**深 圳 大 学 实 验 报 告**

**课程名称： 算法设计与实现**

**实验项目名称： 实验3 消消乐实验**

**学院： 计算机与软件学院**

**专业： 计算机科学与技术**

**指导教师： 杨烜老师**

**报告人： 李若龙 学号：2018171028 班级： 计科02**

**实验时间：**

**实验报告提交时间：**

**教务处制**

1. 问题描述：

消消乐问题，如果有连续三个同类方块，那么消去得1分，4个4分，5个10分，K为方块的种类，M,N为棋盘尺寸，X为可以移动的次数

1. 给定K, M, N编写代码计算通过一步操作（X=1）可得的最大得分。

例如，K=4, M=8, N=4

测试数据（对应图1）

3 3 4 3

3 2 3 3

2 4 3 4

1 3 4 3

3 3 1 1

3 4 3 3

1 4 4 3

1 2 3 2

2. 在1的基础上利用回溯算法，找出X交换步骤之后的最大得分。

3. 对于数值较大的K、M、N、X，在允许近似最优解的情况下，对2中实现的算法进行优化剪枝。并与内容2中最终结果和执行速度进行比较。

4. 如果能实现可视化输出计算结果（包括回溯过程），如图2，可加分。

实验要求：

1. 对不同K，M, N, X的问题依次求解，演示你的求解结果，请提供你机器上能求解的问题最大规模。

2. 在blackboard提交电子版实验报告。源代码和PPT作为实验报告附件上传。

3. 在实验完成之后，将进行一次PPT介绍。

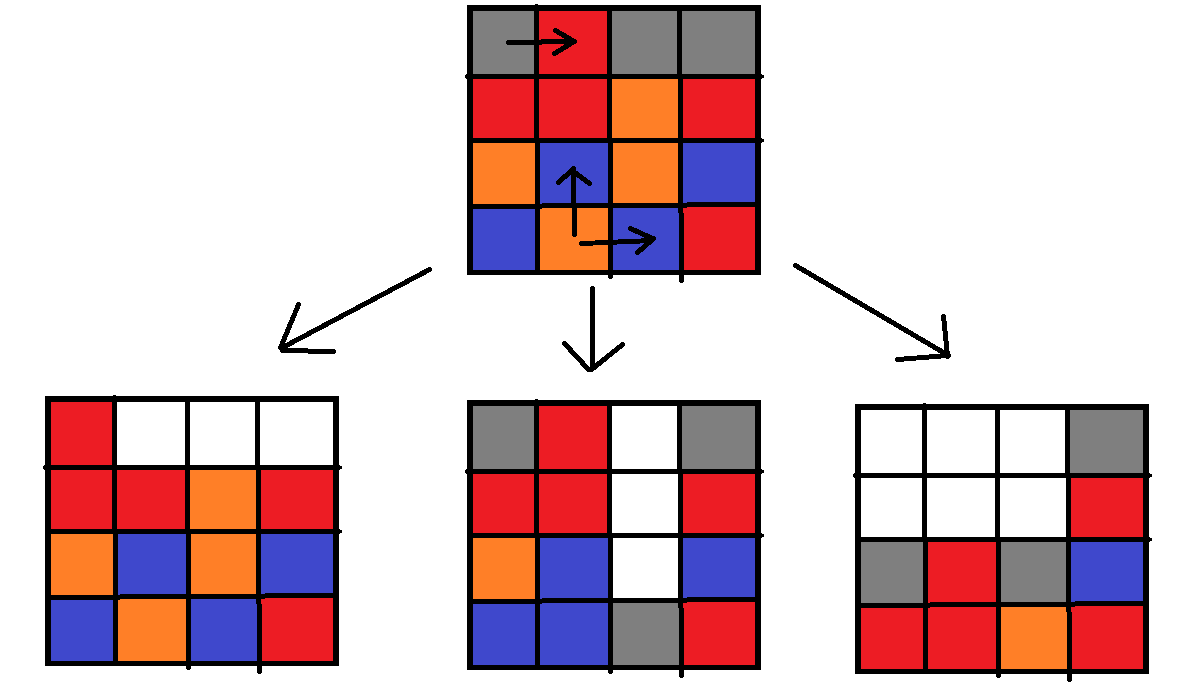
4. 在实验报告中要求详细说明”实验内容1”和“实验内容2”的实现思想。

5. 讨论”实验内容1”和“实验内容2”问题复杂度和K, M, N, X的关系。

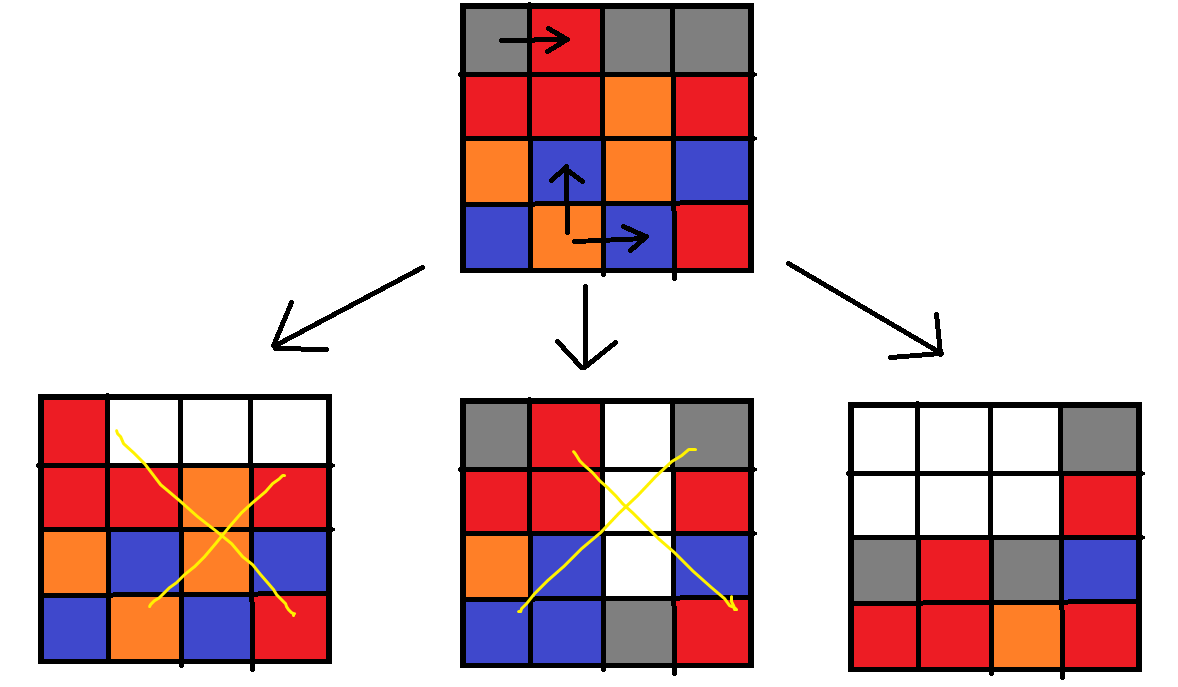
1. 求解问题的算法原理描述

对于棋盘中的每一个方块，我们都可以做四个操作：与 上 下 左 右 交换

对于每个操作，都可能产生方块的消除，进而产生新的棋盘状态，而这些新状态又可以继续操作，进而可以通过回溯法**遍历每个状态**来查看所有操作的可能的得分，找到最大的得分



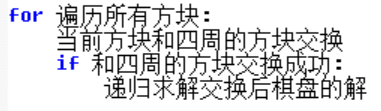
剪枝优化：不一定要对所有的状态都进行递归，而是选取前topk大的状态进行递归，这很容易剪掉一些显然不可能是最大答案的分支，topk的选择可以是1，3，5，7，….



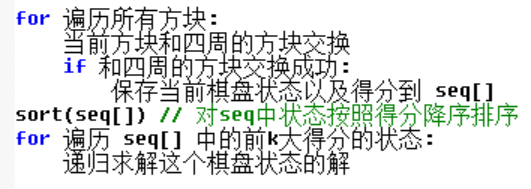
记忆化：在回溯法的基础上，将答案保存，如果遇到相同状态的棋盘，不必再次求解，直接取保存的答案，节省计算的时间开销

1. 算法实现的核心伪代码

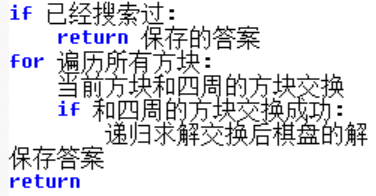
回溯法



回溯法+剪枝，只保留前k大的状态



记忆化递归：



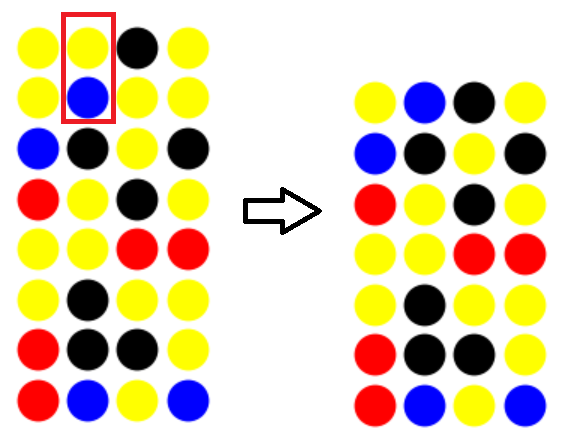
1. 算法测试结果及效率分析

回溯法：得分测试

最大得分测试：样例：老师提供的样例 k=4 n=8 m=4 x=1,2,3,4,5,6,7,8,9

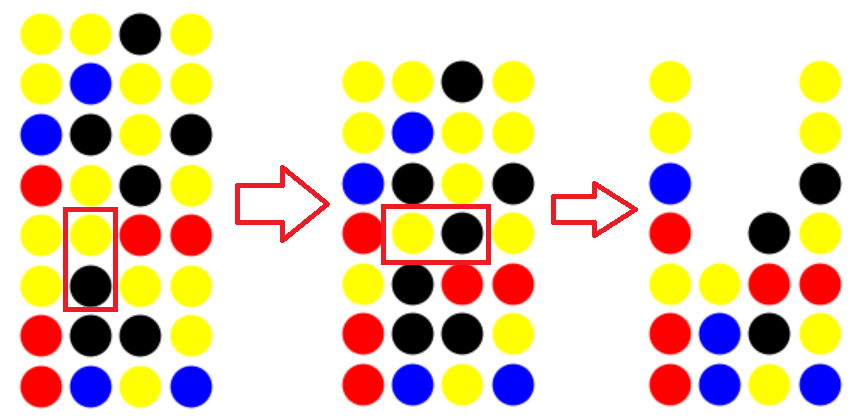
**移动一次：**

回溯法最大得分 4



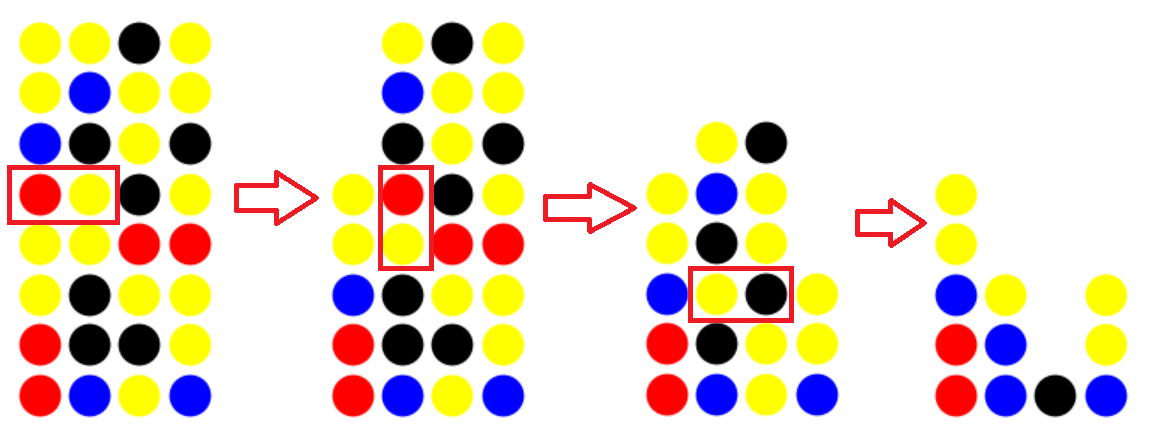
**移动两次：**

回溯法最大得分：9



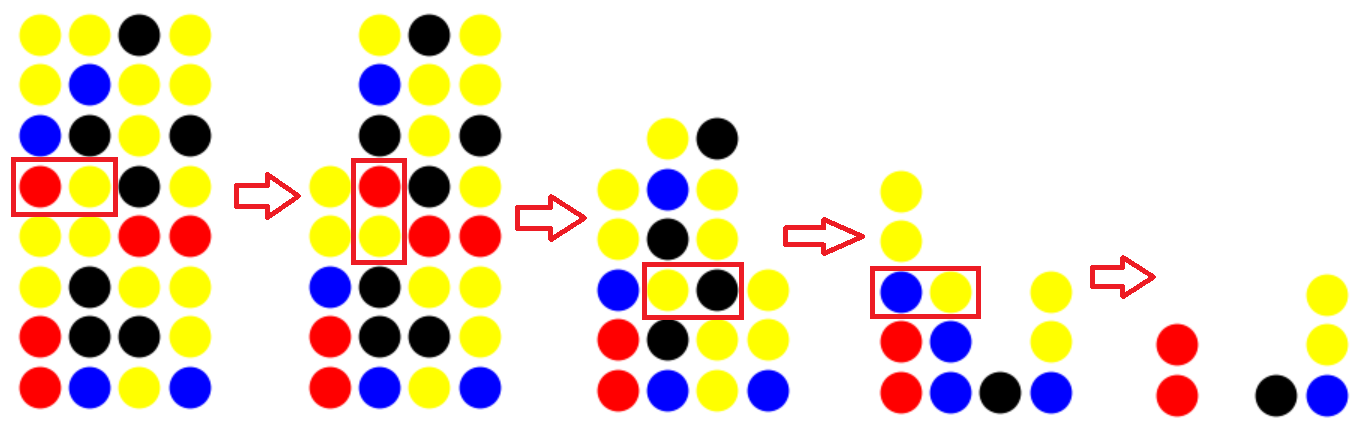
**移动三次**

回溯法最大得分：15



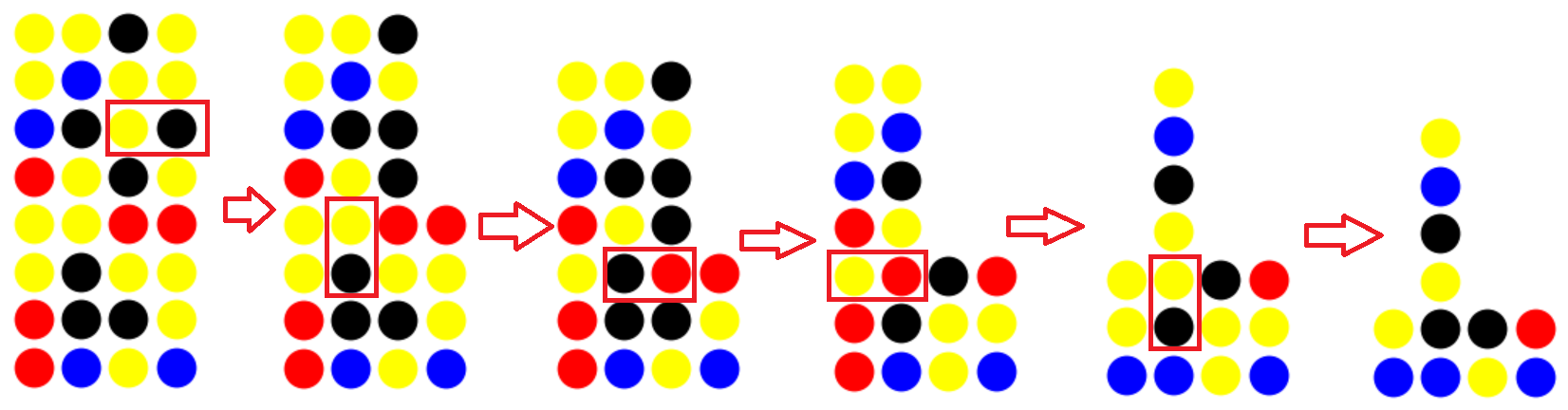
**移动四次**

回溯法最大得分：17



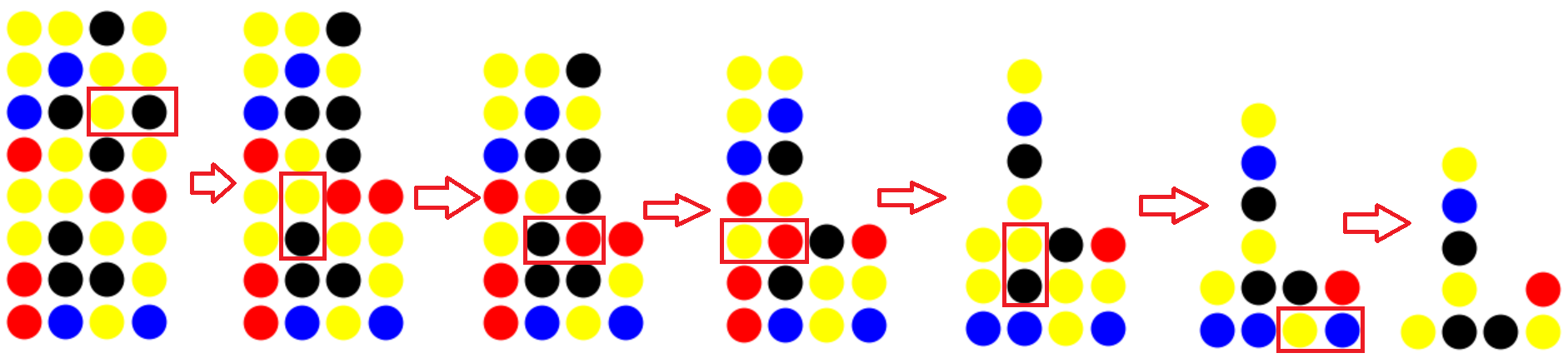
**移动五次**

回溯法最大得分：20



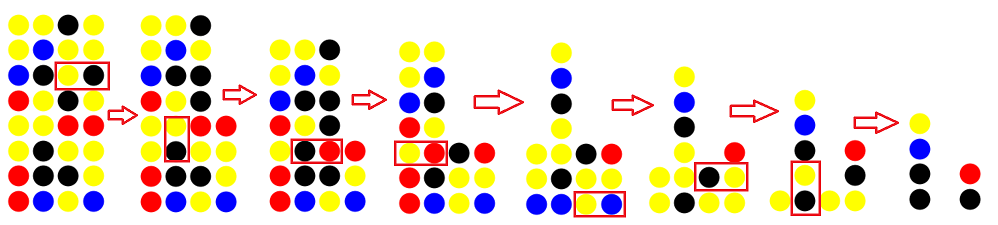
**移动6次**

回溯法最大得分：21



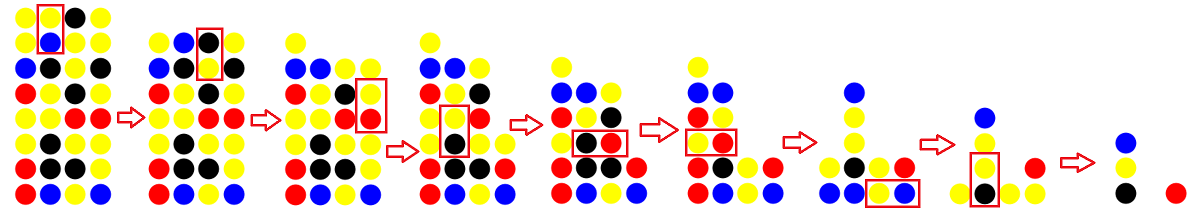
**移动7次：**

回溯法最大得分：22



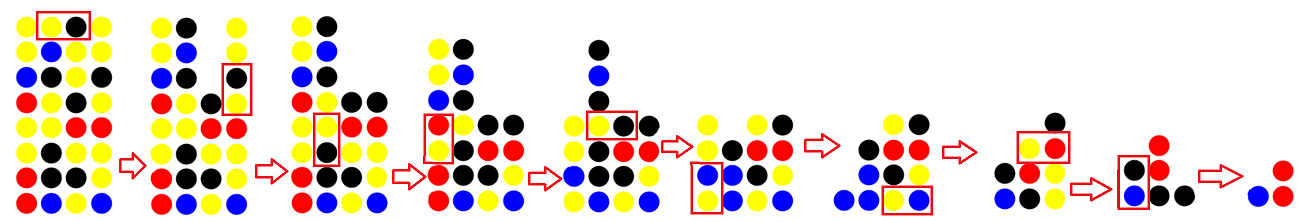
**移动8次：**

回溯法最大得分：20



**移动9次**

回溯法最大得分：15



**移动10次**

无 法 做 到 移 动 十 次

**回溯法：最大规模**

因为数据是随机生成的关系，时间并不好测量，具有很大的随机性，如下例子

K=8 m=8 n=8 x=7 用时79.054 s

K=8 m=9 n=9 x=7 用时12.284 s

但是这个规模K=8 m=9 n=9 x=5的随机样例，经过大量测试，能在15s左右解完

可以近似认为这就是求解问题的最大规模了

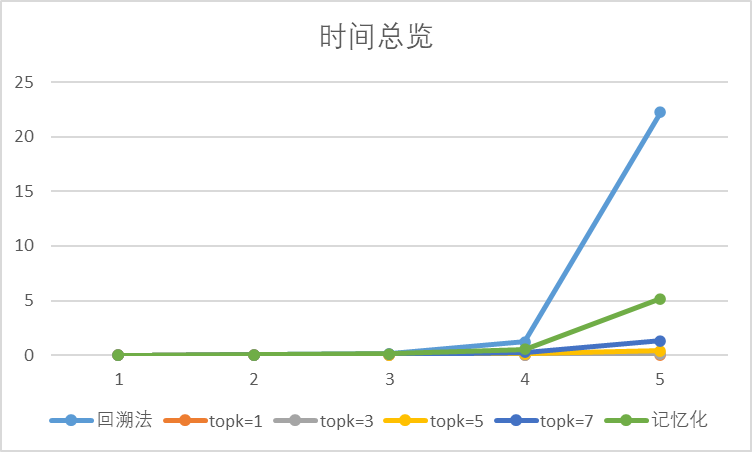
**回溯法：时间复杂度**

每一个棋盘都有m\*n个方块，每个方块有4种可能的移动方式，一共4\*m\*n种新状态，而消去方块的复杂度是O(m\*n)，对每种状态都要消去方块，操作的次数是x意味着递归的深度是x，时间复杂度O((m\*n\*m\*n)x) = O((m\*n)2x)

**时间测试**

测试规模：k=6 m=8 n=8 x=1,2,3,4,5，横坐标操作次数，纵坐标时间(s)





因为回溯法用时太长，导致图表结果不够直观，我们去掉回溯法的用时（如下图表）

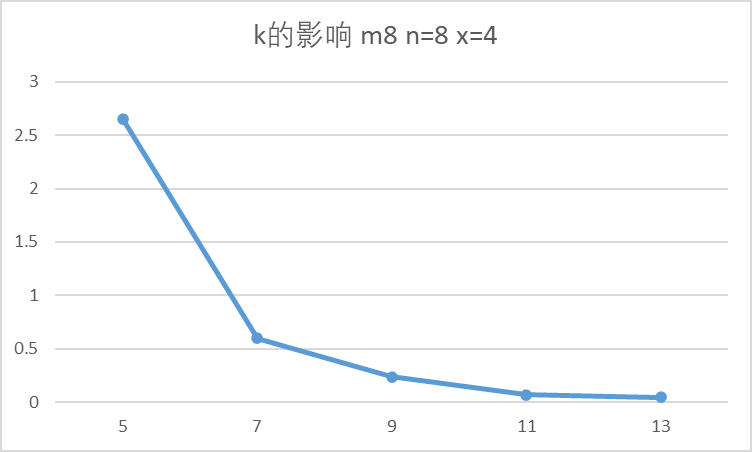




总结：可以看到回溯法用时最长，记忆化有效缩短了时间，而剪枝过后时间大大短于回溯法，其中只保留最大的前1个分支，即topk=1的剪枝回溯法（其实退化为贪心法了）耗时最短，只保留最大的前7个分支，即topk=7的剪枝回溯法，用时最长，且**所有剪枝回溯法的用时都与topk成正相关，topk越大，保留分支越多，用时越长，这也符合我们的认知**

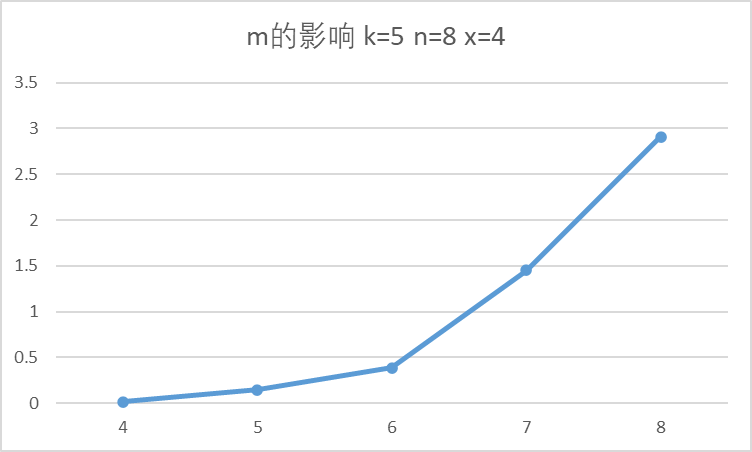
**回溯法：不同参数规模的影响**





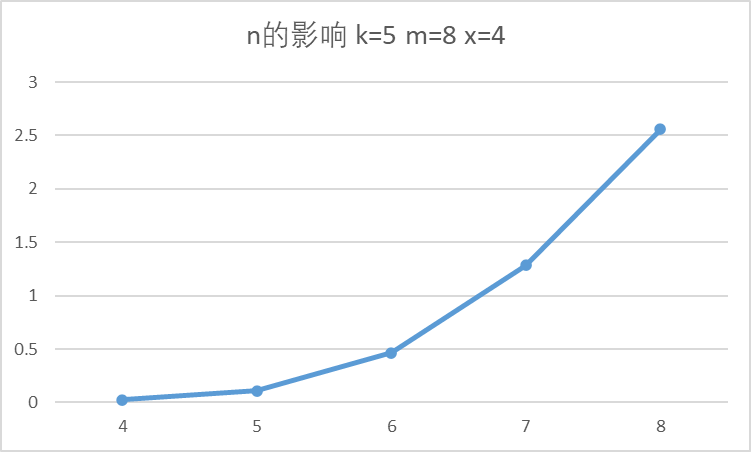
可以看到随着k（方块种类）的增加，时间消耗逐渐减小，因为**大的k提供了更少的消除可能性**





可以看到随着m（行数）的增加，时间开销逐渐增大，因为更大的m，使得每一个棋盘的情况变多，使得搜索的分支数增加



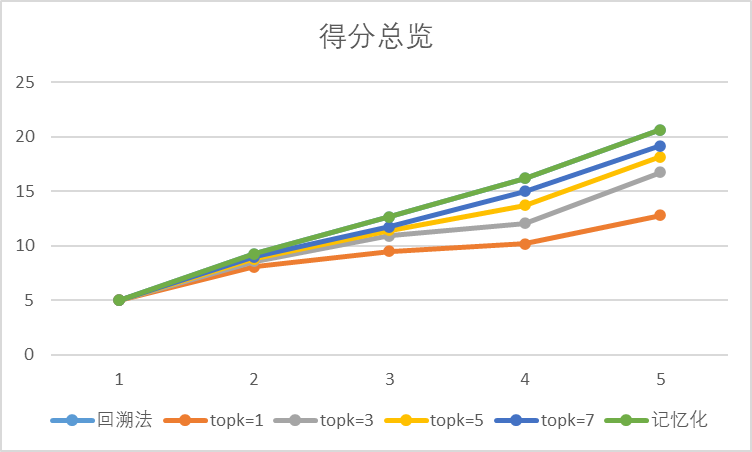


可以看到随着n（列数）的增加，时间开销逐渐增大，因为更大的n，使得每一个棋盘的情况变多，使得搜索的分支数增加，但是与行数m对比，n对时间开销的改变没有m那么大，因为m更多的决定了消去的可能（消去方块后，沿着行方向掉落）

**回溯法与剪枝回溯法：得分测试**

测试规模：k=6 m=8 n=8 x=1,2,3,4,5，横坐标操作次数，纵坐标得分





总结：可以看到，回溯法和记忆回溯法的得分总是最大的，因为回溯法是全局最优解

而剪枝回溯法，随着保留的分支数目的增加，**即随着topk的增加，得分逐渐逼近回溯法的得分（上界）**，这表明保留的分支越多，越容易逼近全局最优解，这也符合我们的认知

**时间与得分 分析：**

可以看到保留7或5个分支的剪枝回溯法最能够逼近最优解，而且耗时相当少，而且在可以接受的程度

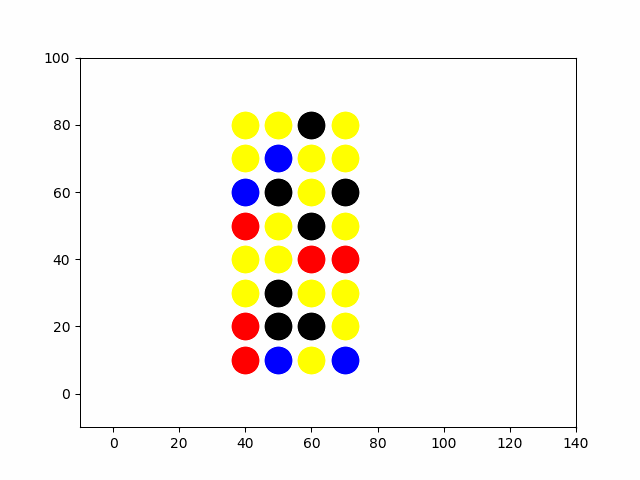
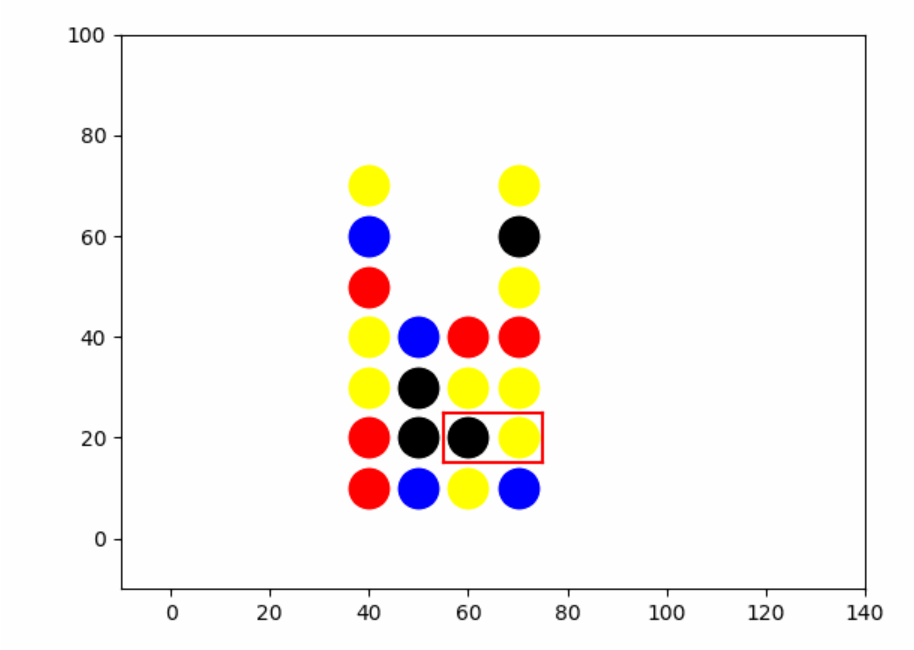
而保留3个分支的剪枝回溯法在得分上稍显劣势，但是运行速度比5或7的剪枝快很多

而保留1个分支（即贪心法）是最快的，但是得出的得分却很低，**因为一个具有丰富技巧的消消乐选手往往需要通过舍弃局部最优，来为后续的消去拼凑更优的解**，这和贪心的“无后效性”矛盾，自然不能用贪心法得出最优解

带记忆化的回溯法可以有效缩短时间，但是在面对大规模问题，仍然具有相当的时间复杂度

图形演示：

由于word无法展示动图，所以老师请关注我的实验答辩及ppt，或者是移步到我的GitHub上查看详细的演示<https://github.com/AKGWSB/grapic-demo-of-XiaoXiaoLe-algorithm>



1. 对求解这个问题的经验总结

记忆化的回溯法能够有效减少时间开销，而且节省的时间开销随着规模的扩大而增加，但是在小规模问题时，因为查询和保存操作需要一定的时间，所以会略慢于回溯法，而查询和插入的过程可以通过更好的哈希函数来解决

消去方块后可能形成新的可以消去的方块，这点在计算得分的时候及其容易出错，消去方块一定要保证消到无可在消，这个过程可以用递归进行

因为回溯法是递归的，一旦有错误满盘皆错，所以在递归之前，应该改好一次回溯法的bug，否则调试起来非常困难

回溯法一定要通过小规模样例，打印回溯过程，以及结果，以确保正确性

因为for循环ij从小到大，坐标系是 ↓→ 而不是↑→，在debug的时候要注意

（此处写你的过程，比如遇到的错误，以及解决方法，你的所想、所得）

|  |
| --- |
| 指导教师批阅意见：  成绩评定：  指导教师签字：  年 月 日 |
| 备注： |

注：1、报告内的项目或内容设置，可根据实际情况加以调整和补充。

2、教师批改学生实验报告时间应在学生提交实验报告时间后10日内。