中華大學生物資訊學系系統開發專題報告

解謎程式與活用

Puzzle program and use

專題英文名稱

專題組員: 楊宇傑、陳昱蓁、劉冠廷

專題編號：104011

指導老師:侯玉松老師

1. 摘要

此次專題用電腦程式 C++代替人工嘗試解出誰養魚，既減少解謎所花費的時間，更可以確保答案的正確率。而這次C++程式中，我們運用了表示法,深度優先搜尋法和限制式這三個回溯法的重大要素來推出解謎的程式碼。

利用 C++程式執行誰養魚，比起人工下去計算要來的快速許多，再者，我們運用此程式碼增加遊戲玩法，些微的改寫程式碼讓題目條件改變甚至多變，比如題目可以因為省略其中一個條件而增加謎題難度，以及更改條件可以創造出不同謎題，再讓玩家以人工方式解題，看誰能最快解出謎題。

2. 簡介

之所以選擇於誰養魚的題目，是因為當時老師介紹一些需要寫程式的題目，然後我們就對誰養魚這個題目很感興趣,所以就選擇這個題目去做。

因為這個謎題世界上有 98%的人回答不出來，所以我們選擇對大學生不會太困難卻也能夠讓其動動頭腦的謎題誰養魚。

本想以此謎題做為教學用，沒想到解了三十幾分鐘卻還沒解出來，經過跟同學以及老師的討論我們試了許多方法，最後我們以演算法來衍生出此程式碼，而在數位化之前，我們也嘗試過許多的解法，例如:拼圖法，最後再運用電腦程式來找到最佳或最快速的方法。

雖然過程中並不比人工法輕鬆，可這方法不但可以學習演算法，並且還可以學習如何用演算法來推出其他程式碼，再者還可以運用程式碼的多變性來產生不同難度以及不同條件的謎題。

3. 專題進行方式

我們選擇的謎題為愛因斯坦謎題，其內容為:【有五間房屋排成一列、所有的房屋外表顏色都不一樣，所有的屋主都來自不同國家、養不同的寵物、喝不同的飲料、跟抽不同牌的香煙

1 英國人住在紅色房屋裡
2 瑞典人養了一隻狗
3 丹麥人喝茶
4 綠色的房子在白色房子的左邊
5 綠色房屋的屋主喝咖啡
6 抽 Pall Mall 香煙的屋主養鳥
7 黃色屋主抽 Dunhill
8 位於最中間的屋主喝牛奶
9 挪威人住在第一間房屋裡
10 抽 Blend 的人住在養貓人家的隔壁
11 養斑馬的屋主隔壁住抽 Dunhill 的

人家
12 抽 Blue Master 的屋主他喝啤酒
13 德國人他抽 Prince
14 挪威人住在藍色房子隔壁
15 只喝開水的人家住在抽 Blend 的隔壁
請問：誰養魚?】

（一）回溯法

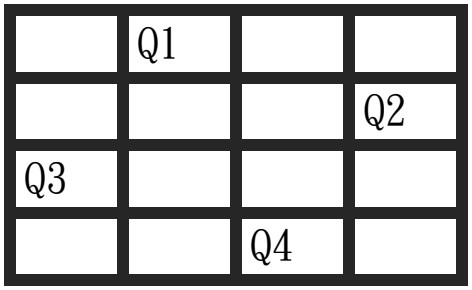
回溯法較常被討論的例子是 N 后問題(N-Queen Problem)，是否能在一個 N×N 的西洋棋盤上放置 N 個皇后而每一個皇后的直線、橫線和對角線不存在著其他皇后 ， 下面用四后問題當做例子，(四后問題就是一個西洋棋遊戲的概念， 直線、橫線和對角線不存在著其他皇后，不然會被其他的皇后吃掉)。

- 回溯法之三大要素分別為：
- 表示法
- 深度優先搜尋法
- 限制式

利用以上 3 個要素來解四后問題。

- 表示法:

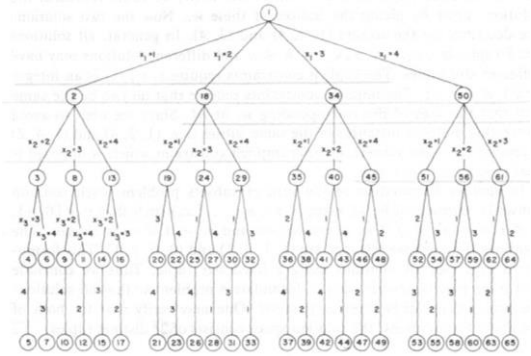
利用表示法去解四后問題，每一橫排只放一個皇后，第一個皇后放第 0 列，第二個放第 1 列，第三個放第 2 列，第四個放第 3 列，會得到一個(n1,n2,n3,n4)的陣列，這個陣列的意思是第 i 個皇后放在第 ni 列。以圖三為例，Q1 列的皇后放在第 1 行，所以 n1 為 1，Q2 列的皇后放在第 3 行，所以 n2 為 3，Q3 列的皇后放在第 0 行，所以 n3 為 0，Q4 列的皇后放在第 2 行，所以 n4 為 2，以陣列表示為(1,3,0,2)。【下圖】陣列表示(1,3,0,2)所對應四后問題



- 深度優先搜尋法:

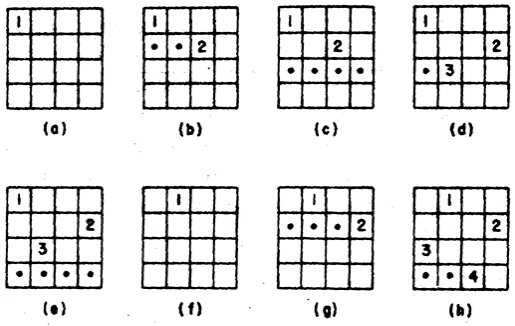
深度優先搜尋(Depth First Search)。如果以走路來比喻，我們要走路到一個指定地點，有著許多岔路，而深度優先搜尋指的就是一直往前走，只要前面還有路，就往前，直到沒有路可走為止。之後，退回到最近一個岔路口,在探索另一條岔路，直到到達指定地點。

將每一種可能的放法全部列出，再去檢查是否有互相干擾。以此類推 ，四后就有 4!，共 24 種可能解，N 后就有有 N! 的放置方式，如果將所有的可能解畫成樹狀圖。

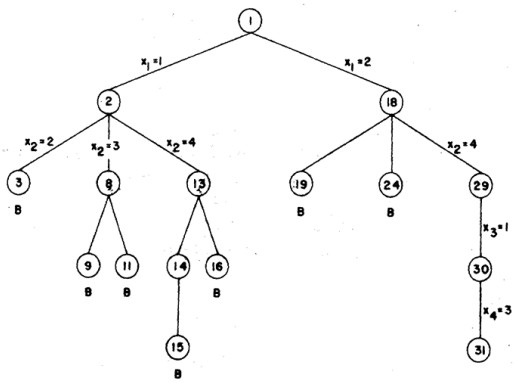


➢ 限制式:

檢查在同一行、同一列、斜對角均不可同時存在兩個皇后。



四后問題，利用回溯法來找出第一組解法。



四后問題加入限制式樹狀圖

以上兩圖互相對應，上圖樹狀圖之數字代表著限制式矩陣圖的順序，加入限制式後，我們可以發現深先法樹狀圖及限制式樹狀圖相比較下，限制式樹狀圖的枝葉較少，步驟也較少，能有效幫助我們找到解答。

● 將解謎程式套入回朔法

將解謎程式套入回溯法之三大要素:

(1)表示法

用來定位 K 在二維陣列上的位置

EX：(K/5) 0/5=0(商)→第 0 排

(K%5) 0%5=0(餘)→第 0 列

	J=0	J=1	J=2	J=3	J=4
I=0	0	1	2	3	4
I=1	5	6	7	8	9
I=2	10	11	12	13	14
I=3	15	16	17	18	19
I=4	20	21	22	23	24

(2)深先搜尋法

找出養魚的是第 j 行, 然後印出屋主的國籍 x[1][j].

再運用雙重迴圈，列印 5*5 的 x[0~4][0~4]的解內容.

如果與限制式的條件不符合，就會回傳 0 表示和事實不符【return(0)】，確定不符合事實的條件都排除後，最下面 return(1)，然後回到 Logic_game 填值 x[k/5][k%5] = i;

(3)限制式

```
int j, r, c; // r: row, c: column

count++;
r = k/5;
c = k%5;
for (j=0; j<c; j++)
        if (x[r][j] == i) return(0); //同列數字不得相同
if (r==1 && i==0 && x[0][c] != 0) return(0); //英國人住紅色房
if (r==2 && i==1 && x[1][c] != 1) return(0); //瑞典人養狗
if (r==3 && i==0 && x[1][c] != 2) return(0); //丹麥人喝茶
if (r==0 && i==2 && c==0) return(0); //白色屋在最左邊
if (r==0 && i==2 && x[0][c-1] != 1) return(0); //綠屋在白屋左邊
if (r==3 && i==1 && x[0][c] != 1) return(0); //綠屋主喝咖啡
if (r==4 && i==0 && x[2][c] != 2) return(0); //抽 Pall 屋主養鳥
if (r==4 && i==1 && x[0][c] != 3) return(0); //黃屋主抽 Dunhill
if (r==3 && i==2 && c != 2) return(0); //中間屋主喝牛奶
if (r==1 && i==3 && c != 0) return(0); //挪威人住第一間房
if (r==4 && i==2 && c==0 && x[2][1] != 3) return(0); //抽Blend住最左, 右鄰養貓
if (r==4 && i==2 && c==4 && x[2][3] != 3) return(0); //抽Blend住最右, 左鄰養貓
if (r==4 && i==2 && c>0 && c<4 && x[2][c-1] != 3 && x[2][c+1] != 3) return(0);
if (r==4 && i==1 && c==0 && x[2][1] != 4) return(0); //抽Dunhill住最左, 右鄰養馬
if (r==4 && i==1 && c==4 && x[2][3] != 4) return(0); //抽Dunhill住最右, 左鄰養馬
if (r==4 && i==1 && c>0 && c<4 && x[2][c-1] != 4 && x[2][c+1] !=4) return(0);
if (r==4 && i==3 && x[3][c] != 3) return(0); //抽BlueMaster屋主喝啤酒
if (r==4 && i==4 && x[1][c] != 4) return(0); //德國屋主抽Prince
if (r==1 && i==3 && c==0 && x[0][1] != 4) return(0); //挪威屋主住最左, 右屋藍色
if (r==1 && i==3 && c==4 && x[0][3] != 4) return(0); //挪威屋主住最右, 左屋藍色
if (r==1 && i==3 && c>0 && c<4 && x[0][c-1] != 4 && x[0][c+1] !=4) return(0);
if (r==4 && i==2 && c==0 && x[3][1] != 4) return(0); //抽Blend住最左, 右鄰喝水
if (r==4 && i==2 && c==4 && x[3][3] != 4) return(0); //抽Blend住最右, 左鄰喝水
if (r==4 && i==2 && c>0 && c<4 && x[3][c-1] != 4 && x[3][c+1] !=4) return(0);
return(1);
```

● 改寫程式碼

(1)改變條件 得出一個新的答案

```
if (r==4 && i==1 && x[3][c] != 3) return(0); //抽Dunhill層卡喝啤酒
if (r==4 && i==4 && x[1][c] != 4) return(0); //德國層卡抽Prince
if (r==1 && i==3 && c==0 && x[0][1] != 4) return(0); //挪威層卡住最
```

```
ans: 1
0 : 3 4 0 1 2
1 : 3 2 0 1 4
2 : 1 4 2 0 3
3 : 3 0 2 1 4
4 : 1 3 0 2 4
count = 33990
```

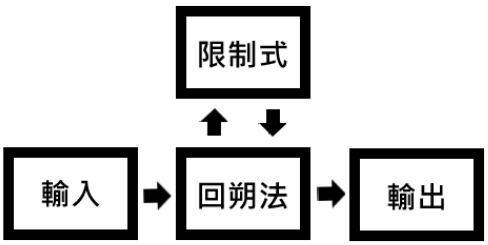
(2)省略特定條件 增加題目難度 也得出唯一答案

```
    if (r==4 && i==2 && c==0 && x[2][1] != 3) return(0); //抽Blend住最左, 右鄰實貓
    if (r==4 && i==2 && c==4 && x[2][3] != 3) return(0); //抽Blend住最右, 左鄰實貓
    if (r==4 && i==2 && c>0 && c<4 && x[2][c-1] != 3 && x[2][c+1] !=3) return(0);
    if (r==4 && i==1 && c==0 && x[2][1] != 4) return(0); //抽Dunhill住最左, 右鄰實馬
    if (r==4 && i==1 && c==4 && x[2][3] != 4) return(0); //抽Dunhill住最右, 左鄰實馬
    if (r==4 && i==1 && c>0 && c<4 && x[2][c-1] != 4 && x[2][c+1] !=4) return(0);
    if (r==4 && i==3 && x[3][c] != 3) return(0); //抽BlueMaster層卡喝啤酒
    if (r==4 && i==4 && x[1][c] != 4) return(0); //德國層卡抽Prince
    if (r==1 && i==3 && c==0 && x[0][1] != 4) return(0); //挪威層卡住最左, 右層藍色
    if (r==1 && i==3 && c==4 && x[0][3] != 4) return(0); //挪威層卡住最右, 左層藍色
    if (r==1 && i==3 && c>0 && c<4 && x[0][c-1] != 4 && x[0][c+1] !=4) return(0);
//  if (r==4 && i==2 && c==0 && x[3][1] != 4) return(0); //抽Blend住最左, 右鄰喝水
//  if (r==4 && i==2 && c==4 && x[3][3] != 4) return(0); //抽Blend住最右, 左鄰喝水
//  if (r==4 && i==2 && c>0 && c<4 && x[3][c-1] != 4 && x[3][c+1] !=4) return(0);
    return(1);
```

```
ans: 4
0 : 3 4 0 1 2
1 : 3 2 0 4 1
2 : 3 4 2 0 1
3 : 4 0 2 1 3
4 : 1 2 0 4 3
count = 33360
```

● 系統分析

系統架構圖:



(1) 輸入:把謎題輸入我們設計出的程式中。

(2) 回溯法:將 k 帶入限制式，視其是否符和事實。

(3) 限制式:呼叫限制式,檢查是否符合條件。

(4) 輸出:跑出代表”魚”的數字，即可得知其謎底。

(5) 改寫程式碼，進而增加題目多樣性。

● 時程與人員配置

本專題進度執行表、工作分配表如下

職責分配	討論主題	蒐集資料	電腦程式製作	檢查程式	報告製作	PPT製作	海報製作	總整理
劉冠廷	◎	◎	◎		◎	◎		◎
楊宇傑	◎	◎	◎		◎		◎	◎
陳昱蓁	◎		◎	◎	◎	◎		◎

進度	討論主題	蒐集資料	電腦程式製作	檢查程式	報告製作	PPT製作	海報製作	總整理
3月								
4月								
5月								
6月								
7月								
8月								

9月	◎	◎						
10月			◎	◎				
11月				◎		◎	◎	
12月				◎	◎	◎	◎	◎

- 實作平臺

RAM:8.00 GB

處理器名稱:

Inter(R)Core(TM)i5-4590

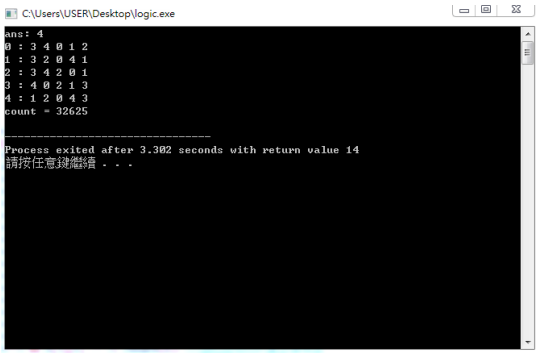
CPU@ 3.30GHz

作業系統:使用 Windows7 64 位元。

技術:logic.c-Dev-C++5.11

- 測試方式:

依題目寫出指令,得出最終答案。



- 困難與解決之道

Q:我們如何把一道謎題程式化?

A:經過我們與老師的討論,最後決定用遞迴關係式。

Q:寫程式的過程有不同的意見紛歧?

A:能夠吸取各自的優缺點,把程式變得更好。

4. 主要成果

```
// 題目的參數: 房子顏色, 屋主國籍, 飼養寵物, 飲料選擇, 香煙品牌
// 0. 房子顏色: (0) 紅色  (1) 綠色  (2) 白色  (3) 黃色  (4) 藍色
// 1. 屋主國籍: (0) 英國  (1) 瑞典  (2) 丹麥  (3) 挪威  (4) 德國
// 2. 飼養寵物: (0) 魚  (1) 狗  (2) 鳥  (3) 貓  (4) 馬
// 3. 飲料選擇: (0) 茶  (1) 咖啡  (2) 牛奶  (3) 啤酒  (4) 白開水
// 4. 香煙品牌: (0) Pall  (1) Dunhill  (2) Blend  (3) BlueMaster (4) Prince
```

```
#include <stdio.h>
int x[5][5];
int count = 0;
//int found = 0;
int place(int k, int i)
{
    int j, r, c; // r: row, c: column

    count++;
    r = k/5;
    c = k%5;
    for (j=0; j<c; j++)
        if (x[r][j] == i) return(0);
    if (r==1 && i==0 && x[0][c] != 0) return(0);
    if (r==2 && i==0 && x[1][c] != 1) return(0);
    if (r==3 && i==0 && x[1][c] != 2) return(0);
    if (r==0 && i==2 && c==0) return(0);
    if (r==0 && i==2 && x[0][c-1] != 1) return(0);
    if (r==3 && i==1 && x[0][c] != 1) return(0);
    if (r==4 && i==0 && x[2][c] != 2) return(0);
    if (r==4 && i==1 && x[0][c] != 3) return(0);
    if (r==3 && i==2 && c != 2) return(0);
    if (r==1 && i==3 && c != 0) return(0);
    if (r==4 && i==2 && c==0 && x[2][1] != 3) return(0);
    if (r==4 && i==2 && c==4 && x[2][3] != 3) return(0);
    if (r==4 && i==2 && c>0 && c<4 && x[2][c-1] != 3 && x[2][c+1] !=3) return(0);
    if (r==4 && i==1 && c==0 && x[2][1] != 4) return(0);
    if (r==4 && i==1 && c==4 && x[2][3] != 4) return(0);
    if (r==4 && i==1 && c>0 && c<4 && x[2][c-1] != 4 && x[2][c+1] !=4) return(0);
    if (r==4 && i==3 && x[3][c] != 3) return(0);
    if (r==4 && i==4 && x[1][c] != 4) return(0);
    if (r==1 && i==3 && c==0 && x[0][1] != 4) return(0);
    if (r==1 && i==3 && c==4 && x[0][3] != 4) return(0);
    if (r==1 && i==3 && c>0 && c<4 && x[0][c-1] != 4 && x[0][c+1] !=4) return(0);
    if (r==4 && i==2 && c==0 && x[3][1] != 4) return(0);
    if (r==4 && i==2 && c==4 && x[3][3] != 4) return(0);
    if (r==4 && i==2 && c>0 && c<4 && x[3][c-1] != 4 && x[3][c+1] !=4) return(0);
    return(1);
}
logic_game(int k, int n)
{
    int i, j, j2;

    for (i=0; i<5; i++)
        if (place(k, i))
```

```
logic_game(int k, int n)
{
    int i, j, j2;

    for (i=0; i<5; i++)
        if (place(k, i))
        {
            x[k/5][k%5] = i;
            if (k==n)
            {
                for (j=0; j<5; j++)
                    if (x[2][j] == 0) break;
                printf("ans: %d\n", x[1][j]);
                for (j=0; j<5; j++)
                {
                    printf("%d :", j);
                    for (j2=0; j2<5; j2++)
                        printf(" %d", x[j][j2]);
                    printf("\n");
                }
                //found = 1;
            }
            else logic_game(k+1, n);
            //if (found) break;
        }
}
main()
{
    logic_game(0, 24);
    printf("count = %d\n", count);
}
```

新謎題:

5. 評估與展望

希望我們的解謎程式可以解出全世界現有的所有同類型題目,甚至將高難度的謎題寫得更加困難,讓大家都發現解謎程式的有趣。

6. 結語

這次作品我們運用的是回溯法來解題，讓我們找到謎題的解法，以便提供更多想知道謎題解法的人。藉由此謎題也能讓我們知道回溯法的多種用途，對程式的了解也更深一步，期許將來能夠利用回溯法來衍生出謎題的不同變化。

7. 銘謝

感謝**侯玉松**老師在我們製作專題時不斷的指導，而且老師在百忙之中還是不厭其煩的空出時間與我們討論專題，並告訴我們哪些地方需要改進，也讓我們自己去思考如何寫出一個完整的程式，藉由我們小組成員中互相討論，找出解決方法，可以從討論之中獲得更多知識。也很感謝組員互相合作，不斷的一直嘗試、從錯誤中學習，讓我們了解分工合作的重要性，所以讓我們專題可以事半功倍的順利完成。

8. 參考文獻

http://blog.udn.com/Mathplayer/260357

https://en.m.wikipedia.org/wiki/Zebra_Puzzle

https://www.youtube.com/watch?v=1rDVz_Fb6HQ&feature=youtu.be