Scalable ML

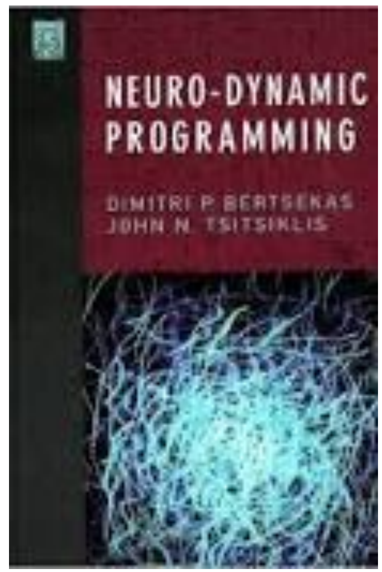
10605-10805

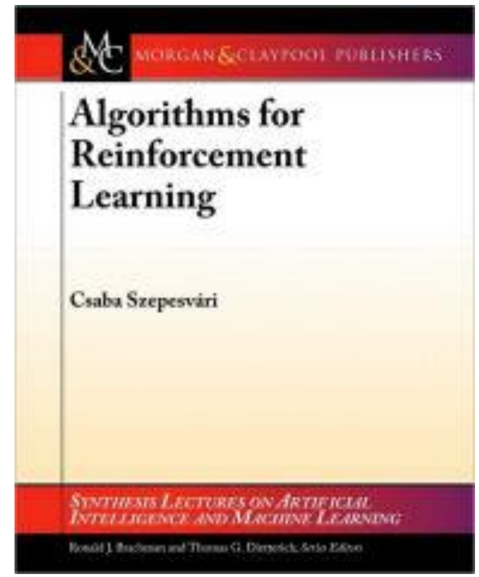
Introduction to Reinforcement Learning

Barnabás Póczos

RL Books


Reinforcement
Learning
Richard S. Sutton and Andrew G. Barto


NEURO-DYNAMIC
PROGRAMMING
DIMITRI P. BERTSEKAS
JOHN N. TSITSIKLIS

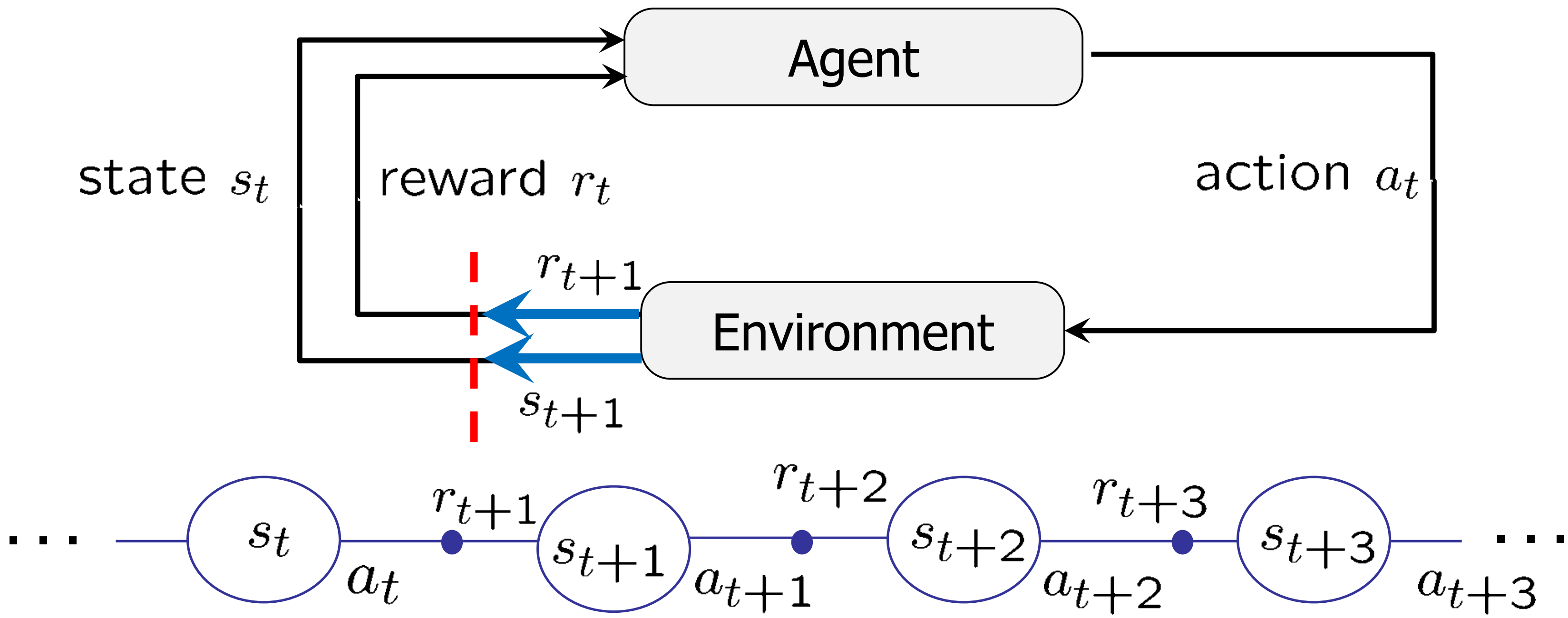

MORGAN & CLAYPOOL PUBLISHERS
Algorithms for
Reinforcement
Learning
Csaba Szepesvári
Synthesis Lectures on Artificial
Intelligence and Machine Learning

Introduction to Reinforcement Learning

Reinforcement Learning Applications

- Finance
  - Portfolio optimization
  - Trading
- Control
  - Air conditioning, power grid, helicopter…
- Robotics
- Games
  - Go, Chess, Backgammon
  - Computer games
- Chatbots
- …

Reinforcement Learning Framework



- ⋆ Agent and environment interact in discrete time steps: $t = 0, 1, 2, \ldots$
- ⋆ Agent observes state $s_t \in \mathcal{S}$ in time step t.
- ⋆ Produces action $a_t \in \mathcal{A}(s_t)$ in time step t.
- ⋆ Get reward $r_{t+1} \in \mathbb{R}$
- ⋆ observe next state $s_{t+1} \in \mathcal{S}$

Markov Decision Processes

RL Framework + Markov assumption

MDP $= (\mathcal{S}, \mathcal{A}, P, R, s_0, \gamma)$.

- $\mathcal{S}$: observable state space
- $\mathcal{A}$: action space
- P: state transition probabilities
- R: reward function
- s_0: starting state
- γ: reward discont rate.

Markov assumption: $P(s_{t+1}|s_0, a_0, \ldots, s_t, a_t) = P(s_{t+1}|s_t, a_t)$

Reward assumption: $R(s_0, a_0, \ldots, s_t, a_t, s_{t+1}) = R(s_t, a_t, s_{t+1}) = r_{t+1} \in \mathbb{R}$

Policy: $\pi(s, a) = P(a_t|s_t) \in [0, 1]$, that is $a_t \sim \pi(s_t, \cdot)$

Goal : $\max_{\pi} \mathbb{E}[r_0 + r_1 + \ldots]$

Discount Rates

Goal: $\max_{\pi} \mathbb{E}[r_0 + r_1 + r_2 + \ldots]$

An issue: $r_0 + r_1 + r_2 + \ldots$ can be infinte...

Solution:

New goal: $\max_{\pi} \mathbb{E}[\sum_{t=0}^{\infty} \gamma^t r_t]$, for some $0 < \gamma < 1$ discount rate

RL is different from Supervised/Unsupervised learning

- ⋆ Functions to be learned: $\pi : \mathcal{S} \to \mathcal{A}$
- ⋆ However, training examples are not in the form of (s, a) pairs!
- ⋆ Training examples are in the form of $\{(s_t, a_t), r_t\}_{t=1}^{T}$

  (or $\{(s_t, a_t, s_{t+1}), r_t\}_{t=1}^{T}$)

State-Value Function

For a given state s and policy π, the value of state s:

$$V^{\pi}(s) := \mathbb{E}_{\pi}\left[\sum_{k=0}^{\infty} \gamma^{k} r_{t+k+1} \,\middle|\, s_t = s\right]$$

This is the state-value function of policy π

Action-Value Function

Value of state s after taking action a.

$$Q^\pi(s,a) := \mathbb{E}_\pi \left[\sum_{k=0}^{\infty} \gamma^k r_{t+k+1} \,\middle|\, s_t = s, a_t = a \right]$$

Relation between Q and V Functions

Q from V:

$$Q^{\pi}(s,a) = \sum_{s' \in \mathcal{S}} P(s'|s,a)\left[R(s,a,s') + \gamma V^{\pi}(s')\right]$$

V from Q:

$$V^{\pi}(s) = \sum_{a \in \mathcal{A}} \pi(a|s) Q^{\pi}(s,a)$$

The Optimal Value Function and Optimal Policy

Partial ordering between policies:

$$\pi_1 \geq \pi_2 \Leftrightarrow V^{\pi_1}(s) \geq V^{\pi_2}(s) \qquad \forall s \in \mathcal{S}$$

Some policies are not comparable!

Optimal policy and optimal state-value function:

$$V^*(s) := \max_{\pi} V^{\pi}(s) = V^{\pi^*}(s), \qquad \forall s \in \mathcal{S}$$

π^*: policy whose value function is the maximum out of all policies simultaneously for all states

V*(s) shows the maximum expected discounted reward that one can achieve from state s with optimal play

The Optimal Action-Value Function

Similarly, the optimal action-value function:

$$Q^*(s,a) := \max_{\pi} Q^{\pi}(s,a)$$

Important Properties:

$$Q^*(s,a) = \mathbb{E}\left[r_{t+1} + \gamma V^*(s_{t+1}) \mid s_t = s, a_t = a\right]$$

$$V^*(s) = \max_{a \in \mathcal{A}} Q^*(s,a)$$

$$Q^*(s,a) = \sum_{s' \in \mathcal{S}} P(s'|s,a)\left[R(s,a,s') + \gamma V^*(s')\right]$$

The Existence of the Optimal Policy

Theorem: For any Markov Decision Processes

($\star$) there exists an optimal policy π^* that is at least as good as all other policies:

$$\pi^* \geq \pi \quad \forall \pi$$

($\star$) There can be many optimal policies, but all optimal policies achieve the optimal value function:

$$V^{\pi^*}(s) = V^*(s) \quad \forall s$$

($\star$) All optimal policies achieve the optimal action-value function,

$$Q^{\pi^*}(s,a) = Q^*(s,a) \quad \forall s, a$$

(*) There is always a deterministic optimal policy for any MDP

Bellman optimality equation for V*

Theorem [Bellman optimality equation for V*]:

$$V^*(s) = \max_{a \in \mathcal{A}} \sum_{s' \in \mathcal{S}} P(s'|s,a)[R(s,a,s') + \gamma V^*(s')]$$

Greedy Policy for Q(s,a)

Definition: Greedy policy for a given $Q(s,a)$ function:

$$\pi(s,a) = \begin{cases} 1, & \text{if } a = \arg\max_a Q(s,a) \\ 0, & \text{otherwise;} \end{cases}$$

RL Tasks

- Policy evaluation:

  Given policy π, what is $V^{\pi}(s)$ and $Q^{\pi}(s,a)$?

- Policy improvement

  Given policy π, can we create another policy π' such that $\pi' \geq \pi$, that is $V^{\pi'}(s) \geq V^{\pi}(s)\ \forall s$?

- Finding an optimal policy

  How can we find an optimal policy π^*?

Monte Carlo Policy Evaluation Without knowing the model

⋆ Let $R(s)$ be the reward that can be achieved from state s following policy π.

⋆ It is a random variable with expected value $V^\pi(s)$.

$$\begin{aligned} V^\pi(s) &:= \mathbb{E}_\pi \left[\sum_{k=0}^{\infty} \gamma^k r_{t+k+1} \,|\, s_t = s \right] \\ &= \mathbb{E}_\pi[R(s)] \end{aligned}$$

Monte Carlo Estimation of $V^\pi(s)$

- Empirical average: Let us use *N* simulations starting from state *s* following policy π.

- The observed rewards are: $R_1(s), R_2(s), \ldots, R_N(s)$

- Let $\hat{V}(s) := \frac{1}{N} \sum_{k=1}^{N} R_k(s)$

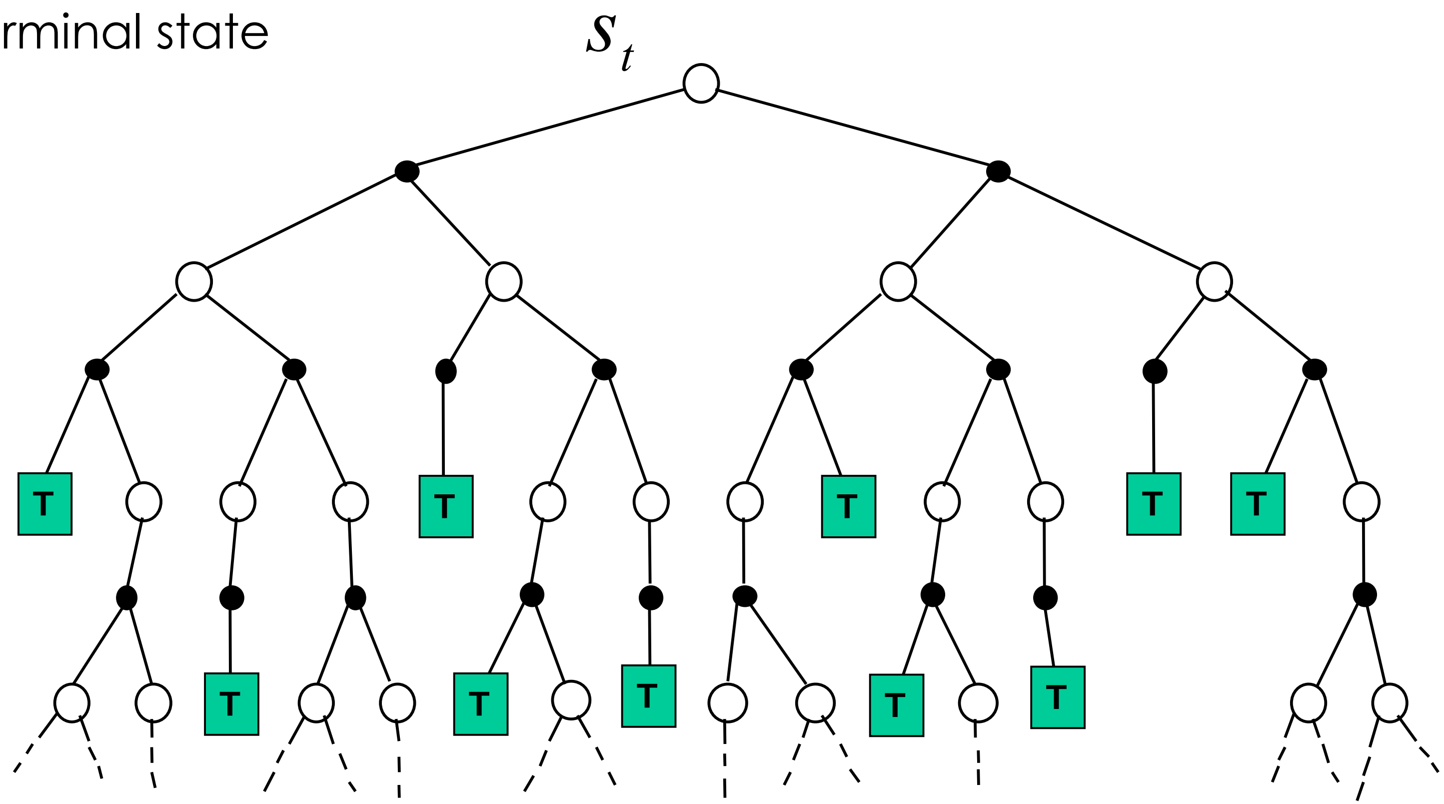
- This is the so-called „Monte Carlo" method.

- MC can estimate $V^\pi(s)$ without knowing the model

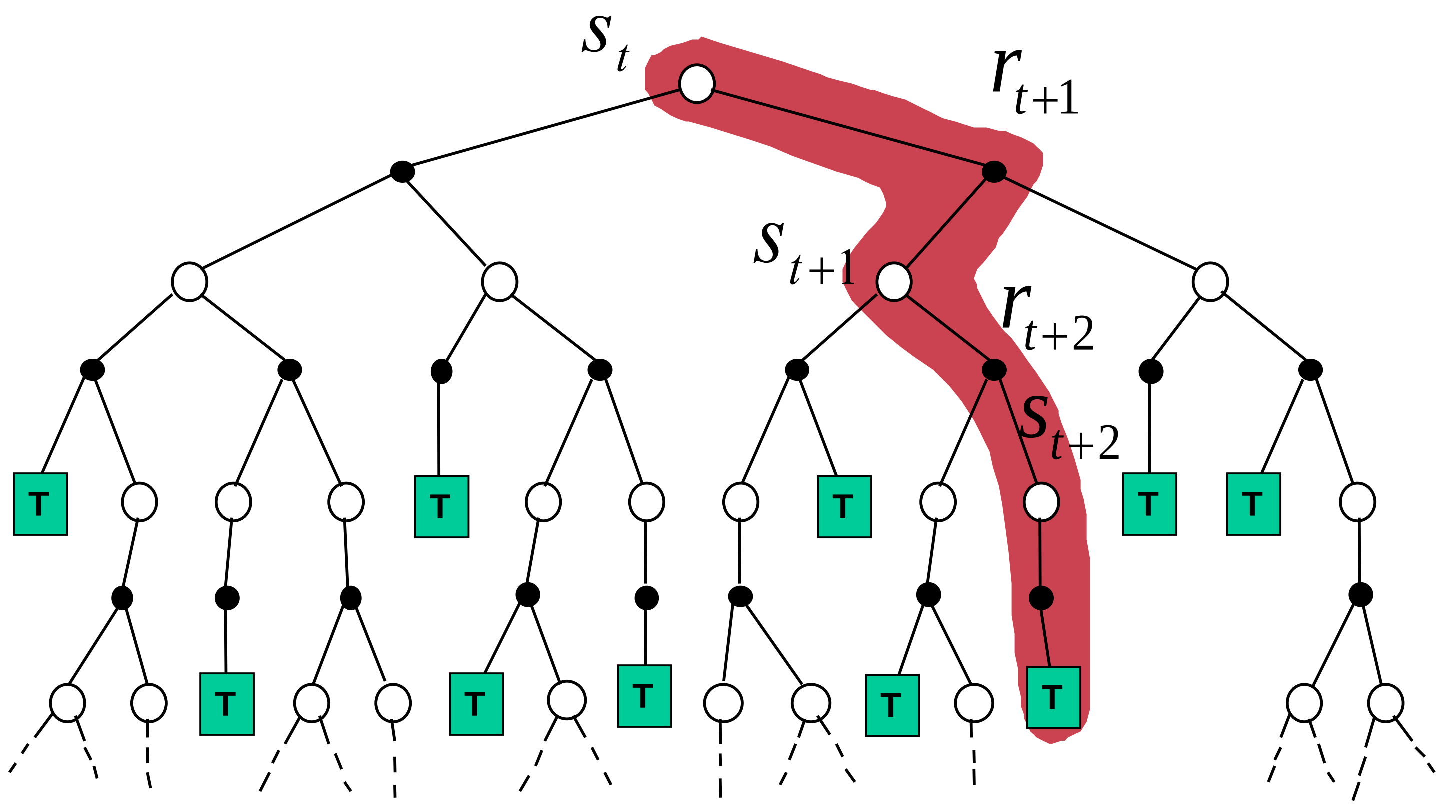
$$\hat{V}(s) \to V^\pi(s)$$

MDP Backup Diagrams

- White circle: state
- Black circle: action
- T: terminal state

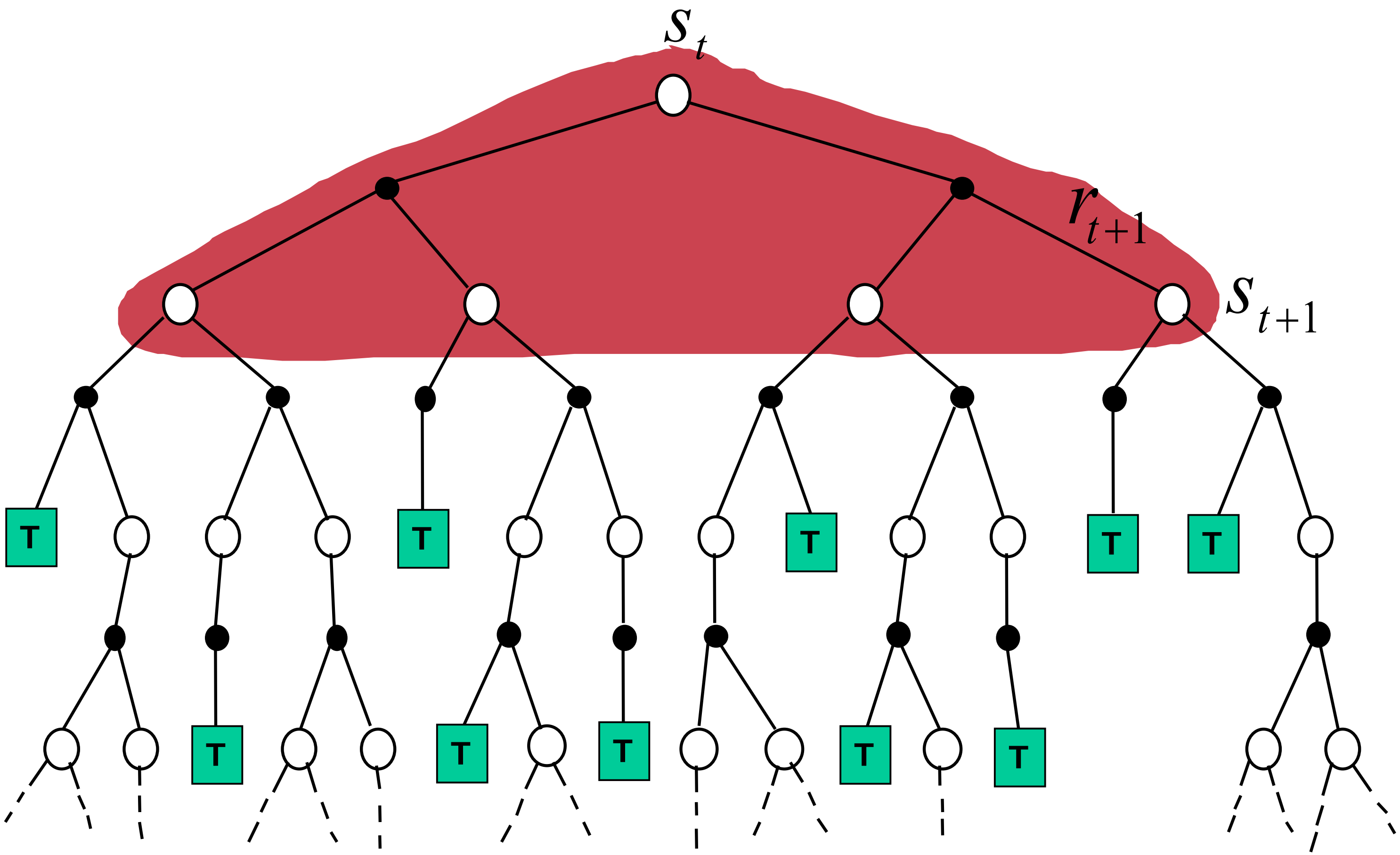


Monte Carlo Backup Diagram



MC estimate: $V_k(s_t) := V_{k-1}(s_t) + \alpha_k \cdot (R_k(s_t) - V_{k-1}(s_t))$

Dynamic Programming Backup Diagram



$$V^\pi(s) = \sum_{a \in \mathcal{A}} \pi(a|s) \sum_{s' \in \mathcal{S}} P(s'|s,a)[R(s,a,s') + \gamma V^\pi(s')]$$

Thank you for your attention!