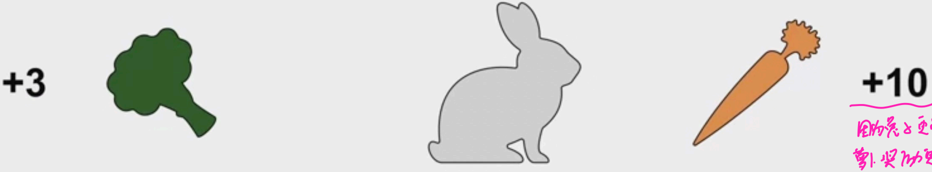
Markov Decision Processes:

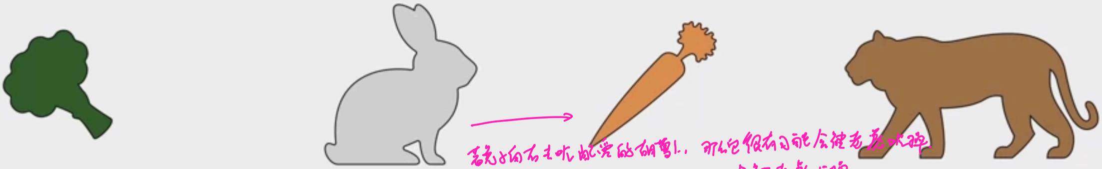
Different situations call for different actions



因为兔子更爱吃胡萝卜，∴胡萝卜奖励更高，兔子也会更倾向于向右走去吃胡萝卜.

注: k-armed bandit problem 并没有考虑到不同的 situation 需要不同的 action，它在每一次都做出相同的行动

1:13 / 6:33



若兔子向右去吃喜爱的胡萝卜，那么它很有可能会被老虎吃掉。

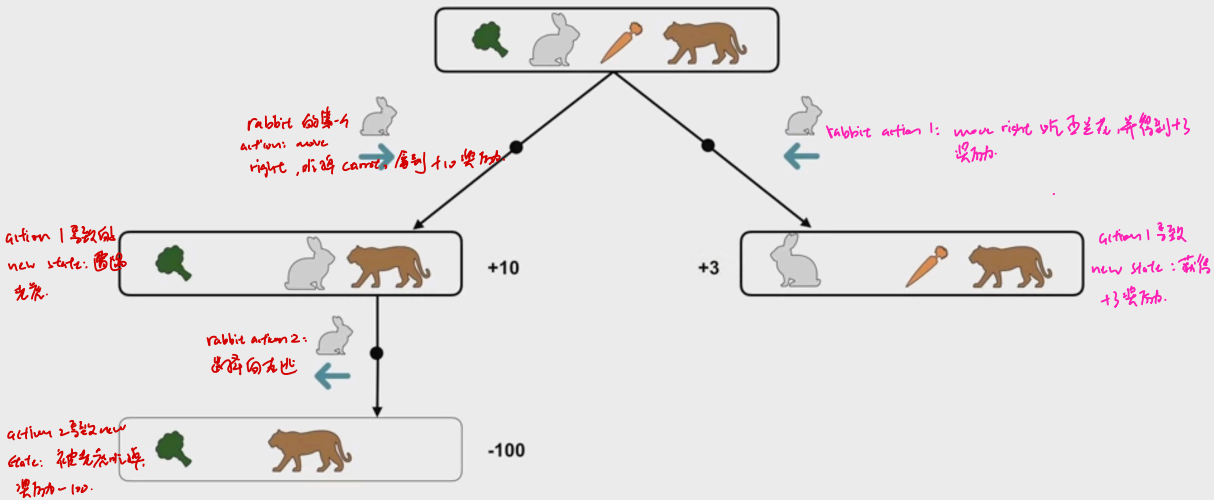
∴兔子更应该向左转去吃西兰花，才不会被老虎吃掉。

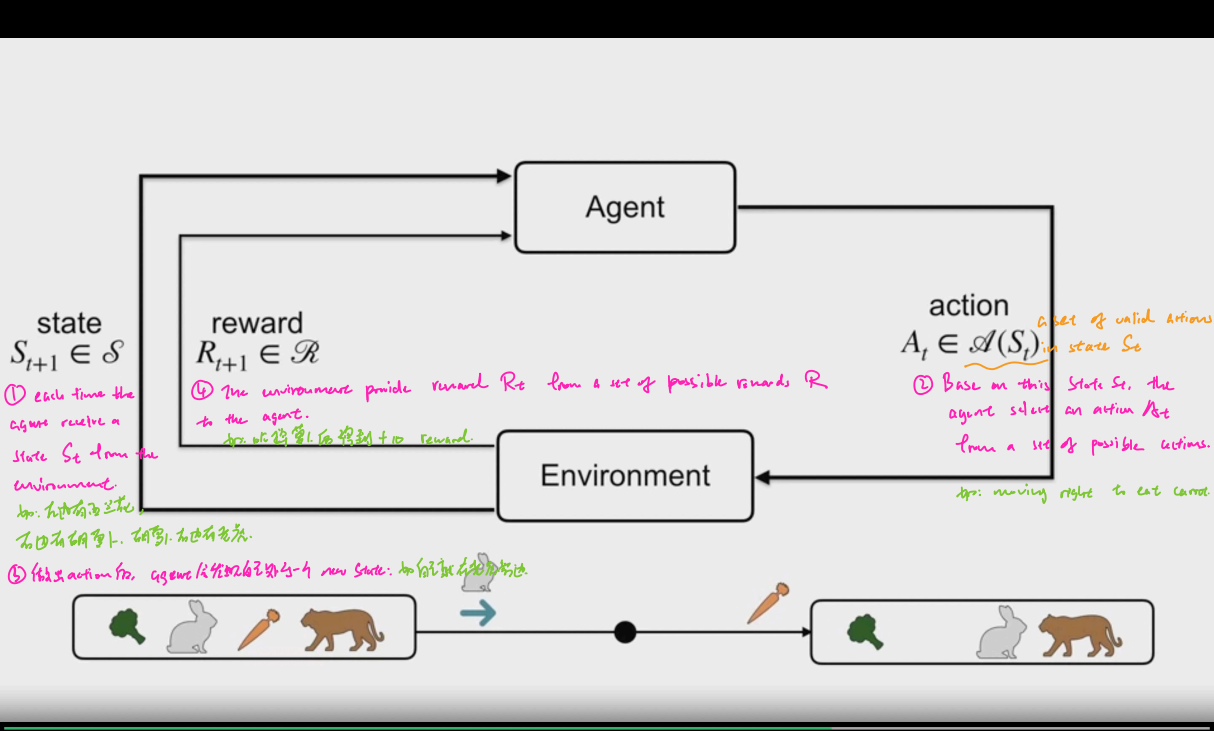
但 bandit rabbit 只考虑 immediate reward，而吃胡萝卜获得的奖励比吃西兰花多。

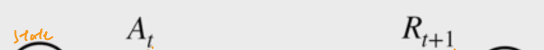
∴它会 go for the carrot.

1:42 / 6:33

The sequence happend depends on the action that the rabbit takes:

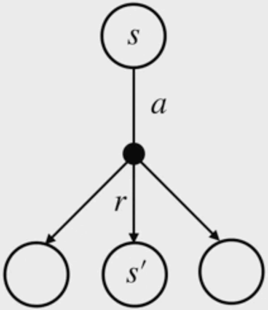



Agent
Environment
state
$S_{t+1} \in \mathcal{S}$
① each time the agent receive a state S_t from the environment.
reward
$R_{t+1} \in \mathcal{R}$
④ The environment provide reward R_t from a set of possible rewards $\mathcal{R}$ to the agent.
action
$A_t \in \mathcal{A}(S_t)$
a set of valid actions in state S_t
② Base on this state S_t, the agent select an action A_t from a set of possible actions.
⑤ After the action is, agent is in a new state.



Actions influence immediately rewards as well as future state.

The dynamics of an MDP



$$p(s', r \mid s, a)$$

reward

tells the joint probability of s′ and r. — next state — given a state s and action a.

P must be non-negative (因为是一个概率)
and sum of all new states and rewards must equal 1.

$$p : \mathcal{S} \times \mathcal{R} \times \mathcal{S} \times \mathcal{A} \rightarrow [0, 1]$$

$$\sum_{s' \in \mathcal{S}} \sum_{r \in \mathcal{R}} p(s', r \mid s, a) = 1, \forall s \in \mathcal{S}, a \in \mathcal{A}(s)$$

Future state and rewards only depends on current state and action!! 注:

5:31 / 6:33

The present state contains all the information necessary to predict the future

(The Markov property).

马尔科夫性

> 马尔科夫性质（英语：Markov property）是概率论中的一个概念，因为俄国数学家安德雷·马尔科夫得名。当一个随机过程在给定现在状态及所有过去状态情况下，其未来状态的条件概率分布仅依赖于当前状态；换句话说，在给定现在状态时，它与过去状态（即该过程的历史路径）是条件独立的，那么此随机过程即具有马尔科夫性质。马尔科夫性-百度百科

马尔科夫性，也就是无后效性：**某阶段的状态一旦确定，则此后过程的演变不再受此前各状态及决策的影响**。也就是说，**未来与过去无关**。

具体地说，如果一个问题被划分各个阶段之后，阶段k中的状态只能通过阶段$k+1$中的状态通过状态转移方程得来，与其他状态没有关系，特别是与未发生的状态没有关系，这就是无后效性。

公式描述：

$$P[S_{t+1}|S_t] = P[S_{t+1}|S_1, \dots, S_t]$$

强化学习问题中的状态也符合马尔科夫性，即在当前状态s_t下执行动作a_t并转移至下一个状态s_{t+1}，而不需要考虑之前的状态$s_{t-1}, \dots, s_1$。

强化学习中默认状态的转移是符合马尔科夫性质的，状态具体是什么，需要根据不同的问题进行不同的设定。

Dynamics of the Recycling Robot

waiting for cans does not drain the battery

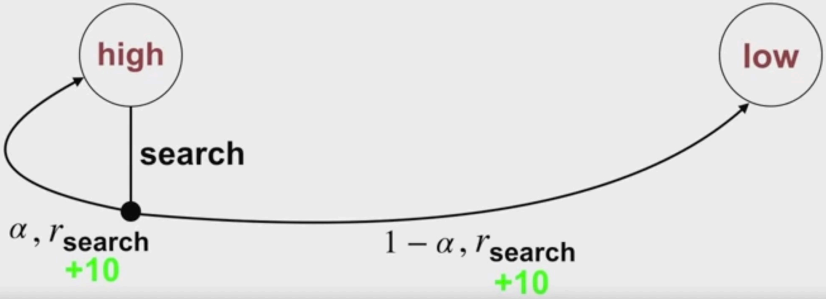
high energy level GS property

the robot finds to cans.

low energy level GS property.

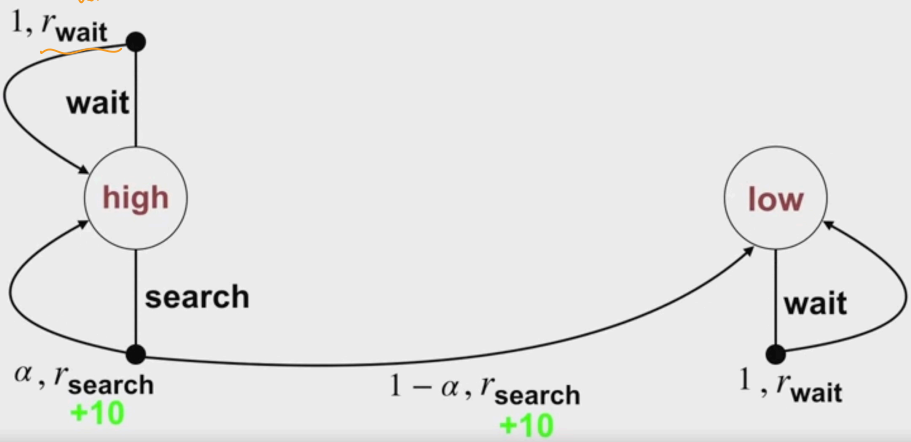
Dynamics of the Recycling Robot

waiting for cans does not drain the battery



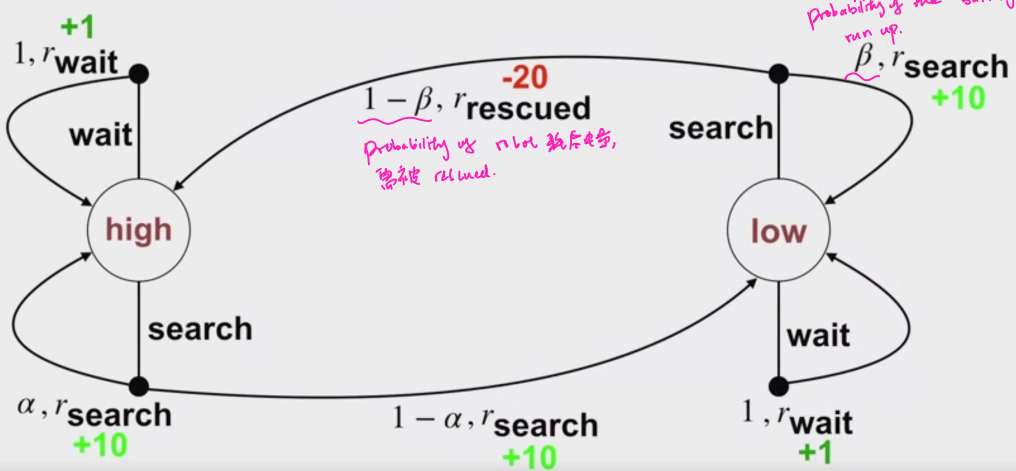
Dynamics of the Recycling Robot

waiting for cans does not drain the battery

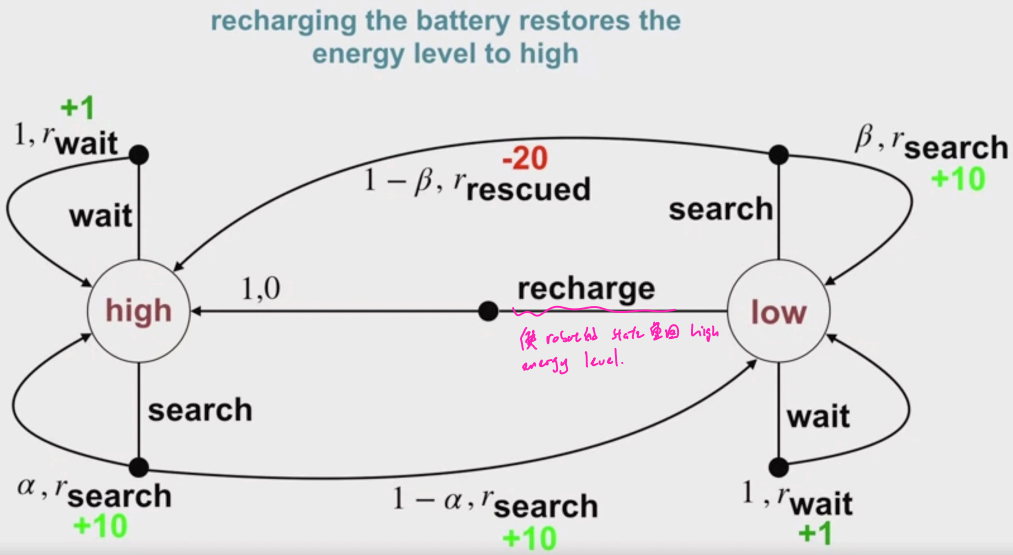


Dynamics of the Recycling Robot

search with energy level low may deplete the battery
and the robot must be rescued



Dynamics of the Recycling Robot



在强化学习中，代理人的目标是使未来的奖励最大化。让我们来定义代理人的目标。也许我们可以像在bandits中那样，只将眼前的奖励最大化。不幸的是，这在MDP中是行不通的。在现在这个时间步骤上(time step)的行动可能会产生大的奖励，因为在后面的时间步骤上，代理人会过渡到一个产生低奖励的状态。因此，在短期内看起来不错的行动，在长期内可能不是最好的。

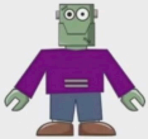
考虑到一个机器人在学习走路。奖励可能与向前的运动成正比。猛烈的向前冲显然会使眼前的回报最大化。然而，这个动作会导致机器人摔倒。如果机器人将总的向前运动最大化，它就会快速但小心地行走。现在，让我们正式定义一下我们所说的未来总回报最大化是什么意思。时间步骤t的回报，仅仅是时间步骤t之后获得的奖励的总和。我们用字母G表示回报。为了更好地理解这一点，我设想一个收集罐子的机器人从这里开始。从这个状态开始，机器人总是采取相同的行动序列。有时它可能得到一个大的回报，有时它可能得到一个较小的回报。这是由于单个奖励和状态转换的随机性造成的。一般来说，从同一状态出发有许多不同的轨迹是可能的。这就是为什么我们要最大化预期收益。为了使之得到良好的定义，奖励的总和必须是有限的。具体来说，假设有一个最后的时间步骤，称为资本T(capital T)，在那里代理人环境互动结束。互动结束后会发生什么？在最简单的情况下，互动自然地分成几块，称为情节(episodes)。每一情节的开始与前一情节的结束方式无关。在终止时，代理被重置到一个起始状态。每个情节都有一个最终状态，我们称之为终端状态(terminal state)。我们把这些任务称为偶发任务(episodic tasks)。为了更好地理解偶发任务，让我们看一个具体的例子。考虑一下国际象棋的游戏。一盘国际象棋总是以将死、平局或认输而告终。在下棋时，一个情节会是什么样子呢？你可能已经猜到了，一盘棋就构成一个情节。每一局都从相同的起始状态开始，所有的棋子都被重置。

Learning to Walk

maximizing the immediate reward

reward $\propto$ forward motion

Goal of an Agent : Formal definition

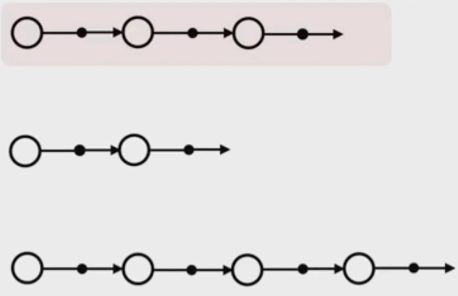


return $G_t \doteq R_{t+1} + R_{t+2} + R_{t+3} + \dots$

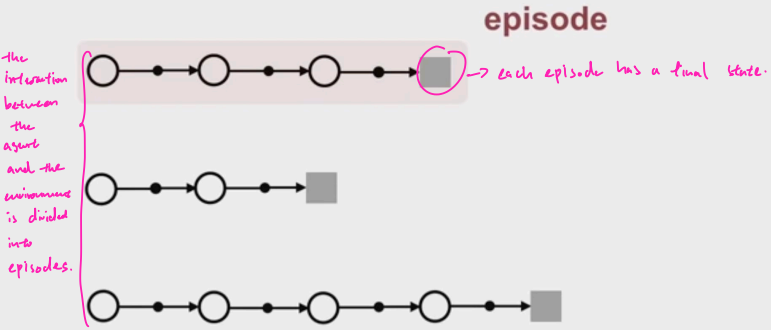
R.V. 因为 because the dynamic of the MDP can be stochastic.

Episodic Tasks

episode



Episodic Tasks



Episodic Tasks

episode

Example : Episodic Task

Every game of chess has a final move

Example : Episodes

Each game of chess is an episode

- each game
2) begins from the same state with all pieces reset.

- each game of chess has a final move (the final state).

奖励假说（reward hypothesis）的基本思想在这句名言中得到了说明，给人一条鱼，他可以吃一天，教人钓鱼，他可以吃一辈子。给人以鱼的味道，即使细节改变，他也会想出如何捕鱼。好吧，也许这不是名言，但有一个不是我的人转发了这句话，所以这也是一种收获。总之，有三种方法可以考虑创造智能行为。

第一种，授人以鱼，是老式的人工智能。如果我们想让机器变得聪明，我们就用我们想要的行为为它编程。但随着新问题的出现，机器将无法适应新的环境。它需要我们一直在那里提供新的程序。

第二种，授人以渔（teach a man to fish），是监督式学习。如果我们想让机器变得聪明，我们就提供训练实例，然后机器自己写程序来匹配这些实例。它就会学习，所以只要我们有办法提供训练实例，我们的机器就能写出自己的程序，这就是进步。但是情况会发生变化，我的意思是我们大多数人没有机会每天吃鱼或钓鱼来获取食物。

第三，给人以鱼的味道，这是强化学习。这是一个想法，我们不需要指定实现目标的机制。我们可以只对目标进行编码，机器可以设计自己的策略来实现它。我的意思是现在，你不必为了吃鲑鱼而去抓鲑鱼，有超市，有海鲜连锁餐厅。如果一切都失败了，还有加油站的寿司，所以这就是高层次的想法。

但是假设本身呢？现在，我很确定我是从里奇-萨顿那里得到的，但我听说里奇-萨顿把它归功于我。因此，如果我打算告诉你，我想最好先了解一下这个词的历史。谷歌趋势是一项了不起的服务，它提供关于一个术语在历史上使用频率的信息。因此，当我问它关于奖励假说的问题时，它说没有结果。搜索奖励假说的强化学习有3580个结果，所以这是很重要的。在结果的前几页，我发现了一些例子，Muhammad Ashraf的一篇博文说，所有的目标都可以用预期累积奖励的最大化来描述。

在之前的视频中，我们讨论了偶发性问题。然而，在许多问题中，代理环境的交互是无止境的。今天，我们将看到如何将这类问题制定为持续任务。

让我们来看看偶发任务(episodic taska)和 持续任务(continuing tasks)之间的区别。

正如我们前面所讨论的，偶发任务被分解成若干集。偶发性任务中的每一集都必须以一个终端状态结束。下一集的开始与上一集的结束方式无关。在时间步骤t的回报是直到终止的奖励的总和。相反，持续任务不能被分解成独立的情节。互动持续进行。不存在终端状态。

为了更具体地说明这一点，考虑一个调节建筑物温度的智能恒温器。这可以被表述为一个持续的任务，因为恒温器从未停止与环境的交互。状态可以是当前的温度以及情况的细节，如一天中的时间和楼里的人数。只有两个动作，打开加热器或关闭它。每当有人需要手动调节温度时，奖励为负1，否则为0。为了避免负奖励，恒温器将学习预测用户的喜好。那么，我们如何制定持续任务的回报？我们可以像对待偶发任务那样，尝试总结所有的未来回报。但是现在，我们要对一个无限的序列进行求和。这个回报可能不是有限的。那么，我们怎样才能修改这个总和，使之总是有限的呢？一个解决方案是将未来的回报用一个叫作贴现率的系数Gamma进行贴现。Gamma至少是0，但小于1。然后，回报的表述可以被修改以包括贴现。贴现对收益的影响很简单，即时奖励对收益的贡献更大。远在未来的回报则贡献较少，因为它们被乘以Gamma提高到k的连续较大的幂。今天的一美元比一年后的一美元对你更有价值。我们可以把这个总和简明地写成这个表达式，它保证是有限的。让我们看看为什么？假设R_max是我们的老龄化在任何时间步骤中可以得到的最大奖励。现在我们可以通过用R_max替换每一个奖励来对回报G_t进行上界。由于R_max只是一个常数，我们可以把它从求和中拉出来。请注意，第二个因素只是一个几何数列，几何数列的值是1除以1减去Gamma，R_max乘以1除以1减去Gamma是有限的，是G_t的一个上限。所以我们知道G_t是有限的。现在，让我们来看看贴现因子对代理人行为的影响。我们可以看看Gamma等于0和Gamma接近1时的两个极端情况。当Gamma等于零时，回报只是下一个时间步骤的奖励。所以代理人是短视的，只关心眼前的预期回报。另一方面，当Gamma接近1时，近期和未来的回报在回报中的权重几乎相等。在这种情况下，代理人是比较有远见的。

最后，让我们讨论一下回报的一个简单但重要的属性。它可以被递归地写出来。让我们从总和中的第二项开始把Gamma分解。令人惊讶的是，括号里的序列是下一个时间步骤的回报。所以我们可以直接用G_t加1来代替它。现在，我们有一个递归方程，左边是G_t，右边是G_t加1。这个简单的方程比它看起来更强大。

Episodic Tasks

- Interaction breaks naturally into episodes
- Each episode ends in a terminal state
- Episodes are independent

$$G_t \doteq R_{t+1} + R_{t+2} + R_{t+3} + \ldots + R_T$$

for time step t 的獲得報酬.

Continuing Tasks

- Interaction goes on continually
- No terminal state

$$G_t \doteq R_{t+1} + R_{t+2} + R_{t+3} + \ldots$$

$= \infty$

Discounting

$$G_t \doteq R_{t+1} + R_{t+2} + R_{t+3} + \ldots + R_{t+k} + \ldots$$

How to make sure G_t is finite?

Discount the rewards in the future by γ

Where $0 \leq \gamma < 1$

γ is counting rate.

Discounting

$$G_t \doteq R_{t+1} + \gamma R_{t+2} + \gamma^2 R_{t+3} + \ldots + \gamma^{k-1} R_{t+k} + \ldots$$

$$= \sum_{k=0}^{\infty} \gamma^k R_{t+k+1}$$

Finite as long as $0 \leq \gamma < 1$

How to make sure G_t is finite?

Discount the rewards in the future by γ

Where $0 \leq \gamma < 1$

Discounting

Assume R_{max} is the maximum reward the agent can receive

$$G_t = \sum_{k=0}^{\infty} \gamma^k R_{t+k+1} \leq \sum_{k=0}^{\infty} \gamma^k R_{max} = R_{max} \sum_{k=0}^{\infty} \gamma^k$$

$$= \frac{1}{1-\gamma}$$

Converging geometric series when $\gamma < 1$

Discount rewards in the future by γ

Where $0 \leq \gamma < 1$

Discounting

Assume R_{max} is the maximum reward the agent can receive

$$G_t = \sum_{k=0}^{\infty} \gamma^k R_{t+k+1} \leq \sum_{k=0}^{\infty} \gamma^k R_{max} = R_{max} \sum_{k=0}^{\infty} \gamma^k$$

$$= R_{max} \times \frac{1}{1-\gamma}$$

Finite!

the upper bound of G_t. $\therefore$ G_t is finite.

Discount rewards in the future by γ

Where $0 \leq \gamma < 1$

Effect of γ on agent behavior

$$G_t \doteq R_{t+1} + \gamma R_{t+2} + \gamma^2 R_{t+3} + \ldots + \gamma^{k-1} R_{t+k} + \ldots$$

$\gamma = 0$

$$= R_{t+1} + 0 R_{t+2} + 0^2 R_{t+3} + \ldots + 0^{k-1} R_{t+k} + \ldots$$
$$= R_{t+1}$$

Agent only cares about the immediate reward!

$\Longrightarrow$ **Short-sighted agent!**

$\gamma \to 1$

Agent takes future rewards into account more strongly

$\Longrightarrow$ **Far-sighted agent!**

Recursive nature of returns

$$G_t \doteq R_{t+1} + \gamma R_{t+2} + \gamma^2 R_{t+3} + \gamma^3 R_{t+4} + \ \ldots$$

$$= R_{t+1} + \gamma(R_{t+2} + \gamma R_{t+3} + \gamma^2 R_{t+4} + \ \ldots)$$

$$G_t = R_{t+1} + \gamma G_{t+1}$$

This is just G_{t+1}

6. Suppose $\gamma = 0.8$ and the reward sequence is $R_1 = 5$ followed by an infinite sequence of 10s. What is G_0?

- (●) 45
- ( ) 15
- ( ) 55

$$G_0 = R_1 + \gamma R_2 + \gamma R_3 + \gamma R_4 + \cdots$$

$$= 5 + 0.8 R_2 + (0.8)^2 R_3 + (0.8)^3 R_4 + \cdots$$

$$= 5 + 0.8 \times 10 + (0.8)^2 \times 10 + (0.8)^3 \times 10 + \cdots$$

$$= 5 + 10\left[(0.8) + (0.8)^2 + (0.8)^3 + \cdots\right] \quad \left(= \frac{0.8}{1-0.8}\right)$$

$$= 5 + 10 \cdot \frac{0.8}{1-0.8}$$

$$= 45$$

$\therefore G_1 = R_2 + 0.8 \cdot R_3 + (0.8)^2 \cdot R_4 + \cdots$

從第 Reward $R_1 = 5$ follow by infinite sequence $= 10$'s.

$$= \underbrace{10} + 0.8 \times 10 + (0.8)^2 \times 10 + \cdots$$ $\therefore$ 從 R_2 一直到 R_∞ 都是 10.

$$= 10 + \left[0.8 + (0.8)^2 + (0.8)^3 + \cdots\right]$$

$$= 10 \times \frac{0.8}{1-0.8}$$

$$= 40.$$

$\therefore G_2 = R_3 + 0.8 R_4 + (0.8)^2 R_5 + \cdots$

$$= 10\left[0.8 + (0.8)^2 + (0.8)^3 + \cdots\right]$$

$$= 10 \times \frac{0.8}{1-0.8}$$

$$= 40$$

Recursive nature of returns

$$
\begin{aligned}
G_t &\doteq R_{t+1} + \gamma R_{t+2} + \gamma^2 R_{t+3} + \gamma^3 R_{t+4} + \ \ldots \\
&= R_{t+1} + \gamma (R_{t+2} + \gamma R_{t+3} + \gamma^2 R_{t+4} + \ \ldots) \\
G_t &= R_{t+1} + \gamma G_{t+1}
\end{aligned}
$$

Summary

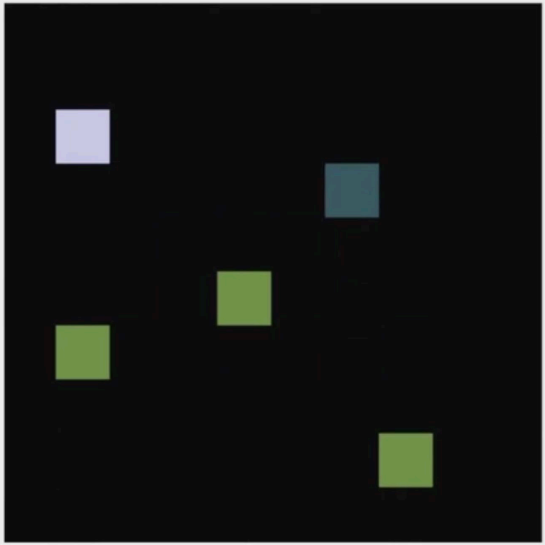
- In continuing tasks, the agent-environment interaction goes on indefinitely
- Discounting is used to ensure returns are finite
- Return can be defined recursively

什么时候我们应该把一个问题制定为偶发或持续的任务?

首先让我们看看一个偶发任务（episodic task）的例子。考虑一个代理学习玩一个简单的视频游戏。用蓝色表示的玩家通过收集白色的宝物块获得分数。当玩家接触到绿色的敌人方块时，游戏就结束了。这个游戏自然被表示为一个偶发的NDP。代理人试图获得高分，在游戏结束前收集尽可能多的分数。状态是一个对应于当前屏幕的像素值数组。有四个动作，上、下、左、右。代理人只要收集到一个宝物块，就会得到加一的奖励。当代理人接触到一个绿色的敌人时，一个情节就会结束。无论剧情如何结束，下一集我们将从特工在屏幕中央开始，没有敌人存在。顺便说一下，你目前正在观看的特工是用钥匙学习法训练的。它很擅长这个游戏，不是吗?在课程2中，你将学习这种算法并自己实现它。

现在我们来看看一个持续任务（continuing task）的例子。代理人要在一组服务器上安排工作。假设我们有三台服务器，由强化研究人员用来进行实验。研究人员将具有不同优先级的作业提交到一个队列（queue）中。状态是空闲服务器的数量，以及排在队列顶部的作业的优先级。行动是拒绝或接受队列顶部的作业，如果有服务器是空闲的。接受作业，运行它，并产生一个与作业优先级相等的奖励。拒绝作业会产生一个与优先级成正比的负奖励，并将作业送到队列的后面。代理人应该小心安排低优先级的工作，因为他可能会阻止高优先级的工作以后被安排。服务器在完成工作后再次变得空闲。研究人员不断地将工作添加到队列中，代理（服务器）接受或拒绝它们。由于这个过程从未停止过，它被很好地描述为一个持续的任务。

偶发性任务自然地分成独立的片段。持续任务被认为是无限期地持续下去。你现在应该能够确定哪种表述方式最适合于某个特定的问题。



State

Actions

+1 Reward

Example Scheduler

States: {Priority,
Free Servers}

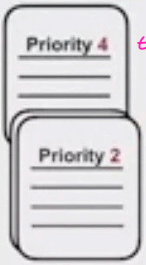
Actions: {Accept, Reject}

Example Scheduler

States: {Priority,
Free Servers}

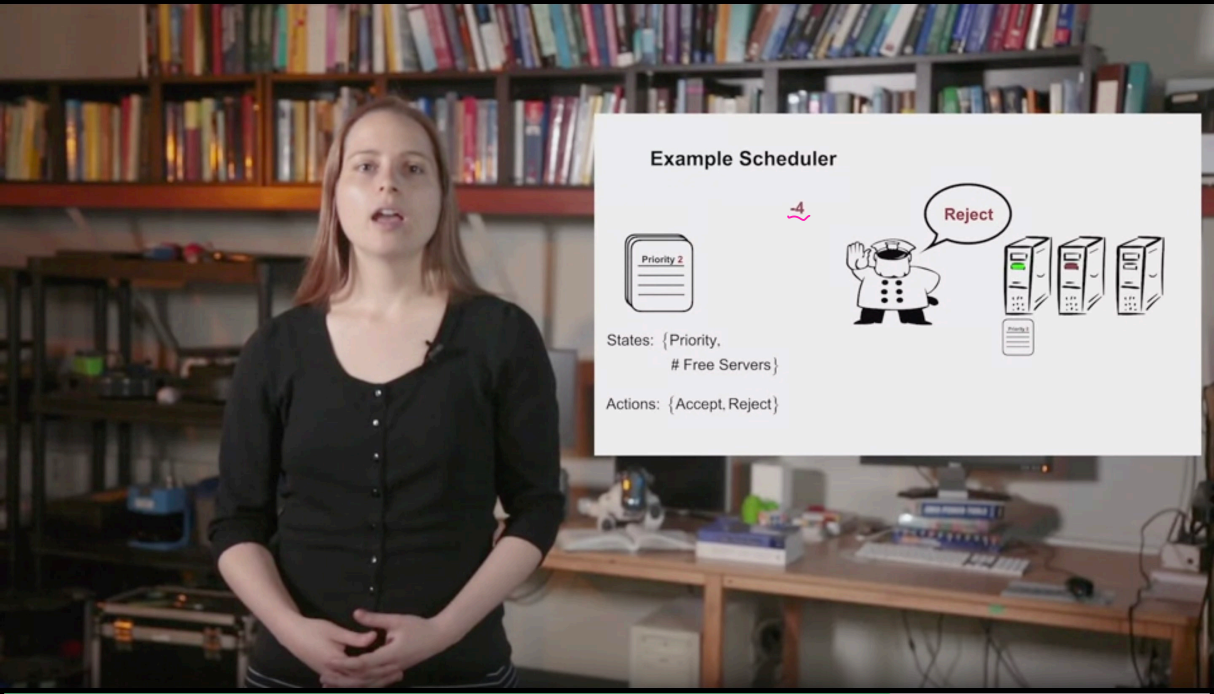
Actions: {Accept, Reject}

Example Scheduler



States: {Priority,
Free Servers}

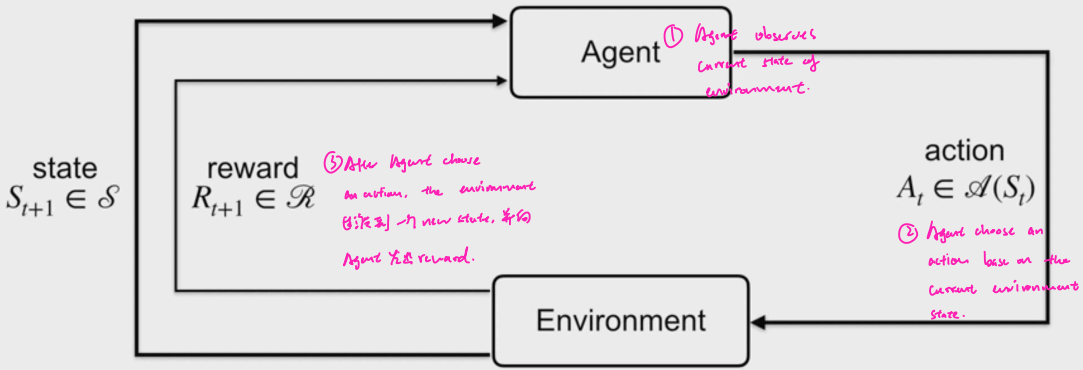
Actions: {Accept, Reject}

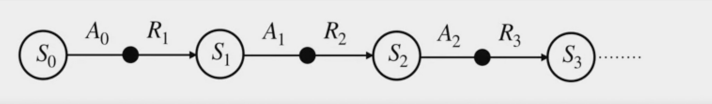

Example Scheduler
-4
Reject
Priority 2
States: {Priority,
Free Servers}
Actions: {Accept, Reject}

Summary

- Episodic tasks break naturally into independent episodes
- Continuing tasks are assumed to continue indefinitely

简而言之，MDPs形式化了一个代理人与环境互动的问题。代理人和环境以离散的时间步数进行互动。在每个时间段，代理观察环境的当前状态（the current state of the environment）。基于这种状态（state），代理人选择一个行动（action）。之后，环境过渡到一个新的状态，并发出一个奖励（reward）。记住，代理人的选择有长期的后果。它所选择的行动会影响未来的状态和奖励。强化学习的目标是使未来的总奖励最大化。这通常意味着要平衡 即时奖励（immediate reward）和 行动的长期后果（the long-term consequences of actions）。我们用预期回报（expected return）来正式确定这一目标，它是未来回报的预期折现总和。通过用小于1的Gamma进行贴现，我们可以保证回报是有限的。Gamma的精确值定义了我们对短期回报和长期回报的关注程度。我们讨论了一些问题的例子，这些问题可以被自然地表述为偶发的或持续的MDPs。MDP形式主义可以用来模拟许多现实世界的问题。应用强化学习的第一步总是将问题表述为MDP。


Agent
① Agent observes current state of environment.
state
$S_{t+1} \in \mathcal{S}$
reward
$R_{t+1} \in \mathcal{R}$
③ After Agent chose an action, the environment 转移到一个 new state, 并给 Agent 一个 reward.
action
$A_t \in \mathcal{A}(S_t)$
② Agent choose an action based on the current environment state.
Environment

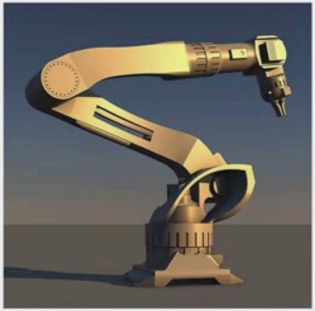

S_0
A_0
R_1
S_1
A_1
R_2
S_2
A_2
R_3
S_3
.....

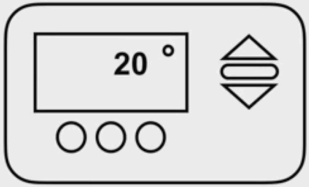
The Goal of Reinforcement Learning

=> 使未来的总奖励最大化.

$$G_t \doteq R_{t+1} + \gamma R_{t+2} + \gamma^2 R_{t+3} + \ \ldots = \sum_{k=0}^{\infty} \gamma^k R_{t+k+1}$$

Examples of Continuing and Episodic MDPs





1. The learner and decision maker is the _____. 1 / 1 分

- (●) Agent
- ( ) State
- ( ) Environment
- ( ) Reward

正确
Correct!

2. At each time step the agent takes an _____. 1 / 1 分

- ( ) State
- (●) Action
- ( ) Reward
- ( ) Environment

正确
Correct!

3. Imagine the agent is learning in an episodic problem. Which of the following is true? 1 / 1 分

- ○ The number of steps in an episode is always the same.
- ○ The agent takes the same action at each step during an episode.
- ◉ The number of steps in an episode is stochastic: each episode can have a different number of steps.

正确
Correct!

4. If the reward is always +1 what is the sum of the discounted infinite return when $\gamma < 1$ 1 / 1 分

$G_t = \sum_{k=0}^{\infty} \gamma^k R_{t+k+1}$

- ○ Infinity.
- ◉ $G_t = \frac{1}{1-\gamma}$
- ○ $G_t = \frac{\gamma}{1-\gamma}$
- ○ $G_t = 1 * \gamma^k$

正确
Correct!

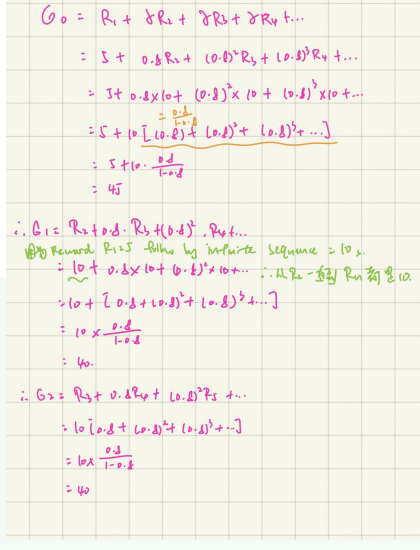
5. What is the difference between a small gamma (discount factor) and a large gamma? **1 / 1 分**

- ○ The size of the discount factor has no effect on the agent.
- ○ With a smaller discount factor the agent is more far-sighted and considers rewards farther into the future.
- ◉ With a larger discount factor the agent is more far-sighted and considers rewards farther into the future.

✓ **正确**
Correct!

6. Suppose $\gamma = 0.8$ and the reward sequence is $R_1 = 5$ followed by an infinite sequence of 10s. What is G_0? **1 / 1 分**

- ◉ 45
- ○ 15
- ○ 55



✓ **正确**
Correct!

$G_2 = 10/(1 - 0.8) = 50$

$G_1 = 10 + .8 * (50) = 50$

$G_0 = 5 + .8 * 50 = 45$

6. Suppose $\gamma = 0.8$ and we observe the following sequence of rewards: $R_1 = -3$, $R_2 = 5$, $R_3 = 2$, $R_4 = 7$, and $R_5 = 1$, with $T = 5$. What is G_0? Hint: Work Backwards and recall that $G_t = R_{t+1} + \gamma G_{t+1}$.

1 / 1 分

$G_t = R_{t+1} + \gamma R_{t+2} + \gamma^2 R_{t+3} + \gamma^3 R_{t+4} + \gamma^4 R_{t+5}$ ($\gamma R_{t+2} + \gamma^2 R_{t+3} + \gamma^3 R_{t+4} + \gamma^4 R_{t+5}$ 为 G_{t+1})

$\therefore G_0 = R_1 + \gamma R_2 + \gamma^2 R_3 + \gamma^3 R_4 + \gamma^4 R_5$

$= -3 + (0.8) \times 5 + (0.8)^2 \times 2 + (0.8)^3 \times 7 + (0.8)^4 \times 1$

$= 6.2736$

- 11.592
- **6.2736**
- -3
- 8.24
- 12

正确
Correct!

7. What does MDP stand for? **1 / 1 分**

- ○ Markov Decision Protocol
- ○ Meaningful Decision Process
- ○ Markov Deterministic Policy
- ◉ Markov Decision Process

✓ **正确**
Correct!

8. Consider using reinforcement learning to control the motion of a robot arm in a repetitive pick-and-place task. If we want to learn movements that are fast and smooth, the learning agent will have to control the motors directly and have low-latency information about the current positions and velocities of the mechanical linkages. The actions in this case might be the voltages applied to each motor at each joint, and the states might be the latest readings of joint angles and velocities. The reward might be +1 for each object successfully picked up and placed. To encourage smooth movements, on each time step a small, negative reward can be given as a function of the moment-to-moment "jerkiness" of the motion. Is this a valid MDP? **1 / 1 分**

考虑使用强化学习来控制机器人手臂在重复的取放任务中的运动。如果我们想学习快速而流畅的动作，学习代理必须直接控制电机，并拥有关于机械连接的当前位置和速度的低延迟信息。在这种情况下，动作可能是应用于每个关节的电机的电压，而状态可能是关节角度和速度的最新读数。每成功拿起和放置一个物体，奖励可能是+1。为了鼓励平稳的运动，在每个时间步骤中，可以给一个小的、负的奖励，作为运动的瞬间 "抖动 "的函数。这是一个有效的MDP吗？

- ◉ Yes
- ○ No

只要满足 Markov Property 就是一个合理的 (valid) MDP.

✓ **正确**
Correct!

案例1：想象一下，你是一个视觉系统。当你第一次开启一天的工作时，一个图像涌入你的相机。你可以看到很多东西，但不是所有东西。你看不到被遮挡的物体，当然你也看不到在你身后的物体。在看到第一个场景后，你是否能获得环境的马尔可夫状态?

案例2：想象一下，视觉系统从未正常工作：它总是返回相同的静态想象，永远如此。那么，你能获得马尔可夫状态吗?

马尔科夫性

> 马尔科夫性质（英语：Markov property）是概率论中的一个概念，因为俄国数学家安德雷·马尔科夫得名。当一个随机过程在给定现在状态及所有过去状态情况下，其未来状态的条件概率分布仅依赖于当前状态；换句话说，在给定现在状态时，它与过去状态（即该过程的历史路径）是条件独立的，那么此随机过程即具有马尔科夫性质。马尔科夫性-百度百科

马尔科夫性，也就是无后效性：**某阶段的状态一旦确定，则此后过程的演变不再受此前各状态及决策的影响**。也就是说，**未来与过去无关**。

具体地说，如果一个问题被划分各个阶段之后，阶段k中的状态只能通过阶段$k+1$中的状态通过状态转移方程得来，与其他状态没有关系，特别是与未发生的状态没有关系，这就是无后效性。

公式描述：

$$P[S_{t+1}|S_t] = P[S_{t+1}|S_1, \dots, S_t]$$

强化学习问题中的状态也符合马尔科夫性，即在当前状态s_t下执行动作a_t并转移至下一个状态s_{t+1}，而不需要考虑之前的状态$s_{t-1}, \dots, s_1$。

8. Suppose reinforcement learning is being applied to determine moment-by-moment temperatures and stirring rates for a bioreactor (a large vat of nutrients and bacteria used to produce useful chemicals). The actions in such an application might be target temperatures and target stirring rates that are passed to lower-level control systems that, in turn, directly activate heating elements and motors to attain the targets. The states are likely to be thermocouple and other sensory readings, perhaps filtered and delayed, plus symbolic inputs representing the ingredients in the vat and the target chemical. The rewards might be moment-by-moment measures of the rate at which the useful chemical is produced by the bioreactor. ∴ finite MDP

1 / 1 分

Notice that here each state is a list, or vector, of sensor readings and symbolic inputs, and each action is a vector consisting of a target temperature and a stirring rate.

Is this a valid MDP?

- (●) Yes. Assuming the state captures the relevant sensory information (inducing historical values to account for sensor delays). It is typical of reinforcement learning tasks to have states and actions with such structured representations; the states might be constructed by processing the raw sensor information in a variety of ways.
- ( ) No. If the instantaneous sensor readings are non-Markov it is not an MDP: we cannot construct a state different from the sensor readings available on the current time-step.

✓ **正确**
Correct!

假设强化学习被应用于确定生物反应器（用于生产有用化学品的营养物和细菌大桶）的逐时温度和搅拌率。在这样的应用中，行动可能是目标温度和目标搅拌率，它们被传递给下级控制系统，反过来直接激活加热元件和电机以达到目标。状态可能是热电偶和其他感官读数，也许经过过滤和延迟，加上代表大桶中的成分和目标化学品的符号输入。奖励可能是对生物反应器生产有用化学品的速度的逐时测量。

请注意，这里的每个状态都是传感器读数和符号输入的列表或向量，而每个动作都是由目标温度和搅拌速率组成的向量。

a. 是的。假设状态捕捉到了相关的感觉信息（诱导历史值以考虑到传感器的延迟）。强化学习任务的典型特征是具有这种结构化表示的状态和动作；状态可能是通过以各种方式处理原始传感器信息而构建的。

b. 如果瞬时传感器读数是非马尔科夫的，它就不是一个MDP：我们不能构建一个与当前时间步长的传感器读数不同的状态。

9. **Case 1**: Imagine that you are a vision system. When you are first turned on for the day, an image floods into your camera. You can see lots of things, but not all things. You can't see objects that are occluded, and of course you can't see objects that are behind you. After seeing that first scene, do you have access to the Markov state of the environment?

0 / 1 分

Case 2: Imagine that the vision system never worked properly: it always returned the same static imagine, forever. Would you have access to the Markov state then? (Hint: Reason about $P(S_{t+1}|S_t, ..., S_0)$, where S_t = AllWhitePixels)

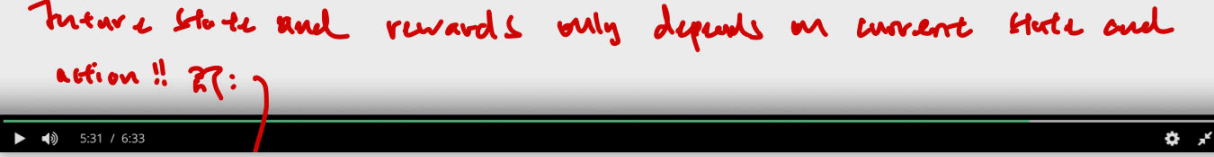
- ○ You have access to the Markov state in both Case 1 and 2.
- ◉ You have access to the Markov state in Case 1, but you don't have access to the Markov state in Case 2.
- ○ You don't have access to the Markov state in Case 1, but you do have access to the Markov state in Case 2.
- ○ You don't have access to the Markov state in both Case 1 and 2.

⊗ **错误**

Incorrect. Because there is no history before the first image, the first state has the Markov property. The Markov property does not mean that the state representation tells all that would be useful to know, only that it has not forgotten anything that would be useful to know.

The case when the camera is broken is different, but again we have the Markov property. All the possible futures are the same (all white), so nothing needs to be remembered in order to predict them.

Future state and rewards only depends on current state and action !! 环境:



(The Markov property).

10. What is the reward hypothesis? 1 / 1 分

- ○ Goals and purposes can be thought of as the minimization of the expected value of the cumulative sum of rewards received.
- ○ Always take the action that gives you the best reward at that point.
- ○ Ignore rewards and find other signals.
- ◉ Goals and purposes can be thought of as the maximization of the expected value of the cumulative sum of rewards received.

正确
Correct!

The hypothesis implies that all such cases can be reduced to maximizing expected cumulative reward for some choice of reward. Thus the hypothesis may be at odds with the fact that there exist large bodies of research that treat risk-sensitive planning as a special case.

该假说意味着，所有这些情况都可以简化为对某些奖励选择的预期累积奖励最大化。因此，该假设可能与存在大量将风险敏感规划作为一种特殊情况的研究的事实相矛盾。

11. Imagine, an agent is in a maze-like gridworld. You would like the agent to find the goal, as quickly as possible. You give the agent a reward of +1 when it reaches the goal and the discount rate is 1.0, because this is an episodic task. When you run the agent its finds the goal, but does not seem to care how long it takes to complete each episode. How could you fix this? (Select all that apply) 1 / 1 分

想象一下，一个代理人在一个类似迷宫的网格世界中。你希望代理人能尽快找到目标。当代理到达目标时，你给它的奖励是+1，折扣率是1.0，因为这是一个偶发的任务。当你运行代理时，它找到了目标，但似乎并不关心它需要多长时间来完成每一集。你可以如何解决这个问题？(选择所有适用的)

- ☐ Give the agent a reward of 0 at every time step so it wants to leave.
- ☑ Set a discount rate less than 1 and greater than 0, like 0.9.

正确
Correct! From a given state, the sooner you get the +1 reward, the larger the return. The agent is incentivized to reach the goal faster to maximize expected return.

- ☐ Give the agent a reward of +1 at every time step.
- ☑ Give the agent -1 at each time step.

正确
Correct! Giving the agent a negative reward on each time step, tells the agent to complete each episode as quickly as possible.

12. When may you want to formulate a problem as episodic?

1 / 1 分

- (●) When the agent-environment interaction naturally breaks into sequences. Each sequence begins independently of how the episode ended.
- ( ) When the agent-environment interaction does not naturally break into sequences. Each new episode begins independently of how the previous episode ended.

正确
Correct!

Episodic Tasks

- Interaction breaks naturally into episodes
- Each episode ends in a terminal state
- Episodes are independent

$$G_t \doteq R_{t+1} + R_{t+2} + R_{t+3} + \ldots + R_T$$

for time step t 的獲得報酬.

Continuing Tasks

- Interaction goes on continually
- No terminal state

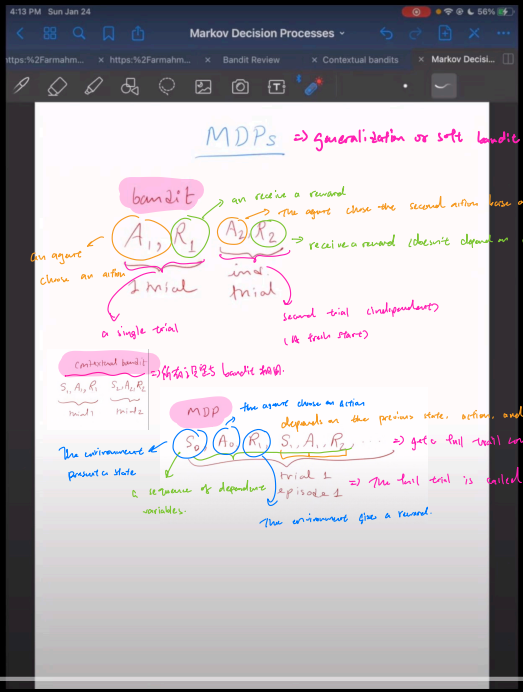
$$G_t \doteq R_{t+1} + R_{t+2} + R_{t+3} + \ldots$$

$= \infty$

There appears to be some confusion on state vs terminal state. The states for the cat might be whether or not it sees / smells / hears a mouse. The terminal state could be whether or not the cat catches a mouse. The state for the maze should include the robots position in the maze. This is because if it is binary like above, the robot will just randomly explore inside the maze until it stumbles its way out, instead of learning an optimal path. This is because it treats all positions inside the maze as the same. The state for Go, should be the actual current game board (i.e. the positions of all the white and black stones). The terminal state would be the final game board, from which we would know whether the robot wins or not

在状态与终端状态上似乎存在一些混淆。猫的状态可能是它是否看到/闻到/听到一只老鼠。终端状态可能是猫是否抓到了老鼠。迷宫的状态应该包括机器人在迷宫中的位置。这是因为如果像上面那样是二进制的，机器人就会在迷宫里随机探索，直到它跌跌撞撞地走出来，而不是学习一条最佳路径。这是因为它把迷宫里的所有位置都当作是一样的。围棋的状态，应该是实际的当前棋盘（即所有白棋和黑棋的位置）。终端状态将是最终的棋盘，我们可以从中知道机器人是否获胜。

mdp-rev.mp4



1:04 / 4:23

history: $H_t = (S_0, A_0, R_1, \dots, S_{t-1}, A_{t-1}, R_t)$

$$P(H_t = h, S_t = s, A_t = a, R_{t+1} = r, S_{t+1} = s')$$

$$= P(R_{t+1} = r, S_{t+1} = s' \mid S_t = s, A_t = a, H_t = h) \times P(A_t = a \mid S_t = s, H_t = h) \times P(S_t = s, H_t = h)$$

this set of conditional properties.

Markov property: if true (the state) gives above, then it means that the future doesn't depend on the past!!

∴ for mdp, the conditioned equation should be:

$$P(H_t = h, S_t = s, A_t = a, R_{t+1} = r, S_{t+1} = s')$$

$$= P(R_{t+1} = r, S_{t+1} = s' \mid S_t = s, A_t = a, H_t = h) \times P(A_t = a \mid S_t = s, H_t = h) \times P(S_t = s, H_t = h)$$

current state (and the history).

The agent chooses their action based on both

(i.e.: Agents can choose actions only based on the states, 즉 state 만 보고 agents represent the best possible behavior !!

optimal properties.

if the state is Markov, then the agent can only without losing anything previous the history).

comes from the environments

agent

comes from the agents.

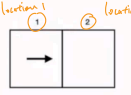
mdp-exmp-full.mp4

4:38 PM Sun Jan 24 — 58%

Markov Decision Processes

https://2Farmahm... × https://2Farmahm... × Bandit Review × Contextual bandits × Markov Decisi...

Example 1: An MDP



location 2 $\mathcal{S} = \{(1,\rightarrow), (1,\leftarrow), (2,\rightarrow), (2,\leftarrow)\}$ — set of possible states.

State is the location and the orientation: (1, →)

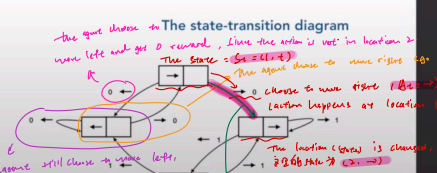
Action is ← or →

Reward is +1 for any action at location 2, and 0 otherwise.

$$P(S_{t+1} = (2,\rightarrow) \mid S_t = (1,\rightarrow), A_t = \rightarrow) = 1$$

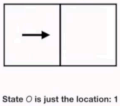
$$P(S_{t+1} = (2,\leftarrow) \mid S_t = (2,\rightarrow), A_t = \leftarrow) = 1$$

The state-transition diagram



0:11 / 3:32

Example 2: Not an MDP



State O is just the location: 1 or 2

Action is ← or →

Reward is +1 for any action at location 2, and 0 otherwise

Show that: $P(O_{t+1}=2 \mid O_t=2, A_t=\leftarrow) \neq P(O_{t+1}=2 \mid O_t=2, A_t=\leftarrow, (R_t=0))$

That is state O is not Markov