时序差分算法实验报告

吴孟周　2100013053

一、实验要求

（1）阅读《基于时序差分的路径规划-实验指导书》，尝试运行并理解时序差分算法在冰湖路径规划问题上的示例代码。
（2）在示例代码的基础上，尝试实现 n-step TD 算法，比较并分析 n 的不同取值对策略收敛速度的影响（由于随机性建议多次实验）。
（3）在示例代码的基础上，尝试实现 TD(λ)算法，比较并分析λ的不同取值对策略收敛速度的影响（由于随机性建议多次实验）。

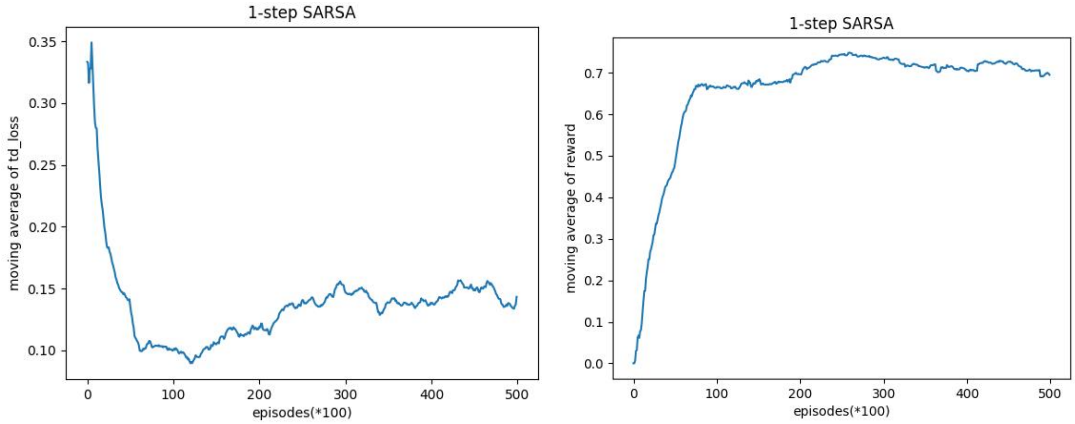
二、代码修改思路

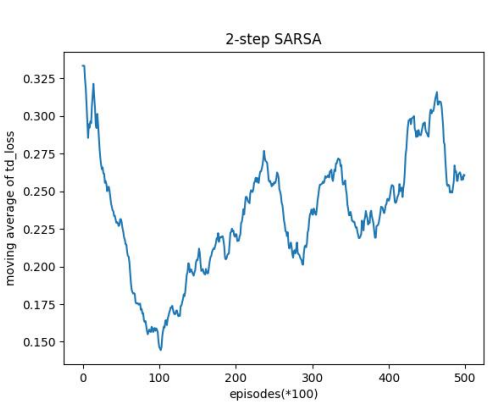
在 SARSA 的基础上实现了 n-step SARSA 算法。思路与 SARSA 类似，区别在于在与环境交互的过程中存储 states, actions, rewards 三个列表，在交互结束后通过列表计算 n-step 的 td_target 和 td_error，对 Q 数组进行更新。

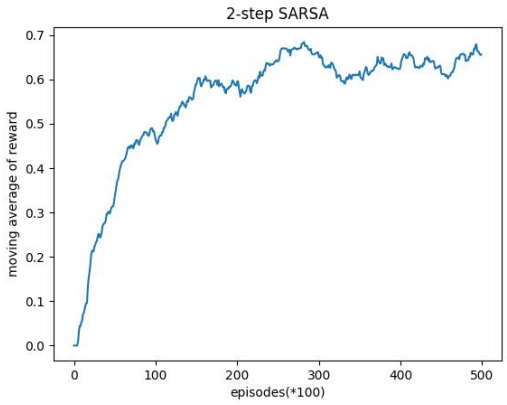
在 SARSA 的基础上实现了 SARSA(lambda)算法。思路与 SARSA 类似，区别在于加入了资格迹矩阵，这个矩阵的大小为 state_number*action_number，表示了在从前向后进行更新时，需要为前面走过的 Q[s, a] 加上的系数。

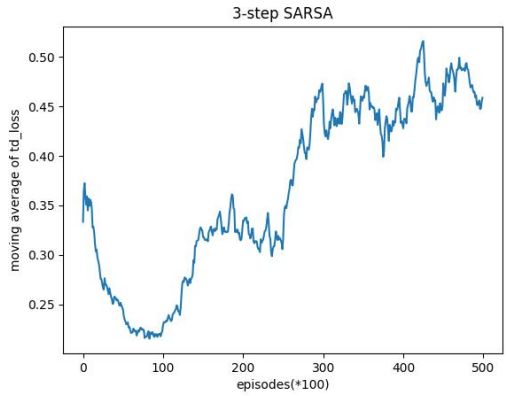
为了评估收敛程度，加入 Getloss 函数，这个函数以 Q, env.P, gamma 作为参数，枚举每个 s,a 对，计算 Q[s, a] 与 target 的差值的平方并加和。

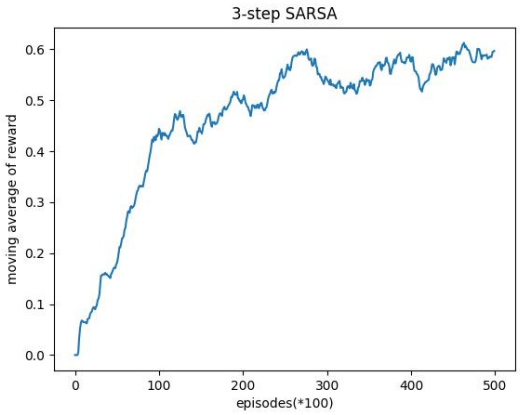
三、实验结果和分析

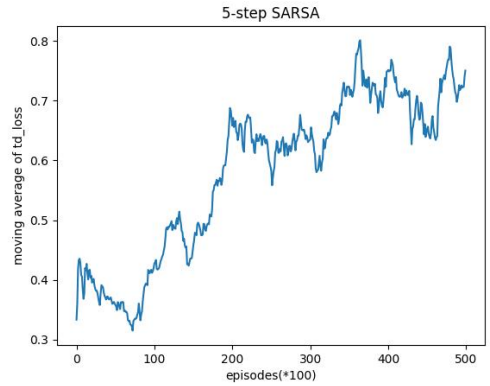


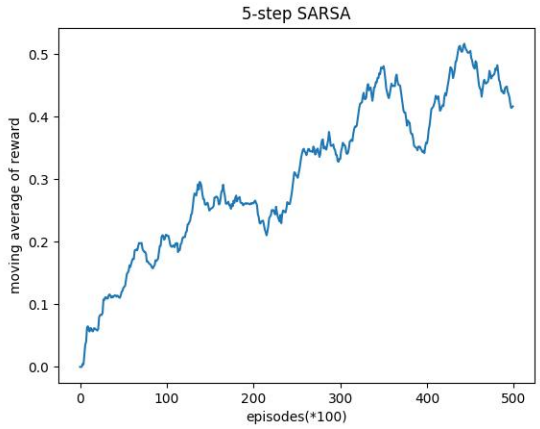






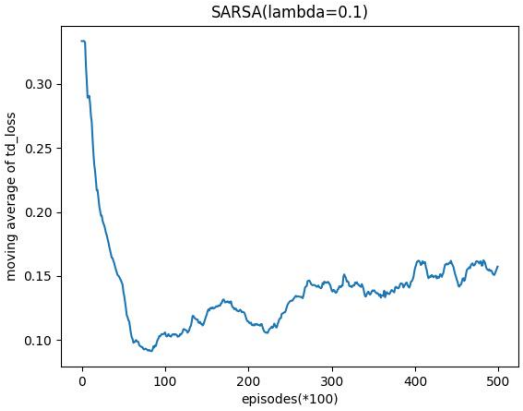


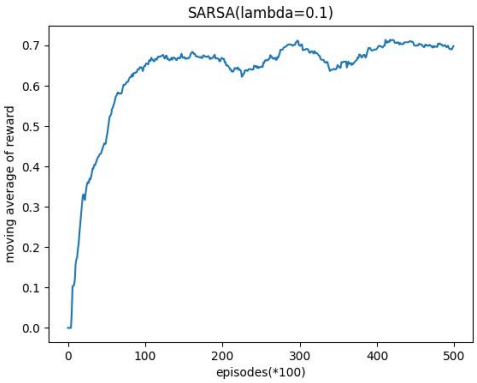


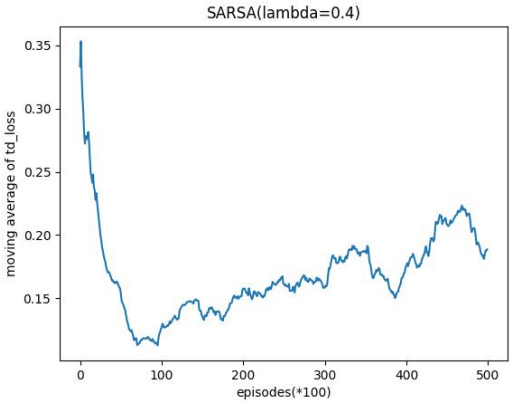


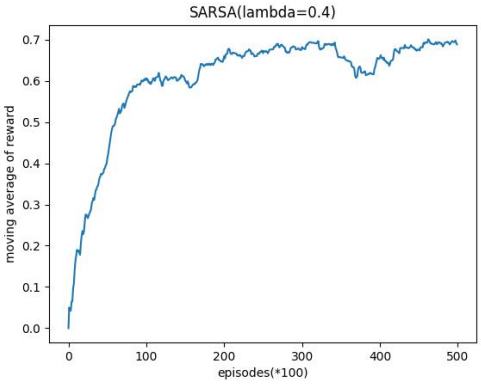
以上是对于 n-step SARSA 在 n=1,2,3,5 下的测试结果。首先可以发现的是收敛速度（收敛到相同 td_loss 所需的 episodes 数量）随着 n 的增大而降低，收敛到的策略效果也随着 n 的增大而变差。可能的原因是本次的实验环境具有马尔可夫性，且只在终点有大小为 1 的 reward。另一个可以发现的是 td_loss 在开始逐渐降低，但在约 10000 个 episodes 之后开始升高。可能的原因有两方面，一是从 reward 可以发现此时收敛到的策略并不是最优策略，在此之后开始了对更优策略的探索；二是 td_loss 计算的是所有 s,a 对的误差，随着 episodes 的增加 epsilon 不断降低，对一些不在较优策略中的状态和动作的探索越来越少，这些位置的 Q 值长期得不到更新。

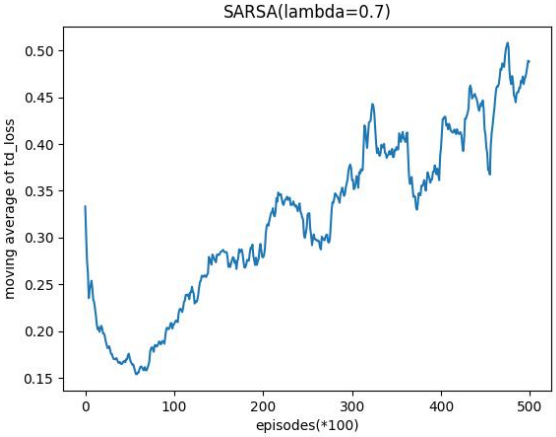
下面是对于 SARSA(lambda) 在 lambda=0.1, 0.4, 0.7, 1 下的测试结果。观察是与 n-step SARSA 类似的，背后的原因也是如此。lambda 参数越大就会越多的使用多个 step 的信息，这导致了收敛速度和收敛效果等不佳。

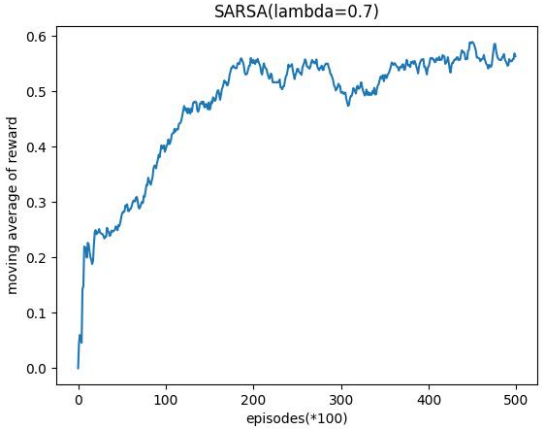

SARSA(lambda=0.1)
moving average of td_loss
episodes(*100)

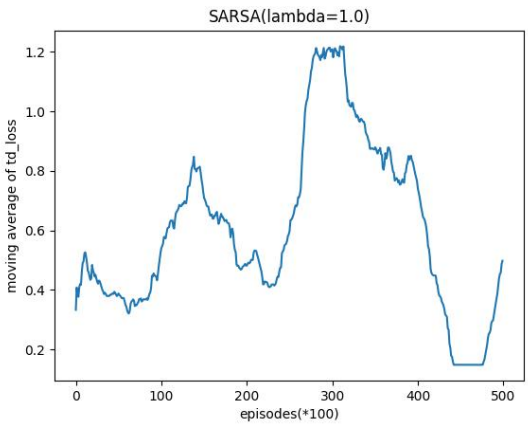

SARSA(lambda=0.1)
moving average of reward
episodes(*100)

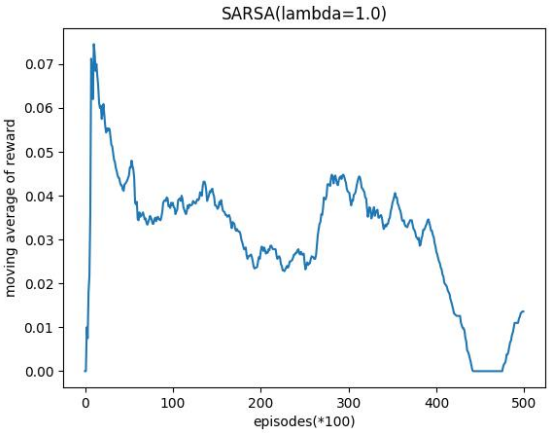

SARSA(lambda=0.4)
moving average of td_loss
episodes(*100)


SARSA(lambda=0.4)
moving average of reward
episodes(*100)


SARSA(lambda=0.7)
moving average of td_loss
episodes(*100)


SARSA(lambda=0.7)
moving average of reward
episodes(*100)


SARSA(lambda=1.0)
moving average of td_loss
episodes(*100)


SARSA(lambda=1.0)
moving average of reward
episodes(*100)