暨南大学本科实验报告专用纸

课程名称 算法分析与设计实验 成绩评定

实验项目名称 N后问题 指导教师 李展

实验项目编号 实验八 实验项目类型 综合性 实验地点

学生姓名 张印祺 学号 2018051948

学院 信息科学技术 系 计算机科学 专业 网络工程

实验时间 2020 年 5 月 20 日 ~ 5 月 20 日下午温度 ℃湿度

一、问题描述

在 n×n 棋盘上放彼此不受攻击的 n 个皇后。按国际象棋规则，皇后可攻击同行、同列、同一斜线的棋子。等价于在 n×n 格的棋盘上放置 n 个皇后，任何 2 个皇后不放在同一行或同一列或同一斜线上。

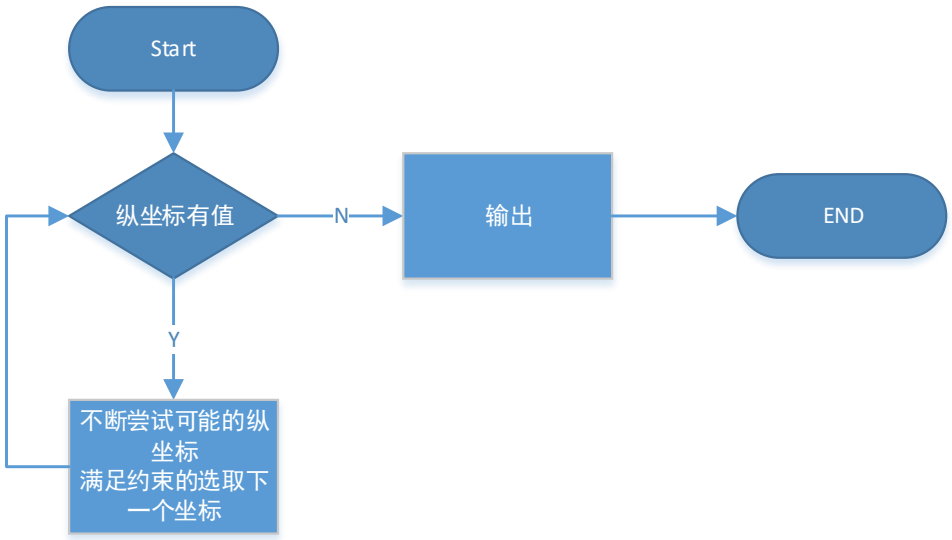
二、算法思路

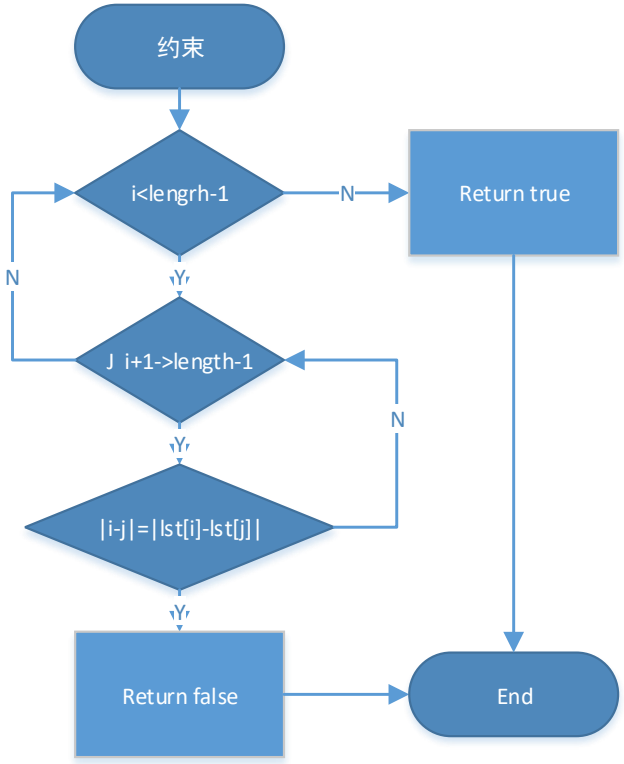
这个问题也可以描述为 n 个数字的排列问题，约束条件为

$$|k - j|! = |x[j] - x[k]|$$

这样该问题就可以用全排列的算法求解。

三、流程图


Start
纵坐标有值
N
输出
END
Y
不断尝试可能的纵坐标
满足约束的选取下一个坐标


约束
i<lengrh-1
N
Return true
N
Y
J i+1->length-1
N
Y
|i-j|=|lst[i]-lst[j]|
Y
Return false
End

四、测试结果

```
J = 5
Nums = [i for i in range(J)]
backtracing(J, Nums, [])
```

```
0 0
1 2
2 4
3 1
4 3

0 0
1 3
2 1
3 4
4 2

0 1
1 3
2 0
3 2
4 4

0 1
1 4
2 2
3 0
4 3
```

```
N = 4
Nums = [i for i in range(N)]
backtracing(N, Nums, [])
```

```
0 1
1 3
2 0
3 2

0 2
1 0
2 3
3 1
```

五、实验总结

如果不进行任何剪枝，其时间复杂度就是 $O(N) = O(n^n)$。因为 N 行 N 列，皇后的排列方式共有 N^N 种。如果进行优化，会将最坏时间复杂度优化至 $N!$ 。

六、源代码

```
def constraint(lst):
    length = len(lst)
    for i in range(length-1):
        for j in range(i+1, length):
            if abs(i-j) == abs(lst[i]-lst[j]):
                return False
    return True
def backtracing(n, nums, lst):
    if len(nums) == 0:  # 当纵坐标没有值时，输出
        for x, y in enumerate(lst):
            print(x, y)
        print()
    else:
        for i in nums:  # 不断尝试可能的纵坐标
            temo_lst = [x for x in lst]
            temo_lst.append(i)
            if constraint(temo_lst):  # 满足约束的选取下一个坐标
                temp_nums = [e for e in nums if e not in temo_lst]
                backtracing(n, temp_nums, temo_lst)
```