计算物理第 5 次作业第 12 题

姓名：姚星宇　　学号：PB21000188　　班级：2021 级少年班学院四班

2022 年 11 月 15 日

1　题目重述

推导正方格子点阵上键逾渗的重整化群变换表达式 p,=R(p)，求临界点 p_c 与临界指数 ν，与正确值相比较。这里没有直接推导出结果，而是采用 Monte Carlo 重整化群方法在计算机上进行计算。

2　题目分析

由参考材料 [1] 中关于 Monte Carlo 重整化群方法相关内容，进行 Monte Carlo 重整化群方法的步骤大致如下：

1. 生成初始逾渗结构

2.i) 判断是否形成逾渗通路

ii) 若形成，则结束循环；若未形成，则添加联通结构并返回 i

如上循环循环，并在形成通路时结束循环，根据循环次数对概率的数组进行操作

3. 得到通路个数与联通概率的数组后，解出中间的不稳定不动点，得到临界概率

4. 根据临界概率计算其他临界参数

下文将分别叙述本次作业在这四个步骤中用到的方法

2.1　生成初始逾渗结构

对于本题所研究的二维正方形键逾渗，由于纵键与横键具有一定的差别（需要判断纵向联通而不需要判断横向联通），且在连接性质上也具有一定的不同，故将两种键的联通情况分开存储。对于边长为 n 的网格，横键采用大小为 (n,n+1) 的二维 Bool 数组 par 存储。其中，边缘的两列（即第 0 列与第 n 列）是为了编程方便与防止数组越界而引入的保护列，固定为 False。数组中的第 i，j 个元素（即 par[i][j]）表示座点 (i,j) 左方横键的联通状态，True 为联通，False 为断开。同理，纵键采用大小为 (n+1,n) 的二维 Bool 数组 ver 存储。其中，边缘的两行（即第 0 行与第 n 行）是为了编程方便与防止数组越界而引入的保护列，固定为 False。数组中的第 i，j 个元素（即 ver[i][j]）表示座点 (i,j) 下方横键的联通状态。

在构建初始结构时，首先新建上述素组，并全部初始化为 False（即不连通）。然后根据传入的参考 p 值，确认需要添加的数量。综合效率和精确度，这里选择当参考概率为 p 时，添加 $const = int(1.9pb(b+1)-40)$ 处联通（总键数 $N = 2b(b+1)$）。对于每一次添加，首先在选择生成在 [0,1] 间均匀分布的随机数 ξ，然后得到抽样键的编号 $temp = int(2*b*(b+1)*\xi)$。根据一定的规则映射到两个网格中（如：temp<b(b+1) 时,ver[temp//b+1][temp%b] = True, 反之则 par[temp%b][temp//b-b+2] = True）。需要注意的是，若目标抽样点已经联通，则返回生成随机数的步骤，并不增加计数；反之，若成功添加，则增加计数。可以估计的是，这种实现起来较为容易的不重复抽取会随着已经添加点的增多而效率降低，但在 $p \sim 0.5$ 的情况下，效率的降低是可以接受的。当计数等于目标数量时，停止增加，得到初始化完毕的结构。

2.2 判断是否形成通路、重复添加与重复判断

对于初始化完毕的结构，我们需要一定的方法来判断是否形成通路。首先，我们需要一个数据结构来存储与下端联通的座点。这里采用大小为 (n,n) 的二维 Bool 数组 site，并全部初始化为 False，若与下端联通，则改为 True。对于 site，最下端的点按照定义全部为通路起点；且若最上端的点有联通的，即 site[n-1] 中存在 True，则视作整个图联通。已知起点与终点，要判断是否有路径联通起点与终点，可使用广度优先的寻路算法。遍历每个起点 site[0][i]————若起点已经被赋值为 True(site[0][i] == True)，则该起点所对应路径已经被搜索过，应该跳过这一点；若 site[0][i] == False，则表明还未被搜索过，故应该从这点开始一次搜索。在一次搜索中，较为重要的是通路边界点和对应的储存边界点坐标的列表 pio。每次搜索开始时，首先将将起始点加入 pio，然后将起始点赋值为 True。然后进行循环。每一轮循环中，编历 pio 中的每一点 j(x,y)，并进行如下判断：如果 j 的四周中 ((x+1,y), (x-1,y),(x,y+1),(x,y-1)) 存在点 k 满足以下条件：k 在 site 中的值为 False，且 k 与 j 的连接为通路，则将 k 加入 pio 中，并将点 k 赋值为 True。若 k 为顶端点（k(x,n-1)）则直接跳出所有循环，并返回已经联通的信息（flag = True）。在搜索完周边点后，将 j 从 pio 中移除。如此循环，直至 pio 中无数据（len(pio)==0），即完成 site[0][i] 的搜索。将所有起点搜索完后，即可得到包含所有与下端联通的点的数组 site。

在继续添加数组时，使用与第一节相同的算法，得到新的联通键。为识别与标记新增的联通座，首先，判断新的联通点两端的座的状态是否相异。若相同，则不会造成变化，无需更新 site。若相异，则将原本为 False 的座赋值为 True，并将它的坐标加入 pio 中。以与上文相同的办法进行搜索，即可对 site 进行更新。

这一阶段输出的结果为长为 (N-const+1) 的一维数组 res。添加了 n 个点后检测到联通后，res[n]+=1；若在初始化的过程中即联通，则 res[0]+=1。上述过程重复给定的 M 次后，对 res 进行如下处理：res[i+1] = res[i]，其中 i 遍历 0-(N-const-1)。然后将 res/N 返回，作为这一阶段计算的输出。

2.3 解出不动点

对于上一步得到的结果 res，其意义为：

$$p^* = \sum_{i=0}^{N} P_N(i) = \sum_{i=0}^{N} k(i) C_N^i p^i (1-p)^{N-i}, \text{其中} k(i) = res[i-const] \tag{1}$$

对于 b 较大的情况，上述公式可近似为：

$$0 = p - \sum_{i=const}^{N} \frac{res[i-const]}{\sqrt{2\pi Np(1-p)}} \exp\left(-\frac{(i-pN)^2}{2Np(1-p)}\right) \tag{2}$$

即可从一个估计 p 值开始（如 p = 0.3）定步长搜索零点，即可得到临界 p 值。

2.4 临界指数的计算

由临界指数计算公式 (参考材料 [1])

$$\nu = \frac{\log_2(b)}{\log_2(\frac{dp^*}{dp})} \tag{3}$$

又由式 (1)，可得：

$$\frac{dp^*}{dp} = \sum_{i=0}^{N} \left(\frac{i}{p} - \frac{N-i}{1-p}\right) P_N(i) \tag{4}$$

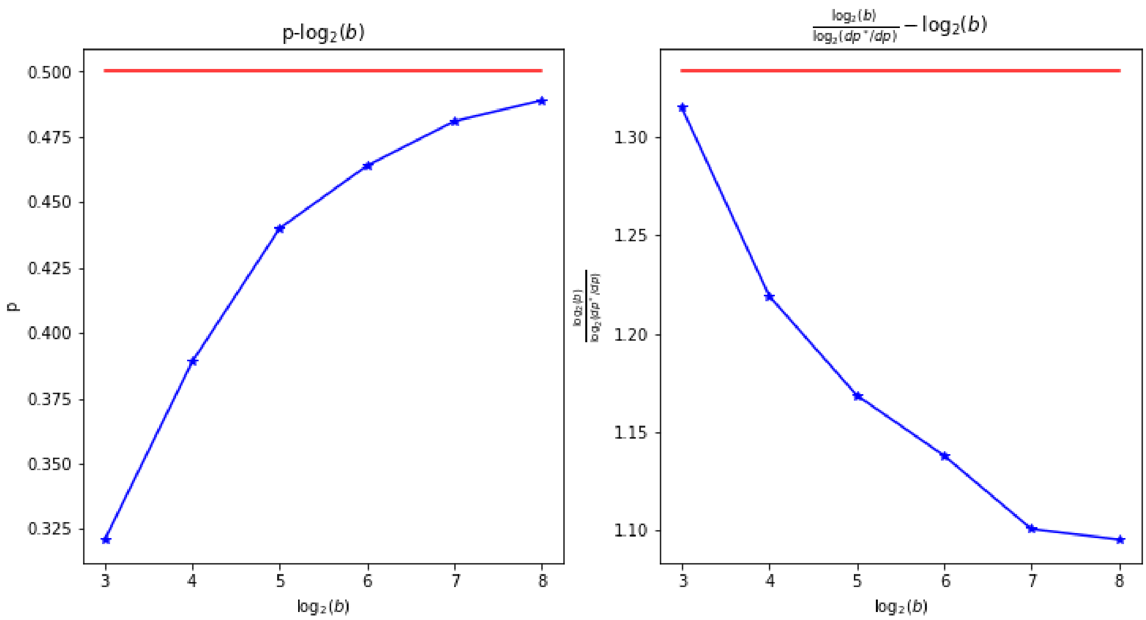

图 1: 程序运行结果

即为:

$$\frac{dp^*}{dp} = \sum_{i=const}^{N}(\frac{i}{p} - \frac{N-i}{1-p})\frac{res[i-const]}{\sqrt{2\pi Np(1-p)}}\exp(-\frac{(i-pN)^2}{2Np(1-p)}) \tag{5}$$

即可计算出临界指数。

3 结果

如图所示，临界概率以较快的速度逼近标准值 0.5，在 $b = 2^8 = 256$ 时，p = 0.487, 与标准值误差小于百分之二，可认为程序实现了目标。对于临界指数，也可发现其在 0.86 附近震荡，可认为此时较好的近似值为 $\nu = 1.10$，和精确值差别较大且没有靠近的趋势。造成这一情况的原因未知。

参考文献

[1] 丁泽军. 计算物理讲义 [M]