强化学习及其应用

Reinforcement Learning and Its Applications

第七章 模型与规划

Model and Planning

授课人：周晓飞
zhouxiaofei@iie.ac.cn

2023-6-15

第七章 模型与规划

第七章 模型与规划

模型学习

问题描述

- *Last lecture*: learn policy directly from experience
- *Previous lectures*: learn value function directly from experience
- *This lecture*: learn model directly from experience
- and use planning to construct a value function or model
- Integrate learning and planning into a single architecture

模型学习

问题描述

- *Last lecture*: learn policy directly from experience
- *Previous lectures*: learn value function directly from experience
- *This lecture*: learn model directly from experience
- and use planning to construct a value function or model
- Integrate learning and planning into a single architecture

- Model-Free RL
  - No model
  - Learn value function (and/or policy) from experience
- Model-Based RL
  - Learn a model from experience
  - Plan value function (and/or policy) from model

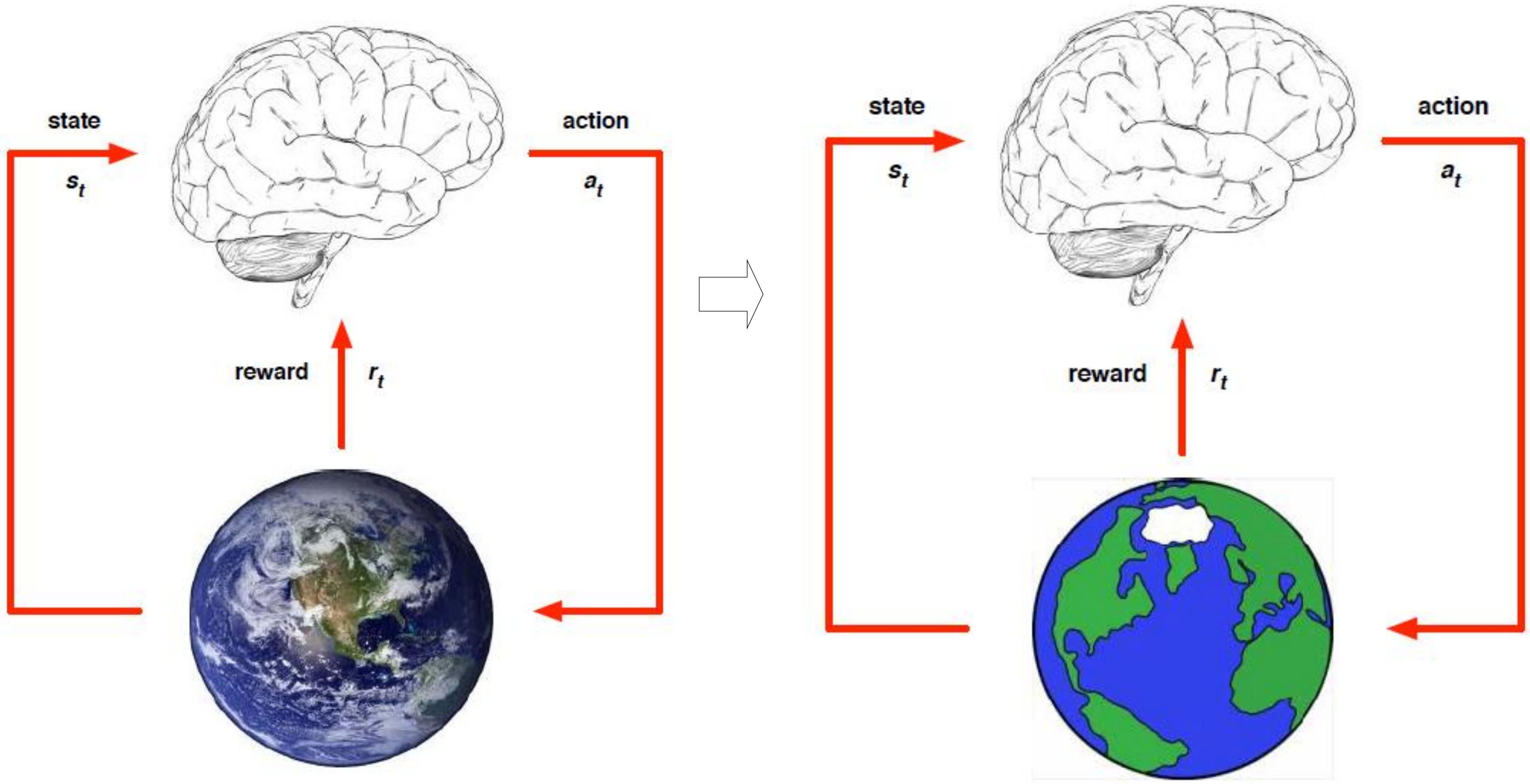
模型学习

问题描述

- *Last lecture*: learn policy directly from experience
- *Previous lectures*: learn value function directly from experience
- *This lecture*: learn model directly from experience
- and use planning to construct a value function or model
- Integrate learning and planning into a single architecture

- Model-Free RL
  - No model
  - Learn value function (and/or policy) from experience
- Model-Based RL
  - Learn a model from experience
  - Plan value function (and/or policy) from model

问题描述



模型学习

问题描述

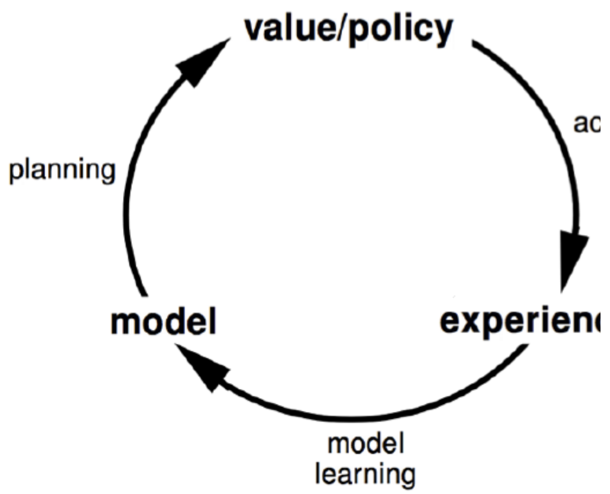
基本流程：源于已知 episodes from experiences，学习 Model,
产生 episodes from Model,
由产生的 episodes，学习 Policy 和 Value。

Advantages:

- Can efficiently learn model by supervised learning methods
- Can reason about model uncertainty

Disadvantages:

- First learn a model, then construct a value function
  $\Rightarrow$ two sources of approximation error



模型学习

什么是模型?

- A *model* $\mathcal{M}$ is a representation of an MDP $\langle \mathcal{S}, \mathcal{A}, \mathcal{P}, \mathcal{R} \rangle$, parametrized by η
- We will assume state space $\mathcal{S}$ and action space $\mathcal{A}$ are known
- So a model $\mathcal{M} = \langle \mathcal{P}_\eta, \mathcal{R}_\eta \rangle$ represents state transitions $\mathcal{P}_\eta \approx \mathcal{P}$ and rewards $\mathcal{R}_\eta \approx \mathcal{R}$

$$
\begin{aligned}
s_{t+1} &\sim \mathcal{P}_\eta(s_{t+1} \mid s_t, a_t) \\
r_{t+1} &= \mathcal{R}_\eta(r_{t+1} \mid s_t, a_t)
\end{aligned}
$$

- Can assume conditional independence between state transitions and rewards

$$
\mathbb{P}\left[s_{t+1}, r_{t+1} \mid s_t, a_t\right] = \mathbb{P}\left[s_{t+1} \mid s_t, a_t\right] \mathbb{P}\left[r_{t+1} \mid s_t, a_t\right]
$$

模型学习

模型学习过程

- Goal: estimate model $\mathcal{M}_\eta$ from experience $\{s_1, a_1, r_2, ..., s_T\}$
- This is a supervised learning problem

$$
\begin{aligned}
s_1, a_1 &\rightarrow r_2, s_2 \\
s_2, a_2 &\rightarrow r_3, s_3 \\
&\vdots \\
s_{T-1}, a_{T-1} &\rightarrow r_T, s_T
\end{aligned}
$$

- Learning $s, a \rightarrow r$ is a *regression* problem
- Learning $s, a \rightarrow s'$ is a *density estimation* problem
- Pick loss function, e.g. mean-squared error, KL divergence, ...
- Find parameters η that minimise empirical loss

模型学习

模型函数种类

- Table Lookup Model
- Linear Expectation Model
- Linear Gaussian Model
- Gaussian Process Model
- Deep Belief Network Model
- ...

例子：Table Lookup Model

- Model is an explicit MDP, $\hat{\mathcal{P}}, \hat{\mathcal{R}}$
- Count visits $N(s, a)$ to each state action pair

$$\hat{\mathcal{P}}^a_{s,s'} = \frac{1}{N(s,a)} \sum_{t=1}^{T} \mathbf{1}(s_t, a_t, s_{t+1} = s, a, s')$$

$$\hat{\mathcal{R}}^a_{s} = \frac{1}{N(s,a)} \sum_{t=1}^{T} \mathbf{1}(s_t, a_t = s, a) r_t$$

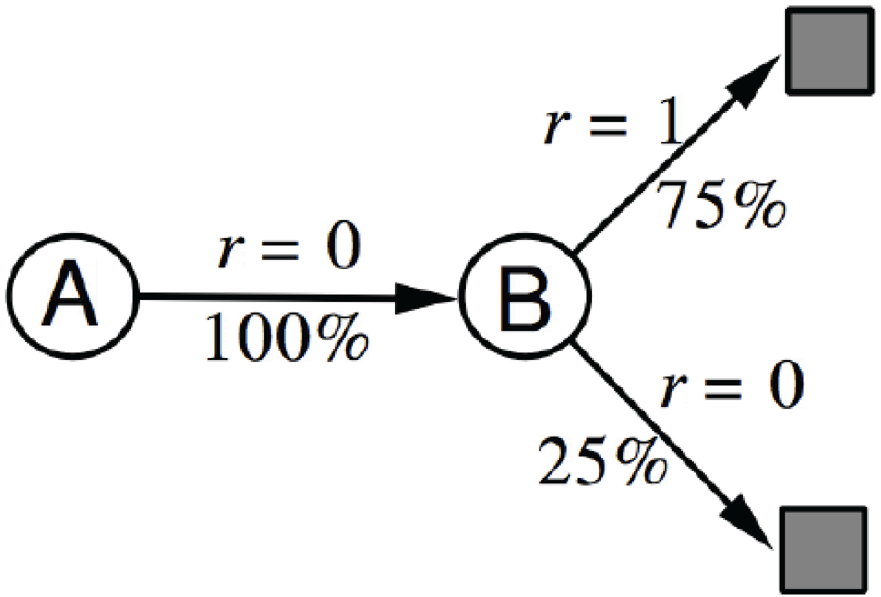
- Alternatively
  - At each time-step t, record experience tuple $\langle s_t, a_t, r_{t+1}, s_{t+1} \rangle$
  - To sample model, randomly pick tuple matching $\langle s, a, \cdot, \cdot \rangle$

例子：Table Lookup Model

AB Example

Two states A, B; no discounting; 8 episodes of experience

A, 0, B, 0
B, 1
B, 1
B, 1
B, 1
B, 1
B, 1
B, 0



We have constructed a table lookup model from the experience

模型学习

AB 例子：没有 a. 要估计 $P_{ss'}$, $S=\{A,B\}$；估 $P[r|S]$, $r=\{0,1\}$.

A, 0, B, 0
B, 1
B, 1
B, 1
B, 1
B, 1
B, 1
B, 0

$P(S'=B|S=A)=100\%$

~~P(S'~~

$P(r=0|S=A)=100\%$.

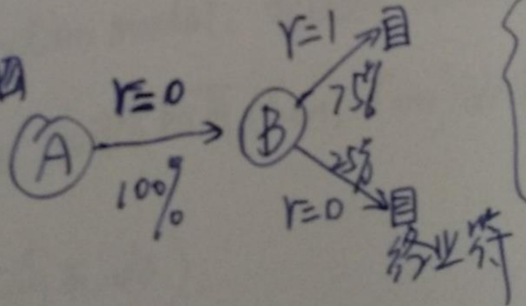
$P(r=1|S=B)=75\%$

$P(r=0|S=B)=25\%$

$R_{S=B}=E[r|S=B]=1\times75\%+0\times25\%=0.75$

$R_{S=A}=E[r|S=A]=100\times0=0$

省略一个终止符



⑥

第七章 模型与规划

Planning with a Model

- A simple but powerful approach to planning
- Use the model only to generate samples
- Sample experience from model

$$S_{t+1} \sim \mathcal{P}_\eta(S_{t+1} \mid S_t, A_t)$$
$$R_{t+1} = \mathcal{R}_\eta(R_{t+1} \mid S_t, A_t)$$

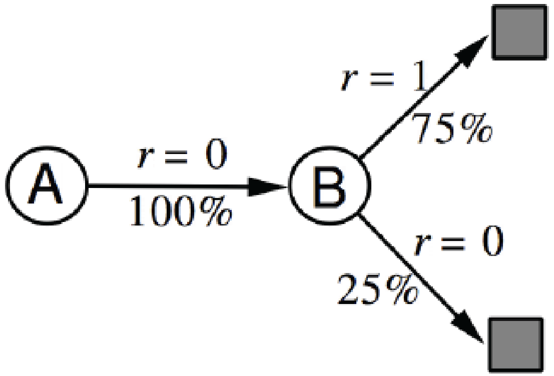
- Apply model-free RL to samples, e.g.:
  - Monte-Carlo control
  - Sarsa
  - Q-learning
- Sample-based planning methods are often more efficient

Planning with a Model

AB Example

- Construct a table-lookup model from real experience
- Apply model-free RL to sampled experience

Real experience			
A,	0,	B,	0
B,	1		
B,	1		
B,	1		
B,	1		
B,	1		
B,	1		
B,	0		



Sampled experience			
B,	1		
B,	0		
B,	1		
A,	0,	B,	1
B,	1		
A,	0,	B,	1
B,	1		
B,	0		

e.g. Monte-Carlo learning: $V(A) = 1, V(B) = 0.75$

第七章 模型与规划

7.1 模型学习

7.2 模型与规划

7.3 算法

算法

Dyna

We consider two sources of experience

Real experience Sampled from environment (true MDP)

$$s' \sim \mathcal{P}^a_{s,s'}$$
$$r = \mathcal{R}^a_s$$

Simulated experience Sampled from model (approximate MDP)

$$s' \sim \mathcal{P}_\eta(s' \mid s, a)$$
$$r = \mathcal{R}_\eta(r \mid s, a)$$

算法

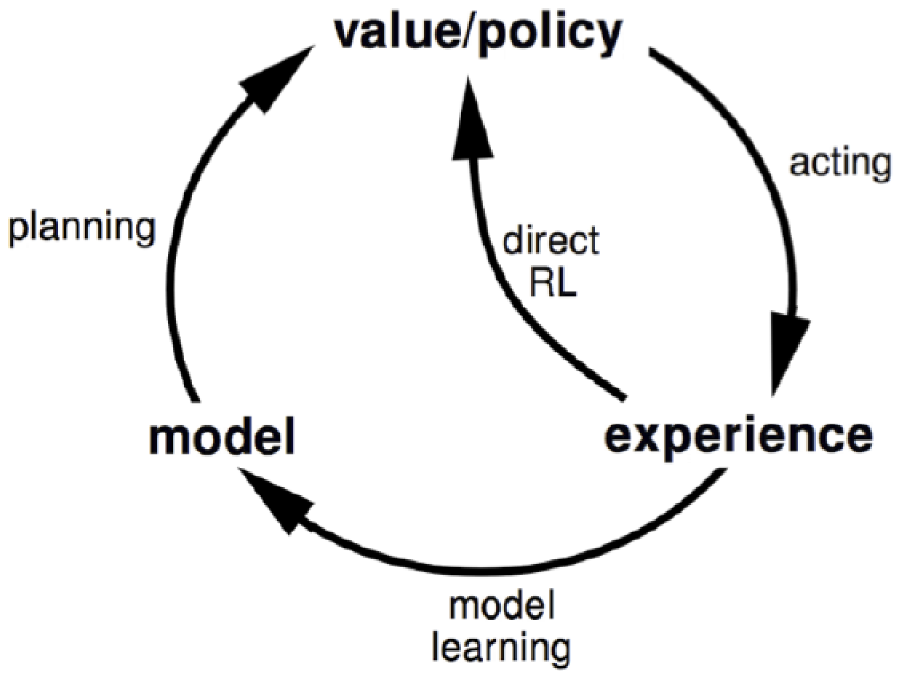
Dyna

- Model-Free RL
  - No model
  - Learn value function (and/or policy) from real experience
- Model-Based RL (using Sample-Based Planning)
  - Learn a model from real experience
  - Plan value function (and/or policy) from simulated experience
- Dyna
  - Learn a model from real experience
  - Learn and plan value function (and/or policy) from real and simulated experience

Dyna 方法综合了 Model-Free 和 Model-Based.

算法

Dyna



value/policy
acting
planning
direct
RL
model
experience
model
learning

Dyna-Q Algorithm

Initialize $Q(s, a)$ and $Model(s, a)$ for all $s \in \mathcal{S}$ and $a \in \mathcal{A}(s)$

Do forever:

(a) $s \leftarrow$ current (nonterminal) state

(b) $a \leftarrow \varepsilon$-greedy(s, Q)

(c) Execute action a; observe resultant state, s', and reward, r

(d) $Q(s, a) \leftarrow Q(s, a) + \alpha[r + \gamma \max_{a'} Q(s', a') - Q(s, a)]$

(e) $Model(s, a) \leftarrow s', r$ (assuming deterministic environment)

(f) Repeat N times:

$s \leftarrow$ random previously observed state

$a \leftarrow$ random action previously taken in s

$s', r \leftarrow Model(s, a)$

$Q(s, a) \leftarrow Q(s, a) + \alpha[r + \gamma \max_{a'} Q(s', a') - Q(s, a)]$

Dyna-Q Algorithm

Initialize $Q(s,a)$ and $Model(s,a)$ for all $s \in \mathcal{S}$ and $a \in \mathcal{A}(s)$

Do forever:

(a) $s \leftarrow$ current (nonterminal) state

(b) $a \leftarrow \varepsilon$-greedy(s, Q)

(c) Execute action a; observe resultant state, s', and reward, r

(d) $Q(s,a) \leftarrow Q(s,a) + \alpha[r + \gamma \max_{a'} Q(s',a') - Q(s,a)]$

(e) $Model(s,a) \leftarrow s', r$ (assuming deterministic environment)

(f) Repeat N times:

$s \leftarrow$ random previously observed state

$a \leftarrow$ random action previously taken in s

$s', r \leftarrow Model(s,a)$

$Q(s,a) \leftarrow Q(s,a) + \alpha[r + \gamma \max_{a'} Q(s',a') - Q(s,a)]$

Model-Free

Model-Based

Simulated experience & Planning

本讲参考文献

1. Richard S. Sutton and Andrew G. Barto. Reinforcement Learning: An Introduction. （Second edition, in progress，draft）.
2. David Silver，Slides@《Reinforcement Learning:An Introduction》,2016.