强化学习实验报告

王松宸
学号：2024201594

1 TODO 1

1.1 代码思路

```python
def compute_return(start_index, chain, gamma):
    G = 0
    for i in reversed(range(start_index, len(chain))):
        # TODO ~1: 实现回报函数
        G += rewards[chain[i]-1] * (gamma**(i-start_index))
    return G
```

将所有从 start_index 开始到链条末尾的奖励值按照折扣因子进行加权求和，得到从 start_index 开始的回报值 G。

1.2 运行结果截图

Figure 1: 运行结果截图

1.3 实验结果分析

该 chain 的回报值计算无误，符合预期。但它明显不是最优的 chain，因为包含 s_4 的 chain 回报值更高。

2 TODO 2

2.1 原始数据运行结果截图

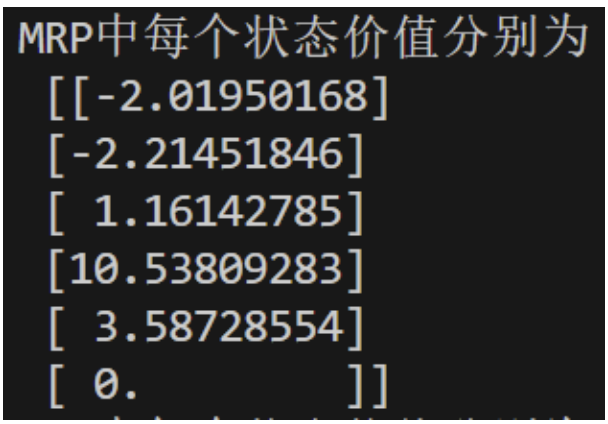
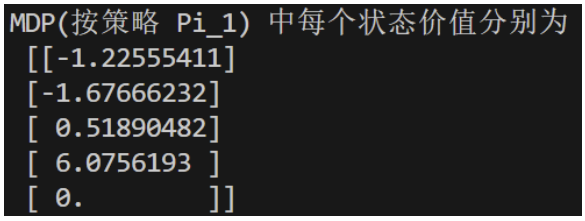
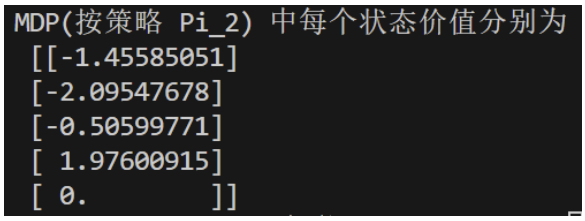


Figure 2: 原始数据运行结果截图（MRP）



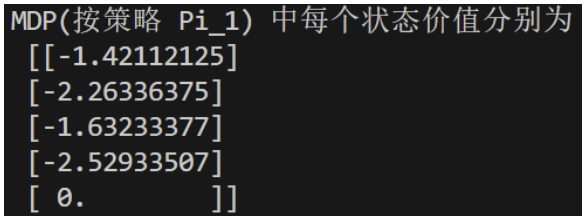
(a) original_MDP_result1



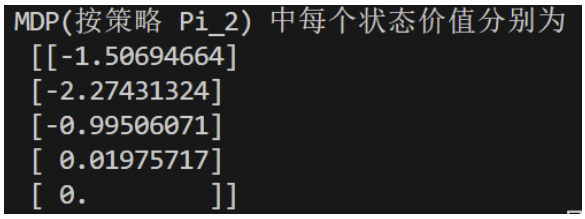
(b) original_MDP_result2

Figure 3: 原始数据运行结果截图（MDP）

2.2 修改奖励函数



(a) reward1



(b) reward2

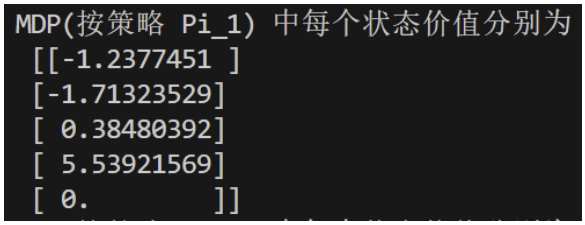
Figure 4: 修改奖励函数后的运行结果截图（MDP）

将奖励函数中”s4-前往 s5” 的奖励值从 10 修改为-5，明显看出 s4 的状态价值降低，从而使得其他路径更具吸引力。

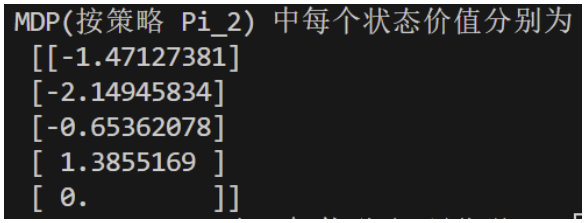
此外，距离 s4 状态较近的状态（如 s3）也受到了影响，其价值同样降低，而距离 s4 状态较远的状态（如 s2、s1）则几乎没有变化。

同时，受策略影响，更“主观”的 π_2 相比于纯随机的 π_1，其状态价值变化更小，反映出自主设置的相应策略产生的结果对参数变化具备一定的鲁棒性。

2.3 修改状态转移函数



(a) transition1_1



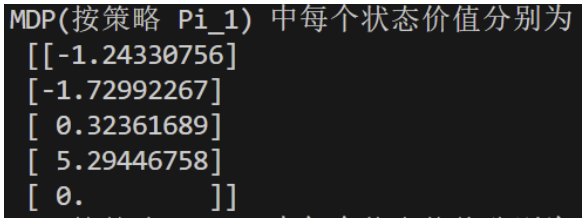
(b) transition1_2

Figure 5: 第一次修改状态转移函数后的运行结果截图（MDP）

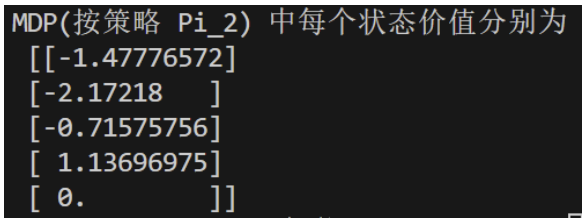
将状态转移函数中”s4-概率前往” 的概率从 (0.2, 0.4, 0.4) 修改为 (0.6, 0.2, 0.2)，使得从 s4 状态更容易回到 s2 状态。

这种处理使得整个决策过程多了很多“重新开始”的机会，因为 s2 状态位于较前的位置。这样的设置可能使状态价值在实际观测结果上更能反映大多数情况的真实价值，因为引起从 s4 状态到 s5 状态带来的巨额奖励的概率降低，随机性减弱。

从结果截图中可以看出两种策略下，状态价值的变化均不明显，因此进一步提升从 s4 状态回到 s2 状态的概率观察是否能验证我的想法。



(a) transition2_1



(b) transition2_2

Figure 6: 第二次修改状态转移函数后的运行结果截图（MDP）

将状态转移函数中”s4-概率前往” 的概率从 (0.6, 0.2, 0.2) 修改为 (0.8, 0.1, 0.1)，进一步提升从 s4 状态回到 s2 状态的概率。

这种处理使得从 s4 状态回到 s2 状态的概率进一步增加，从而使得 s4 的状态价值进一步降低，其他状态的价值则保持平稳，说明增长路径（更易回到 s2）对产生大额奖励的动作影响较大，通过作用于状态价值上体现出来。

3 个人思考

基于概率的奖励模式设计，更能反映现实中的不确定因素以及模拟一些实际的情况，但这同样也会造成计算结果的波动性增大，从而影响对策略效果的评估。因此，在设计奖励函数时，需要综合考虑任务的实际需求与环境的随机性，选择合适的奖励结构以平衡探索与利用，确保学习过程的稳定性与有效性。同时，在设计状态转移函数时，应考虑状态之间的关联性及其对整体策略的影响，避免过度依赖某一状态或路径，从而提升模型的泛化能力和适应性。