元     智     大     學

資 訊 工 程 學 系

專題製作成果報告

**利用Hungarian method 及 Edmonds-Karp algorithm 實作 maximum bipartite matching**

  專 題 生：賴仲倫 、張宇越

學 號：1083301、1081542

指導教授：張經略 教授

中 華 民 國 111 年 12 月

**目 錄**

**一、** **前言** 2

[**二、 文獻探討**](#_Toc92032362) 3

[**三、 研究方法**](#_Toc92032363) 5

[**3.1、 初始架構 - 輸入設定**](#_Toc92032364) 5

[**3.2、 初始架構 –初始值設定與宣告**](#_Toc92032365) 5

[**3.3、 方法流程 - Hungarian Algorithm**](#_Toc92032366) 6

[**3.4、 方法流程 - Edmonds Karp Algorithm**](#_Toc92032366) 9

[**3.5、 測資來源**](#_Toc92032366) 10

[**四、 討論與分析**](#_Toc92032367) 12

[**五、 結論**](#_Toc92032368) 16

[**六、 References**](#_Toc92032369) 19

1. **前言**

　　在圖論中，bipartite graph是特殊的圖，又稱為二分圖、偶圖、二部圖。在bipartite graph中的點會被分成V0、V1兩個互斥集，同個集合中的點不會有邊相連。而二分圖用途是來研究兩種不同類型的物件之間的關係，例如：有 M 個求職者和 N 個職位，每個申請人都有他們感興趣的工作子集，每個職位空缺只能接受一個申請人，並且只能為一個職位任命一個職位申請人。為求職者分配工作，讓盡可能多的求職者找到工作。

　　Hungarian Algorithm的核心就是尋找增廣路徑，是一種用增廣路徑求二分圖最大匹配的算法，主要是為了在一個多項式時間內解決與二分圖匹配有關的問題，在1955年由美國數學家Harold W. Kuhn所開發並發表，他將此演算法命名為匈牙利演算法，是因為演算法很大一部分是基於以前匈牙利數學家Dénes Kőnig和Jenő Egerváry的早期作品建立起來的。James Raymond Munkres在1957年審查該演算法時，發現它是（強）多項式的。此後該演算法被稱為Kuhn–Munkres演算法或Munkres分配演算法。

Edmonds Karp Algorithm主要是將Maximum Bipartite Matching 的問題轉換成最大流的問題，而最大流就是圖中最大的匹配數。Edmonds Karp演算法在1970 年由 Yefim Dinitz首次發表。

　　我們的專題將探討如何利用Hungarian method及Edmonds-Karp algorithm實作maximum bipartite matching

**二、文獻探討**

**(一) Bipartite graph：**

根據文獻[1]的第四章節，Bipartite graph的定義為圖中的每個點可以分成兩個互斥集合V0和V1，使得所有邊的兩個端點，一個點屬於V0集合，一個點屬於V1集合。

**(二) Bipartite matching：**

　　根據文獻[1]的第四章節，Bipartite matching的定義是圖G=(V,E)是子圖G′= (V,E′), E′ ⊆ E，這樣沒有兩條邊 e1,e2 ∈ E′共享同一個頂點。

**(三) Maximum Matching：**

　根據文獻[2]，令G=(V,E) 為有限無向連通圖，其頂點集合為V、邊的集合為E、匹配的集合為M，且M是E的子集，使得M中沒有兩條邊共用一個頂點，而Maximum Matching就是基數最大的匹配。

**(四)增廣路徑(Augmenting path)：**

　　根據文獻[3]的第二章節，增廣路徑是說起點和終點都是目前的matching所沒碰到的點，且中間一條邊不在目前的matching裡、一條邊在目前的matching裡、下一條邊又不在目前的matching裡，如此交錯的路徑，稱為增廣路徑。例如，Figure 1中的(v5, u2, v1, u3)即為一條增廣路徑。

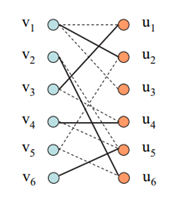


Figure 1

**三、研究方法**

**3.1、初始架構 - 輸入設定**

將一個分類好的二分圖輸入到程式中：

1. 首先輸入欲測試之演算法，輸入1為Hungarian Method、輸入2為Edmonds Karp(BFS)、輸入3為Edmonds Karp(DFS)。
2. 接著分別輸入兩個頂點集合的個數，分別放入程式中已宣告的m和n中。
3. 最後使用while迴圈讓使用者輸入目前所有的配對邊，而我們透過先前宣告的m和n創造出兩個可以容納所有配對邊且不會overflow的vector來儲存此用者所輸入的配對邊。

**3.2、初始架構 – 初始值設定與宣告**

(1) Hungarian Algorithm:

**vector<int>girls:**

紀錄女生(V0集合的頂點)和哪個男生(V1集合的頂點)配對。

**int matches:**紀錄最大的配對數。

**vector<bool>visited:**

紀錄男生(V1集合的頂點)是否被拜訪過。

(2) Edmonds Karp Algorithm:

**vector<vector<int>>grid:**

grid [i][j]=1表示V0集合的i和V1集合的j 是有路徑的。

**int source:** 代表源點S的編號。

**int sink:** 代表滙點T的編號。

**vector<bool>vis:**記錄頂點是否已被拜訪過。

**vector<int>p:** 記錄頂點的父親。

**3.3、方法流程 - Hungarian Algorithm**

Hungarian algorithm的核心就是尋找增廣路徑，是一種用增廣路徑求二分圖最大匹配的算法，而我們程式的執行流程如下，以Figure 2為例，其初始配對數為2，且在Figure 2中原有配對邊的集合為(1,6)和(2,7)：

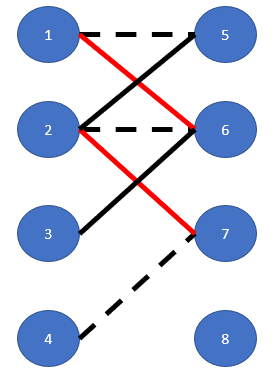


Figure 2

Step1：為了使配對數增加，我們繼續尋找有沒有其他更大的增廣路徑，而經過尋找後可以發現一條由虛線和紅線組成的增廣路徑，如Figure 3中由頂點5、1、6、2、7、4所形成的綠色Path。

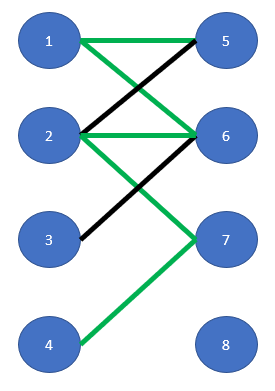


Figure 3

Step2： 在增廣路徑中移除本來在配對裡面的邊，也就是將(1,6)和(2,7)從配對邊的集合中移除，並將新的一組配對(1,5)、(2,6)、(4,7)加入配對邊的集合中，如Figure 4，因此成功將原配對數2擴大至3，且此時已無法找到更大的增廣路徑，所以得到最大匹配數為3。

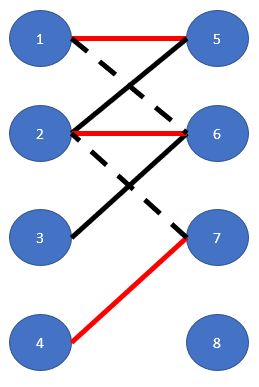


Figure 4

若還不是最大匹配數，則可以找到更大的增廣路徑，使配對數增大，因此可以依照此二步驟繼續執行，直到無法找到更大的增廣路徑為止，而此時可得到最大匹配數。

**3.4、方法流程 - Edmonds Karp Algorithm**

Edmonds Karp Algorithm 會將Maximum Bipartite Matching 的問題轉換成最大流問題。程式剛開始會建立源點S和滙點T，S會連到V0集合中的每個點，T會連到V1集合中的每個點(Figure 5)。

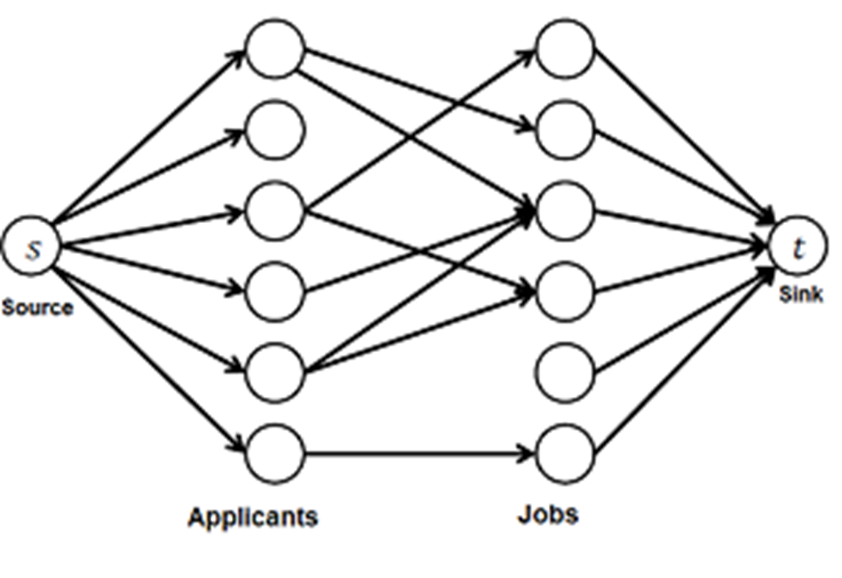
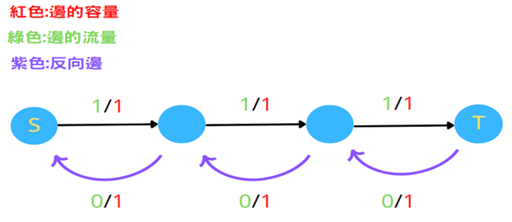


Figure 5

若將Bipartite Graph中的每條邊視為水管，則每條邊的最大容量會設成1，且剛開始每條邊的流量都會是0。Edmonds Karp Algorithm中的增廣路徑的定義是，一條路徑，起點是S，終點是T。因此，在Bipartite Graph中，每條增廣路徑的最大容量就會是3。程式剛開始會從源點使用BFS(或DFS)找到一條增廣路徑，當第一次到達滙點 T 後就停止搜索，再將增廣路徑上的每條邊的流量設成1，並在增廣路徑上，建立反向邊，反向邊的流量都設成0，反向邊的容量都設成1(如Figure 6)。然後再繼續從源點S開始找增廣路徑，若不能再找到增廣路徑則得到最大流，而最大流即此圖最大匹配數。

Figure 6

**3.5、測資來源**

我們的測資是透過程式讓電腦從20個頂點中隨機選取右邊點集合個數與左邊點集合個數，接著用兩層的for迴圈繼續讓電腦隨機選取配對邊個數與其欲配對之兩的頂點，之後將選取的結果寫入input.txt，並透過while迴圈讓上述步驟執行100次，使我們可以在這100次中取得這3個演算法的平均，而其中因為重複之邊不影響實驗結果，因此我們在此程式中沒有特別去刪除重複之邊。

**四、討論與分析**

**(一)為何可以透過Hungarian Algorithm找尋並置換增廣路徑來找到二分圖的最大匹配？**

證明匈牙利演算法的正確性:

Theorem:

對所有matching M ，若M非最大的配對集，則存在一個M的增廣路徑。(相當於:若配對集M沒有增廣路徑，則配對集M是最大的)

Pf:

現在有一個比M更大的配對集M’。

M’一定存在，因為M不是最大的配對集。

設在**M集合中但不在M’中的任意一個邊為e**。

設在**M’集合中但不在M中的任意一個邊為e’**。

現在考慮一張圖G，此張圖的每個點最多只能碰到一個e及一個e’，否則會違反**matching裡面的邊不能共用端點的規則**，因此圖G中每個點的Degree會是小於等於2。

此證明引用一個「事實」:

當每個點的Degree最大為2時，可以將圖G切割成cycle或path。

利用上面的「事實」，因此可以將圖G切割成cycle或path(如Figure 7)。

cycle的長度一定是偶數，否則會違反**matching裡面的邊不能共用端點的規則。**因為圖G中每個點最多只能碰到一個e及一個e’，因此cycle中的e及e’的個數相同。

由於**M’集合**比**M集合**大，因此**e’的個數**比**e的個數多**。因為cycle中的e及e’的個數相同，因此圖G中一定會有一條path Q存在，且此條path Q中**e’的個數**比**e的個數多。**

由於圖G中每個點最多只能碰到一個e及一個e’，且path Q中**e’的個數**比**e的個數多**，因此path Q中的兩端點A、B連到的邊一定都是e’，則此條**path Q**滿足增廣路徑的定義，因此**path Q**為一條增廣路徑。

證明**path Q**一定是一條增廣路徑:(反證法)

設**path Q**中的端點為A、B。

若A點能向外再延伸出一條邊連到新的端點C，則此條邊必存在M集合中，因為path Q中的兩端點向內連到的邊一定都是e’，而且圖G中每個點最多只能碰到一個e及一個e’。設此條為新的path R(端點是C、B)，矛盾(因為path Q的兩端點是A、B，與原假設不符)。同理可證明**path Q**中的另一個端點B。{\displaystyle O(n^{2.5})}。

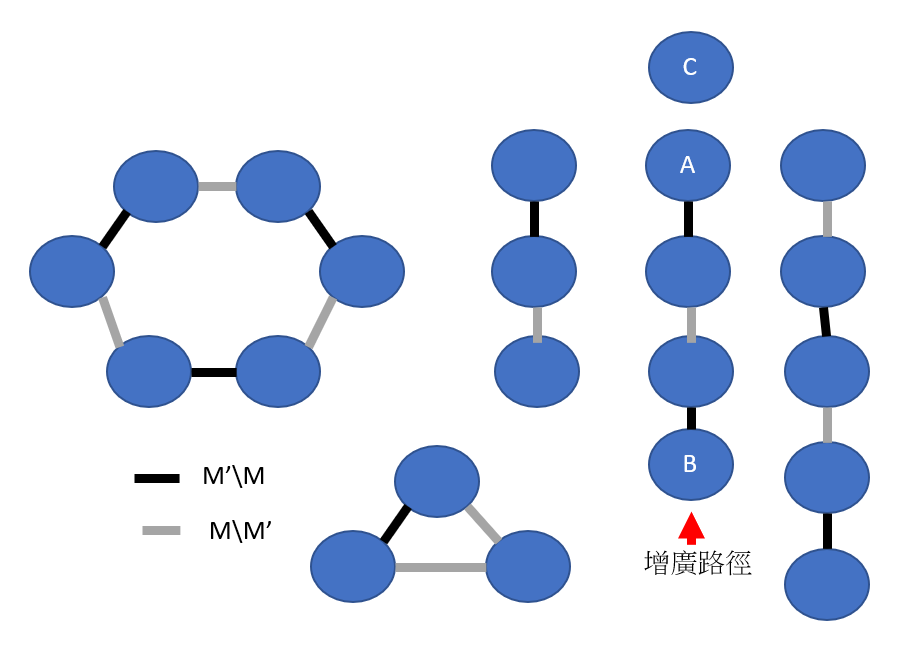


Figure 7

**(二)為何測資的產生方式是用隨機的？**

因為Maximum Bipartite Matching問題的測資，沒有什麼特別的規則，因此程式比賽的命題委員們才會覺得困擾，所以我們才用隨機的方式產生測資。

**(三)** **Hungarian algorithm的應用？**

有 M 個考生和 N 個科目，每位考生都需要看資演、計系、數學，但是由於可以準備考試的時間不足，因此讓每位考生只準備一個科目，並讓每位考生互相教學沒看的科目，以此增進效率。下表的圖是每位考生看各科所需要的時間，以此表決定各考生該看哪科，可以讓總體所花費的時間成本最小。

科目

|  |  |  |  |
| --- | --- | --- | --- |
| 人名 | 資結演算法 | 計組OS | 線代離散 |
| 元智 | 2 hr | 3 hr | 3 hr |
| 陽交 | 3 hr | 2 hr | 3 hr |
| 清華 | 2 hr | 3 hr | 2 hr |

當把匈牙利方法應用於上面的表格時，會得到最低的時間成本也就是最有效率的方法為6hr，讓元智讀資演、讓陽交讀計系、讓清華讀數學，就可以達到這個時間成本。

**五、結論**

　　從Figure 8中可以發現，這三種演算法經由程式所隨機產生的100筆測資後，接著讓程式在取得每個測資的時間並取平均值做比較，因此根據Figure 8，可以發現Hungarian algorithm的平均時間是最短的，而Edmonds Karp algorithm (BFS) 的平均時間是最長的，Edmonds Karp algorithm (DFS) 的平均時間是中間值。

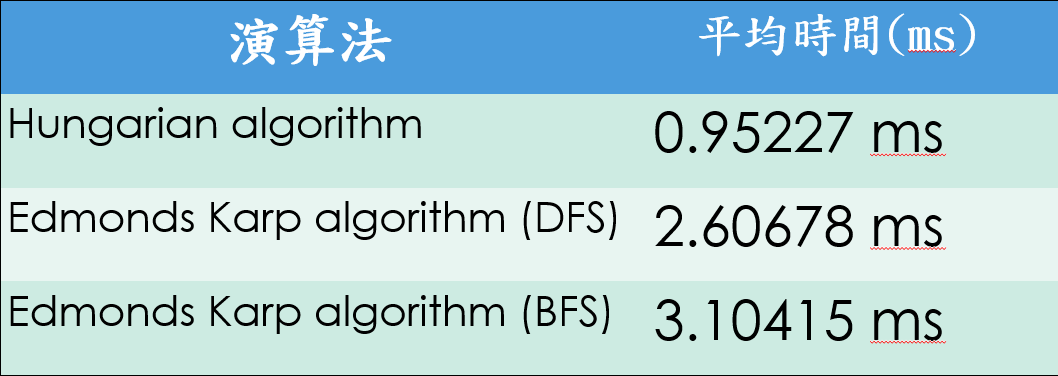
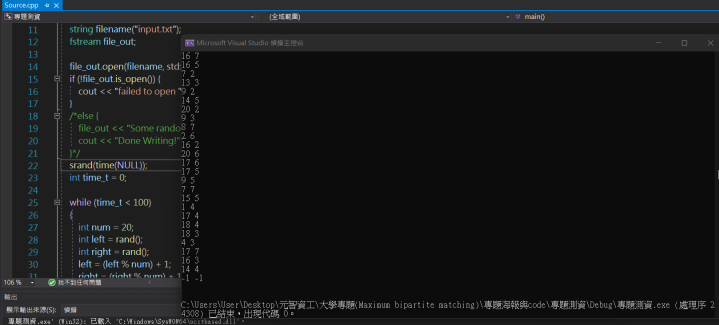
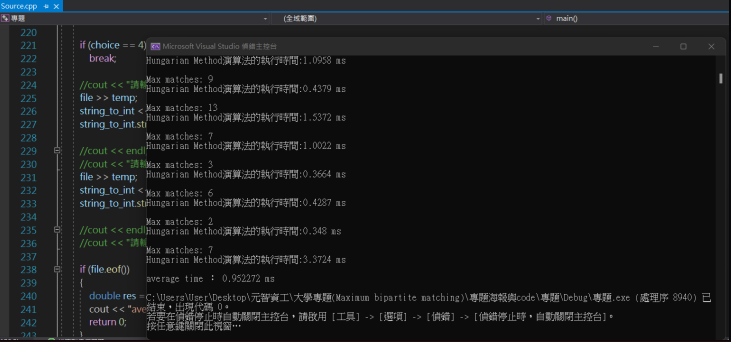


Figure 8

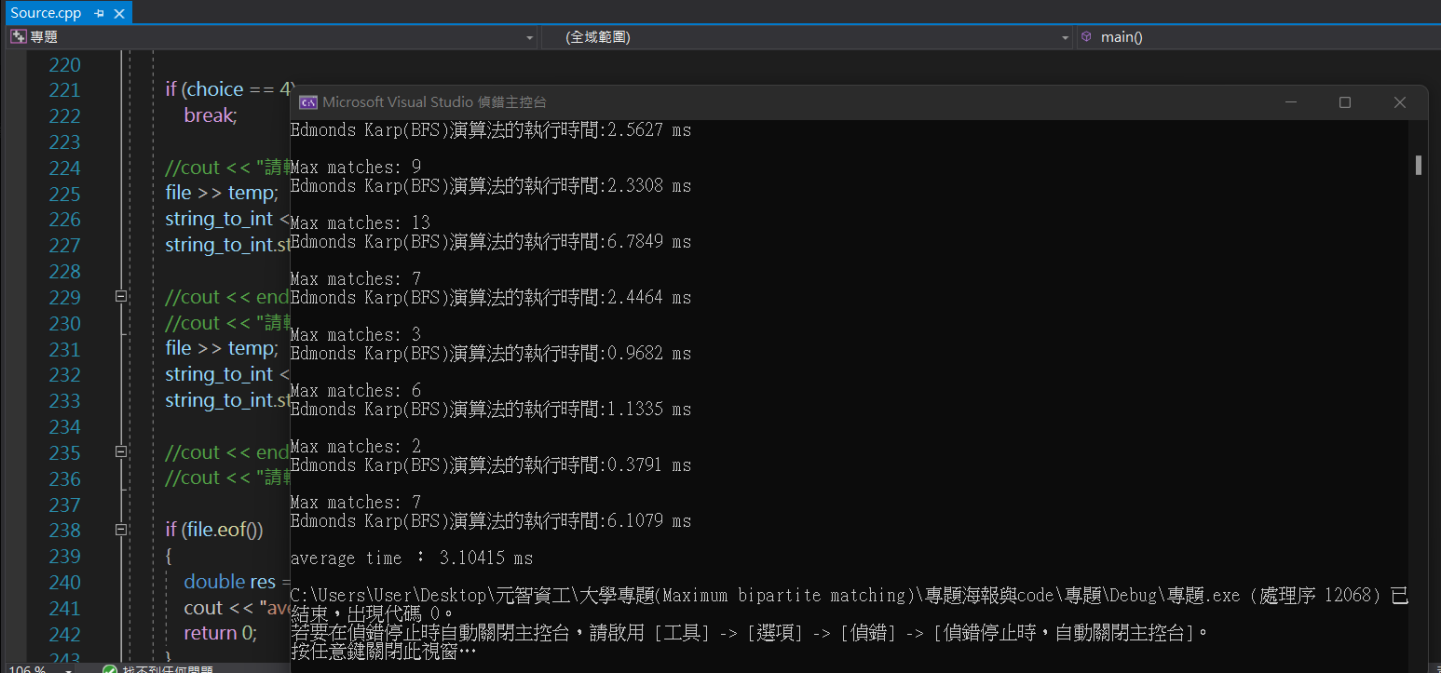
* **實際成果 - 測資產生**



* **實際成果 - Hungarian method**



* **實際成果 - Edmonds Karp(BFS)**



* **實際成果 - Edmonds Karp(DFS)**



**六、References**

　　[1] Bellur, Umesh, and Roshan Kulkarni. "Improved matchmaking algorithm for semantic web services based on bipartite graph matching." IEEE international conference on web services (ICWS 2007). IEEE, 2007.

[2] Giel, Oliver, and Ingo Wegener. "Evolutionary algorithms and the maximum matching problem." Annual Symposium on Theoretical Aspects of Computer Science. Springer, Berlin, Heidelberg, 2003.

　　[3] Mills-Tettey, G. Ayorkor, Anthony Stentz, and M. Bernardine Dias. "The dynamic hungarian algorithm for the assignment problem with changing costs." Robotics Institute, Pittsburgh, PA, Tech. Rep. CMU-RI-TR-07-27 (2007).