資料結構作業二 - Find Maximum Using Binary Tree

|  |  |  |  |
| --- | --- | --- | --- |
| 科系 | 學號 | 姓名 | 日期 |
| 電機系大二 | E24099059 | 陳旭祺 | 2021/05/23 |

* 程式文字說明(實際範例)

|  |  |  |
| --- | --- | --- |
| Output(紅字為使用者輸入值) | | |
| CMD Window on Win10 OS | | 功能解釋 |
| type data number:  7 | | 輸入有n個數字要找其中的最大值 |
| each element of array generated from time seed :  number\_array[0] = 58  number\_array[1] = 1  number\_array[2] = 36  number\_array[3] = 27  number\_array[4] = 50  number\_array[5] = 87  number\_array[6] = 24 | | 從時間種子裡面取亂數，並限定data範圍在 |
| type search data array:  1 4 | | 輸入要比較的區間 |
| The structure of tree:  tree[0][0] = 87  tree[0][1] = 58  tree[0][2] = 87  tree[0][3] = 58  tree[0][4] = 36  tree[0][5] = 87  tree[0][6] = 24  tree[0][7] = 58  tree[0][8] = 1  tree[0][9] = 36  tree[0][10] = 27  tree[0][11] = 50  tree[0][12] = 87  tree[0][13] = 24  tree[0][14] = 0 | 考量到使用linked list會造成pointer額外空間的付出成本，故使用**3維陣列建立二元樹**，**row=0存的是data**，對應咖啡色為terminal node由上一步驟number\_array複製過來，但考慮到陣列n大小可能不是 ，因此會擴充terminal node為最接近的 ，擴充data為一開始已初始化的0，即tree[0][14] = 0。  而藍色為terminal node的上層，故tree[0][6] = 24 = max(tree[0][13] = 24, tree[0][14] = 0)，就是找出兩者之間的最大值，如此iteration做到root為止 | |
| tree[1][0] = 0  tree[1][1] = 0  tree[1][2] = 4  tree[1][3] = 0  tree[1][4] = 2  tree[1][5] = 4  tree[1][6] = 6  tree[1][7] = 0  tree[1][8] = 1  tree[1][9] = 2  tree[1][10] = 3  tree[1][11] = 4  tree[1][12] = 5  tree[1][13] = 6  tree[1][14] = 7 | **row=1存的是下界(lower)**，  當前節點的下界=左下節點的下界，  如此iteration做到root為止 | |
| tree[2][0] = 7  tree[2][1] = 3  tree[2][2] = 7  tree[2][3] = 1  tree[2][4] = 3  tree[2][5] = 5  tree[2][6] = 7  tree[2][7] = 0  tree[2][8] = 1  tree[2][9] = 2  tree[2][10] = 3  tree[2][11] = 4  tree[2][12] = 5  tree[2][13] = 6  tree[2][14] = 7 | **row=2存的是上界(upper)**，  當前節點的上界=右下節點的上界，  如此iteration做到root為止 | |

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Step to find candidate node with the maximum value:  split to two part  first part: lower = 1, upper = 3, current node = 1  second part: lower = 4, upper = 4, current node = 2  split to two part  first part: lower = 1, upper = 1, current node = 3  second part: lower = 2, upper = 3, current node = 4  lower = 1, upper = 1, current node = 3 go to right tree  lower = 1, upper = 1, current node = 8 hit the range  lower = 2, upper = 3, current node = 4 hit the range  lower = 4, upper = 4, current node = 2 go to left tree  lower = 4, upper = 4, current node = 5 go to left tree  lower = 4, upper = 4, current node = 11 hit the range | 為找候選節點的詳細流程，我的程式共分4種狀況   |  | | --- | | 1.range介在兩者之間 | | Split to 2 parts | | 2.upper<=右下  孩子節點的上界 | | go to left tree | | 3.lower>右下  孩子節點的上界 | | go to right tree | | 4.上下界皆符合 | | hit the range  Push in Stack |   採用recursive call的寫法，因為在上述情況一range介在兩者之間，會”增生”。另外因為不知道有幾個候選節點，所以用stack去存。 |
| choosing point 1 = 1  choosing point 2 = 36  choosing point 3 = 50 | 用1個array和1個指標做出stack容器去存候選節點的值 |
| in number\_array[1] ~ number\_array[4], max = 50 | 用for loop把候選節點pop出來，由於數量較少，故直接用一般的方式比大小 |

* 流程圖演示(實際範例)

|  |
| --- |
| 從時間種子裡面取亂數與給定陣列大小n得a[7] |
|  |

|  |
| --- |
| 所有元素初始化，歸零(build\_tree subfunction) |
|  |
| 從array複製data到terminal nodes (build\_tree subfunction) |
|  |

|  |
| --- |
| 在terminal nodes填入lower與upper(build\_tree subfunction) |
|  |
| |  |  | | --- | --- | | Row == 0 | 找出兩者之間的最大值 | | Row == 1 | 當前節點的下界 = 左下節點的下界 | | Row == 2 | 當前節點的上界 = 右下節點的上界 |   如此iteration，做到root為止(build\_tree subfunction) |
|  |
| split to two part(max\_candid\_f subfunction)  first part: lower = 1, upper = 3, current node = 1  second part: lower = 4, upper = 4, current node = 2 |
|  |
| split to two part(max\_candid\_f subfunction)  first part: lower = 1, upper = 1, current node = 3  second part: lower = 2, upper = 3, current node = 4 |
|  |

|  |
| --- |
| lower = 1, upper = 1, current node = 3 go to right tree  lower = 1, upper = 1, current node = 8 hit the range  lower = 2, upper = 3, current node = 4 hit the range  (max\_candid\_f subfunction) |
|  |
| lower = 4, upper = 4, current node = 2 go to left tree  lower = 4, upper = 4, current node = 5 go to left tree  lower = 4, upper = 4, current node = 11 hit the range  (max\_candid\_f subfunction) |
|  |

|  |
| --- |
| 用for loop把候選節點從stack裡面pop出來，直接用一般的方式比大小 |
|  |

* 心得與討論

這次程式作業我學到了用Binary Tree找最大值。主要卡在地方在於在函式間傳遞二維以上陣列，在看了以下教學才懂，也算是複習以前所學的，我在指標上應多多練習加強，畢竟這是C語言的基本功。

[](https://www.youtube.com/embed/ZkBZmyk17Q8?feature=oembed)