

期末考實作2

DFS及BFS

林琨硯 | 資料結構 | 2018/12/31

目錄

[前言 2](#_Toc534058647)

[流程圖 3](#_Toc534058648)

[DFS 3](#_Toc534058649)

[BFS 4](#_Toc534058650)

[程式碼解說 5](#_Toc534058651)

[**題目:** 5](#_Toc534058652)

[DFS 6](#_Toc534058653)

[BFS 8](#_Toc534058654)

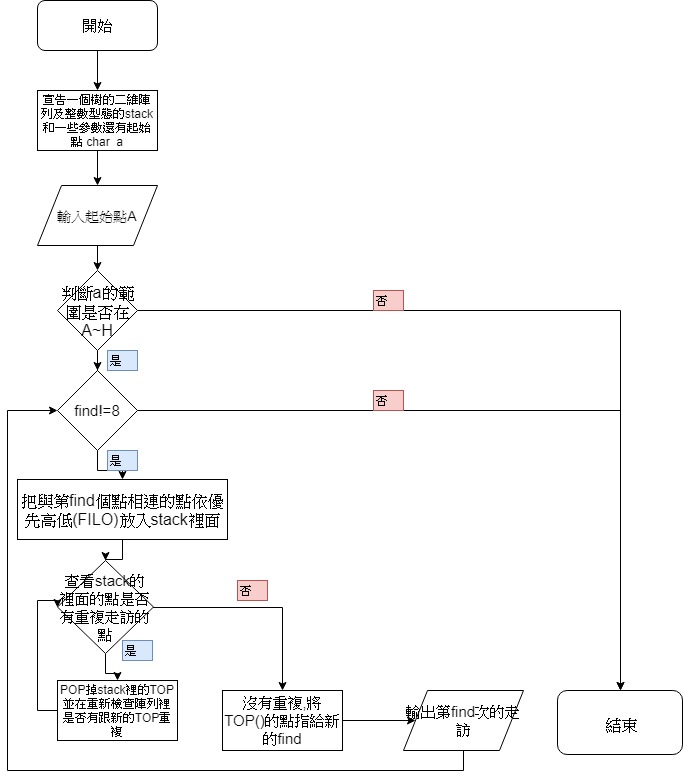
[結論及心得 10](#_Toc534058655)

前言

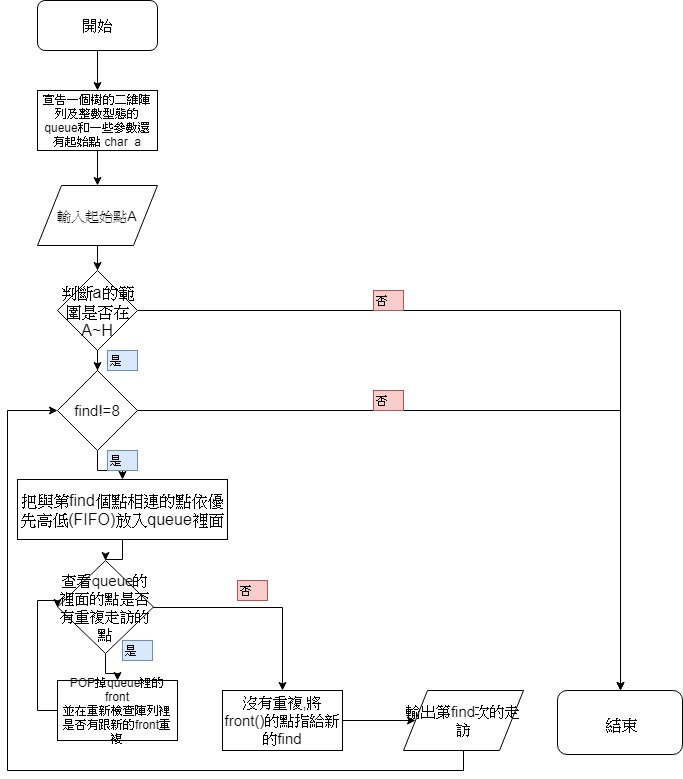
老實說，第一次期末考實作的題目有點難，然而這次的DFS和BFS的程式對於我來說相對簡單，可能是因為老師有給我圖吧。DFS([深度優先搜尋)相對於BFS(廣度優先搜尋)難做。DFS 就是先遇到的vertex就先Visiting，若有兩個點，就以優先權高的先輸出，並且以先遇到的vertex作為新的搜尋起點，直到所有「有edge相連的vertex」都被探索過。而BFS 就是按照level由小到大依序對node進行Visiting。兩者的演算法也不一樣喔。](http://alrightchiu.github.io/SecondRound/graph-depth-first-searchdfsshen-du-you-xian-sou-xun.html)

流程圖

DFS



BFS



程式碼解說

**題目:**

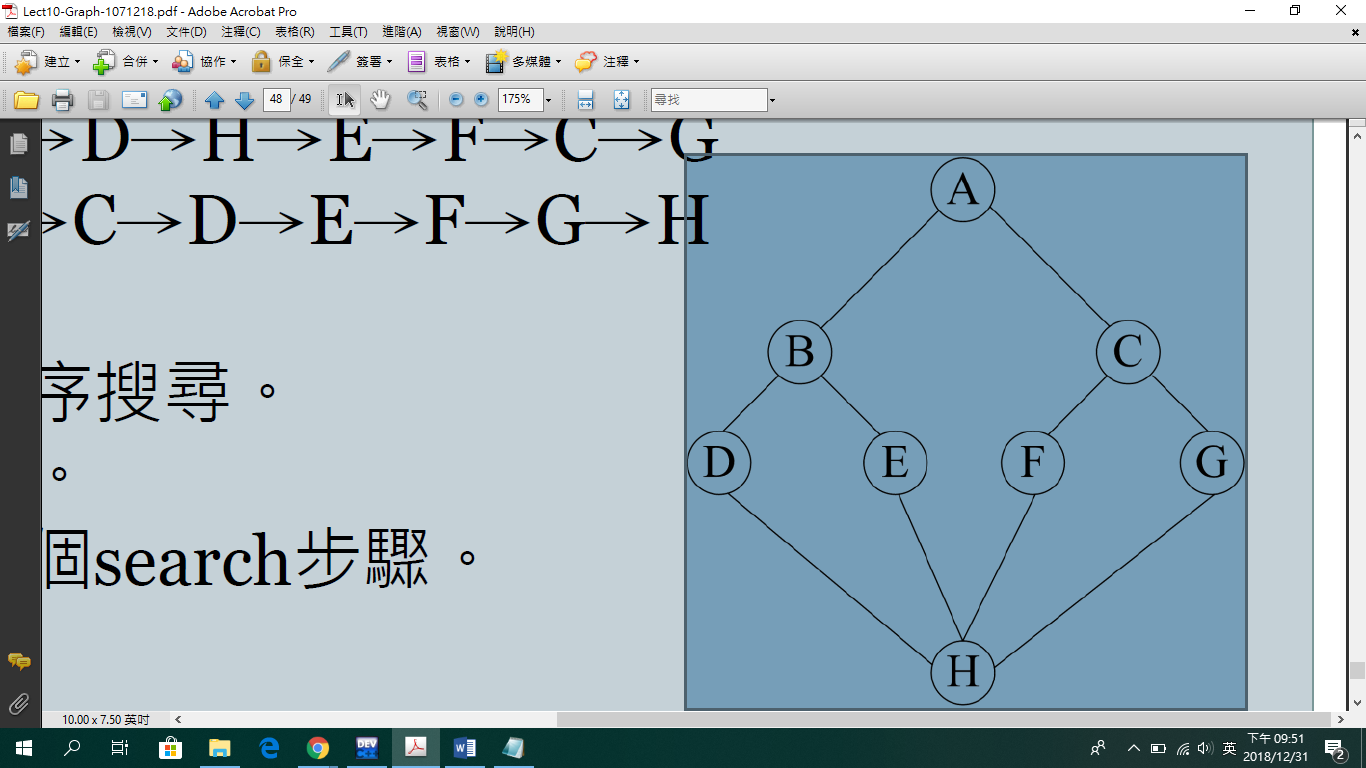
請對下圖進行BFS及DFS並列印結果。

題目如下圖↓

**例(A為起點):**

**DFS:A→B→D→H→E→F→C→G**

**BFS:A→B→C→D→E→F→G→H**



DFS

#include <iostream>

#include <cstdlib>

#include <stdio.h>

#include <stack>

using namespace std;

int main()

{

char a;

// 此處為對TREE 的node 進行處理,使用二維陣列

int a1[8][4]={{1,2,-1,-1},{0,3,4,-1},{0,5,6,-1},{1,7,-1,-1},{1,7,-1,-1},{2,7,-1,-1},{2,7,-1,-1},{3,4,5,6}}

,a2[8]={-1,-1,-1,-1,-1,-1,-1,-1},num,find=1,j=0;

//在此處宣告一個整數型態的stack 對TREE 進行走訪的處理

stack<int>myints;

cout << "This is a DFS search:";

a = getchar();

num = a-65;

a2[0] = num;

cout << "The first:" << char (65+a2[0]) << endl;

while(find != 8)

{

j =0;

for (int i=3;i>=0;i--)

{

if (a1[num][i] != -1)

myints.push(a1[num][i]);

}

/\*判斷是否有重複出現的節點被走訪，若有則pop()出去，然後再重 新檢查陣列的值與TOP是否相同，直到有新的節點出現即結束迴圈\*/

while(j!=8)

{

if (a2[j] != -1)

{

if (a2[j] == myints.top())

{

myints.pop();

j = -1;

}

}

j++;

}

cout << endl;

cout << find;

a2[find] = myints.top();

num = myints.top();

myints.pop();

find = find+1;

// 輸出地第find 次的走訪

for (int q=0;q<8;q++)

{

if (a2[q] !=-1)

cout <<"-->"<< char (65+a2[q]);

}

cout << endl;

}

}

BFS

#include <iostream>

#include <cstdlib>

#include <queue>

using namespace std;

int main()

{

char a;

// 此處為對TREE 的node 進行處理,使用二維陣列

int a1[8][4]={{1,2,-1,-1},{0,3,4,-1},{0,5,6,-1},{1,7,-1,-1},{1,7,-1,-1},{2,7,-1,-1},{2,7,-1,-1},{3,4,5,6}}

,a2[8]={-1,-1,-1,-1,-1,-1,-1,-1},num,find=1,j=0;

//在此處宣告一個整數型態的queue 對TREE 進行BFS的處理

queue<int> bfs;

cout << "This is a BFS search:";

a = getchar();

num = a-65;

a2[0] = num;

cout << "The first:" << char (65+a2[0]) << endl;

while(find != 8)

{

j =0;

for (int i=0;i<4;i++)

{

if (a1[num][i] != -1)

bfs.push(a1[num][i]);

}

/\*判斷是否有重複出現的節點被走訪，若有則pop()出去，然後再重新檢查陣列的值與front()是否相同，直到有新的節點出現即結束迴圈\*/

while(j!=8)

{

if (a2[j] != -1)

{

if (a2[j] == bfs.front())

{

bfs.pop();

j = -1;

}

}

j++;

}

cout << find;

a2[find] = bfs.front();

num = bfs.front();

bfs.pop();

find = find+1;

// 輸出地第find 次的走訪

for (int q=0;q<8;q++)

{

if (a2[q] !=-1)

cout <<"-->"<< char (65+a2[q]);

}

cout << endl;

}

}

結論及心得

本次的期末考實作2對我的程度來說算是好做的，我在一開始就預先設想要用<stack>做DFS，及用 <queue>做BFS，設計演算法的時候也非常流暢順利，做出來的結果也令人滿意，只有一些小錯誤要修改而已。本次實作也讓我更加熟悉stack及queue的用法了，並且還知道C++還是有其他的演算法可以好好使用的。

謝謝大家~