Data mining project3　　資訊所　P78071549　　陳珞安

Implementation Detail

一、 HITS & PageRank

- 執行方式： python3 hits_pr.py graph_file.txt delta

  ex: python3 hits_pr.py hw3dataset/graph_1.txt 0.001

- 程式內容：

  I. HITS:

  首先先將所有 nodes 的 inlinks 及 outlinks 求出

```
def calculate_inoutDegree(file):
    inlink = dict()
    outlink = dict()
    node_num = 1

    for line in file:
        link = line.rstrip().split(',')
        if int(link[0]) > node_num:
            node_num = int(link[0])
        if int(link[1]) > node_num:
            node_num = int(link[1])

        if link[1] in inlink:
            inlink[link[1]].append(link[0])
        else:
            inlink[link[1]] = [link[0]]

        if link[0] in outlink:
            outlink[link[0]].append(link[1])
        else:
            outlink[link[0]] = [link[1]]
    return inlink, outlink, node_num
```

  接著將所有 nodes 的 authority 及 hub 值初始化為 1

```
# auth value and hub value initial
for i in range(1, node_num+1):
    auth[str(i)] = 1
    hub[str(i)] = 1
```

  最後在兩次 iterate 間 authority 及 hub 的差異小於 delta 值前不斷的 iterate 直到收斂為止。

```
while delta > delta_limit: # iterate
    # calculate auth and hub, then normalize
    auth, hub, delta = HITS_calculate_and_normalization(auth, hub, inlink, outlink, node_num)
    iterate_num += 1
```

II. PageRank:

首先如同 HITS 的作法一般，先求出所有的 inlinks 及 outlinks

```
def calculate_inoutDegree(file):
    inlink = dict()
    outlink = dict()
    node_num = 1

    for line in file:
        link = line.rstrip().split(',')
        if int(link[0]) > node_num:
            node_num = int(link[0])
        if int(link[1]) > node_num:
            node_num = int(link[1])

        if link[1] in inlink:
            inlink[link[1]].append(link[0])
        else:
            inlink[link[1]] = [link[0]]

        if link[0] in outlink:
            outlink[link[0]].append(link[1])
        else:
            outlink[link[0]] = [link[1]]
    return inlink, outlink, node_num
```

接著對所有 nodes 的 pr 值做一個平均 (1/N) 的初始化

```
# pr value initial
for i in range(1, node_num+1):
    pr[str(i)] = 1 / node_num
```

最後在兩次 iterate 間 pagerank 的差異小於 delta 值前不斷的 iterate 直到收斂為止。

```
while delta > delta_limit: # iterate
    # calculate auth and hub, then normalize
    pr, delta = PR_calculate_and_normalization(pr, inlink, outlink, node_num)
    iterate_num += 1
```

二、 SimRank

- 執行方式：python3 simrank.py graph_file.txt

  ex: `python3 simrank.py hw3dataset/graph_1.txt`

- 程式內容：

  首先先計算出 graph 所有的 inlink

```python
def calculate_inDegree(file):
    inlink = dict()
    node_num = 1

    for line in file:
        link = line.rstrip().split(',')
        if int(link[0]) > node_num:
            node_num = int(link[0])
        if int(link[1]) > node_num:
            node_num = int(link[1])

        if link[1] in inlink:
            inlink[link[1]].append(link[0])
        else:
            inlink[link[1]] = [link[0]]

    return inlink, node_num
```

接著就是利用 inlink 來 Implement simRank 的演算法，在求出值之前會不斷 recursive 下去（倒數第二行），為了避免 cycle graph 造成程式無限 recursive，所以有設一個 table 來記錄所計算過的 simRank，要是有重複計算同兩個 node 間的 simRank 則代表這個 graph 裡有 cycle

```python
def calculate_simrank(node_a, node_b, inlink, sim_inuse, c):
    if node_a == node_b:
        return 1
    elif (node_a not in inlink) or (node_b not in inlink):
        return 0
    else:
        if frozenset({node_a, node_b}) in sim_inuse['used']: # if there is cycle between two nodes
            sim_inuse['cycle'] = True
            return 0
        else:
            sim_inuse['used'].append(frozenset({node_a, node_b}))

        a_length = len(inlink[node_a])
        b_length = len(inlink[node_b])
        simrank = 0

        for i in inlink[node_a]:
            for j in inlink[node_b]:
                simrank = simrank + calculate_simrank(i, j, inlink, sim_inuse, c)

        return (c / (a_length * b_length)) * simrank
```

Result analysis and discussion

一、 HITS & PageRank　(delta value = 0.001)

i. graph_1:

◆ 執行結果：

```
annie  project3  python3 hits_pr.py hw3dataset/graph_1.txt 0.001
HITS algorithm iterate 2 times:
node    authority       hub
1       0.00000         0.20000
2       0.20000         0.20000
3       0.20000         0.20000
4       0.20000         0.20000
5       0.20000         0.20000
6       0.20000         0.00000

PageRank algorithm iterate 17 times:
node    pageRank
1       0.03481
2       0.07603
3       0.12484
4       0.18254
5       0.25055
6       0.33122
```

圖一為由 node1 為 root，單向一一串接至 node6。因為 node1 沒有任何 inlink，所以其 authority 為 0，而其他 nodes 皆有一個 inlink，所以平分 authority，都為 0.2。相反的，由於 node6 沒有任何 outlink，所以其 hub 為 0，其餘則為 0.2。

至於 pageRank 的值則可以看出由 node1 至 node6 是漸增的趨勢，其實也符合 pageRank 只看 inlink 的特性，且 parent 會將他的 pageRank 值再分給其 chilren，所以才會呈現與 authority 不同、漸增的趨勢。最後也因為有 damping value，所以 node1 的 pageRank 才不是 0。

ii. graph_2:

◆ 執行結果：

```
annie project3 python3 hits_pr.py hw3dataset/graph_2.txt 0.001
HITS algorithm iterate 2 times:
node    authority       hub
1       0.20000         0.20000
2       0.20000         0.20000
3       0.20000         0.20000
4       0.20000         0.20000
5       0.20000         0.20000

PageRank algorithm iterate 1 times:
node    pageRank
1       0.20000
2       0.20000
3       0.20000
4       0.20000
5       0.20000
```

圖二為頭尾相接的 cycle graph，並且每個的 parent/children 也都只有一個，也就是每個 node 之間在途上並沒有差異，因此不管是用 HITS 還是 pageRank 其重要程度都一樣，hub/authority/pagerank 也就都一樣。

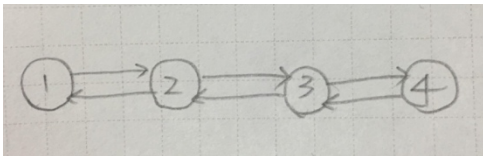
iii. graph_3:

◆ 執行結果：

```
annie project3 python3 hits_pr.py hw3dataset/graph_3.txt 0.001
HITS algorithm iterate 8 times:
node    authority       hub
1       0.19101         0.19101
2       0.30899         0.30899
3       0.30899         0.30899
4       0.19101         0.19101

PageRank algorithm iterate 9 times:
node    pageRank
1       0.17540
2       0.32460
3       0.32460
4       0.17540
```

圖三為 1 與 2、2 與 3、3 與 4 間皆為雙向的連結，如右上角圖所示，可以看出，node2 以及 node3 的 inlink 及 outlink 皆為 node1,node4 的兩倍，因此這樣的結果是正常的。值得一提的是，pageRank 的值之所以與 authority/hub 不同便是因為之前所提到的 damping factor。

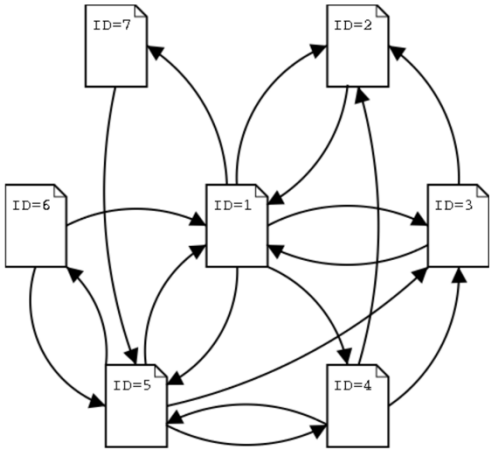


iv. graph_4:

◆ 執行結果：

```
annie project3 python3 hits_pr.py hw3dataset/graph_4.txt 0.001
HITS algorithm iterate 13 times:
node    authority       hub
1       0.13967         0.27538
2       0.17785         0.04781
3       0.20078         0.10871
4       0.14018         0.19860
5       0.20134         0.18380
6       0.05615         0.11675
7       0.08403         0.06894

PageRank algorithm iterate 9 times:
node    pageRank
1       0.28032
2       0.15872
3       0.13890
4       0.10827
5       0.18410
6       0.06061
7       0.06909
```



圖四為講義上的 graph，首先，pageRank 的值與講義上所算出的並不相同，理由還是因為 damping factor 所導致的，要是將其設為 0 的話出來的結果便會與講義上的結果一致。

在這裡也可以看到，雖然說 pageRank 與 authority 較為接近，但還差別還是挺大的。

v. graph_5(節錄):

```
annie project3 python3 hits_pr.py hw3dataset/graph_5.txt 0.001
HITS algorithm iterate 14 times:
node    authority       hub
1       0.00000         0.00000
2       0.00000         0.00000
3       0.00000         0.00000
4       0.00000         0.00000
5       0.00000         0.00001
6       0.00000         0.00000
7       0.00000         0.00001
8       0.00000         0.00000
9       0.00000         0.00000
10      0.00000         0.00000
11      0.00000         0.00000
12      0.00000         0.00001
13      0.00000         0.00000
14      0.00000         0.00000
15      0.00000         0.00000
```

```
PageRank algorithm iterate 32 times:
node    pageRank
1       0.00059
2       0.00059
3       0.00059
4       0.00059
5       0.00059
6       0.00075
7       0.00069
8       0.00073
9       0.00083
10      0.00083
11      0.00073
12      0.00069
13      0.00083
14      0.00083
```

vi. graph_6(節錄):

```
HITS algorithm iterate 73 times:
node    authority       hub
1       0.00000         0.00271
2       0.00090         0.00000
3       0.00000         0.00000
4       0.00014         0.00000
5       0.00000         0.00000
6       0.00113         0.00000
7       0.00043         0.00984
8       0.00100         0.01446
9       0.00217         0.01222
10      0.00056         0.00000
11      0.00038         0.00000
12      0.00045         0.00000
13      0.00000         0.00000
14      0.00000         0.00000
15      0.00000         0.00000
```

```
PageRank algorithm iterate 25 times:
node    pageRank
1       0.00027
2       0.00032
3       0.00036
4       0.00038
5       0.00031
6       0.00301
7       0.00069
8       0.00035
9       0.00047
10      0.00039
11      0.00118
```

vii. graph_7:

由於 hw1 所挑的 dataset (./hw1graph/mushrooms.csv)跟一般的 transactions 不同，如下圖

class	cap-shape	cap-surface	cap-color	bruises	odor
p	x	s	n	t	p
e	x	s	y	t	a
e	b	s	w	t	l
p	x	y	w	t	p
e	x	s	g	f	n
e	x	y	y	t	a
e	b	s	w	t	a

因此，除非所有 attributes 皆為相等，否則不會有集合(parents/children)的產生，但所選的資料又是香菇的分類，資料中並沒有兩筆完全相同的資料，因此此筆資料並沒有 graph。

二、 SimRank　(C = 0.8)

I. graph_1:

◆ 執行結果：

結果是除了自己跟自己外(ex: 1,1)

其他任兩點的 simRank 值皆為 0，

代表這種完全單向的 tree 是沒有

任兩點在 simRank 上是代表相似的。

```
annie  project3  python3 simrank.py hw3dataset/graph_1.txt
pair-wise similarity of nodes:
nodes   simrank
(1, 2)  0.0000
(1, 3)  0.0000
(1, 4)  0.0000
(1, 5)  0.0000
(1, 6)  0.0000
(2, 1)  0.0000
(2, 3)  0.0000
(2, 4)  0.0000
(2, 5)  0.0000
(2, 6)  0.0000
(3, 1)  0.0000
(3, 2)  0.0000
(3, 4)  0.0000
(3, 5)  0.0000
(3, 6)  0.0000
(4, 1)  0.0000
(4, 2)  0.0000
```

II. graph_2:

◆ 執行結果：

```
annie  project3  python3 simrank.py hw3dataset/graph_2.txt
pair-wise similarity of nodes:
nodes   simrank
(1, 2)  cycle
(1, 3)  cycle
(1, 4)  cycle
(1, 5)  cycle
(2, 1)  cycle
(2, 3)  cycle
(2, 4)  cycle
(2, 5)  cycle
(3, 1)  cycle
(3, 2)  cycle
(3, 4)  cycle
(3, 5)  cycle
(4, 1)  cycle
(4, 2)  cycle
(4, 3)  cycle
(4, 5)  cycle
(5, 1)  cycle
(5, 2)  cycle
(5, 3)  cycle
(5, 4)  cycle
```

由於 graph 中存在著 cycle，所以在 iterate 的時候會進入無限的迴圈，因此特別標注。

III. graph_3:

◆ 執行結果：

```
annie project3 python3 simrank.py hw3dataset/graph_3.txt
pair-wise similarity of nodes:
nodes   simrank
(1, 2)  cycle
(1, 3)  cycle
(1, 4)  cycle
(2, 1)  cycle
(2, 3)  cycle
(2, 4)  cycle
(3, 1)  cycle
(3, 2)  cycle
(3, 4)  cycle
(4, 1)  cycle
(4, 2)  cycle
(4, 3)  cycle
```

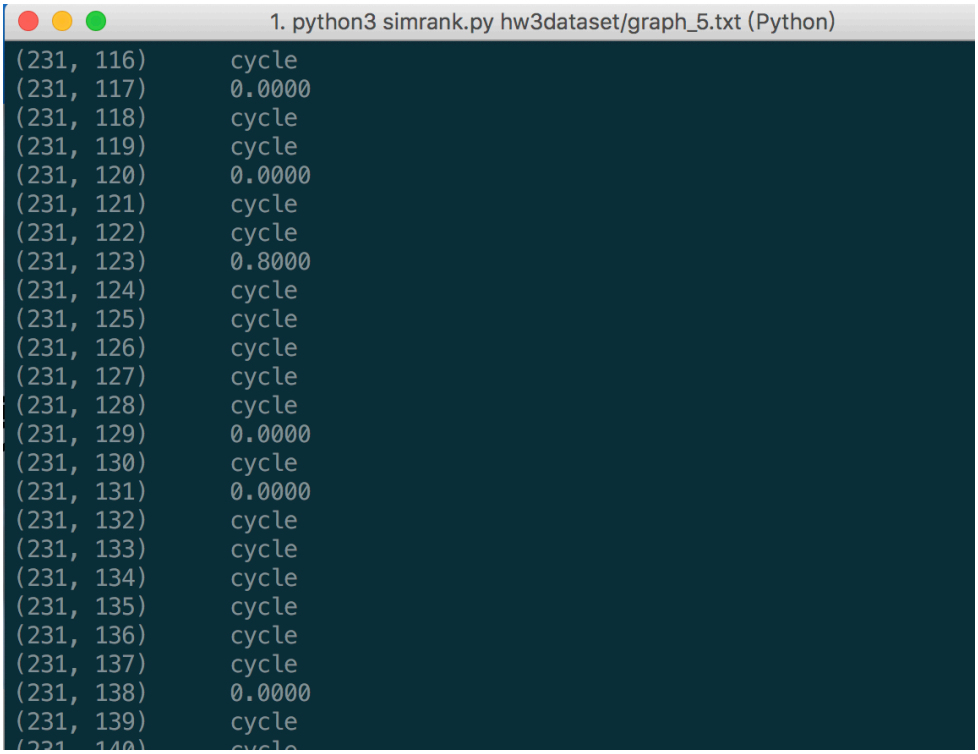
IV. graph_4:

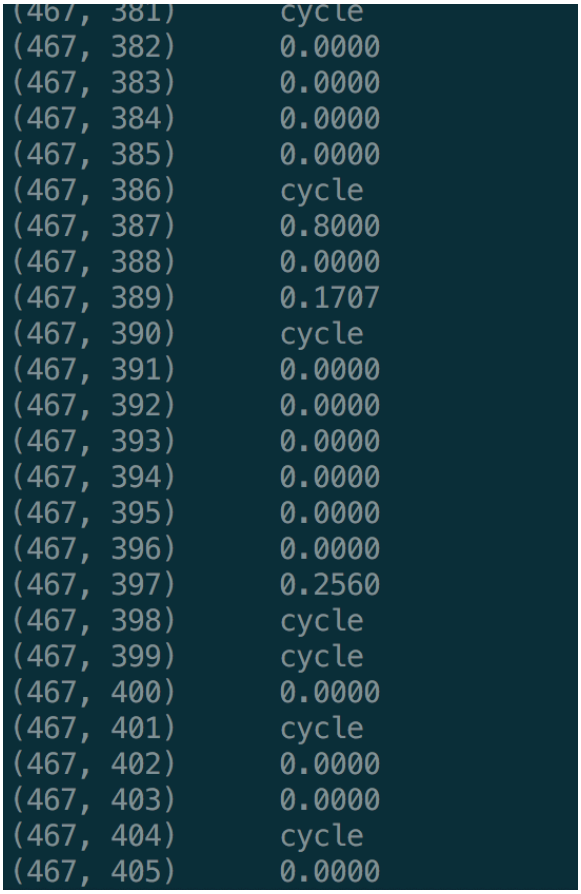
◆ 執行結果：

```
annie project3 python3 simrank.py hw3dataset/graph_4.txt
pair-wise similarity of nodes:
nodes   simrank
(1, 2)  cycle
(1, 3)  cycle
(1, 4)  cycle
(1, 5)  cycle
(1, 6)  cycle
(1, 7)  cycle
(2, 1)  cycle
(2, 3)  cycle
(2, 4)  cycle
(2, 5)  cycle
(2, 6)  cycle
(2, 7)  cycle
(3, 1)  cycle
(3, 2)  cycle
(3, 4)  cycle
(3, 5)  cycle
(3, 6)  cycle
(3, 7)  cycle
(4, 1)  cycle
(4, 2)  cycle
(4, 3)  cycle
```

V. graph_5:

◆ 執行結果：





Find a way to increase hub/auth/pr

一、 graph_1

i. authority:

增加 authority 最直接的方法便是增加 parent/ parent 的 hub。

- 多加一個 (6,1) 的 link，變成如同 graph_2 的形式。
- 增加一條指向自己的 link (1,1)也可以有效地增加 authority。

ii. hub:

增加 hub 最直接的方法便是增加 children/ children 的 authority。

- 多加任一條 outlink 至任意 node 上。

iii. pageRank:

至於 pageRank 的增加就沒那麼單純，不過簡單來說也是大致與增加 authority 相同。

- 多加一條 (6,1)的 link，變成如同 graph_2 的形式。

二、 graph_2

i. authority:

- 增加一條指向自己的 link (1,1)。
- 增加任意一條指向自己的 inlink。

ii. hub:

- 增加一條指向自己的 link (1,1)。
- 增加任意一條指向其他 nodes 的 outlink。

iii. pageRank:

- 增加一條指向自己的 link (1,1)。

三、 graph_3

i. authority:

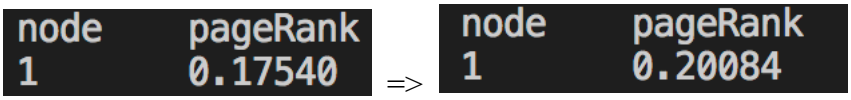
- 增加一條指向自己的 link (1,1)。
- 增加任意一條指向自己的 inlink。

ii. hub:

- 增加一條指向自己的 link (1,1)。
- 增加任意一條指向其他 nodes 的 outlink。

iii. pageRank:

- 增加任意一條指向自己的 inlink。
- 增加 parent 的 pageRank，例如 (4,2)



Computation performance analysis

一、 Memory (from graph_1 ~ graph_8)

- HITS: 581 KB
- PageRank: 295 KB

二、 Time (from graph_1 ~ graph_8)

- HITS: 0.527 s
- PageRank: 0.180 s

Discussion

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

在實際去調整 C 的大小後，發現其實結果就只差在 C 所帶來的偏移，

ex: C = 1 => S(A, B) = 1

C = 0.8 => S(A, B) = 0.8

以及上網查詢後，decay factor 的作用就是在兩個不同 node 間，讓其 simRank 值不可能為 1，畢竟 A 跟 B 是不同的點，其值區間為[0, 1]，可以依照 node 間的相似度來去做調整。

二、Can link analysis algorithms really find the “important” pages from Web？

我想在現實狀況下，link analaysis 還是有它一定的效用在的，不論是 HITS 還是 pageRank，更何況 pageRank 已經被 google 使用了那麼多年，只是這些演算法都還有他們的問題在，ex: Rank sink、投票部隊等等。

三、結論

這次的作業雖然大多都是在分析結果，但還是可以看出 link analysis 在現實上還是非常困難的，不光只是我們所 implement 那樣的單純，還需要考慮十分多的問題。但是還是能感覺出 link analysis 這樣的概念不僅僅只能應用在這種問題，這樣去算 graph 間 ranking 或是相似度的情況想必以後也會時常遇到吧。