

Chapter 5: Temporal difference methods (TD)

Planning

- MDP model known/given
- Solve to get V_* , q_* , π_*
 - Bellman opt. eq.
 - Policy iteration
 - Value iteration

Learning

- MDP model unknown
- Estimate V_* , q_* , π_*
 - Sampling
 - Exploration
 - RL

Prediction

- Predict effect of policy
- Policy evaluation V_π

Control

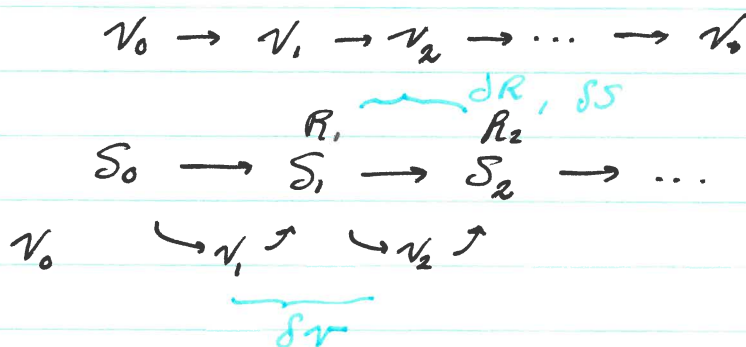
- Find optimal policy
- Find V_* , q_*

needed for learning

- Idea of TD:

- Use observed/visited states to estimate V_* iteratively
- Sampling / exploration informs estimation
- New states / actions depend on estimated V_*

feedback



See HC course

- Stochastic approximation:

- Iteration, map: $X_{n+1} = T(X_n)$
 $X_n \rightarrow x^*$

$x^* = T(x^*)$ attractive fixed point

- Stochastic iteration: $X_{n+1} = T(X_n)$

$$\begin{aligned}
 X_{n+1} &= (1 - a_n) X_n + a_n T(X_n) \\
 &= X_n + a_n \underbrace{(T(X_n) - X_n)}_{\text{difference}}
 \end{aligned}$$

TD error $\rightarrow 0$

$$a_n \geq 0$$

$$\sum a_n = \infty$$

$$\sum a_n^2 < \infty$$

5.1 TD(0)

· Policy iteration (prediction, not control)

· Bellman equation:

$$V_{\pi}(s) = E_{\pi}[R_{t+1} + \gamma V_{\pi}(S_{t+1}) | S_t = s]$$

$$V_{\pi} = \rho_{\pi} + \gamma P_{\pi} V_{\pi} = T(V_{\pi})$$

Bellman operator

· TD(0) iteration:

TD error

$$\begin{aligned} V_{k+1}(S_t) &= V_k(S_t) + \alpha_k [R_{t+1} + \gamma V_k(S_{t+1}) - V_k(S_t)] \\ &= (1 - \alpha_k) V_k(S_t) + \alpha_k [R_{t+1} + \gamma V_k(S_{t+1})] \end{aligned}$$

Bellman iteration update

· SA version of value iteration

· SA:

expectation

$$E_{\pi}[R_{t+1} + \gamma V_{\pi}(S_{t+1}) | S_t = s]$$

do nothing

RV

$R_{t+1} + \gamma V_{\pi}(S_{t+1})$ In transition $S_t \rightarrow S_{t+1}$

s s'

· α_k : annealing sequence

$\alpha_k = \alpha$ constant: Convergence in mean with episode loop

$\alpha_k \sim \frac{1}{k}$: Convergence in probability

TD(0) algorithm SB p.120

Input: policy π

Parameters: $\alpha \in (0,1]$, # episodes, # time steps

Initialize $V(s)$

Loop for each episode:

Initialize S

Loop for each time steps:

$A \leftarrow$ action from π for S

Get S', R from action A

$V(S) \leftarrow V(S) + \alpha (R + \gamma V(S') - V(S))$

$S \leftarrow S'$

5.2 Sarsa

On-policy control iteration for estimating q_*

Bellman optimality equation:

$$q_*(s, a) = E[R_{t+1} + \gamma \max_{a'} q_*(S_{t+1}, a') \mid S_t = s, A_t = a]$$

Sarsa iteration:

$$\begin{aligned} Q_{k+1}(S_t, A_t) &= Q_k(S_t, A_t) + \alpha_k [R_{t+1} + \gamma Q_k(S_{t+1}, A_{t+1}) - Q_k(S_t, A_t)] \\ &= (1 - \alpha_k) Q_k(S_t, A_t) + \alpha_k [R_{t+1} + \gamma Q_k(S_{t+1}, A_{t+1})] \end{aligned}$$

A_t chosen ϵ -greedy from $Q_k(S_t, \cdot)$ before update
 A_{t+1} " " " " $Q_k(S_{t+1}, \cdot)$

↳ define policy π at each stage

- SA of Bellman opt. eq.
- Optimal actions A_t, A_{t+1} from current Q_n
- Use all states: s, a, r, s', a'
- On-policy: policy used to update Q

" policy used to generate new actions
- Convergence: Same as TD(0)

5.3. Q-learning

· Off-policy control situation for estimating q_*

· Q-learning situation:

$$\begin{aligned} Q_{k+1}(S_t, A_t) &= Q_k(S_t, A_t) + \alpha_k [R_{t+1} + \gamma \max_a Q_k(S_{t+1}, a) - Q_k(S_t, A_t)] \\ &= (1 - \alpha_k) Q_k(S_t, A_t) + \alpha_k [R_{t+1} + \gamma \max_a Q_k(S_{t+1}, a)] \end{aligned}$$

Find max over action

A_t chosen in ϵ -greedy way from $Q_k(S_t, \cdot)$

· Variation of Sarsa

- A_t chosen ϵ -greedy from Q_k
- A_{t+1} " greedy " Q_k $\epsilon = 0$
- A_{t+1} not used after

· Off-policy: Action selected

action used to update Q

· Convergence: Same as TD(0)

· Q-learning algorithm SB, p.181

Parameters: α , # episodes, # time steps, ϵ

Initialize $Q(s, a)$

Loop for each episode:

Initialize S

Loop for each time steps:

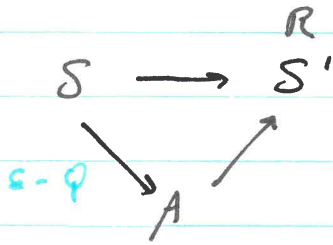
Choose A from S using ϵ -greedy from Q

Get S', R from A

$$Q(S, A) \leftarrow Q(S, A) + \alpha (R + \gamma \max_a Q(S', a) - Q(S, A))$$

Define policy

$S \leftarrow S'$



$$\max_a Q(S', a)$$

for update

· Maximization bias:

max estimated $q \neq$ max in expectation

$$\max \hat{f}(\cdot) \neq \max E[f(\cdot)]$$

· Choosing largest value necessarily above expectation

$$\Rightarrow Q \text{ i.e. } \hat{q} \text{ generally } > \text{ true } q_*$$

$$V \text{ i.e. } \hat{v} \quad " \quad " \quad " \quad v_*$$

Conclusions - to go further

- Represent $V(s)$ $Q(s, a)$
 $|S|$ $|S| \times |A|$ SB
Chaps
9-12

- Parameterizations / : $V(s; \theta)$
Projections $Q(s, a; \theta)$

- Neural networks : $s \rightarrow \boxed{}^{\theta} \rightarrow V(s; \theta)$
deep RL $s, a \Rightarrow \boxed{} \rightarrow Q(s, a; \theta)$

- Adapt TD(0), Sarsa, Q-learning
to update parameters θ

- Estimation, sampling : $Q_n \rightarrow Q_{n+1}$
 $V_n \rightarrow V_{n+1}$ SB
Chaps
5-7

- Use longer history
- TD(λ)
- Eligibility traces

- Policy approximation: $\pi(s) \rightarrow \pi(s; \theta)$ SB
Chap 13

- Gradient ascent : $\theta_{n+1} = \theta_n + \beta \nabla J(\theta)$
?

- Policy gradient
- Actn-critic methods



- Other topics
 - Multi-agent
 - Partially observed MDPs : POMDPs
 - etc.