Project 3

Link Analysis

N96094196 張維峻

工程科學系

摘要 0

File structure [1]

```
+ – algorithm
| + – Alg_Node_Graph.py
+ – figures
+ – hw3datset
+ – ouput
+ – script_exp
| + – P1_graph1.py
| + – P2_graph2.py
| + – P3_graph3.py
| + – P4_graph4.py
| + – P5_graph5.py
| + – P6_graph6.py
| + – P7_graph7.py
+ – tool
| + – trans2graph.py
```

- algorithm/Alg_Node_Graph.py: 讀入圖片,建立節點。
- figures/: 存放輸出的圖片
- hw3datset: 資料集
- ouput/: 存放目標檔案格式
- script_exp/:檔案執行範例
- tool/trans2graph.py:

Implement²

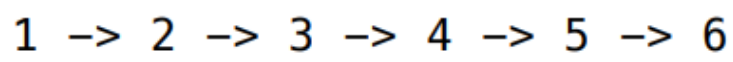
- **HITS**
  - 使用 list 結構記錄每個節點 Authority 和 hub 值,
    並且初始化成 1 / n ^ 0.5。
  - 程式邏輯-進入迴圈
    - 對每個節點，將其 Authority 更新為其全部父節點之 Hub 值總和。
      將其 Hub 更新為其全部子節點之 Authority 值總和。
    - 將全部 Authority 除以全部 Authority 平方和開根號 (歐基理德距離)
    - 將全部 Hub 除以全部 Hub 平方和開根號 (歐基理德距離)
    - 如果新舊 Authority 之差值平方和，加上新舊 Hub 之差值平方和小於 epsilon。(default: 1e-10)，則跳出迴圈。
  - 輸出 Authority 及 Hub list。
- **PageRank**
  - 使用 list 結構記錄每個節點 PageRank 的值,初始化成 1 / n ^ 0.5。
  - 程式邏輯-進入迴圈
    - 對每個節點，將其 PageRank 更新為 d 除以總節點數，加上(1-d)乘以其全部父節點 PageRank 值除以父節點對外分支數的總和
      (default : d=0.1)
    - 將全部 Authority 除以全部 Authority 平方和開根號 (歐基理德距離)
    - 將全部 Hub 除以全部 Hub 平方和開根號 (歐基理德距離)
    - 如果新舊 Authority 之差值平方和，加上新舊 Hub 之差值平方和小於 epsilon。(default: 1e-10)，則跳出迴圈。
  - 輸出 Authority 及 Hub list

SimRank

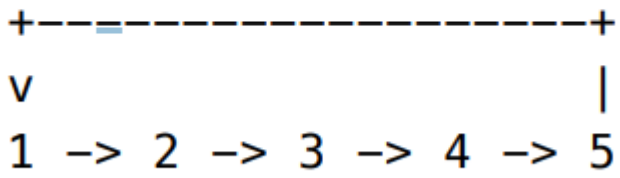
- 建立一個 dictionary 來儲存節點與節點之間的倆倆 SimRank 值，並初始化自己對自己的 SimRank 為 1 ，其餘的 SimRank 為 0 。
- 程式邏輯-進入迴圈
  - 對於全部節點中的倆倆配對 a, b 節點，其中 a 不等於 b ，將 SimRank(a, b) 及 SimRank(b, a) 更新為 C 除以 a 之父節點數，除以 b 之父節點數，乘以全部 a 之父節點配對上全部 b 之父節點之 SimRank 總和。(default: C=1.0)
  - 如果更新前後 SimRank 之差值平方和小於 epsilon (default: 1e-10)，則跳出迴圈。
- 輸出 SimRank dictionary。

Graph[3]

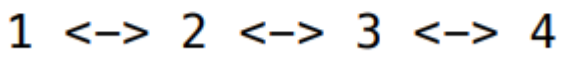
- Graph1



- Graph2



- Graph3



- **Graph4**

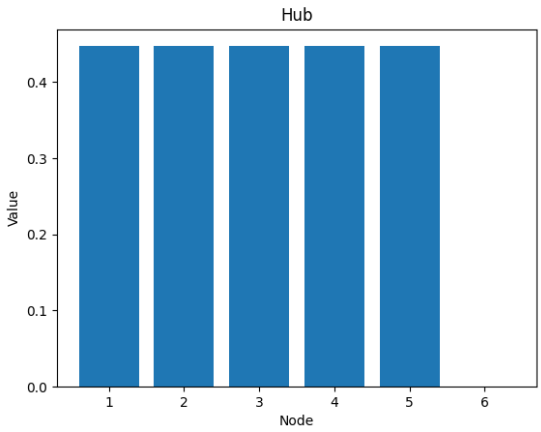
```
Node 1
In  Node: [2, 3, 5, 6]
Out Node: [2, 3, 4, 5, 7]
Node 2
In  Node: [1, 3, 4]
Out Node: [1]
Node 3
In  Node: [1, 4, 5]
Out Node: [1, 2]
Node 4
In  Node: [1, 5]
Out Node: [2, 3, 5]
Node 5
In  Node: [1, 4, 6, 7]
Out Node: [1, 3, 4, 6]
Node 7
In  Node: [1]
Out Node: [5]
Node 6
In  Node: [5]
Out Node: [1, 5]
```

Result[4]

- ## HITS

  - ### Graph1

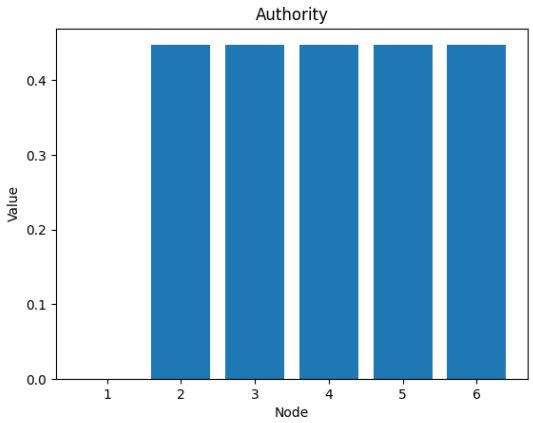
  - **Hub:**




```
Hub:
Top1 | Node: 1| Value: 0.44721
Top2 | Node: 2| Value: 0.44721
Top3 | Node: 3| Value: 0.44721
Top4 | Node: 4| Value: 0.44721
Top5 | Node: 5| Value: 0.44721
Top6 | Node: 6| Value: 0.00000
```

```
HITS_hub:
[0.44721 0.44721 0.44721 0.44721 0.44721 0.     ]
```

  - **Authority:**




```
Authority:
Top1 | Node: 2| Value: 0.44721
Top2 | Node: 3| Value: 0.44721
Top3 | Node: 4| Value: 0.44721
Top4 | Node: 5| Value: 0.44721
Top5 | Node: 6| Value: 0.44721
Top6 | Node: 1| Value: 0.00000
```

```
HITS_authority:
[0.44721 0.44721 0.44721 0.44721 0.44721 0.     ]
```

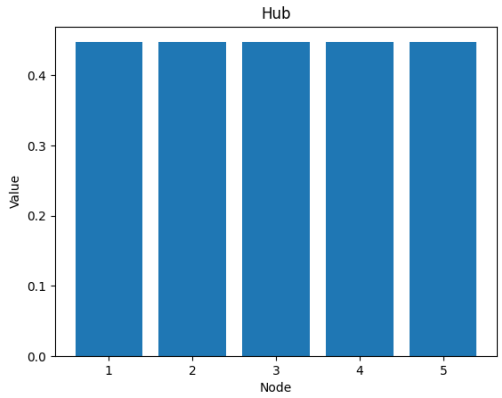
  - **Discussion:**

1. 由節點連線關係可以看的出來 Node1 並沒有被其他節點指向(推薦)，因此其 Authority 值為 0 ，其他節點都可以接收其父節點之 Hub 值為其 Authority 值。
2. 由節點連線關係可以看的出來 Node6 並沒有指向(推薦)其他節點，因此其 Hub 值為 0 ，其他節點都可以接收其子節點之 Authority 值為其 Hub 值。

3. 因此說明了如果對外指向的節點數愈多 Hub 就會有機會愈高，如果被愈多點指向則 Authority 將會有機會愈高。

✧ Graph2

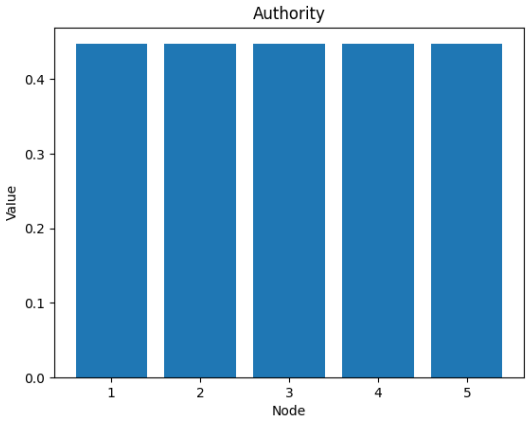
➢ Hub:




```
Hub:
Top1 | Node: 1| Value: 0.44721
Top2 | Node: 2| Value: 0.44721
Top3 | Node: 3| Value: 0.44721
Top4 | Node: 4| Value: 0.44721
Top5 | Node: 5| Value: 0.44721
```

```
HITS_hub:
[0.44721 0.44721 0.44721 0.44721 0.44721]
```

➢ Authority:




```
Authority:
Top1 | Node: 1| Value: 0.44721
Top2 | Node: 2| Value: 0.44721
Top3 | Node: 3| Value: 0.44721
Top4 | Node: 4| Value: 0.44721
Top5 | Node: 5| Value: 0.44721
```

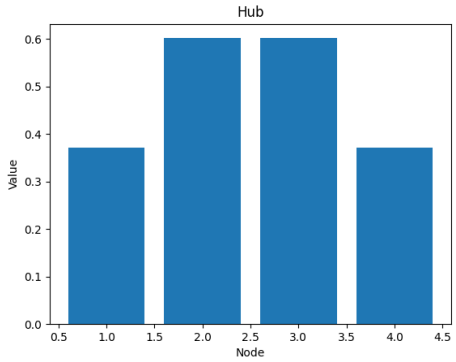
```
HITS_authority:
[0.44721 0.44721 0.44721 0.44721 0.44721]
```

➢ Discussion:

1. 因為這個 Graph 是一個環狀，每個節點都是被前一個節點指向，同時也指向下一個節點，沒有任何差異，因此不論在 Hub 及 Authority 皆為一致。
2. 如果將此環狀架構打破一個 edge 那麼就會跟 graph1 一樣，最末端的節點 Hub 值為 0 ，最前端節點 Authority 為 0。

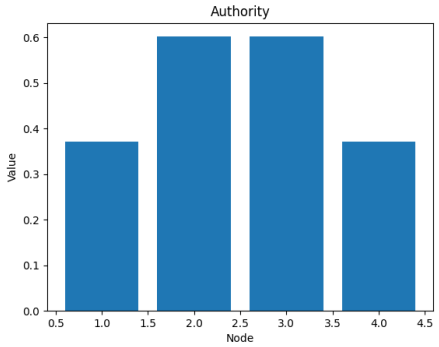
✧ Graph3

➢ Hub:




```
Hub:
Top1 | Node: 2| Value: 0.60150
Top2 | Node: 3| Value: 0.60150
Top3 | Node: 1| Value: 0.37175
Top4 | Node: 4| Value: 0.37175
HITS_hub:
[0.6015  0.6015  0.37175 0.37175]
```

➢ Authority:



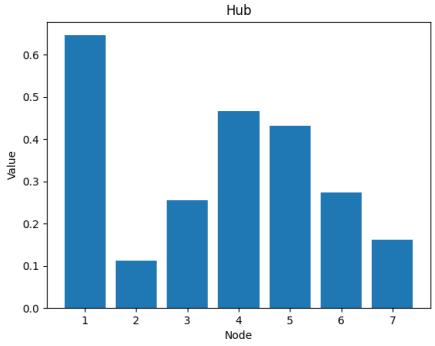

```
Authority:
Top1 | Node: 2| Value: 0.60150
Top2 | Node: 3| Value: 0.60150
Top3 | Node: 1| Value: 0.37175
Top4 | Node: 4| Value: 0.37175
HITS_authority:
[0.6015  0.6015  0.37175 0.37175]
```

➢ Discussion:

3. 因為這個 Graph 是一個環狀，每個節點都是被前一個節點指向，同時也指向下一個節點，沒有任何差異，因此不論在 Hub 及 Authority 皆為一致。
4. 如果將此環狀架構打破一個 edge 那麼就會跟 graph1 一樣，最末端的節點 Hub 值為 0 ，最前端節點 Authority 為 0。

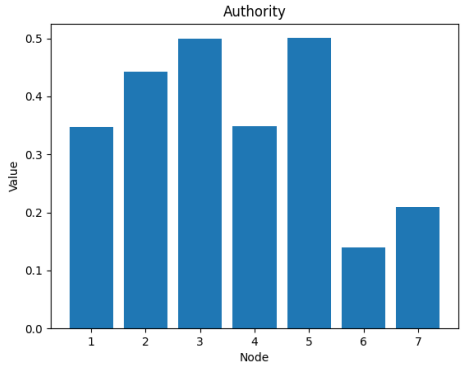
✧ Graph4

➢ Hub:




```
Hub:
Top1 | Node: 1| Value: 0.64642
Top2 | Node: 4| Value: 0.46621
Top3 | Node: 5| Value: 0.43119
Top4 | Node: 6| Value: 0.27395
Top5 | Node: 3| Value: 0.25506
Top6 | Node: 7| Value: 0.16186
Top7 | Node: 2| Value: 0.11209
HITS_hub:
[0.64642 0.46621 0.43119 0.27395 0.25506 0.16186 0.11209]
```

➢ Authority:




```
Authority:
Top1 | Node: 5| Value: 0.50063
Top2 | Node: 3| Value: 0.49914
Top3 | Node: 2| Value: 0.44219
Top4 | Node: 4| Value: 0.34841
Top5 | Node: 1| Value: 0.34669
Top6 | Node: 7| Value: 0.20900
Top7 | Node: 6| Value: 0.13941
HITS_authority:
[0.50063 0.49914 0.44219 0.34841 0.34669 0.209   0.13941]
```

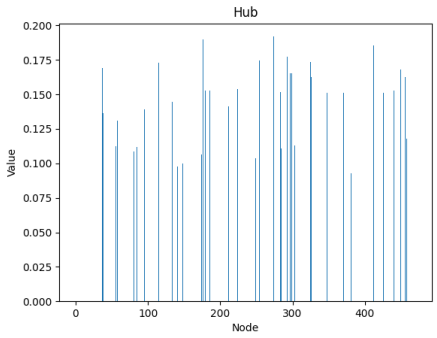
➢ Discussion:

觀察 HUB 圖表,node1 推薦 2,3,4,5,7 這些高 authority 的 node,所以 hub 值相對高於其他 node。node2 只推薦了 node1 這個低 authority 的節點,因此 hub 值相對低於其他 node。

觀察 Authority 圖表,node3,5 兩個點被推薦的次數最多, 因此相較於其他節點 authority 是高的。Node6 只被 Node5 推薦，因此 Node6 之 authority 相較其他節點為最低的。

✧ Graph5

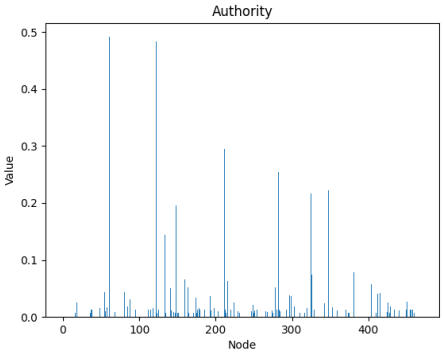
➢ Hub:




```
Hub:
Top1  | Node: 274| Value: 0.19194
Top2  | Node: 176| Value: 0.18981
Top3  | Node: 412| Value: 0.18574
Top4  | Node: 293| Value: 0.17759
Top5  | Node: 254| Value: 0.17468
Top6  | Node: 325| Value: 0.17377
Top7  | Node: 115| Value: 0.17293
Top8  | Node: 182| Value: 0.17095
Top9  | Node: 37| Value: 0.16909
Top10 | Node: 450| Value: 0.16789
Top11 | Node: 299| Value: 0.16523
Top12 | Node: 297| Value: 0.16515
Top13 | Node: 326| Value: 0.16247
Top14 | Node: 416| Value: 0.16247
Top15 | Node: 456| Value: 0.16247
Top16 | Node: 285| Value: 0.16103
Top17 | Node: 224| Value: 0.15385
Top18 | Node: 186| Value: 0.15305
Top19 | Node: 440| Value: 0.15305
Top20 | Node: 179| Value: 0.15286
```

```
HITS_hub:
[0.19194 0.18981 0.18574 0.17759 0.17468 0.17377 0.17293 0.17095 0.16909
 0.16789 0.16523 0.16515 0.16247 0.16247 0.16247 0.16103 0.15385 0.15305
 0.15305 0.15286]
```

➢ Authority:




```
Authority:
Top1  | Node: 61| Value: 0.49135
Top2  | Node: 122| Value: 0.48265
Top3  | Node: 212| Value: 0.29511
Top4  | Node: 104| Value: 0.28670
Top5  | Node: 282| Value: 0.25483
Top6  | Node: 185| Value: 0.25338
Top7  | Node: 348| Value: 0.22195
Top8  | Node: 325| Value: 0.21657
Top9  | Node: 148| Value: 0.19539
Top10 | Node: 134| Value: 0.14463
Top11 | Node: 381| Value: 0.07814
Top12 | Node: 154| Value: 0.07491
Top13 | Node: 326| Value: 0.07441
Top14 | Node: 160| Value: 0.06644
Top15 | Node: 216| Value: 0.06258
Top16 | Node: 404| Value: 0.05705
Top17 | Node: 164| Value: 0.05166
Top18 | Node: 278| Value: 0.05166
Top19 | Node: 141| Value: 0.05104
Top20 | Node: 315| Value: 0.04669
```

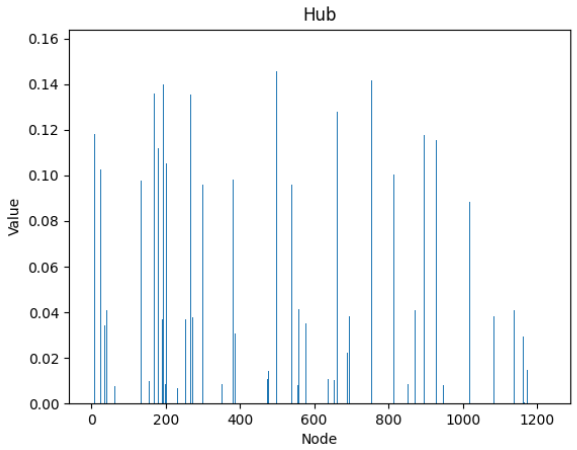
```
HITS_authority:
[0.49135 0.48265 0.29511 0.2867  0.25483 0.25338 0.22195 0.21657 0.19539
 0.14463 0.07814 0.07491 0.07441 0.06644 0.06258 0.05705 0.05166 0.05166
 0.05104 0.04669]
```

➢ Discussion:

1. 觀察 HUB 圖表,node274 推薦 61,104,122…. 這些高 authority 的 node 幾乎都推薦到了,所以 hub 值相對高於其他 node。沒有推薦其他節點的 node, hub 值皆為 0。
2. 觀察 Authority 圖表,node6 被節點 274,176,412…. 推薦，這些高 hub 值的 node 幾乎都推薦到了,所以 authority 值相對高於其他 node。至於沒有被推薦的節點,authority 值為 0。

✧ Graph6

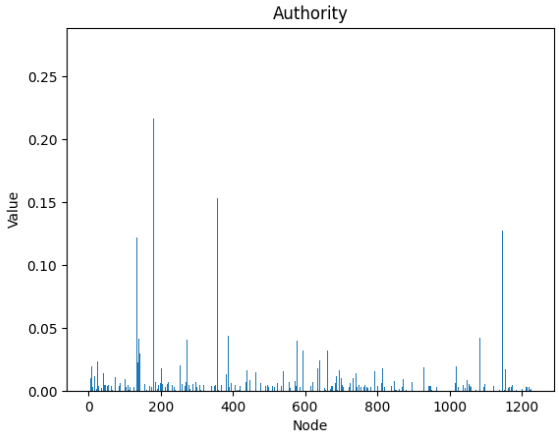
➢ Hub:




```
Hub:
Top1  | Node: 171| Value: 0.15626
Top2  | Node: 857| Value: 0.15014
Top3  | Node: 185| Value: 0.14917
Top4  | Node: 91| Value: 0.14794
Top5  | Node: 79| Value: 0.14753
Top6  | Node: 1199| Value: 0.14579
Top7  | Node: 499| Value: 0.14553
Top8  | Node: 835| Value: 0.14439
Top9  | Node: 386| Value: 0.14339
Top10 | Node: 755| Value: 0.14190
Top11 | Node: 462| Value: 0.14030
Top12 | Node: 8| Value: 0.14007
Top13 | Node: 194| Value: 0.13976
Top14 | Node: 50| Value: 0.13915
Top15 | Node: 140| Value: 0.13847
Top16 | Node: 593| Value: 0.13724
Top17 | Node: 169| Value: 0.13595
Top18 | Node: 267| Value: 0.13557
Top19 | Node: 1150| Value: 0.13484
Top20 | Node: 501| Value: 0.13424
```

```
HITS_hub:
[0.15626 0.15014 0.14917 0.14794 0.14753 0.14579 0.14553 0.14439 0.14339
 0.1419  0.1403  0.14007 0.13976 0.13915 0.13847 0.13724 0.13595 0.13557
 0.13484 0.13424]
```

➢ Authority:




```
Authority:
Top1  | Node: 761| Value: 0.27506
Top2  | Node: 1151| Value: 0.27506
Top3  | Node: 62| Value: 0.27302
Top4  | Node: 78| Value: 0.27169
Top5  | Node: 394| Value: 0.26527
Top6  | Node: 863| Value: 0.25899
Top7  | Node: 1123| Value: 0.25515
Top8  | Node: 501| Value: 0.22971
Top9  | Node: 1052| Value: 0.22255
Top10 | Node: 180| Value: 0.21679
Top11 | Node: 819| Value: 0.18149
Top12 | Node: 506| Value: 0.17382
Top13 | Node: 528| Value: 0.15832
Top14 | Node: 1199| Value: 0.15651
Top15 | Node: 357| Value: 0.15366
Top16 | Node: 1147| Value: 0.12721
Top17 | Node: 1227| Value: 0.12351
Top18 | Node: 134| Value: 0.12191
Top19 | Node: 386| Value: 0.12109
Top20 | Node: 931| Value: 0.11824
```

```
HITS_authority:
[0.27506 0.27506 0.27302 0.27169 0.26527 0.25899 0.25515 0.22971 0.22255
 0.21679 0.18149 0.17382 0.15832 0.15651 0.15366 0.12721 0.12351 0.12191
 0.12109 0.11824]
```

➢ Discussion:

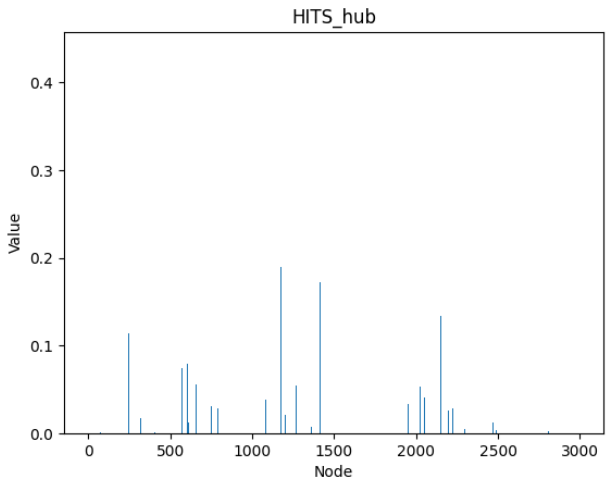
3. 觀察 HUB 圖表,node171 推薦 761,1151,62.... 這些高 authority 的 node 幾乎都推薦到了,所以 hub 值相對高於其他 node。完全沒有推薦其他節點的 node, hub 值皆為 0。

4. 觀察 Authority 圖表,node761 被節點 171,857,185.... 推薦，這些高 hub 值的 node 幾乎都推薦到了,所以 authority 值相對高於其他 node。至於沒有被推薦的節點,authority 值為 0。

✧ Graph7

本資料的構成是由 project1 之 IBM dataset 構成，將每個 transaction 的各個 item 以單向連接到下一個 item，詳見 graph

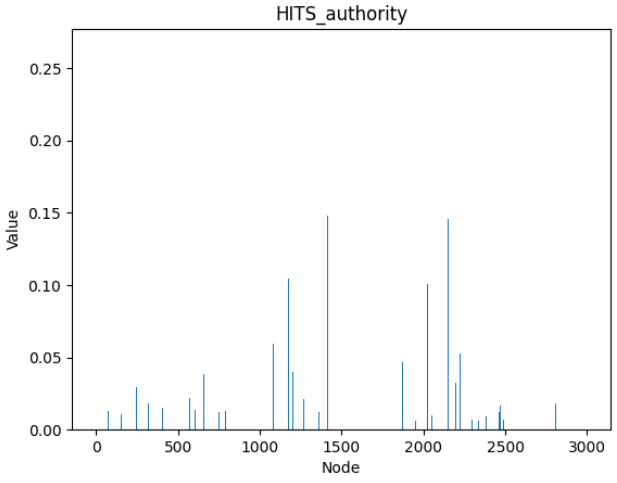
➢ Hub:




```
Hub:
Top1 | Node: 851| Value: 0.43612
Top2 | Node: 1518| Value: 0.31023
Top3 | Node: 1175| Value: 0.19024
Top4 | Node: 1187| Value: 0.17339
Top5 | Node: 1414| Value: 0.17271
Top6 | Node: 1411| Value: 0.16487
Top7 | Node: 1999| Value: 0.15748
Top8 | Node: 1284| Value: 0.15604
Top9 | Node: 1694| Value: 0.15012
Top10 | Node: 1216| Value: 0.14388
Top11 | Node: 274| Value: 0.14333
Top12 | Node: 1499| Value: 0.13749
Top13 | Node: 1767| Value: 0.13398
Top14 | Node: 2151| Value: 0.13354
Top15 | Node: 217| Value: 0.13148
Top16 | Node: 813| Value: 0.11986
Top17 | Node: 1917| Value: 0.11796
Top18 | Node: 246| Value: 0.11474
Top19 | Node: 1957| Value: 0.10992
Top20 | Node: 795| Value: 0.10974
```

```
HITS_hub:
[0.15626 0.15014 0.14917 0.14794 0.14753 0.14579 0.14553 0.14439 0.14339
 0.1419  0.1403  0.14007 0.13976 0.13915 0.13847 0.13724 0.13595 0.13557
 0.13484 0.13424]
```

➢ Authority:




```
Authority:
Top1 | Node: 2285| Value: 0.26433
Top2 | Node: 1694| Value: 0.23294
Top3 | Node: 2165| Value: 0.21395
Top4 | Node: 2203| Value: 0.20730
Top5 | Node: 1518| Value: 0.20233
Top6 | Node: 2173| Value: 0.20036
Top7 | Node: 2112| Value: 0.18488
Top8 | Node: 2053| Value: 0.15245
Top9 | Node: 2268| Value: 0.15227
Top10 | Node: 2214| Value: 0.15186
Top11 | Node: 1414| Value: 0.14785
Top12 | Node: 2151| Value: 0.14561
Top13 | Node: 2648| Value: 0.14122
Top14 | Node: 1411| Value: 0.14024
Top15 | Node: 1999| Value: 0.13999
Top16 | Node: 1499| Value: 0.13979
Top17 | Node: 1775| Value: 0.13815
Top18 | Node: 2677| Value: 0.13615
Top19 | Node: 851| Value: 0.12568
Top20 | Node: 2322| Value: 0.12487
```

```
HITS_authority:
[0.26433 0.23294 0.21395 0.2073  0.20233 0.20036 0.18488 0.15245 0.15227
 0.15186 0.14785 0.14561 0.14122 0.14024 0.13999 0.13979 0.13815 0.13615
 0.12568 0.12487]
```

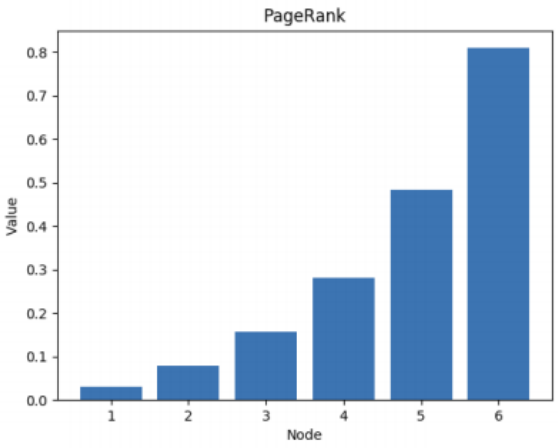
➢ **Discussion:**

5. 觀察 HUB 圖表,node851 推薦 2285,1694,2165…. 這些高 authority 的 node 幾乎都推薦到了,所以 hub 值相對高於其他 node。沒有推薦其他節點的 node, hub 值皆為 0。
6. 觀察 Authority 圖表,node2885 被節點 851,1518,1175…. 推薦，這些高 hub 值的 node 幾乎都推薦到了,所以 authority 值相對高於其他 node。至於沒有被推薦的節點,authority 值為 0。

- **PageRank(參數 d = 0.1)**

✧ **Graph1**

➢ **PageRank:**




```
Page Rank:
Top1 | Node: 6| Value: 0.80986
Top2 | Node: 5| Value: 0.48352
Top3 | Node: 4| Value: 0.28121
Top4 | Node: 3| Value: 0.15580
Top5 | Node: 2| Value: 0.07806
Top6 | Node: 1| Value: 0.02987
```

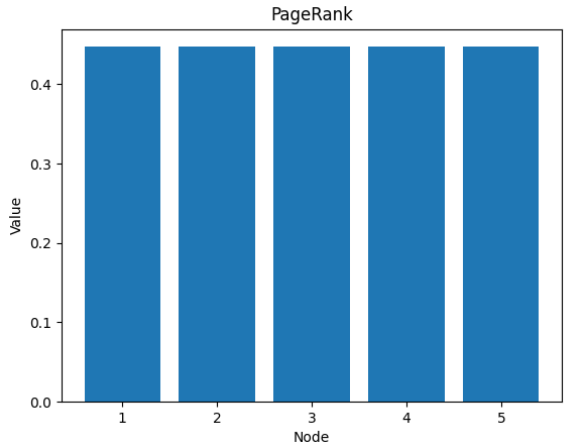
```
PageRank:
[0.80986 0.48352 0.28121 0.1558  0.07806 0.02987]
```

➢ **Discussion:**

由於 graph1 是單向往前指的架構，因此坐落在起頭的 Node1 沒有任何的父節點，PageRank 只有 d/n 在貢獻， Node2 的話除了 d/n 以外，還可以額外繼承父節點的 PageRank，因此 Node2 PageRank 比起 Node1 更多了，依此類推，愈往後面的節點 PageRank 就愈高。

✧ Graph2

➢ PageRank:




```
Page Rank:
Top1 | Node: 1| Value: 0.44721
Top2 | Node: 2| Value: 0.44721
Top3 | Node: 3| Value: 0.44721
Top4 | Node: 4| Value: 0.44721
Top5 | Node: 5| Value: 0.44721
```

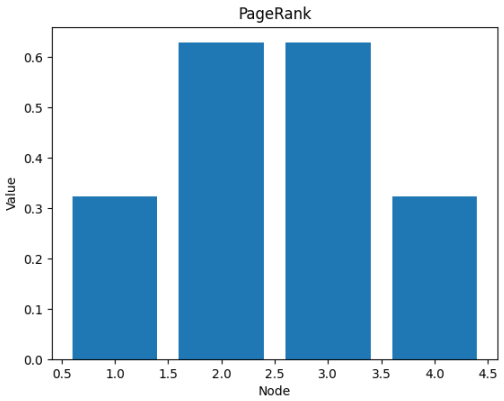
```
PageRank:
[0.44721 0.44721 0.44721 0.44721 0.44721]
```

➢ Discussion:

1. 因為這個 graph 是一個環狀架構，每個節點都是被前一個節點指向，同時也指向下一個節點，沒有任何差異，因此 PageRank 大家都是一致。
2. 如果將此環狀架構打破一個 edge 那麼就會跟 graph1 一樣，愈末端的節點 PageRank 值會愈來愈高。

✧ Graph3

➢ PageRank:



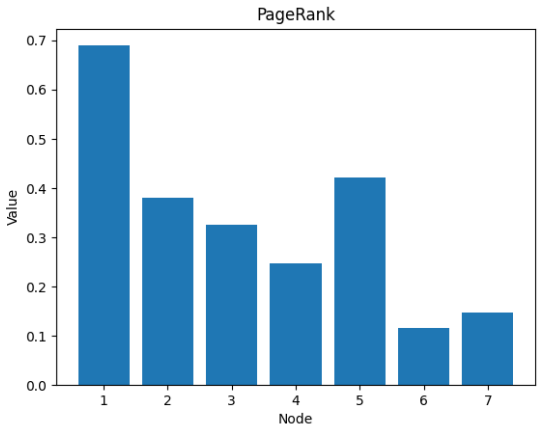

```
Page Rank:
Top1 | Node: 2| Value: 0.62885
Top2 | Node: 3| Value: 0.62885
Top3 | Node: 1| Value: 0.32334
Top4 | Node: 4| Value: 0.32334
```

```
PageRank:
[0.62885 0.62885 0.32334 0.32334]
```

- Discussion:
  1. 若將 Node2 .5 之 edge 當分界分為兩個子圖，會 發現 Node2 這側的子圖與 Node3 這側的子圖是一模一樣，因此可以預期其 PageRank 會有對稱的情形發生。
  2. 整體架構因為也是對稱的狀況，Node1 對應上 Node4 及 Node2 對應上 Node3 ，因此 Node1 及 Node4 之 PageRank 互相一致， Node2 及 Node3 之 PageRank 互相一致。
  3. 對於 Node2 來說，因為 Node1 對外只指向 Node2 ，因此 Node2 可以完全繼承 Node1 的 PageRank ，又能繼承部份 Node3 的 PageRank，但對於 Node1 來說，因為 Node2 對外有連接到 Node1 及 Node3 ，因此只能繼承一半的 Node2 PageRank，故最後的 PageRank 的表現上 Node1 會較 Node2 少。

✧ Graph4

- PageRank:




```
Page Rank:
Top1 | Node: 1| Value: 0.68973
Top2 | Node: 5| Value: 0.42070
Top3 | Node: 2| Value: 0.38094
Top4 | Node: 3| Value: 0.32583
Top5 | Node: 4| Value: 0.24719
Top6 | Node: 7| Value: 0.14681
Top7 | Node: 6| Value: 0.11553
```

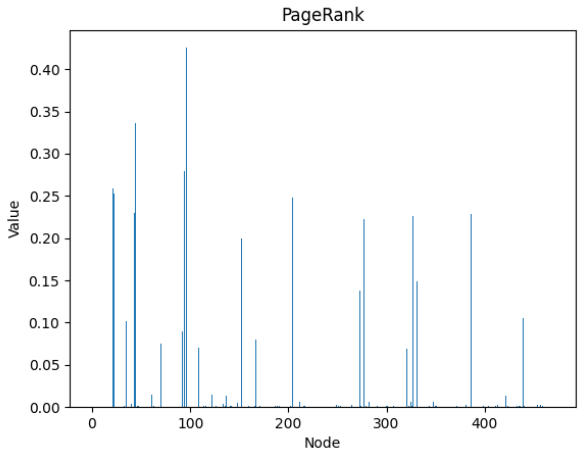
```
PageRank:
[0.68973 0.4207  0.38094 0.32583 0.24719 0.14681 0.11553]
```

- Discussion:

1. 觀察圖形連接關係，由於 Node1 被 Node2, 3, 5, 6 連接了，而這些連接 Node1 的節點除了 Node5 以外，對外的分支數都在 1 至 2 個左右，因此比較能夠承接各 個節點完整的 PageRank ，故 Node1 在 PageRank 上的表現勝過其他節點。

✧ Graph5

➢ PageRank:




```
Page Rank:
Top1  | Node: 96| Value: 0.42530
Top2  | Node: 44| Value: 0.33663
Top3  | Node: 24| Value: 0.30515
Top4  | Node: 94| Value: 0.27994
Top5  | Node: 21| Value: 0.25852
Top6  | Node: 22| Value: 0.25318
Top7  | Node: 204| Value: 0.24792
Top8  | Node: 43| Value: 0.23022
Top9  | Node: 386| Value: 0.22897
Top10 | Node: 327| Value: 0.22667
Top11 | Node: 277| Value: 0.22243
Top12 | Node: 152| Value: 0.19906
Top13 | Node: 331| Value: 0.14885
Top14 | Node: 273| Value: 0.13796
Top15 | Node: 439| Value: 0.10548
Top16 | Node: 35| Value: 0.10181
Top17 | Node: 92| Value: 0.08905
Top18 | Node: 167| Value: 0.07953
Top19 | Node: 70| Value: 0.07546
Top20 | Node: 109| Value: 0.07011
```

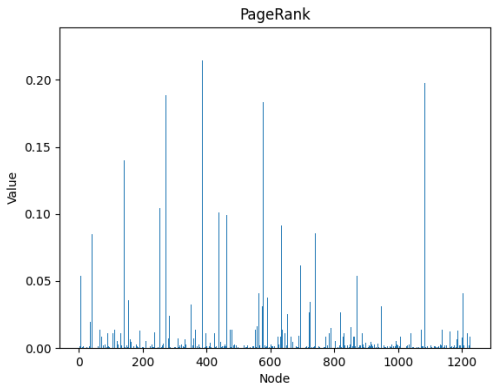
```
PageRank:
[0.4253  0.33663 0.30515 0.27994 0.25852 0.25318 0.24792 0.23022 0.22897
 0.22667 0.22243 0.19906 0.14885 0.13796 0.10548 0.10181 0.08905 0.07953
 0.07546 0.07011]
```

➢ Discussion:

1. 觀察 graph5 圖， Node96 被 109 (out-degree = 6), 273 (out-degree = 8), 386 (out-degree = 9) 連接，以這圖形來說，扣除 out-degree 為 0 的 Node ，平均每個 Node 之 out-degree 為 9.33898 ，又平均每個 Node 之 in- degree 為 2.34968 ，按照 Node96 被連接的狀況，in-degree 高於平均值甚 多，且其父親節點的分支數又低於平均值，因此 Node96 在 PageRank 表現上 勝過其他節點。

✧ Graph6

➢ PageRank:




```
Page Rank:
Top1  | Node: 410| Value: 0.22794
Top2  | Node: 1052| Value: 0.22308
Top3  | Node: 374| Value: 0.21416
Top4  | Node: 387| Value: 0.21416
Top5  | Node: 1021| Value: 0.21416
Top6  | Node: 554| Value: 0.21170
Top7  | Node: 43| Value: 0.20534
Top8  | Node: 415| Value: 0.20455
Top9  | Node: 468| Value: 0.20444
Top10 | Node: 494| Value: 0.19918
Top11 | Node: 1084| Value: 0.19753
Top12 | Node: 946| Value: 0.19214
Top13 | Node: 273| Value: 0.18862
Top14 | Node: 220| Value: 0.18738
Top15 | Node: 578| Value: 0.18363
Top16 | Node: 1134| Value: 0.18317
Top17 | Node: 367| Value: 0.17571
Top18 | Node: 848| Value: 0.17456
Top19 | Node: 95| Value: 0.14262
Top20 | Node: 142| Value: 0.14028
```

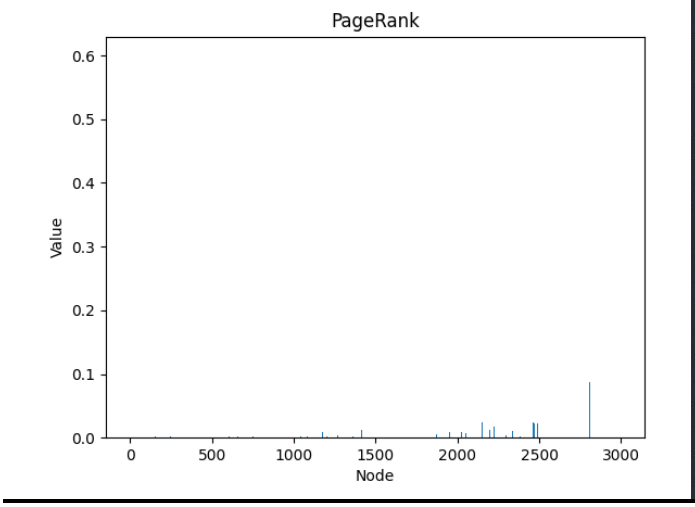
```
PageRank:
[0.22794 0.22308 0.21416 0.21416 0.21416 0.2117  0.20534 0.20455 0.20444
 0.19918 0.19753 0.19214 0.18862 0.18738 0.18363 0.18317 0.17571 0.17456
 0.14262 0.14028]
```

➢ Discussion:

1. 觀察 graph6 圖， Node2908 被 Node1997 (out-degree = 3), 851 (out-degree = 69), 2285 (out-degree = 43) 連接，以這圖形來說，扣除 out- degree 為 0 的 Node ，平均每個 Node 之 out-degree 為 27.91444 ，又平均 每個 Node 之 in-degree 為 4.25081 ，按照 Node410 被連接的狀況，in- degree 高於平均值甚多，且其父親節點的分支數接近於平均值，因此 Node2908 在 PageRank 表現上勝過其他節點。

✧ Graph7

➢ PageRank:




```
Page Rank:
Top1  | Node: 2908| Value: 0.59970
Top2  | Node: 2878| Value: 0.38999
Top3  | Node: 2897| Value: 0.27057
Top4  | Node: 2858| Value: 0.26490
Top5  | Node: 2976| Value: 0.25690
Top6  | Node: 2993| Value: 0.24838
Top7  | Node: 2893| Value: 0.23913
Top8  | Node: 2824| Value: 0.19624
Top9  | Node: 2677| Value: 0.10914
Top10 | Node: 2732| Value: 0.10195
Top11 | Node: 2869| Value: 0.09342
Top12 | Node: 2808| Value: 0.08652
Top13 | Node: 2722| Value: 0.08530
Top14 | Node: 2595| Value: 0.08037
Top15 | Node: 2842| Value: 0.07534
Top16 | Node: 2625| Value: 0.07474
Top17 | Node: 2508| Value: 0.06982
Top18 | Node: 2647| Value: 0.05960
Top19 | Node: 2684| Value: 0.05820
Top20 | Node: 2285| Value: 0.05534
```

```
PageRank:
[0.5997  0.38999 0.27057 0.2649  0.2569  0.24838 0.23913 0.19624 0.10914
 0.10195 0.09342 0.08652 0.0853  0.08037 0.07534 0.07474 0.06982 0.0596
 0.0582  0.05534]
```

➢ Discussion:

1. 觀察 graph7 圖， Node2908 被 Node1 (out-degree = 26), 36 (out-degree = 25), 41 (out-degree = 29) 連接，以這圖形來說，扣除 out- degree 為 0 的 Node ，平均每個 Node 之 out-degree 為 27.91444 ，又平均 每個 Node 之 in-degree 為 4.25081 ，按照 Node410 被連接的狀況，in- degree 高於平均值甚多，且其父親節點的分支數接近於平均值，因此 Node410 在 PageRank 表現上勝過其他節點。

- **SimRank** (參數 C = 1.0 及 C = 0.8)
  - ✧ **Graph1**
  - ➢ **SimRank:**

```
Similarity Rank:
SimRank(1, 1) = 1.0000
SimRank(1, 2) = 0.0000
SimRank(1, 3) = 0.0000
SimRank(1, 4) = 0.0000
SimRank(1, 5) = 0.0000
SimRank(1, 6) = 0.0000
SimRank(2, 2) = 1.0000
SimRank(2, 3) = 0.0000
SimRank(2, 4) = 0.0000
SimRank(2, 5) = 0.0000
SimRank(2, 6) = 0.0000
SimRank(3, 3) = 1.0000
SimRank(3, 4) = 0.0000
SimRank(3, 5) = 0.0000
SimRank(3, 6) = 0.0000
SimRank(4, 4) = 1.0000
SimRank(4, 5) = 0.0000
SimRank(4, 6) = 0.0000
SimRank(5, 5) = 1.0000
SimRank(5, 6) = 0.0000
SimRank(6, 6) = 1.0000
```

Discussion:

- **參數 C 之設定 1.0 與 0.8 ，結果皆為一樣的。**
- **由於 graph1 是單向往前指的架構，完全不存在有共同父節點的情況 (out-degree 最多只有 1)，因此只有相同節點的 SimRank 為 1 ，其他倆倆組合的節點 SimRank 都為 0 。**

  - ✧ **Graph2**
  - ➢ **SimRank:**

```
Similarity Rank:
SimRank(1, 1) = 1.0000
SimRank(1, 2) = 0.0000
SimRank(1, 3) = 0.0000
SimRank(1, 4) = 0.0000
SimRank(1, 5) = 0.0000
SimRank(1, 6) = 0.0000
SimRank(2, 2) = 1.0000
SimRank(2, 3) = 0.0000
SimRank(2, 4) = 0.0000
SimRank(2, 5) = 0.0000
SimRank(2, 6) = 0.0000
SimRank(3, 3) = 1.0000
SimRank(3, 4) = 0.0000
SimRank(3, 5) = 0.0000
SimRank(3, 6) = 0.0000
SimRank(4, 4) = 1.0000
SimRank(4, 5) = 0.0000
SimRank(4, 6) = 0.0000
SimRank(5, 6) = 0.0000
SimRank(6, 6) = 1.0000
```

Discussion:

- **由於 graph 2 是一個環狀架構，但狀況跟 graph1 一樣，完全不存在有共同父節點的情況 ，因此只有相同節點的 SimRank 為 1 ，其他倆倆組合的節點 SimRank 都為 0 。**

✧ Graph3

➢ SimRank:

```
Similarity Rank:
SimRank(1, 1) = 1.0000
SimRank(1, 2) = 0.0000
SimRank(1, 3) = 0.6667
SimRank(1, 4) = 0.0000
SimRank(2, 2) = 1.0000
SimRank(2, 3) = 0.0000
SimRank(2, 4) = 0.6667
SimRank(3, 3) = 1.0000
SimRank(3, 4) = 0.0000
SimRank(4, 4) = 1.0000
```

```
Distance Base Similarity
SimRank(1, 1) = 1.0000
SimRank(1, 2) = 0.3847
SimRank(1, 3) = 0.1146
SimRank(1, 4) = 0.0307
SimRank(2, 2) = 1.0000
SimRank(2, 3) = 0.3846
SimRank(2, 4) = 0.1146
SimRank(3, 3) = 1.0000
SimRank(3, 4) = 0.3847
SimRank(4, 4) = 1.0000
```

Discussion:

1. **因為這個 graph 的特性，對於最短路徑為基數個 edge 才能到達的節點配對 (如 (1,2)、(1,4)、(2,3)、(3、4))，其父節點仍然是互相隔了基數個 edge ，故這樣 的節點配對，即使不斷的遞迴去找共同的父節點也不可能會找到，因為基數個 edge 才能到達的節點配對其父節點必定隔了基數個 edge 。**
2. **而相隔偶數 edge 的節點配對，其父節點也是相隔偶數 (包含相隔 0 個) edge ，因此只要是相隔偶數 edge 的配對不斷遞迴找下去一定都能找到共同父親節 點，而在 C = 1 的設定下，可以想像成 SimRank 不會因為遞迴的深度而衰減， 因此相隔偶數 edge 的配對其 SimRank 為 1 。**
3. **然而我們修改 C = 0.8 時，發現 SimRank(1, 3) 及 SimRank(2, 4) 不再是 1 了， 這顯示了當配對節點要從父節點繼承 SimRank 時，需透過折減係數 C 來進行折 減，因此繼承而來的 SimRank 會隨著遞迴深度而衰減。**

✧ Graph4

➢ **SimRank:**

```
Similarity Rank:
SimRank(1, 1) = 1.0000
SimRank(1, 2) = 0.3603
SimRank(1, 3) = 0.3490
SimRank(1, 4) = 0.3537
SimRank(1, 5) = 0.3377
SimRank(1, 6) = 0.4151
SimRank(1, 7) = 0.2924
SimRank(2, 2) = 1.0000
SimRank(2, 3) = 0.4068
SimRank(2, 4) = 0.3697
SimRank(2, 5) = 0.4122
SimRank(2, 6) = 0.2854
SimRank(2, 7) = 0.4541
SimRank(3, 3) = 1.0000
SimRank(3, 4) = 0.4496
SimRank(3, 5) = 0.3901
SimRank(3, 6) = 0.4481
SimRank(3, 7) = 0.4510
SimRank(4, 4) = 1.0000
SimRank(4, 5) = 0.3427
SimRank(4, 6) = 0.5351
SimRank(4, 7) = 0.5351
SimRank(5, 5) = 1.0000
SimRank(5, 6) = 0.2731
SimRank(5, 7) = 0.4122
SimRank(6, 6) = 1.0000
SimRank(6, 7) = 0.2701
SimRank(7, 7) = 1.0000
```

```
Distance Base Similarity Rank:
SimRank(1, 1) = 1.0000
SimRank(1, 2) = 0.4062
SimRank(1, 3) = 0.4071
SimRank(1, 4) = 0.2719
SimRank(1, 5) = 0.4058
SimRank(1, 6) = 0.2598
SimRank(1, 7) = 0.2468
SimRank(2, 2) = 1.0000
SimRank(2, 3) = 0.2935
SimRank(2, 4) = 0.2697
SimRank(2, 5) = 0.1554
SimRank(2, 6) = 0.0682
SimRank(2, 7) = 0.0930
SimRank(3, 3) = 1.0000
SimRank(3, 4) = 0.2938
SimRank(3, 5) = 0.2951
SimRank(3, 6) = 0.0977
SimRank(3, 7) = 0.1224
SimRank(4, 4) = 1.0000
SimRank(4, 5) = 0.4085
SimRank(4, 6) = 0.1222
SimRank(4, 7) = 0.0945
SimRank(5, 5) = 1.0000
SimRank(5, 6) = 0.3852
SimRank(5, 7) = 0.2654
SimRank(6, 6) = 1.0000
SimRank(6, 7) = 0.0927
SimRank(7, 7) = 1.0000
```

Discussion:

1. **個別討論幾種配對 SimRank 非 0 的原因:**

   *** SimRank(1, 2): 透過共同父節點 Node3 。**

   *** SimRank(2, 4): 透過共同父節點 Node1 。**

   *** SimRank(3, 6): 透過共同父節點 Node5 。**

   *** SimRank(4, 5): 透過共同父節點 Node1 。**

   *** SimRank(5, 6): Node5 取出一個父節點 Node1 ， Node6 取出一個父節點 Node5 ，而依據前面的結論 SimRank(1, 5) 為非 0。**

   *** SimRank(6, 7): Node6 取出一個父節點 Node5 ， Node7 取出一個父節點 Node1 ，而依據前面的結論 SimRank(1, 5) 為非 0 。**

2. 因此扣除相同節點配對的狀況，依照上面的討論，其他的配對 SimRank 皆不為 0 ，而且還互相糾纏，也就是說自己本身的 SimRank 已經不為 0 的狀況，還可 以在繼續接收其父親節點配對產生的 SimRank ，因此這樣的相關性在 C = 1.0 的狀況下 (不會因遞迴深度而損失傳遞的相關性) 最後會收斂到大家都是 1 。
3. 而由 C = 0.8 的數據中我們可以看到有些非零的 SimRank 不再是 1 了，主要是 在於遞迴深度使得 SimRank 在繼承父節點配對時產生了衰減，而 SimRank 愈 接近 0 者，表示這兩兩配對的節點要找到相關性，必須通過好幾曾的遞迴才能 找到共同的父節點。

✧ Graph5

➢ SimRank:

```
SimRank(135, 373) = 0.0000
SimRank(135, 374) = 0.1789
SimRank(135, 375) = 0.1789
SimRank(135, 376) = 0.0000
SimRank(135, 377) = 0.0000
SimRank(135, 378) = 0.0000
SimRank(135, 379) = 0.0000
SimRank(135, 380) = 0.0000
SimRank(135, 381) = 0.2327
SimRank(135, 382) = 0.0000
SimRank(135, 383) = 0.0000
SimRank(135, 384) = 0.0000
SimRank(135, 385) = 0.0000
SimRank(135, 386) = 0.0000
SimRank(135, 387) = 0.0000
SimRank(135, 388) = 0.0000
SimRank(135, 389) = 0.0000
SimRank(135, 390) = 0.0000
SimRank(135, 391) = 0.1789
SimRank(135, 392) = 0.0000
SimRank(135, 393) = 0.0000
SimRank(135, 394) = 0.0000
SimRank(135, 395) = 0.0000
SimRank(135, 396) = 0.0000
SimRank(135, 397) = 0.0000
SimRank(135, 398) = 0.0000
SimRank(135, 399) = 0.0000
SimRank(135, 400) = 0.0000
SimRank(135, 401) = 0.0000
SimRank(135, 402) = 0.0000
```

Discussion:

- 由於配對數量過多 (約 11 萬種配對)，因此詳細 SimRank 放在 result/simrank_5_c10.txt result/simrank_5_c08.txt 。
- 由結果可以發現，SimRank 分佈開始有很多配對為 0 ，這可能說明著這個 graph 連接的結構沒有辦法讓 SimRank 傳遞到每種配對上，如果把這些 SimRank 為 0 的配對再多個幾條指進來的 edge，應該是會有效的提升 SimRank ，因為這配對可以再透過這些額外連進來的 edge 來去讓其他的父節 點配對傳遞非 0 的 SimRank 回來，這樣一來原本為 0 的配對就不會為 0 了。

✧ **Graph6**

➢ **SimRank:**

配對數量太多

詳見/output /graph_6/graph_6_SimRank.txt

How to Increase Hub, Authority and PageRank?[5]

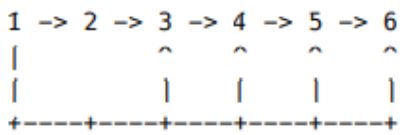
➢ More limitations about link analysis algorithms?

Hub

Graph1

▪ Origin

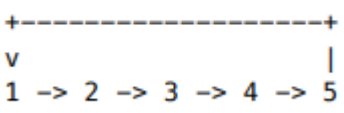
▪ Modify



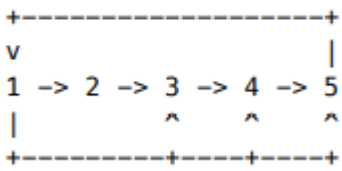
- 我們可以將 Node1 指向其他的節點，使得其他節點的 Authority 可以分享給 Node1 Hub。
- 實驗結果 Node1 之 Hub 原為 0.44721 ，提升為 0.92388 。

Graph2

▪ Origin



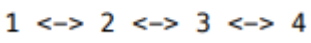
▪ Modify



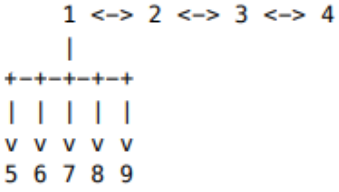
- 將 Node1 指向其他的節點，使得其他節點的 Authority 可以分享給 Node1 Hub。
- 實驗結果 Node1 之 Hub 原為 0.44721 ，提升為 0.90958 。

Graph3

▪ Origin



▪ Modify



- 可以新增 Node5 ~ Node9 將 Node1 指向新增的節點，便能提升 Node1 之 Hub 值。
- 實驗結果 Node1 之 Hub 原為 0.60150 ，提升為 0.97325 。

Authority

Graph1

- Origin

```
1 -> 2 -> 3 -> 4 -> 5 -> 6
```

- Modify

```
1 -> 2 -> 3 -> 4 -> 5 -> 6
^^
| \
7  8
```

- 可以透過新增 Node7 及 Node8 指向 Node1 便能提升 Node1 之 Authority 值。
- 實驗結果 Node1 之 Authority 原為 0.0 ，提升為 1.0 。

Graph2

- Origin

```
+-------------------+
v                   |
1 -> 2 -> 3 -> 4 -> 5
```

- Modify

```
+-------------------+
v                   |
1 -> 2 -> 3 -> 4 -> 5
^
|
6
```

- 可以透過新增 Node6 指向 Node1 便能提升 Node1 之 Authority 值。
- 實驗結果 Node1 之 Authority 原為 0.44721 ，提升為 1.0 。

Graph3

- Origin

```
1 <-> 2 <-> 3 <-> 4
```

- Modify

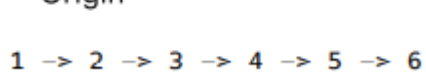
```
  1 <-> 2 <-> 3 <-> 4
  ^
 /|\
5 6 7
```

- **可以新增 Node5 ~ Node7 將 Node1 指向新增的節點，便能提升 Node1 之 Authority 值。**
- **實驗結果 Node1 之 Authority 原為 0.37175 ，提升為 0.92388 。**

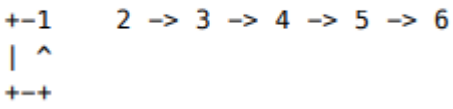
PageRank

Graph1

- Origin



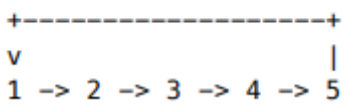
- Modify



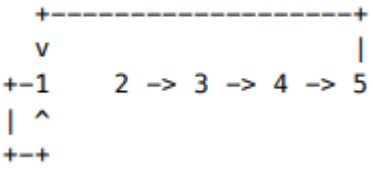
- 可以刪除 1->2 之 edge ，新增 1->1 之 edge ，使得 Node1 在 iteration 中 share PageRank 給 child 時只會分配給自己。
- 實驗結果 Node1 之 PageRank 原為 0.02987 ，提升為 0.99137。

Graph2

- Origin



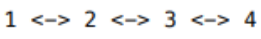
- Modify



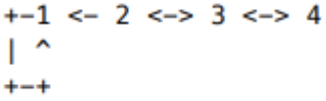
- 可以刪除 1->2 之 edge ，新增 1->1 之 edge ，使得 Node1 在 iteration 中 share PageRank 給 child 時只會分配給自己。
- 實驗結果 Node1 之 PageRank 原為 0.44721 ，提升為 0.99492。

Graph3

- Graph3
  - Origin



- Modify



- 我們可以刪除 1->2 之 edge ，新增 1->1 之 edge ，使得 Node1 在 iteration 中 share PageRank 給 child 時只會分配給自己。
- 實驗結果 Node1 之 PageRank 原為 0.32334 ，提升為 0.97425 。

Discussion[6]

- What are practical issues when implement these algorithms in a real Web?

想把這要應用到實際上網站上，面臨最關鍵的問題在於計算時間，這樣的分析所需計算時間會隨著節點數及邊數的提升而有快速成長的趨勢，而真實的搜尋引擎要處理的網站節點量可想而知是超出我們實驗範圍很多的，像是 google 隨便搜尋一個辭彙都會在 0.5 秒鐘內找出 5,000 多萬以上個符合網站，如此才能 提供使用者一個可接受的搜尋，但我們實驗中 |V| = 10,000, |E| = 10,000,000 就要跑個 半分鐘以上，實在是很難被使用者所接受，因此要把這樣的方法應用到實際的運作上存在難以克服的難度。

- Any new idea about the link analysis algorithm?

link analysis algorithm 面臨最大的問題在於運算時間需要很大的負擔，在面臨真 實網站處理時，這麼漫長的處理時間是不被允許的，或許我們可以先行依據 similarity 進行網站的分群，把相似度高的網站組合成群，如此一來搜尋引擎在搜尋網站 時可以直接計算每個大群的 PageRank 或是 Authority ，在近一步到這些評分較高的網 站群內，再做群內部的 PageRank 或是 Authority 計算工作，這或許會舒緩龐大的運算量。

- What is the effect of “C” parameter in SimRank?

觀察前面的 result ，在兩兩配對的節點計算 SimRank 時，會因為兩節點想透過遞迴去找尋到相同的父節點，但每通過一層的遞迴所繼承而來的 SimRank 都會被 C 值所減少，因此隨著 C 值的設定愈接近 0 的話，兩兩不同的節點互相配對其 SimRank 值必定 會愈接近 0 ，而 C 值愈接近 1 的話，兩兩不同的節點互相配對，在追溯到共同父節點 時，其繼承的 SimRank 將不會有折減。

- What you learned from this project and your comments about this project?

實作出來的演算法計算速度遠遠不足實際上搜尋引擎的要求。

在 project 中，我瞭解到搜尋引擎的運作原理。可以透過一些手段提升這些演算法分析出來的 rank 。所以在搜尋引擎所找到的網站未必是真正重要的，這些網站也許是用了某些手段提升了 rank 使得搜尋引擎比較推薦。