

ch1: Markov Decision Process

Static Concepts:

state(S) - policy(π) - action(A) - model(p) - state(S')

reward(R') from transition

return(G) from trajectory

value(V+Q)

策略与价值:

①reward、return、value，用来评估一个策略的好坏。

策略与价值一一对应。

②价值比较，策略比较，策略改进定理。

③强化学习的终极目标，求取最优策略，

最优策略不唯一，最优价值唯一。

最优动作价值，意味着选取这个动作，未来回报的期望最大。

④r线性变换，V+Q线性变换，改变最优价值，不改变greedy最优策略。

⑤迭代时，最优策略可能已经稳定了，但是对应的最优价值还没稳定。

⑥从终止状态反向迭代更新价值，速度更快。但是哪里是终止状态？上帝视角。

$$p(s', r | s, a) = \Pr\{S_t = s', R_t = r | S_{t-1} = s, A_{t-1} = a\}$$

$$\sum_{s' \in S} \sum_{r \in R} p(s', r | s, a) = 1$$

$$p(s' | s, a) = \sum_{r \in R} p(s', r | s, a)$$

$$r(s, a) = \sum_{s' \in S} \sum_{r \in R} (p(s', r | s, a) * r)$$

ch2: Bellman Equations

Static Relationship

实质：描述状态值之间的静态关系（单项形式、矩阵形式）

求解：（矩阵求逆、数值迭代） — （policy-evaluation）

$$\begin{aligned}
 v_{\pi}(s) &= E_{\pi} [G_t \mid S_t = s] && (Definition) \\
 &= E_{\pi} [R_{t+1} + \gamma G_{t+1} \mid S_t = s] && (TD - 0) \\
 &= E_{\pi} [R_{t+1} + \gamma R_{t+2} + \gamma^2 R_{t+3} + \dots \mid S_t = s] && (TD - n)(TD - \infty = MC) \\
 &= \sum_{a \in A} \pi(a \mid s) \sum_{s', r} p(s', r \mid s, a) * [r + \gamma E_{\pi} [G_{t+1} \mid S_{t+1} = s']] \\
 &= \sum_{a \in A} \pi(a \mid s) \sum_{s', r} p(s', r \mid s, a) * [r + \gamma v_{\pi}(s')] && (BEs) \\
 &= \sum_{a \in A} \pi(a \mid s) * q_{\pi}(s, a)
 \end{aligned}$$

$$\begin{aligned}
 q_{\pi}(s, a) &= E_{\pi} [G_t \mid S_t = s, A_t = a] && (Definition) \\
 &= E_{\pi} [R_{t+1} + \gamma G_{t+1} \mid S_t = s, A_t = a] && (TD - 0) \\
 &= E_{\pi} [R_{t+1} + \gamma R_{t+2} + \gamma^2 R_{t+3} + \dots \mid S_t = s, A_t = a] && (TD - n)(TD - \infty = MC) \\
 &= \sum_{s', r} p(s', r \mid s, a) * [r + \gamma E_{\pi} [G_{t+1} \mid S_{t+1} = s']] \\
 &= \sum_{s', r} p(s', r \mid s, a) * [r + \gamma v_{\pi}(s')] \\
 &= \sum_{s', r} p(s', r \mid s, a) * \left[r + \gamma \sum_{a' \in A} \pi(a' \mid s') * q_{\pi}(s', a') \right] && (BEs)
 \end{aligned}$$

policy-comparison:

$$\pi' \geq \pi \quad \longleftrightarrow \quad v_{\pi'}(s) \geq v_{\pi}(s) \quad \forall s \in S$$

policy-improvement:

$$E_{\pi'} [q_{\pi}(s, \pi'(s))] \geq v_{\pi}(s) = E_{\pi} [q_{\pi}(s, \pi(s))] \quad \forall s \in S$$

Bellman Optimal Equations:

$$\begin{aligned} v_*(s) &= \max_{\pi} v_{\pi}(s) && (Definition) \\ &= \max_{\pi} (r_{\pi} + \gamma P_{\pi} v) && (BOEs) \\ &= \max_{a \in A} q_{\pi^*}(s, a) && \forall s \in S \end{aligned}$$

Contraction Mapping Theorem (迭代收敛至唯一不动点)
 贝尔曼最优方程的收缩迭代过程，即是value iteration算法

ch3: Dynamic Programming

理解:

Model-based. Dynamics with Model p .

①已知模型 p , 给定策略 π , 解BEs, 得到价值 V 。

两种解法: 矩阵求逆、数值迭代。

(1) Policy Iteration:

Policy Evaluation:

$$v_{\pi_k} = r_{\pi_k} + \gamma P_{\pi_k} v_{\pi_k}$$

Policy Improvement:

$$\pi_{k+1} = \arg \max_{\pi} (r_{\pi} + \gamma P_{\pi} v_k)$$

(2) Value Iteration:

$$v_{k+1} = \max_{\pi} (r_{\pi} + \gamma P_{\pi} v_k)$$

Policy Update:

$$\pi_{k+1} = \arg \max_{\pi} (r_{\pi} + \gamma P_{\pi} v_k)$$

Value Update:

$$v_{k+1} = r_{\pi_{k+1}} + \gamma P_{\pi_{k+1}} v_k$$

(3) Turncated Iteration:

值迭代有限次数（介于1次与无穷次之间）；值也未稳定，就进行策略改进

ch4: Monte Carlo

Sample:

$$\begin{aligned}v_{\pi}(s) &= E_{\pi} [G_t = R_{t+1} + \gamma R_{t+2} + \gamma^2 R_{t+3} + \dots \mid S_t = s] & (TD-\text{ } \bowtie = MC) \\q_{\pi}(s, a) &= E_{\pi} [G_t = R_{t+1} + \gamma R_{t+2} + \gamma^2 R_{t+3} + \dots \mid S_t = s, A_t = a] & (TD-\text{ } \bowtie = MC)\end{aligned}$$

理解:

Model-free. Dynamics with Trajectory.

- ①采样进行估计，依据概率论的大数定理。
- ②episode长度（探索半径是否覆盖终点？）对估值影响，最优价值是否反向传播。
- ③估计的更新方式，非增长式（等着一起算）和增长式（来一个算一个）。
- ④epsilon关乎采样策略的探索性和最优性，
大则探索性强、最优性弱，小则探索性弱、最优性强，
- ⑤如果epsilon大到一定程度，可能会导致epsilon-greedy与最优greedy不一致。

(1) MC-Basic

二次循环，遍历所有(s, a)；某个策略下，每对采足够样，非增长式估计相应Q。
策略相应的，一套稳定Q值下，策略改进。
迭代。

(2) MC-Exploring-Starts

起始分布，覆盖(s, a)全集。

Pi下，充分利用每一个trajectory里的所有(s, a)对，访问，增长式估计相应Q。
每一个trajectory结束后，Q值未必稳定，都进行策略改进。
迭代。

(3) MC-epsilon-greedy

过程分布，覆盖(s, a)全集。

e-Pi下，充分利用每一个trajectory里的所有(s, a)对，访问，增长式估计相应Q。
每一个trajectory结束后，Q值未必稳定，都进行策略改进，生成e-Pi。
迭代。

*stochastic approximation

理解:

以某形式的公式，为理论依据，进行实际采样与近似估计。

(1) Incremental-Estimation:

$$w_k = \frac{1}{k} \sum_{i=1}^k x_i \qquad w_{k-1} = \frac{1}{k-1} \sum_{i=1}^{k-1} x_i$$
$$w_k = \frac{1}{k} [(k-1)w_{k-1} + x_k] = w_{k-1} + \frac{1}{k} [x_k - w_{k-1}]$$

(2) Robbins-Monro:

$$g(w) = 0 \quad (\text{g is unknown, w is input, 0 is output})$$

$$g(w) = \nabla_w L(w) = 0 \quad (\text{w is parameters})$$

$$g(w) = L(w) - C = 0$$

$$w^* \text{ is the solution (Convergence Condition)}$$

$$w_{k+1} = w_k + a_k [\tilde{g}(w_k, \eta_k) - 0]$$

$$= w_k + a_k (g(w_k) + \eta_k)$$

$$\text{iteration : } \{w_k\} + \{\tilde{g}_k\} + \{a_k\}$$

(3) Optimazation:

$$\min_w J(w) = E[f(w, X)]$$

$$\Rightarrow \Rightarrow \quad \nabla_w E[f(w, X)] = 0$$

$$\Rightarrow \Rightarrow \quad \text{Gradient : } \text{InputSpace}, \quad \text{direction} + \text{magnitude}$$

GD + (Mini)Batch GD + Stochastic GD:

$$w_{k+1} = w_k - \alpha_k \nabla_w E[f(w_k, X)] = w_k - \alpha_k E[\nabla_w f(w_k, X)]$$

$$w_{k+1} = w_k - \alpha_k \frac{1}{n} \sum_{i=1}^n \nabla_w f(w_k, x_i)$$

$$w_{k+1} = w_k - \alpha_k \nabla_w f(w_k, x_k)$$

ch5: Temporal Difference

理解:

Model-free. Dynamics with Transition.

①TD时序差分: 在不同时刻, 对同一个量估计, 有差, 利用差改进估计。

②没有模型 p 、只有数据 t , 进行估计:

MC利用整条trajectory, 估计 V 、 Q , 离线/无偏/大方差;

TD利用片段transition, 估计 V 、 Q , 在线/有偏/小方差。

③SARSA, 用TD估计某个 P_i 的 Q 。

④Q-Learning, 用TD直接估计 Q^* 。(异轨, 从数据中提取共性MDP- Q^* 信息)

⑤行为策略采集数据, 利用数据计算TD-target (数据组织形式),

目标策略的估计值趋向TD-target, 即TD-target代表着目标策略。

⑥on-policy: 用一个策略和环境交互, 得到experience, 估计这个策略, 改进这个策略。

再用改进的策略循环, 交互、估计、改进, 直至最优 (这个系列的最优)。

off-policy: 一个策略和环境交互, 得到experience,

另一个策略被估计、被改进。(目标策略不易采集数据, 所以需要行为策略, 异轨)

experience应当具有一些性质, 能够架通两个策略,

行为策略能够采到, 目标策略能够使用。

(1) TD-V: s-a-r-s

$$v_{t+1}(s_t) = v_t(s_t) + \alpha_t [[r_{t+1} + \gamma v_t(s_{t+1})] - v_t(s_t)]$$

$$v_{t+1}(s) = v_t(s) \quad \forall s \neq s_t$$

(2) TD-Q: SA-R-SA

Policy Evaluation:

$$q_{t+1}(s_t, a_t) = q_t(s_t, a_t) + \alpha_t(s_t, a_t)[r_{t+1} + \gamma q_t(s_{t+1}, a_{t+1}) - q_t(s_t, a_t)]$$

$$q_{t+1}(s, a) = q_t(s, a) \quad \forall (s, a) \neq (s_t, a_t)$$

Policy Improvement:

某个策略下的Q值可能还未估计稳定,
就可以进行策略提升。最终目标提升至Q*。

(3) TD-Q*: Q*-Learning (s-a-r-s)

Value Improvement:

$$q_{t+1}(s_t, a_t) = q_t(s_t, a_t) + \alpha_t(s_t, a_t)[r_{t+1} + \gamma \max_a q_t(s_{t+1}, a) - q_t(s_t, a_t)]$$

$$q_{t+1}(s, a) = q_t(s, a) \quad \forall (s, a) \neq (s_t, a_t)$$

*(DP+MC+TD)-Summary:

value	DP-model(p)	TD-data(t)
v	Policy-Evaluation(vBEs)	TD(0/n/...)
v*	Value-iteration(vBOEs)	-
q	Policy-Evaluation(qBEs)	TD(0)=SARSA
q*	Value-iteration(qBOEs)	Q*-Learning

ch6: Value Function Approximation

理解:

①函数拟合的优势: 用更少的参数量 (存储), 估计, 更多的状态量 (泛化);
原因在于, 函数这种表达形式, 存储了更本质的信息。

函数拟合的劣势: 存在精度损失。

函数拟合的关键: 最优的结构 + 最优的参数, 提升拟合准确程度。

②状态分布(.): 平均分布U、稳态分布D ($D + (\pi + p) = D$)。

优化时, SGD将不同分布吸收拉平了。

③某个真实分布中的一系列真值, 形成一个超面;

经过各种采样, 得到超面的部分观测真值;

建立一个模型并优化参数, 用来拟合部分观测真值得到的超面。

④自举, 让不均匀的高低估扩展, 可能导致最优性发生变化。

(1) Objective Function:

$$J(w) = E_{S \sim (\cdot)} [(v_\pi(S) - \tilde{v}(S, w))^2]$$

(2) Optimazation:

$$\begin{aligned} w_{k+1} &= w_k - \alpha_k \nabla_w J(w_k) \\ w_{t+1} &= w_t + \alpha_t (v_\pi(s_t) - \tilde{v}(s_t, w_t)) \nabla_w \tilde{v}(s_t, w_t) \end{aligned}$$

(3) Algorithm:

$$\begin{aligned} w_{t+1} &= w_t + \alpha_t (g_t - \tilde{v}(s_t, w_t)) \nabla_w \tilde{v}(s_t, w_t) & MC + VA \\ w_{t+1} &= w_t + \alpha_t [r_{t+1} + \gamma \tilde{v}(s_{t+1}, w_t) - \tilde{v}(s_t, w_t)] \nabla_w \tilde{v}(s_t, w_t) & TD + VA \end{aligned}$$

(4) Approximator:

$$\begin{aligned} linear &= kx + b \\ nonlinear &= neural \ network \end{aligned}$$

(5) SARSA + VA: (SA-R-SA + q- + w)

Value Update(w):

$$w_{t+1} = w_t + \alpha_t [r_{t+1} + \gamma \tilde{q}(s_{t+1}, a_{t+1}, w_t) - \tilde{q}(s_t, a_t, w_t)] \nabla_w \tilde{q}(s_t, a_t, w_t)$$

Policy Improvement: epsilon-greedy

(6) Q*-Learning + VA: (S-A-R-S + q- + w)

Value Update(w):

$$w_{t+1} = w_t + \alpha_t [r_{t+1} + \gamma \max_a \tilde{q}(s_{t+1}, a, w_t) - \tilde{q}(s_t, a_t, w_t)] \nabla_w \tilde{q}(s_t, a_t, w_t)$$

Policy Improvement: (epsilon-)greedy

(7) Deep Q*-Learning: (Buffer(S-A-R-S) + NN(w)*2)

Value Update(w):

$$w_{t+1} = w_t + \alpha_t [r_{t+1} + \gamma \max_a \tilde{q}(s_{t+1}, a, w_T) - \tilde{q}(s_t, a_t, w_t)] \nabla_w \tilde{q}(s_t, a_t, w_t)$$

Policy Improvement: (epsilon-)greedy

(8) Double Q*-Net: (Buffer(S-A-R-S) + NN(w)*2)