Data Mining HW3 Report

👧 P77111037 樊紹萱

1️⃣ Implementation detail

🌟 解釋你是怎麼實作這三個演算法、三個演算法的原理（可比較相同或不同處）

🔆 原理與實作方式

Algorithm	原理	實作方式
HITS	透過計算該節點**被多少個節點指到**，以及該節點**指向多少個節點**，來計算該節點的權重	⚫ 每一個 iteration 時，會先計算所有節點的 authority & hub 🔺 Authority = Sum(該節點所有的父節點的 Hub 值) 🔺 Hub = Sum(該節點所有的子節點的 Authority 值) ⚫ 接著進行標準化 🔺 Authority = 該節點未標準化前的authority / 未標準化前所有節點的Authority總和 🔺 Hub = 該節點未標準化前的 Hub / 未標準化前所有節點的 Hub總和
PageRank	PageRank 是指網頁被看到的可能性，每個網頁都有個別的 PageRank，取決於網頁間連結關係 💡 一個網站的 PageRank 值，來自於**加總所有連結到該網站的網站的PageRank 值除以本身的導出連結數**	⚫ 每一個 iteration 時，會先計算所有節點的 PageRank 🔺 先算該節點所有父節點的總和 (每個父節點的 PageRank/父節點的子節點總數) ⚫ 接著進行標準化 🔺 PageRank = 該節點未標準化前的 PageRank / 未標準化前所有節點的PageRank總和
SimRank	假設用戶和物品在空間中形成了一張圖，如果兩個用戶相似，則與這兩個用戶相關聯的物品也類似；如果兩個物品類似，則與這兩個物品相關聯的用戶也類似	⚫ 迭代每一對節點，分別計算個節點的SimRank，且更新並取代舊的SimRank的值

☼ HITS & PageRank 比較

1. 相同:
    - 他們都利用 Web 圖的鏈接結構來決定頁面的相關性
2. 相異

HITS	PageRank
需**線上**計算 👉 計算效率較**低** (因與用戶輸入的查詢請求有高度的相關性，所以在接收到用戶請求時，需要即時的計算)	可**離線**計算
計算物件數量較**少**	是**全局性演算法**，對所有互聯網頁面節點進行處理
適合部署在**用戶端**	適合部署在**伺服器端**

☼ PageRank & SimRank 比較

相異： PageRank只能得到某一個節點自己的權重，而SimRank卻可以得到兩兩之間的權重度量

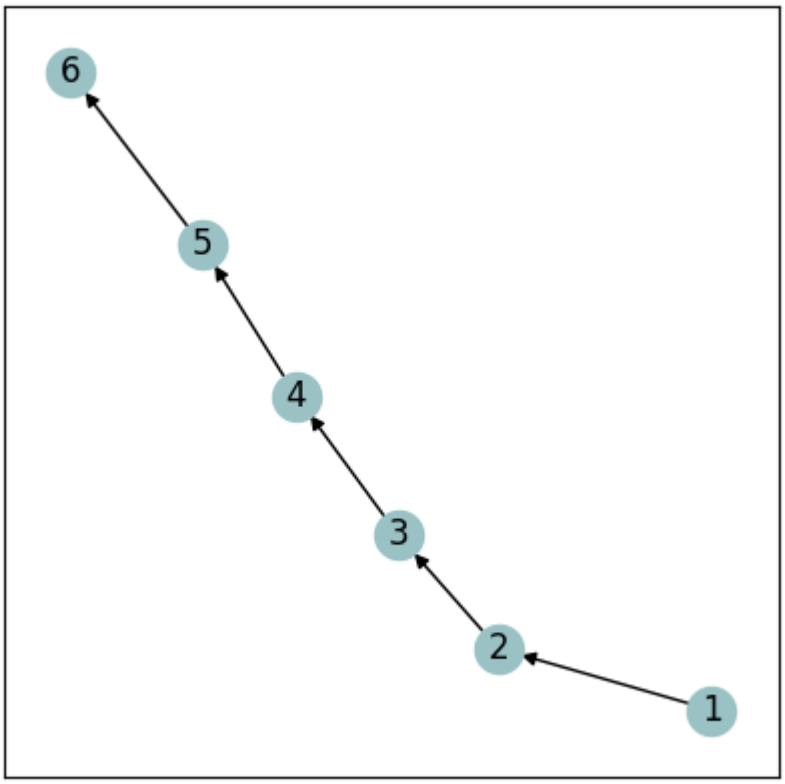
2️⃣ Result analysis and discussion (iteration = 1)

⭐ 包含投影片中的find a way: Find a way (e.g., add/delete some links) to increase hub, authority, and PageRank of Node 1 in first 3 graphs respectively. 對圖1到3的 node 1，試著增加或減少 links，或增加減少點的數量，使node 1在 hub/authority/PageRank三個值中「擇一」增加

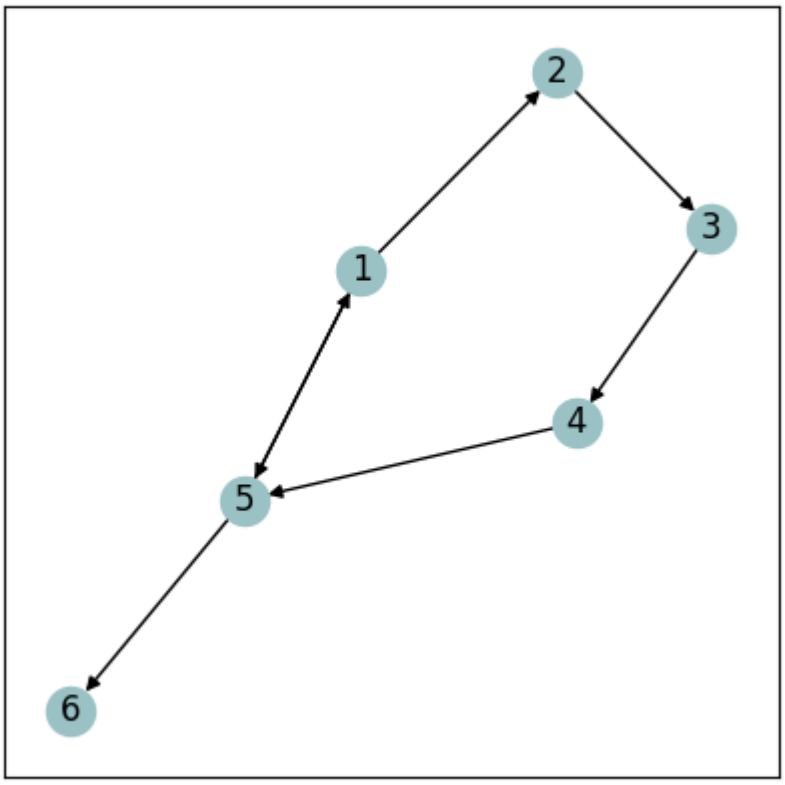
以下分別進行 圖1到3的 node 1 分析

🔆 Graph1

原本的圖一：👇



後來的圖一：👇🔎讓 node 1 指到 node5，且讓 node5 也指回 node1



✏️解釋與比較

新的圖我把 **node 1 指到 node5，且讓 node5 也指回 node1**

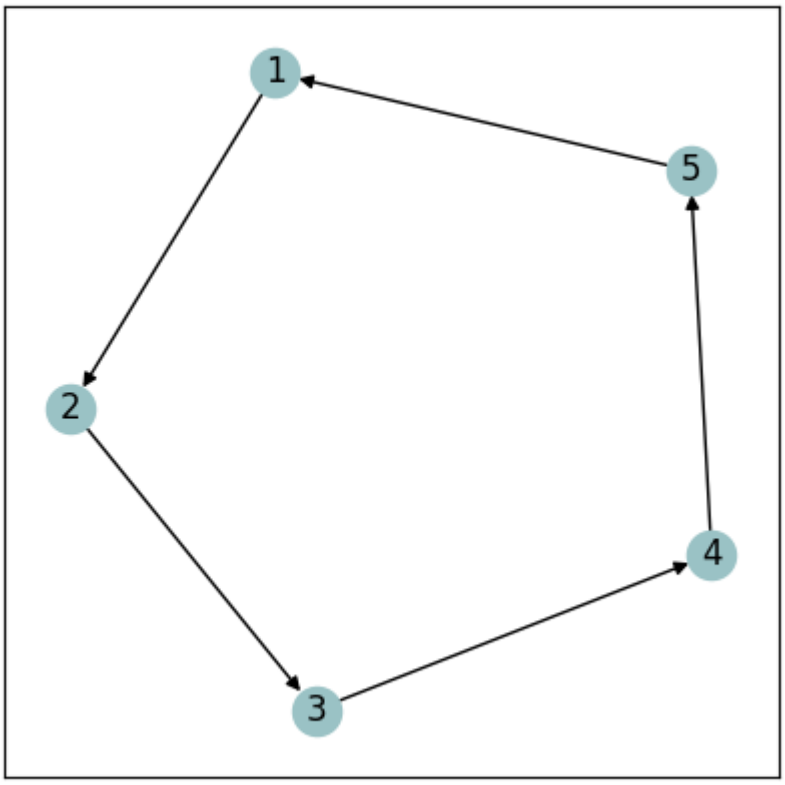
node 1：

- children: 2,5 （原本只有2)
- parent: 5 (原本沒有)
- Authority: 0.0 👉 0.14286
- Hub: 0.2 👉 0.333

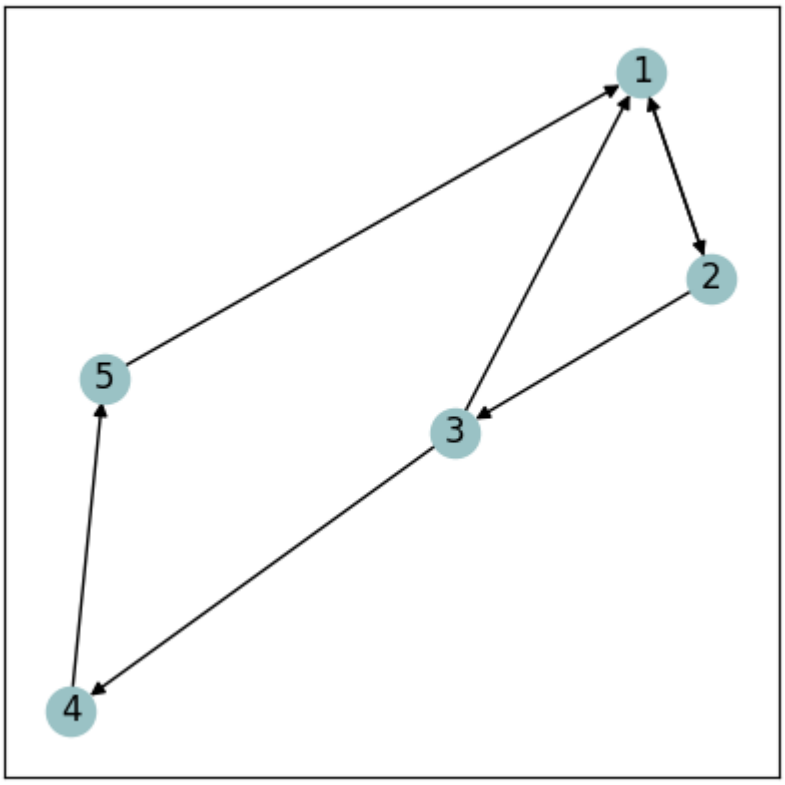
Grapgh	Result
graph1	Authority: [0.0 0.2 0.2 0.2 0.2 0.2] Hub: [0.2 0.2 0.2 0.2 0.2 0.0]
graph1 - new	Authority: [0.14286 0.14286 0.14286 0.14286 0.28571 0.14286] Hub: [0.333 0.111 0.111 0.222 0.222 0.0]

Graph2

原本的圖二：



後來的圖二： 👇 (讓 node 2 & 3 都指到 node1)



📝 解釋與比較

🔎 新的圖我把 **node 2 & 3 都指到 node1**

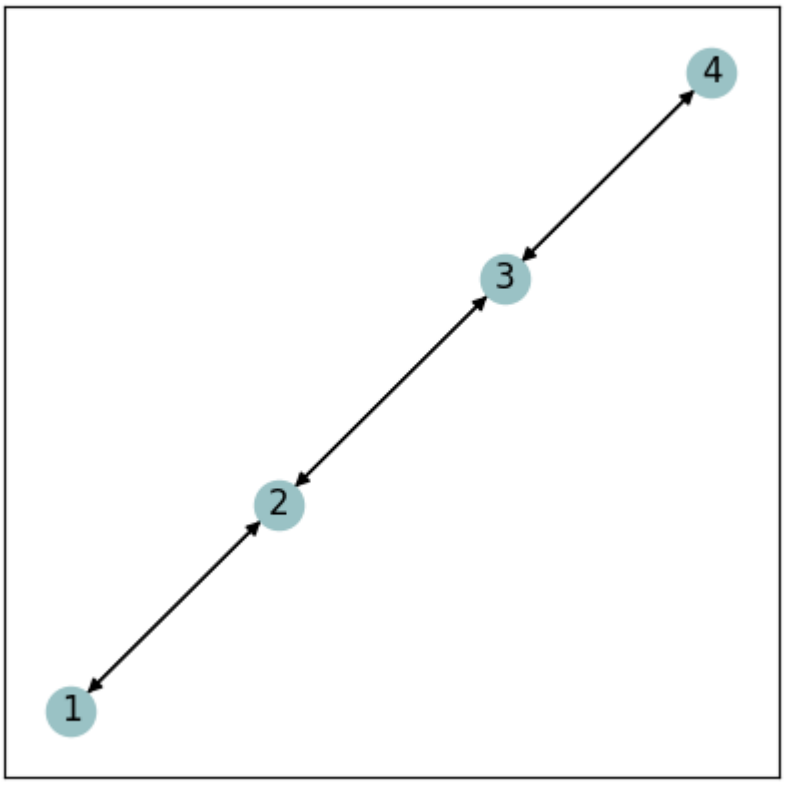
node 1 :

- children: 2 （沒更動)
- parent: 2,3,5 (原本只有5)
- Authority: 0.2 👉 0.42857

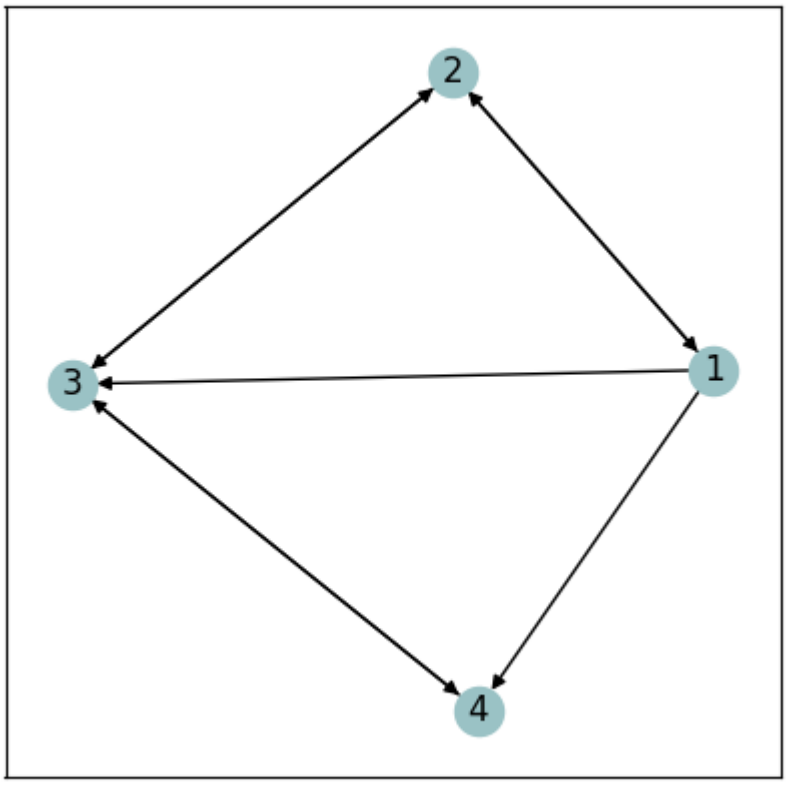
Grapgh	Result
graph1	Authority: [0.2 0.2 0.2 0.2 0.2] Hub: [0.2 0.2 0.2 0.2 0.2 0.0]
graph2 - new	Authority: [0.42857 0.14286 0.14286 0.14286 0.14286] Hub: [0.077 0.308 0.308 0.077 0.231]

Graph3

原本的圖三：



後來的圖三： 👇 (node1 多指到 3 & 4)



✏️ 解釋與比較

🔎 新的圖我把 **node1 多指到 3 & 4**

node 1 :

- children: 2,3,4 （原本只有2)
- parent: 2 (沒更動)
- Hub: 0.2 👉 0.389

Grapgh	Result
graph3	Authority: [0.16667 0.33333 0.33333 0.16667] Hub: [0.2 0.3 0.3 0.2]
graph3 - new	Authority: [0.125 0.25 0.375 0.25] Hub: [0.389 0.222 0.222 0.167]

3️⃣⏱️ Computation performance analysis

🌟 對 dataset 中的圖1到5，計時每張圖在三個演算法上的秒數。並試著解釋為什麼某些演算法或某些圖要跑特別久。

Graph	HITS ⏱️Time second	PageRank ⏱️Time second	SimRank ⏱️Time second
1	0.002520	0.001192	0.001477
2	0.003210	0.001221	0.001474
3	0.003011	0.001218	0.001154
4	0.002703	0.001078	0.001954
5	0.018052	0.008929	⚠️11.646827

📝 Explanation

graph_5.txt 用 SimRank 演算法的時候花費很久的時間，我覺得應該是因為 SimRank 跑過的節點數量是其他的平方倍 (n * n)，因為有兩個 for loop，如下 👇

```
    for node1 in graph.nodes:
        for node2 in graph.nodes:
```

💡 我後來又多跑了 graph6 (圖形又比 graph5複雜許多)，耗時如下：

Graph	HITS ⏱️Time second	PageRank ⏱️Time second	SimRank ⏱️Time second
1	0.010403	0.005775	⚠️467.249918

4️⃣ Discussion (iteration = 1)

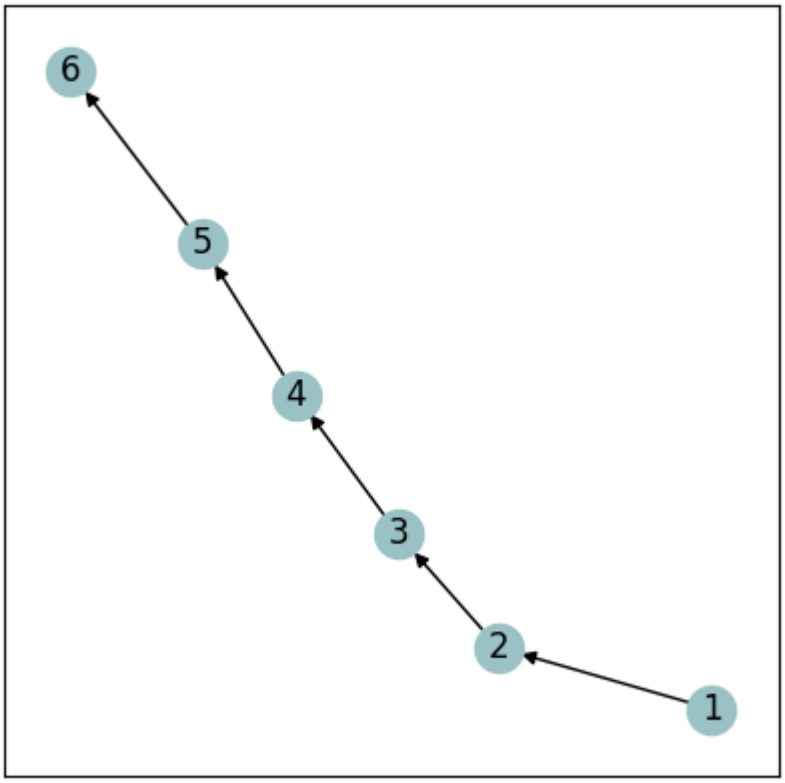
效能

以第3️⃣上面那張比較圖為例，發現說在 iteration=1的時候，**在圖1~5最慢的都是 HITS 演算法，最快的幾乎都是 PageRank 演算法**。但在圖形越來越複雜的時候 (ex: graph 5)， 反而是 SimRank 最慢，而且慢許多

展示 graph 1~6 的圖

🌟 (我用 `networkx.draw_networkx` & `matplotlib.pyplot` 來畫的)

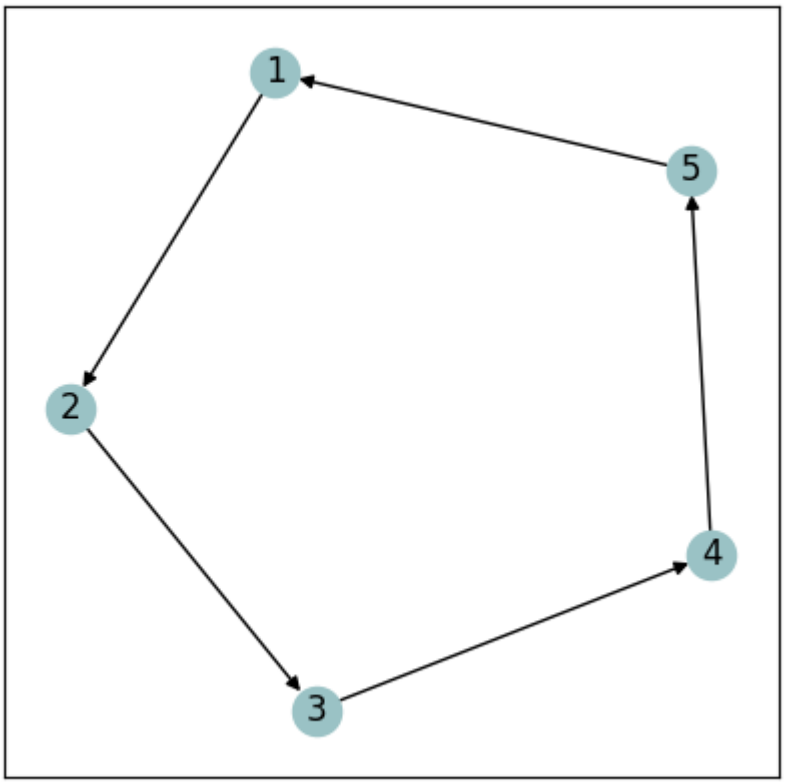
🌏 **Graph 1**



Algorithm	Result	Explanation
HITS	Authority: [0.0 0.2 0.2 0.2 0.2 0.2] Hub: [0.2 0.2 0.2 0.2 0.2 0.0]	💡 發現到 node1 的 **authority 是 0**，因為 node1 沒有被任何節點指到 💡 發現到 node6 的 **hub 是 0**，因為 node6 沒有指到任何節點
PageRank	PageRank: [0.061 0.112 0.156 0.193 0.225 0.252]	💡 因為剛好是由1指到6 （**單向**)，所以 PageRank 是由低到高

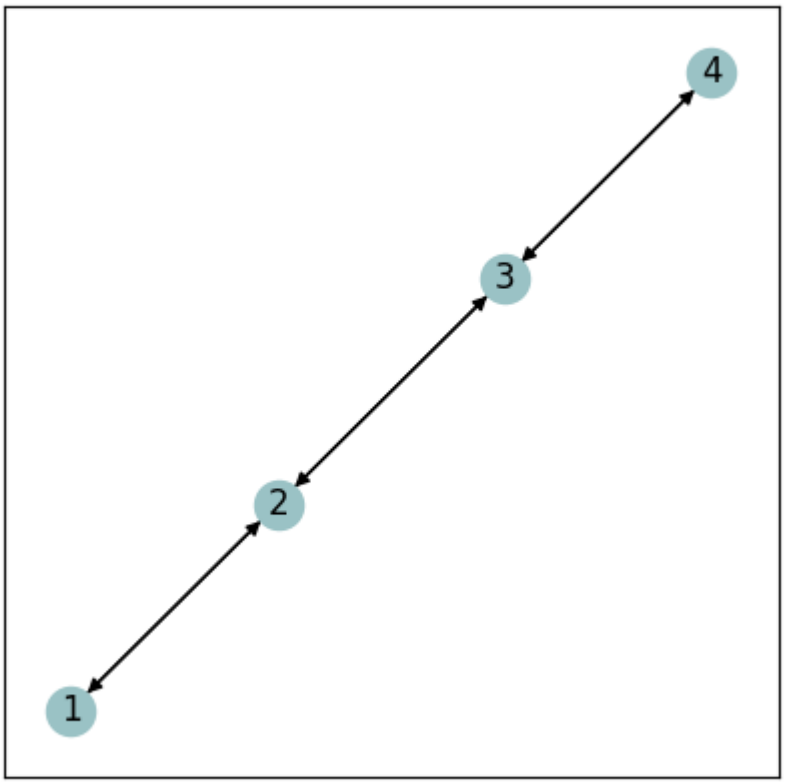
Algorithm	Result	Explanation
SimRank	1.000 0.000 0.000 0.000 0.000 0.000 1.000 0.000 0.000 0.000 0.000 0.000 1.000 0.000 0.000 0.000 0.000 0.000 1.000 0.000 0.000 0.000 0.000 0.000 1.000	

🌏 Graph 2



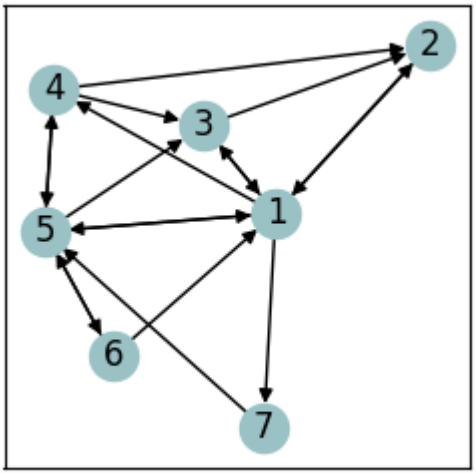
Algorithm	Result	Explanation
HITS	Authority: [0.2 0.2 0.2 0.2 0.2] Hub: [0.2 0.2 0.2 0.2 0.2]	💡 **環狀結構** : 每個節點的 child & parent 都是1
PageRank	PageRank: [0.25 0.221 0.196 0.175 0.158]	**環狀結構**
SimRank	1.000 0.000 0.000 0.000 0.000 0.000 1.000 0.000 0.000 0.000 0.000 0.000 1.000 0.000 0.000 0.000 0.000 0.000 1.000 0.000 0.000 0.000 0.000 0.000 1.000	**環狀結構**

🌏 Graph 3

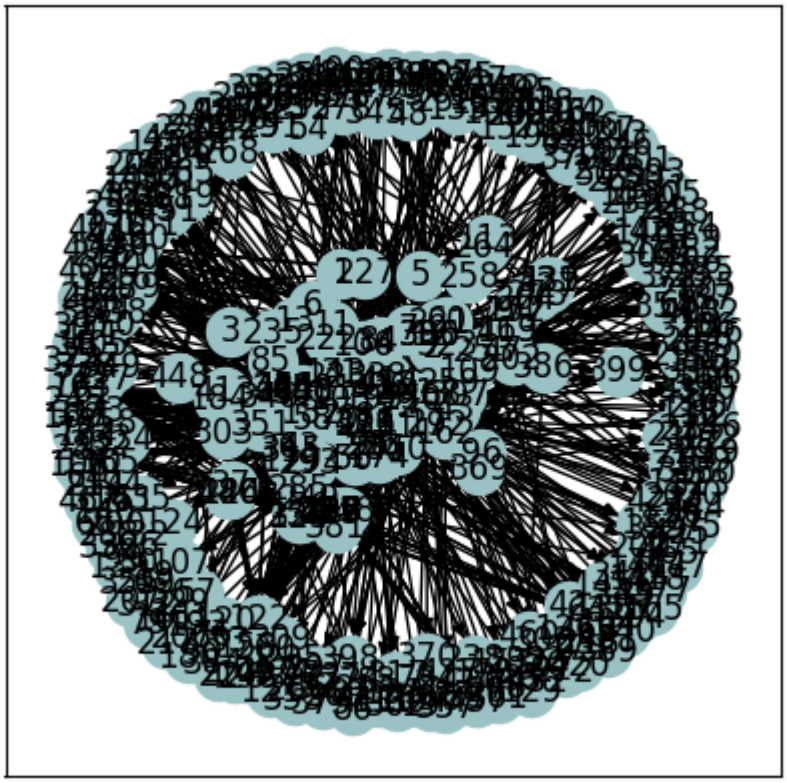


Algorithm	Result	Explanation
HITS	Authority: [0.16667 0.33333 0.33333 0.16667] Hub: [0.2 0.3 0.3 0.2]	💡 node1 & 4 的 children & parent 數量都相同 💡 node2 & 3 的 children & parent 數量都相同
PageRank	PageRank: [0.147 0.273 0.399 0.181]	💡 **node2 & 3 的 PR值都較高**
SimRank	1.000 0.000 0.45 0.000 0.000 1.000 0.000 0.000 0.45 0.000 1.000 0.000 0.000 0.45 0.000 1.000	

Graph 4



Graph 5



Graph 6