資管碩一 R02725005 林子勻

Web Retrieval and Mining

Programming Assignment 2 Report

一、 開發環境

- 程式語言：JAVA。
- 執行環境：
  (1) Ubuntu 12.04.4 LTS (GNU/Linux 3.8.0-38-generic x86_64)。
  (2) Java(TM)SE Runtime Environment (build 1.7.0_55-b13)。
  (3) Java HotSpot(TM) 64-Bit Server VM (build 24.55-b03, mixed mode)。

二、 程式執行

- 執行時間
  (1) cs.stanford.2004-10.graph：大約兩秒。
  (2) stanford-08-03.graph：大約十秒。
  (3) 03-2006-wk3.graph：大約三分鐘。
- 程式流程
  (1) 對.graph 做資料處理，轉換成所需的儲存方式。
  (2) 計算每一輪 page rank 的結果，直到上一輪跟下一輪的 Euclidean distance 小於所設的 threshold 才停止。
  (3) 將最後的分數結果輸出。

三、 前處理

- 處理.graph
  (1) 原本的資料為記錄每一個節點往外指向哪些節點，將其做轉換，改為記錄每一個節點有哪些節點往回指向他。因 graph 為 sparse graph，此部分不紀錄 out degree 為 0 的節點。

```
#maxnode 350004
1:76 7 10 14 16 18 20 22 24 26 28 30 33 35 37 39 41 43 47 48 49 50 51 52 54 56 57 58 59 60 62 63 66 67 68 70 72 74 75 76 77 79 85 86 87 88 89 92
93 94 95 96 97 98 99 100 101 102 103 104 105 106 107 108 109 110 111 112 113 114 115 116 117 118 119 120 125
2:1 16047
7:51 16 18 20 22 26 28 30 33 35 37 39 41 43 56 87 88 89 93 94 95 96 97 98 99 100 101 102 103 104 105 106 107 108 109 110 111 112 113 114 115 116
117 118 119 120 125 144 145 146 147 148
10:1 150
16:62 16 18 20 22 26 28 30 33 35 37 39 41 43 50 51 56 77 87 88 89 93 94 95 96 97 98 99 100 101 102 103 104 105 106 107 108 109 110 111 112 113 114
```

▲圖、原本的資料紀錄方式

  (2) 記錄每個節點總共擁有的 out-node 數目(out degree)，並將原本沒有任何 out-

node 的節點，將其往外指向所有節點，包括指向自己，因此該種類的節點所擁有的 out-node 數目為 graph 中的所有節點數。

(3) 紀錄那些節點原本 out degree 為 0。

- 資料結構

(1) 使用 HashSet，做動態新增節點資訊，並快速搜尋到所需的節點。

(2) 使用 HashMap，key 跟 value 對照到節點及那些節點指向它。

(3) 其餘單純儲存資訊，無特別需求的則使用 array 儲存。

四、 模型使用

- PageRank

利用下列公式計算第 K 輪中，每個節點 i 的分數。若節點 i 共有 j 個節點指向它，則 i 的分數為加總所有 j 對 i 的貢獻。

$$P_k(i) = (1-d) + d\sum_{(j,i)\in E}\frac{P_{k-1}(j)}{o_j}, \quad P_0(i) = 1$$

(1) 減少計算次數

假設 out degree 為 0 的節點數目為 N，轉換之後這些 N 個節點，指向 graph 中的所有節點，因此在計算每一節點 i 的分數時，都會計算到 N 對 i 的貢獻，而 N 又占了整體資料中的多數，因此為了減少計算次數以提升速度，再計算第 K 輪的所有節點分數之前，先計算所有 N 所加總起來的貢獻為 PN，計算 i 的分數時排除計算 N，之後再將 PN 加上。如此原本 PN 的計算為 graph 中所有的節點數，現在則降為 1 次。

(2) 第 K 輪的分數更新

計算 i 的分數時，使用的為上一輪的結果，因此計算完 i 之後，不能馬上將結果更新，可能會造成計算 i+1 時的錯誤。需計算完 graph 中的所有節點之後，再更新結果分數。