**科⽬：1071\_資料探勘 DATA MINING**

**學⽣：楊沛霖**

**學號：Q36071156**

**日期：2018-11-18**

**作業要求：**

1. **利用課程給的八個Graphs來計算HITS、PageRank和SimRank。**
2. **運算所需的參數為random jumping probability**
3. **討論和分析每個演算法和結果。**
4. **各圖間的比較**

**graph**\_1:

是一條線單一方向的圖由1→2→3→4→5→6

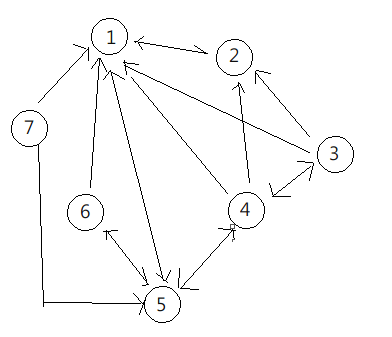
**graph**\_2:

是一條線但是方向是雙向的由1↹2↹3↹4

**graph**\_3:

是一條線單一方向的圖由1>2>3….>6

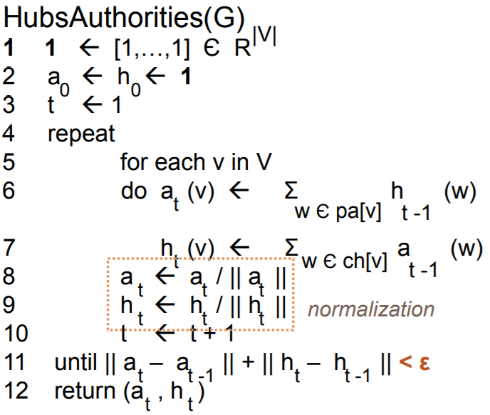
**graph**\_4:



**graph**\_5,6:

已過於複雜

1. **Hits**

實作方式:

* 1. 資料初始化,看有幾個點就創幾個a和h而他們的值皆為1(1.2.3行)
  2. 對每個有連接到at(v)的點的ht-1(w)作加總(第6行)
  3. 對每個有連接到ht(v)的點的at-1(w)作加總(第7行)
  4. 正規化(第8.9行)
  5. 重複2-5直到a和h與前項相減絕對值得加總小於誤差值
  6. 取得at和ht

D設定為：0.15

正規化方式為：平方後加總在開根號

**graph**\_1:

時間與cpu：

0.46s user 0.19s system 14% cpu 4.626 total

結果：

收斂執行次數 : 3

收斂結束時Error : 0.9190116821894447



**graph**\_2:

時間與cpu：

0.46s user 0.15s system 4% cpu 14.856 total

結果：

收斂執行次數 : 3

收斂結束時Error : 0.0

authority page :

{'1': 0.4472135954999579, '2': 0.4472135954999579, '3': 0.4472135954999579, '4': 0.4472135954999579, '5': 0.4472135954999579}

hub page :

{'1': 0.4472135954999579, '2': 0.4472135954999579, '3': 0.4472135954999579, '4': 0.4472135954999579, '5': 0.4472135954999579}

**graph**\_3:

時間與cpu：

0.46s user 0.15s system 4% cpu 14.856 total

結果：

收斂執行次數 : 3

收斂結束時Error : 0.24854976622531513

authority page :

{'1': 0.3922322702763681, '2': 0.5883484054145521, '3': 0.5883484054145521,'4': 0.3922322702763681}

hub page :

{'1': 0.3922322702763681, '2': 0.5883484054145521, '3': 0.5883484054145521, '4': 0.3922322702763681}

**graph**\_4:

時間與cpu：

0.44s user 0.15s system 48% cpu 1.256 total

結果：

收斂執行次數 : 3

收斂結束時Error : 0.5012605876917594

authority page :

{'1': 0.5063160398440778, '2': 0.23014365447458082, '3': 0.276172385369497, '4': 0.3222011162644131, '5': 0.552344770738994, '6': 0.4142585780542455, '7': 0.18411492357966464}

hub page :

{'1': 0.5063160398440778, '2': 0.23014365447458082, '3': 0.276172385369497, '4': 0.3222011162644131, '5': 0.552344770738994, '6': 0.4142585780542455, '7': 0.18411492357966464}

**graph**\_5:

時間與cpu：

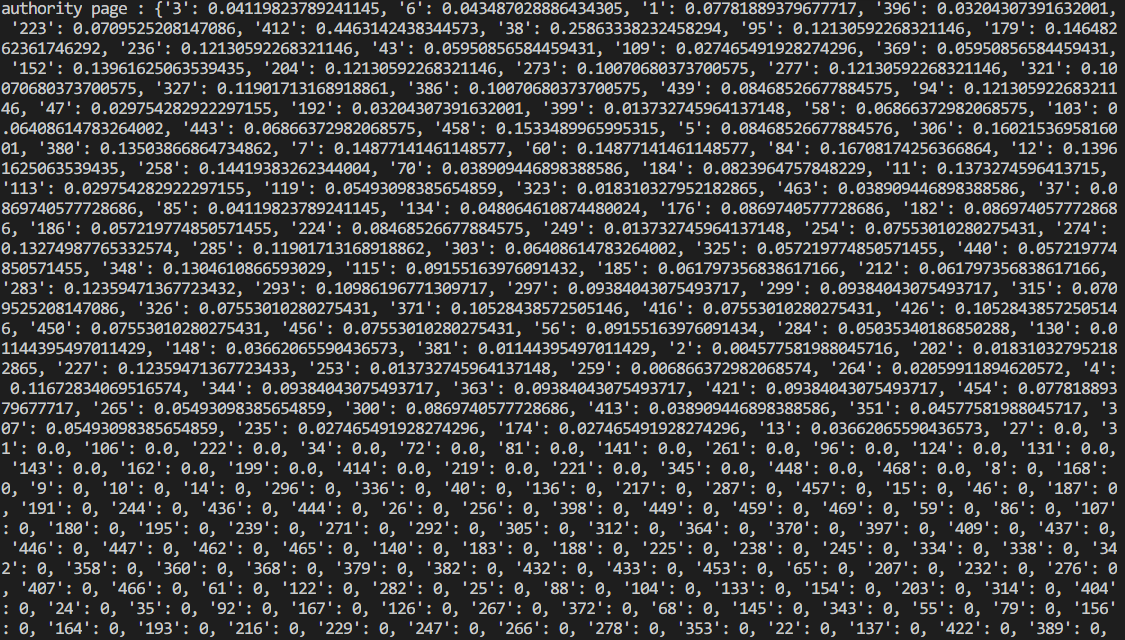
0.45s user 0.15s system 55% cpu 1.056 total

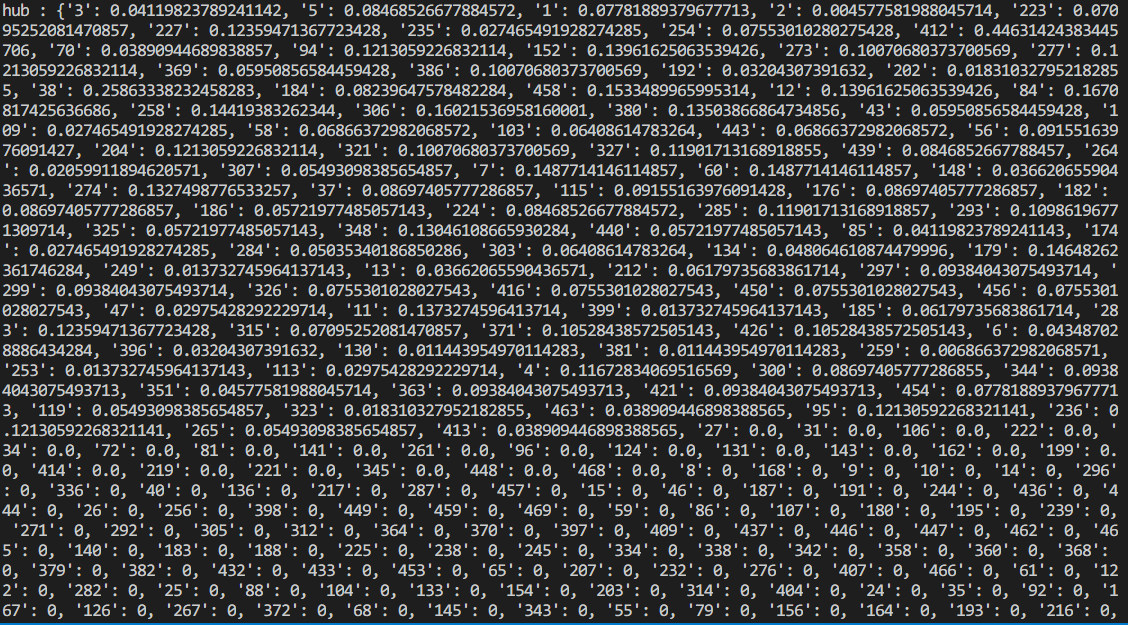
結果：

收斂執行次數 : 3

收斂結束時Error : 1.4092964448407446

authority page :

hub page :



**graph**\_6:

時間與cpu：

0.48s user 0.25s system 57% cpu 1.456 total

結果：

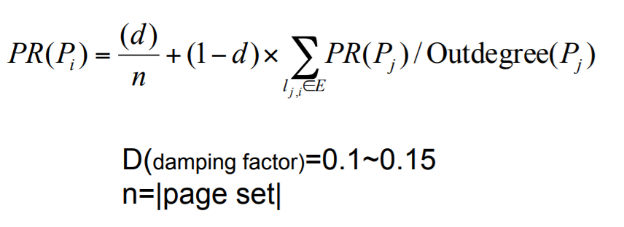
收斂執行次數 : 3

收斂結束時Error : 0.8876169361023691

資料量太大其他結果放在: some\_output\hits\_G6.txt

**討論:** 由於Hits的算法就算被多個node指但沒有對其它node指的點hub值會是0而如果都指向人而沒被指的話authority也會是0，而相較於PageRank Hits是比較容易被作弊的，比如作弊者可以建立一個網頁，頁面內容增加很多指向高質量網頁或者著名網站的網址，這就是一個很好的Hub頁面，之後作弊者再將這個網頁連結指向作弊網頁，於是可以提升作弊網頁的Authority得分。

1. **PageRank**

實作方式:

因此為Quick reference所以只要照著這個公式去實作即可。

為Ｐｊ的ＰＲ除上所有射入Ｐｊ點的數量

D設定為：0.15

**graph**\_1:

時間與cpu：

0.41s user 0.13s system 49% cpu 1.071 total

結果：

{'1': 0.024999999999999998, '2': 0.16666666666666666, '3': 0.16666666666666666, '4': 0.16666666666666666, '5': 0.16666666666666666, '6': 0.16666666666666666}

**graph**\_2:

時間與cpu：

0.34s user 0.07s system 92% cpu 0.471 total

結果：

{'1': 0.2, '2': 0.2, '3': 0.2, '4': 0.2, '5': 0.2}

**graph**\_3:

時間與cpu：

0.38s user 0.12s system 72% cpu 0.711 total

結果：

{'1': 0.14375, '2': 0.35624999999999996, '3': 0.35624999999999996, '4': 0.14375}

**graph**\_4:

時間與cpu：

0.33s user 0.06s system 92% cpu 0.381 total

結果：

{'1': 0.2946428571428571, '2': 0.1469047619047619, '3': 0.11654761904761904, '4': 0.07607142857142857, '5': 0.2683333333333333, '6': 0.05178571428571428, '7': 0.045714285714285714}

**graph**\_5:

時間與cpu：

0.41s user 0.05s system 92% cpu 0.781 total

結果： 由於數量眾多,只取前面常出現的數值,以下格式為:

(數值,出現次數)

(0.0005787389582698751, 12), (0.00044065387348969436, 15), (0.00044928419128845566, 17), (0.00041521714734597684, 19), (0.00045924225028702635, 20), (0.0005463752665245202, 20), (0.0004264392324093816, 20), (0.00039534470504619753, 20), (0.0003923240938166311, 24), (0.00036752328582650653, 25), (0.0004331023454157782, 30), (0.0003663003663003663, 39)

**graph**\_6:

時間與cpu：

0.45s user 0.04s system 92% cpu 0.981 total

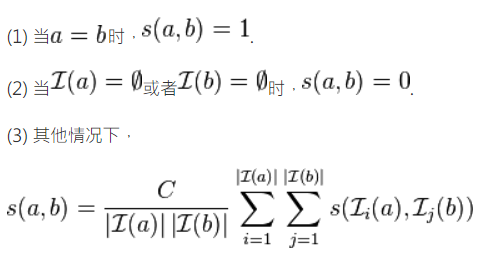
結果： 由於數量眾多,只取前面常出現的數值,以下格式為:

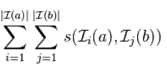
(數值,出現次數)

(0.00015099077090119435, 21), (0.00013248091788613933, 21), (0.00015361267397098017, 24), (0.0001452225841476656, 25), (0.00013788125555226532, 28), (0.00014312506169183694, 29), (0.00015511090429657206, 32), (0.00014983713355048862, 35), (0.00014687063750581667, 39), (0.00014378053745928339, 43), (0.0001425081433224756, 46), (0.00015224472454326583, 48), (0.00014447830198591993, 48), (0.00014877223753445253, 51), (0.00013903233494875664, 62)

**討論:** 相對於Hits PageRank的算法是比較全面性的,因為Hits只需要負責兩個網站之間的關係,但是PageRank則會牽扯到整個網路(會去抓指向該node的node的資訊)

1. **SimRank**

實作方式：

意思為執行每個射入的node兩兩做SimRank並且加總

Ｃ為：０６

搜尋深度為：10

**graph**\_1:

時間與cpu：0.45s user 0.12s system 20% cpu 2.814 total

執行結果：除了node 相同的為1以外的node與node都為0

討論:

由於G\_1是一個由1>2>3….>6一條線的圖, 因此除了自己和自己其他的射入點都為0

**graph**\_2:

時間與cpu：0.37s user 0.83s system 70% cpu 0.605 total

執行結果：除了node 相同的為1以外的node與node都為0

**graph**\_3:

時間與cpu：７.88s user 0.11s system 90% cpu 8.607 total

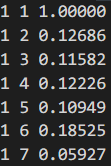
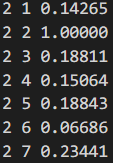
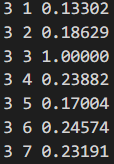
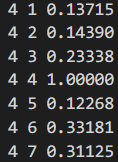
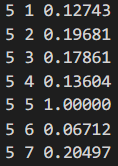
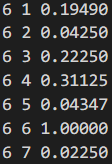
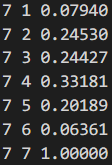
執行結果：

除了點(1,3)(3,1)(2,4)(4,2)為0.42856、node 相同的為1外，其他都為0

**graph**\_4:

時間與cpu：0.35s user 0.06s system 95% cpu 0.432 total

執行結果：

**graph**\_5:

時間與cpu：0.35s user 0.06s system 95% cpu 0.432 total

執行結果：除了node 相同的為1以外的node與node都為0

**討論:**

對於SimRank的算法比起Hits更接近PageRank,因為他們的關聯度都不單單只是兩個網頁,而是在整個網路上,而SimRank更是將跌代的概念用的更為貫徹.

1. **增加node 1的hub, authority, and PageRank** 
   1. **graph**\_1
      * Hits:可以再增加一個點在node1的前面指向node1即可增加他的hub和authority
      * PageRank:可以拿掉node6即可
   2. **graph**\_2
      * Hits:只要再隨意找一個node並且將它指向node1即可
      * PageRank: 只要再隨意找一個node並且將它指向node1即可
   3. **graph**\_3
      * Hits:只要將node4指向node1即可
      * PageRank: 只要將node4指向node1即可
2. **問題討論**
   1. More limitations about link analysis algo
      * 在這些演算法中都必須要設定最大深度才行，否則就會出現永遠跑不完的情況出現，而這個深度的其實也不是越多越好，在這些dataSet中常常只要跑個3.4次就可以知道有沒有循環的輪迴了。
   2. What are practical issues when use in real web
      * 在真實的web中應該沒辦法像我們的dataset一樣可以看到他的完整有向圖，因為你在逛網頁的時候那個網頁通常只知道你去了哪裡，而沒辦法知道你是從哪裡來(in\_edges)
   3. What do the result say for your actor/movie graph
      * 我們可以從演算法裡面得知這個graph的哪個node有著比其他node都還要高的權重(重要性),比起人用肉眼看還要準確並且有明確性
   4. What is the effect of ‘C’ in SimRank
      * C的值影響性其實沒有我一開始想的還要大，由於在同一個圖中Ｃ的值基本上是一樣的，因此對其他node來說反而沒意義，除非拿來跟不同的圖的node做比較才會顯示出他的用意。不過在同一個圖中也會因為C所以不段的滾動某個較長的邊的node值
3. 心得

這次的project和前幾次的很不一樣,前面的都可以用肉眼稍微計算一次就可以得到答案,但這次的需要大量的運算,真的很佩服當初可以想出這些演算法的科學家們。而在這次的project中學到最大的應該就是圖形的應用了吧!原本還在思考要用怎麼樣的方式去把圖給實做出來,但幸好後來找到了networkx讓圖這邊可以快速地找到inoutedge,真的方便了許多。