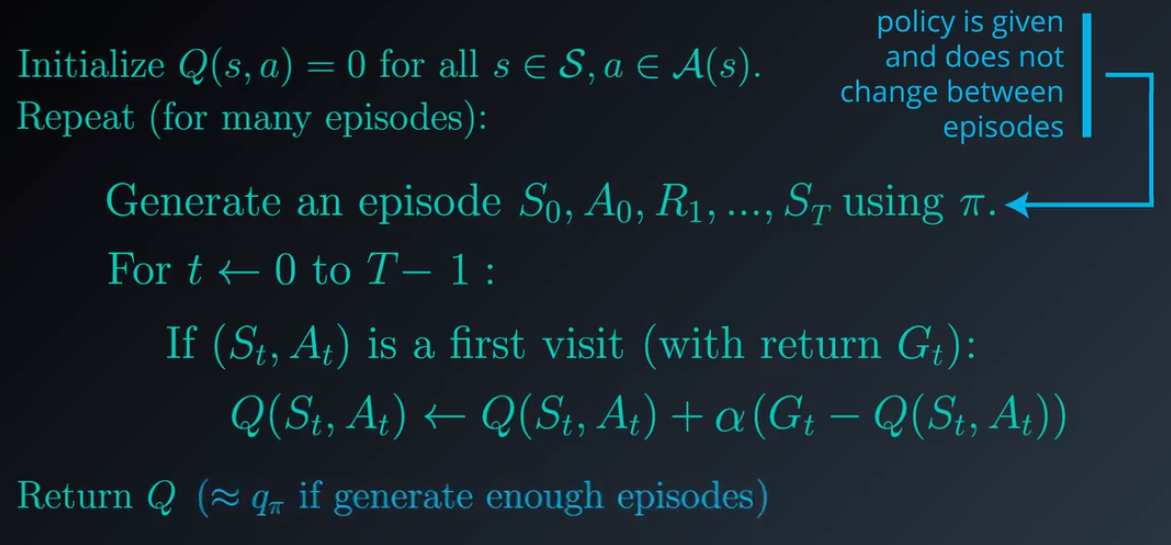
# TD预测-状态值

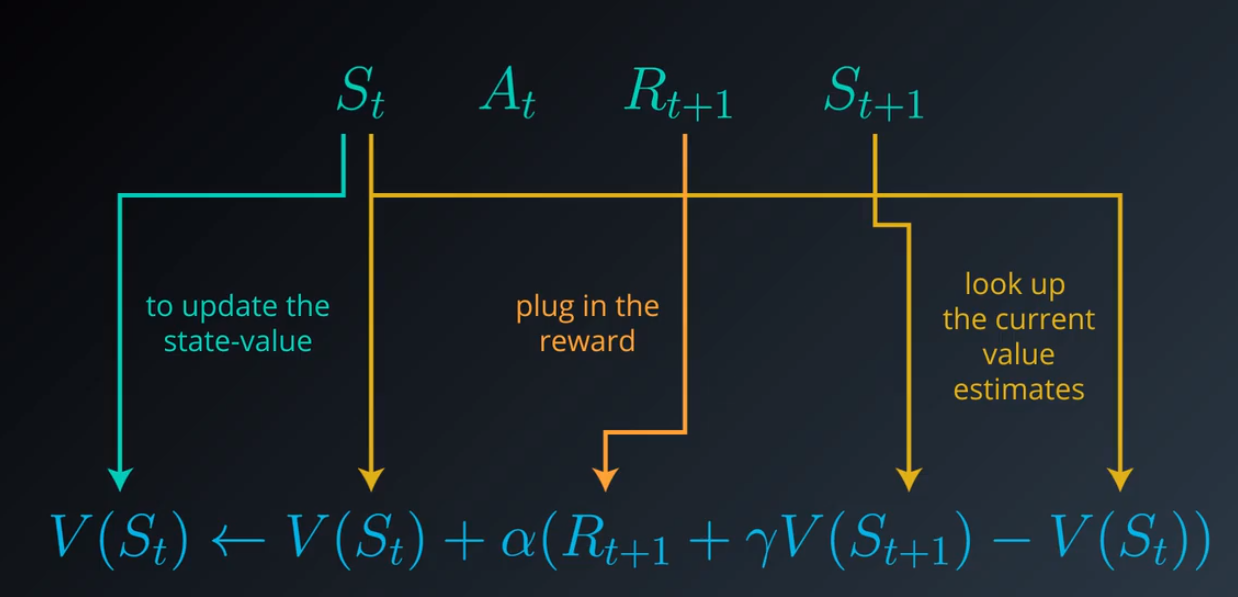
在蒙特卡洛方法中，智能体以阶段形式与环境互动，当一个阶段结束后，按顺序查看每个状态动作对

如果是首次经历，则计算相应的回报 并使用它来 更新动作值

当经历了很多个阶段后（需要注意的是 只要我们不在阶段之间更改策略）,该算法就可以解决预测问题，只要我们运行该算法足够长的时间，就肯定能够获得一个很完美的动作值函数估计结果

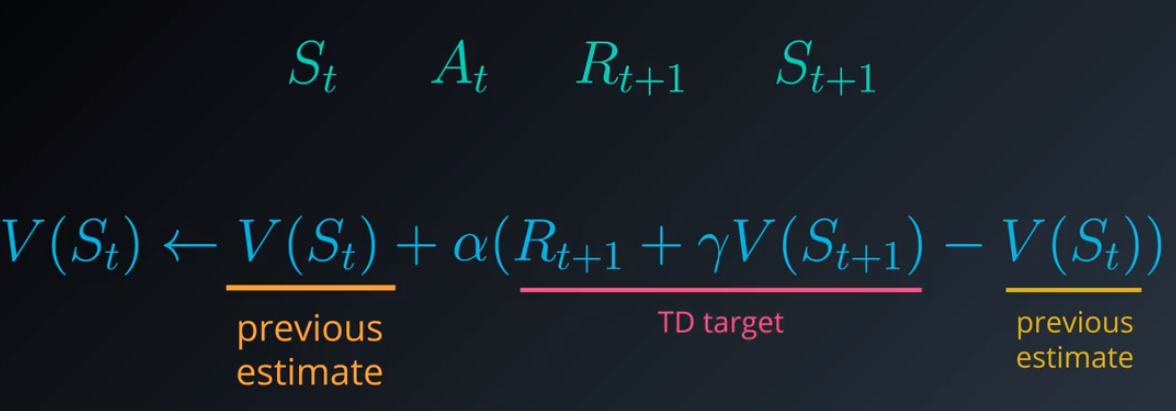






在智能体采取动作之前，当前状态的最佳值估算，包含在值函数中。

然后智能体采取动作并获得奖励和下个状态,将这个 后续状态的值 的估值称为 TD 目标

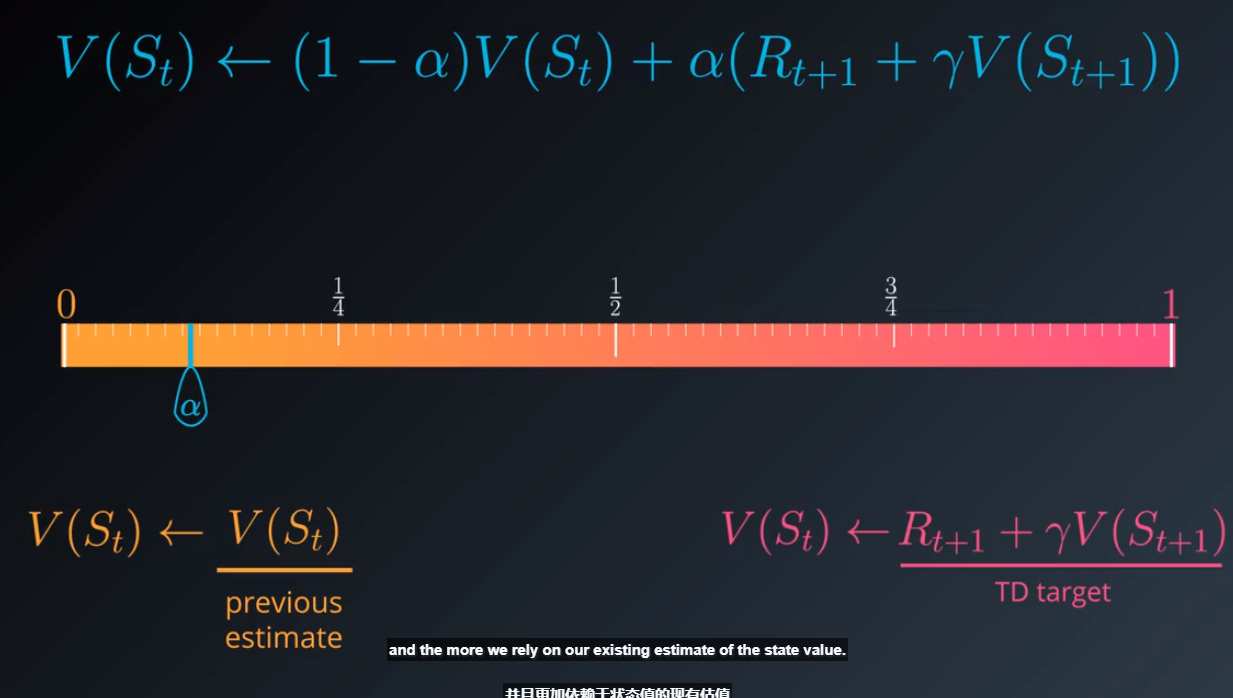


重写方程式

当 α 设为 1 时，新的估值是 TD 目标

我们完全忽略 并 替换之前的估值

如果将 α 设为 0，则完全忽略目标并保留旧的估值

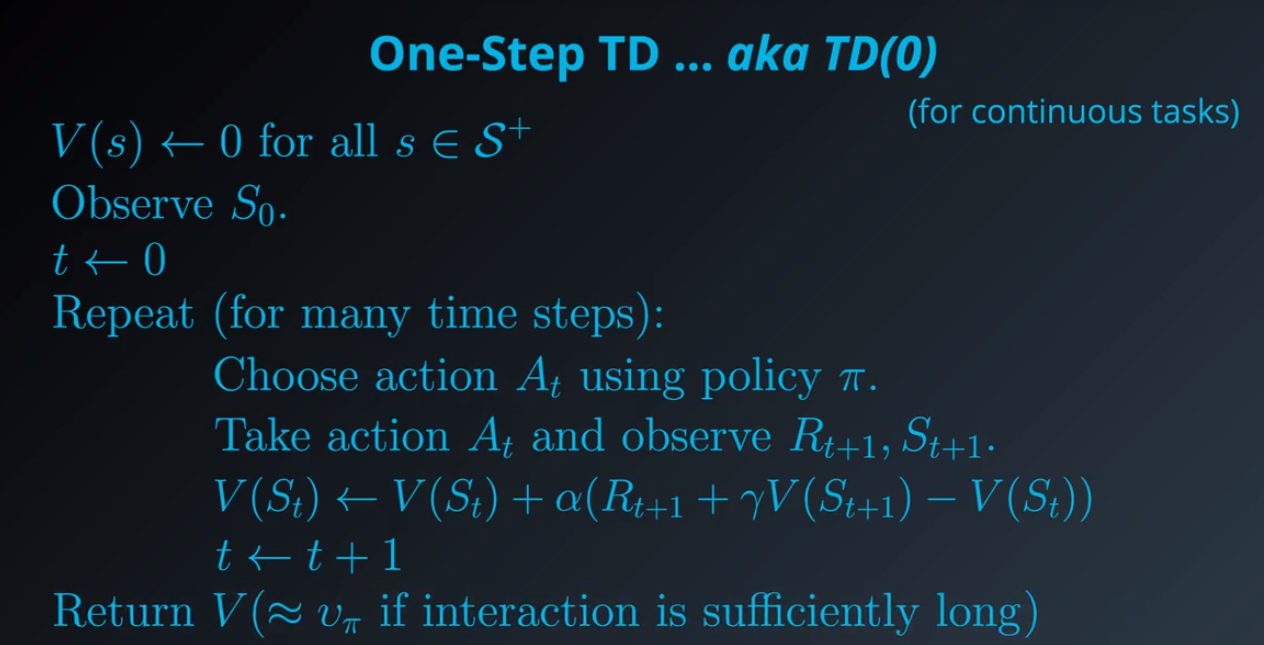


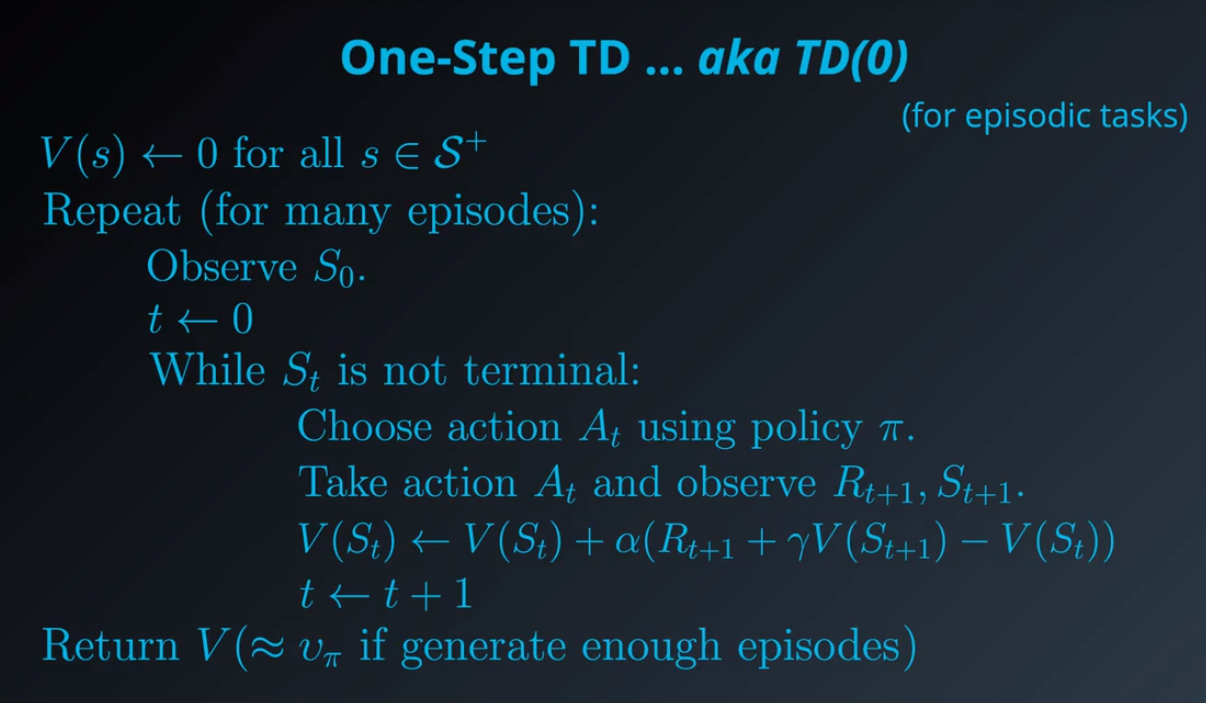
通常将 α 设为一个接近 0 的小值很有帮助

通常 α 越小，在进行更新时，对目标的信任就越低

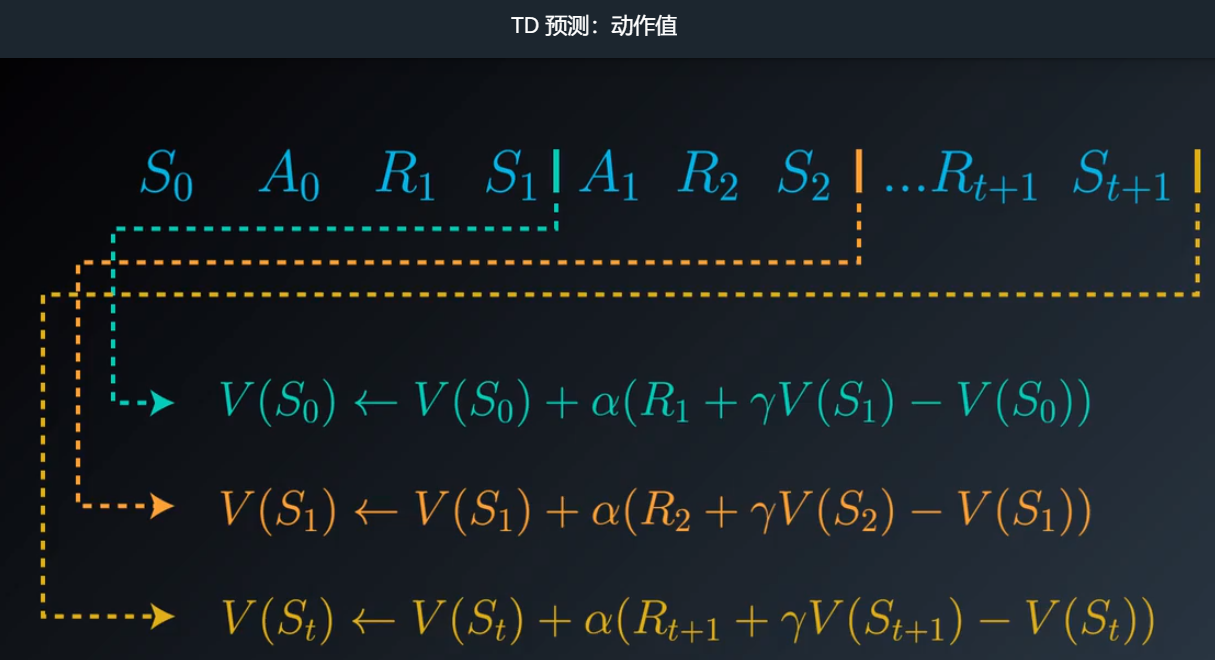
并且更加依赖于状态值的现有估值

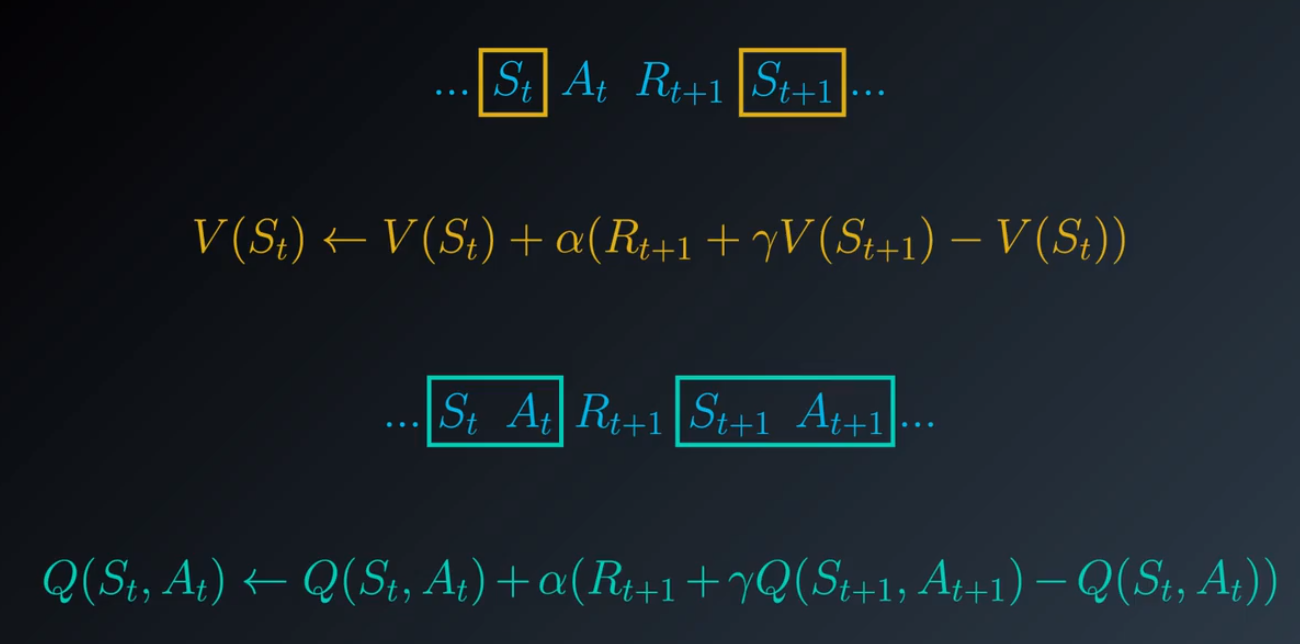
一步时间差分 简称 TD

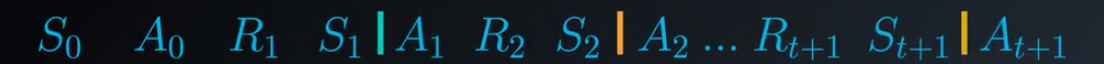




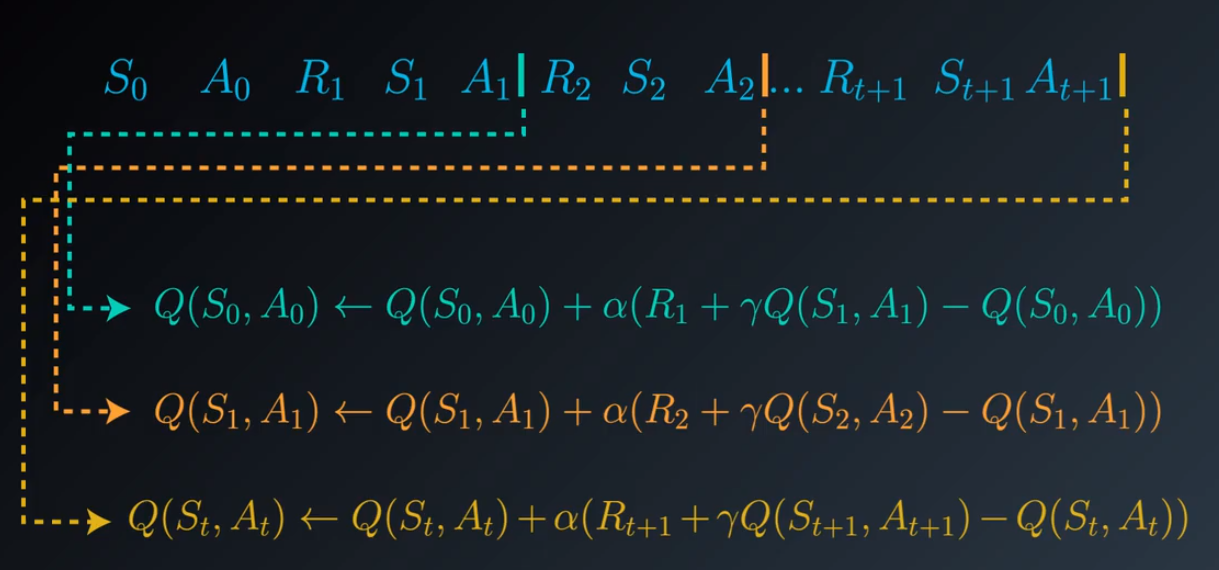
# TD动作值







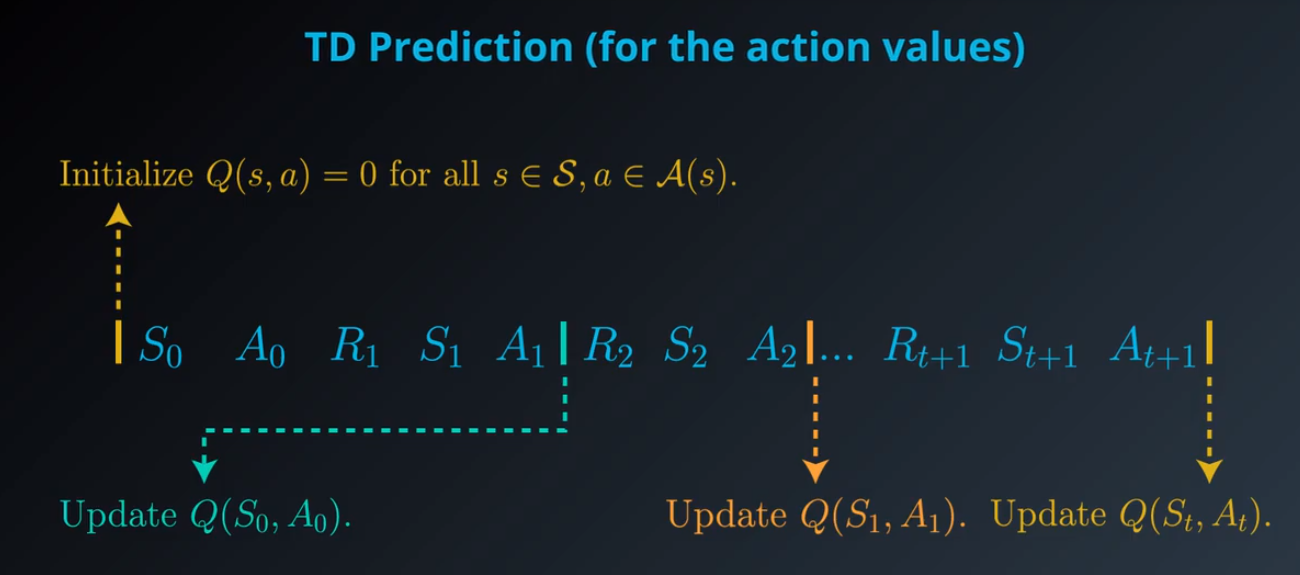
我们不再使用与后续状态的值相关的更新方程，而是需要获得一个与后续状态动作对的值

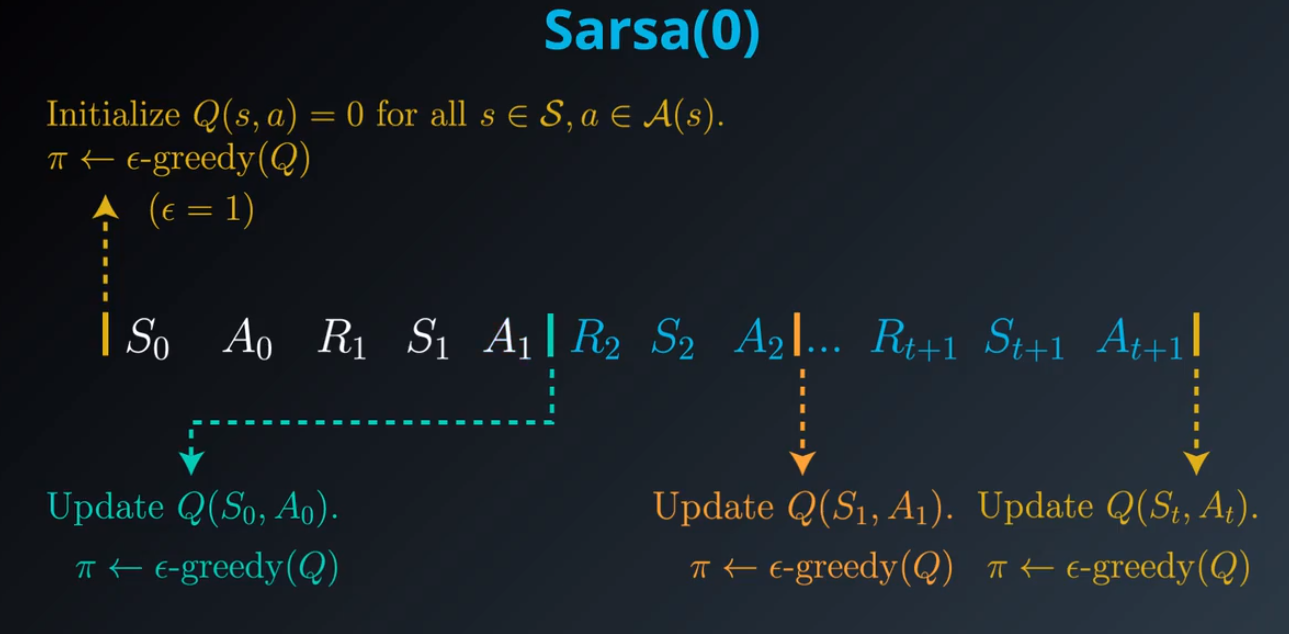


智能体将在每次选择动作后都更新值，而不是在接收每个状态后更新值

如果智能体与环境互动足够长的时间，它将能够很准确地估算动作值函数

# TD控制-Sarsa



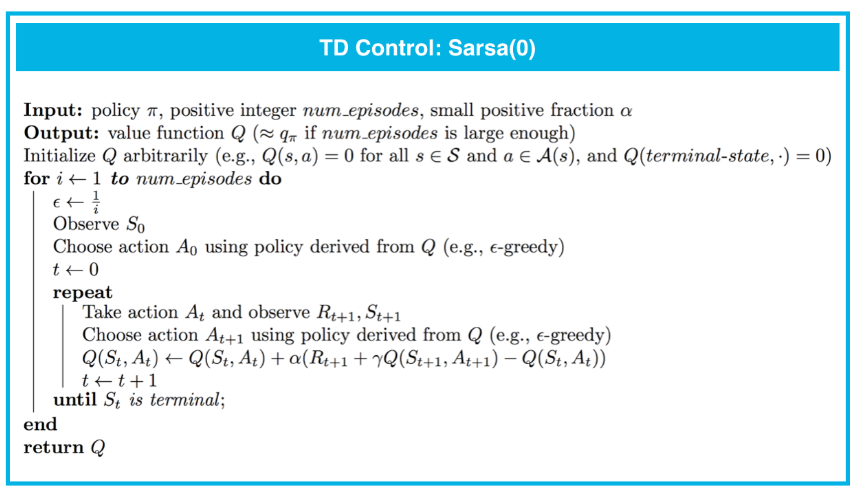


即在每个时间步使用一个针对当前动作估值的,Epsilon 贪婪策略选择一个动作

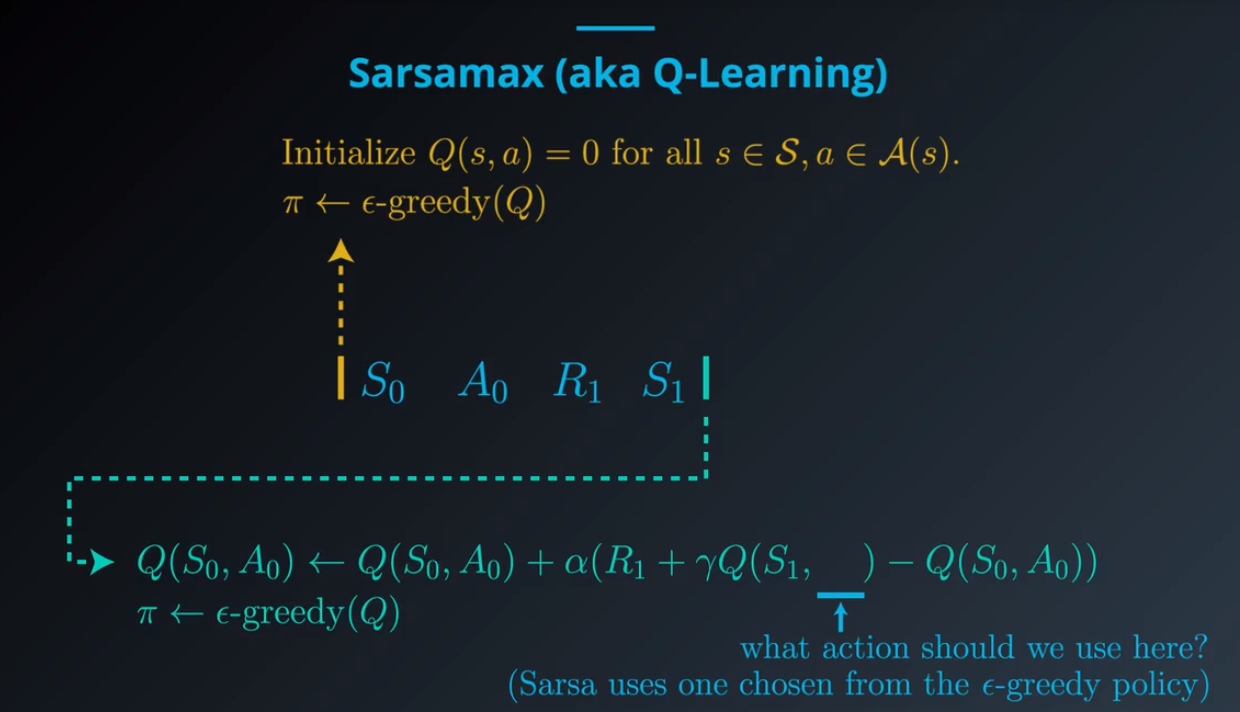
在初始时间步,我们先将 ε 设为 1,然后根据对等概率随机策略选择 A0 和 A1

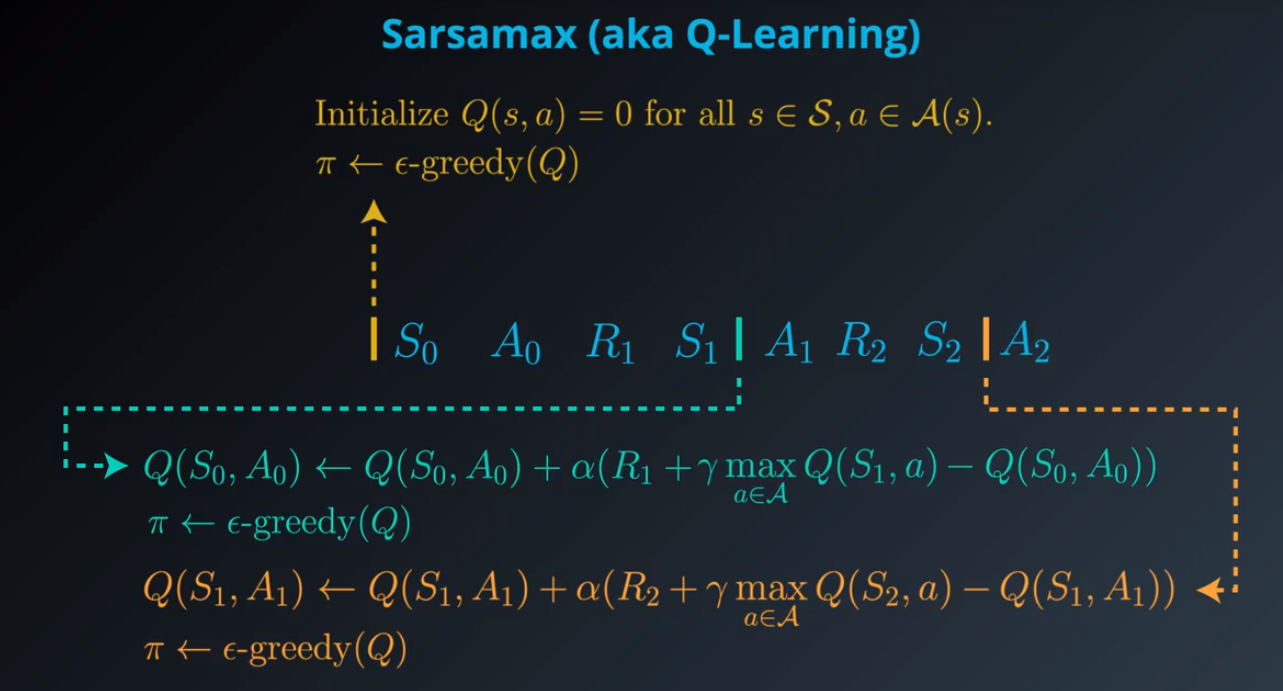
在选择某个动作之后的未来所有时间步，更新动作值函数,并构建相应的 Epsilon 贪婪策略

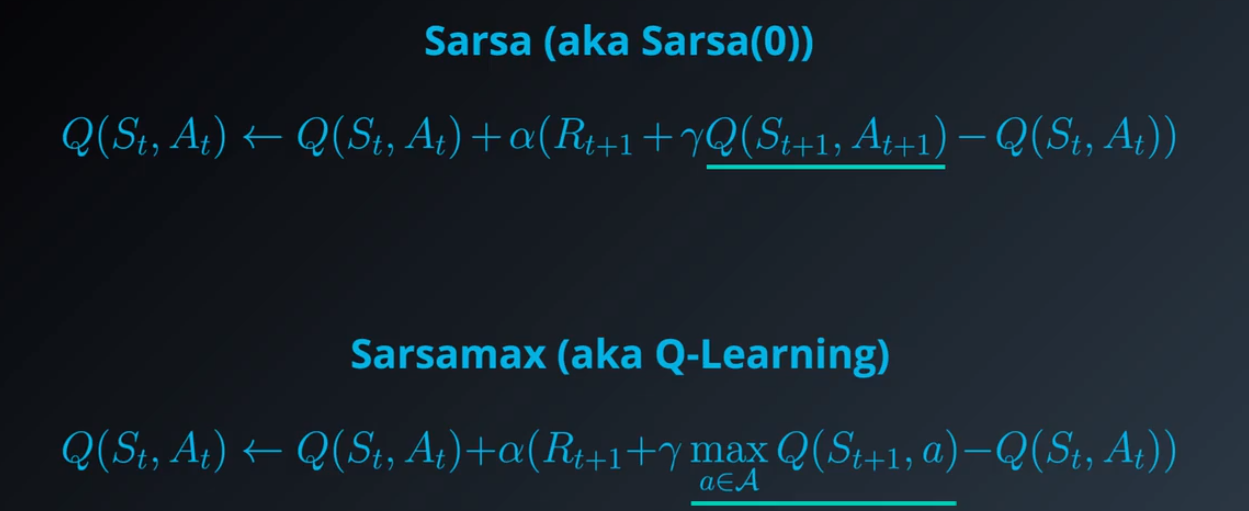
只要我们为 ε 指定合适的值该算法就肯定会收敛于最优策略，该算法的名称叫做 Sarsa 0



# TD 控制-Sarsamax（Q Learning）







# TD控制-预期

