinforcement learning: theory and practice in Python*. Addison-Wesley Professional.

- In SL, just apply a pre-drawn dataset.
- In RL, the model interacts with the environment. It makes up feedback loop.

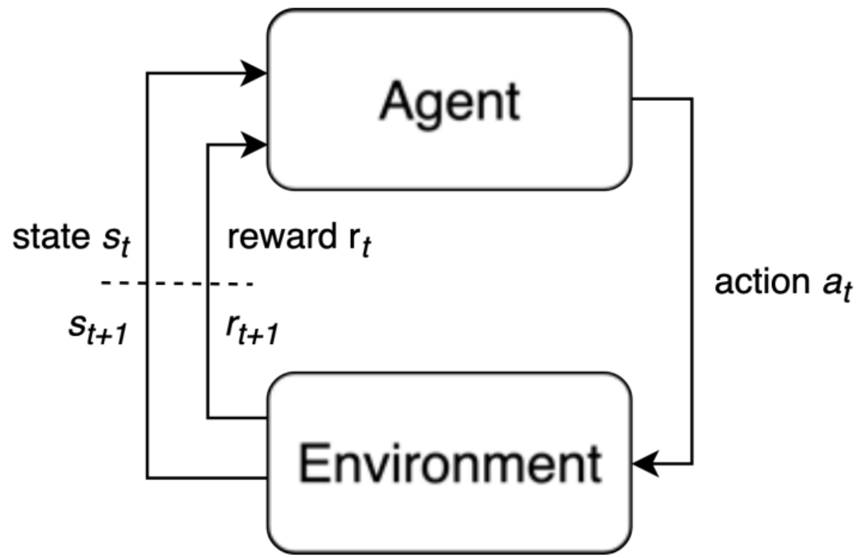


Figure 1.2 The reinforcement learning control loop

Graesser, L., & Keng, W. L. (2019). *Foundations of deep reinforcement learning: theory and practice in Python*. Addison-Wesley Professional.