姓名：黃啟軒

學號 : F84004022

網路資訊檢索與文字探勘

Project 2: Link Analysis Practice

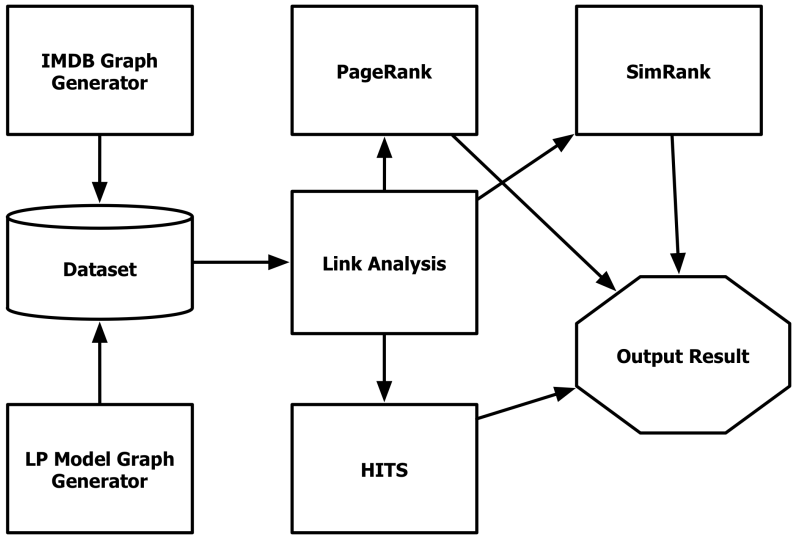
一、Introduction

連結分析是一種用來評斷兩個nodes關係的技術，在本次的project中，我將實作三個連結分析的演算法，亦即為PageRank、HITS、SimRank，前兩個演算法是搜尋引擎中用來分析網頁排序著名的演算法，後者則為分析節點相似度的演算法。

實驗資料集將包含10個graph dataset，六個hw2 dataset，兩個從LP model中產生，一個從IMDB產生，最後一個為從其他課程中取得的資料集。

在後續的章節中，我將繼續介紹實作的流程和架構、Graph的特性、和結果的分析及討論。

二、Implementation Detail



圖（一） 程式流程

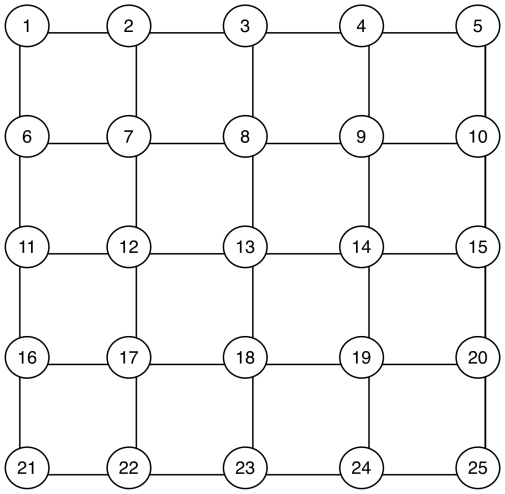
本次程式是以python作為程式語言開發，程式流程如圖（一）所示，除了原本hw2dataset及自己產生的graph之外，分別會有兩個程式來產生IMDB Graph和LP Model Graph，輸出到原本的dataset。

之後Link Analysis 程式將分別讀入這些dataset以且自訂的Graph 物件的形式產生，同時這個物件實作了提供nodes和all in-neighbors和all out-neighbors的查詢。

在三個連結分析演算法部分，我分別寫成了三個模組，並在Link Analysis程式引入依序執行，最後產生以json格式的結果檔案。pageRank、HITS、SimRank三個演算法，最大iteration次數分別為20、40、20，除了SimRank用array，其他兩者皆用hash取代以減少空間和增加效率。

三、Characteristics of graphs

• Characteristics of LP model graphs



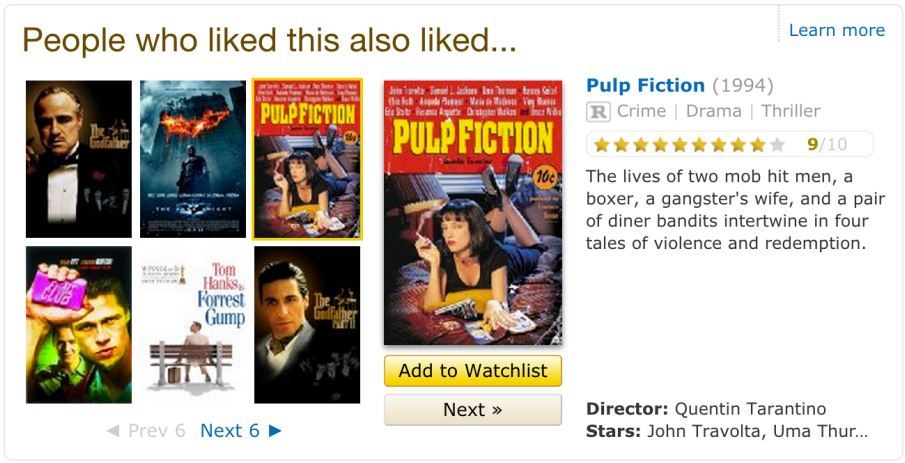
圖（二） LP model Graph

LP model graphs的產生從先以regular network初始化(4x4 grid, 25 nodes)，如圖（二）所示，之後以0.2及0.8的rewire probabilities產生要分析的graph，並輸出成Json格式。這種graph本身的初始的特性是中間的連接比較多，且degree相同，因此在沒有rewire的情況下，中間的rank其實會相同。

• Characteristics of IMDB graph

在IMDB的分析中，我想探討排行榜上的電影和分析後排序的電影的關係，IMDB的graph的產生，我parse了IMDB電影的頁面，方法是利用排行榜上top-50的電影[1]來當root set而利用“People who liked this also liked…”的推薦連結（圖三）當成out-edges，擴張一層當作 base set，最後graph中有112個nodes。

[1] http://www.imdb.com/chart/top?ref_=nv_ch_250_4



圖（三）IMDB推薦區塊

• Characteristics of graph I generate

這個自定義的graph是從其他課程的dataset中產生，由於這個dataset剛好就是模擬網頁格式，且連結用是以 http://page2.txt 表示，其樣式如圖（四），其中共有5個page，因此拿來觀察與分析。

雖然說這種方式導致單向度的社會，但在資源的應用與實務上，本來就不可能用太多資源去做市場區隔，更何況當面對每一個人不同的須求，最簡單，花最少資源，賺越多的錢就是給你最暢銷的商品，若你不買單就想辦法說服你這是你要的商品，你是甚麼樣的人並不重要，而你買了之後發現這東西可能不是真正你想要的，你買下的是商品背後的"標價虛榮"，以及你以為獲得的情境，其實不然。

透過資料探勘可以做到的是超過傳統行銷學想像的，利用各種行為去計算出關係與距離，就可以知道人與人之間的關係，商品與商品之間的關係，這種方法進一步的還可以算出每一個人對每一個商品的購買機率，因此當你進入商場或網路商城，就可以知道你會買甚麼東西是做得到的。 http://page3.txt

但拉回來說，我們都知道一個好的店長對於回頭消費的顧客當他出現時，心理就會有個譜，他會買甚麼東西都瞭如指掌，這比任何資料探勘還要更準，只是這前提是要熟的客戶與小的商場，當商品變多，客戶變多就做不到了，目前而言，資料探勘能做到的事，事實上人都做得到，甚至更好，但一個好的店長是相當少見，甚至他也只能了解幾十到幾百個客戶就了不起了，此時資料探勘就會在這邊發揮效用。
http://page2.txt

一個好的資料探勘系統，是能夠計算相當大量的資料，這是以百萬到數十億等級以上的資料來計算的，且能夠在一定的時間(幾秒到幾分鐘)內幫助數十萬到數百萬或足夠多的用戶，讓他們透過這樣的資訊來幫助他們的決策，進一步的完成使用者的須求，達成提升業績的結果，而前面的"資料量"，"時間限制"，"使用者廣度"是資料探勘系統是否成功重要的要素。
而往往因為無法達到這樣的要求，或者是開發者本來就不知道這樣的演算法，而選擇用統計式或條件式(Rule-BasedorCondition-Base)來做出這樣的輔助決策系統，當然好壞的演算法做出來的準確度有差，甚至有時候須要為了整個企業智慧(BusinessIntelligence)做一段時間的調校才能夠準，所以有時不是做得好不好的問題，而是做得出來與否的問題。

不得不否認的，用條件式就以達到2~3成的準確率，若是列出10個有2~3個準確也就很夠了，因此就聽到Cubie的Tempo說，他們當時想利用DataMining 做朋友推薦，但因為沒有時間就先做條件式的，而效果已經足夠了，這句話代表他很了解資料探勘的好處與困難，也知道與一般條件篩選式與之間的差異。而資料探勘就是從統計因子分析的2~3成的準確率變成7~8成，雖然說效果多了2~5倍，但須要的時間，資源的成本就高很多，若這是只是個推薦朋友的話，或許用統計條件就夠了，若這個是在大型電子商務的商品推薦的話，這就是很重要的工具與決勝點了。

圖（四）

四、Result analysis

• Result of HW2 dataset graph

表（一）列出了Graph1 ~ 4 pageRank和HITS的結果，我們可以透過增加node1 out-edge到其他的node和增加in_edge的node提高 node1的hub, authority, and PageRank，比方說graph1中增加 1, 3 和 4, 1。

從表（一）我們可以看到node 1和node1 和 node 6 以pageRank來看排序在後面，這是因為Graph1有“Rank Sink”的問題，node1是沒有parent，node6是沒有children，因此pageRank一個是最小值，一個會是沒有答案值。至於HITS的值除了node 1和node6（一

個有hub值，一個有auth值）其他都相同。同時在SimRank上因為沒有node有同樣的in-node，因此爲單位矩陣。

Graph2是一個單一cycle的圖，從結果來看其實每個node的pageRank都很接近於1，實質上是可以推測如果再繼續增加iteration其值會都變成一樣。因此HITS的auth和hubs都是一樣的值，而SimRank同樣是單位矩陣。

表（一）Graph 1 ~ 4 HITS & PageRank

Graph dataset	HITS結果排序 auth	HITS結果排序 hub	PageRank 結果排序
1	3, 2, 5 , 4, 6 , 1 （2~6相同）	1, 3, 2, 5, 4, 6（1~5相同）	5, 4, 3, 2, 1, 6
2	1, 3, 2, 5, 4	相同	相同
3	3, 2, 1, 4	3, 2, 1, 4	3, 2, 4, 1
4	5, 3, 2, 4, 1, 7, 6	1, 4, 5, 6, 3, 7, 2	1, 5, 2, 3, 4, 7, 6

Graph3中的HIST值 node1 和 node 4 接近， node2 和 node3接近，這是因為他們的in-degree 和out-degree接近，然而儘管如此，在SimRank的部分其實是 node1和 node3相似，node2 和 node4相似，這是因爲SimRank演算法是看連入的node的關聯性，而非單純連入或連出的數目。

Graph4 是一個edge數目較多，且在相似度分析上值都較高的graph(在simRank的部分，每個值幾乎都大於0.5)，Graph4的PageRank和HITS則沒有顯著的特徵。

Graph5和Graph6是兩個比較大的Graph，由於篇幅問題，在此並沒有呈現完整的結果。從Graph5的結果而言，node1~5可能都是屬於沒有parent的node，因此無論是pageRank和HITS auth都是最低的值，而於有它們都有out edges因此HITS hubs較高。

Graph6 屬於一種 hubs 集中在幾個 nodes，而分散指向其他nodes的graph，因此在HITS的結果中，hubs的值很多會是0，而auth是則均勻分散。

- Result of LP model graphs

LP model graph的節點編號如圖（二）所示，用來分析的兩個LP model graphs分別以0.2及0.8的rewire probabilities產生，其結果如表（一）所示。從這個結果可以看到，越接近Organized Networks中間的節點rank會越高，相反的越接近“Disorganized” Networks可以提高一些比較低分的節點，進而產生更多可能性。

從這些結果來看，我認為rewire probabilities高一點會比較接近實際上的一些network，以社群網路為例，重新連結產生越多代表互動越多，而任何的擾動都是在這社群上的活動。相對地，rewire probabilities低的可能就是代表一些比較靜態的網路。

表（一）LP model graph TOP-10 結果排序

rewire probabilitie	HITS結果排序 auth	HITS結果排序 hub	PageRank 結果排序
0.2	12, 18, 8, 14, 16, 20, 22, 6, 24, 2,	13, 17, 7, 19, 9, 11, 23, 15, 18, 3	1, 6, 2, 3, 7, 11, 4 , 8, 5, 12
0.8	17, 12, 19, 18, 8, 14, 16, 4, 25, 13	13, 18, 9, 17, 19, 11, 20, 12, 7, 22	20, 2, 11, 3, 7, 1, 8, 12, 16, 5,

- Result of IMDB graph

在這小節，首先先列出了兩個演算法結果中前5部排序的電影，如表（二）所示，從這結果中可以發現出對於HITS演算法，前5的authorities大部份落在原本IMDB排行榜上的10~30名之間，而hubs則是屬於原本排名比較前面的電影，如“The Shawshank Redemption”原本排名第一，“The Godfather”原本排名第二，PageRank的結果則是散落在不同的區段中。

在IMDB graph分析中，呈現IMDB推薦的電影，大多數會落在排行榜的中間到前半區間，我想是因為排名前的電影會指向中間區段，而後面的電影也會指向前面的電影，但是這樣的結果也可能是跟目前的dataset大小有關。

表（二）IMDB電影graph分析結果

演算法 \ 排名	1	2	3	4	5
HITS結果排序 auth	Forrest Gump	The Matrix	Fight Club	Pulp Fiction	Se7en
HITS結果排序 hub	Goodfellas	The Shawshank Redemption	The Silence of the Lambs	The Godfather: Part II	The Godfather
PageRank 結果排序	Interstellar	It's a Wonderful Life	Gone Girl	Fury	The Princess Bride

- Result of graph I generate

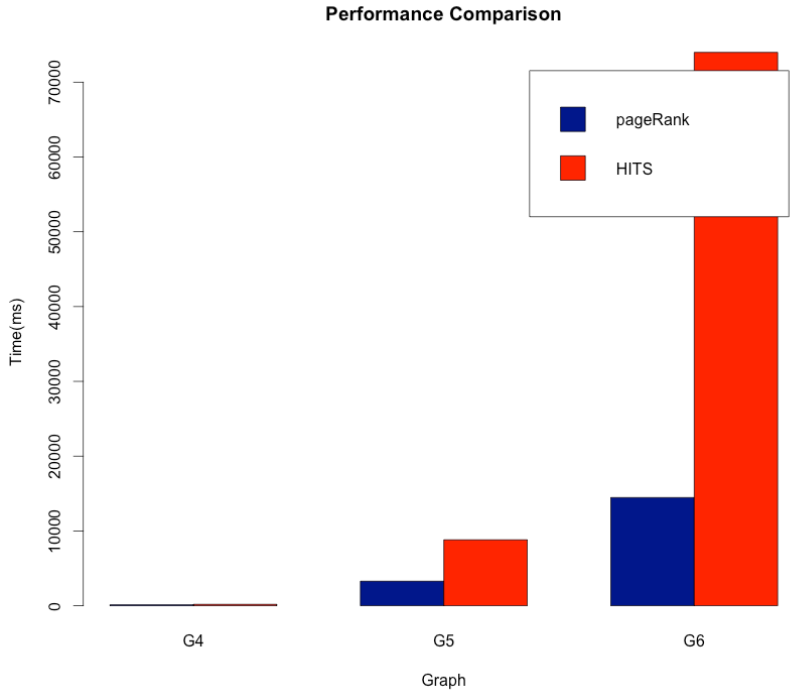
其結果如表（三）所示，在這個資料中，我覺得會是個很好的例子，用來解釋說link analysis的限制。由於我們並沒有去做文字處理和搜尋，而是單純透過link analysis排名，雖然它或許可以顯示出比較重要的頁面，但其實不一定是我想要找的頁面。

表（三）自定義的graph分析結果

HITS結果排序 auth	HITS結果排序 hub	PageRank 結果排序
page3, page2, page4, page1, page 5	page1, page4, page2, page3, page5	page1, page3, page2, page5, page4

五、Computation performance analysis

在performance部分，比較了PageRank和HITS在同樣的iteration次數下graph 4~6的時間，我們可以從圖（五）雖然PageRank和HITS的時間複雜度差異不大，但當graph越大的時候，可以清楚地發現PageRank地效率相對好很多。



圖（五）Performance: PageRank vs HITS

六、Discussion

在這個project中我透過實作深刻了解到這三個演算法的特性，雖然Google是以PageRank崛起，但我覺得HITS其實比較有趣，或是這也是在學術上HITS評價比較好的原因，然而在效率上卻是真的不如Page Rank。

我覺得在資料集中IMDB是比較有趣的case，而最後一個graph原本想用freebase的graph，但卻因為沒有時間則作罷。

我認為link analysis其實只能達到一定的效果，因為回到搜尋的議題上，我們需要思考的是什麼才是使用者所需要的資訊？由於link analysis分析其實沒有並考慮到頁面的內容和語意，就像是之前提到的問題，其實link也可以被造假，如果單純依照這個演算法，很容易會有問題，因此我認問在網頁上它只能是一項評斷的指標，但不能代表一切。

而在實際的情況上，是有好幾億的網頁，Graph是非常複雜的，當數量是如此龐大的時候，光是單純要執行就會有很多的問題，加上跑出來的結果很可能出現一堆非常相近的頁面（但具有誤差），這時候要怎麼解決也會是個問題。