作业报告——Car-Rental

方鸿宇 2001213098

问题描述

本实验实现了对Car-Rental问题中策略表和价值表的学习，问题具体描述如下：

Jack manages two locations for a nationwide car rental company. Each day, some number of customers arrive at each location to rent cars. If Jack has a car available, he rents it out and is credited $10 by the national company. If he is out of cars at that location, then the business is lost. Cars become available for renting the day after they are returned. To help ensure that cars are available where they are needed, Jack can move them between the two locations overnight, at a cost of $2 per car moved. We assume that the number of cars requested and returned at each location are Poisson random variables, meaning that the probability that the number is n is $\frac{\lambda^n}{n!}e^{-\lambda}$, where is the expected number. Suppose is 3 and 4 for rental requests at n! the first and second locations and 3 and 2 for returns. To simplify the problem slightly, we assume that there can be no more than 20 cars at each location (any additional cars are returned to the nationwide company, and thus disappear from the problem) and a maximum of five cars can be moved from one location to the other in one night. We take the discount rate to be = 0.9 and formulate this as a continuing finite MDP, where the time steps are days, the state is the number of cars at each location at the end of the day, and the actions are the net numbers of cars moved between the two locations overnight. Figure 4.2 shows the sequence of policies found by policy iteration starting from the policy that never moves any cars.

算法说明

本实验使用策略迭代法实现对策略表和价值表的学习，该方法通过迭代进行策略评估和策略改善从而习得关于目标问题的价值表和策略表。策略评估部分对于当前策略，依据贝尔曼方程对价值表进行迭代更新直至收敛；策略改善部分对于当前价值表，使用贪心法对策略表进行更新。算法的伪代码如下：

Policy Iteration (using iterative policy evaluation) for estimating $\pi \approx \pi_*$

1. Initialization
   $V(s) \in \mathbb{R}$ and $\pi(s) \in \mathcal{A}(s)$ arbitrarily for all $s \in \mathcal{S}$

2. Policy Evaluation
   Loop:
   $\Delta \leftarrow 0$
   Loop for each $s \in \mathcal{S}$:
   $v \leftarrow V(s)$
   $V(s) \leftarrow \sum_{s',r} p(s', r|s, \pi(s))\big[r + \gamma V(s')\big]$
   $\Delta \leftarrow \max(\Delta, |v - V(s)|)$
   until $\Delta < \theta$ (a small positive number determining the accuracy of estimation)

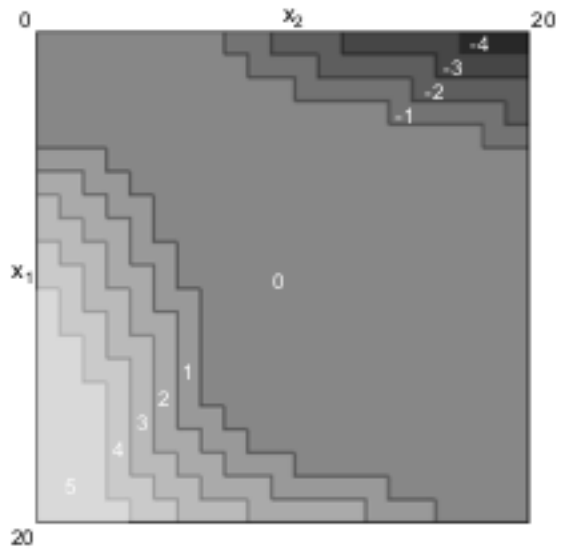
3. Policy Improvement
   policy-stable $\leftarrow$ *true*
   For each $s \in \mathcal{S}$:
   old-action $\leftarrow \pi(s)$
   $\pi(s) \leftarrow \arg\max_a \sum_{s',r} p(s', r|s, a)\big[r + \gamma V(s')\big]$
   If *old-action* $\neq \pi(s)$, then *policy-stable* $\leftarrow$ *false*
   If *policy-stable*, then stop and return $V \approx v_*$ and $\pi \approx \pi_*$; else go to 2

实验设置

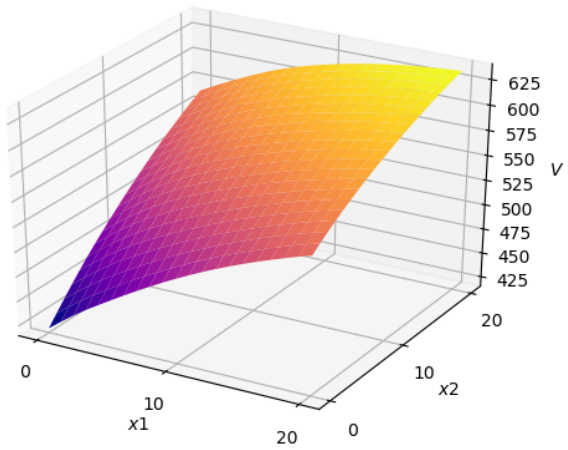
实验中设置折扣因子γ为0.9，策略评估部分的误差阈值为1。

实验结果

实验中共进行了五轮"策略评估-策略改善"迭代，获得如下图所展示的策略表和价值表，具体数值可用`show_result.py`展示（见代码说明）。



策略表



价值表

代码说明

实验代码包括`car_rental.py`和`show_result.py`两部分，使用python3运行，需安装numpy、opencv和matplotlib库。

`car_rental.py`进行模型的训练，并将每轮迭代的策略表和价值表保存在`log`目录中，保存格式分别为`pi-x.npy`和`v-x.npy`，其中x表示迭代次数。

`show_result.py`读取并在命令行中输出保存的策略表和价值表，并使用opencv和matplotlib对两个表进行可视化，可视化结果保存在`log`目录中，保存格式分别为`pi.png`和`v.png`。