

Reinforcement learning

2023年10月3日 星期二 下午9:33

Math : set & functions. Metric \Rightarrow distance & convergence.

System $\left\{ \begin{array}{l} \text{Identify pieces} \\ \text{Assign possible states.} \end{array} \right. \Rightarrow \text{"Learning"}$

给定当前状态，是否能够预测下一个状态的系统。 (变化建模)

Discrete: $f(n) = f(0) + \sum_{k=1}^n (f(k) - f(k-1))$ Calculus: $f(x) = f(0) + \int_0^x f'(t) dt$.

PDE: 揭示了变化之间的关系。

Probability: 为我们不知道的东西建模。

Stochastic Calculus: 给不确定的量建模。

#42 Bellman Equation:

控制问题 |
- Agent采取行动。
- Dynamics + Objective.

$$V(x) = \sup_a \left\{ R(x, a) + \lambda \sum_{z \in S} p_a(x, z) V(z) \right\}$$

$\overbrace{\quad}$
奖励

$\overbrace{\quad}$
其他地方的 potential.

强化学习

给出 (S, A, R) 如何 optimally 行动?
 $\begin{array}{c} / \quad | \quad \backslash \\ \text{State} \quad \text{Action} \quad \text{Reward.} \end{array}$

例: Playing Boardgame [离散状态] [离散行为]

交通控制

金融行为

- 状态空间: chokers 的位置

-

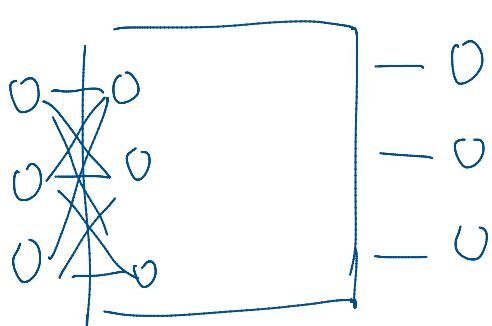
- 行为是 bavarian 的规则.

- winning \rightarrow positive, losing \rightarrow negative

理论上, 人们可以在 Bellman 方程的矩阵中迭代 contraction mapping, 来找到唯一的 $V(x)$.

但实际上它难以实现。

RL: 神经网络。



· 随机化是关键!

· 所有的东西以用 Markov 模型

$x_0, x_1 \in \{0, 1\}$ uniformly 但独立。

$$x_2 = \begin{cases} 1 & x_1 = 1 \& x_0 = 1 \\ 0 & \text{otherwise} \end{cases}$$

虽然只要知道了 x_0 , 这就是确定的. 但也可以用 Markov 模型

$$\begin{cases} p(x_2=1 | x_1=1) = \frac{1}{2} \\ p(x_2=1 | x_1=0) = 0 \end{cases}$$

而对于宽泛的值, Q-learning.

$$V(x) = \sup_{a \in A} \underbrace{\{R(x, a) + \lambda \sum P_a(x, z) V(z)\}}_{Q(x, a)} \quad \text{learn to model } Q(x, a) \text{ 而不是 } V(x)$$

然后最优行为是 $\arg\max_{a \in A} Q(x, a)$

$$\text{回代: } V(x) = \sup_{\alpha} \mathbb{E}_x [R(x^\alpha, \alpha(x^\alpha))]$$

步骤: ① 从环境中学习/建模, 即对于转移矩阵的建构.

②一开始选择随机的一个动作. 最好是随机化的.

③ 目标是逼近一个有用的 Q (也可以反过来用 V).

- 蒙特卡罗: 从①生成很多道路. 把 变成平均的 rewards

(Advantage: versatile. Disad: 高偏差的)

- TD: 只产生一步 set $Q(x, a) \leftarrow Q(x, a) + \alpha [R + \lambda Q(z, a') - Q(x, a)]$.