**多臂赌博机实验报告 周报4.12-4.19**

**进展**

本周，工作如下：

1. 对毕设的代码框架进行了重构，重构后可视化效果更佳，独立的算法子类继承于总的算法父类，模块化特征更显著，便于添加自定义算法。
2. 【实验】针对Stochastic Bandit场景编写了ETC，ε-greedy，UCB1算法流程，进行对比试验。

下周计划：

补充UCB2算法等，继续阅读论文。

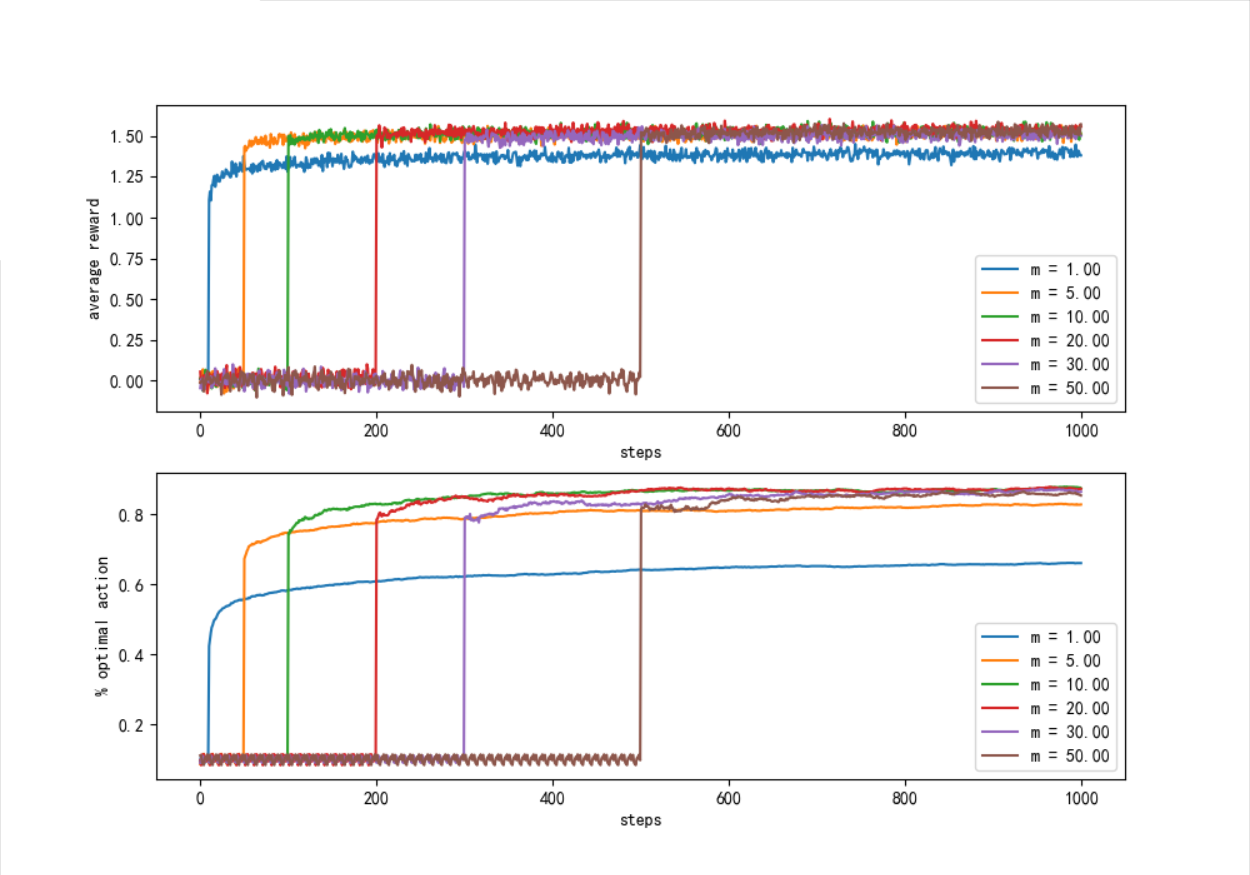
**实验背景**

通过之前对多臂赌博机问题的学习，笔者了解了UCB算法，ETC算法，ε-greedy算法等Stochastic Bandit问题的经典算法，本文主要采用控制变量的方法，改变算法公式中的某些变量，旨在探究不同变量对算法本身效果的影响。最后进行了一组算法之间的对比试验，能更直观的体现UCB算法Stochastic Bandit问题中的优势。

**Stochastic Bandits实验数据**

设定步数step为2000步，每步执行2000轮取平均值，即steps=2000，runs=2000 。

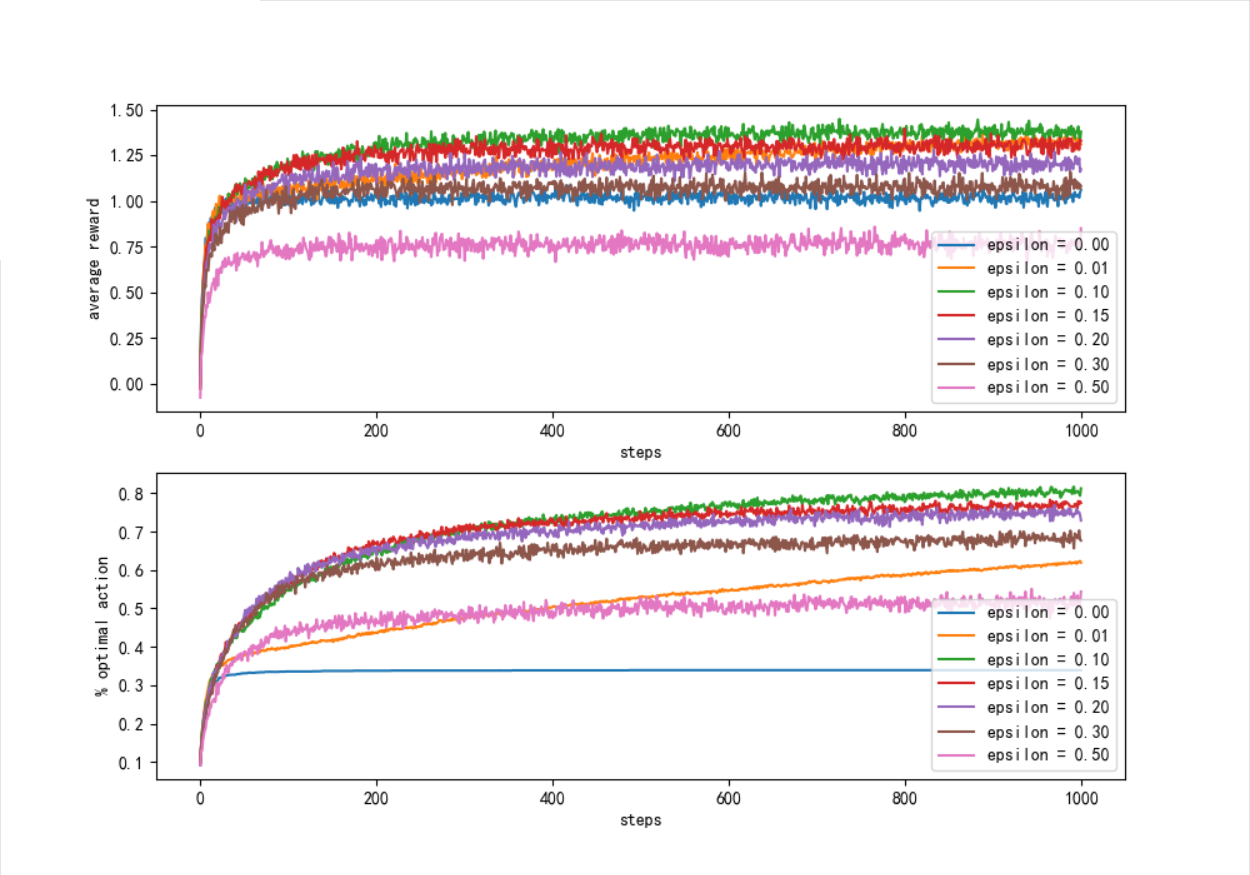
**1.ETC算法参数对比实验**



**分析：**

1. explore-then-commit算法通俗理解就是对每一个arm都探索M次，M轮之后贪婪的选择平均奖励最大的那个arm。从第一张图中我们可以得出，除了仅仅探索M轮的实验，其余各组对照组都取得了1.50左右的平均奖励，说明在M较小，探索不充分的情况下，我们无法准确的选择较优的机械臂。
2. 由第二个图可知，在explore更充分的情况下，选择最优动作的概率越大。图中M>20的对照组中，选择最优动作的概率趋近于0.8，而m=1，m=5的情况下由于探索不充分，选择最优动作的比例相对较低。

**2.ε-greedy调参实验**

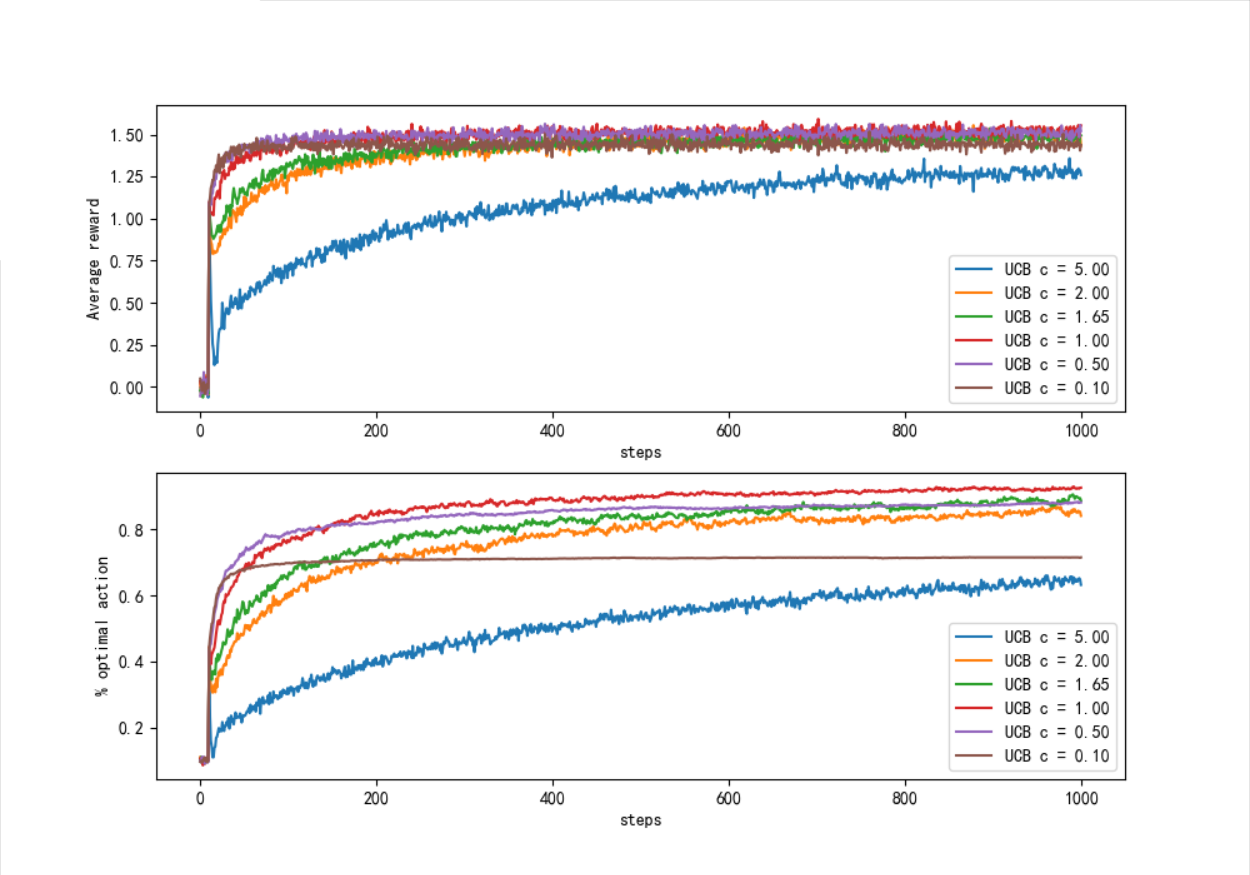


算法流程：ε-greedy算法的流程是，在每一轮选择动作的时刻，智能体以ε的的概率进行探索，随机的选择任意一个动作，以1-ε的概率选择当前平均奖励最大的动作。

实验分析：

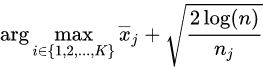
1. 当ε等于0时，相当于不做任何探索，从2000runs的结果可见，选择最优动作的频率是0.3（蓝色曲线。
2. 即使ε为极小的数值0.01，最优动作的占比随着轮数的增加有了明显的增长。
3. 当ε等于0.1时，取得最好的效果（绿色），而当ε增大到0.5时，无论是累计奖励和最有动作频率都明显下降，原因是智能体做了过多的探索，反而适得其反。

**3.UCB算法调参实验**



UCB算法流程：

UCB全称为置信度上界算法，根据如下UCB值选择动作。

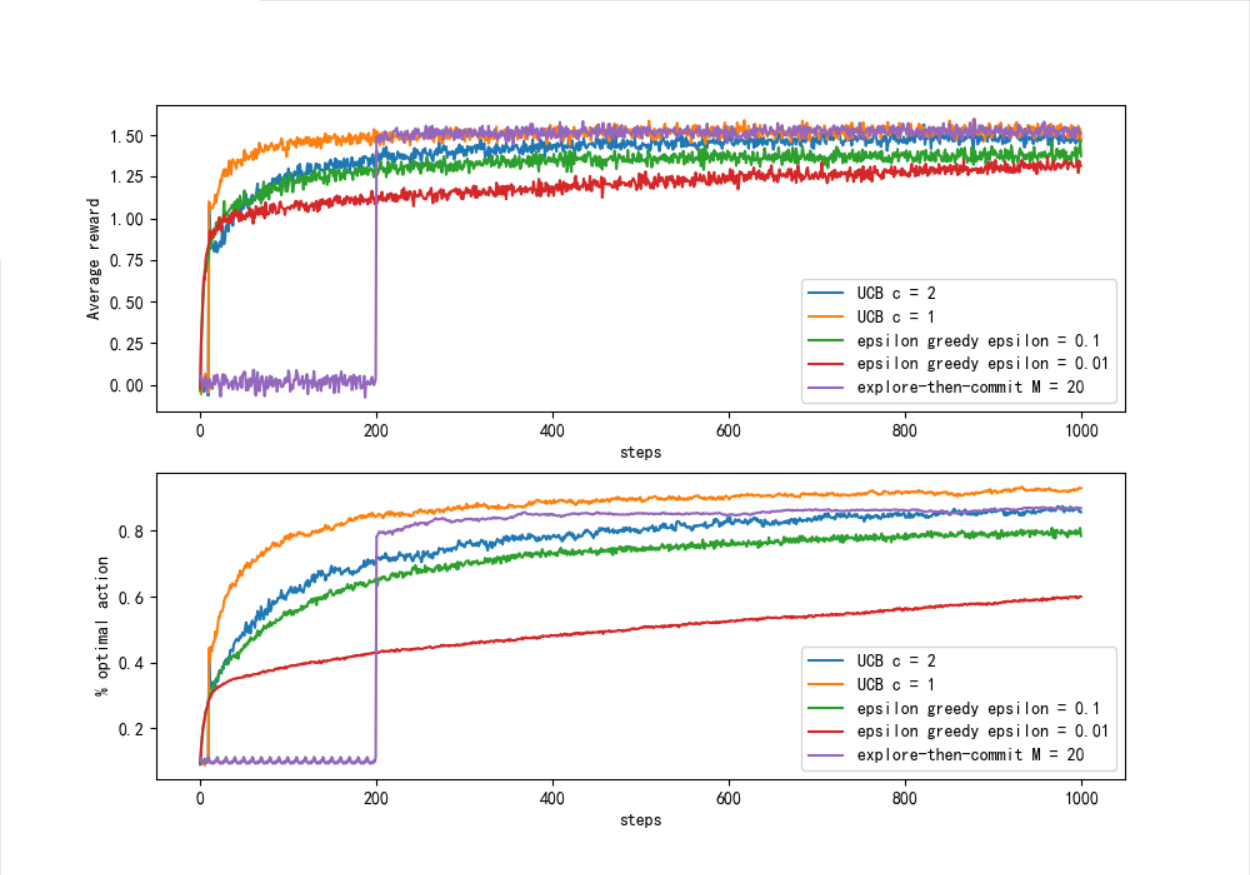


本实验过程中设置UCB\_param即动作选取公式中的c（**以c取代根号2）**，可解释为：**平方根项是对a动作值估计的不确定性或方差的度量，参数c决定了置信水平，但随着某个动作选取次数Nj的增多，其不确定性会逐渐减少。**

实验结论：

1. 当置信水平c非常小的时候（咖啡色），前期动作j的执行次数Nj较小，探索奖励的比重非常小，平均奖励的比重非常大，使得**前期平均奖励能够较大，最优动作的选择占比也较高**。然而随着动作执行次数增大，探索奖励逐渐失效。
2. 当c等于0.5或1时，UCB算法取得较好的效果（红色与粉色两条曲线）
3. c越小，前期选择最优动作的比率越大，但是后期随着探索奖励不确定度的下降，最优动作选择频率下降（粉色低于红色）。

**4.算法对比实验**



实验结论：

1. 通过ε-greedy，ETC，UCB1算法的对比，我们能够发现UCB1算法在Stochastic Bandit问题中取得的效果最好。
2. UCB算法和ETC算法取得的平均奖励显著高于ε-greedy算法。但是，由于ETC算法前期需要进行比较充分的探索，因此最优动作选择的概率总体小于UCB1算法，综上所述UCB1算法表现最好。