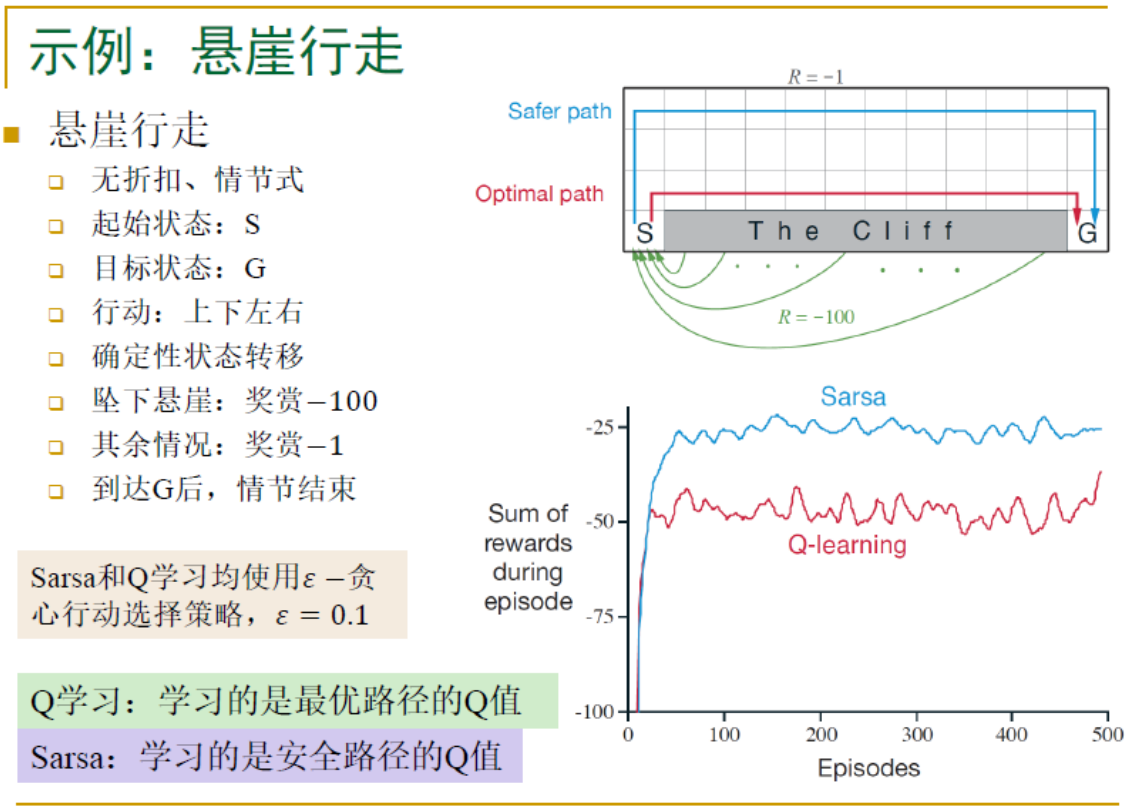
智能系统作业

人工智能学院　丁豪
181220010@smail.nju.edu.cn

问题描述



问题为悬崖行走问题，奖赏和状态转移情况如上图所示。为了便于统一管理此环境，将环境相关部分全部放在cliff_environment.py中，主要是以下函数，他接受当前状态和行动作为参数，返回对应奖赏以及下一状态，其中返回特殊状态[-1,-1]表示情节结束，将在后续判断中使用。

```
# cliff_environment.py
def RS_next(state,a):
    return reward,next_State
```

算法描述

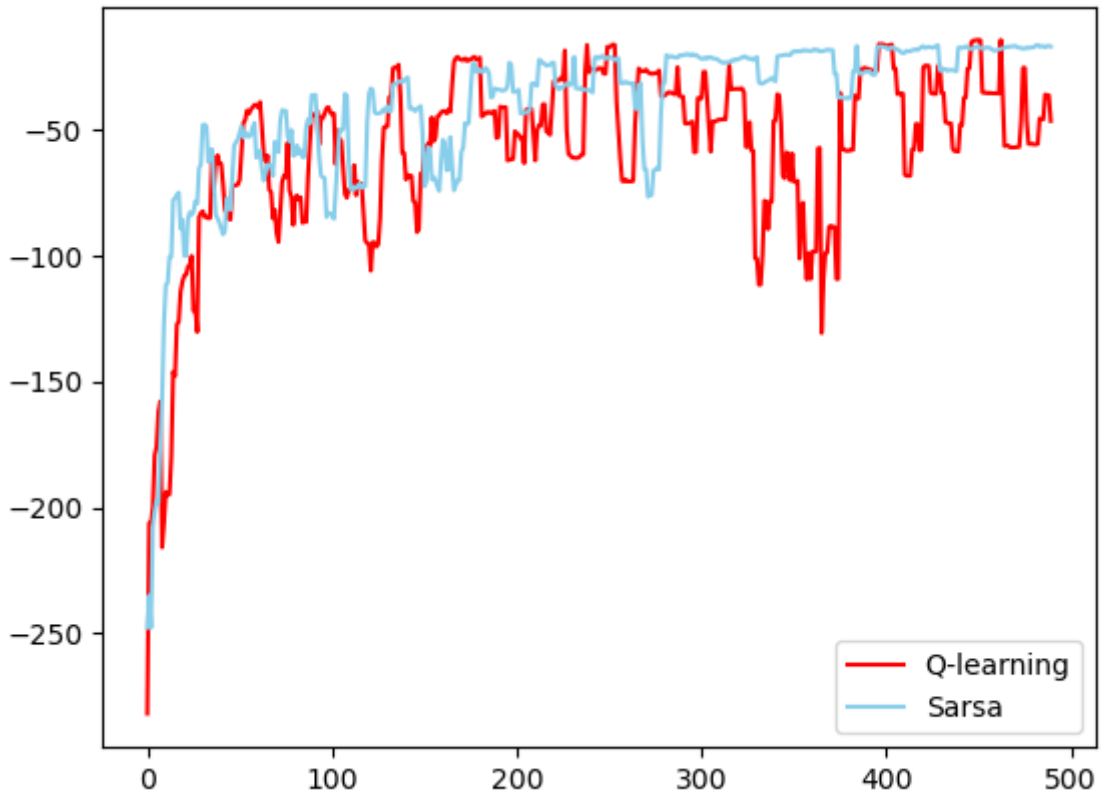
分两个文件：Q-learning VS Sarsa.py 和 nSarsa VS SarsaLambda.py，分别用python实现了这四个算法。其中公共的部分有参数初始化和$\epsilon-$贪心，因此把它定义在cliff_environment.py中，增加代码复用程度。在回报评价方面，参照QQ群中老师的要求改为图上每一个点代表第i次到第i+9次无折扣回报之和的平均值，这里使用了队列来实现。

```
## cliff_environment.py

# 返回全0的Q
def initialize_Q()
```

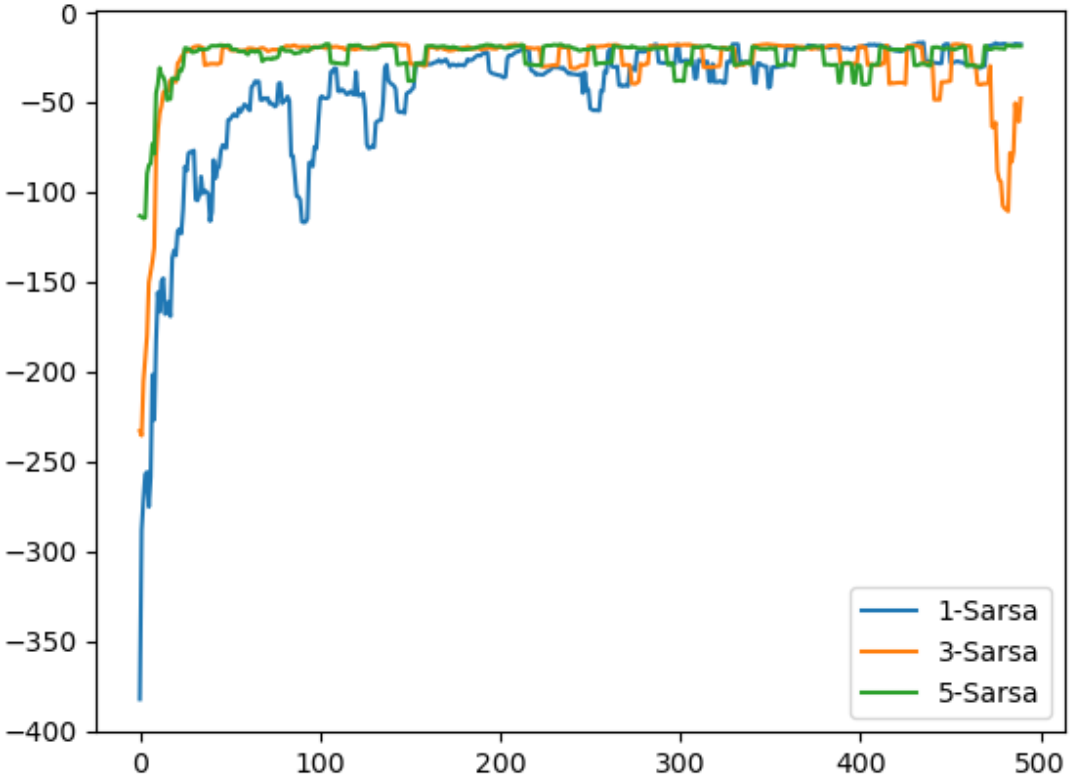
```
# 返回True/False表示是否在这一步贪心
def epsilon_greedy(epsilon)

# 返回由epsilon_greedy规则选择的行动A
def select_A(S,Q,epsilon)


## Q-learning VS Sarsa.py & nSarsa VS SarsaLambda.py

# 参数定义
epsilon = 0.1
alpha = 0.1
gamma = 1
num_episode = 500

# 返回无折扣回报平均值列表
def Q_learning()
def Sarsa()
def n_Sarsa(n)
def Sarsa_lambda(lam)
```

实验

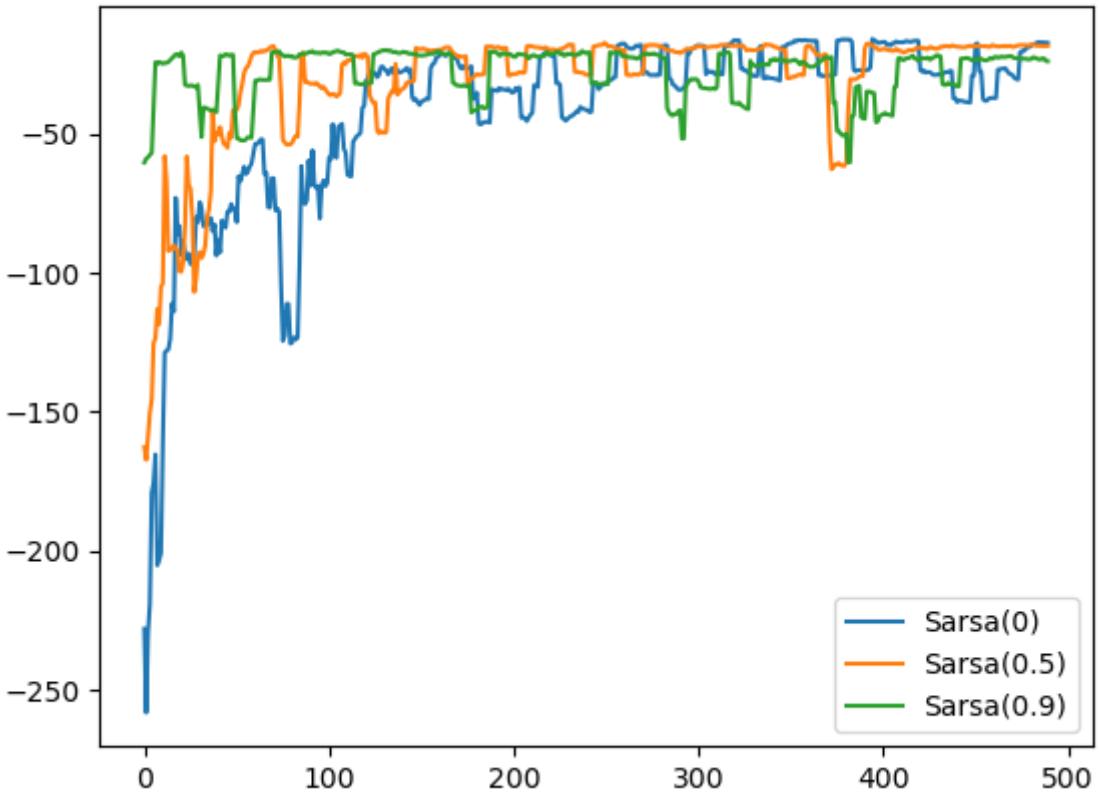
第一个实验是基础Sarsa与Q学习的性能对比，参数选择上ϵ按照题目的要求设定为0.1，因为是无折扣情况所以γ设为1，学习率α题目并没有给出要求，这里采用0.1。可以看出Sarsa与Q学习收敛速度基本相当，但是收敛情况来看Sarsa更佳稳定且平衡点收益大于Q学习。我认为出现这种情况的原因，可能是由于有$\epsilon - greedy$的引入使得行动选择具有了相当的不确定性，在这种情况下Sarsa使用同策略真实状态作为评估将相比Q-学习的使用最大行动值评估更为符合真实情况。有些遗憾的是并没有复现与PPT上完全相同的实验图像，推测和学习率等未指定参数的选择有一定关系。



第二个实验是n-Sarsa在n=1，3，5情况下的收益情况。可以发现随着n值的增加，收益收敛的速度呈现出显著的正相关关系。其中初始点的含义是前10个情节的平均收益，三者在初始点处就有较大差异，体现出在前10次情节中收敛情况随着n的上升有着显著的提升。从最终收敛结果来看三者基本上相同，使用n=5的效果最佳，使用n=3的在实验的最后几十次内出现了一波小的反弹，这可能是由于学习率设置较大，导致单次$\epsilon - greedy$选择到非最优状态产生的负反映过大导致的。



第三个实验研究的是使用累计迹的Sarsa(λ)算法，其中λ原本设置的是0，0.5，1，但是在使用1的时候无法收敛，经过老师同意，改为使用0.9代替。可以看出随着Lambda取值的提升，收敛速度有明显的正相关。而收敛的最终结果来看，三者无显著差异，都在-50以内小范围内波动。产生这样结果的原因是使用更大的lambda，将使得Z的累计值在同一时刻更大，这样对于Q的更新幅度也相应更大，于是产生了较快的收敛性，而收敛结果只在于同一状态各行动Q值的相对关系，因而无明显相关性。


−50
−100
−150
−200
−250
0
100
200
300
400
500
Sarsa(0)
Sarsa(0.5)
Sarsa(0.9)