**資料結構報告**

張凱茗

2024/10/18

**目錄**

1  **解題說明** P2

2 **演算法設計與實作**  P3

3  **效能分析** P4

4 **測試與過程**  P5

**解題說明**

Problem 1:根據題目所給的遞迴公式實作出遞迴函式(圖1)，並將其轉為非遞迴函式(圖2)

Problem 2:列出輸入的元素所有可能的子集合，並用二進制來控制元素是否輸出(圖3)

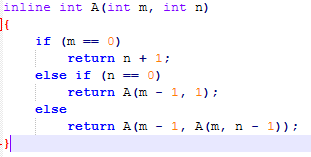


圖 1

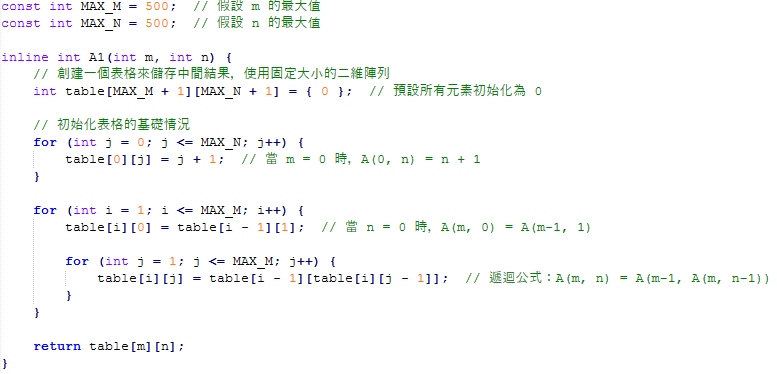


圖 2

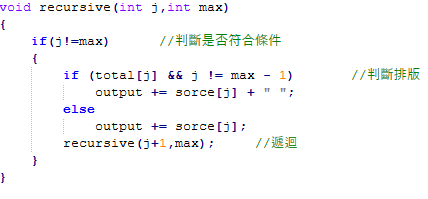


圖 3

**演算法設計與實作**

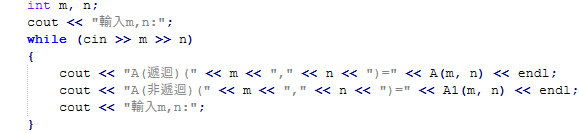
Problem 1: 如圖4

圖 4

Problem 2:如圖5

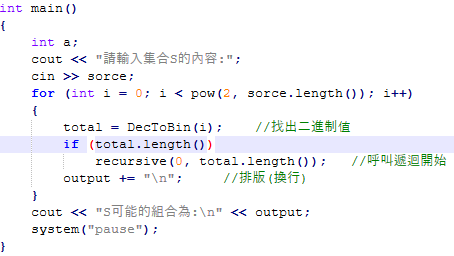


圖 5

**效能分析**

Problem 1:

遞迴版本 A(m, n)：

**時間複雜度**：

當 m == 0 時，函數直接返回 n + 1，時間複雜度為 O(1)O(1)O(1)。

當 m > 0 且 n == 0 時，會調用 A(m - 1, 1)，這樣會簡化 m 的值，但遞迴深度依然隨著 m 增加。

m > 0 且 n > 0 時，會進行兩次遞迴。對於大的 m 和 n，遞迴次數和深度會變得非常大。

**空間複雜度**：最壞情況下，遞迴深度等於函數的計算深度，因此空間複雜度是 **超指數級**，取決於 m 和 n 的值。

動態規劃版本 A1(m, n)：

**時間複雜度：** O(m×n) (在此處為 O(500×500))

**空間複雜度：** O(m×n)

Problem 2:

**時間複雜度**： 主要的迴圈遍歷了次，因為每個元素都有兩種狀態。在每次迴圈中，會調用 DecToBin(i)（時間複雜度為O(∣sorce∣) )和 recursive(0, total.length())（時間複雜度也為 O(∣sorce∣)）。因此，整個迴圈的總時間複雜度為：O()

**空間複雜度**： 主要的空間使用來自於兩個部分：

1. 用來儲存所有結果的output，大小為O()。
2. 每次遞迴過程中的棧空間，深度為sorce.length()

因此，空間複雜度S(P)為O(+sorce)。

**測試與過程**

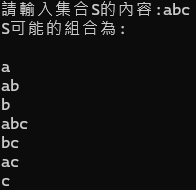
Problem 1:



驗證

A(1,**1**)→A(0,**A(1,0)**) →A(0,**A(0,1)**) →A(0,**2**) →3

Problem 2:



驗證

S={a,b,c}，可能的子集合有{}{a}{b}{c}{ab}{ac}{abc}{bc}