Data Mining Project3

姓名 :電機所 卓冠廷
學號 : N26114277

演算法敘述

HITS(Hyperlink-Induced Topic Search):

此演算法基於連結分析網頁排名，對於獲得許多推薦的網頁，可視為擁有一定的聲譽，即權威型（authority）的網頁，能提供最好的相關資訊，相對地，給出許多推薦連結的網頁，將視為目錄型（hub）網頁，該網頁指向其他高聲譽的權威型網頁。

演算法流程:

計算經 k 個 iteration 後，N 個網頁的 authority 與 hub 值:

1. 初始化 N 個網頁的 authority 與 hub 值為 1。
2. 進行 authority 與 hub 的更新，前者為所有指向它的網站 hub 值總和，後者為所有指離它的網站 authority 值總和。
3. 進行 authority 與 hub 的 normalization。

$$authority(j) = \frac{authority(j)}{\sum_{i \in N} authority(i)}$$

$$hub(j) = \frac{hub(j)}{\sum_{i \in N} hub(i)}$$

PageRank:

此演算法也是做網頁排名，假若某網頁被很多其他網頁連結到，說明此網頁相對重要，pagerank 值會比較高，而如果一個 pagerank 值高的網頁連結到某個其他網頁，那個被連結的網頁 pagerank 值也會因此提高。

演算法流程:

計算經 k 個 iteration 後，N 個網頁的 pagerank 值:

1. 初始化 N 個網頁的 pagerank 值為 1。
2. 進行 pagerank 的更新，更新方式如下式，其中 d 為 damping factor。

$$pagerank(j) = \frac{d}{N} + (1-d) * \sum_{i \in j.parents} pagerank(i)/OutDegree(i)$$

3. 進行 pagerank 的 normalization。

$$pagerank(j) = \frac{pagerank(j)}{\sum_{i \in N} pagerank(i)}$$

Simrank:

此演算法希望建立用戶與物品的關聯推薦，simrank 算法的思想是如果兩個用戶相似，則其相關聯的物品也類似；如果兩物品類似，則與者兩物品相關聯的用

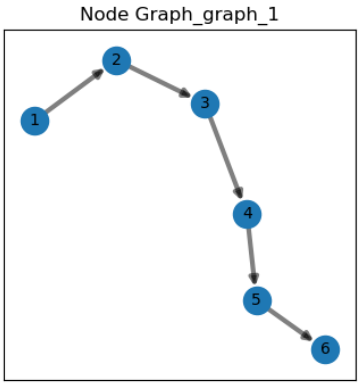
戶也會類似。

演算法流程:

1. 初始化 simrank 值為單位矩陣(N*N)，其中 N 為 node 數。
2. 進行 simrank 的更新，更新方式如下式，其中 c 為 decay factor。
   (1) node(i) 與 node(j)為同一 node:
   $simrank(i,j) = 1$
   (2) 其中某個 node 沒有 in-neighbors:
   $simrank(i,j) = 0$
   (3) 其餘情形:

$$simrank(i,j) = \frac{c}{InDegree(i)*InDegree(j)} * \sum_{k \in i.parents} \sum_{l \in j.parents} simrank(k,l)$$

Graph1(damping factor = 0.1 , decay factor = 0.7):



(1) 結果分析與討論

	1	2	3	4	5	6
Hub	0.2	0.2	0.2	0.2	0.2	0
Authority	0	0.2	0.2	0.2	0.2	0.2
PageRank	0.025	0.060	0.107	0.171	0.259	0.378

HITS:

Authority 取決於它被多少 node 指向，Hub 取決於它指向多少 node，node1 沒被其他 node 指向，因此 Authority 為 0，node6 沒指向任何節點，因此 Hub 為 0，其餘的指向或被指向都是 1 條，因此 Authority 與 Hub 皆為 0.2。

PageRank:

某個 node 若被其他 node 指向，此 node 的 PageRank 值會提升，因此會發現 node1 到 node6 的 PageRank 值由小到大。

SimRank:

結果為一六乘六的單位矩陣，對角線總是 1，是因為兩相同 node 的 SimRank 值始終為 1，而從 SimRank 的更新方式，是否有一共同的 parent 在計算中很重要，但從圖中可看到沒有一對 nodes 具有共同的 parent，所以

SimRank 值都是 0。

(2) 增加或刪除某些連結使 node1 的 Hub、Authority、PageRank 值提升:

增加 node6 指向 node1 的連結:

```
authority [0.167 0.167 0.167 0.167 0.167 0.167]
hub [0.167 0.167 0.167 0.167 0.167 0.167]
pagerank  [0.167 0.167 0.167 0.167 0.167 0.167]
```

根據分析的結果，增加 node1 的 Authority 只要增加指向它的連結數即可。也因為 node1 被其 node6 指到，PageRank 值隨之提升。

增加 node5 指向 node1 的連結:

```
authority [0.5 0.  0.  0.  0.  0.5]
hub [0. 0. 0. 0. 1. 0.]
pagerank  [0.131 0.152 0.173 0.195 0.217 0.131]
```

從結果可看到， Authority 一樣有上升，且上升許多，因為 node5 的 Hub 極高，因此被其指向的 node 的 Authority 也會提升很多，而在 PageRank 值的部分上升的就沒那多，這也應證演算法，PageRank 值高的網頁連結到某個其他網頁，那個被連結的網頁 PageRank 值也會因此提高，而 node5 原先的 PageRank 值就沒 node6 高，因此提升的幅度較少。

增加 node1 指向 node3 的連結:

```
authority [0.    0.382 0.618 0.    0.    0.   ]
hub [0.618 0.382 0.    0.    0.    0.   ]
pagerank  [0.026 0.044 0.104 0.171 0.263 0.392]
```

根據分析的結果，增加 node1 指出的連結，Hub 值就會提升。

(3) 不同 damping factor 和 decay factor 討論:

damping factor = 0.3:

```
pagerank  [0.061 0.112 0.156 0.193 0.225 0.252]
```

damping factor 調大，代表每次傳遞的速度下降，這部分可從演算法的地方發現，因此同樣都是疊代 30 次，node1 的 PageRank 值上升(因為降比較慢)。

damping factor = 0.01:

```
pagerank  [0.004 0.014 0.039 0.098 0.244 0.6  ]
```

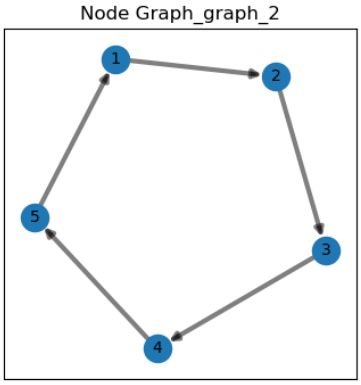
damping factor 調小，代表每次傳遞的速度上升，因此同樣都是疊代 30 次，node1 的 PageRank 值下降(因為降比較快)。

decay factor = 0.5 or 0.9:

```
simrank:
 [[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. 1.]]
```

會發現不管 decay factor = 0.5、0.7、0.9，SimRank 的值始終為單位矩陣，原因同分析結果，沒有一對 nodes 具有共同的 parent，所以 SimRank 值除了對角線以外都是 0。

Graph2(damping factor = 0.1 , decay factor = 0.7):



(1) 結果分析與討論

	1	2	3	4	5
Hub	0.2	0.2	0.2	0.2	0.2
Authority	0.2	0.2	0.2	0.2	0.2
PageRank	0.2	0.2	0.2	0.2	0.2

HITS:

此圖的 node 行成一個循環，他們都有一個 Parent 與 Children，因此擁有同樣的 Authority 與 Hub，值皆為 0.2。

PageRank:

node 形成一個循環，PageRank 值的傳遞也會成一個循環，因此皆為 0.2。

SimRank:

結果為一五乘五的單位矩陣，原因同 Graph1。

(2) 增加或刪除某些連結使 node1 的 Hub、Authority、PageRank 值提升:

增加 node4 指向 node1 的連結:

```
authority [0.618 0.    0.    0.    0.382]
hub [0.    0.    0.    0.618 0.382]
pagerank  [0.224 0.222 0.22  0.217 0.118]
```

可以發現 node1 的 Authority 值上升，因為多了 node4 的指向，node4 的 Hub 值也因此提升，相對應，其指向的另一節點 node5 的 Authority 值提升，但沒有 node1 高(其被兩個 node 指向)。node1 的 PageRank 值有提升，但變化不大，因為原本的 PageRank 值大家都一樣，因此 node 間的 PageRank 差異也不像 Graph1 那般顯著。

增加 node1 指向 node3 的連結:

```
authority [0.    0.382 0.618 0.    0.   ]
hub [0.618 0.382 0.    0.    0.   ]
```

根據分析的結果，增加 node1 指出的連結，Hub 值就會提升，其指到的 node(node2)的 Authority 值也隨之上升。

(3) 不同 damping factor 和 decay factor 討論:

damping factor = 0.3 or 0.01:

```
pagerank  [0.2 0.2 0.2 0.2 0.2]
```

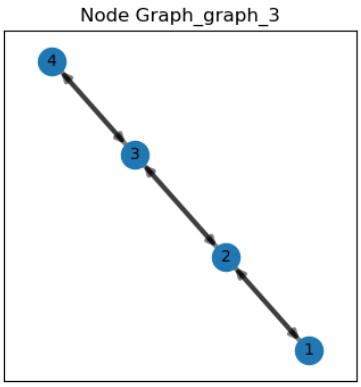
damping factor 調大或調小，對 node 的 PageRank 值都沒有影響，原因如分析所說，node 連成一環，PageRank 值的傳遞也會成一個循環。

decay factor = 0.5 or 0.9:

```
simrank:
 [[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. 1.]]
```

會發現不管 decay factor = 0.5、0.7、0.9，SimRank 的值始終為單位矩陣，原因同 Graph1。

Graph3(damping factor = 0.1 , decay factor = 0.7):



(1) 結果分析與討論

	1	2	3	4
Hub	0.191	0.309	0.309	0.191
Authority	0.191	0.309	0.309	0.191
PageRank	0.172	0.328	0.328	0.172

HITS:

Authority 取決於它被多少 node 指向，node1 與 node4 皆被一個 node 指向，其餘則被兩個指向，故前者 Authority 為 0.191，後者 Authority 為 0.309；Hub 取決於它指向多少 node，node1 與 node4 皆指向一個 node，其餘則指向兩個，故前者 Hub 為 0.191，後者 Hub 為 0.309。

PageRank:

node2 與 node3 皆被兩個 node 指向，其在 PageRank 值上接收的傳遞會較多，反之，node1 與 node4 收到的較少。

SimRank:

```
simrank:
 [[1.    0.    0.538 0.   ]
 [0.    1.    0.    0.538]
 [0.538 0.    1.    0.   ]
 [0.    0.538 0.    1.   ]]
```

前面提到 SimRank 的更新，是否有一共同的 parent 在計算中很重要，Graph 在 node1 和 node3，node2 和 node4 皆具有一個共同的 parent，所以 SimRank 值不為 0，為 0.538。

(2) 增加或刪除某些連結使 node1 的 Hub、Authority、PageRank 值提升:

增加 node4 指向 node1 的連結:

```
authority [0.499 0.001 0.499 0.001]
hub [0.001 0.499 0.001 0.499]
pagerank  [0.25  0.363 0.25  0.138]
```

根據分析的結果，增加 node1 的 Authority 要增加指向它的連結數，node4 連結到 node1 使得 node1 的 Authority 上升。node1 的 PageRank 值因收到 node4 的傳遞而提升，node2(收到來自 node1 和 node3 的傳遞)的 PageRank 值也因 node1 上升而略為提高。

增加 node1 指向 node4 的連結:

```
authority [0.001 0.499 0.001 0.499]
hub [0.499 0.001 0.499 0.001]
pagerank  [0.138 0.25  0.363 0.25 ]
```

增加 node1 指到 node4 的連結，node1 的 hub 值因此提升。

(3) 不同 damping factor 和 decay factor 討論:

damping factor = 0.3:

```
pagerank  [0.185 0.315 0.315 0.185]
```

damping factor 調大，傳遞的速度下降，因此同樣都是疊代 30 次，node1 的 PageRank 值上升(因為降比較慢)。

damping factor = 0.01:

```
pagerank  [0.167 0.333 0.333 0.167]
```

damping factor 調小，傳遞的速度上升，因此同樣都是疊代 30 次，node1 的 PageRank 值下降(因為降比較快)。

decay factor = 0.5 :

```
simrank:
 [[1.    0.    0.333 0.   ]
 [0.    1.    0.    0.333]
 [0.333 0.    1.    0.   ]
 [0.    0.333 0.    1.   ]]
```

從演算法可看出 decay factor 會影響最終輸出，之所以需要它是為了用來區分極高相似度和完全相同之間的差異，decay factor 越低代表受到前一次鄰居相似度的影響較小，因此經過疊代，值會往 0 靠近。

decay factor = 0.9:

```
simrank:
 [[1.    0.    0.818 0.   ]
 [0.    1.    0.    0.818]
 [0.818 0.    1.    0.   ]
 [0.    0.818 0.    1.   ]]
```

decay factor 越高代表受到前一次鄰居相似度的影響較大，因此經過疊代，值會往 1 靠近。

效能分析:

```
Graph1_HITS 執行時間：0.002992 秒
Graph1_PageRank 執行時間：0.002989 秒
Graph1_SimRank 執行時間：0.003989 秒
```

```
Graph2_HITS 執行時間：0.001995 秒
Graph2_PageRank 執行時間：0.002993 秒
Graph2_SimRank 執行時間：0.003003 秒
```

```
Graph3_HITS 執行時間：0.001994 秒
Graph3_PageRank 執行時間：0.002503 秒
Graph3_SimRank 執行時間：0.003530 秒
```

```
Graph4_HITS 執行時間：0.004987 秒
Graph4_PageRank 執行時間：0.005439 秒
Graph4_SimRank 執行時間：0.021977 秒
```

```
Graph5_HITS 執行時間：0.681112 秒
Graph5_PageRank 執行時間：2.258217 秒
Graph5_SimRank 執行時間：449.159470 秒
```

```
Graph6_HITS 執行時間：3.687667 秒
Graph6_PageRank 執行時間：14.035831 秒
```

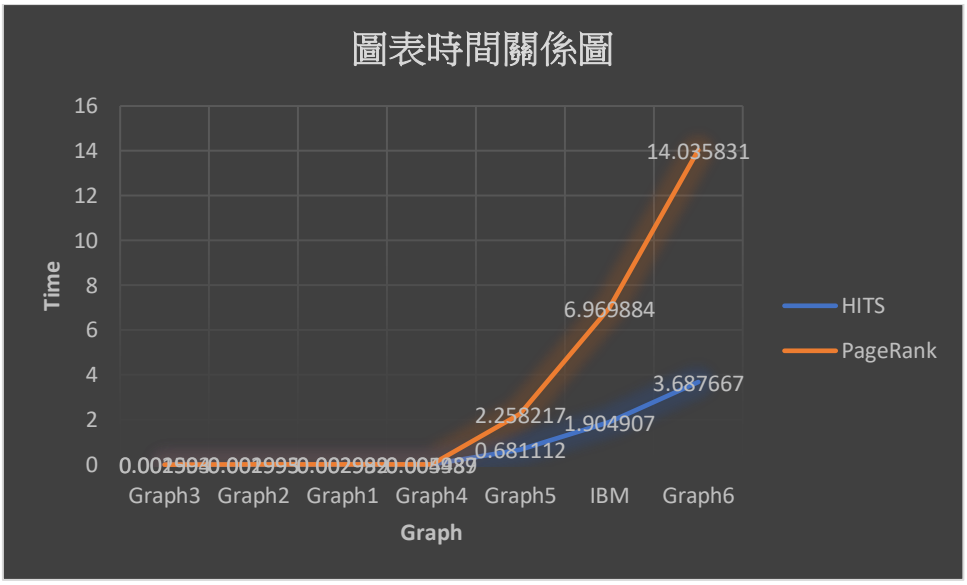
```
IBM_HITS 執行時間：1.904907 秒
IBM_PageRank 執行時間：6.969884 秒
```

時間複雜度:

PageRank: O(K| V |^2)，K 為 Iteration，V 為 Node 數

SimRank: O(K| E |^2)，K 為 Iteration，E 為 Edge 數

```
graph_1's number of node 6
graph_1's number of edge 5
graph_2's number of node 5
graph_2's number of edge 5
graph_3's number of node 4
graph_3's number of edge 6
graph_4's number of node 7
graph_4's number of edge 18
graph_5's number of node 469
graph_5's number of edge 1102
graph_6's number of node 1228
graph_6's number of edge 5220
ibm-5000's number of node 836
ibm-5000's number of edge 4798
```



從 Graph1 到 Graph5 的三種演算法花費時間可以看到，所需時間 HITS<PageRank<SimRank，針對 HITS 與 PageRank，先對 Node 數排序，去對時間繪圖，可發現隨 Node 數增加，HITS 與 PageRank 的時間差距越來越明顯，更不用說 SimRank 的時間複雜度與 Edge 有關，而 Edge 又是 Node 數的好幾倍，時間差距在 Graph5 就非常明顯了，更不用說不用跑的 Graph5 與 IBM-5000 這兩筆資料了。