100囚犯问题仿真报告

100 名囚犯编号为 1 至 100。监狱长准备一个房间，内有 100 个盒子，每个盒子内随机放入一张囚犯编号的纸条（编号不重复）。囚犯依次进入房间，每人可打开最多 50 个盒子寻找自己的编号。若所有囚犯均在 50 次尝试内找到自己的编号，则全体获释；否则全员失败。

一.算法分析

1.随机策略

每个囚犯从所有 N 个盒子中随机抽取 K 个盒子进行检查；如果任意一个囚犯没有在抽查的盒子中找到自己编号，则全体失败；每个人的抽查完全独立，不考虑全局排列结构。

设N是囚犯数，K 是尝试次数，T为模拟次数，则单次模拟时间复杂度为$O(N \times K)$，而总体模拟时间复杂度为$O(T \times N \times K)$

2.循环策略

每个囚犯从自己编号的盒子开始，看里面的编号 a1；如果没有找到再去看编号为 a1 的盒子，得到 a2，……直到最多跳 K 次；如果在 K 步内找到自己的编号，成功，否则失败；最终的算法实现即为检查排列中的所有循环长度是否 $\leqslant K$

单次实验对应最坏时间复杂度为$O(N \times K)$，总体模拟为$O(T \times N \times K)$，但实际情况由于使用了上文提到的检查循环长度条件的剪枝优化，最终的结果实际优于随机策略

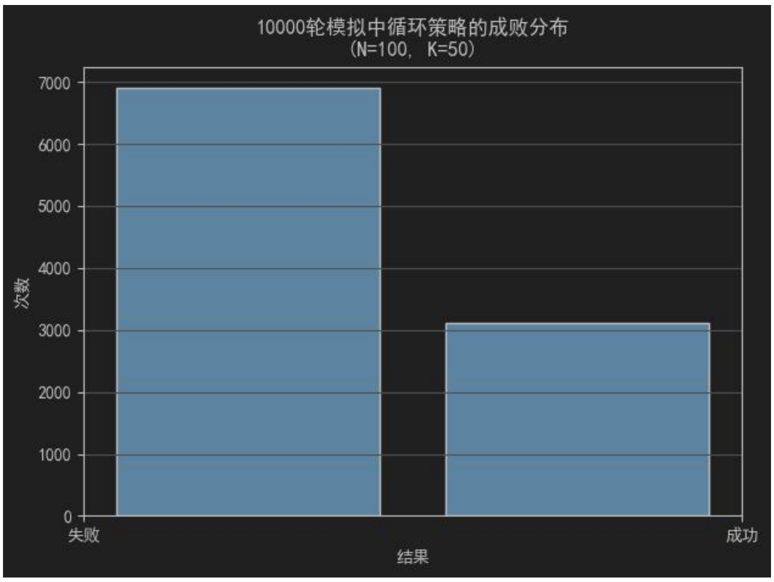
二.实验结果

在默认情况下，得到的两种策略成功率结果如下：

策略成功率对比：

策略	成功率
随机策略	0.00%
循环策略	31.04%

其中策略2的对应模拟结果如图：



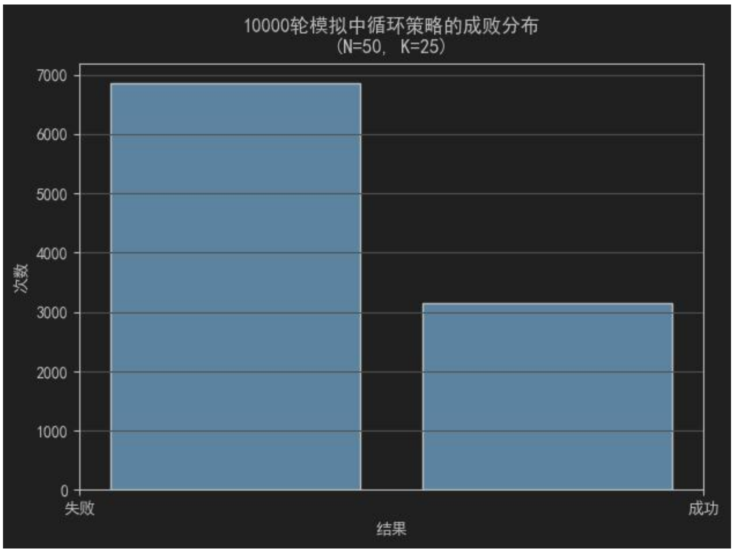
三. 扩展分析

1. 调整 N 和 K，观察成功率变化

调整为N = 50， K = 25：

策略成功率对比：

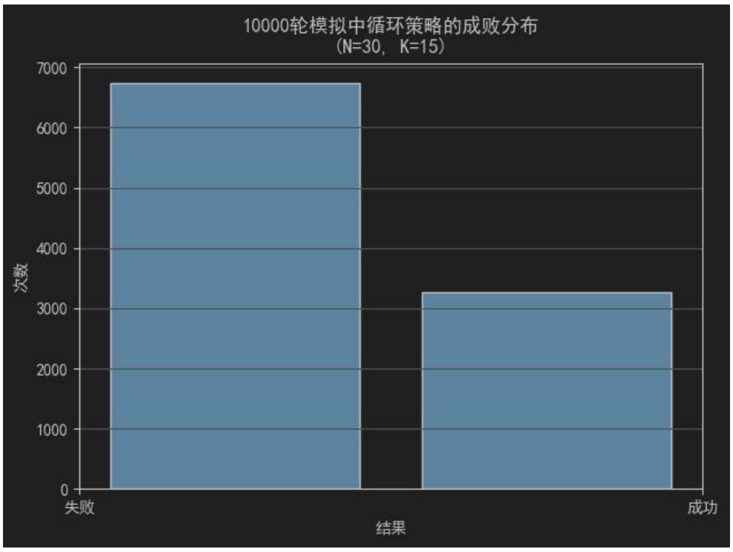
策略	成功率
随机策略	0.00%
循环策略	31.50%



调整为N = 30, K = 15：

策略成功率对比：

策略	成功率
随机策略	0.00%
循环策略	32.72%



2. 理论计算最优策略的成功率

由于在循环策略中，每个囚犯会沿着一条排列循环移动。一个囚犯成功的前提是他所在的循环长度 ⩽ K。如果所有循环长度 ⩽ K，则所有囚犯成功。那么问题就转换为：一个随机排列中，是否所有循环的长度 ⩽ K？我们通过仿真方式统计最大循环长度是否 ⩽ K，次数足够时即可逼近真实理论成功率

在默认情况下有：

理论成功率估计（基于排列循环模拟）：30.96%

实际模拟成功率 ≈ 理论成功率，说明代码已成功实现仿真。同时理论值是大数定律下的“平均期望值”，而实际模拟值也收敛到了类似的结果，说明模拟轮数已基本足够，不再剧烈波动，总之循环策略的数学模型是准确的，模拟程序也是可信的。

实际上，对于此问题的数学分析如下：

假设某个排列中存在一个长度为 l 的循环（其中 $l > 50$），那么包含这样一个循环的排列数为：

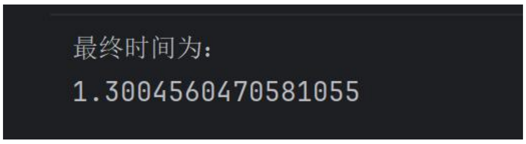
$$C(100, l) \times (l - 1)! \times (100 - l)! = 100! / l$$

对所有长度 $l > 50$ 的循环求和（即 $l = 51$ 到 100）：
总失败数 $= \sum_{i=51}^{100} (100! / l)$
失败概率 $= (1 / 51 + 1 / 52 + ... + 1 / 100)$
成功概率 $= 1 - (1/51 + 1/52 + ... + 1/100)$
成功概率 $\approx 1 - (H_{100} - H_{50}) \approx 0.31183$
其中 H_n 表示第 n 个调和数$H_n = 1 + 1/2 + 1/3 + ... + 1/n$

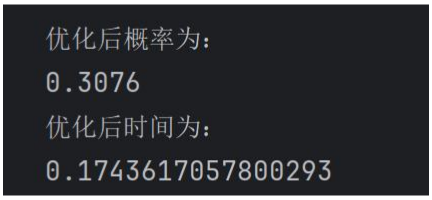
四. 优化策略

本实验通过判断最大循环长度是否 ≤ K进行了相应的剪枝，从而达到优化的目的
同时使用NumPy数组代替list，在默认情况下重构计算，得到以下结果：

原方案运行时间：



现方案运行时间及概率：



可以看出，优化后方案在相同数量级下降低时间消耗的同时保证了正确率，程序也是可信的