實作於 Hadoop 平台之網頁瀏覽樣式利潤偏好度探勘

賴鈺芬
靜宜大學資訊管理學系
g9971014@pu.edu.tw

葉介山*
靜宜大學資訊管理學系
jsyeh@pu.edu.tw

曹翔富
靜宜大學資訊管理學系
g1000597@pu.edu.tw

摘要

在過去幾年來，網際網路快速發展的情況下，資料探勘中的網頁探勘在電腦技術上越來越重要，要如何從大量的網頁中找出隱而未顯的使用者瀏覽樣式，成為網頁探勘的重要議題。本研究使用利潤探勘的原理去定義網頁的利潤值並且找出頻繁的網頁瀏覽樣式。

隨著資料量越來越大，如何提升演算法的執行效能也越受到關注，本研究則採用雲端 Hadoop 平台環境並以 MapReduce 技術嘗試解決網頁瀏覽樣式利潤偏好度探勘問題，透過探勘的結果，我們可以預測接下來被存取的網頁或是提供給網頁設計者去了解更多有意義的網頁資訊，使網頁架構可以更加完善。實驗結果顯示，本研究所採行的演算法，能產生更好的執行效能。

關鍵詞：網頁探勘、利潤探勘，雲端運算、MapReduce、資料探勘

壹、 緒論

在過去幾十年來網際網路快速發展的情況下，要如何從龐大的資料來源中找尋有用的資訊越來越受到關注。資料探勘裡的關聯法則探勘(Association Rule Mining)嘗試在大型資料庫領域中可以探討出有用的項目集(2005)，例如: Apriori，傳統方法去考慮項目與個數找出通過最小門檻值的項目集，然而以傳統的方法因為沒有去考慮到此項目的利潤，所以可能把高利潤的產品給修剪。因此 Yao 等學者提出利潤探勘(Utility Mining, 2004)，這個方法考慮利潤、個數以及產品，所以高利潤的產品將會被保留而不會被修剪。

而網站使用度探勘(Web Usage Mining)主要是藉由網頁記錄檔，並在有用的網頁中找出隱而未覺的瀏覽樣式，Zhou 等學者(2007)提出把利潤探勘演算法從商品的計算改變成網頁記錄檔的分析，也是分為兩階段利潤探勘演算法去找尋通過門檻值的高利潤路徑瀏覽模式。 Chen 和 Yeh 學者(2010)提出網頁瀏覽樣式之利潤偏好度探勘(Utility Preference Mining)，與傳統不同的做法是加入選擇偏好度與時間偏好度來定義網頁的利潤價值。透過探勘的結果，網頁伺服器可以預測接下來被存取的網頁，並且提供網頁設計師了解更多有意義的資訊。

此外，隨著資訊化的發展，許多領域都面臨著大規模的資料量，如今有效率的處理大量資料變成主要的議題。傳統作法在單機上處理大量的資料造成記憶體無法負荷，執行時間也越來越久，所以分散式系統受到越來越多的關注。分散式系統是使用許多低成本的電腦一起運算資料去進行探勘，雲端運算是分散式系統的一種，可以處理大量資料探勘，解決單一機器的效能問題。我們使用雲端環境去執行分散式運算，使實驗達到更好的效能。

本研究主要針對資料量越來越來大的環境下，並採用雲端 Hadoop 平台環境以及 MapReduce 技術嘗試解決網頁瀏覽樣式利潤偏好度探勘問題。實驗結果顯示，相較於單機模式的計算方式，本研究所採行的演算法，能產生更好的執行效能。

貳、 相關文獻

本小節首先介紹網頁探勘，並且介紹雲端運算的相關資訊，回顧利潤探勘的定義最後介紹網頁瀏覽樣式之利潤偏好度探勘的演算法以及範例。

一、網頁探勘

依探勘主題的不同，網頁探勘分為以下三類(2000)，分別為網站內容探勘(Web Content Mining)、網站結構探勘(Web Structure Mining)以及網站使用度探勘(Web Usage Mining)。

- 網站內容探勘

網站內容探勘主要著重於如何從網路資源上，尋找出有用且可存取使用的資訊。可以使用在搜尋引擎、推薦機制上能夠更有效的幫助使用者尋找想要的內容。

- 網站結構探勘

網站結構探勘重點在於辨認在各個網站間網頁的連接相關性，並且測量網頁間的連接權重，以便進行網頁的群集分析與分類分析。網站設計者可以用此探勘檢視網站的設計

架構。

● 網站使用度探勘

網站使用度探勘是針對網頁紀錄檔分析，由資料中發現使用者瀏覽特徵的流程。透過使用者瀏覽網頁所留下的資訊，分析出使用者的瀏覽模式，可以提供使用者個人的瀏覽模式，順便提高使用者的滿意度。

網頁紀錄檔案(web log)照時間順序排列紀錄所連線的資訊，有一個普通紀錄格式包含資料有回應的日期與時間、操作型態(GET、POST…等)、請求的狀態代碼以及操作系統和瀏覽器類型，如表 1 所示。

表 1 基本的網頁紀錄檔格式

125.230.6.188 - - [04/Jan/2009:04:02:30 +0800] "GET /templates/default/main_44_over.jpg HTTP/1.0" 500 - "http://www.pu.edu.tw/" "Mozilla/4.0 (compatible; MSIE 7.0; Windows NT 5.1; .NET CLR 1.1.4322; .NET CLR 2.0.50727)"

使用網頁紀錄檔的時候，有一些資料其實不是我們所要探勘的，例如：瀏覽器類型，為了避免資料量過大，我們要對資料作前處理，此步驟主要是針對所有的資料進行清理過濾的動作，把不必要的資訊可以先刪除過後，所探勘出來的資料才有意義。

一般來說網頁會包含很多的圖片，因此在讀取網頁時會一併讀取這些圖片並存在網頁紀錄檔裡，進行探勘的時候，重點是在分析網頁的瀏覽行為，因此要過濾網頁日記檔，把圖片格式的紀錄檔刪除。

經過資料清理的步驟後， 所留下的網頁紀錄檔就是我們想探勘的資訊。所以此步驟主要目的是把沒有用的資料刪除掉，提高資料探勘的速度及準確度。

二、 雲端運算

雲端運算其實是屬於「分散式運算」的其中一種方法，就是將我們的工作分給許多電腦分別去做運算，最後再將結果匯整起來。許多資料探勘的運算已經漸漸的使用雲端環境去做計算(Ekanayake et al. 2008；Huang et al. 2009；Zhao et al. 2009)。。雲端運算涵蓋了三種層次分別為 IaaS(Infrastructure as a Service)、PaaS(Platform as a Service)以及 SaaS(Software as a Service)。IaaS 層次是架設實體或是虛擬機器，例如：Amazon EC2；PaaS 層次建構一個雲端開發平台供大家去設計，例如： Google App Engine；SaaS 層次是一般人隨手可得的雲端服務，例如：Gmail。

Hadoop(2012)適合處理大量的資料，它的優點有可測量性、經濟性、有效性以及可靠性。它可以部署很多低規格的電腦組成一個群集。Hadoop API 可以依照使用者的需求去修改執行的參數，以便改善性能。

Hadoop 的架構分成兩個部分，一個是 HDFS(Hadoop distributed file system)(2012)，另一個是分散式程式模型 MapReduce(Dean and Ghemawat, 2004)。此架構必須透過網路去做運算，所以沒有網路的話，就無法使用此架構。

HDFS 會切割大型資料並存進好幾個區塊裡，為了改善容錯機制，HDFS 提供備份策略，以防資料遺失。MapReduce 是一種程式模型，來自兩個核心的運算分別是 Map 和 Reduce，執行程式的過程如圖 1，各執行步驟如下：

Step 1：首先，把我們所要使用的文件放進 HDFS 裡，他會依文件大小切成 M 的區塊。

Step 2：把這 M 個區塊分到不同的機器上去執行，每一個 Map 會經過計算產生 Key/Value(一對的值)，並且把算出來的結果放進 Reduce 去做運算。

Step 3：Reduce 將按照使用者定義的程式對 Key/Value 做排序或是加總，最後輸出在分散式檔案系統裡。

Step 4：當程式都執行完畢之後，才會跟使用者提示以執行完畢。

此外，MapReduce 的程式碼可以依照使用者的 Code 而產生出不同的結果。

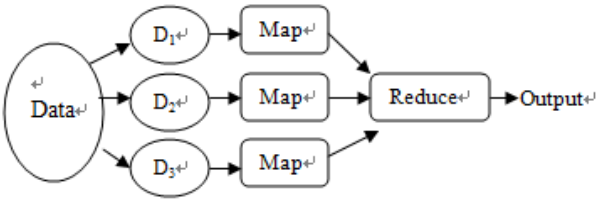

圖 1.MapReduce 程式模型

三、 利潤探勘

利潤探勘是由關聯法則探勘改進而出現的一種演算法。傳統的關連法則擁有向下閉合的概念例如：設門檻值為 40，{A,D}=42、{A}=20、{D}=80，此時{D}跟{A,D}過門檻值但是{A}沒過門檻值，因此 Liu 和 Liao 學者提出二階段演算法(2005)，改善了多餘的候選集以及執行時間，更能準確的找出高利潤項目集。

給定以下符號定義：

- $I = \{i_1, i_2, i_3 \cdots, i_m\}$有 m 個項目
- $D = \{T_1, T_2, T_3 \cdots T_n\}$交易資料庫裡每一筆交易$T_i \in D$
- $o(i_p, T_q)$在T_q交易裡項目i_p的數量
- $s(i_p)$在i_p項目的利潤值
- $u(i_p, T_q)$在T_q交易裡項目i_p的利潤值，也就是$o(i_p, T_q) \times s(i_p)$
- $u(X, T_q)$在T_q交易裡 X 項目集的利潤值，也就是$\sum_{i_p \in X} u(i_p, T_q)$，其中

  $X = \{i_1, i_2 \cdots, i_k\}, X \subseteq T_q$且$1 \leq k \leq m$
- $u(X)$指所有交易裡項目集 X 的利潤，也就是$\sum_{T_q \subseteq D \wedge X \subseteq T_q} u(X, T_q)$

利潤探勘的基本定義如下:

輸入：有 n 個交易的資料庫 D，m 個項目，每一個項目的利潤值以及門檻值 λ。

輸出：通過門檻值的項目集

利潤探勘可經由二階段的探勘方式完成，其二階段的相關定義如下:

階段一:

定義 1.在T_q交易裡所有項目的利潤加總。

$$tu(T_q) = \sum_{i_p \in T_q} u(i_p, T_q) \tag{1}$$

定義 2. 把項目集 *X* 有出現在T_q裡的交易利潤加總起來

$$twu(X) = \sum_{X \subseteq T_q \in D} tu(T_q) \tag{2}$$

定義 3. 當$twu(X) \geq \lambda$，項目集 X 屬於高利潤項目集

階段二:

把第一階段求出來項目集 *X* 的實際利潤算出來，假設有超過門檻值λ就是高利潤門檻值。

四、網頁瀏覽樣式偏好度演算法

此小節，將介紹 Chen 和 Yeh 學者在 2010 年提出網頁瀏覽樣式之利潤偏好度探勘(Utility Preference Mining)的演算法。該文章所採用符號與定義如下：

定義 1. $I=\{URL_1, URL_2, \cdots URL_n\}$，總共有 *n* 個網頁。

定義 2. $D=\{T_1, T_2, \cdots T_m\}$，總共有 *m* 個 *ip*。

定義 3. $o(URL_p, T_q)$在T_q裡連結URL_p的次數

定義 4. $s(URL_p)$在URL_p網址的權重利潤值，在此論文以此網頁點擊次數為權重值。

定義 5. $sp(URL_p, T_q)$，URL_p在T_q裡所拜訪網頁的總次數比例

$$sp(URL_p, T_q) = \frac{o(URL_p, T_q)}{\left(\sum_{URL_p \in T_q} o(URL_r, T_q)\right)} \tag{3}$$

定義 6. $tp(\mathrm{URL_p}, \mathrm{T_q})$在$T_q$裡所拜訪$URL_p$網頁的停留時間

$$tp(URL_p, T_q) = t(URL_p, T_q)/60 \tag{4}$$

定義 7. $pu(URL_p, T_q)$，URL_p在T_q裡的加權利潤

$$pu(URL_p, T_q) = s(URL_p) \times sp(URL_p, T_q) \times tp(URL_p, T_q) \tag{5}$$

定義 8. $pu(X, T_q)$ 在 T_q 交易裡有項目集 X 的利潤加總。其中 $X = \{URL_1, URL_2 \cdots, URL_k\}, X \subseteq T_q$ 且 $1 \leq k \leq n$

$$pu(X, T_q) = \sum_{URL_p \in I} pu(URL_p, T_q) \tag{6}$$

定義 9. $pu(X)$在所有資料庫 *D* 裡的項目 *X* 的利潤加總

$$pu(X) = \sum_{T_q \in D} \sum_{URL_p \in I} pu(URL_p, T_q) \quad (7)$$

定義 10．把IP_q有點選的網頁利潤加總起來

$$tpu(T_q) = \sum_{URL_p \in T_q} pu(URL_p, T_q) \quad (8)$$

定義 11. 設門檻值λ

$$minutil = \lambda \times \sum_{T_q \in D} tpu(T_q) \quad (9)$$

定義 12.$twu(X)$在所有資料庫 D 裡的有包含項目 X 的T_q利潤加總

$$tpu(X) = \sum_{X \subseteq T_q \in D} tpu(T_q) \quad (10)$$

網頁瀏覽樣式之利潤偏好度探勘則是由給定的網頁瀏覽記錄資料庫中，探勘超過設定的利潤門檻值的所有網頁集合。

五、 範例說明

在此章節我們提供一個簡單的例子來說明網頁瀏覽樣式之利潤偏好度探勘的演算法如何找出高利潤項目集。表 2 先定義資料庫 D，資料庫裡總共有 6 筆紀錄(TID)，5 個網頁以及每筆資料所點選網頁的總次數。表 3 顯示出每筆記錄在每個網頁所停留的時間。

首先計算URL_p在T_q裡所拜訪網頁的總次數比例稱作為 *sp*，例如:{a}在T_1中的比例為 1/(1+1+1+2)=1/5=0.2;並且把所停留的時間格式化稱作 *tp*，例如{a}在T_1所停留的時間為 1030/60=17.17。此步驟結束我們將彙整於表 4，以及每個網頁的 *pu* 值顯示在表 5。

表 2 資料庫資料

TID	點擊網頁	總次數
T_1	*a(1),b(1),c(1),d(2)*	5
T_2	*a(1),b(1),*	2
T_3	*a(1),e(1)*	2
T_4	*a(1),c(1),d(1)*	3
T_5	*a(1)*	1
T_6	*c(1),e(1)*	2

表 3 每筆紀錄所停留在各網頁的時間

TID	*a*	*b*	*c*	*d*	*e*
T_1	1030	316	360	530	0
T_2	430	660	0	0	0

T_3	1272	0	0	0	1369
T_4	510	0	694	401	0
T_5	1202	0	0	0	0
T_6	0	0	709	0	827

表 4 每筆紀錄的 *sp* 值及 *tp* 值

TID	Selection Preference (SP)	Time Preference (TP)
T_1	*a(0.2),b(0.2), c(0.2),d(0.4)*	*a(17.17),b(5.27), c(6),d(8.83)*
T_2	*a(0.5),b(0.5),*	*a(7.17),b(11),*
T_3	*a(0.5),e(0.5)*	*a(21.2),e(22.82)*
T_4	*a(0.33),c(0.33) ,d(0.33)*	*a(8.5),c(11.57), d(6.68)*
T_5	*a(1)*	*a(20.03)*
T_6	*c(0.5),e(0.5)*	*c(11.82),e(13.78)*

表 5 每筆紀錄的 *pu* 值以及加總的 *tpu* 值

TID	*pu*	*tpu*
T_1	*a(3.43),b(1.05),c(1.2),d(7.06)*	12.74
T_2	*a(3.59),b(5.5),*	9.09
T_3	*a(10.6),e(11.41)*	22.01
T_4	*a(2.81),c(3.82),d(2.2)*	8.83
T_5	*a(20.03)*	20.03
T_6	*c(5.91),e(6.89)*	12.8

再來先計算每個網頁的最大點擊數，例如: {*a*}網頁在所有 IP 裡面的最大點擊數 $=T_1+T_2+T_3+T_4+T_5=5+2+2+3+1=13$，再算所有資料庫 *D* 裡的有包含項目 *X* 的 T_q 利潤加總，例如：

$tpu(a)=tpu(T_1)+tpu(T_2)+tpu(T_3)+tpu(T_4)+tpu(T_5)=12.74+9.09+22.01+8.83+20.03=72.7$

。結果於表 6 呈現。

表 6 每個網頁的最大點擊數與 *tpu*

Page	最大點擊數	*tpu*
a	13	72.7
b	7	21.83
c	10	34.37
d	8	21.57
e	4	34.81

門檻值的設定把所有紀錄的 *tpu* 相加乘上我們的門檻值，例如λ=10%，則

$minutil = 0.1 \times [tpu(T_1) + tpu(T_2) + tpu(T_3) + tpu(T_4) + tpu(T_5) + tpu(T_6)] = 0.1 \times (12.74 + 9.09 + 22.01 + 8.83 + 20.03 + 12.8) = 8.55$

。再去觀察每個網頁是否有通過門檻值，求出第一階段的項目集。結果如表 7。

表 7 是否通過第一階段門檻值

Page	最大點擊數	*tpu*	是否通過門檻值
a	13	72.7	V
b	7	21.83	V
c	10	34.37	V
d	8	21.57	V
e	4	34.81	V

根據上面的計算可以得知第一階段通過的有*{a}*、*{b}*、*{c}*、*{d}*、*{e}*、*{a,c}*、*{a,d}*、*{a,e}*、*{c,d}*、*{c,e}*、*{a,c,d}*以上這些組合都有通過我們的門檻值$mintuil = 8.55$

最後把有通過第一階段的項目及原始利潤求出，並且找出超過門檻值的項目集，例如

$pu(a) = tpu(T_1) + tpu(T_2) + tpu(T_3) + tpu(T_4) + tpu(T_5) = 3.43 + 3.59 + 10.6 + 2.81 + 20.03 = 40.46$

，結果顯示在表 8，打勾的部分是有通過實際門檻值的網頁。

表 8 原始利潤是否有通過門檻值

Page	最大點擊數	實際利潤	是否通過門檻值
a	13	40.46	V
b	7	6.55	X
c	10	10.93	V
d	8	9.26	V
e	4	18.3	V

以及最終通過門檻值的所有項目集顯示在表 9。

表 9 通過門檻值的項目集

Page	實際利潤	Page	實際利潤
a	40.46	*a,d*	15.5
c	10.93	*a,e*	22.01
d	9.26	*c,d*	14.28
e	18.3	*c,e*	12.8
a,c	11.26	*a,c,d*	20.52

參、 提出架構

本研究針對 Chen 和 Yeh 學者所提出的網頁瀏覽樣式之利潤偏好度探勘模型，以 Hadoop 平台上的 MapReduce 技術來實作探勘演算方法，圖 2 顯示我們如何利用 MapReduce 去實作演算法的步驟說明，藍色框表顯示該步驟的運算是實作於 MapReduce 架構上，黑色框顯示該步驟的運算是實作於本機端，下面將附上步驟說明。

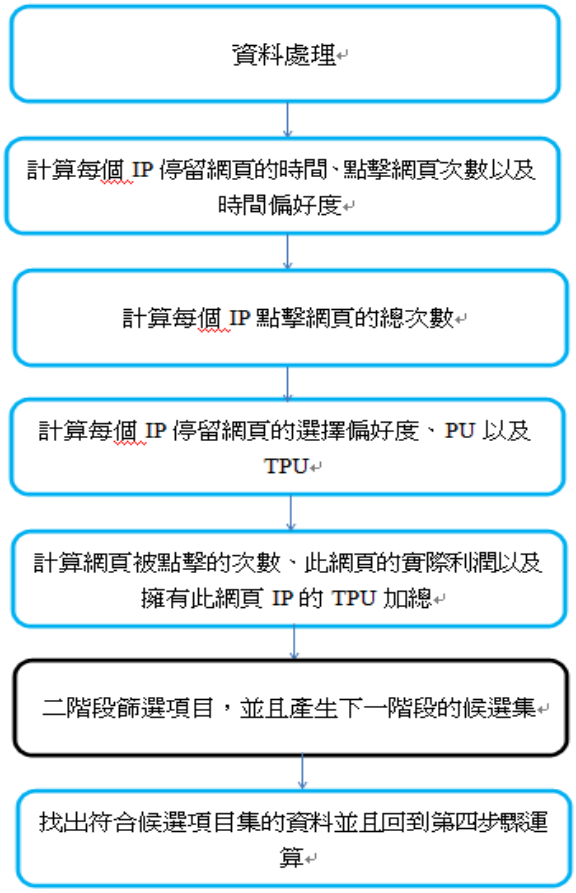

圖 2.MapReduce 架構步驟

Step 1. 先對我們的資料作前處理，篩選出符合我們的資料，如表 10。

表 10 Step 1 說明

Input：	Weblog
Map	日期,IP /網址,時間
Reduce	日期,IP /網址,時間, 網址,時間…

Map 跟 Reduce 是以 Key/Value 組成的，以 Map 為例，Key 為日期,IP、Value 為網址,時間，以下表格將以同樣方式呈現 Key/Value。

Step 2. 我們去計算每個 IP 的停留時間，並且算出時間偏好度(停留時間/60)，以及此 IP 所點擊網頁的總次數，如表 11。

表 11　Step 2 說明

Input	Step1 的結果
Map	IP,網址/1 IP,網址/網址,停留時間(SEC)
Reduce	IP,網址/停留時間(SEC),點擊次數,時間偏好度(分)

Step 3. 計算每個 IP 點擊網頁的總次數，如表 12。

表 12　Step3 說明

Input	Step2 的結果
Map	IP/點擊次數 IP /停留時間(SEC),點擊次數,時間偏好度(分)
Reduce	IP /網址,停留時間(SEC),點及次數,時間(分); IP 總點閱網頁次數

Step 4.　計算每個 IP 的網頁選擇偏好度(點擊網頁次數/此 IP 總共點擊網頁的次數)，以及計算 *pu*(選擇偏好度×時間偏好度×網頁權重值)，最後計算 *tpu*(此 IP 的所有 *pu* 的加總) ，如表 13。

表 13　Step4 說明

Input	Step3 的結果
Map	IP/網址,點擊次數,*sp*,*tp*,*pu*,IP 總點閱網頁次數
Reduce	IP /網址,點擊次數, *sp*,*tp*,*pu*,IP 總點閱網頁次數; IP 全部的 *pu* 加總(*tpu*)

Step 5.　計算網頁被所有 IP 點擊的次數，門檻值，擁有此網頁 IP 的 *tpu* 加總(第一階段)，以及此網頁的實際利潤(第二階段)，如表 14。

表 14　Step5 說明

Input	Step4 的結果
Map	-/每個 IP 的 *tpu* 網址/總點擊次數, 全部 IP 的利潤(*tpu*),實際利潤
Reduce	-/全部 IP 的利潤(*tpu*)加總 網址/擁有此網址的所有 IP *tpu* 加總, 此網址的所有 IP 點擊次數,以及此網頁的實際利潤

Step 6.　做二階段的篩選，找出通過門檻值的項目集。並且去生成下一階段的候選項目集，直到找不出下一階段候選項目集為止。

Step 7.　先去篩選符合項目集的紀錄，並且回到步驟 5 去做運算，如表 15。

表 15　Step7 說明

Input	候選項目集 Step4 的結果

Map	項目集/此網址有項目集的總點擊次數, 全部 IP 的利潤(*tpu*) 項目集/ 有此項目集 IP 的利潤(*tpu*)
Reduce	項目集/此網頁的實際利潤, 此網址的所有 IP 點擊次數,以及此項目集的所有 IP *tpu* 加總

肆、 實驗分析

以下為本實驗的環境：一為是 Hadoop 環境，總共有 8 台 VM，每一台 VM 有 1CPU 2.6GHz、1G Memory，分別以 2-node、4-node 以及 8-node 去執行。因為每一台 VM 的配額不大，所以無法分析大量的資料量，所以我們將以另一台單機跑網頁瀏覽樣式之利潤偏好度探勘去做為比較，單機的型號為 Acer Veriton M670、3G Memory、Intel Core 2 Quad CPU 2.83GHz。所有的程式碼以 JAVA 程式撰寫。

本實驗採用靜宜大學網站從 2009 年 6 月 10 日到 6 月 14 日的網頁記錄檔為數據，分成 1 天、3 天以及 5 天的實驗，其實驗數據顯是於表 16。第一步驟先做資料處理，只留下我們所需要的資料，再使用我們的架構去執行演算法算出超過門檻值的項目集。

表 16 實驗數據說明

	資料的大小 (Size)	原始資料總筆數 RD
0610	278MB	2971495
0610-0612	570MB	6081463
0610-0614	730MB	7787078

同樣都是三天的資料量，我們去觀察單機與 MapReduce 架構的執行時間，圖 3 顯示出使用環境所執行的時間比單機好 20 倍左右，證實使用 MapReduce 架構執行演算法可以提升我們的實驗效能。

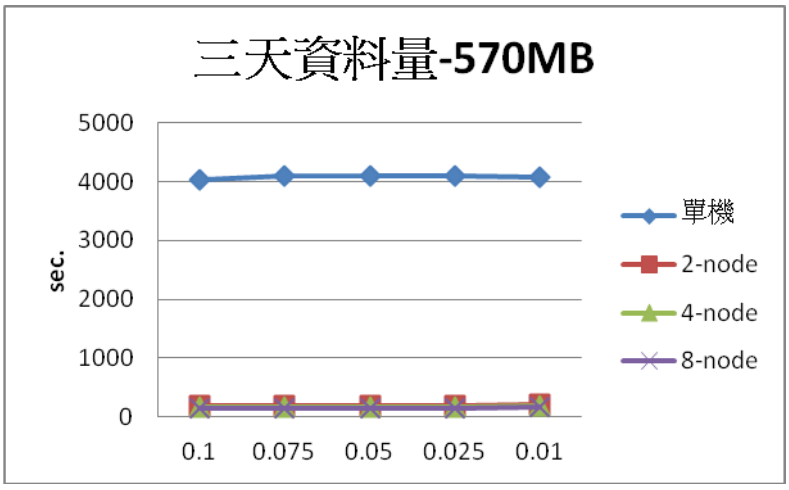

圖 3.三天資料量

圖 4 顯示當門檻值是 0.1 的時候，資料量以 1 天、3 天以及 5 天，我們的執行時間優於單機，原因是因為當資料量越大，單機的負荷量會越大，執行的行間相對會比較久，相反的我們運行在 MapReduce 上，所以對於資料量越大，執行時間跟單機比較起來差距會更大。

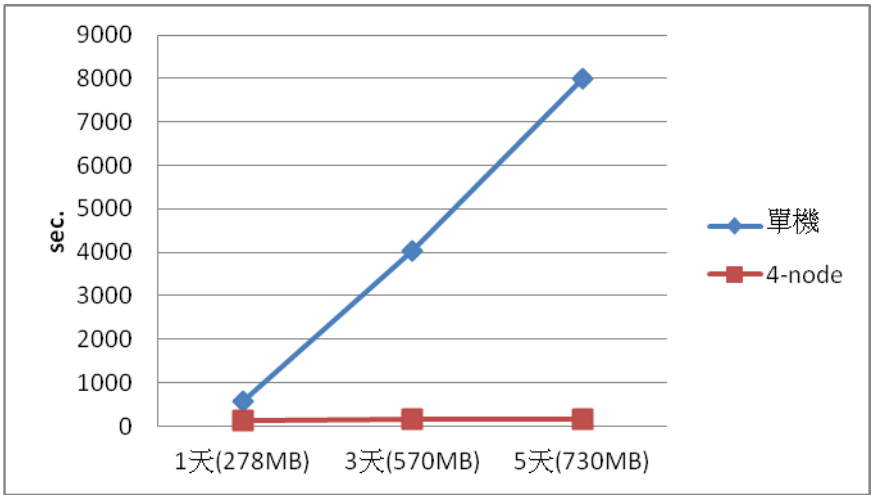

圖 4.門檻值為 0.1

表 17、表 18、表 19 分別呈現 1 天、3 天以及 5 天在不同 node 台數的情況下所執行的時間以及候選項目集，因為許多 IP 都點擊靜宜大學的首頁，所以首頁的利潤拉高了我們的實驗結果，所以最終通過門檻值的就只有首頁。

表 17 一天的結果分析

一天	階段一	階段二	2 台	4 台	8 台
門檻			秒數	秒數	秒數
0.1	34	1	142.77	139.84	126.40
0.075	34	1	143.07	140.05	126.58
0.05	36	1	142.92	139.84	126.40
0.025	44	1	147.03	139.98	126.57
0.01	45	3	168.44	163.41	150.99

表 18 三天的結果分析

三天	階段一	階段二	2 台	4 台	8 台
門檻			秒數	秒數	秒數
0.1	63	1	162.81	154.04	161.29
0.075	66	1	162.99	154.11	161.76
0.05	66	1	162.81	153.95	161.15
0.025	67	1	163.01	154.01	161.29
0.01	68	3	187.38	178.36	186.64

表 19 三天的結果分析

五天	階段一	階段二	2 台	4 台	8 台
門檻			秒數	秒數	秒數
0.1	67	1	189.67	167.65	152.95
0.075	70	1	189.80	167.85	153.11
0.05	73	1	189.60	167.57	152.94
0.025	74	1	189.79	167.78	153.08
0.01	75	3	213.17	191.19	177.33

表 20 將顯示三天的資料量運行的單機的執行時間，數據顯示執行的時間還比 MapReduce 的 5 天時間還要久，並且單機的設備還比 VM 來的好，在這樣的情況下，更能證明 MapReduce 的性能大大超過單機的性能。

表 20 三天的單機執行時間

門檻值	秒數
0.1	4026.99
0.075	4101.2
0.05	4107.73
0.025	4099.88
0.01	4086.84

伍、 結論

這幾十年來隨著網際網路的快速發展，要如何在龐大資料量快速找出潛在或有用的資訊，我們試著把偏好度利潤探勘與 MapReduce 做結合，在我們的實驗數據看來，執行效果確實是比單機來的快速許多，證實雲端運算也可以有效的運用在資料探勘上面。

未來研究方向可以朝著調整網頁利潤探勘的重要性，以網頁架構給予每個網頁一個利潤值而不是以點擊次數去做為網頁利潤。此外，目前的實驗資料還不夠大量，在資料處理過後所產生的檔案小於 64MB，在不同 node 運行的情況下差異沒有很大(照理資料量越大，node 數越多運行的結果會比較好)，所以未來可以以更大量的數據去做運算所以未來可以以更大量的數據去做運算，以更加凸顯 Hadoop 的特性。

陸、 致謝

本研究受行政院國家科學委員會補助 (NSC100-2221-E-126-016 及 NSC99-2632-E-126-001-MY3)，特此致謝。

參考文獻

1. "Crawl the Nutch-MapReduce," http://blogger.org.cn/blog, 2012.
2. D. Borthakur, "The Hadoop distributed file system: architecture and design," http://lucene.apache.org/hadoop/hdfs.html, 2012.
3. Hadoop, http://hadoop.apache.org, 2012.
4. L. Huang, X. W. Wang, Y. D. Zhai, and B. Yang,, "Extraction of User Profile Based on the Hadoop Framework," *Proceedings of the 5th International Conference on Wireless Communications, Networking and Mobile Computing (WiCOM 2009)*, Beijing, China, September(2009)
5. H. Yao, H. J. Hamilton, and C. J. Butz, "A foundational approach to mining itemset utilities from databases," *Proceedings of the 3rd SIAM International Conference on Data Mining*, pp. 482-486 (2004)
6. J. Dean and S. Ghemawat, "MapReduce: simplified data processing on large clusters," OSDI04: Sixth Symposium on Operating System Design and Implemention, San Francisco, CA, December(2004)
7. J. Ekanayake , S. Pallickara, and G. Fox "MapReduce for data intensive scientific analyses," *Proceedings of the 4th IEEE International Conference on eScience*, p.277-284, December 07-12(2008)
8. L. Zhou, Y. Liu, J. Wang, and Y. Shi, "Utility-based web path traversal pattern mining," *Proceedings of the 7th IEEE International conference on Data Mining Workshops*, pp. 373–378 (2007)
9. R. Kosala and H. Blockeel, "Web mining research: a survey," *ACM SIGKDD Explorations Newsletter*, v.2 n.1, p.1-15, June(2000)
10. S. S. Tseng, "Data mining," ISBN 10-9-574-42236-4, Flag Publishing Co., Taipei, (2005)
11. W. Zhao, H. Ma, and Q. He, " Parallel K-means clustering based on MapReduce," *In Proceedings of the 1st International Conference on Cloud Computing (CloudCom)*, pp. 674-679 (2009)
12. Y. C. Chen and J. S. Yeh, "Preference utility mining of web navigation patterns," *IET International Conference on Frontier Computing. Theory, Technologies & Applications (CP568)* Taichung, Taiwan, pp.49-54 (2010)
13. Y. Liu, W. K. Liao, and A. Choudhary, "A fast high utility itemsets mining algorithm," *Proceedings of the 1st International Conference on Utility-Based Data Mining*, pp. 90-99 (2005)

Implementing Utility Preference Mining of Web Navigation Patterns On Hadoop Platform

LAI Yu-Fen
Providence University
g9971014@pu.edu.tw

YEH Jieh-Shan
Providence University
jsyeh@pu.edu.tw

TSAU Shiang-Fu
Providence University
g1000597@pu.edu.tw

Abstract

Due to the rapid development of World Wide Web in the past decades, web mining becomes a vital technology in data mining. To efficiently discover implicit but important user navigation patterns is one of important issues in web mining. The study first reviewed the utility preference mining of web navigation patterns proposed by Chen and Yeh. With the increasing amount of web log data, how to improve the algorithm performance attracts the more attention. This study adopted the cloud computing platform, Hadoop, and implemented MapReduce methodology for utility preference mining of web navigation patterns. The experimental results show that the proposed method is more efficient and scalable than a single machine platform.

Keywords: Data mining, Utility mining, Cloud computing, MapReduce, Web mining.