强化学习 第一讲.

O_t, S_t^a, S_t^e:
- Full observation → $O_t = S_t^a = S_t^e \Rightarrow$ MDP (Markov decision process)
- Partial observation → $S_t^a \neq S_t^e \Rightarrow$ POMDP

agent:
- policy.
- value function:
- (Model ~~model free~~ ~~model~~
  - Transitions model: $P_{ss'}^a = P[S'=s' \mid S=s, A=a]$.
  - Rewards model: $R_s^a = E[R \mid S=s, A=a]$.

value based RL ~~agent~~ :
- No Policy (或称为隐含)
- Value function

policy based RL ~~agent~~ :
- Policy
- No value function

Actor Critic RL :
- Policy
- value function.

Model free RL :
- Policy and/or Value function
- No Model.

Model based RL :
- Policy and/or Value function
- Model.

- Exploration : 探索 (随机)
- Exploitation : 开发 (最好的).

Prediction problem : evaluate the future.

⇓

Control problem : Find best policy.

强化学习第二讲

Markov Process (Markov Chain) : $\langle S, P \rangle$

Markov Reward Process (MRP) : $\langle S, P_{ss'}, R, \gamma \rangle$

Markov Desision Process (MDP) : $\langle S, A, P_{ss'}^{a}, R_{s}^{a}, \gamma \rangle$

	MRP $\langle S,P,R,\gamma \rangle$	MDP $\langle S,A,P,R,\gamma \rangle$
S定义	finate set of state.	finate set of state.
A定义	无A	finate set of actions.
P定义.	state transition probability matrix	state transition probability matrix.
P方程	$P_{ss'} = P[S_{t+1} = s' \mid S_t = s]$	$P_{ss'}^{a} = P[S_{t+1} = s' \mid S_t = s, A_t = a]$
R定义	reward function	reward function
R方程	$R_s = E[R_{t+1} \mid S_t = s]$	$R_s^a = E[R_{t+1} \mid S_t = s, A_t = a]$
γ定义	discount factor $\in [0,1]$	discount factor $\in [0,1]$
G_t定义	$G_t = R_{t+1} + \gamma R_{t+2} + \gamma^2 R_{t+3} + \cdots$	$G_t = R_{t+1} + \gamma R_{t+2} + \gamma^2 R_{t+3} + \cdots$
$V(s)$	$V(s) = E[G_t \mid S_t = s]$	$V_\pi(s) = E_\pi[G_t \mid S_t = s]$
Bellman	$V(s) = E[G_t \mid S_t = s] = E[R_{t+1} + \gamma V(S_{t+1}) \mid S_t = s]$	$V_\pi(s) = E_\pi[R_{t+1} + \gamma V_\pi(S_{t+1}) \mid S_t = s]$
	$V(s) = R_s + \gamma \sum_{s \in S} P_{ss'} V(s')$.	$V_\pi(s) = \sum_{a \in A} \pi(a \mid s)\, q_\pi(s,a)$.
$q_\pi(s,a)$	无.	$q_\pi(s,a) = E_\pi[G_t \mid S_t = s, A_t = a]$
		$q_\pi(s,a) = E_\pi[R_{t+1} + \gamma q_\pi(S_{t+1}, A_{t+1}) \mid S_t = s, A_t = a]$
		$q_\pi(s,a) = R_s^a + \gamma \sum_{s' \in S} P_{ss'}^a V_\pi(s')$.
$V_*(s)$定义		$V_*(s) = \max_\pi V_\pi(s)$.
$q_*(s,a)$定义.		$q_*(s,a) = \max_\pi q_\pi(s,a)$
$V_*(s)$.		$V_*(s) = \max_a q_*(s,a)$.
$q_*(s,a)$		$q_*(s,a) = R_s^a + \gamma \sum_{s' \in S} P_{ss'}^a V_*(s')$.

强化学习 第二讲

Markov Process (Markov Chain) : $<S, P>$.

MRP : $<S, P_{ss'}, R, \gamma>$

Bellman Equation : $\begin{bmatrix} v_1 \\ \vdots \\ v_n \end{bmatrix} = \begin{bmatrix} R_1 \\ \vdots \\ R_n \end{bmatrix} + \gamma \begin{bmatrix} P_{11} & \cdots & P_{1n} \\ \vdots & & \\ P_{n1} & \cdots & P_{nn} \end{bmatrix} \begin{bmatrix} V(1) \\ \vdots \\ V(n) \end{bmatrix} \xrightarrow[\text{method}]{\text{iterative}} \begin{cases} \text{Dynamic programming} \\ \text{Mote-Carlo evaluation} \\ \text{temporal-difference leaning} \end{cases}$

MDP : $<S, A, P^a_{ss'}, R^a_s, \gamma>$.

Policy (π) : $\pi(a|s) = P[A_t = a \mid S_t = s]$.

State-value function : (following policy π).

$$V_\pi(s) = E_\pi[G_t \mid S_t = s] \overset{\text{Bellman}}{=\!=} E_\pi[R_{t+1} + \gamma V_\pi(S_{t+1}) \mid S_t = s].$$

~~State~~ action-value function : (following policy π).

$$q_\pi(s,a) = E_\pi[G_t \mid S_t = s, A_t = a] \overset{\text{Bellman}}{=\!=} E_\pi[R_{t+1} + \gamma q_\pi(S_{t+1}, A_{t+1}) \mid S_t = s, A_t = a]$$

V和q的关系 $\begin{cases} V_\pi(s) = \sum_{a \in A} \pi(a|s)\, q_\pi(s,a). \\ q_\pi(s,a) = R^a_s + \gamma \sum_{s' \in S} P^a_{ss'} V_\pi(s'). \end{cases}$

$$\begin{cases} V_\pi(s) = \sum_{a \in A} \pi(a|s) \left(R^a_s + \gamma \sum_{s' \in S} P^a_{ss'} v_\pi(s') \right), \\ q_\pi(s,a) = R^a_s + \gamma \sum_{s' \in S} P^a_{ss'} \sum_{a' \in A} \pi(a'|s')\, q_\pi(s', a'). \end{cases}$$

Optimal state value function :

$$V_*(s) = \max_\pi V_\pi(s).$$

optimal action-value function :

$$q_*(s,a) = \max_\pi q_\pi(s,a)$$

强化学习 第三讲 动态规化.

动态规化(DP) $\xrightarrow{\text{solve}}$ planning $\begin{cases} \text{predict} & \begin{cases} \text{Input: MDP and policy } \pi \\ \text{Output: value function } V_\pi \end{cases} \\ \text{control} & \begin{cases} \text{Input: MDP} \\ \text{output: } V_* \rightarrow \pi_* \end{cases} \end{cases}$

evaluation (using synchronous backups):

At each iteration k++;

for all state $s \in S$:

update $V_{k+1}(s)$ from $V_k(s')$ ~~using~~ where s' is ~~the~~ a successor state of s.

Policy Iteration:

$$\begin{tikzcd} \pi \arrow[r, bend left, "\substack{\text{evaluation} \\ V \to V^\pi}"] & V \arrow[l, bend left, "\substack{\pi \to \text{greedy}(V) \\ \text{improvement}}"] \arrow[r, Rightarrow, "\text{Iteration}"] & \pi^* \rightleftarrows V^* \end{tikzcd}$$

greedy: $\pi'(s) = \underset{a \in A}{\text{argmax}}\, q_\pi(s,a)$.

证明:(?) 收敛到 π^* 和 V^*

Value Iteration: At each iteration k+1.

For all state $s \in S$:

Update $V_{k+1}(s)$ from $V_k(s')$.

Problem	Bellman Equation	Algorithm
Predict	Bellman Expectation Equation	Iteration Policy Evaluation
Contral	Bellman Expectation Equation + Greedy Policy Improvement	Policy Iteration
Contral	Bellman Optimality Equation	Value Iteration

强化学习第四讲. 无模型预测

无模型 ⇔ 无MDP

$$V(S_t) \leftarrow V(S_t) + \frac{1}{N(S_t)}\left(G_t - V(S_t)\right).$$

⇓ 忘记过去很久的.

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

TD: $$V(S_t) \leftarrow V(S_t) + \alpha\left(\underbrace{R_{t+1} + \gamma V(S_{t+1})}_{\text{Bellman Equation}} - V(S_t)\right)$$

↳ Bellman Equation → 在MDP中TD更有效.

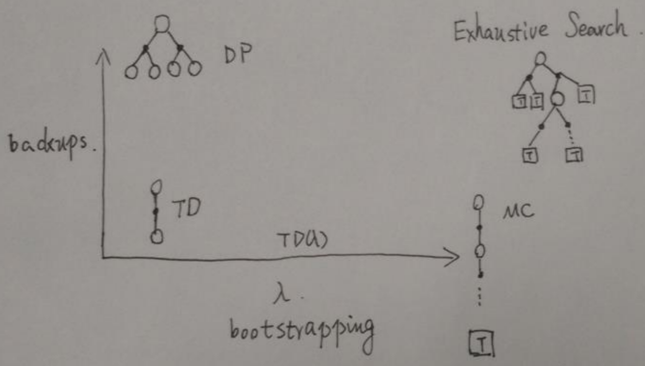
收敛于对数据解释最好的MDP.

TD: $$V(S_t) \leftarrow V(S_t) + \alpha\left(\underbrace{R_{t+1} + \gamma V(S_{t+1})}_{\text{Bellman Equation}} - V(S_t)\right)$$

↳ Bellman Equation → 在MDP中TD更有效.

收敛于对数据解释最好的MDP.

DP: $$V(S_t) \leftarrow V(S_t) + E_\pi\left[R_{t+1} + \gamma V(S_{t+1})\right].$$



TD(λ). $$G_t^\lambda = (1-\lambda)\sum_{n=1}^{\infty} \lambda^{n-1} G_t^{(n)}$$

combine all n-step return

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