**Program Description File**

資訊管理研究所 林家輝 R76061045

1. **資料集說明**
2. 此次作業採用**八組圖型**，六組採用課程提供之資料集；另外一組使用IBM Quest Data Generator 產生交易集，**總交易數量為300筆**、**平均每筆交易共8個商品、共15個不同商品**，我們以單筆買賣項目為單位；最後一組圖型則是將上述IBM Quest Data Generator 產生出來的交易集，**放到Weka的Associator實作Apriori演算法，最後將產出的關係當作圖型**。將此八個圖型分別以HITS、Page Rank、Sim Rank三種演算法，算出各個節點之分數，並加以比較不同演算法實作出來的結果差異和限制。(Sim Rank只實作前五個圖型)
3. **程式碼撰寫過程**

* **Pagerank implementation :**

1. **讀取資料:**

先利用**FileReader**,老師給的txt檔案吃進來。再用**Scanner s = new Scanner()**;的函式將老師給的函式吃進暫存區，再透過**nextLine()**的函式一行一行讀取存入陣列中。

1. **算出圖論有幾個點以及做出圖論矩陣:**

透過**flag標籤判斷有無相同**，若有才加入陣列中藉此來判斷有幾個結點**colum以及row必須要兩個同時判斷**。並建立矩左邊為入射點右邊為射入點的二維圖論矩陣。且給予表格**初始值為1/N。**

1. **計算每個點的流入數量以及流出數量:**

用兩個for迴圈計算數量，其中**i,j互換即可分別算出流入以及流出的數量**。

1. **設定一個若向量距離大於設定的收斂數值時就繼續做的迴圈**
2. **算出每個點的新pagevalue:**

利用兩個for迴圈,依序算出每個點的outgoing的數量,且將當下點的pagevalue除以該點的流出數量，一欄做出一個加總得出一個temp值，最後再透過**finalvalue = temp \* d + (1 - d) / Nodenumber**的公式算出新的pagevalue並存到新的表格中。

1. **算出兩個向量的幾何平均距離並將新表格複印到舊表格後進行疊代:**

將算出的值丟入while迴圈中做判斷，**若為true則繼續執行疊代直到整個迴圈收斂為止**。

* **HITS implementation :**

1. **讀取資料:**

先利用**FileReader**,老師給的txt檔案吃進來。再用**Scanner s = new Scanner()**;的函式將老師給的函式吃進暫存區，再透過**nextLine()**的函式一行一行讀取存入陣列中。

1. **算出圖論有幾個點以及做出圖論矩陣:**

透過**flag標籤**判斷有無相同，若有才加入陣列中藉此來判斷有幾個結點**colum以及row必須要兩個同時判斷**。並建立矩陣左邊為入射點右邊為射入點的二維圖論矩陣。

1. **建立兩個根據點查詢該點的Authority與Hub的陣列:**

Authority與Hub分別必須建立新舊兩個陣列，為了讓疊代可以順利進行，其中舊的表格先給**予初始值為1**。

1. **設定一個若向量距離大於設定的收斂數值時，就繼續做的while迴圈**
2. **算出每個點新的Authority值與Hub值:**

Hub以及Authority進行依序加總，公式為**HubA=A(B)+A(C)+A(D)**以及**AuthorityA=H(1)+H(2)+H(3)**。且將新的值更新在新的陣列中。

1. **算出正規化的距離值:**

此值為Authority的新舊向量的距離幾何平均加上Hub的**新舊向量的距離幾何平均**。

1. **進行正規化後算出兩個向量的幾何平均距離並將新表格複印到舊表格後進行疊代:**

將算出的值丟入while迴圈中做判斷，若為**true則繼續執行疊代直到整個迴圈收斂為止**。

* **Simrank implementation :**

1. **讀取資料:**

先利用**FileReader**,老師給的txt檔案吃進來。再用**Scanner s = new Scanner()**;的函式將老師給的函式吃進暫存區，再透過**nextLine()**的函式一行一行讀取存入陣列中。

1. **算出圖論有幾個點以及做出圖論矩陣:**

透過**flag標籤**判斷有無相同，若有才加入陣列中藉此來判斷有幾個結點colum以及row必須要兩個同時判斷。並建立矩左邊為入射點右邊為射入點的二維圖論矩陣。

1. **做出單位矩陣:**

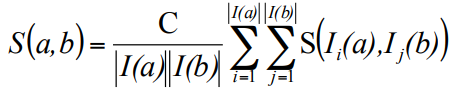
利用**兩個for迴圈先將所有的值給0，在將所有i=j的值給1**。

1. **計算每個點的流入數量**

用兩個for迴圈計算流入該點的數量，以利後面的公式計算。

1. **設定一個若向量距離大於設定的收斂數值時，就繼續做的while迴圈**
2. **算出每個點與點的相關性且更新到新表格中:**

先利用兩個for迴圈依序找出兩個要算相關性的點，其中若兩個點i=j則相關性為1，再去抓出每個點的父點並且判斷**若其中一個點沒有父點則相關性為0**，之後我們才透過一下公式算出兩兩點之間的相關性



1. **算出兩個二微陣列的幾何平均距離並將新表格複印到舊表格後進行疊代:**

將算出的值丟入while迴圈中做判斷，若為**true則繼續執行疊代值到整個迴圈收斂為止**。

1. **各圖形在不同演算法觀察**
2. **Graph1**

* **Pagerank:**

因為圖形是一條龍的接下去，所以越後面的點分數越高。

* **HITS:**

因為Authority值要透過母點，所以點1的值為0。Hub要透過子點來計算故其值為0。其他點的母點子點數量都一樣且初始值也一樣故分數一樣。

* **Simrank:**

點與點之間都沒有共同母點，故值為0。

1. **Graph2**

* **Pagerank:**

因為圖形是一個環形的點，因此每個點的射入點，射出點以及點的初始值都一樣所以每個點的初始值都一樣,因此算出來的值都為相等。

* **HITS:**

同上因為環狀的條件都一樣，故每個點的Authority&Hub值都一樣。

* **Simrank:**

環狀圖形點與點之間都沒有共同母點，故值為0。

1. **Graph3**

* **Pagerank:**

因為圖形為點與點之間環環相扣，其中點1與點4因為只有一個射入點故分數較點2與點3有兩個射入點來得低。

* **HITS:**

因為點2與3射出的點較多故其Hub值較高,並且2,3的高Hub值會互相指向對方提升彼此的Authority.故點2與點3的值相對點1與點4來得高。

* **Simrank:**

在環形圖中，點1與點3有共同的母點以及點2與點4有共同的母點有值，其他的點均沒有共同母點故沒有值。

1. **Graph4**

* **Pagerank:**

在此圖形中，學生觀察到點一的射入點最多，因此分數為最高而點6與7射入點最少故分數最低。

* **HITS:**

相對於Pagerank,HITS會考慮到射入點的Hub權重，因此點1的射入點雖然最多，但因為考量到射入點的Hub值因此點5的值變成最高，但點1的Hub值因為射出點的Hub值較高故點一的Authority仍為最高。

**Simrank:**

由此圖形演算法會參考到母點在第二次疊代甚至會參考到母點的母點，根據這樣的特性可以看出點4與點7的相關性為最高，因為點4與點7的母點，點5與其他點都具有高度相關性。

1. **Graph5**

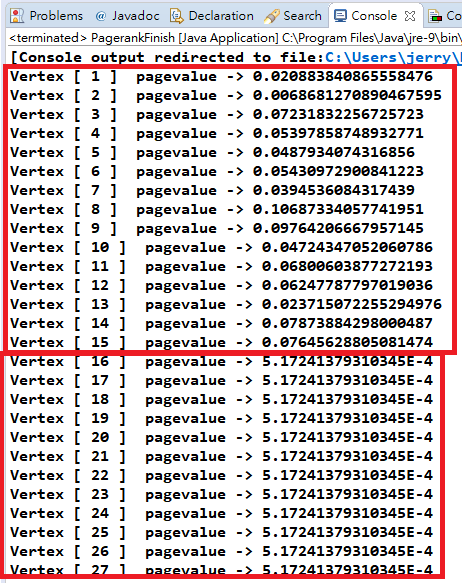
* **Pagerank:**因點太多，因此透過程式直接做比較。
* **HITS:** 因點太多，因此透過程式直接做比較。
* **Simrank:** 因點太多，因此透過程式直接做比較。

1. **Graph6**

* **Pagerank:** 因點太多，因此透過程式直接做比較。
* **HITS:** 因點太多，因此透過程式直接做比較。
* **Simrank:** 因點太多，因此透過程式直接做比較。

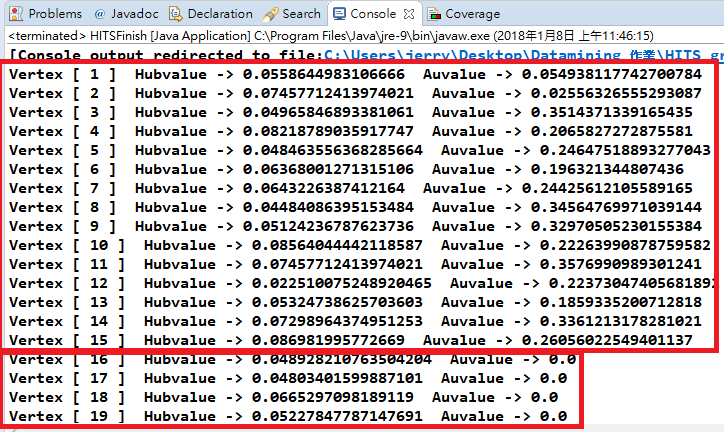
1. **Graph7**

* **Pagerank:**

****

因為此圖形為15個搜尋網站被其他285個網站連結，因此可以看到前15個網站的值相對其他285個網站來得高很多，且其中又以點8的搜尋網站為最火熱的網站。

* **HITS:**

****

呈上題的網頁架構，被參照的15個搜尋網站的Authority最高。

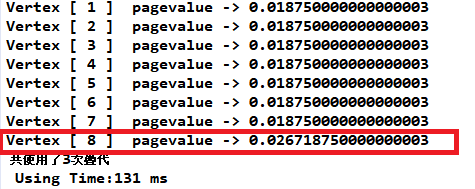
* **Simrank:**

因為圖形整體為多個網站常去連結到搜尋網站，因此母點的關連性的值非常的低收斂後期值為0。

1. **Graph8**

此部分為產生confident>0.93的關聯資料，學生在網頁世界裡假想成預覽這個網站就很有可能預覽下一個網站的網站架構圖。

* **Pagerank:**

****

**由此可知點八為搜尋某網頁一定會搜尋的網頁中最常被搜尋到的網頁。**

* **HITS:**

由上述的網路架構得到的解為，最高頻率被參照的網頁的Authority為最高也就是點11在來是點8。

* **Simrank:**

只有點8以及點11之間有共同母點，其他的點則沒有共同的母點。

1. **實驗結果與分析**

* **實驗結果表格**

**(以圖五，六為例，因圖形較大時間較能比較，其他圖結果會附在壓縮檔中)**

**(圖五)**

|  |  |  |  |
| --- | --- | --- | --- |
| 比較項目 | **Pagerank implementation**  algorithm | **HITS implementation**  algorithm | **Simrank** **implementation**  algorithm |
| 分析正確率 | 正確 | 正確 | 正確 |
| 進行疊代次數 | 9次 | 99次 | 35 |
| 過程使用空間 | 16,384 kbytes | 15,360,kbytes | 89,422 kbytes |
| 執行時間 | 182ms | 3246ms | 1247481 ms |

**(圖六)**

|  |  |  |
| --- | --- | --- |
| 比較項目 | **Pagerank implementation**  algorithm | **HITS implementation**  algorithm |
| 分析正確率 | 正確 | 正確 |
| 進行疊代次數 | 7次 | 99次 |
| 過程使用空間 | 14,336 kbytes | 16,384,kbytes |
| 執行時間 | 16143ms | 2280ms |

* **實驗結果分析**

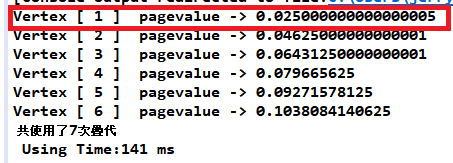
由圖五來比較三個演算法可以看出**收斂速度**最快的為Pagerank再來是Simrank最後是HITS。**使用空間最大者**為Simrank**學生想原因是因為**

**必須要做兩個二微陣列去做新舊之間的更新**，Pagerank以及HITS只需要一個一維陣列去做更新。**執行時間最快**者為Pagerank再來是HITS最後是

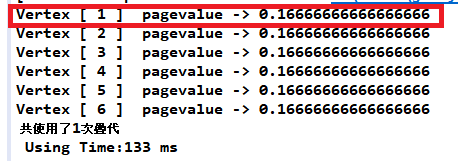
Simrank。學生想**其中的原因除了與疊代進行的次數外還跟時間複雜度有關，因為Simrank演算法中有四層for迴圈**，也因此每一次的疊代都要花比較久的時間，最後最後的執行時間將盡快要20分鐘。

1. **增刪連結束使得結點1的Pagerank,Authority,Hub增加**

* **圖形1(Pagerank)**
  1. **增加一條(6,1)**
  2. **增加前:**

****

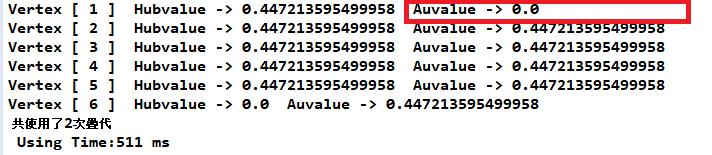
* 1. **增加後**

****

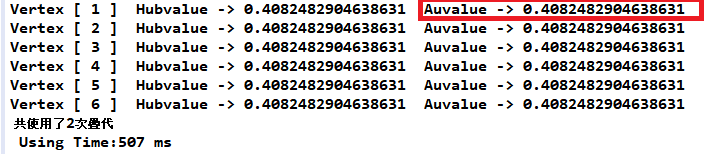
* **圖形1(Authority)**

**1. 增加一條(6,1)**

**2. 增加前:**

****

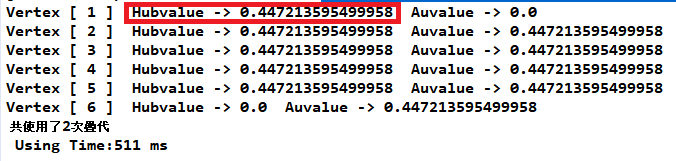
**3. 增加後**

****

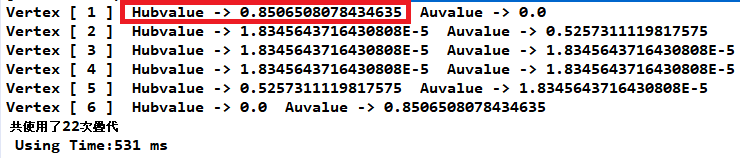
* **圖形1(Hub)**

**1. 增加一條(1,6)**

**2. 增加前:**

****

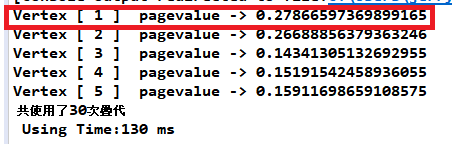
**3. 增加後**

****

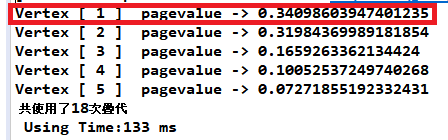
* **圖形2(Pagerank)**

**1. 增加三條(2,1), (3,1), (4,1)**

**2. 增加前:**

****

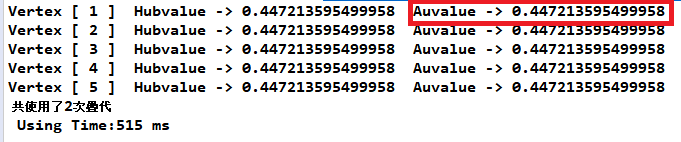
**3. 增加後**

****

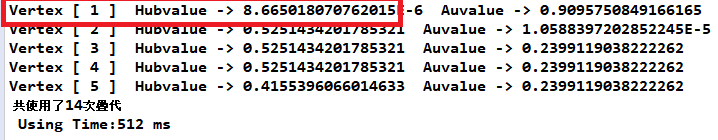
* **圖形2(Authority)**

**1. 增加一條(6,1)**

**2. 增加前:**

****

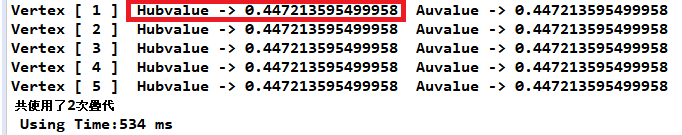
**3. 增加後**

****

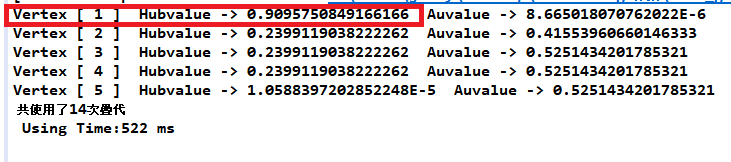
* **圖形2(Hub)**

**1. 增加三條(1,3) (1,4) (1,5)**

**2. 增加前:**

****

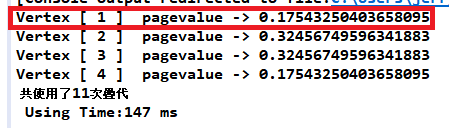
**3. 增加後**

****

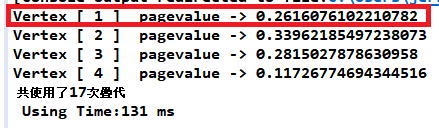
* **圖形3(Pagerank)**

**1. 增加一條(3,1)**

**2. 增加前:**

****

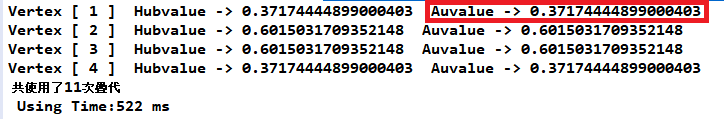
**3. 增加後**

****

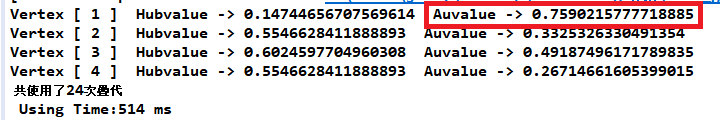
* **圖形3(Authority)**

**1. 增加兩條(3,1) (4,1)**

**2. 增加前:**

****

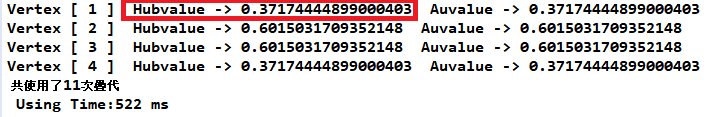
**3. 增加後**

****

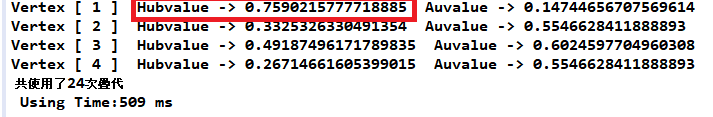
* **圖形3(Hub)**

**1. 增加一條(1,3) (1,4)**

**2. 增加前:**

****

**3. 增加後**

****

1. **作業遇到的困難與成長**

* **網頁分數計算觀念的理解**

1. 從完全不知道怎麼做網頁算分 -> 上完課以及實作完後後**懂得三個關計算網頁分數演算法所有的流程與運算邏輯**。

* **數學公式理解能力**

1. 從因為要寫出三個演算法之前，**必須透過理解演算法流程以及公式的理解才有辦法實作寫出程式**，故在過程中透過**不斷揣測公式的目地的過程中提升了數學公式的理解能力與速度**。
2. 學會用甚麼方法可以使得**公式所求的值收斂**，即為透過算出向量距離在進行正規化。

* **程式撰寫能力**

1. 提升將**程式進行邏模組化**，使得在寫程式時可以透過直接配對創造出更多

有彈性且不同功能的程式。

1. **學會利用圖形的結構**，理解並熟悉如何透過二維陣列操作圖論的每個結點。
2. **提升debug的能力**，因為學生在這項作業的努力時間至少70個小時以上，學生明顯感受到對程式的熟悉度以及思考錯誤以及修正錯誤的能力大幅提升。
3. 提升程式撰寫的速度:學生過去在寫程式時都會習慣一次把程式想寫的邏輯寫出來後再去做debug,但在這次作業中學生採用**每寫完一個段落就利用System.out.print()的函數來進行確認**，學生發現這樣的作法較好，原因是因為如果整棟大樓蓋好再去找問題在哪裡真的會花非常多的時間，不如一層完就做檢查在往下一層，如此才不會花費太多時間在尋找bug在哪裡進而提升效率。
4. 學會使用JAVA套件中**開始時間與結束時間的函數，如此一來即可以透過兩兩相減得出此程式的執行時間**。
5. 學會使用java中的**jdk-9的console做出memory的測試可以看出記憶體使用量的整體過程**。
6. **實驗其他面向探討**
   1. **連結分析限制?**

HITS的演算法在**單一鏈結雙向時**，且**結點數量為奇數**時會無法收斂。

* 1. **實際應用上有甚麼問題?**

學生光是利用Sinrank的**演算法運算469個網頁結點就要運算20幾分鐘**，

然而在現實生活中得網站數目要等到全部運算完畢，使用者並沒有法辦等待這麼久的時間。因此學生發想可以利用設計演算法可以使用Map Reduce的方法透過較多的電腦分工來提升運算。(**因為本次學到的演算法必須要透過矩陣疊代一次一次算因此無法做一個比較有效率的Map Reduce的方法**。)

* 1. **對於連結分析有甚麼新的想法?**

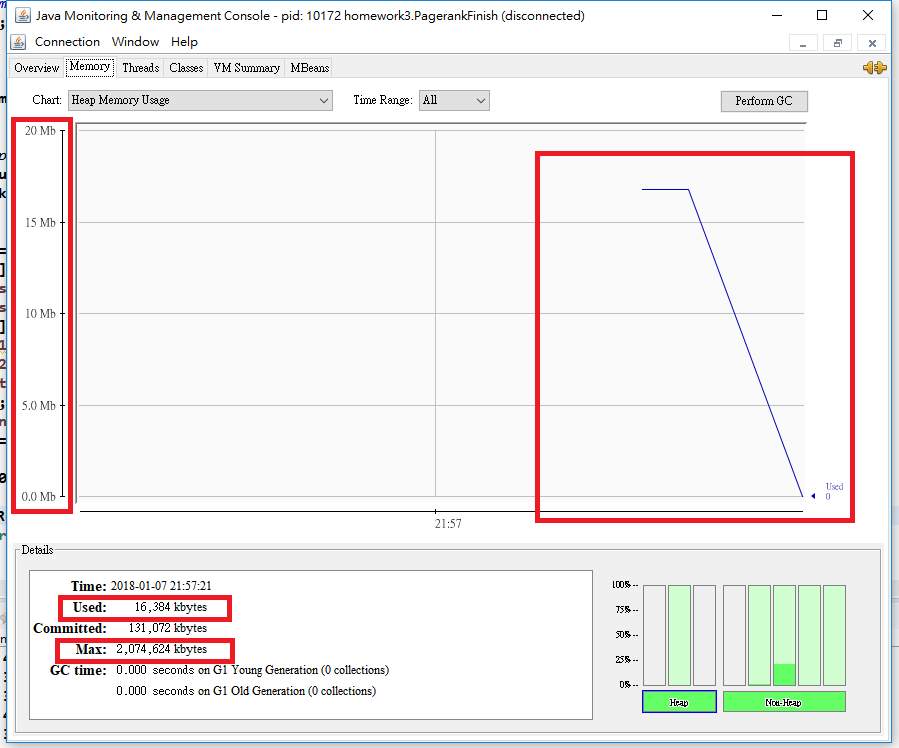
學生認為**網頁中可以加入時間的變數讓網頁價值隨著時間增加而遞減**，原因是因為如果有一個網站寫得比過去的網站還好，若用過去的演算法，新的網站很難去取代舊的網站，

* 1. **C對於Simrank的影響是甚麼?**

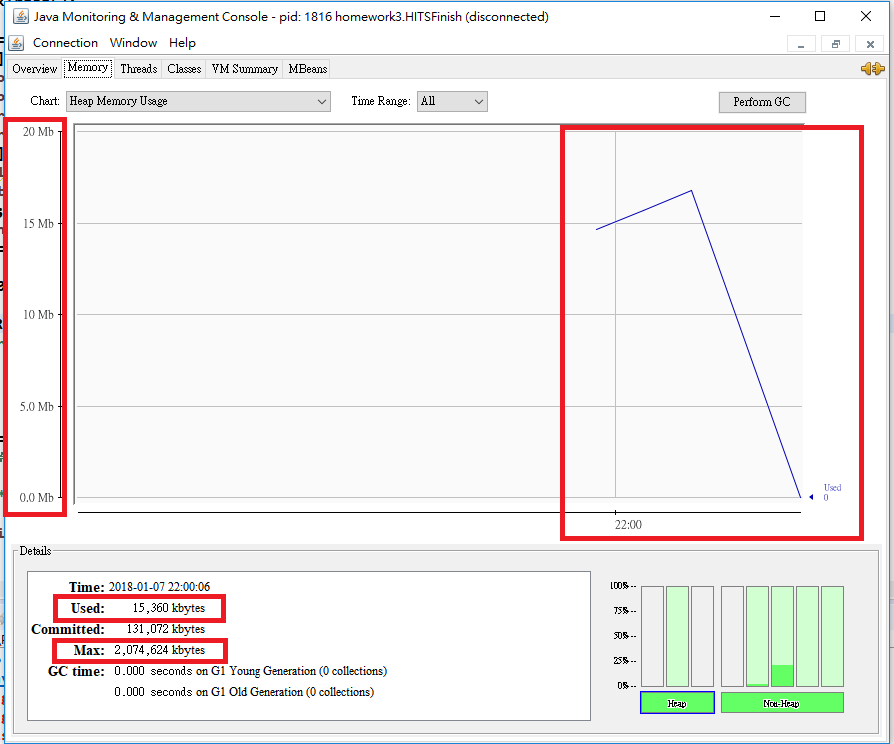
C一個介於0與1之間的係數，是用來修正現實生活與理論之間的誤差。由於我們在計算兩點之間的Similarity時，通常考量的問題皆是簡化後的模型，難免會有資訊不全的問題存在，因此**C的功用即用來減少兩點之間的相似程度**。且其中隨著疊代的次數增加相似度的值也會隨之減少，因此**C值的係數也像是一個衰退速度的設定值**。

1. **附上實驗結果圖片**

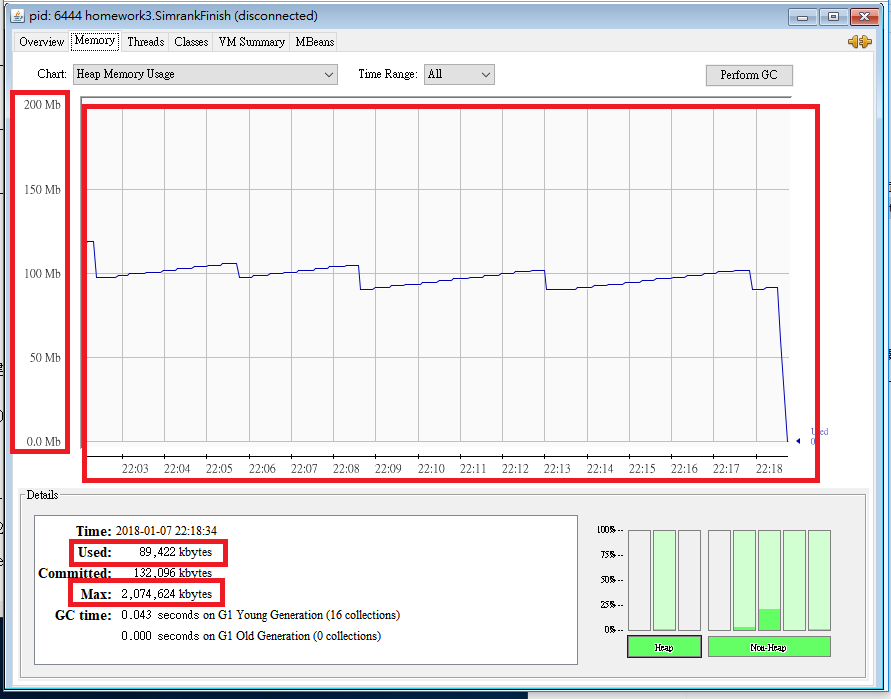
**圖五Pagerank**



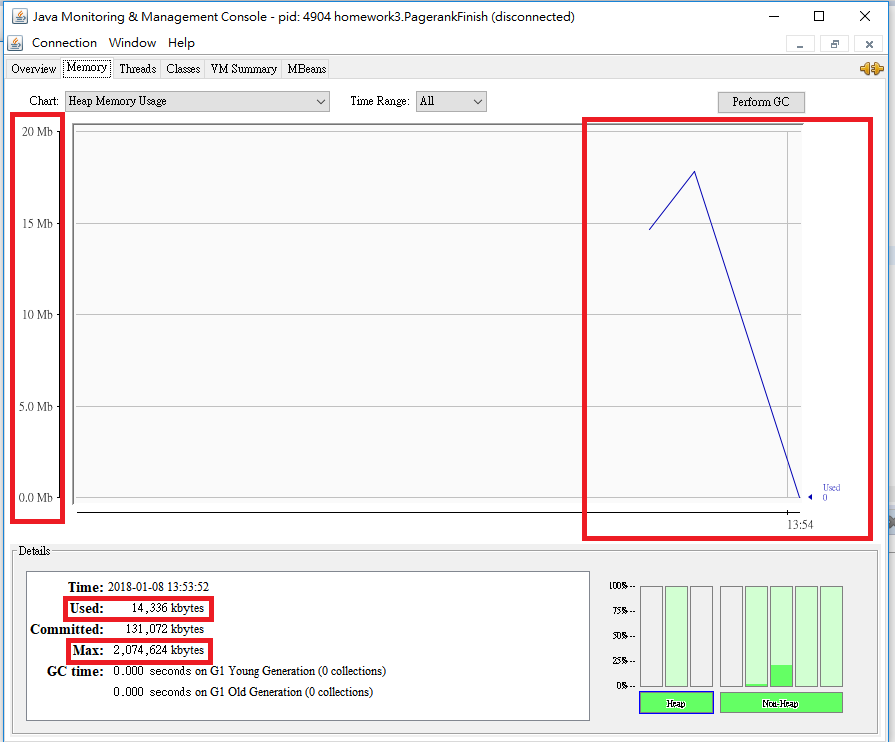
**圖五HITS**



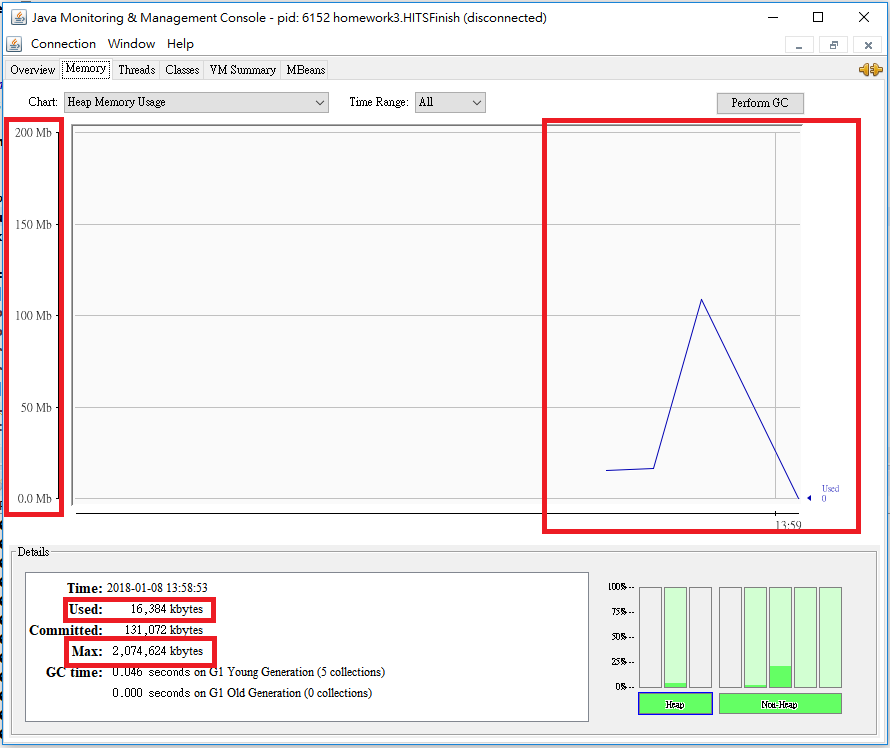
**圖五Simrank**



**圖六Pagerank**

****

**圖六HITS**

****