16 Reinforcement Learning - MDP

本节实验目标：在网格世界中实现和体会马尔科夫决策过程及其求解方案。

报告说明（重要）：本部分“马尔可夫决策和强化学习”包含三周内容，本节课为第一周内容，请留意该文档目录中标有“报告应实现任务[分值]”的部分，请在实现后及时保存文件和运行结果截图，在下两周课程报告中一并提交。

Introduction

在该堂实验课，你将**实现价值迭代(Value iteration)函数**，以及设计一些参数值来比较最佳策略(Policies)。 你将在 Gridworld 上测试你的agent。

实验代码包括一个autograder，以下命令对所有问题运行以测试你的实现：

```
python autograder.py
```

你也可以通过以下形式的命令做特定case的测试：

```
python autograder.py -t test_cases/q2/1-bridge-grid
```

实验代码

你应该要编辑的文件：	
valueIterationAgents.py	**Q1所需：用于解决已知 MDP 的价值迭代agent**
analysis.py	**Q2所需**
你应该看但不要编辑的文件：	
mdp.py	Defines methods on general MDPs. 里面的methods可以调用。
learningAgents.py	Defines the base classes `ValueEstimationAgent` and `QLearningAgent`, which your agents will extend.
util.py	Utilities, including `util.Counter`, which is particularly useful for Q-learners.
gridworld.py	The Gridworld implementation.
featureExtractors.py	Classes for extracting features on (state, action) pairs. Used for the approximate Q-learning agent (in `qlearningAgents.py`).
你可以忽略的文件：	
environment.py	Abstract class for general reinforcement learning environments. Used by `gridworld.py`.
graphicsGridworldDisplay.py	Gridworld graphical display.
graphicsUtils.py	Graphics utilities.
textGridworldDisplay.py	Plug-in for the Gridworld text interface.
crawler.py	The crawler code and test harness. You will run this but not edit it.
graphicsCrawlerDisplay.py	GUI for the crawler robot.
autograder.py	Project autograder
testParser.py	Parses autograder test and solution files
testClasses.py	General autograding test classes
test_cases/	Directory containing the test cases for each question

你应该要编辑的文件:	
reinforcementTestClasses.py	Specific autograding test classes

注意事项

- 请***不要***更改代码中提供的任何函数或类的名称，否则会对 autograder 造成严重破坏。
- **正确使用数据集**：你在此项目中的部分分数取决于你训练的模型在自动评分器的测试集上的表现。我们不提供任何 API 供你直接访问测试集。
- 如果你发现自己卡在某事上，请联系老师、助教或同学寻求帮助。我们希望这些项目是有益的和指导性的，而不是令人沮丧和沮丧的。

MDPs in Grid World

首先，试着用键盘上下左右箭头在手动控制模式下运行 Gridworld：

```
python gridworld.py -m
```

你将看到课堂上所讲Grid World的两个出口布局。 蓝点是agent。 请注意，**当你按下 *up* 时，agent实际上只有 80% 的时间向北移动。 这就是这个世界中agent的生活，总有不如意的时候！**

你可以控制模拟器的许多方面。 运行以下命令可获得完整的选项列表：

```
python gridworld.py -h
```

默认agent随机移动

```
python gridworld.py -g MazeGrid
```

你应该看到随机agent在网格上来回弹跳，直到它到达了exit的网格（右上角）时。对于 AI agent来说，这不是最好的情况。

- 注意：Gridworld MDP 是这样的，你首先必须进入预终止状态（**GUI 中显示的右上角双框），然后在一段经历（episode）实际结束之前采取特殊的"exit"操作（真正的终止状态称为 `TERMINAL_STATE` ，未在 GUI 中显示）**。如果你手动运行一次 episode，你的总回报可能会低于你的预期，这是由于**折扣因子（可通过 `-d` 更改；默认为 0.9）**的存在。
- 查看跟图形界面同时输出的**控制台(Terminal)输出，你可以看到agent经历的每次转换**（要关闭此功能，请在上述命令中使用 `-q` ）。
- 与 Pacman 中一样，agent 位置由 `(x,y)` 笛卡尔坐标表示，数组由 `[x][y]` 索引， `'north'` 动作增加 `y` ，等等。默认情况下，大多数动作将获得零回报。**你可以使用实时奖励选项 (`-r`) 更改回报值。**

Question 1: Value Iteration 价值迭代 报告应实现任务[2分]

- **说明：该小题截图只需截 python autograder.py -q q1 的运行结果，获得评分器给的5分中，这道题才算得它占的2分。**

回想一下价值迭代状态更新方程：

$$V_{k+1}(s) \leftarrow \max_{a} \sum_{s'} T(s,a,s') \left[R(s,a,s') + \gamma V_k(s') \right]$$

任务：在 `ValueIterationAgent` 中编写一个价值迭代agent，在 `valueIterationAgents.py` 中已为你部分实现。你的价值迭代 agent是离线规划器，而不是强化学习agent，因此相关的训练选项是它在初始规划阶段运行的价值迭代的**迭代次数**（命令行选项 `-i` ）。 `ValueIterationAgent` 在构造函数返回之前输入 MDP 并运行指定迭代次数的价值迭代。

记价值迭代计算最优状态值的 k 步估计为 V_k。**除了 `runValueIteration` 函数，基于你计算的 V_k 可为 `ValueIterationAgent` 实现以下函数。**

- `computeActionFromValues(state)` ：基于 `self.values` 给出的价值函数计算最佳行动。
- `computeQValueFromValues(state, action)` ：基于 `self.values` 给出的价值函数计算(state, action)对的Q值。

这些数值都将显示在 GUI 中，其中状态值(values)是正方形的数字，Q 值是正方形四分之一的数字，策略是每个正方形的箭头。

提示和注意事项：

- **重要：**使用“batch”版本的价值迭代（**即我们理论课上讲的那种价值迭代版本**），其中每个向量 V_k 从固定向量V_{k-1}计算，而不是一个单一权重向量就地更新的“online”版本。**这意味着当一个状态的值在迭代 k 中基于其后继状态的值进行更新时，用于值更新计算的后继状态值应该是来自迭代 $k-1$ 的那些**（即使某些后继状态已经已在迭代 k 中更新）。 Sutton & Barto 第 91 页第 4.1 章讨论了这两种版本的差异。
- 从深度 k 的状态值（反映下一个 k 的回报）提取的策略实际上会反映下一个 $k+1$ 的回报（即你返回 π_{k+1}）。同样，Q 值也将反映比状态价值多一个迭代深度的回报（即你返回 Q_{k+1}）。
- 你应该**返回**由状态价值中提取的策略 π_{k+1}。
- 你可以选择使用 `util.py` 中的 `util.Counter` 类，这是一个默认值为零的字典。但是，请小心使用 `argMax`：你想要的 argmax 可能不是Counter中的key!
- 注意：确保你的代码能处理在 MDP 中**某状态下agent可能没有可用行动的情况**（想想这对未来的回报意味着什么）。

要在所有测试用例上测试你的实现，请运行autograder：

```
python autograder.py -q q1
```

以下说明可能**对你的调试过程有所帮助**：

1. 以下命令加载你的 `ValueIterationAgent`，它将计算策略并执行该策略 10 次。 按任意一个键循环显示值、Q 值和模拟器。你应该会发现开始状态的值（`V(start)`，你可以从 GUI 中读取）和经验所得的平均奖励（在 10 轮执行完成后打印）非常接近(命令终端中会显示出来)。

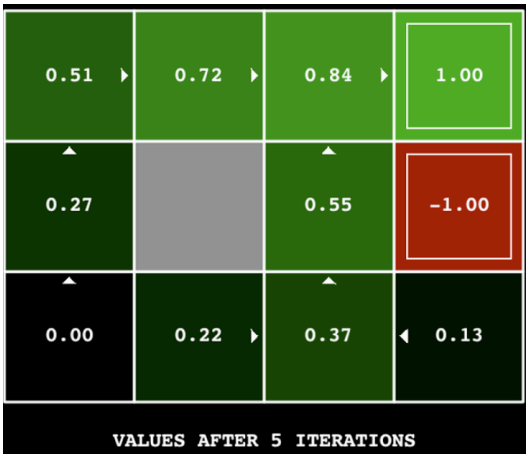
```
python gridworld.py -a value -i 100 -k 10
```

2. 可用以下命令查看 `gridworld.py` 的命令参数，如 -v，-s等， 对调试有用：

```
python gridworld.py -h
```

3. 在默认的 BookGrid 上，运行 5 次迭代的价值迭代应该会为你提供以下输出：（正如课堂所示）

```
python gridworld.py -a value -i 5
```

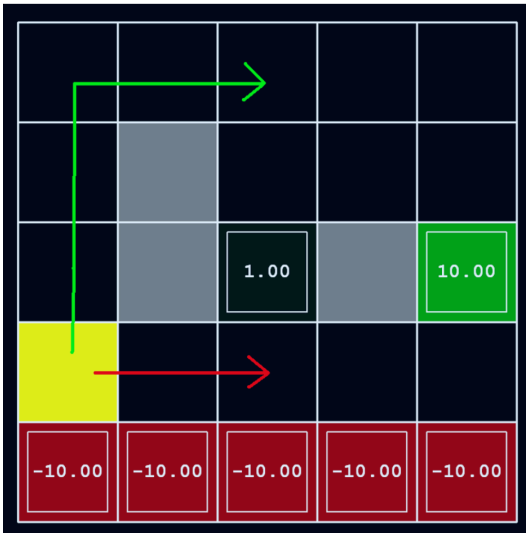


Question 2: Policies 报告应实现任务[1分]

- **说明：该小题截图只需截 python autograder.py -q q2 的运行结果，获得评分器给的5分中，这道题才算得它占的1分。**

考虑如下图所示的 `DiscountGrid` 布局。这个网格有两个具有正回报(+1, +10)的终止状态（在中间行），一个回报 +1 的近距离出口(exit)和一个回报 +10 的远距离出口(exit)。网格的底行由具有负回报的终止状态组成（以红色显示）；这个“悬崖”区域中的

每个状态都有回报-10。起始状态是黄色方块。我们区分了两种类型的路径：(1) 在网格底行附近行进的“冒险悬崖”（“risk the cliff”）路径；这些路径较短，但有可能获得较大的负回报，如下图中的红色箭头所示。 (2) 沿着网格顶部边缘行进的“避开悬崖”（“avoid the cliff”）路径。这些路径更长，但不太可能产生巨大的负面回报，如下图中的绿色箭头所示。



在这个问题中，你将在`analysis.py`为此 MDP 设计折扣（discount）、噪声（noise，即 1 - 采取行动后确实能到达目标下一状态的概率）和生存回报（living reward）参数，以生成几种不同类型的最优策略。 你**为每个部分设置的参数值应具有以下属性：如果你的agent遵循其在 MDP 中的最佳策略，它将表现出给定的行为。** 如果任何参数设置都未实现特定行为，通过返回字符串 `'NOT POSSIBLE'` 断言该策略是不可能的。

以下是你应该尝试生成的最佳策略类型：

a. 更喜欢近距离exit口（+1），**risking** the cliff（-10）

b. 更喜欢近距离exit口（+1），but **avoiding** the cliff（-10）

c. 更喜欢远距离exit口（+10），**risking** the cliff（-10）

d. 更喜欢远距离exit口（+10），**avoiding** the cliff（-10）

e. Avoid both exits and the cliff（so an episode (经历) should **never terminate**）

要测试你的设计，请运行autograder：

```
python autograder.py -q q2
```

`question2a()` - `question2e()` 应该在`analysis.py`中返回一个 包含三个元素（折扣因子、噪音、生存回报）的元组。

提示：你可以用类似以下的命令去进行参数尝试，其中 -a value 指当前agent是价值迭代agent， -i 100指迭代100次，-s 0.1 代表一个较慢动作的展示动画(0<s≤1)，-g DiscountGrid 指当前环境用到的Grid，**最后三个参数是这个问题中要调试的，-d 0.9 指折扣因子为0.9，-n 0.1指噪声为0.1，-r 0.6指生存奖励为0.6。**

```
python gridworld.py -a value -i 100 -s 0.1 -g DiscountGrid -d 0.9 -n 0.1 -r 0.6
```

注意：你可以在 GUI 中查看你的策略。例如，使用 2(a) 的正确答案，(0,1) 中的箭头应指向东方，(1,1) 中的箭头也应指向东方，(2,1) 中的箭头应指向北方.

注意： 在某些机器上，你可能看不到箭头。在这种情况下，按下键盘上的任一按钮切换到 qValue 显示，并通过获取每个状态的可用 qValues 的 arg max 来计算策略。

本节课实现内容保存说明 - 为提交第七次上机报告做准备

- **第18周上机课结束后，一并提交本部分”马尔可夫决策-强化学习“的报告，该报告包含本节课的Q1,Q2的代码和运行截图，请按Q1,Q2大标题下的说明进行截图，保存。**

- 本节课实现的代码文件也请保存，你将需要把其中实现的代码加到第18周相应的代码文件中跟第18周的任务实现一并进行提交。