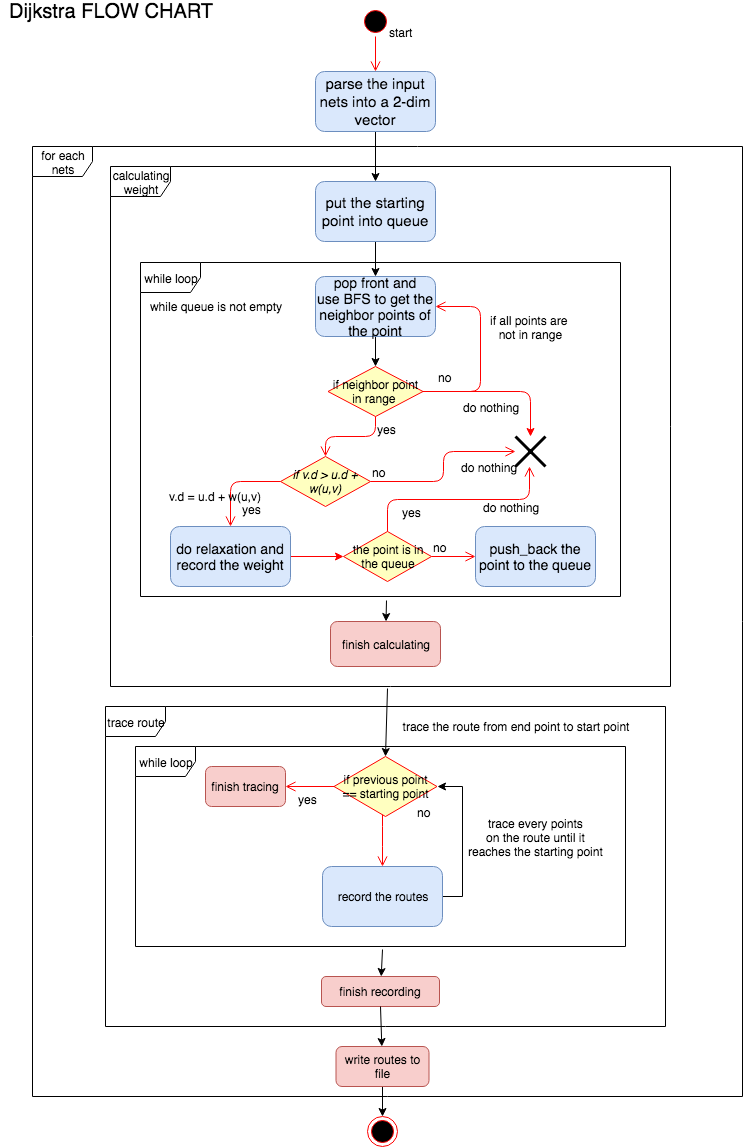
**Programming Assignment 3 Report**

**ESOE B05505053柯哲邦**

1. **演算法流程 (Algorithm Flow) <解釋程式的運作>**

****

一共寫了9個函式：

1. main：主程式，負責呼叫parser，將input file讀入程式中，並利用這些資料呼叫dijkstra。
2. dijkstra：主要執行的函式，建立所要用的所有矩陣，並執行dijkstra algorithm。
3. check\_adjacency：利用BFS，對檢查點的四周做relaxation。
4. relaxation\_x : 在x方向做relaxation
5. relaxation\_y : 在y方向做relaxation
6. compare\_weight : 將兩點比較(v.d > u.d + w(u,v) ?)
7. calculate\_weight : 計算出該點的weight
8. addline : 用來trace終點到起點的路徑，並記錄在對應的矩陣上。
9. write\_to\_file: 將答案輸出到對應的輸出檔中。

以函式來解釋程式流程：

main{

將檔案讀入，並呼叫dijkstra()

}

dijkstra{

for 每個net:

while queue不是空的

將起點放入一個queue，並從起點鄰居的每個點做relaxation。

check\_adjacency() //對queue.front()的鄰居做relaxation

addline()

write\_to\_file()

}

利用BFS從起點開始，把所有weight的表格計算並填完。最後再從終點trace回去。並以矩陣紀錄走過的路，在計算demand時需要用到。且weight的選擇，以2demand來計算。

**資料結構 (Data Structure) <解釋”特別的”資料結構>**

本次實作使用了：

1. std::pair : 用pair來裝每一個node，可以同時表示x, y值。
2. weight\_matrix : 自定義的資料結構，用來記錄表格中每一點的weight，及儲存距離和上一個點。
3. route\_matrix : 自定義的資料結構，記錄走過的path。
4. std:;deque : 用deque當作queue來做BFS的步驟。
5. vector<vector<pair<int,int> > > 二維陣列儲存pair的資料。第一維為net num數量，第二維為start, end 兩個。
6. **問題與討論 (Discussion) <討論實作中遇到的問題及疑問>**
7. 在weight的選擇。若照spec中的weight計算方式：

(int)weight = demand / capacity

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  | 4x4 | 5x5 | 10x10 | 20x20 | 60x60 |
| Overflow | 0 | 1 | 14 | 3283 | 167350 |
| Wire length | 13 | 40 | 294 | 33196 | 475456 |
| Error | 0 | 0 | 0 | 0 | 0 |

weight = 2demand

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  | 4x4 | 5x5 | 10x10 | 20x20 | 60x60 |
| Overflow | 0 | 0 | 4 | 0 | 57070 |
| Wire length | 13 | 38 | 272 | 20440 | 334070 |
| Error | 0 | 0 | 0 | 0 | 0 |

對weight的選擇不知道用何種方法取可以減少更多overflow和wire length

1. 原本不知道要怎麼記錄demand的部分，後來想到可以用一個矩陣記錄走過的路徑，並如果多一條線走過該點就將其+1，以此方式來記錄路徑大小。
2. 本次作業，花了整整2天de一個完全不知道為什麼的bug，還耗掉助教時間，原本以為是讀檔問題。我不斷嘗試，才發現其實只是某個二維陣列在new出來之前沒有先給他初始化。非常感謝助教耐心幫我debug。

**時間分析**

本實作運用兩個2個迴圈來進行，推測時間複雜度應為O(n2)。

在程式中運用algtimer來計算時間上的經過。運用題目所給之input來進行量測。

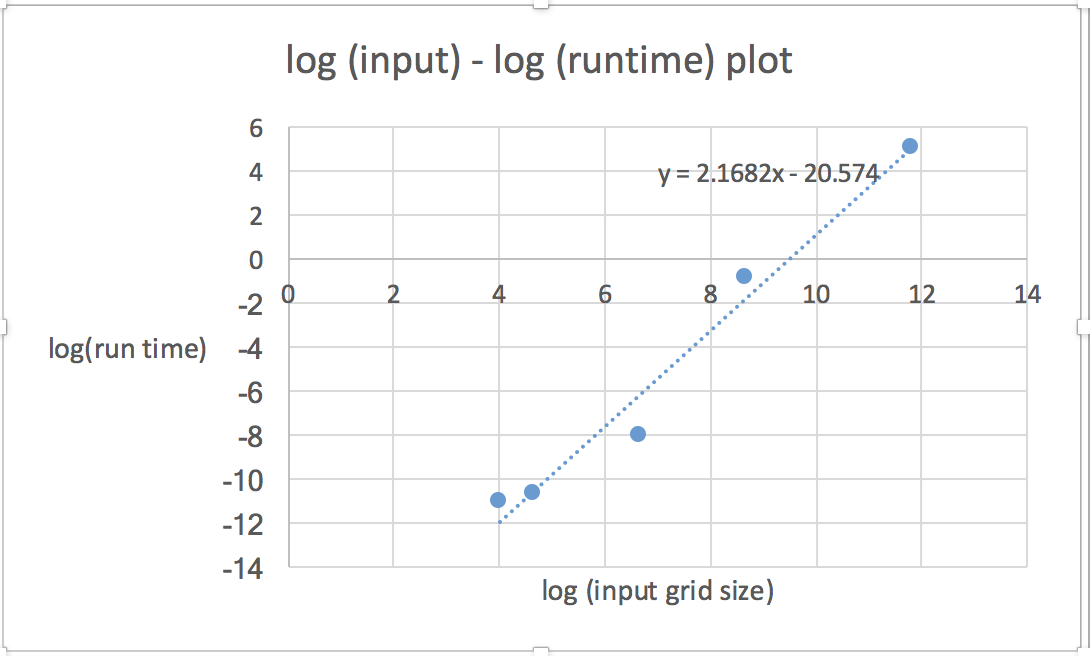
（本量測只截取中間實作dijkstra的部分，沒有囊括讀檔的部分。）

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  | 4x4 | 5x5 | 10x10 | 20x20 | 60x60 |
| Runtime(s) | 0.000498 | 0.000627 | 0.003886 | 0.566 | 33.6634 |

運用上次作業中，助教教的empirical analysis，對input和runtime都取log。

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  | 4 | 4.64385619 | 6.64385619 | 8.64385619 | 11.81378119 |
| Log(runtime) | -10.97156664 | -10.63924694 | -8.007498384 | -0.821126042 | 5.073108991 |

運用此數據畫出趨勢圖：



以此圖可以推測time complexity = O(n2.16)，與預測中O(n2)相差不遠。