3 Acting in a Markov decision process

定义agent的状态和动作空间为：S和A,有如下定义：

* Model：一个模型是指agent环境的动态和奖励的数学描述，包含从一个初始状态开始采取一个行动并成功转化为状态的转移概率,同时奖励函数R(s,a)(确定或随机)被包含在状态s执行动作a时返回的信号之中。
* Policy：一个policy是一个S->A的函数映射，Policies可以是随机或者是确定的。
* Value function：值函数对应相应的policy ，并且值函数对于每一个状态而言，它是指agent从状态s开始并且遵循policy的未来的累积奖励(折扣)的总和(对未来奖励的预测)。

复习之前介绍的Markov property，考虑一个根据一些动态转移的随机过程()的演化。假如一个随机过程有Markov property，当且仅当：。基于历史数据的下一个状态的转移概率是等于基于当前状态的转移概率。在这样的场景下，当前状态是足以有效描述随机过程的历史统计数据(the current state is a sufficient statistic of history of the stochastic process)。我们称未来和给定现在的过去是独立的(the future is independent of the past given present)。

我们将在上述的基础上建立Markov process,接着是Markov reward process,最后基于它们两个定义Markov decision process(MDP)，我们将会在最后讨论一些算法使得我们在MDP完备的情况下做出好的决策。

3.1 Markov process

通常一个Markov process是一个具有马尔可夫属性的随机过程，因此我们说Markov process具有无记忆(memoryless)的特性。针对这一节的目的，我们将增加在强化学习中很常用的两个假设：

* + Finite state space：Markov process的状态空间是有限的。这意味着对于Markov process()而言状态空间S,|S|<∞,这样对于所有的Markov process的实现，我们有(目的是将状态序列化?)成立。
  + Stationary transition probabilities：转移概率和时间是无关的,stationary就是不与时间域相关(does not depend on time step),从数学的角度：



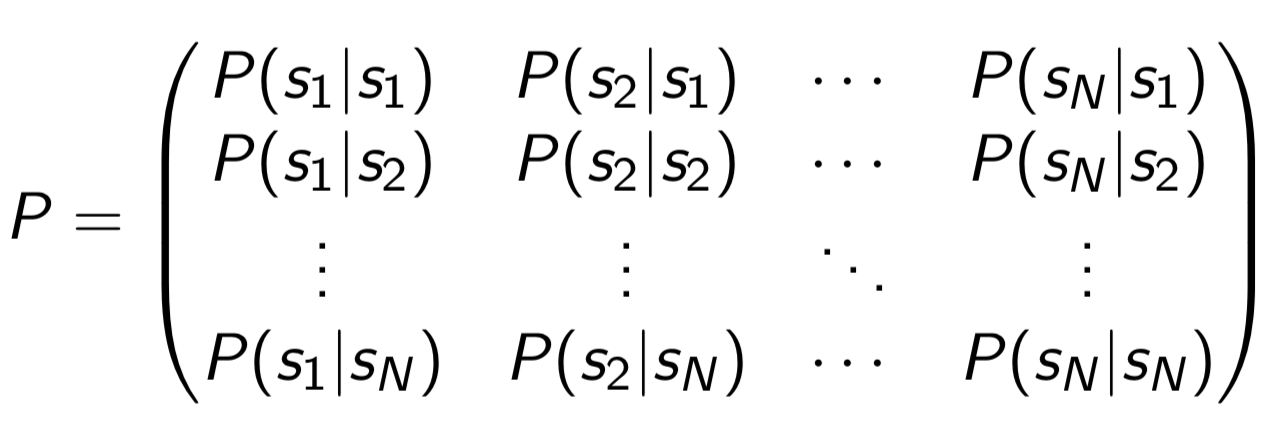
除非有其他情况被指出，否则我们将总是假设这两个条件用于支持我们在这一章节遇到的Markov process，包括之后介绍的Markov reward process和Markov decision process，它们是增加了额外的处理结构的Markov process，注意，符合Markov process这些假设条件的也可以被成为Markov chain，虽然Markov chian的精确定义跟Markov process有所不同。

对于Markov process而言，这些假设引导出了一个非常漂亮的动态转换的特性：转换概率矩阵P的维度为|S|×|S|，矩阵第(i,j)元素的值表示:,i,j可以表示状态S的任意顺序。值得注意的是矩阵P是一个非负的行随机(non-negative row-stochastic)矩阵,也就是说每行的和为1(行随机矩阵又称概率矩阵，转移矩阵，概率转移矩阵，马尔可夫矩阵)。

往后，我们定义Markov process为二元组：(S,P),它由下面的选项组成：

* S: 一个有限的状态空间
* 一个符合的转移概率模型

如果状态N是一个有限的数字，那么P可以表示为一个矩阵:



随机矩阵实际上应当分为行随机和列随机矩阵，行随机是指方阵的行和为1，而列随机矩阵就是其列和为1的非负矩阵，同时满足行和列就是双随机矩阵，单位矩阵就是一种双随机矩阵，其实只要研究行矩阵的性质，毕竟列随机只是行随机的转置。马尔可夫矩阵用于描述随机过程中的“转移概率”。

**练习3.1**

(a)证明矩阵P是一个row-stochastic矩阵

因为，所以第i行其实是表示状态的所有可变换的状态，所以所以为行随机矩阵。

(b)找到一个特征值为1的行随机矩阵，并找到其对应的特征向量

单位矩阵I，因为A的行的和为0,所以下式成立,即1是A的特征值。



(c)找到一个行随机矩阵，其任意特征值有最大值1

反证法：

假设存在，

假设中的最大值为,又A为行随机矩阵,所以行和为1,所以

，加入则存在这是矛盾的,所以行随机矩阵的特征值最大值为1。

**练习3.2**

向量的最大范式(无限范式)记为:



如上式,它是表示x分量的最大值的绝对值。对于矩阵:



(a)证明符合范式的属性，上述定义的矩阵又称为”诱导无穷大范式(induced infinity norm)”

反证法:假设不符合范式的属性，则，中的分量不是最大值

因为是最大范数所以是最大值，所以均是最大值(分母相同)，于假設相反，所以其符合范式的属性。

(b)证明：



这个求和就是取每一行的最大值,显然符合最大范数的定义

(c)得出结论：如果A是行随机矩阵则

这必须符合啊哥，如果是行随机矩阵意味着每行的和均为1，所以必然为1啊

(d)证明对于wpsoffice

这好像没什么好证明的?

3.1.1 Example of a Markov process: Mars Rover

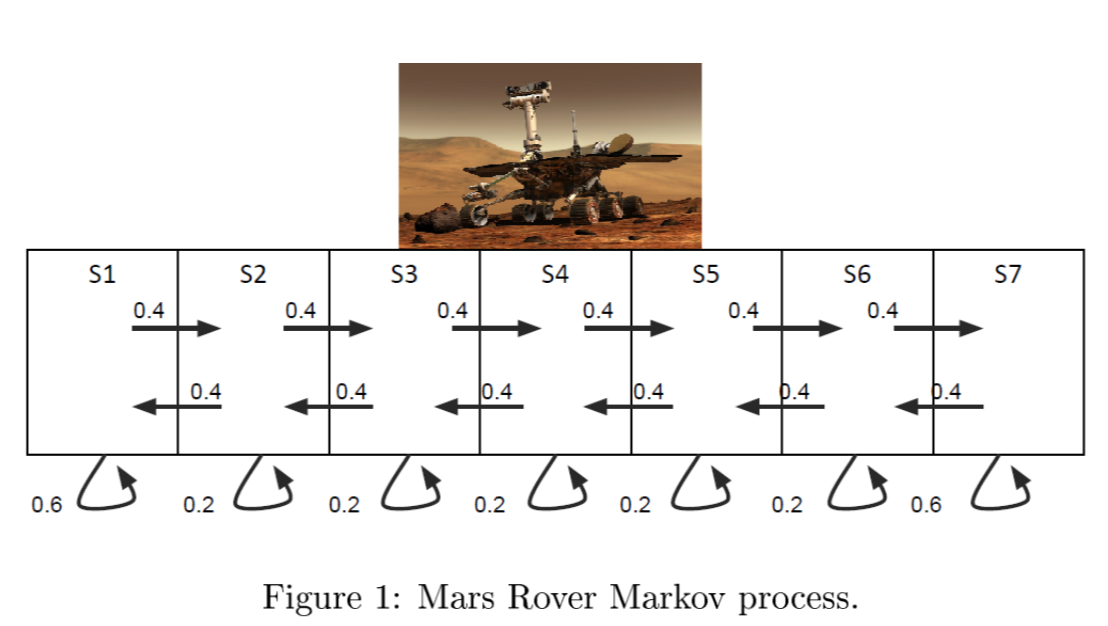
为了加强我们的理解，思考下图的Markov Process。我们的agent是火星漫游者，其状态为：。状态的转换概率如图中箭头所示。例如，当漫游者在状态S4的时候，它的下一步(下一个时间步骤)能够走到S3,S4,S5,其概率分别为0.4,0.2,0.4。

假设漫游者从S4出现，在Markov process中的一些可能的步骤(episode,插曲,剧集)为：

-S4,S5,S6,S7,....

-S4,S4,S5,S4,S5,S6...

-S4,S3,S2,S1...



**练习3.3**

思考上图Markov process，写下转换概率矩阵。



3.2 Markov reward process

一个Markov reward process是Markov process,可以描述为Markov chain + reward，他们具有同样的奖励函数和折扣因子。通常用一个元组来表示：(S,P,R,γ):

* + S:一个有限的状态空间
  + P:一个有确定概率的P(s’|s)的转换概率模型
  + R:奖励函数，从state映射到reward(实数),比如:S→R
  + γ:一个0到1之间的折扣因子

注意，没有actions.

之前我们已经解释了S和P在Markov process中所扮演的角色。下一步我们将会解释在Markov reward process中的奖励函数R和折扣因子γ的概念，额外的还会定义并解释一些重要的概念，例如horizon,return, state value function

3.2.1 Reward function

在一个Markov reward process中，当,当前状态s成功转换为下一个状态s’时，奖励是依赖于当前的状态来获取的(转换到s’返回的奖励是当前s的奖励)。因此对于Markov process,每一个转换都会伴随一个奖励所以一个已知的Markov reward process的剧集(epsoide)可以表示为，我们应该注意到这个奖励可以是随机的也可以是确定的。对于一个状态s,,我们定义期望奖励：

 (4)

当Markov process从状态s开始，R(s)是s在第一次转换期间获取的期望奖励。就好像静态转换概率一样，我们也做出如下假设：

* Stationary rewards：在Markov reward process中固定的奖励意味着他们与时间是相互独立的。在确定的情况下，数学上意味着对于所有可实现的过程我们都必须有(假如不同时间段的相同状态有相同的奖励值)：

 (5)

在随机奖励的情况下，我们要求以当前状态为条件的奖励值的cumulative distribution functions(cdf)与时间无关，数学上可以写为：

 (6)

中的cdf是基于。注意结合(5)和(6),我们此外还有如下关于期望奖励的结果：

 (7)

我们将看到只要Markov reward process的“与时间无关的固定奖励(stationary rewards)”是正确的，其实我们感兴趣的仅有期望奖励R(这个R是最重要的),我们甚至可以完全废除。“reward”这个单词可以再R和中交换使用，从上下文来看是非常容易理解的。最后注意，在有限的状态空间S中，R能够被表示为向量，其维度为|S|。

**练习3.4**

(a)在静态转换概率和奖励的假设下，证明(7)。

设状态为i,其回报为r,在状态i的情况下其状态转移概率和必然为1,因为下一步发生是必然事件,所以最后还是为r。数学描述如下：



3.2.2 Horizon,Return and value function

接下来我们定义Markov reward process的 horizon,return,value function的标记：

**Horizon**：一个Markov reward process的Horizon记为H，代表该过程的每个episode的时间步骤，horizon可以是无限或有限的。如果是有限的，该过程也称为*finite Markov reward process。*

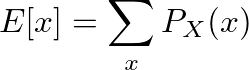
**Return**：一个Markov reward process的return被记为：,它是值从时间t开始直到horizon H的累积折扣奖励的总和，我们可以通过下列的数学公式来描述，强调一下,是当前时刻及未来的回报总和，再次强调是当前时刻及未来：

 (8)

**State value function**：一个Markov reward process的状态值函数记为：,一个状态的state value function被定义为在时间为t,状态为s的期望返还(expected return),通过下列公式描述：

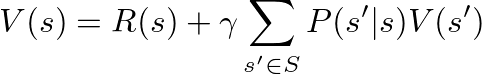
/var/folders/4_/wsyh4np55ms9lgn8jhhpylgr0000gp/T/com.kingsoft.wpsoffice.mac/wpsoffice.Xy1010wpsoffice

值函数其实是一个期望值，是基于当前状态S的的期望：



计算MRP的值方法：

1. 模拟估计
   1. 生成大量的剧集
   2. 求平均返还
   3. Concentration inequalities bound how quickly average concentrates to expected value
   4. 不需要假设Markov structure
2. Markov property yields additional structure
3. MRP的值函数符合：



注意，当horizon H是无限的时候，公式(9)以及固定的奖励以及状态转移概率的假设意味着 ,即，当在不同时间段的同一状态其状态值函数相同，同时在这种情况下，我们将定义(即任一不同的状态s，其状态值函数同第一次的状态值函数相同,即当时间段为a,b,c时agent均处于状态s，下式都成立)：

 (10)

**练习3.5**

(a)假设已知静态的转移概率，奖励函数，并且horizon H是无限的，使用(8)和(9)证明

根据(5)，假如有静态的奖励函数，则



再次强调，这小节之前的固定假设，是指不同时间段的相同状态有相同的奖励值；是指相同状态不同时间段有相同的状态值函数的值。

3.2.3 Discount factor

注意在(8)中定义的返还,如果horizon是无限的并且=1，那么return将会变成无限大，即便rewards全部都是一个个固定的数值。如果这种情况发生，那么值函数V(s)也会变成无限大。这样的问题是无法使用计算机解决的。为了避免这样数学上的困难，并且使得问题是可以计算的，我们设置，这样在未来的时间步骤中的奖励将会随时间呈现指数级的下降，就例如公式(8)中的return计算方式。大部分情况下称为折扣因子。除了为纯粹的计算的理由，也应当注意到它和人类的行为大致相同-人类倾向于更加重视即时奖励，而不是一段时间后的奖励。时，我们仅关心当下的奖励，时，我们认为未来的奖励和现在的奖励是同等重要的。最后，我们可以注意到若Markov rewards process的horizon是有限的，比如说*H*<∞，我们就可以设置γ=1，因为返还和值函数也都是有限的。

**练习3.6**

Consider a finite horizon Markov reward process, with bounded rewards.Specifically assume that  such that across all episodes(realizations).

(a)show that the return for any episode as defined in (8) is bounded.



假设H是有限的，设h=H-1,且根据假设|r|<M，我们可以得出



显然M和h均是常数,所以得证

(b)Can you suggest a bound? Specifically can you find C() such that  for any episode?



**练习3.7**

Consider a finite horizon Markov reward process, with bounded rewards and <1.

(a)Prove that the return for any episode  as defined in(8) converges to a finite limit.Hint: Consider the partial sums .show that  is a Cauchy sequence.

此题真的没什么好证明的

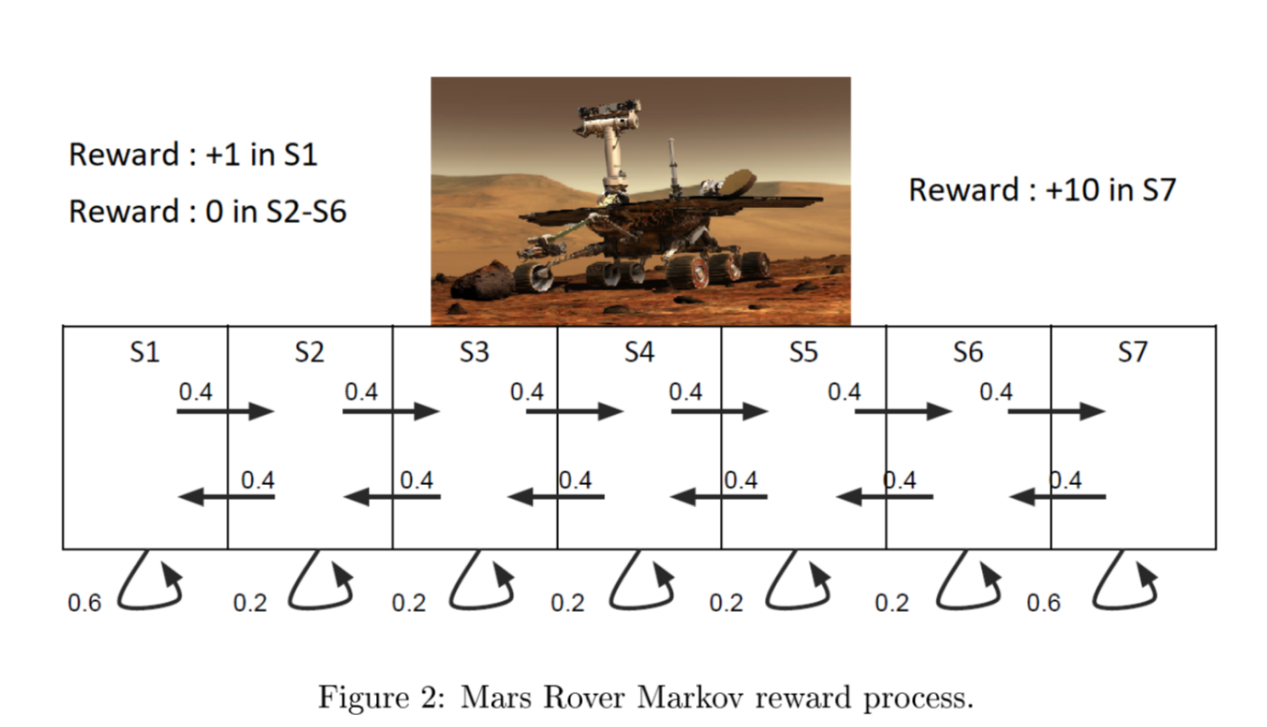


3.2.4 Example of a Markov reward process:Mars Rover

我们考虑下图2中的Markov reward process。如图所示，状态和转移概率都和练习3.3中的Markov process Mars rover一样。当状态处于{S2,S3,S4,S5,S6}时奖励均为0,当处于S1,S7时，奖励分别为1和10。显然奖励是固定的，在本例中,假设γ=0.5。

为了说明计算过程，我们假设漫游者的初始状态在S4。考虑下述情况：当horizon是有限时，设H=4，一些可能的剧集(episode)的回报设为，我们可以给出如下算法：(离开S1时才能得到S1的奖励)





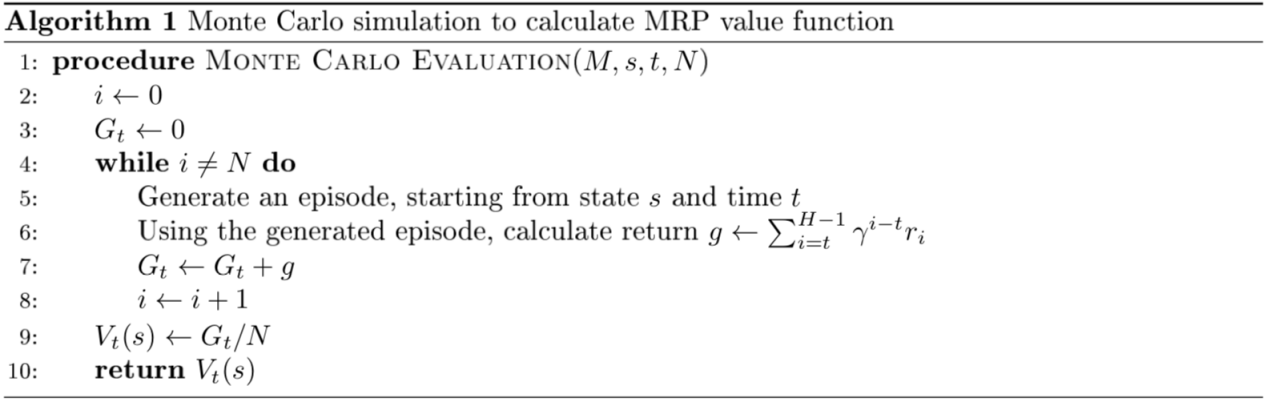
3.3 Computing the value function of a Markov reward process

这一小节我们会给出几种不同的方法去计算Markov reward process的value function：

* Simulation
* Analytic solution
* Iterative solution

3.3.1 Monte Carlo simulation

第一个方法是使用Markov reward process的转移概率模型和奖励生成大量的剧集。对于每一个剧集的回报是通过计算平均回报得到的。关注不等式边界如何快速的将平均收敛到均值。对于一个Markov reward process M=(S,P,R,γ),状态为s,时间为t,模拟的剧集数量为N,算法伪代码如下图所示：



3.3.2 Analytic solution

这个方法仅适用于无限的horizon Markov reward processes，并且其γ<1。使用公式(9)，horizon是无限的，同时我们可以为任意的状态使用固定的Markov property：下面定义为公式(11)

 (11)

(a)是源自(8),(9),(10),(b)是源自law of total expection，(c)是源自markov property和静态属性。(d)是来源于公式(4).对于公式(11)的最后推导结果有一个非常漂亮的解释：第一项R(s)是即时回报，第二项是未来回报的折扣总和。值函数V(s)是这两个数的和。只要|S|<∞，我们可以用矩阵的形式写出下列公式：

 (12)

P是早前介绍的概率矩阵，R和V是维度为|S|的矩阵，其列向量由对应的R(s)和V(s)组成。公式(12)可以再被写为：，它有另一个形式：，注意，并且P是行随机矩阵，是非奇异，因此这里可逆，所以公式(12)总是有唯一解，然而计算的代价却是：，因为这包含了逆的计算，因此当对于状态空间巨大的条件这种计算方法是完全不适用的。

**练习3.8**

思考矩阵，(a)显示这个矩阵的特征值，并且找到特征向量。(b)对于,使用练习3.1的结果得出是非奇异矩阵，并且是可逆的

(a) 

(b)P行随机矩阵，可逆，且每行和为1，所以gamma\*P的每行小于1，所以I-gamma\*P每个值都会在0到1之间，所以det不为0，所以为非奇异矩阵，且可逆。

**练习3.9**

考虑Markov reward process，(a)如果horizon H是无限的，计算所有状态的值函数。

公式(11)处理

3.3.3 Iterative solution

我们现在给出一个迭代解去估算在无限horizon的情况下的值函数()，同时对于有限horizon情况，我们可以用动态规划来解。令人惊讶的是这两个算法来起来惊人的相似，为了讲明白这两者的不同，我们先考虑有限horizon的情况。这非常容易能够证明(通过同证明公式(11)基本类似的过程)公式(11)在有限horizon条件下的情况**(注意下,这里右边等式是t+1,无限的情况是t-1)**：

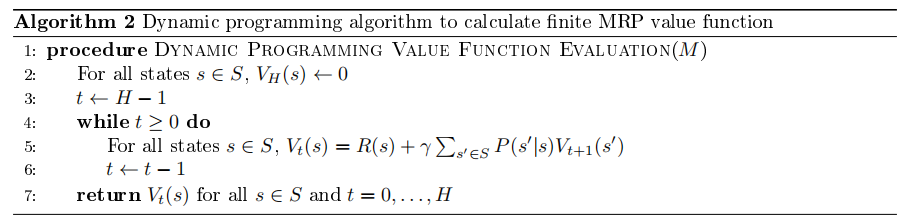
 (13)

**练习3.10**

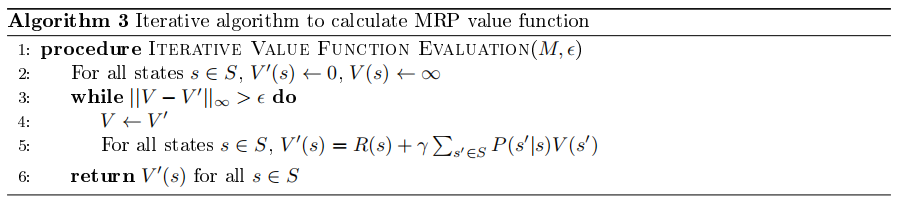
证明对于有限horizon的Markov reward process而言,公式(13)是成立的。

没啥好证明的，有限的话，最后会收敛

这些方程可以引导它们自己身形成一个动态规划的解，伪代码如下图所示：



现在看一下无限horizon的迭代算法()，伪代码如下所示：



对于算法2和算法3而言，每次循环计算的代价都是，这个比起先前的是一个改进，然而算法3需要一定次数的迭代才能收敛，而收敛是依赖于公差wpsoffice的数量级。

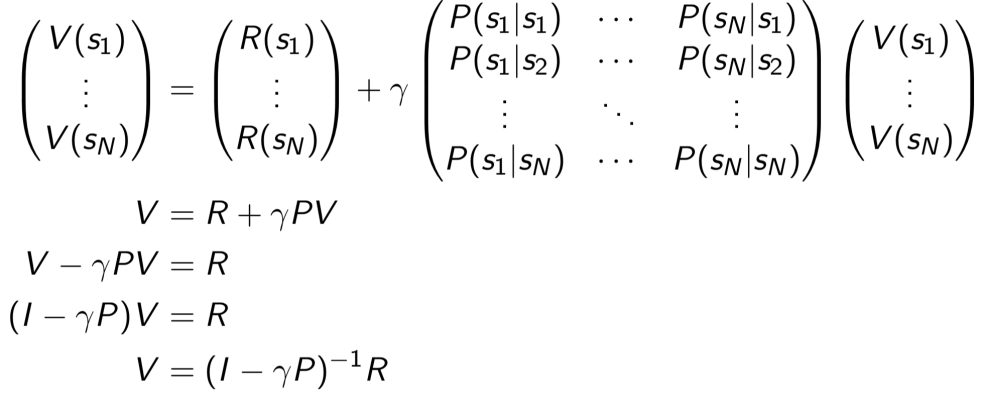
虽然在有限时域的情况下算法2的正确性是显而易见的，但是对于无线时域的情况算法3是否能总是收敛并且收敛到正确解还不是很明确。这两个问题的答案都是肯定的，我们将通过下列定理证明。

**定理3.1**.算法3总是能终止，若算法的输出是V’，并且我们将真实解表达为

V=，那么我们会有误差估计：



其中V的推导如下所示(其时间复杂度为:wpsoffice):



证明，我们已知向量空间配备了最大范数(见练习3.2),复习一下构成一个巴拿赫空间(见Section A对范数空间的讨论)。我们首先注意到V和算法3所有的迭代都是的元素。

定义算子B：(也被成为“Bellman backup”operator，贝尔曼算子)，该算子作用在元素：

 (14)

用矩阵的形式可以改写为：

 (15)

我们首先证明算子B是一个严格收敛的算子(查看A.3)。对于每个,使用公式(15)，我们有：

(16)

这里第二步的推导是根据练习3.2，注意，这里的，我们可以得出结论B是严格收敛于。根据定理A.5，我们得出B有一个固定值。从公式(15)和(12)，可以得出,因此V是B的一个固定值，所以通过唯一性证明它必须是一个固定值。

接下来我们思考算法3产生的迭代(如果它不允许终止)，将其表达为：，注意，迭代符合下列的关系：

 (17)

通过定理A.5，我们进一步归纳出是一个柯西序列，通过定义A.1,我们得出，使得.这完整的证明了算法3的终止。注意定理A.5也暗含着(看定义A.2)。

证明当算法终止时的错误边界，令算法在k次迭代后终止，并且令最后一次迭代为.我们有。然后使用三角不等式和已知的条件,我可以得到：

 (18)



最终我们可以得出：

 (19)

**练习3.11**

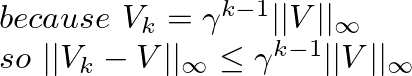
假设在算法3中，初始条件V’改为随机值，问(a)算法还会收敛吗?(b)算法还会保留同定理3.1同样的误差估计吗?

2019.07.21,以以往的实验看，随便设值往往不会收敛，但是算法的误差估计是一样的。

**练习3.12**

假设定理3.1的条件均被支持，使用同样的符号证明下列引理:









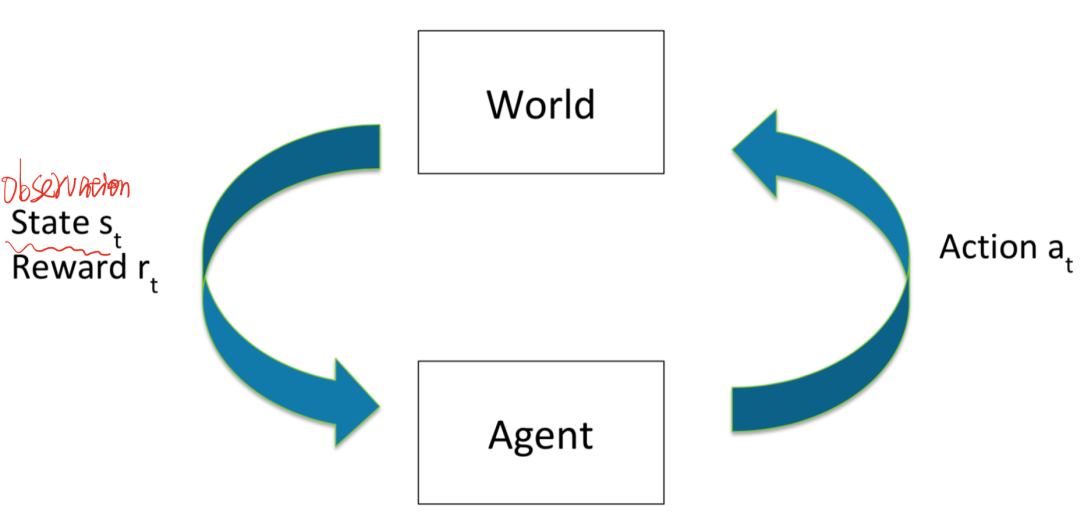
b和c没什么好证明 的。

3.4 Markov decision process

我们现在定义Markov decision process(MDP)。一个MDP继承了Markov reward process的基础结构，同时在一些关键部分又有些不同，例如它指出了agent的动作集合，也就是在每个状态下agent能执行的动作集。通常一个MDP用一个五元组表示:

* + S: 一个有限的状态空间
  + A：一个有限的动作集，改动作集是指每个状态s下能执行的可行动作
  + P：一个状态转移模型
  + R：奖励函数，是指从一个state-action对映射到reward(实数)，例如： 
  + ：折扣因子，范围在0到1之间

有一些已经在Markov reward process上下文中解释过了。然后再这篇MDP的讲述中，我们必须要提醒各位一些重要的不同。基础的动态模型有状态空间S和动作集A，我们把这两个都设为有限的。在某个时间i，agent处于状态,并从该状态开始从动作集中选择动作，执行后会得到一个奖励,并且成功抵达下一个状态。一个MDP的剧集可以表示为:。



在Markov process和Markov reward process中转换概率仅仅是一个成功状态或者当前状态的函数，而在MDP中，转换概率是成功转移到下一个状态的函数，是指下一状态在i时刻，状态为，动作为的函数，记为：。我们仍然假设静态转换概率的原理，在MDP中以数学的形式可以写为：

 (20)

在MDP中,i时刻的奖励是依赖于,,相比于Markov reward process的奖励是依赖于当前状态，两者是有大大的不同。奖励同Markov reward process中一样可以是随机的也可以是固定的，我们假设奖励是固定的，并且唯一有价值的变量是期望奖励，我们将其表示为R(s,a),s表示状态,a表示动作：

 (21)

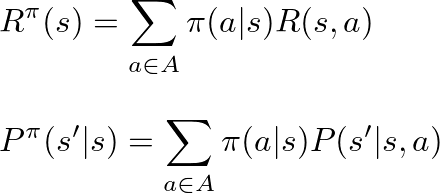
折扣因子,horizon H,return 的符号表达在MDP和MRP中是一致的，然而state value function在MDP中是有明确的修改，在下一节将指出。

在这里指的指出的是:

wpsoffice

准确的来说在：

/var/folders/4_/wsyh4np55ms9lgn8jhhpylgr0000gp/T/com.kingsoft.wpsoffice.mac/wpsoffice.Ao1010wpsoffice中,



这意味着我们能够使用同样的技术去估算MDP策略的值，因为我们能够通过计算wpsoffice和/var/folders/4_/wsyh4np55ms9lgn8jhhpylgr0000gp/T/com.kingsoft.wpsoffice.mac/wpsoffice.hY1010wpsoffice的值,进而计算MRP的值。

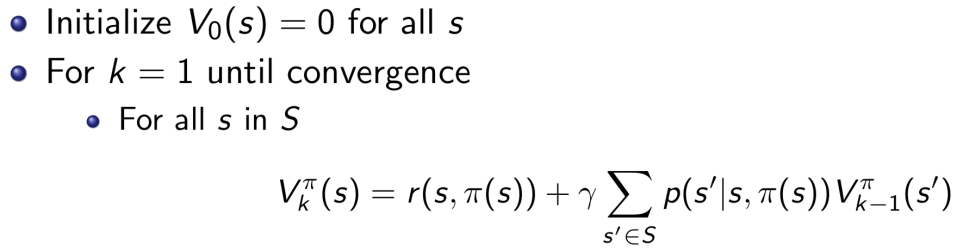
3.4.1 MDP policy and policy evaluation

给出一个MDP，一个策略是指出在每一个状态下agent会采取怎样的行动。一个策略可以是确定的也可以是随机的。为了阐述这些情况，我们考虑把策略看做是在某个状态的动作集之上的概率分布(probability distribution over actions given the current state)。值得注意的是策略是可以随着时间而变化的，尤其在有有限时间域的MDP中是相当正确的。我们将使用来作为策略的符号，无限时间域可以写为:,是指在t时刻的策略。我们将不随着时间变化的策略称为“固定策略(stationary policies)”,并将它们记为:,固定策略也可以写为:。对于一个固定策略，如果在时刻t且处于状态s,agent会根据给定的概率选择一个动作，并且这个概率不依赖于时间t，然而对于不确定策略而言，其概率依赖于时间t，我们将其表示为：。

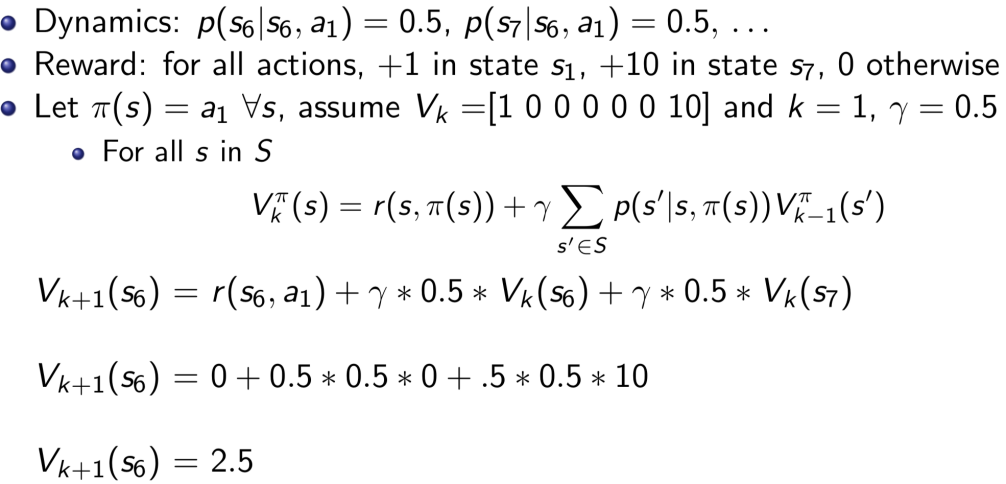
注意：

wpsoffice

MDP Policy Evaluation,Iterative Algorithm:



以火星漫游者为例:



给定一个策略能够定义两个将MDP映射到的函数：state value function和state-action value function：

* state value function：状态值函数，被定义为在t时刻从状态开始遵循策略的期望回报(注意是一个期望值)，用数学表达式可以写为：，是关于策略的期望值。通常我们为了简化标记会拿掉下标。因此，E被记为关于策略的期望，如果没有别的说明，我们就记为：

 (22)

可以注意到当时间域是无穷时，假设奖励，转换概率和策略均是固定的，对于公式(22)意味着：for allfor all i,j=0,1....，并且在这种情况下，我们将定义一个类似Markov reward process的公式:  
  (23)

* state-action value function：对于一个状态s和一个动作a的行动值函数被定义为在t时刻,从状态开始,执行动作,随后执行策略的期望回报。以数学形式记为:

 (24)

在无限时间域的条件下和状态值函数类似,假设奖励，转换概率和策略均是固定的，意味着：for allfor all i,j=0,1....，并且可以记为:  
  (25)

**练习3.13**

有一个固定策略π=(π,π,...)。假设转换概率和奖励都是固定的，同时时间域是无限的，使用定义(22)和定义(24)证明：for all ,

(a) 

因为和 

(即任一不同的状态s，其状态值函数同第一次的状态值函数相同,即当时间段为a,b,c时agent均处于状态s，下式都成立)

所以

(b)  for all i,j = 0,1....

证明同(a)。

在无限时间域的情况下，关于固定转移概率和奖励的假设会导致以下重要的恒等式，该恒等式对于一个固定的策略将连接state value function和state-action value funcation

 **(26)**

(a)的推断来自于公式(24),(25)，(b)是由law of total expection(期望定律?)，(c)源自Markov属性，(d)是源自练习3.13和期望的线性属性。

在MDP上指定一个固定的策略的一个有趣方面是：对于策略的值函数的计算是等价于Morkov reward process上的值函数计算。

尤其是我们定义的Markov reward process 中定义为：



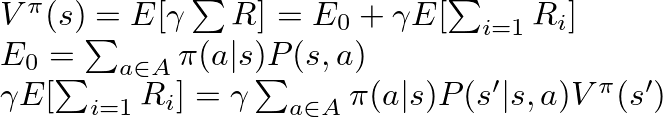
**练习3.15**

有一个MDP的固定策略。

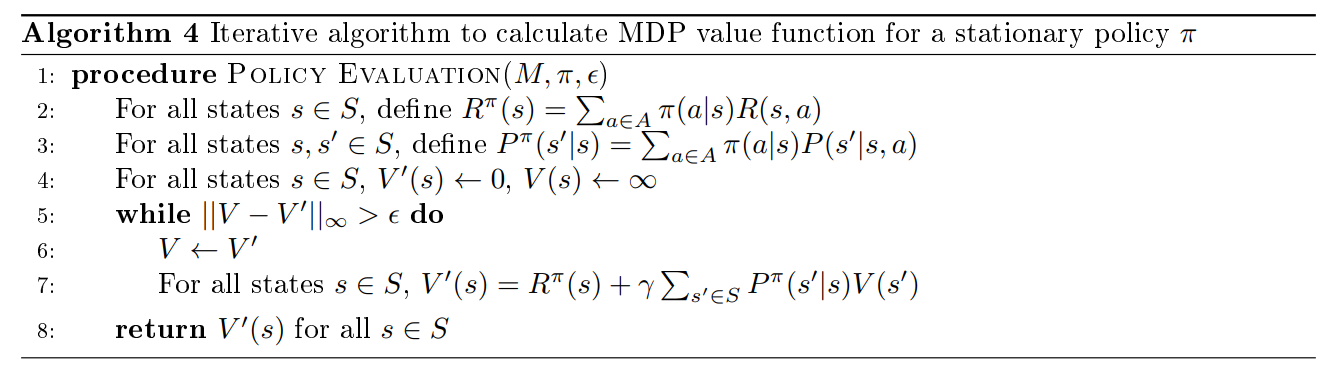
(a)通过使用公式(27)证明策略的值函数符合等式：



答:



对应策略值函数的估计能够通过使用Markov reward processes技术来解决。例如，假设,γ<1且无限时间域，计算给定固定策略π的值函数的迭代算法如下面algorithm 4。算法的输入是Markov decision process M=(S, A, P, R, γ),一个固定的策略π,和一个公差(tolerance)ε,并计算所有状态的值函数。



**练习3.16**

1. 证明当γ<1时,algorithm 4会收敛,提示,使用定理3.1
2. 考虑一个实数的正序列使得，假设algorithm 4在每个时终止，定义算法每个相应的输出表示为，证明,是策略的值。

解(a)

**定理3.1**.算法3总是能终止，若算法的输出是V’，并且我们将真实解表达为

V=，那么我们会有误差估计：



根据定理3.1：



(b) 好像没什么需要证明的?如果ε的值均接近于0,意味着V’→V。

3.4.2 Example of a Markov decision process : Mars Rover

思考figure3的MDP的例子，agent还是火星漫步者其状态空间为{S1,S2,S3,S4,S5,S6,S7}。agent在每一个状态均有两个动作:左,右，所以动作空间为A={TL,TR}。假设采取行动总是能成功，除非我们处在边缘，也就是S1和S7，那么这种情况agent将会保持原有的状态不变，也就是还处于S1和S7。这导致了两个动作(左,右)不同的转换概率矩阵，如figure3所示。每个状态的奖励对于所有的动作都是一样(奖励是固定的不随动作变化)，当agent处于{S2,S3,S4,S5,S6}时奖励为0，当状态处于S1和S7时，奖励分别为1和10，MDP的折扣因子

值的一提的是，火星漫游者有wpsoffice个确定性的策略，假设有A个动作,S个状态，则有wpsoffice,例如火星漫游者中A=2(左和右),S=7(S1-S7)。

在MDP中最优策略总是唯一的吗?显然不是的，可能存在不同策略但具有同值最优值函数的策略。

对于一个无限时间域的MDP问题而言，agent的最优策略永远是：

* Deterministic
* **Stationary(does not depend on time step)**
* Unique?不一定,可能具有相同的最优质

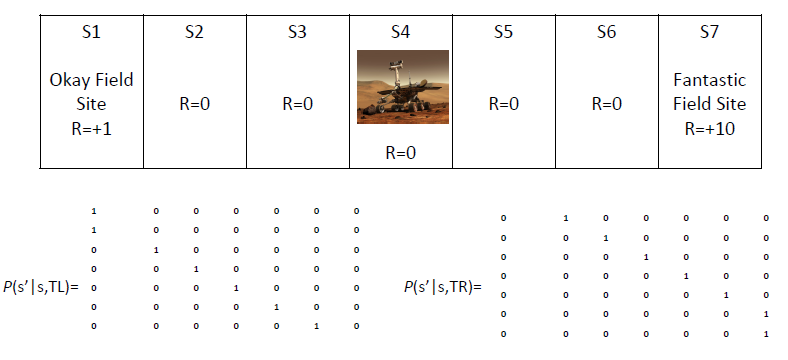


Figure 3: Mars Rover Markov decision process.

**练习3.17**

继续思考figure3的MDP讨论，设γ=0,然后思考一个固定执行TL的策略π(a)假设时间域是有限的，要求计算对于所有状态的策略的值函数。(b)当时间域是无限时要求计算策略的值函数，提示使用定理A.3。

Theorem A.3. Suppose T is a strict contraction on a normed vector space V (not necessarily Banach) with contraction factor γ= 0. Then T is a constant map.

Proof.

Consider an element v ∈ V , and let c = Tv. Now for every element w∈V , we have ||Tv−Tw|| ≤0, which implies ||Tv−Tw|| = 0. By property of norms this implies that Tw = Tv = c.

a,b都是V=[1,0,0,0,0,0,10]

3.5 Bellman backup Operators

这一节，我们要介绍bellman backup operators的概念，并证明它的一些性质，这对于下一节讨论MDP control是非常有用的。在定理3.1证明过程的公式(14), 公式(15),我们已经遇到了一个bellman backup operator。我们现在定义两个非常接近(但不相同)的Bellman backup operators:

* the Bellman expectation backup operator
* the Bellman optimality backup operator

3.5.1 Bellman expectation backup operator

假设给定一个MDP M (S,A,R,P,γ)以及一个固定policy π，它可以是确定的也可以是随机的。我们已经在第3.4.1小节见过MRP方程M’=(),在公式(27)中被定义。对于M的policy π的值函数被定义为，M’和M具有相同的值函数，我们分别使用MDP和MRP的值函数相对应。注意，的维度是在有限维的巴拿赫空间中(之前在练习3.2我们已经介绍了无穷范数)。

对于:，policy π, Bellman expectation backup operator 被定义为：

 (28)

我们应该已经注意到这个算子在algorithm 4中出现过一次。我们现在要证明这个算子的一些性质。

**定理3.2**.在公式(28)中定义的算子是一个压缩映射。如果γ<1，那么它将严格收缩并且有一个唯一固定点。

证明：给定,对应一个状态s,,我们利用三角不等式从公式(28)中可以得到：

 (29)

对于,公式(29)永远是正确的。我们总结出：



因此是一个严格映射在的算子。

根据公式(29),γ<1,我们推断是一个严格缩小的算子，因子应用定理A.5,它有一个固定的点。

**推论：3.2.1**。设γ<1。对于任意的,序列是一个柯西序列并且都会收敛到中的固定点(wpsoffice)。

证明：根据定理3.2，定理A.4,定理A.5。

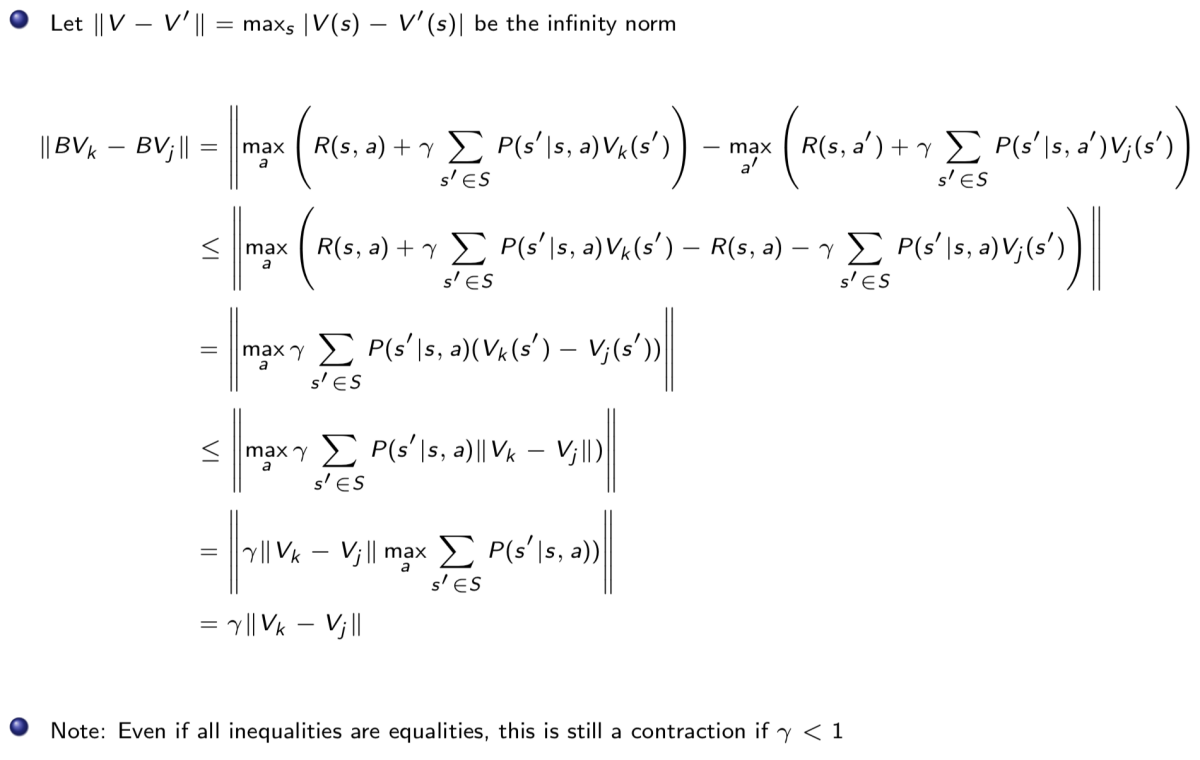
这意味着对于固定的plicy π,其值函数是中的一个固定点，我们如下推论：

**推论3.2.2**.设γ<1的无限时间域的MDP的policy 为π，其值函数为是中的一个固定的

证明：根据公式(28)的定义和练习3.1.5可以证明：



PPT完整证明严格收缩如下：



下一个定理证明Bellman expectation backup operation的“单调性”。

**定理3.3**.假设我们有，for all ,。同时对于每个固定policy π,我们有for all 。若不等式严格不等，例如for all ,那么我们有for all 。

证明：当 for all sS,利用公式(28):

 (30)

注意， for all sS和 for all sS是同样的步骤。

3.5.2 Bellman optimality backup operator

假设给定一个MDP M=(S,A,P,R,γ)。我再一次思考有限维的巴拿赫空间(配备无穷范数)。for every element U,Bellman optimality backup operation B\*被定义为：

 (31)

接下来我们同Bellman expectation backup operator类似的去证明分析Bellman optimality backup operation的性质。

**定理3.4**.对于每一个以及每一个状态下列不等式都是正确的：

(a)

 (32)

(b)

 (33)

我们首先证明(a),固定一个状态，利用公式(31)，条件是action space A是有限的，我们推断出存在，但没有必要不同，所以下面的式子可以成立：

 (34)

根据公式(31)中最大的定义，对我们也有：

 (35)

根据公式(34)和公式(35)，我们可以推断出：

 (36)

至此证明了第一个不等式(a)。对于第二个不等式注意到，我们对于所有的都乘上正数,，并求和：

 (37)

该结果可以利用最大函数的单调性进行证明。

证明(b),注意到通过交互律交换,我们可以从(a)中得到：

 (38)

然后联立公式(38)和公式(32),我们可以得到：



至此证明了b。

**定理3.5**.在公式(31)中定义的算子是一个严格缩小的映射，如果γ<1，那么它是一个严格缩小且有一个唯一的固定点。

证明：通过观察公式(32)以及定理3.4可以知道对于所有的,B\*最终是一个收敛的算子,所以尤其对于:

一定是正确的。

因此，，可以证明当γ在[0,1]区间时,B\*是一个收敛的映射算子。

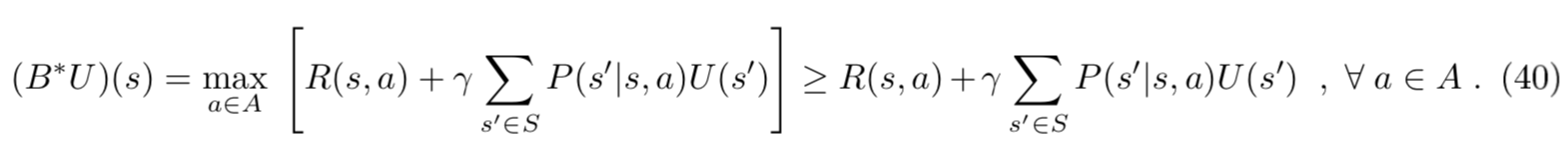
**推论3.5.1**.令γ<1，那么对于任意的，序列是一个柯西序列，并且会收敛到B\*的不动点。

证明:根据定理3.5,A.4,A.5。

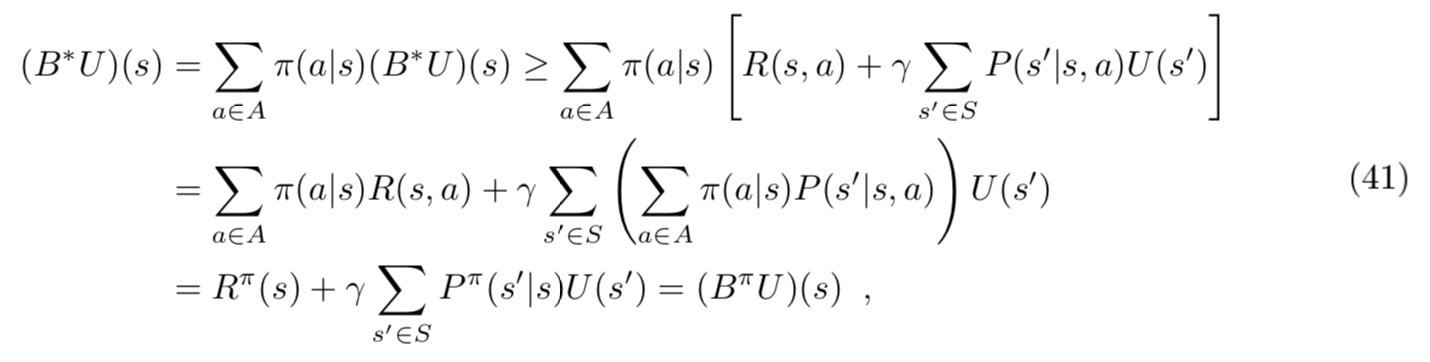
下一个定理是把与应用的结果与一些做比较。

定理3.6对于每个固定的policy π的每个U,和所有的wpsoffice，都有/var/folders/4_/wsyh4np55ms9lgn8jhhpylgr0000gp/T/com.kingsoft.wpsoffice.mac/wpsoffice.AI3448wpsoffice

证明:给定一个固定policy π,使得wpsoffice为Bellman expectation backup operator。给定一些。在给定一些wpsoffice。从B\*的定义(31)中，我们可以得知:



将公式(40)乘以π(a|s)，并对所有的 wpsoffice 做求和，给出下式:



方程最后的的推导是来自定义(27)和(28)

3.6 MDP control in the infinite horizon setting

到目前为止，我们已经具备了讨论”*MDP control*”的知识背景，在MDP control中我们需要寻找最好的policy(通常只有一个policy),该策略实现了最大的值函数，即在所有可能的策略中具有最大值函数的为最佳policy。在强化学习环境中，这也正是agent的目标。我们在这个小节中将第一次讨论无限时间域的情况，有限时间域的情况将在下一小节中讨论。我们这样做的原因是“无限时间域的情况”是一个更为困难的问题，并且它提出了许多数学上的挑战等待我们去解决。

首先，我们需要解决“通过找到最优政策，对我们究竟意味着什么?”的问题。准备的说，我们想要知道这样的最优策略是否存在，假设存在，我们将这个最优策略定义为,它的值函数至少和其他policy的值函数一样好。换句话说，我们需要确认一些policy的值函数真正能达到的上限!要理解这一点的微妙之处，请考虑最大化功能的例子: on(0,1) 定义为,值得注意的是这个问题没有解。But sup f(x)=1,although  for which this is attained。

我们首先精确给出了policy成为最优策略的定义。

定义3.1. a policy  is an optimal policy,iff for every policy ,for all t=0,1,.., and for all state 。

我们留给读者证明对于一个无限时间域MDP的状态，其最优策略的存在性，也就是说固定的最优策略是否存在。这个结果是非常直观的，同时也是非常重要的，如果存在最优策略的话,它大大减少了在寻找最优策略时需要考虑的策略范围。尤其它表明我们只需要考虑固定的policies。

**练习3.18**

(a)考虑一个无限时间域的MDP。令是该MDP的一个最优策略。证明存在一个固定的策略也是最优的。

因为每一个策略都存在一个最优策略，所以策略集存在一组值函数最大的策略集，所以得证!

接下来的两个定理改进了”练习3.18”的结论，并证明了我们可以将搜索限制在有限的确定性平稳策略集上。

**定理3.7**.The number of deterministic stationary policies is finite,and equals。

Proof

因为策略是具有确定性和稳定性，每个policy都能用一个函数表示，该函数:.这些不同函数的数量为:。这也证明了确定性的固定策略的数量是有限的。

**定理3.8**.如果是一个针对有限时间域MDP()的固定策略,那么存在一个确定的固定策略使得: for all states .一个这样的策略以如下的形式给出：

 (42)

它符合等式。

并且我们继续推论：



证明：详看英文文档L02 17pages。

**引理3.8.1**.使用定理3.8的符号，如果wpsoffice使得wpsoffice成立,则/var/folders/4_/wsyh4np55ms9lgn8jhhpylgr0000gp/T/com.kingsoft.wpsoffice.mac/wpsoffice.cU1010wpsoffice。在这种情况下我们说/var/folders/4_/wsyh4np55ms9lgn8jhhpylgr0000gp/T/com.kingsoft.wpsoffice.mac/wpsoffice.FC1010wpsoffice是一个严格比/var/folders/4_/wsyh4np55ms9lgn8jhhpylgr0000gp/T/com.kingsoft.wpsoffice.mac/wpsoffice.Un1010wpsoffice更好的策略。

证明引理3.8.1.可以通过将不等式(44)转换为严格的不等，再结合定理3.3可证明。

**练习3.19**.假设一个infinite horizon MDP with γ<1.定义II是一组确定的固定策略。

1. 证明wpsoffice,使得所有的/var/folders/4_/wsyh4np55ms9lgn8jhhpylgr0000gp/T/com.kingsoft.wpsoffice.mac/wpsoffice.Mq1010wpsoffice，同时对于所有的/var/folders/4_/wsyh4np55ms9lgn8jhhpylgr0000gp/T/com.kingsoft.wpsoffice.mac/wpsoffice.zJ1010wpsoffice成立.

解:

因为wpsoffice是最优策略，前面已经证明了最优策略的存在性，所有其值函数是最大的。

1. 得出结论wpsoffice是一个最优策略。提示:定理3.10

解：

显然每个/var/folders/4_/wsyh4np55ms9lgn8jhhpylgr0000gp/T/com.kingsoft.wpsoffice.mac/wpsoffice.sf1010wpsoffice都是最优的所以wpsoffice的值函数也是最大。

到目前为止,我们已经证明了最优策略的存在性,并且得出结论:一个确定性的固定策略就足够了。这就允许我们做出如下定义：

**定义3.2**.对于一个无限时间域的MDP其optimal value function定义为:

/var/folders/4_/wsyh4np55ms9lgn8jhhpylgr0000gp/T/com.kingsoft.wpsoffice.mac/wpsoffice.Cf1010wpsoffice

同时存在一个确定的最优策略:

wpsoffice

这个最优策略使的：

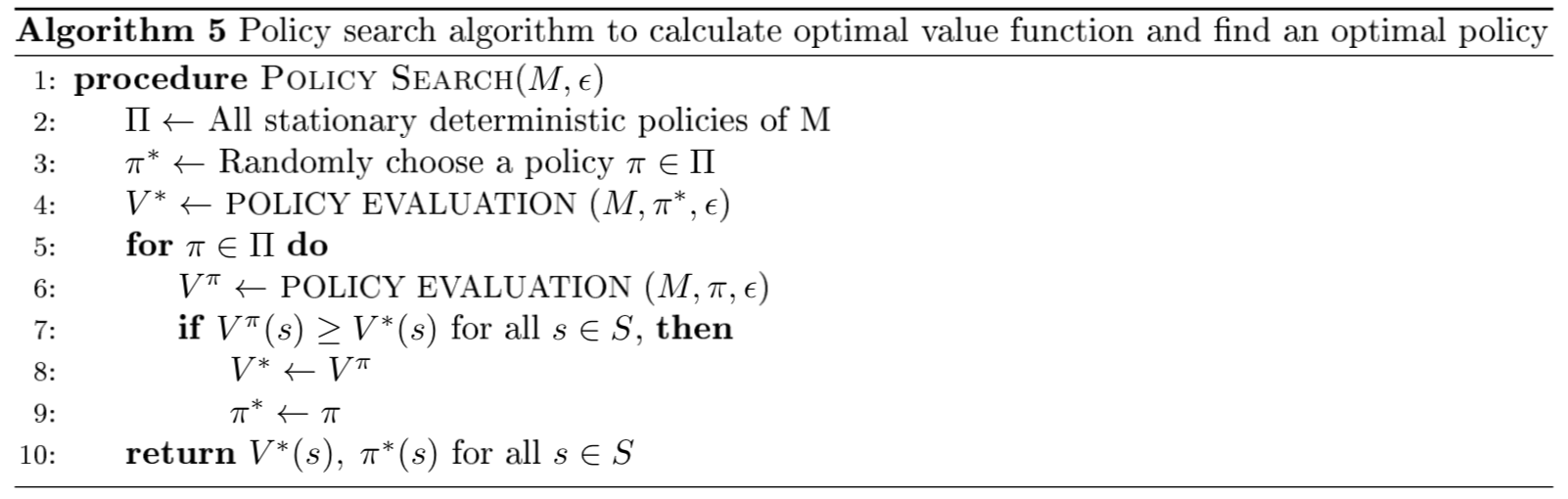
wpsoffice

注意wpsoffice是一组确定的固定策略。

接下里我们将介绍一些算法去计算最优值函数和最优策略。

3.6.1 Policy Search

由定义3.2可以推导出一个穷举算法被称为policy search，该方法用于查询最优值函数wpsoffice以及最优策略wpsoffice,该算法的伪代码如下所示,算法以无限时间域MDP M={S,A,P,R,γ}以及tolerance wpsoffice为输入,其中wpsoffice是为了控制policy evalution的精确度，并返回最优值函数和一个最优策略。



Algorithm 5总是在它检测完所有的wpsoffice的确定性的稳定策略后终止。它的运行时复杂度为wpsoffice。当wpsoffice=0时,可以去证明算法的正确性，即，每一次迭代中policy evaluation都要精确的完成。实际上wpsoffice会被设置成一很小的数，大致在/var/folders/4_/wsyh4np55ms9lgn8jhhpylgr0000gp/T/com.kingsoft.wpsoffice.mac/wpsoffice.QX1010wpsoffice，

**定理3.9**.Algorithm 5在wpsoffice=0时返回的是最优值函数和最优策略。

证明:令wpsoffice为一个最优策略，因此wpsoffice。因为算法为运行检测在wpsoffice在每一个策略，也就意味着wpsoffice一定是算法进行一些迭代后被选中的策略。因此在之后迭代的策略的值函数不能再严格增长。未来的迭代可能会选中一写不同形式但具有相同最优值函数的不同policy，定理得证。

显然Policy Search的缺点很明显就是计算量太大，因为策略的数量为wpsoffice是一个幂级增加的数量。

**练习3.20**.对于在3.4.2讨论并由图3展示的MDP，假设时间域是无限的。(a)agent能有多少种确定的固定策略(b)如果γ<1,那么最优策略是唯一的吗?(c)如果γ=1，存在唯一的最优策略吗?

解:

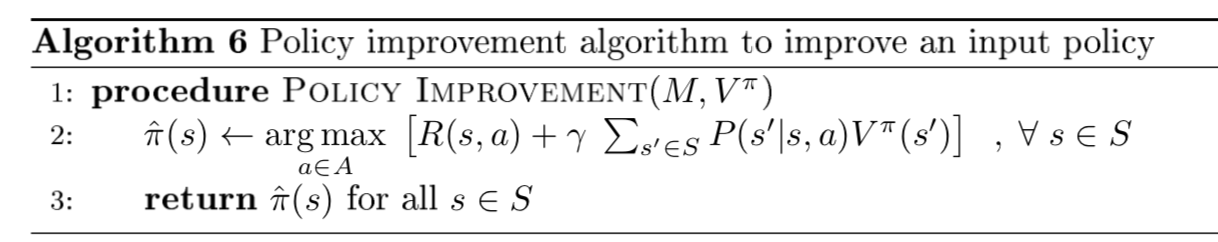
(a)存在wpsoffice个固定策略。

(b)存在最优策略但不唯一。

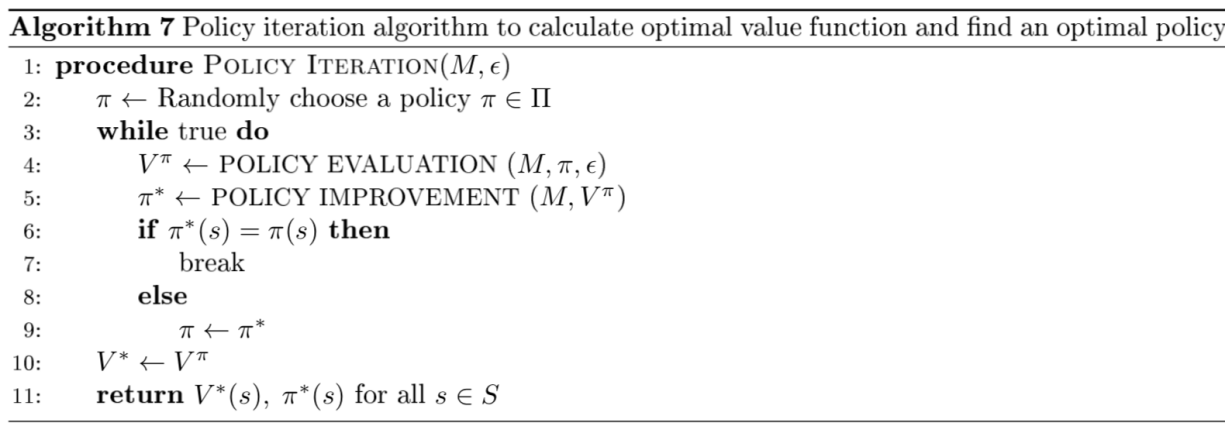
(c)不存在

3.6.2 Policy Iteration

现在我们要讨论一个比policy search更有效的算法叫做policy iteration,该算法是定理3.8的直接应用，定理3.8指出，给定任何稳定策略，我们可以找到一个不比当前存在的策略差的确定性的固定策略(也就是找到一个至少跟当前策略一样好的策略)。特别指出，该定理也适用于确定的策略。这个简单的步骤称为policy improvement，其伪代码如algorithm 6所示。其思路是我们有一个已经的policy和其在无线时间域下的值，然后慢慢的改进它。



Algorithm 6的输出总是被保证至少和值函数为wpsoffice的policy π一样好。并且使用贪心算法去尝试改进策略。当policy evaluation算法(algorithm 4)开始执行迭代算法时，就产生了policy iteration算法。Policy iteration的伪代码如algorithm 7所示。



Algorithm 7的正确性就留给读者在下一个练习证明。注意，algorithm 7总是会停止，因为通过定理3.7可知总是存在有限数量的stationary deterministic policies。

显然policy iteration的最大迭代次数为:wpsoffice

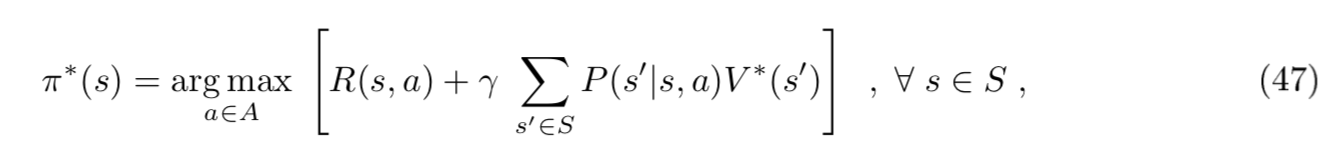
Policy Iteration和Value iteration是不同的,因为策略迭代是说你已经有了一个策略，并且知道它的值是什么，但它可能不是好的策略。值迭代表示已经知道策略的最优值是什么，但前提是必须要运行K个时间步。

它们计算不同的东西，但是最终会收敛成同样的值。

3.6.3 Value iteration

我们现在讨论value iteration算法，它是计算MDP最优值函数和最优策略的技术。为了开发这种方法，我们需要下列的定理：

**定理3.10**.For a MDP with γ<1,let the fixed point of the Bellman optimality backup operator B\* be denoted by wpsoffice.Then the policy given by:

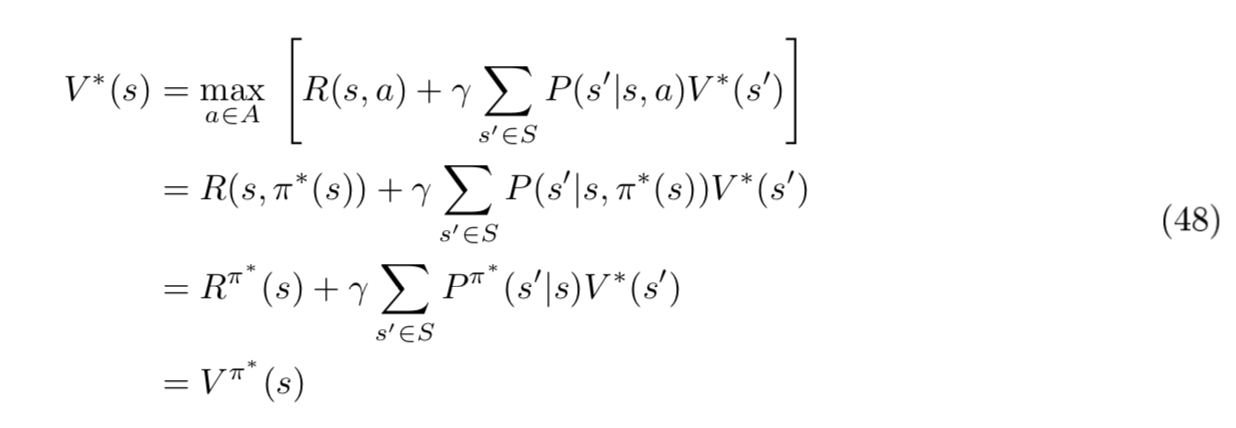


这是一个stationary deterministic policy.wpsoffice的值函数符合:/var/folders/4_/wsyh4np55ms9lgn8jhhpylgr0000gp/T/com.kingsoft.wpsoffice.mac/wpsoffice.HE1010wpsoffice,并且/var/folders/4_/wsyh4np55ms9lgn8jhhpylgr0000gp/T/com.kingsoft.wpsoffice.mac/wpsoffice.tG1010wpsoffice也是算子/var/folders/4_/wsyh4np55ms9lgn8jhhpylgr0000gp/T/com.kingsoft.wpsoffice.mac/wpsoffice.ue1010wpsoffice的一个固定点。重点是其阐述了存在一个stationary deterministic policy wpsoffice,其值函数是B\*的一个固定点。通俗的讲，wpsoffice是一个最优策略。

证明。我们通过公式(47)的定义了解到wpsoffice是一个stationary deterministic policy，所以我们能够总结下述:

wpsoffice

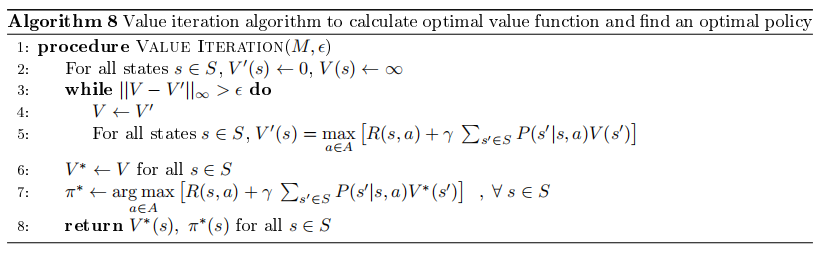
因为/var/folders/4_/wsyh4np55ms9lgn8jhhpylgr0000gp/T/com.kingsoft.wpsoffice.mac/wpsoffice.tG1010wpsoffice是B\*的一个固定点,我们有:/var/folders/4_/wsyh4np55ms9lgn8jhhpylgr0000gp/T/com.kingsoft.wpsoffice.mac/wpsoffice.WN1010wpsoffice，所以使用公式(32)中定义的B\*，同时公式(47),我们可以做如下变换:



for all ,定理的第一部分证明已经完成。

接下来证明wpsoffice是一个最优策略，我们已经证明了如果存在一个最优策略，则其值函数必为算子B\*的一个固定点。所以假设最优策略存在，通过定理3.8我们可以认为该策略是一个stationary deterministic policy，定义其为μ且值函数为。现在进行反证(Now for the sake of contradiction),假设不是的一个固定点，所以存在使得，结合定理3.8可知：。而引理3.8.1告诉我们存在一个严格比μ好的policy ，而这前后矛盾。如此便证明了必须为的一个固定点。结合之前证明的/var/folders/4_/wsyh4np55ms9lgn8jhhpylgr0000gp/T/com.kingsoft.wpsoffice.mac/wpsoffice.tG1010wpsoffice为最优值函数和现在证明的wpsoffice是一个最优策略，定理得证！

定理3.10给出一个直接的方式去计算最优值函数/var/folders/4_/wsyh4np55ms9lgn8jhhpylgr0000gp/T/com.kingsoft.wpsoffice.mac/wpsoffice.tG1010wpsoffice和最优策略wpsoffice。其思想是应用引理3.5.1,通过固定点的迭代计算去寻找的固定点(其实也是寻找经过投影矩阵的投影后仍为其自身的点)。一旦我们有了/var/folders/4_/wsyh4np55ms9lgn8jhhpylgr0000gp/T/com.kingsoft.wpsoffice.mac/wpsoffice.tG1010wpsoffice，我们就能够通过公式(47)提取最优策略wpsoffice，其伪代码如algorithm 8所示，algorithm以无限时间域MDP=(S,A,P,R,γ)和公差wpsoffice为输入，最终返回最优值函数和最优策略。



如果γ<1或者终止状态的概率为1则值迭代一定会收敛。

如果把公差wpsoffice设为0，我们能够精确的获得最优值函数和最优策略。然而在实际中wpsoffice通常被设置为

3.7 MDP control for a finite horizon MDP

现在简要讨论有限时间域的MDP contnrol问题。 我们之前已经在无限时间域MDPs中讨论过control problem，**我们简要的说明一下：在有限时间域的情况下可以确定最优的确定性策略。但策略不再是stationary,并且在每个时间t策略都是不同的**。证明不会太困难，读者可以从下面的连续中获得该证明。

**练习3.22**.假设一个有限时间域H以及有限回报的MDP,其episode具有如下形式:。假设MDP的策略写为:，证明下面的命题：

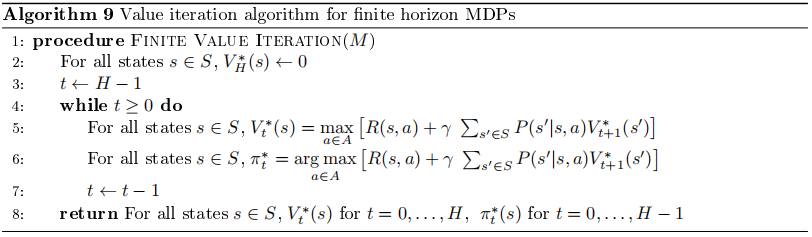
(a)求给定的MDP 的deterministic policies数量

(b)假设最优策略wpsoffice存在，其衍生出的关于最优值函数循环关系：，其中。准确的说，衍生一个在和之间的关系。

(c)假设是一组deterministic policies,i.e. 对于每一个,是一个在t时刻deterministic policy，t=0,...,H-1。证明对于每一个策略，无论是deterministic或者stochastic,均存在一个不差的策略，。

(d)证明存在一个最优策略。

因为练习3.22的结论，就像在无限时间域的情况下一样，我们可以将对最优策略的搜索限制在确定性策略的集合中。为此，我们提出了一种算法，即value iteration，它类似于无限视界情况下的对应算法。



算法正确性的证明，留给读者在接下来的练习中完成。

**练习3.23.**

(a)证明算法9的正确性(提示:使用练习3.22的结果)

(b).下一个练习也不难证明，它建立了在有限和无限时间域情况下的value iteration之间的对应关系。

**练习3.24.**

假设一个MDP,M=(S,A,P,R,γ),γ<1,令是M的最优值函数。定义一个有限时间域MDPs ,其时间域为的序列，使得，对于所有的k=1,2,...当运行输入为时，令为algorithm 9返回的最优值函数序列,对应于t=0.

(a)证明

3.8 Reference

