MDA_HW2

102062130 劉鳳軒

1. **Overview**

這次的 PageRank 總共用了四種 MapReduce，分別是

- FirstMapper_GenerateM
- FirstReducer_GenerateM
- SecondMapper_CalculateR
- SecondReducer_CalculateR
- ThirdMapper_CheckLeaked
- ThirdMapper_CheckLeaked
- FouthMapper_Output
- FouthReducer_Output

架構：

(1) 先用 FirstMapper_GenerateM 跟 FirstMapper_GenerateM 產生出 adjacency matrix

(2) 用 SecondMapper_CalculateR 跟 SecondReducer_CalculateR 算出 A 並算出每一個 node 的 pagerank

(3) 用 ThirdMapper_CheckLeaked 跟 ThirdReducer_CheckLeaked 檢查是否有 Dead ends 的可能(即所有 pagerank 加起來小於 1)，有的話就做 renomarlize

(4)重複 2、3 20 次

(5) 使用 FouthMapper_Output 跟 FouthReducer_Output 來處理前面的結果，並印出各 Node 與其 pageRank (另可以只取前 10 名)

而以下將會詳細說明

Word Definition

- fromNode: i,
- toNode: j,
- #Nodes: N
- pageRank: R,
- outDegree: D,

- All of the Nodes have edges to j: I…
- All of the Nodes have edges from i: J…)

TOP TEN:

1. 1056 6.3225947E-4
2. 1054 6.294921E-4
3. 1536 5.247023E-4
4. 171 5.123303E-4
5. 453 4.9617706E-4
6. 407 4.851387E-4
7. 263 4.798731E-4
8. 4664 4.7121226E-4
9. 261 4.632354E-4
10. 410 4.6138137E-4

檔案說明:

OutputData.txt: 所有 Node 的 pageRank

PageRank_10.java: sort 出前十名 page Rank 的 code

PageRank_all.java: 印出所有 pageRank 的 code

2. FirstMapper_GenerateM

Input: LongWritable, Text
Output: IntWritable, IntWritable

	Key		Value	
Input	Type	Format	Type	Format
	LongWritable		IntWritable	**i j**
Output	IntWritable	Format	IntWritable	Format

	Text	**i**	Text	**j**

首先先把測資吃進來，然後把 edge 分為 key 與 value 傳出

3. **FirstReducer_GenerateM**

Input: IntWritable, Iterable<Text>
Output: IntWritable, IntWritable

	Key		Value	
Input	Type	Format	Type	Format
	IntWritable	**i**	Text	**Iterable<j>**
Output	Type	Format	Type	Format
	IntWritable	**i**	Text	**Ri Di J…**

再來，將 Node i 的 pageRank(初始為 1/N) 、Outdegree(# of

outlinks) 還有所有向外有連到的 Node 都放到 value 中

4. SecondMapper_CalculateR

Input: IntWritable, Text
Output: IntWritable, Text

	Key		Value	
Input	Type	Format	Type	Format
	LongWritable		Text	**i Ri Di J…**
Output	Type	Format	Type	Format
	IntWritable	**j**	Text	**i Ri Di**

將每一個連到的 out Node 為 key,

以及自己 Node 的 pagerRank 及 OutDegree 放進 value

5. SecondReducer_CalculateR

Input: LongWritable, Text
Output: IntWritable, Text

	Key		Value	
Input	Type	Format	Type	Format
	IntWritable	**j**	Text	**Interable<** **i Ri Di>**
Output	Type	Format	Type	Format
	IntWritable	**j**	Text	**Rj Dj I...**

用所有 inlinks 進 toNode 的 資料，搭配公式:

$$r_j = \sum_{i \to j} \beta \frac{r_i}{d_i} + (1-\beta)\frac{1}{N}$$

即可算出 toNode 的 pageRank

最後將 toNode 為 key, 其 pageRank 及 Outdegree 還有所有

inlink 進來的 Nodes 作 value

另外，每次計算 pageRank 時都會同時 Update S

$$S = \sum_j r_j'^{new}$$

最後再 cleanup 時檢查是否小於 1 ，有的話代表出現 deadends

並在插入一行特殊指令:

Key, value = -1, (1-S)

6. **ThirdMapper_CheckLeaked**

Input: LongWritable, Text
Output: IntWritable, Text

	Key		Value	
Input	Type	Format	Type	Format
	LongWritable		Text	**j Rj Dj I...**
Output	Type	Format	Type	Format
	IntWritable	**i** **or** **j**	Text	**“TONODE ” + j** **or** **“INFO ” + Rj**

如果 input 的 j 進來發現是-1 ，就把(-1, (S-1)) 傳下去，

如果不是，則把 fromNode 作 key,

並在吃每一個 I…時，都會把(i, j)傳下去(前面用”TONODE” 作標示)，

而另外 toNode 也會把自己的 PagerRank 及 Outdegree 用(j, Rj)傳下去(用”INFO”作標示)

7. **ThirdReducer_CheckLeaked**

Input: IntWritable, Text
Output: IntWritable, Text

	Key		Value	
Input	Type	Format	Type	Format
	IntWritable	**i**	Text	**Iterable<**

				“TONODE ” + j> **or** **“INFO ” + Ri**
Output	Type	Format	Type	Format
	IntWritable	**i**	Text	**Ri Di J…**

由於 reducer 會把相同 key 的吃進來，所以 input 就同時擁有 fromNode, 他的 pageRank, outdegree, 以及所有 outlink nodes 將其整理出來並 output

8. **FouthMapper_Output**

Input: LongWritable, Text
Output: IntWritable, Text

	Key		Value	
Input	Type	Format	Type	Format
	LongWritable		Text	**i Ri Di J…**
Output	Type	Format	Type	Format
	IntWritable	**i**	Text	**Ri**

最後將 fromNode 及其 Ri 印出來

(由於要印前十名 pageRank，所以會先 maintain 一個大小 10 的 array,
永遠從大到小排序，一開始放進 10 個值後，如果未來有值大於其中最小的，便將其互換，保證最後只有前 10 名大的)

9. **FouthReducer_Output**

Input: IntWritable, Text
Output: IntWritable, Text

	Key		Value	
Input	Type	Format	Type	Format
	IntWritable	**i**	Text	**Iterable<Ri>**
Output	Type	Format	Type	Format
	IntWritable	**i**	Text	**Ri**

最後將 fromNode 及其 Ri 印出來