Reinforcement Learning for Business, Economics, and Social Sciences

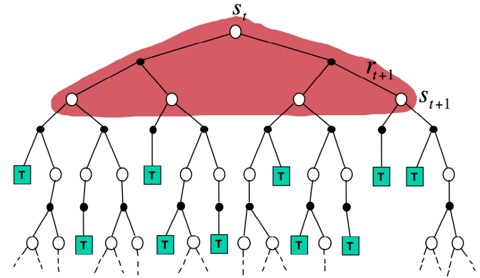
Unit 3-2: Monte Carlo Learning

Davud Rostam-Afschar (Uni Mannheim)

How to learn from episodes?

Dynamic Programming Backup

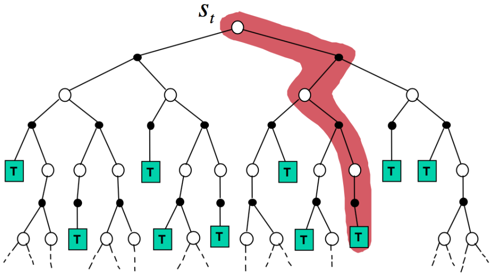
$$V(S_t) \leftarrow \mathbb{E}_\pi \left[R_{t+1} + \gamma V(S_{t+1}) \right]$$



Source: David Silver

Monte Carlo Backup

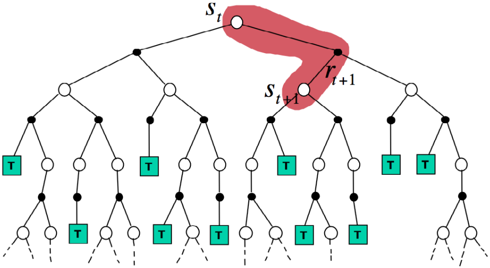
$$V(S_t) \leftarrow V(S_t) + \alpha\left(G_t - V(S_t)\right)$$



Source: David Silver

Temporal Difference Backup

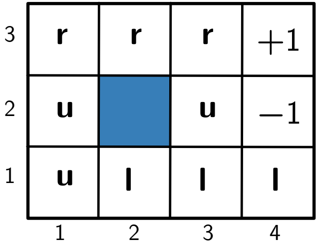
$$V(S_t) \leftarrow V(S_t) + \alpha \left(R_{t+1} + \gamma V(S_{t+1}) - V(S_t) \right)$$



Source: David Silver

- Given a policy π estimate $V^\pi(s)$ without any transition or reward model
- **Monte Carlo** evaluation

$$
\begin{aligned}
V^\pi(s) &= \mathbb{E}_\pi\left[\sum_t \gamma^t r_t\right] \\
&\approx \frac{1}{n(s)} \sum_{k=1}^{n(s)} \left[\sum_t \gamma^t r_t^{(k)}\right] \qquad \text{(sample approximation)}
\end{aligned}
$$



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

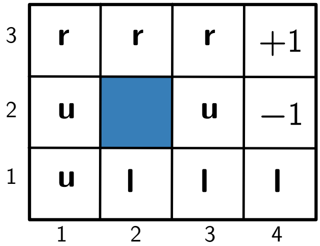
What is the value $V(s)$ of being in state s

Monte Carlo Evaluation

Monte Carlo Evaluation

- Let G_k be a one-trajectory Monte Carlo target

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



Monte Carlo Evaluation

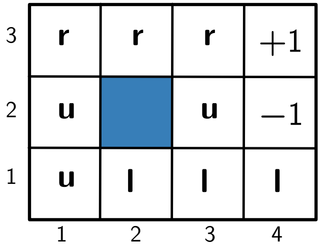
- Let G_k be a one-trajectory Monte Carlo target

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

- First sample $(k = 1)$:

$$(1,1) \to (1,2) \to (1,3) \to (1,2) \to (1,3) \to (2,3) \to (3,3) \to (4,3)$$
$$-0.04 - 0.04 - 0.04 - 0.04 - 0.04 - 0.04 - 0.04 + 1$$

$G_1 = 0.72$



Monte Carlo Evaluation

- Let G_k be a one-trajectory Monte Carlo target

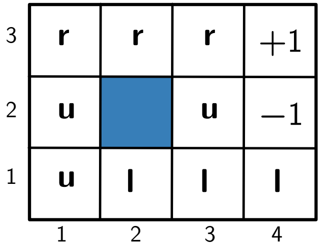
$$G_k = \sum_t \gamma^t r_t^{(k)}$$

- First sample $(k = 1)$:

$$(1,1) \to (1,2) \to (1,3) \to (1,2) \to (1,3) \to (2,3) \to (3,3) \to (4,3)$$
$$-0.04 - 0.04 - 0.04 - 0.04 - 0.04 - 0.04 - 0.04 + 1$$
$$G_1 = 0.72$$

- Second sample $(k = 2)$:

$$(1,1) \to (1,2) \to (1,3) \to (2,3) \to (3,3) \to (3,2) \to (3,3) \to (4,3)$$
$$-0.04 - 0.04 - 0.04 - 0.04 - 0.04 - 0.04 - 0.04 + 1$$
$$G_2 = 0.72$$



Monte Carlo Evaluation

- Let G_k be a one-trajectory Monte Carlo target

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

- First sample ($k = 1$) :

$$(1,1) \to (1,2) \to (1,3) \to (1,2) \to (1,3) \to (2,3) \to (3,3) \to (4,3)$$
$$-0.04 - 0.04 - 0.04 - 0.04 - 0.04 - 0.04 - 0.04 + 1$$
$$G_1 = 0.72$$

- Second sample ($k = 2$) :

$$(1,1) \to (1,2) \to (1,3) \to (2,3) \to (3,3) \to (3,2) \to (3,3) \to (4,3)$$
$$-0.04 - 0.04 - 0.04 - 0.04 - 0.04 - 0.04 - 0.04 + 1$$
$$G_2 = 0.72$$

- Third sample ($k = 3$):

$$(1,1) \to (2,1) \to (3,1) \to (3,2) \to (4,2)$$
$$-0.04 - 0.04 - 0.04 - 0.04 - 1$$
$$G_3 = -1.16$$

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

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

$$V_n^\pi(s) \approx \frac{1}{n(s)} \sum_{k=1}^{n(s)} G_k$$

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

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

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

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

- **Incremental update**

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

where $\alpha_n =$ learning rate $1/n(s)$

Stochastic approximation (Robbins-Monro algorithm)

- **Theorem**: If α_n is appropriately decreased with number of times a state is visited then $V_n^\pi(s)$ converges to correct value
- **Sufficient conditions** for α_n :

$$\sum_n \alpha_n \to \infty \tag{1}$$

$$\sum_n \alpha_n^2 < \infty \tag{2}$$

- Often $\alpha_n(s) = 1/n(s)$, where $n(s) = \#$ of times s is visited

Stochastic approximation (Robbins-Monro algorithm)

- **Theorem**: If α_n is appropriately decreased with number of times a state is visited then $V_n^\pi(s)$ converges to correct value
- **Sufficient conditions** for α_n :

$$\sum_n \alpha_n \to \infty \tag{1}$$

$$\sum_n \alpha_n^2 < \infty \tag{2}$$

- Often $\alpha_n(s) = 1/n(s)$, where $n(s) = \#$ of times s is visited

$n(s)$	α_n
2	50%

Stochastic approximation (Robbins-Monro algorithm)

- **Theorem**: If α_n is appropriately decreased with number of times a state is visited then $V_n^\pi(s)$ converges to correct value
- **Sufficient conditions** for α_n :

$$\sum_n \alpha_n \to \infty \quad (1)$$

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

- Often $\alpha_n(s) = 1/n(s)$, where $n(s) = \#$ of times s is visited

$n(s)$	α_n
2	50%
5	20%

Stochastic approximation (Robbins-Monro algorithm)

- **Theorem**: If α_n is appropriately decreased with number of times a state is visited then $V_n^\pi(s)$ converges to correct value
- **Sufficient conditions** for α_n :

$$\sum_n \alpha_n \to \infty \tag{1}$$

$$\sum_n \alpha_n^2 < \infty \tag{2}$$

- Often $\alpha_n(s) = 1/n(s)$, where $n(s) = \#$ of times s is visited

$n(s)$	α_n
2	50%
5	20%
10	10%

Stochastic approximation (Robbins-Monro algorithm)

- **Theorem**: If α_n is appropriately decreased with number of times a state is visited then $V_n^\pi(s)$ converges to correct value
- **Sufficient conditions** for α_n :

$$\sum_n \alpha_n \to \infty \tag{1}$$

$$\sum_n \alpha_n^2 < \infty \tag{2}$$

- Often $\alpha_n(s) = 1/n(s)$, where $n(s) = \#$ of times s is visited

$n(s)$	α_n
2	50%
5	20%
10	10%
20	5%

Stochastic approximation (Robbins-Monro algorithm)

- **Theorem**: If α_n is appropriately decreased with number of times a state is visited then $V_n^\pi(s)$ converges to correct value
- **Sufficient conditions** for α_n :

$$\sum_n \alpha_n \to \infty \tag{1}$$

$$\sum_n \alpha_n^2 < \infty \tag{2}$$

- Often $\alpha_n(s) = 1/n(s)$, where $n(s) = \#$ of times s is visited

$n(s)$	α_n
2	50%
5	20%
10	10%
20	5%
40	2.5%

Stochastic approximation (Robbins-Monro algorithm)

- **Theorem**: If α_n is appropriately decreased with number of times a state is visited then $V_n^\pi(s)$ converges to correct value
- **Sufficient conditions** for α_n :

$$\sum_n \alpha_n \to \infty \tag{1}$$

$$\sum_n \alpha_n^2 < \infty \tag{2}$$

- Often $\alpha_n(s) = 1/n(s)$, where $n(s) = \#$ of times s is visited

$n(s)$	α_n
2	50%
5	20%
10	10%
20	5%
40	2.5%
80	1.25%

First-visit Monte Carlo (MC) Evaluation

MCevaluation (π, V^π)

Initialize

$\pi \leftarrow$ policy to be evaluated

$V^\pi(s) \leftarrow$ arbitrary state-value function

$n(s) \leftarrow 0, \; \forall s \in \mathcal{S}$

Repeat

Generate the kth episode using $\pi(s)$

For each state t appearing in the episode

Return r following the first occurrence of t

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

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

Update value: $V^\pi(s) \leftarrow V^\pi(s) + \alpha \left(\sum_t \gamma^t r_t^{(k)} - V^\pi(s) \right)$

Until convergence of V^π

Return V^π

Monte Carlo Control

Monte Carlo Control

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

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

- Alternate between
  - **Policy evaluation**

$$Q_n^*(s,a) \leftarrow Q_{n-1}^\pi(s,a) + \alpha_n \left(G_k^a - Q_{n-1}^\pi(s,a)\right)$$

  - **Policy improvement**

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

References I

DeNero, J., D. Klein, B. Miller, N. Hay, and P. Abbeel (2013): "The Pacman AI Projects," http://inst.eecs.berkeley.edu/~cs188/pacman/, Developed at UC Berkeley. Core by John DeNero and Dan Klein; student autograding by Brad Miller, Nick Hay, and Pieter Abbeel.

Poupart, P. (2025): "Pascal Poupart's Homepage," https://cs.uwaterloo.ca/~ppoupart/, Accessed: 2025-05-24.

Russell, S. J., and P. Norvig (2016): *Artificial intelligence: a modern approach*. Pearson.

Sigaud, O., and O. Buffet (2013): *Markov decision processes in artificial intelligence*. John Wiley & Sons.

Sutton, R. S., and A. G. Barto (2018): "Reinforcement learning: An introduction," *A Bradford Book*, Available at http://incompleteideas.net/book/the-book-2nd.html.

Szepesvári, C. (2022): *Algorithms for reinforcement learning*. Springer nature, Available at https://sites.ualberta.ca/~szepesva/RLBook.html.

Takeaways

How to Learn Values Using Monte Carlo Methods?

- No need to know transition probabilities or reward function
  $\rightarrow$ Model free
- Average returns from complete episodes under the target policy
  $\rightarrow$ Unbiased value estimation from samples
- Revises estimates only after each episode using the observed return
  $\rightarrow$ Needs many trajectories