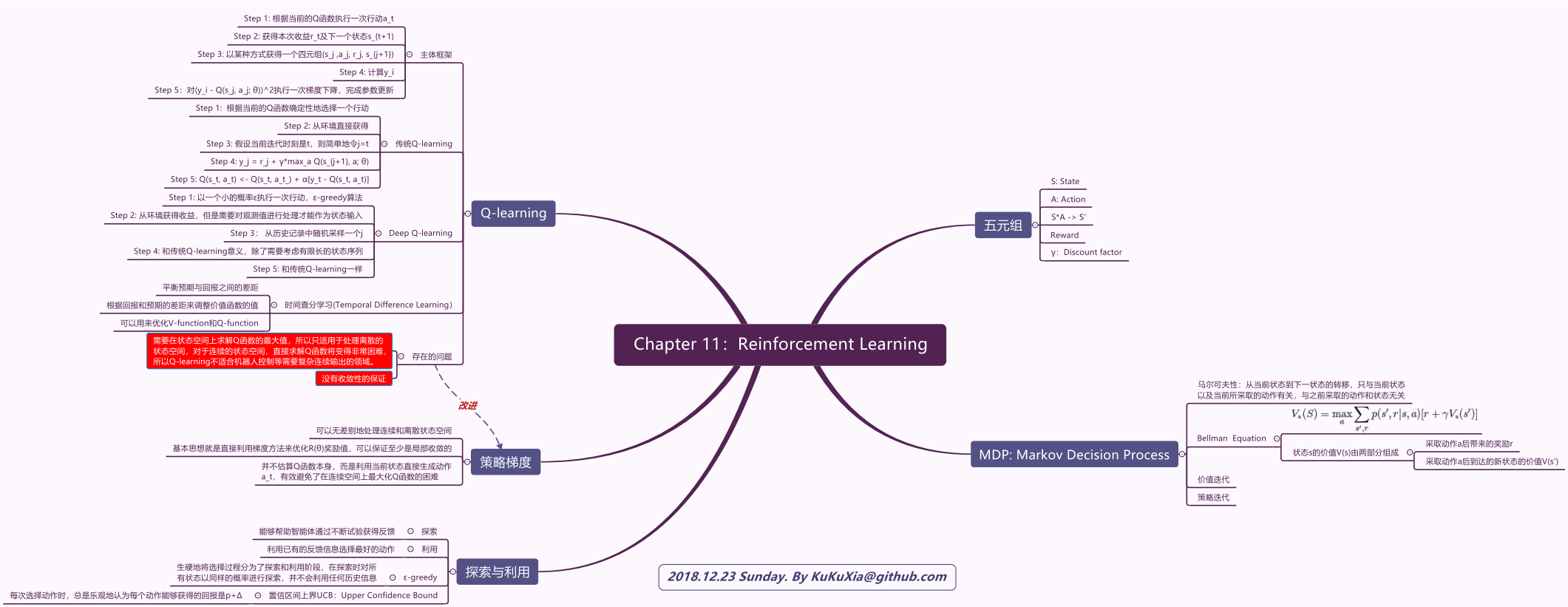
Chapter 11：Reinforcement Learning
Q-learning
主体框架
Step 1: 根据当前的Q函数执行一次行动a_t
Step 2: 获得本次收益r_t及下一个状态s_{t+1}
Step 3: 以某种方式获得一个四元组(s_j ,a_j, r_j, s_{j+1})
Step 4: 计算y_i
Step 5：对(y_i - Q(s_j, a_j; θ))^2执行一次梯度下降，完成参数更新
传统Q-learning
Step 1：根据当前的Q函数确定性地选择一个行动
Step 2: 从环境直接获得
Step 3: 假设当前迭代时刻是t，则简单地令j=t
Step 4: y_j = r_j + γ*max_a Q(s_{j+1}, a; θ)
Step 5: Q(s_t, a_t) <- Q(s_t, a_t) + α[y_t - Q(s_t, a_t)]
Deep Q-learning
Step 1: 以一个小的概率ε执行一次行动，ε-greedy算法
Step 2: 从环境获得收益，但是需要对观测值进行处理才能作为状态输入
Step 3： 从历史记录中随机采样一个j
Step 4: 和传统Q-learning意义，除了需要考虑有限长的状态序列
Step 5: 和传统Q-learning一样
时间查分学习(Temporal Difference Learning)
平衡预期与回报之间的差距
根据回报和预期的差距来调整价值函数的值
可以用来优化V-function和Q-function
存在的问题
需要在状态空间上求解Q函数的最大值，所以只适用于处理离散的状态空间，对于连续的状态空间，直接求解Q函数将变得非常困难，所以Q-learning不适合机器人控制等需要复杂连续输出的领域。
没有收敛性的保证
改进
策略梯度
可以无差别地处理连续和离散状态空间
基本思想就是直接利用梯度方法来优化R(θ)奖励值，可以保证至少是局部收敛的
并不估算Q函数本身，而是利用当前状态直接生成动作a_t，有效避免了在连续空间上最大化Q函数的困难
探索与利用
探索
能够帮助智能体通过不断试验获得反馈
利用
利用已有的反馈信息选择最好的动作
ε-greedy
生硬地将选择过程分为了探索和利用阶段，在探索时对所有状态以同样的概率进行探索，并不会利用任何历史信息
置信区间上界UCB：Upper Confidence Bound
每次选择动作时，总是乐观地认为每个动作能够获得的回报是p+Δ
五元组
S: State
A: Action
S*A -> S'
Reward
γ：Discount factor
MDP: Markov Decision Process
马尔可夫性：从当前状态到下一状态的转移，只与当前状态以及当前所采取的动作有关，与之前采取的动作和状态无关
Bellman Equation
$V_*(S) = \max_a \sum_{s',r} p(s',r|s,a)[r + \gamma V_*(s')]$
状态s的价值V(s)由两部分组成
采取动作a后带来的奖励r
采取动作a后到达的新状态的价值V(s')
价值迭代
策略迭代
2018.12.23 Sunday. By KuKuXia@github.com
ML/DL/RL深度交流讨论群: 622545311，加群时请备注: 姓名-方向-公司or高校