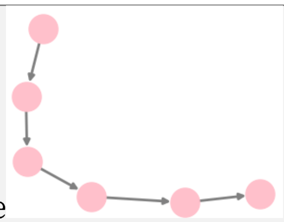
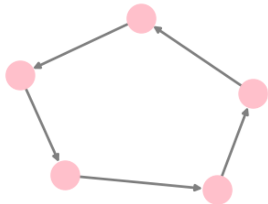
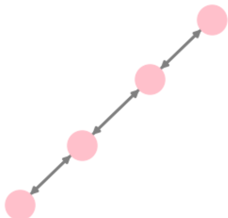
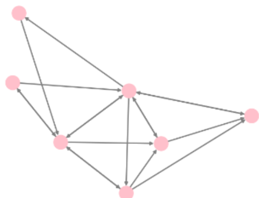
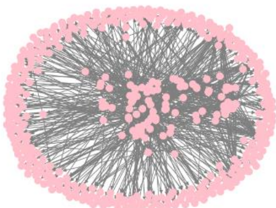
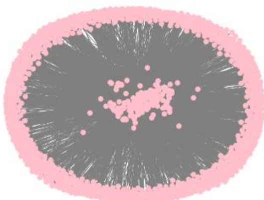
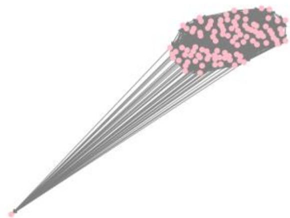
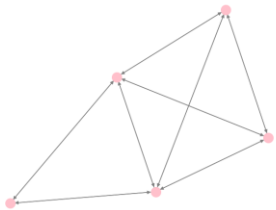
Data Mining -Project 3 Link Analysis

電通甲 Q36061240 冼鈺淇

Introduction of Dataset

■

GRAPH NAME	NODES	EDGES	DESCRIPTION	SOURCE
GRAPH_1	6	5	A Line 	project3dataset
GRAPH_2	5	5	A circle 	project3dataset
GRAPH_3	4	6		project3dataset
GRAPH_4	7	18		project3dataset
GRAPH_5	469	1102		project3dataset
GRAPH_6	1128	5220		project3dataset

GRAPH_7	100	2043		project1 transaction data
GRAPH_8	5	17		project1 association rules mined

Algorithms Implement

■ Convert Format

將 HW1 的 data 轉換為代表 edge 之輸入格式 edge1:(node1,node2)，我選用的 dataset 參數為 ntrans=0.1,tlen=40,nitems=0.1 以及其 association rule(support =0.75 confidence =0.85)

```
def convertIBMdata():
    IBM = open( 'data.ntrans_0.1.tlen_40.nitems_0.1.txt','r')
    f = open('graph_7.txt','w')
    lines=IBM.readlines()

    for line in lines:
        print(line.split()[1]+','+line.split()[2])
        f.write(line.split()[1]+','+line.split()[2]+'\n')
        pass
    IBM.close()
    f.close()
```

```
def convertRuledata():
    Rule = open( 'association_rules.txt','r')
    f = open('graph_8.txt','w')
    lines=Rule.readlines()

    for line in lines:
        print(line.split()[0]+','+line.split()[1])
        f.write(line.split()[0]+','+line.split()[1]+'\n')
        pass
    Rule.close()
    f.close()
```

```
#DATASET1 IBM generator -FP_GROWTH ALGO
tStart = time.time()
print('\nFP_Growth algorithm')
Run_FP_Growth('data.ntrans_0.1.tlen_40.nitems_0.1DAT.txt',0.75,0.75)
tEnd = time.time()
print ("%f sec" % (tEnd - tStart))

'''
```

```
FP_Growth algorithm
======================================================
Frequent itemset:
(17)  support = 0.77
(36)  support = 0.88
(36, 38)  support = 0.84
(36, 63)  support = 0.79
(36, 63, 38)  support = 0.75
(38)  support = 0.94
(63)  support = 0.89
(63, 38)  support = 0.84
(69)  support = 0.84
(69, 38)  support = 0.8
(69, 63)  support = 0.76
(8)  support = 0.75
(87)  support = 0.87
(87, 36)  support = 0.77
(87, 36, 38)  support = 0.75
(87, 38)  support = 0.83
(87, 63)  support = 0.79
(87, 63, 38)  support = 0.76
======================================================
(36) ==> (38 ) confidence= 0.95
(38) ==> (36 ) confidence= 0.89
(36) ==> (63 ) confidence= 0.90
(63) ==> (36 ) confidence= 0.89
(36) ==> (63 38 ) confidence= 0.85
(63) ==> (36 38 ) confidence= 0.84
```

■ Adjacency Matrix

Input data 格式為 *edge1:(node1,node2)*有相連的 node 編號，計算出有幾個 node 並將此轉換成各圖的相鄰矩陣，以下我實作的演算法都會基於此相鄰矩陣作計算。

```python
def adjMetrix(edgeList):
    m = Nodenum(edgeList)
    matrix = np.zeros(shape=(m,m))
    for edge in edgeList:
        u = int(edge[0]) - 1
        v = int(edge[1]) - 1
        matrix[u][v] = 1
    return matrix
```

```python
def Nodenum(edgeList):
    nodeList = []
    for edge in edgeList:
        for node in edge:
            if node not in nodeList:
                nodeList.append(node)
    return len(nodeList)
```

■ HITS

$$\mathrm{auth}(p) = \sum_{i=1}^{n} \mathrm{hub}(i) \qquad \mathrm{hub}(p) = \sum_{i=1}^{n} \mathrm{auth}(i)$$

實作 HITS 演算法，在這邊我的 hub 值都設 1，然後輸入 adjmatrix 依照公式用迴圈迭代至收斂，每做一次就正規化（除以全部的 hub 值/向量長度）一次，收斂的話就是直接看新的值與舊值差多少來判定。

```python
def Converge_checking(vector,vector_pre):
    converge = False
    tolerance = 0.000005
    changes = 0
    m = len(vector)

    for i in range(0,m):
        changes += abs(vector[i] - vector_pre[i])

    if changes < tolerance:
        converge = True
    return converge
```

```python
def HITS(adjmatrix):
    m,m = adjmatrix.shape

    #hub 值都先設1
    hub = np.ones(shape=(m,1))
    hub_prev = np.zeros(shape=(m,1))
    #print(hub,hub_prev)

    #authority = A^T * hub, hub = A * authority
    authority = np.dot(adjmatrix.transpose(),hub)
    hub = np.dot(adjmatrix,authority)

    # check converge
    while not Converge_checking(hub, hub_prev):
        hub_prev = hub

        authority = np.dot(adjmatrix.transpose(),hub)
        hub = np.dot(adjmatrix,authority)

        # normalize
        hub = hub/hub.sum()
        authority = authority/authority.sum()

    return hub, authority
```

■ PageRank

$$PR(p_i) = \frac{1-d}{N} + d \sum_{p_j \in M(p_i)} \frac{PR(p_j)}{L(p_j)}$$

依照 PageRank 公式 N 為 node 數量、d 為 dumping factor 、L(pj) = 為 pj 這個 node 的 outdegree，PR 是(m,1)的向量，每個元素的值皆為 1/N，做迴圈迭代，收斂完後進行正規化。

```python
def PageRank(adjmatrix,d):
    m,m = adjmatrix.shape


    outdegree = adjmatrix.sum(axis=1)
    PR = np.zeros(shape=(m,1))
    PR_prev = np.zeros(shape=(m,1))
    PR_Div_outdeg = np.zeros(shape=(m,1))
    PR_Div_outdeg_sum = np.zeros(shape=(m,1))

    PR = [ 1/m for value in PR]


    while not Converge_checking(PR, PR_prev):
        PR_prev = PR


        for i in range(0,m):
            if outdegree[i]!=0:
                PR_Div_outdeg[i] = PR[i] / outdegree[i]

        for i in range(0,m):
            PR_Div_outdeg_sum_value = 0
            for j in range(0,m):
                if adjmatrix[j][i] == 1:
                    PR_Div_outdeg_sum_value +=  PR_Div_outdeg[j]
            PR_Div_outdeg_sum[i] = PR_Div_outdeg_sum_value

        PR = d / m + (1-d) * PR_Div_outdeg_sum
    #normolized
    PR = PR/PR.sum()
    return PR
```

■ SimRank

$$\mathbf{S} = C \cdot (\mathbf{W}^T \cdot \mathbf{S} \cdot \mathbf{W}) + (1 - C) \cdot \mathbf{I},$$

SimRank 的公式，W 為 normalize 後的圖鄰接矩陣、I 為單位矩陣，再我實作的 simrank 演算法中 iteration 次數設為 100，C 設 0.8，一開始設 S 相似度矩陣為單位矩陣再讓公式下去迭代產出最後的相似度矩陣。

```python
def SimRank(G,n,C=0.8,t=100):

    S = np.identity(n)
    I = np.identity(n)
    G = G/G.sum(0)
    i = 1

    # S(a,b) ifa=b S設1
    for a in range(t):
        S = C * np.dot(np.dot(G.T,S),G) + (1-C) * I
    for j in range(n):
        S[j][j] = 1

    return S
```

Result Analysis

➢ **Parameter Setting: PageRank dumpy factor=1.5/SimRank C=0.8**

以下使用 networkX 套件中的 HITS、PageRank 演算法來驗證及比較程式結果。

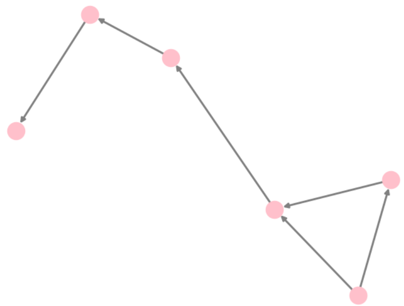
■ Graph_1

```
Graph Dataset :graph_1.txt

Adjacency Matrix :
[[0. 1. 0. 0. 0. 0.]
 [0. 0. 1. 0. 0. 0.]
 [0. 0. 0. 1. 0. 0.]
 [0. 0. 0. 0. 1. 0.]
 [0. 0. 0. 0. 0. 1.]
 [0. 0. 0. 0. 0. 0.]]

Implement >
     Hub Authority                 PageRank                          SimRank
0  [0.2]      [0.0]  [0.06071611200885575]  [1.0, nan, nan, nan, nan, nan]
1  [0.2]      [0.2]  [0.11232480721638316]  [nan, 1.0, nan, nan, nan, nan]
2  [0.2]      [0.2]  [0.15619219814278143]  [nan, nan, 1.0, nan, nan, nan]
3  [0.2]      [0.2]  [0.19347948043021995]  [nan, nan, nan, 1.0, nan, nan]
4  [0.2]      [0.2]  [0.22517367037454272]  [nan, nan, nan, nan, 1.0, nan]
5  [0.0]      [0.2]    [0.2521137318272171]  [nan, nan, nan, nan, nan, 1.0]

HITS Algorithm Execution time :0.00013641302257605476 sec
PageRank Algorithm Execution time :0.00034313584092919504 sec
SimRank Algorithm Execution time :0.00044830006538210067 sec

Using NetworkX >
   Hub  Authority  PageRank
0  0.2        0.0  0.060716
1  0.2        0.2  0.112324
2  0.2        0.2  0.156192
3  0.2        0.2  0.193480
4  0.2        0.2  0.225174
5  0.0        0.2  0.252114

HITS Algorithm Execution time :0.00014692944502134493 sec
PageRank Algorithm Execution time :0.0005174079843082956 sec
```

在 Graph_1 中，只有六個點，而且每個節點只有一條連往下一個節點的邊，第一個點沒有 indegree 為 0、而最後一個節點 outdegree 為 0，若用 Authority 來看 1~5 個點都為 0.2，而用 PageRank 來看的話最後一個節點的值較高；因為大家的邊都不同所以皆不相似，故相似矩陣 SimRank 僅與自己相似。

➢ **新增 Node 0->Node 2 之 Edge**

在原本的圖上新增新的邊為節點 0 指向 2，則圖形變為下圖，創造出一個迴圈；可以觀察到 Node 0 的 hub 及 Authority 增加的量一樣，而由於 Node 2 為 Node0、1 共同指向的節點因此其 Authority 增加許多，而算 PageRank 來說比較沒有甚麼變化，同樣為最後個節點的值會較高，因為其前面節點的 Authority 較高。



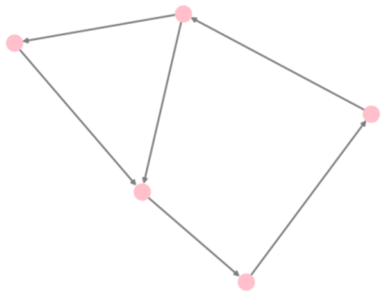
```
        Hub        Authority
2  2.837e-07  6.180e-01
1  3.820e-01  3.820e-01
3  2.837e-07  4.591e-07
4  2.837e-07  4.591e-07
5  0.000e+00  4.591e-07
0  6.180e-01  0.000e+00
   PageRank
5     0.258
4     0.231
3     0.199
2     0.162
1     0.088
0     0.062
```

■ **Graph_2**

```
Graph Dataset :graph_2.txt

Adjacency Matrix :
[[0. 1. 0. 0. 0.]
 [0. 0. 1. 0. 0.]
 [0. 0. 0. 1. 0.]
 [0. 0. 0. 0. 1.]
 [1. 0. 0. 0. 0.]]

Implement >
     Hub  Authority  PageRank                    SimRank
0  [0.2]      [0.2]     [0.2]  [1.0, 0.0, 0.0, 0.0, 0.0]
1  [0.2]      [0.2]     [0.2]  [0.0, 1.0, 0.0, 0.0, 0.0]
2  [0.2]      [0.2]     [0.2]  [0.0, 0.0, 1.0, 0.0, 0.0]
3  [0.2]      [0.2]     [0.2]  [0.0, 0.0, 0.0, 1.0, 0.0]
4  [0.2]      [0.2]     [0.2]  [0.0, 0.0, 0.0, 0.0, 1.0]

HITS Algorithm Execution time :0.00013791536865396677 sec
PageRank Algorithm Execution time :7.601871081881466e-05 sec
SimRank Algorithm Execution time :0.0003737837006266899 sec

Using NetworkX >
   Hub  Authority  PageRank
0  0.2        0.2       0.2
1  0.2        0.2       0.2
2  0.2        0.2       0.2
3  0.2        0.2       0.2
4  0.2        0.2       0.2

HITS Algorithm Execution time :0.00013731443021422225 sec
PageRank Algorithm Execution time :0.0003034739048498134 sec
```

在 Graph_2 中，五個點形成迴圈，因此與 Graph_1 相比，各節點的 Hub、Authority 及 PageRank 值皆相同且相似度矩陣一樣不變。

➢ **新增 Node 0->Node 2 之 Edge**



```
        Hub       Authority
2  2.837e-07  6.180e-01
1  3.820e-01  3.820e-01
0  6.180e-01  4.591e-07
3  2.837e-07  4.591e-07
4  2.837e-07  4.591e-07
   PageRank
2     0.225
3     0.221
4     0.218
0     0.215
1     0.121
```

一樣在原本的圖上新增新的邊為節點 0 指向 2，由於原本的圖就形成迴圈，有再加上這個迴圈後，Node 2 為 Node0、1 共同指向的節點因此其 Authority 增加許多，並且在 PageRank 排名 Node2 為最高。

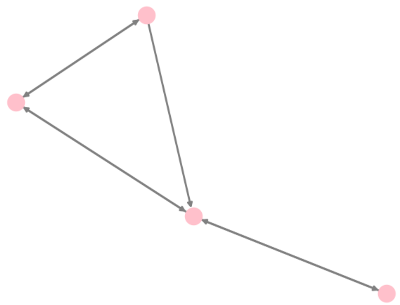
- **Graph_3**

```
Graph Dataset :graph_3.txt

Adjacency Matrix :
[[0. 1. 0. 0.]
 [1. 0. 1. 0.]
 [0. 1. 0. 1.]
 [0. 0. 1. 0.]]

Implement >
                     Hub                 Authority                  PageRank
0  [0.19098309329993374]   [0.1909827760891591]   [0.17543839772251532]
1  [0.3090169067000626]   [0.30901722391084097]    [0.3245616022774846]
2  [0.3090169067000626]   [0.30901722391084097]    [0.3245616022774846]
3  [0.1909830932999374]    [0.1909827760891591]   [0.17543839772251532]

                                      SimRank
0  [1.0, 0.0, 0.4285714285714286, 0.0]
1  [0.0, 1.0, 0.0, 0.4285714285714286]
2  [0.4285714285714286, 0.0, 1.0, 0.0]
3  [0.0, 0.4285714285714286, 0.0, 1.0]

HITS Algorithm Execution time :0.00022985894773277795 sec
PageRank Algorithm Execution time :0.0005369384831352264 sec
SimRank Algorithm Execution time :0.0004053329679627707 sec

Using NetworkX >
        Hub   Authority   PageRank
0  0.190983    0.190983   0.175438
1  0.309017    0.309017   0.324562
2  0.309017    0.309017   0.324562
3  0.190983    0.190983   0.175438

HITS Algorithm Execution time :0.0003236053421022227 sec
PageRank Algorithm Execution time :0.0003428353717164724 sec
```

在 Graph_3 中，四個點兩兩相連 因此中間兩個點 Node1、2 的 Hub、Authority 及

PageRank 值會一樣且較高；從 SimRank 相似度矩陣來看可以得知，Node1 和 3 都有連接到 Node 2 因此較相似，Node 0 和 2 都有連接到 Node 1 因此較相似。

- **新增 Node 0->Node 2 之 Edge**



	Hub	Authority
2	0.173	0.462
1	0.280	0.285
0	0.338	0.156
3	0.209	0.097

	PageRank
2	0.384
1	0.265
3	0.201
0	0.150

在新增此邊之後，Node 2 為 Node 0、1、3 所連接，因此 Authority 值最高，且 Node 0 hub 值變大；在 PageRank 來看原本 Node1 和 2 相同，但新增後，Node 2 就位居其首了。

Graph_4

在 Graph_4 中，與 NetworkX 的結果做比較，排名是一樣的；用 Authority 值來看的話 Node4 最高在來是 Node2、1、3，而用 PageRank 來看的話，由於 Node 0 Hub 值最高，因此也影響了其 PageRank 排名。

HITS Algorithm(左:Implement 右:NetworkX)>

Implement >

	Hub	Authority
4	[0.1837348365905041]	[0.201425041553256]
2	[0.10868335659196027]	[0.20082309403907608]
1	[0.04776250054880305]	[0.17791181395679528]
3	[0.19865932442839782]	[0.1401777919110339]
0	[0.27545289304813597]	[0.13948446662880473]
6	[0.06897229412169785]	[0.08408831301163074]
5	[0.1167347946705009]	[0.05608947889403175]

Using NetworkX >

	Hub	Authority
4	0.184	0.201
2	0.109	0.201
1	0.048	0.178
3	0.199	0.140
0	0.275	0.139
6	0.069	0.084
5	0.117	0.056

PageRank Algorithm(左:Implement 右:NetworkX)>

	PageRank
0	[0.28028775735575334]
4	[0.1841986673136965]
1	[0.1587646872485608]
2	[0.13888170080462586]
3	[0.10821930792439603]
6	[0.06907737460022821]
5	[0.06057050475273926]

	PageRank
0	0.280
4	0.184
1	0.159
2	0.139
3	0.108
6	0.069
5	0.061

在 SimRank 相似矩陣來看，Node 3 與 6 有最高的相似度 0.2011 也可以從 Adjmatrix 找出端倪。

AdjMatrix>

```
Graph Dataset :graph_4.txt

Adjacency Matrix :
[[0. 1. 1. 1. 1. 0. 1.]
 [1. 0. 0. 0. 0. 0. 0.]
 [1. 1. 0. 0. 0. 0. 0.]
 [0. 1. 1. 0. 1. 0. 0.]
 [1. 0. 1. 1. 0. 1. 0.]
 [1. 0. 0. 0. 1. 0. 0.]
 [0. 0. 0. 0. 1. 0. 0.]]
```

SimRank>

```
                                                                                                                          SimRank
0  [1.0, 0.13785734257851975, 0.13393125362184838, 0.13603931447638828, 0.13559737389491208, 0.1588345291114338, 0.11324409984134282]
1   [0.13785734257851973, 1.0, 0.15682983233992698, 0.1403984493736493, 0.15808918029754127, 0.11085130570791396, 0.1699455930393846]
2    [0.13393125362184838, 0.15682983233992698, 1.0, 0.17141821075483393, 0.149946347777832, 0.1724465297307996, 0.17038989177886826]
3    [0.13603931447638828, 0.1403984493736493, 0.17141821075483396, 1.0, 0.13014867473193323, 0.2066103247034261, 0.2011691118777471]
4  [0.13559737389491208, 0.15808918029754127, 0.14994634777783203, 0.13014867473193323, 1.0, 0.10520867961814236, 0.15508866984572411]
5    [0.1588345291114338, 0.11085130570791396, 0.1724465297307996, 0.2066103247034261, 0.10520867961814236, 1.0, 0.10847789911592967]
6   [0.11324409984134282, 0.1699455930393846, 0.17038989177886826, 0.2011691118777471, 0.15508866984572411, 0.10847789911592967, 1.0]
```

■ Graph_5

由於 Graph_5 Node 數很多，因此分別顯示前 30 個 Authority 較大的節點以及 PageRank，從使用 NetworkX 套件來做比較，可以看到顯示出來的是一樣的值以及排名。

HITS Algorithm(左:Implement 右:NetworkX)>

```
Implement >
                           Hub              Authority
60                       [0.0]   [0.09585172527001588]
121                      [0.0]     [0.094153754034592]
211   [0.020763073003488692]   [0.057570354315684996]
103                      [0.0]   [0.05592902517976362]
281                      [0.0]  [0.049712153384528196]
184   [0.021206864055918248]  [0.049428802206426925]
347    [0.02222610143961958]   [0.04329667912146398]
324   [0.025563643033216555]   [0.04224819131477645]
147   [0.014681069139746067]   [0.03811690829989748]
133    [0.02128741904259539]  [0.028214131885132093]
380   [0.013625966339944635]  [0.015242578672001583]
153                      [0.0]   [0.01461356344785299]
325    [0.02390119799337205]  [0.014515790239829093]
159                      [0.0]  [0.012961013422005757]
215                      [0.0]  [0.012208669371181957]
403                      [0.0]  [0.011128774474305325]
277                      [0.0]  [0.010078258396297475]
163                      [0.0]  [0.010078258396297475]
140    [0.01435967296220192]  [0.009956465093481455]
314    [0.01771455527490453]  [0.009107606592112624]
54                       [0.0]  [0.008543342489701808]
80    [0.016013433757721483]  [0.008543342489701808]
414                      [0.0]  [0.008099013962582585]
411    [0.02732437749838148]   [0.00800063128034263]
296     [0.0242946694264437]  [0.007453096722196829]
298    [0.02430746176148597] [0.0072184154194180755]
192                      [0.0]  [0.007102723011167931]
183   [0.018414189550741186]  [0.006778309800610113]
173   [0.015668140803119436]  [0.006673320897215024]
132                      [0.0]  [0.006578470415476278]
```

```
Using NetworkX >
          Hub  Authority
60   0.000000   0.095852
121  0.000000   0.094154
211  0.020763   0.057570
103  0.000000   0.055929
281  0.000000   0.049712
184  0.021207   0.049429
347  0.022226   0.043297
324  0.025564   0.042248
147  0.014681   0.038117
133  0.021287   0.028214
380  0.013626   0.015243
153  0.000000   0.014614
325  0.023901   0.014516
159  0.000000   0.012961
215  0.000000   0.012209
403  0.000000   0.011129
277  0.000000   0.010078
163  0.000000   0.010078
140  0.014360   0.009956
314  0.017715   0.009108
54   0.000000   0.008543
80   0.016013   0.008543
414  0.000000   0.008099
411  0.027324   0.008001
296  0.024295   0.007453
298  0.024307   0.007218
192  0.000000   0.007103
183  0.018414   0.006778
173  0.015668   0.006673
132  0.000000   0.006578
```

PageRank Algorithm(左:Implement 右:NetworkX)>

	PageRank
60	[0.014354781756295969]
121	[0.014128341252091444]
103	[0.010278220789361574]
211	[0.007810888575412315]
281	[0.007408887491205179]
184	[0.00727771425214987]
324	[0.007193086238798108]
347	[0.006887589780980403]
147	[0.00603353294921675]
95	[0.00593958444907247]
43	[0.00472000489951856]
133	[0.004555427731781286]
93	[0.004266562539573045]
23	[0.0042627656882117965]
39	[0.004204636576530278]
286	[0.004204636576530278]
42	[0.0038719577346388047]
20	[0.0038512714703081295]
203	[0.00376722904711336]
453	[0.0037493501719540762]
21	[0.0037445712263752184]
326	[0.0037369236381292704]
362	[0.0036975174325798344]
276	[0.0036970362361724916]
263	[0.0036707972917930545]
432	[0.0036015585361222568]
300	[0.0035603096963882754]
151	[0.0035393127680278208]
385	[0.003537540327737508]
248	[0.003503972903988159]

	PageRank
60	0.014361
121	0.014135
103	0.010283
211	0.007814
281	0.007412
184	0.007281
324	0.007196
347	0.006890
147	0.006036
95	0.005917
43	0.004702
133	0.004557
93	0.004251
23	0.004246
39	0.004206
286	0.004206
42	0.003860
20	0.003837
203	0.003754
453	0.003750
21	0.003731
326	0.003725
362	0.003698
276	0.003685
263	0.003672
432	0.003603
300	0.003562
151	0.003529
385	0.003525
248	0.003505

Execution Time (implement)>

```
HITS Algorithm Execution time :0.017102707588169685 sec
PageRank Algorithm Execution time :1.050069889138879 sec
SimRank Algorithm Execution time :0.6578992755086344 sec
```

Execution Time (NetworkX)>

```
HITS Algorithm Execution time :0.058612829795007926 sec
PageRank Algorithm Execution time :0.01838811488019676 sec
```

■ Graph_6

HITS Algorithm(左:Implement 右:NetworkX)>

Implement >

	Hub	Authority
1150	[0.0]	[0.0304038325958039]
760	[0.0]	[0.0304038325958039]
61	[0.009019162722450331]	[0.0301777196629247]
77	[0.0100180592618223]	[0.030031217911678572]
393	[0.0]	[0.029320864859952826]
862	[0.010438151531110152]	[0.028626691928621165]
1122	[0.0]	[0.02820279653910369]
500	[0.013875433845915535]	[0.025390871424313868]
1051	[0.00022780088055220858]	[0.024598449872034498]
179	[0.011559393704137156]	[0.023962727283461705]
818	[0.0]	[0.020060760461744626]
505	[0.012986667478399952]	[0.019213317916338166]
527	[0.0]	[0.017499842946578817]
1198	[0.015068967744672134]	[0.017299244779085503]
356	[0.0]	[0.016984356743180544]
1146	[0.0]	[0.014060494819373394]
1226	[0.009735632155246145]	[0.01365249954866864]
133	[0.010117264536935545]	[0.013474736256587063]
385	[0.014821173171176176]	[0.013384677251590095]
930	[0.0012768227330391699]	[0.013069112765946227]
224	[0.0]	[0.012509406267670215]
1088	[0.0]	[0.011542090559222925]
369	[0.0]	[0.01070748472791477]
520	[1.7507319415438026e-120]	[0.00985584226029567]
537	[0.0]	[0.007396925502416216]
1112	[0.009868447168662961]	[0.006209719766778676]
1144	[0.0]	[0.005913632046632243]
1070	[0.013447184521997555]	[0.005910025201302627]
1183	[0.0]	[0.005802590798364495]
409	[6.12076924001082e-38]	[0.00500356940621196]

Using NetworkX >

	Hub	Authority
1150	0.000000e+00	0.030404
760	0.000000e+00	0.030404
61	9.019277e-03	0.030178
77	1.001819e-02	0.030032
393	0.000000e+00	0.029321
862	1.043830e-02	0.028627
1122	0.000000e+00	0.028203
500	1.387562e-02	0.025391
1051	2.277988e-04	0.024598
179	1.155956e-02	0.023963
818	0.000000e+00	0.020061
505	1.298684e-02	0.019214
527	0.000000e+00	0.017500
1198	1.506917e-02	0.017300
356	0.000000e+00	0.016985
1146	0.000000e+00	0.014061
1226	9.735757e-03	0.013653
133	1.011741e-02	0.013475
385	1.482137e-02	0.013385
930	1.276813e-03	0.013069
224	0.000000e+00	0.012510
1088	0.000000e+00	0.011542
369	0.000000e+00	0.010708
520	1.043047e-161	0.009856
537	0.000000e+00	0.007397
1112	9.868574e-03	0.006210
1144	0.000000e+00	0.005914
1070	1.344736e-02	0.005910
1183	0.000000e+00	0.005803
409	2.866222e-50	0.005003

PageRank Algorithm(左:Implement 右:NetworkX)>

	PageRank
1051	[0.0038674492258179067]
1150	[0.0031246138015606916]
760	[0.0031246138015606916]
61	[0.003105817386319215]
393	[0.003032724581504898]
77	[0.003032551684444236]
862	[0.0030282152036658263]
1122	[0.0029154968722001702]
500	[0.0026911859835077917]
179	[0.0023171320940278154]
1226	[0.0021267979620712653]
409	[0.0020765882759963706]
818	[0.002023887352798166]
373	[0.0020211580676872155]
386	[0.0020211580676872155]
1020	[0.0020211580676872155]
553	[0.0020108360197525978]
1083	[0.0019485354595331383]
42	[0.0019408950694998546]
414	[0.0019363455958317465]
527	[0.0019269933834889443]
467	[0.0019153805407248804]
945	[0.0019072895879793136]
505	[0.0018973975530984372]
224	[0.0018787565415608022]
272	[0.001857219850809339]
493	[0.0018548132718213554]
219	[0.0018237220249749384]
369	[0.001819999455540955]
577	[0.001816986709255053]

	PageRank
1051	0.003860
1150	0.003126
760	0.003126
61	0.003108
393	0.003035
77	0.003034
862	0.003030
1122	0.002917
500	0.002693
179	0.002318
1226	0.002128
409	0.002068
818	0.002025
373	0.002013
386	0.002013
1020	0.002013
553	0.002003
1083	0.001941
42	0.001933
414	0.001929
527	0.001928
467	0.001908
945	0.001900
505	0.001898
224	0.001880
272	0.001850
493	0.001847
369	0.001821
219	0.001817
577	0.001810

Execution Time (implement)>

```
HITS Algorithm Execution time :0.2098906652628745 sec
PageRank Algorithm Execution time :6.150343376216299 sec
SimRank Algorithm Execution time :7.44573856534364 sec
```

Execution Time (NetworkX)>

```
HITS Algorithm Execution time :1.4151514004269075 sec
PageRank Algorithm Execution time :0.05972606822814441 sec
```

Graph_7

HITS Algorithm(左:Implement 右:NetworkX)>

```
Implement >
                    Hub                     Authority
37   [0.010064670631027511]  [0.024211493741870746]
62   [0.007737122046313935]  [0.022734948057773072]
35   [0.009367132047610484]  [0.022626761600570904]
86   [0.008338463683258215]  [0.022445331612109928]
68   [0.009667610272071109]   [0.02150005783009765]
16   [0.009804636020946644]   [0.01992710294725039]
7    [0.005931065340795744]  [0.019712231367049228]
84   [0.007471594922130131]  [0.018961031239353245]
47   [0.011674196958180182]  [0.018788372265357484]
2     [0.01015972720813854]   [0.01817212336034084]
27   [0.012379066974132704]  [0.017347717889890566]
42   [0.009780930600962882]   [0.01727480578068866]
10    [0.00980852433032276]   [0.01705686692600523]
70   [0.011268047255064783]   [0.01672601600902691]
82   [0.005684712308194978]   [0.01645523535206791]
46   [0.011335188459114114]  [0.016042340998721403]
39   [0.007924747425510434]  [0.015977024008195288]
28   [0.011513170872132276]  [0.015813621707947905]
13   [0.010751810439543243]   [0.01574967824034951]
80   [0.010468219378006231]   [0.01555937393940463]
34   [0.008949653057359216]  [0.015465633719029941]
66   [0.012361588645375234]  [0.015451460370991212]
50   [0.009134160118230641]   [0.01541104663600451]
61    [0.00833632901544672]  [0.015291684877205636]
79   [0.007663810722660631]  [0.015174189884808549]
60   [0.010887213671541216]  [0.015110972962875767]
72  [0.0074823102300899955]  [0.014975712280294012]
38   [0.010944098572785882]  [0.014923960248532216]
85   [0.011326551425068504]  [0.014873972572047218]
8    [0.011867147797487551]  [0.014867358403820833]
```

```
Using NetworkX >
      Hub    Authority
38   0.010        0.024
63   0.008        0.023
36   0.009        0.023
87   0.008        0.022
69   0.010        0.022
17   0.010        0.020
8    0.006        0.020
85   0.007        0.019
48   0.012        0.019
3    0.010        0.018
28   0.012        0.017
43   0.010        0.017
11   0.010        0.017
71   0.011        0.017
83   0.006        0.016
47   0.011        0.016
40   0.008        0.016
29   0.012        0.016
14   0.011        0.016
81   0.010        0.016
35   0.009        0.015
67   0.012        0.015
51   0.009        0.015
62   0.008        0.015
80   0.008        0.015
61   0.011        0.015
73   0.007        0.015
39   0.011        0.015
86   0.011        0.015
9    0.012        0.015
```

PageRank Algorithm(左:Implement 右:NetworkX)>

	PageRank
37	[0.022290977325645417]
62	[0.02178969994215656]
86	[0.02162417893662869]
35	[0.021165282039142034]
68	[0.02042546551504298]
16	[0.018661998389947747]
2	[0.017423873106633316]
84	[0.017238306870069487]
7	[0.01723452552839687]
27	[0.016950759982694758]
47	[0.016802757678424585]
82	[0.016249737410837135]
80	[0.015366404377144786]
70	[0.01527698951580031]
28	[0.015237091933384465]
46	[0.0152209771359549]
42	[0.014797704938644187]
39	[0.014742687968639817]
60	[0.014710461469139341]
10	[0.014499406426163089]
66	[0.014447580603637355]
34	[0.014353920628889832]
38	[0.014309280899036141]
50	[0.014161673452746139]
13	[0.013935747996573515]
61	[0.013885198756188582]
85	[0.0138675098817468]
79	[0.013828528081603269]
8	[0.013457528702245255]
51	[0.013406310830578817]

	PageRank
38	0.022
63	0.022
87	0.022
36	0.021
69	0.020
17	0.019
3	0.017
85	0.017
8	0.017
28	0.017
48	0.017
83	0.016
81	0.015
71	0.015
29	0.015
47	0.015
43	0.015
40	0.015
61	0.015
11	0.014
67	0.014
35	0.014
39	0.014
51	0.014
14	0.014
62	0.014
86	0.014
80	0.014
9	0.013
52	0.013

Execution Time (implement)>

```
HITS Algorithm Execution time :0.002237299135848467 sec
PageRank Algorithm Execution time :0.34749558309195583 sec
SimRank Algorithm Execution time :0.023765669507109122 sec
```

Execution Time (NetworkX)>

```
HITS Algorithm Execution time :0.04589077316923662 sec
PageRank Algorithm Execution time :0.029312945446678618 sec
```

在 Graph_7 中，我的程式跑出的結果與套件的標號差一，可能由於 NetworkX 是用 dictionary 的方式，在轉換成 list 的時候順序有跑，但不影響總結果。

■ Graph_8

由於找出來的規則只有五個節點，其他節點均沒有邊聯結，因此結果如下。

Implement >

	Hub	Authority
62	[0.23129904266157036]	[0.23129954519724327]
37	[0.23129904266157036]	[0.23129954519724324]
86	[0.19910380406771044]	[0.19910353914794413]
35	[0.19910380406771047]	[0.19910353914794413]
68	[0.13919430654143838]	[0.13919383130962507]
55	[0.0]	[0.0]
61	[0.0]	[0.0]
60	[0.0]	[0.0]
59	[0.0]	[0.0]
58	[0.0]	[0.0]
57	[0.0]	[0.0]
56	[0.0]	[0.0]
54	[0.0]	[0.0]
64	[0.0]	[0.0]

	PageRank
62	[0.07093617802416123]
37	[0.07093617802416123]
86	[0.05416398762528666]
35	[0.05416398762528666]
68	[0.03881700974156606]
55	[0.008670520231213872]
61	[0.008670520231213872]
60	[0.008670520231213872]
59	[0.008670520231213872]
58	[0.008670520231213872]
57	[0.008670520231213872]
56	[0.008670520231213872]
54	[0.008670520231213872]

Using NetworkX >

	Hub	Authority
1	0.231	0.231
2	0.231	0.231
0	0.199	0.199
4	0.199	0.199
3	0.139	0.139

	PageRank
1	0.245
2	0.245
0	0.187
4	0.187
3	0.134

Execution Time (implement)>

```
HITS Algorithm Execution time :0.0017117772195714097 sec
PageRank Algorithm Execution time :0.03571956540028605 sec
SimRank Algorithm Execution time :2.2835715058351636e-05 sec
```

Execution Time (NetworkX)>

```
HITS Algorithm Execution time :0.0007172217347932164 sec
PageRank Algorithm Execution time :0.0005856159032727182 sec
```

■ Result Summarize

跑完各個 Graph 後的小結論。觀察到若圖中形成迴圈，則迴圈上的節點其 PageRank 會一樣；邊較少的圖，SimRank 也會較為單純，相似度較低；像 Graph_1 此類型的圖，每個 Node 將自己的 indegree 數送至下一個 Node，則每個節點的 hub 和 authority 值皆相同；若節點互為 in-out 的節點，會發現其每個點自己的 hubs 與其 authority 相同。

Performance Analysis

■ Compare with Exaction time

Algorithm	HITS		PageRank	
	My implementation	Using Networkx	My implementation	Using Networkx
Graph_1	0.00013	0.00014	0.00034	0.00051
Graph_2	0.00013	0.00013	7.6018	0.0003
Graph_3	0.00022	0.00053	0.00032	0.00034
Graph_4	0.00037	0.00082	0.00108	0.00059
Graph_5	0.0169	0.0586	1.0886	0.0183
Graph_6	0.2098	1.4151	6.1503	0.0597
Graph_7	0.0022	0.0458	0.3474	0.0293
Graph_8	0.0017	0.0007	0.0357	0.0005

在我 implementation 的程式與使用 NetworkX 套件的執行時間上來比較，PageRank 演算法均比 HITS 所花費的時間要多，另外由於套件都包好了也有結構化，與我使用矩陣乘法以及迴圈下去跑，花費的時間有絕對的優勢，不過在點的數量比較少的圖中，可以發現我的程式碼比較快一點點點，可能因為建矩陣的速度較快吧，但點多一點就麻煩了，如果要優化的話，會盡量想試著改成稀疏矩陣，只記錄邊，而不要代全部的點下去跑。

Discussion

- **More limitations about link analysis algorithms**

  這些 link analysis algorithms 主要是依循著各點之射入及射出之邊下去運算，因此我覺得跟老師上課的時候說的 link analysis algorithms 其實與單純計算各點的 in&outdegree 有正相關，要找出比這個關係更進一步的資訊我覺得是目前 link analysis algorithms 重要的挑戰。

- **Can link analysis algorithms really find the “important” pages from Web?**

  我認為不大能完全找到重要的 page/node，由於這些 link analysis algorithms 只能依照 page/node 間聯結的程度來分析出其中比較重要的 page/node，但不一定比較多 Node 連接的就比較重要，就算給予不同 Node 不同的 hub 值，那如何給這些 hub 值就相當於這個 Node 重要的程度了。

- **Any new idea about the link analysis algorithm?**

  如果 link analysis algorithm 用於 Web 中的話，可以基於語意來設計每個 page 的 hub 值，例如今天想找關於某個主題的重要網頁的話，直觀的想法大概是越有相關於此主題的

page 就會出現越多關於此主題的相關語句，那我們可以搜尋每個 page 中出現此主題 keyword 數量來定義其 hub 值，再丟進 PageRank 做運算。

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

  網頁連結太多，相關資訊變得難以收集，導致計算參數會變得複雜。若以矩陣運算，又多為稀疏矩陣，實作起來複雜度高實用性變得很低，而且也可能會有衝聯結數量的情形出現。

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

  SimRank 中 c 的取值會影響相似度計算過程的收斂速度，當 c 取值較大時收斂速度較慢，當 c 取值較小時收斂速度較快。

  以下用 Graph_5 跑 SimRank 執行速度做比較:

  C=0.2, Execution time :0.6629111931060179 sec

  C=0.4, Execution time :0.6920787111041669 sec

  C=0.6, Execution time :0.7254053339422857 sec

  C=0.8, Execution time : 0.7110921481196593 sec

- **Design a new link-based similarity measurement**

  我認為可以先用 K-means 分群，讓有相同連接得點放在同一群，然後同群間算 Betweenness Centrality 找最高的中心點，在依照這個中心點連接的點做一個比較重要的參考和加權，主要希望可以找出在網路中比較像是骨幹的 Node，在來衡量其他 Node 的重要度。