一個有效率的漸進式網路交易型樣探勘演算法

顏秀珍	李御璽	黃柏文	吳政瑋
銘傳大學資工系	銘傳大學資工系	銘傳大學資工系	銘傳大學資工系
sjyen@mcu.edu.tw	leeys@mcu.edu.tw	S4450060@ss24.mcu.edu.tw	silvemoonfox@hotmail.com

摘要

網路資料探勘（web mining）是將資料探勘（data mining）運用在大量的網路資料上，以改進網路上的資訊服務。而網路瀏覽型樣探勘（web traversal pattern mining）是從使用者瀏覽網頁的記錄中找出大多數使用者的瀏覽行為，但無法得知使用者在瀏覽網頁過程中的購物行為。網路交易型樣探勘（web transaction pattern mining）是從網路活動的記錄中找出大多數使用者瀏覽與購物的行為。然而隨著時間的推進，瀏覽與購物的記錄會不斷增加，過久的記錄也必需移除。所以在資料更新後必須重新探勘，以得到最新的資訊，但重複探勘會造成時間上的浪費，因此漸進式資料探勘（incremental data mining）近年來廣受各方關注。以往已有一些研究提出漸進式的網頁瀏覽型樣探勘，然而這些研究並沒有考慮到交易的行為。

本論文提出有效率的網路交易型樣探勘 WTPM (Web Transaction Pattern Mining) 與漸進式網路交易型樣探勘 IWTPM（Incremental Web Transaction Pattern Mining）演算法，當使用者瀏覽與交易的記錄增加或被移除時，可以利用之前網路交易型樣探勘後所留下來的資訊，不必重新掃描瀏覽與交易的資料，就可找出資料更新後的網路交易型樣。

關鍵詞：網路資料探勘、漸近式資料探勘、瀏覽交易序列、網路瀏覽型樣、網路交易型樣。

一、導論

由於網際網路的快速普及，使得網路資料急速不斷的增加；如何從龐大的網路資料中正確且快速地找出有用的資訊，就成為了一個重要的研究課題。網路資料探勘（web mining）就是要從大量的網路資料中正確且快速地找出有用的資訊，以改善網路的服務品質。網路瀏覽型樣探勘及網路交易型樣探勘是網路資料探勘中重要的技術；其中網路瀏覽型樣探勘是在顧客瀏覽記錄資料庫中，找出被頻繁瀏覽的網頁路徑，可以用來改善網站設計，例如在高度關聯的網頁之間提供有效率的連結。而網路交易型樣探勘主要是在網路交易資料庫中，找出被頻繁瀏覽的網頁路徑中頻繁被購買的商品組合，除了可以改善網站設計之外，也可改善銷售的效能，例如把商品廣告放在適當的網頁路徑，以及購買商品的路徑推薦，以提供給顧客更多的資訊服務。

以下我們說明網路交易型樣的相關定義。$W = \{w_1, w_2, \ldots, w_n\}$ 是網站中全部網頁的集合。$I = \{i_1, i_2, \ldots, i_m\}$，是項目（items）或商品的集合（itemset）。一個*顧客瀏覽交易序列資料庫*(如表 1)中的每一筆記錄，包含瀏覽編號*(Traversal Identifier（TID）)*與*顧客瀏覽交易序列*，是顧客一次瀏覽與購買的行為。例如：在表 1 中，TID為 1 的

顧客瀏覽交易序列為在依序瀏覽網頁D, A, F, G, B和C的過程中，顧客在網頁F 購買了商品 3, 而在網頁B購買了商品 2, 5 和 7。而<DAFGBC>為*顧客瀏覽序列(traversal sequence)*。

表 1 原始顧客瀏覽交易序列資料庫

TID	顧客瀏覽交易序列
1	<DAF{3}GB{2,5,7}C>
2	<DF{3,4}GB{2,5}CKJ{11}>
3	<AE{9}GIC{1,6,8}H>
4	<AFE{9,10}IC{1,6,8}KJ>
5	<F{4}GIC{6,8}KJ{11}>
6	<AE{9,12}GIC{1,6,8}KH>
7	<DF{3,4}GB{5,7}CK>

因此，一個*瀏覽交易序列(traversal transaction sequence)*可表示為$<w_1\{i_1\} w_2\{i_2\}\ldots w_q\{i_q\}>$, $w_j \in W$, $i_j \subset I$ $(1 \le j \le q)$。對兩個*瀏覽交易序列*$q = <s_1\{i_1\}s_2\{i_2\}\ldots s_n\{i_n\}>$和$p = <r_1\{j_1\} r_2\{j_2\}\ldots r_m\{j_m\}>$，如果存在著$k_1 < k_2 < \ldots < k_n$，使得$s_1 = r_{k1}, \ldots,$和$s_n = r_{kn}$並且$i_1 \subseteq j_{k1}, \ldots, i_n \subseteq j_{kn}$，則稱瀏覽交易序列p包含瀏覽交易序列q。一個瀏覽交易序列的*支持度*為包含此瀏覽交易序列的顧客瀏覽交易序列數與資料庫中總顧客瀏覽交易序列數的比值。而一個瀏覽交易序列的*支持度計數*為包含此瀏覽交易序列的顧客瀏覽交易序列數。若瀏覽交易序列的支持度不小於使用者所指定的*最小支持度(minimum support)*門檻值，則此瀏覽交易序列稱為*網路交易型樣*(web transaction pattern)。例如在表 1 中，瀏覽交易序列<C{6,8}K>的支持度為 3/7 = 42.86%。若最小支持度設為 40%，則<C{6,8}K>是一個網路交易型樣。

過去對網路交易型樣探勘的研究[10, 15]，若是新增資料或移除資料，就必須掃描整個資料庫，重新探勘，所以非常浪費時間。漸進式資料探勘的主要精神就是希望利用先前探勘所保留下來的資訊，只需針對新增或移除的資料來做探勘，可以減少重複探勘所需的時間。因此，漸進式資料探勘會比從頭到尾去探勘整個資料庫要來得有效率，但缺點就是需要額外的空間去儲存先前探勘所保留下來的資訊，要如何達到省時又省空間的目標，就是這篇論文的挑戰。

二、相關研究

路徑瀏覽型樣（Path traversal pattern）[1, 3, 11, 13, 14]是一個用來找出大多數使用者之網路瀏覽行為的技術，網站設計者可以透過這些資訊去改進網站設計，增加網站的效能。此外，這些資訊可以提供瀏覽建議給使用者。這方面的研究有 FS(Full scan）演算法，SS（Selective Scan）演算法[3]，MAFTP 演算法[17]，和其他演算法[1, 2, 5, 6, 11, 13, 14]。不過這些演算法有一個限制，就是他們只能發現簡單瀏覽型樣（Simple Path Traversal Patterns），即此型樣沒有重複的網頁，他們都只考慮向前瀏覽的行為（forward reference），沒有考慮向後瀏覽的行為（backward reference）。FS 及 SS 主要分為兩個部分： 先利用最大向前參考序列 MFR(Maximal Forward reference)演算法把資料由 log data 轉換成最大向前參考序列的資料庫，再運用 FS 或 SS 演算法來發掘出所有長度的頻繁瀏覽序列。所謂最大向前參考是指使用者一直向前瀏覽網頁的動作時，發生有回頭瀏覽(backward reference)的行為，即切斷剛剛向前瀏覽的路線。

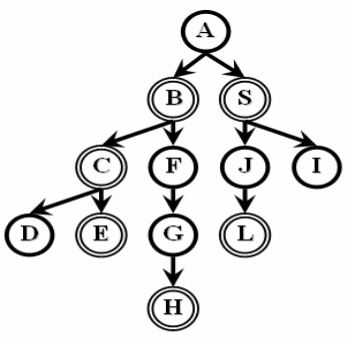
FS 和 SS 演算法是一個多次掃描資料庫的演算法，第一次產生 1-頻繁序列

集，第二次產生 2-頻繁序列集，第 k 次產生 k-頻繁序列集，以此類推。SS 是類似 FS 演算法，不同於 FS 演算法是 SS 演算法它並不是每次都掃描資料庫，而是利用之前的候選序列集直接產生下一個長度的候選序列集，再一起掃描資料庫，進而減少 disk I/O 的時間，而 FS 及 SS 演算法的缺點是經過 MFR 演算法切割後，可能會產生一些隱藏資訊的遺失。

另一個相關於挖掘路徑瀏覽樣式(Mining path traversal patterns)的研究是 J. Han [7]所提出的 WAP mining 演算法，完全不同於其他的演算法是因為它並不產生候選序列集，而是運用 WAP-tree 資料結構。WAP-tree 演算法的優點在於它不產生候選序列集，所以它可以避開候選序列集暴增的情形，另外，它只需掃描資料庫兩次，因此執行速度較快。但其缺點為需要很大的記憶空間，因為該演算法等於把資料庫中的所有交易全部轉換 WAP-tree 結構而存放在記憶體中，特別是當資料庫中瀏覽序列重複出現的情形很低時，所需要的記憶體更是難以估算，使得記憶體的負荷量大增。

網路瀏覽型樣(web traversal patterns)探勘[16]是要挖掘非簡單瀏覽型樣（Non-Simple Path Traversal Patterns）。非簡單瀏覽型樣不只包含向前瀏覽，也包含向後瀏覽的行為，它可以較準確的預測顧客的行為。然而，網站管理者不只要考慮網頁瀏覽的行為，也必須考慮顧客的購買行為。因此，路徑瀏覽型樣探勘無法滿足網站管理者的需求。

在 2000 年 Yun and Chen 首先提出網路交易型樣探勘演算法 MTS[15]，同時考慮路徑瀏覽型樣和顧客購買行為。MTS 演算法首先要將原始的顧客瀏覽與交易記錄轉換成只有向前瀏覽的行為。我們用一個例子說明轉換的方法。圖 1 是一個顧客的網路瀏覽與交易記錄，顧客瀏覽網頁 A 之後，再瀏覽網頁 B，並且在網頁 B 購買商品 1。接著瀏覽網頁 C，然後到網頁 D。之後，顧客回到網頁 C，然後再瀏覽網頁 E，並且在網頁 E 購買商品 3 和 4，再回到網頁 C 購買商品 2，依此類推。



Web-Transaction	
Web Page	Items
A	---
B	1
C	---
D	---
C	---
E	3, 4
C	2
B	---
F	---
G	---
H	6
A	---
S	7, 8
J	---
L	9
S	---
I	---

圖 1: 一個顧客的網路瀏覽與交易記錄

MTS 演算法對每一個顧客的網路瀏覽與交易記錄做切割，當顧客有向後瀏覽的行為，網頁交易紀錄就會被切割出來。例如表 2 是圖 1 的網路瀏覽與交易記錄被切割出來的結果。因為顧客在網頁 D 和網頁 I 沒有購買任何商品，所以路徑 ABCD 和 ASI 不會被切割出來。

表 2: 網頁交易記錄

路徑	網頁交易記錄
ABCE	$B\{1\}, C\{2\}, E\{3\}, E\{4\}$
ABFGH	$B\{1\}, H\{6\}$
ASJL	$S\{7\}, S\{8\}, L\{9\}$

然而這樣的切割會有資訊遺失以及產生錯誤資訊的問題。從圖 1 中可以看出顧客瀏覽到網頁 E 並購買商品 3 和 4，然後回到網頁 C 購買商品 2。但是從切割後的網頁交易記錄(表 2)中，路徑 *ABCE* 的網頁

交易紀錄顯示顧客從網頁 A 瀏覽到網頁 B，再從網頁 B 瀏覽到網頁 C 時購買商品 2，然後到網頁 E 購買商品 3 和 4。這樣的記錄顯然與原始的瀏覽行為不同，不但遺失了原貌也會產生錯誤的結果。另外這種切割方式會造成一種網頁交易記錄，只會對應到唯一的一條路徑，若有兩條以上的路徑有相同的網頁交易記錄，則對於此網頁交易記錄，此方法只會記錄一條路徑，造成原始瀏覽行為的遺失。

Yen and Lee 提出 *IWA* 演算法[8, 10]，不需對原始的顧客瀏覽與交易記錄做切割，同時考慮顧客向前瀏覽與向後瀏覽的行為，因此不會有資訊遺失以及產生錯誤資訊的問題。*IWA* 演算法也利用網站架構，可找出在網站架構中有連結的網路交易型樣，並且允許顧客瀏覽與交易記錄中有雜訊的存在，也就是顧客在瀏覽網頁的過程中有些網頁可能不是他真正想瀏覽的網頁，*IWA* 演算法仍可找出正確的網路交易型樣。但是當有新的顧客瀏覽與交易記錄加入，網路交易型樣的探勘演算法必需重新掃描原始的顧客瀏覽與交易記錄，重新探勘。

漸進式資料探勘就是考慮當有資料新增或被刪除時，如何從新增或移除的資料以及舊有的資訊中挖掘出新的資訊。漸進式的序列型樣探勘研究有由 Zaki 等人提出的 ISL 演算法[12]與 J. Han 等人提出的 IncSpan 演算法[4]，而漸進式的網路瀏覽型樣探勘研究有由 Lee and Yen 提出的 IncWTP 演算法[9]。

ISL 演算法[12]利用 lattice 結構，儲存所有的候選序列及其支持度計數，如圖 2 所示。其中陰影部份是低於最小支持度的候選序列（candidate sequences），而其它節點則為頻繁序列。當有顧客購買序列新增時，ISL 演算法首先從新增的序列中,找出長度為 1 之序列的支持度計數，與原先在 lattice 結構中的支持度計數合併後，即可決定新增資料後長度為 1 的頻繁序列。再將長度為 1 的頻繁序列組成長度為 2 的後選序列，只需更新在 lattice 結構中有變動的部份，依此類推，即可找出所有新增資料後的頻繁序列。由於 ISL 需要儲存所有候選序列及其支持度，甚至連支持度計數為 0 的候選序列都要儲存，還要維護整個 lattice 結構，因此必須花費非常大量的儲存空間。

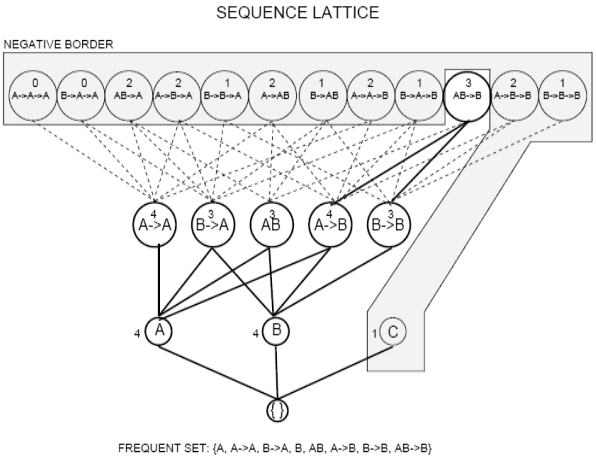


圖 2: lattice 結構 for ISL

IncSpan 演算法[4]是利用遞迴的方式產生每一頻繁序列的投射資料庫(Projected database)。一個頻繁序列的投射資料庫為顧客序列資料庫中與此頻繁序列相關的序列所成的集合。然而 IncSpan 會有投射資料庫過大且過多的問題，導致執行時間與記憶體使用效率不佳。

IncWTP 演算法[14]也是利用 Lattice 結構儲存網路瀏覽型樣與候選瀏覽型樣，以及每一型樣出現在哪幾筆顧客瀏覽序列中，不同於 ISL 演算法的是 IncWTP 在產生候選瀏覽型樣時必須考慮網站架構，而且 IncWTP 不儲存不存在於資料庫中的候選瀏覽型樣，節省了不少記憶體空間，另

外 IncWTP 不但考慮資料新增的情形，也考慮了資料被移除的情況。然而，儲存所有的網路瀏覽型樣與支持度不為 0 的候選瀏覽型樣於 lattice 結構中，仍然會佔用大量的記憶體空間。而維護 lattice 結構也必須花費不少的執行時間。而這些有關漸近式資料探勘的研究，都沒有考慮顧客交易的行為。

三、網路交易型樣探勘

以下我們介紹我們的演算法 WTPM，並以表 1 為例，來解釋演算法的執行過程。為了執行的效率與方便起見，首先我們將原始資料庫(表 1)轉換成如表 3 的瀏覽交易序列資料庫。在原始資料庫的瀏覽交易序列中，若某一網頁 X 有購買項目 1, 2, …, i，也就是存在有<X{1, 2, …, i}>，則會被轉換為<[XX{1}X{2}…X{i}]>。以下我們將瀏覽交易序列簡稱為序列。一序列的長度是此序列所包含之網頁或網頁結合一購買項目的個數。一個長度為 k 的序列稱為 k-序列。例如在表 3 中，TID 為 1 的序列之長度為 10，為一 10-序列。一個網頁(或網頁結合一購買項目)在顧客瀏覽交易序列中的位置為 P，表示此網頁在此顧客瀏覽交易序列中的第 P 個網頁。例如在表 3 中，TID 為 1 的顧客瀏覽交易序列的第 3 個位置為 F，。第 4 個位置為 F{3}，而第 9 個位置為 B{7}。

轉換後的顧客瀏覽交易序列中，中括號內表示在某一網頁同時購買的商品，例如：<[BB{2}B{5}B{7}]>表示在網頁 B 同時購買商品 2，5 和 7。而大括號內表示在某一網頁所購買的某一商品，例如：B{2}表示在網頁 B 有購買商品 2。在資料庫轉換的過程中，WTPM 計算每一網頁的支持度，支持度不小於最小支持度的網頁，稱為頻繁網頁或頻繁 1-序列；支持度小於最小支持度的網頁，為非頻繁網頁。例如在表 1 中，我們將最小支持度設為 50%。在掃描一次原始資料庫後，表 1 可被轉換成表 3，並且產生頻繁網頁 A, B, C, D, E, F, G, I 和 K 與非頻繁網頁 H 和 J。

表 3: 顧客瀏覽交易序列資料庫

TID	顧客瀏覽交易序列
1	<D A [FF{3}] G [B B{2}B{5} B{7}] C>
2	<D [FF{3}F{4}] G [B B{2}B{5}] C K [JJ{11}]>
3	<A [EE{9}] G I [CC{1}C{6}C{8}]H>
4	<A F [EE{9}E{10}] I [CC{1}C{6}C{8}] K J>
5	<A [EE{9} E{12}] G I [CC{1}C{6}C{8}] KH>
6	<D [FF{3}F{4}] G [BB{5}B{7}] C K>

WTPM 接下來掃描如表 3 的瀏覽交易序列資料庫，建立頻繁與非頻繁網頁的條件資料庫(conditional databases)，並刪除瀏覽交易序列資料庫中的非頻繁網頁。對於某一個網頁 x，若顧客瀏覽交易序列中存在有 x，則將此顧客瀏覽交易序列的 TID 和第一次出現 x 的下一個網頁(或網頁結合一購買項目)的位置，輸出到 x 的條件資料庫中。在建立頻繁網頁之條件資料庫的過程中，WTPM 也順便計數與記錄所輸出之位置的網頁(或網頁結合一購買項目)的支持度。若 x 出現在中括號內，則要計數 x 之後中括號之內的所有網頁結合一購買項目的支持度，以及中括號外的第一個網頁的支持度。若所計數之網頁(或網頁結合一購買項目) y 的支持度有達到最小支持度，則表示 xy 為頻繁 2-序列。因此掃描完顧客瀏覽交易序列資料庫後，所有頻繁

網頁的條件資料庫與頻繁 2-序列就可產生。

例如在顧客瀏覽交易序列資料庫中，頻繁網頁 B 有出現在 TID 1, TID 2 和 TID 6。在 TID 1 的顧客瀏覽交易序列中，第一個 B 出現在中括號內，因此輸出下一個網頁結合一購買項目的位置，也就是 B{2} 的位置 7。也順便計數中括號之內的所有網頁結合一購買項目 B{2}, B{5}, B{7}與中括號外的第一個網頁 C 的支持度。頻繁網頁 B 的條件資料庫如表 4 所示。而所產生之以 B 為開頭的頻繁 2-序列為<BC>和<[BB{5}]>。

表 4: 頻繁網頁 B 的條件資料庫

TID	位置
1	7
2	7
6	7

WTPM接著建立頻繁(k+1)-序列(k≧1)的條件資料庫，並產生頻繁(k+2)-序列。對於某一頻繁k-序列X=<x_1, …, x_k>，WTPM掃描X的條件資料庫，建立以X為首的頻繁(k+1)-序列的條件資料庫。對於某一個以X為首的頻繁(k+1)-序列 <x_1, …, x_k, x_{k+1}>，從X的條件資料庫中，根據所記錄的TID t與位置P，在顧客瀏覽交易序列資料庫中，從TID t的顧客瀏覽交易序列的位置P開始掃描，若有出現x_{k+1}則輸出TID t與x_{k+1}的下一個位置到頻繁(k+1)-序列<x_1, …, x_k, x_{k+1}>的條件資料庫，並計數所輸出之位置的網頁(或網頁結合一購買項目)的支持度。若x_{k+1}出現在中括號內，則要計數x_{k+1}之後中括號之內的所有網頁結合一購買項目的支持度，以及中括號外的第一個網頁的支持度。若x_{k+1}的下一個位置已沒有任何網頁，則不做輸出。若所計數之網頁(或網頁結合一購買項目) z的支持度有達到最小支持度，則表示<x_1, …, x_k, x_{k+1},z>為頻繁(k+2)-序列。因此掃描完X的條件資料庫後，所有以X為首的頻繁(k+1)-序列的條件資料庫以及以X為首的頻繁(k+2)-序列就可產生。

表 5: 頻繁 2-序列 <BC>的條件資料庫

TID	位置
2	10
6	10

表 6: 頻繁 2-序列 <[BB{5}]> 的條件資料庫

TID	位置
1	9
2	9
6	8

例如在表 3 中，頻繁網頁 B 的條件資料庫如表 4 所示。對於以 B 為首的頻繁 2-序列 BC，在 B 的條件資料庫中，第一筆記錄為 TID 1 與位置 7，因此我們從表 2 的 TID 1 之顧客瀏覽交易序列的位置 7 開始掃描，因為 C 的下一個位置已沒有任何網頁，所以不做輸出。B 的條件資料庫中，第二筆記錄為 TID 2 與位置 7，表 3 中 TID 2 之顧客瀏覽交易序列的位置 7 之後有出現 C，且 C 的下一個位置是 10，所以輸出 TID 2 與位置 10 到頻繁 2-序列 BC 的條件資料庫，並計數位置 10 的網頁，也就是 K 的支持度。掃描完頻繁網頁 B 的條件資料庫，就可建立以 B 為首的頻繁 2-序列<BC>和<[BB{5}]>的條件資料庫，如表 5 和表 6 所示。WTPM 遞迴地建立頻繁(k+1)-序列(k≧1)的條件資料庫，並產生頻繁(k+2)-序列，直到無法產生頻繁序列為止，所產生的頻繁序列即為網路交易型樣。

四、漸進式網路交易型樣探勘

我們所提出之漸進式網路交易型樣探勘演算法 IWTPM，當有新的顧客瀏覽交易序列加入，或者舊有的顧客瀏覽交易序列被移除，都可以很快速的從新增或移除的顧客瀏覽交易序列，以及之前探勘所留下之頻繁序列的條件資料庫，得到資料更新後的網路交易型樣。

當有顧客瀏覽交易序列新增時，IWTPM 首先會將每一筆新增的顧客瀏覽交易序列轉換成如表 3 的型式。接著由左到右掃描每一筆新增的顧客瀏覽交易序列，將其 TID 與每一網頁 x 在此顧客瀏覽交易序列第一次出現的下一個網頁(或網頁結合一購買項目) y 的位置，新增到網頁 x 的條件資料庫，並累計網頁 x 與 y 的支持度。若 x 出現在中括號內，則要累計 x 之後中括號之內的所有網頁結合一購買項目的支持度，以及中括號外的第一個網頁的支持度。

在所有網頁的條件資料庫都新增完成後，若原來的頻繁網頁的支持度未達到最小支持度，則將此頻繁網頁加到非頻繁網頁的行列中，並將以此頻繁網頁為首的所有頻繁 k-序列(k≧2)的條件資料庫刪除，因為這些頻繁 k-序列也都變成了非頻繁 k-序列。若原來的頻繁網頁的支持度仍有達到最小支持度，但以此頻繁網頁為首的頻繁 k-序列(k≧2)未達到最小支持度，則由此頻繁 k-序列的條件資料庫所衍生出來的所有頻繁序列也都變成非頻繁序列，所以此頻繁 k-序列以及其所衍生的所有頻繁序列的條件資料庫都予以刪除。若以此頻繁網頁為首的非頻繁 k-序列(k≧2)有達到最小支持度，也就是變成頻繁 k-序列，則建立此頻繁 k-序列的條件資料庫，並繼續掃描此條件資料庫，產生以此頻繁 k-序列為首的頻繁(k+1)-序列與其條件資料庫，直到沒有頻繁序列產生為止。

對於原來的非頻繁網頁，若新增顧客瀏覽交易序列後，其支持度有達到最小支持度，則變為頻繁網頁，因此掃描此頻繁網頁的條件資料庫，產生以此頻繁網頁為首的頻繁 2-序列與其條件資料庫，再繼續掃描頻繁 2-序列的條件資料庫，產生以此頻繁網頁為首的頻繁 k-序列(k≧3)與其條件資料庫，直到沒有頻繁序列產生為止。

表 7: 網頁 C 的條件資料庫

TID	位置
2	10
3	7
4	8
5	8
6	10
7	8

例如若我們新增一筆顧客瀏覽交易序列 (7, DAFG[BB{5}]CKH)，因為此顧客瀏覽交易序列有出現網頁 A, B, C, D, F, G, H 和 K，所以這些網頁的條件資料庫都新增一筆記錄為 TID 7 以及每個網頁的下一個位置。例如新增 TID 7 後，網頁 C 的條件資料庫如表 7 所示。網頁 C 的支持度為 6/7，仍然大於最小支持度 50%，故仍為頻繁網頁。但以網頁 C 為首的頻繁 2-序列 <[CC{1}]>, <[CC{6}]>和<[CC{8}]>，原來的支持度為 3/6 = 50%，新增 TID 7 後，累計的支持度為 3/7，小於最小支持度 50%，變為非頻繁網頁。故將 2-序列<[CC{1}]>, <[CC{6}]>和 <[CC{8}]>的條件資料庫刪除，而之前由這些 2-序列所衍生出之所有頻繁序列的條件資料庫也都全部刪除，因為這些頻繁序列也都變成非頻繁序列。

當有顧客瀏覽交易序列被刪除時，IWTPM 由左到右掃描每一筆被刪除的顧客瀏覽交易序列，將其 TID 與每一網頁 x 在此顧客瀏覽交易序列第一次出現的下一個網頁(或網頁結合一購買項目) y 的位置，從網頁 x 的條件資料庫中刪除，並遞減網頁 x 與 y 的支持度計數。若 x 出現在中括號內，則要遞減 x 之後中括號之內的所有網頁結合一購買項目的支持度計數，以及中括號外的第一個網頁的支持度計數。

在所有網頁的條件資料庫都刪除完成後，若原來的頻繁網頁的支持度未達到最小支持度，則將此頻繁網頁加到非頻繁網頁的行列中，並將以此頻繁網頁為首的所有頻繁 k-序列(k≧2)的條件資料庫刪除，因為這些頻繁 k-序列也都變成了非頻繁 k-序列。若原來的頻繁網頁的支持度仍有達到最小支持度，但以此頻繁網頁為首的頻繁 k-序列(k≧2)未達到最小支持度，則由此頻繁 k-序列的條件資料庫所衍生出來的所有頻繁序列也都變成非頻繁序列，所以此頻繁 k-序列以及其所衍生的所有頻繁序列的條件資料庫都予以刪除。若以此頻繁網頁為首的非頻繁 k-序列(k≧2)有達到最小支持度，也就是變成頻繁 k-序列，則建立此頻繁 k-序列的條件資料庫，並繼續掃描此條件資料庫，產生以此頻繁 k-序列為首的頻繁(k+1)-序列與其條件資料庫，直到沒有頻繁序列產生為止。

對於原來的非頻繁網頁，若刪除顧客瀏覽交易序列後，其支持度有達到最小支持度，則變為頻繁網頁，因此掃描此頻繁網頁的條件資料庫，產生以此頻繁網頁為首的頻繁 2-序列與其條件資料庫，再繼續掃描頻繁 2-序列的條件資料庫，產生以此頻繁網頁為首的頻繁 k-序列(k≧3)與其條件資料庫，直到沒有頻繁序列產生為止。

接續上例，在新增 TID 7 之後，我們刪除第一筆顧客瀏覽交易序列(TID 1)，網頁 A, B, C, D, F 和 G 的條件資料庫中，TID 為 1 的記錄也都要刪除，其支持度計數也都要遞減 1。而在刪除第一筆顧客瀏覽交易序列後，原來的非頻繁網頁 E, H 和 I 的支持度都已達到最小支持度，因此轉變成頻繁網頁。對於頻繁網頁 E, H 和 I，IWTPM 根據其條件資料庫，掃描顧客瀏覽交易序列資料庫，找出在每一條件資料庫中的頻繁網頁。以網頁 I 為例, 其條件資料庫中的頻繁網頁為 C，因此<IC>為網路交易型樣，因此建立<IC>的條件資料庫，並計數 IC 條件資料庫中每一網頁的支持度，可產生網路交易型樣<I[CC{1}]>，<I[CC{6}]>和<I[CC{8}]>，依此類推即可產生以 I 為首的所有網路交易型樣。

五、實驗結果

由於目前尚未有遞增式探勘網路交易型樣的論文與演算法，所以我們將我們提出的 IWTPM 演算法與我們的網路交易型樣演算法 WTPM，以及網路瀏覽型樣演算法 IncWTP [9]做執行時間與記憶體使用空間上的比較。在以下的實驗中，我們使用虛擬的網頁瀏覽資料來進行實驗。我們設定最小支持度 0.8%，網頁個數為 1200，我們將原始資料的筆數增加 2K 到 20K。圖 3，4，5 和 6 分別是原始資料筆數為 30K, 50K, 70K 和 100K 三個演算法的執行時間。從圖中可以看出 IWTPM 演算法比 IncWTP 更有效率，也比重新探勘的網路交易型樣演算法 WTPM 有效率。

因為 IncWTP 類似於 Apriori 演算法，將新增的顧客瀏覽序列所產生的頻繁 k-瀏覽序列(k≧1)做組合，產生新的(k+1)-候選瀏覽序列，再將頻繁 k-瀏覽序列的顧客編

號做交集，計算出(k+1)-候選瀏覽序列的支持度，以決定頻繁(k+1)-瀏覽序列，如此遞迴下去，而且在過程中也必需去更新 Lattice 的結構，耗費相當多的時間。而 IWTPM 演算法只需針對新增的顧客瀏覽序列，調整條件資料庫的內容，就可決定哪些頻繁瀏覽序列已變成非頻繁瀏覽序列，哪些非頻繁瀏覽序列轉變成頻繁瀏覽序列，再對新產生的頻繁瀏覽序列產生條件資料庫即可。

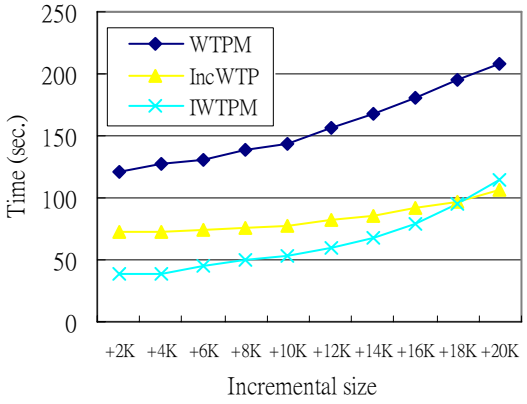


圖 3: 原始資料量為 30K 各演算法的執行時間

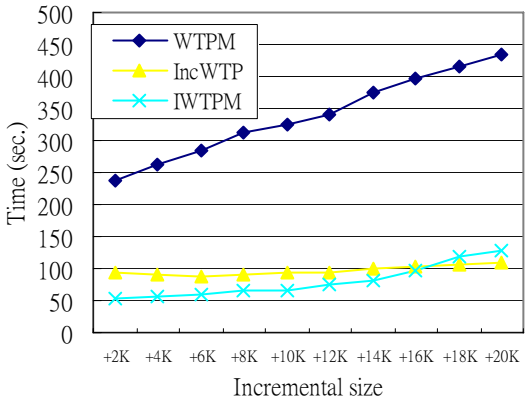


圖 4: 原始資料量為 50K 各演算法的執行時間

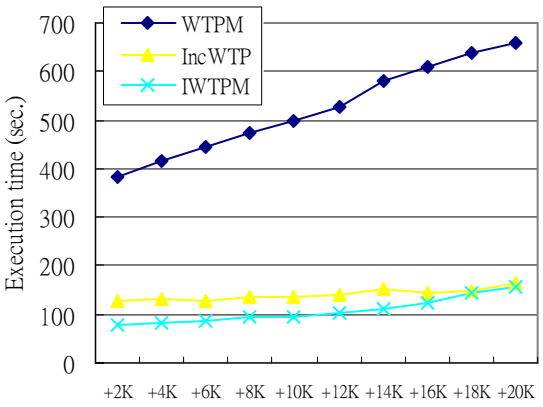


圖 5: 原始資料量為 70K 各演算法的執行時間

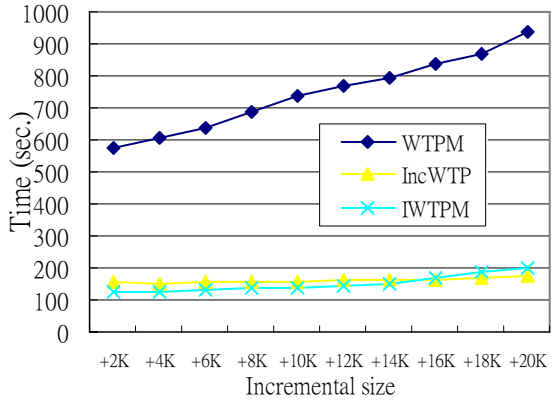


圖 6: 原始資料量為 100K 各演算法的執行時間

圖 7，8，9 和 10 分別是原始資料筆數為 30K, 50K, 70K 和 100K 時演算法 IWTPM 與 IncWTP 記憶體使用空間的比較。從實驗中顯示 IncWTP 比 IWTPM 要花費更多的記憶體空間。因為 IncWTP 需要將頻繁瀏覽序列以及其所組合出來的候選瀏覽序列的支持度與顧客編號儲存在 Lattice 結構中。而 IWTPM 只需對每一頻繁瀏覽序列產生條件資料庫，並記錄其顧客編號與在資料庫中此頻繁瀏覽序列的下一個位置即可。所以所需使用到的記憶體空間比 IncWTP 還少。

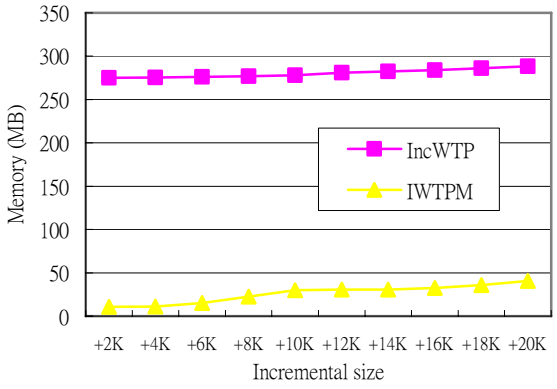


圖 7: 原始資料量為 30K 時 IWTPM 與 IncWTP 的使用空間比較

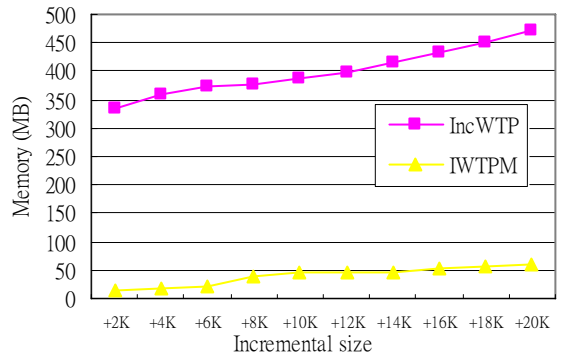


圖 8: 原始資料量為 50K 時 IWTPM 與 IncWTP 的使用空間比較

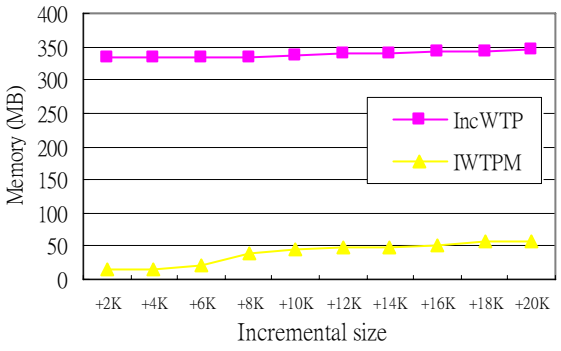


圖 9: 原始資料量為 70K 時 IWTPM 與 IncWTP 的使用空間比較

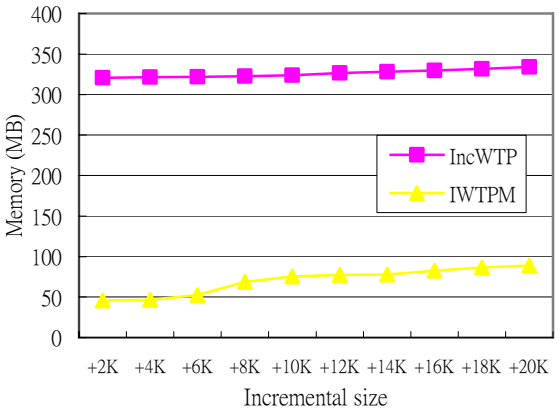


圖 10: 原始資料量為 100K 時 IWTPM 與 IncWTP 的使用空間比較

六、結論

在這篇論文中，我們提出網路交易型樣探勘演算法 WTPM 與漸進式網路交易型樣探勘演算法 IWTPM。WTPM 演算法只需掃描顧客瀏覽交易序列資料庫兩次，再建立網路交易型樣的條件資料庫，就可從資料量極少的條件資料庫中，找出所有的網路交易型樣。而 IWTPM 不需掃描原始的顧客瀏覽交易序列資料庫，只需針對新增或移除的顧客瀏覽交易序列，新增或刪除件資料庫的內容，就可產生更新資料庫後的網路交易型樣。由於目前尚無遞增式探勘網路交易型樣的研究，我們將我們的演算法與遞增式探勘網路瀏覽型樣的演算法 IncWTP 做執行時間與所使用的記憶體空間的比較。實驗結果顯示，雖然 IWTPM 多考慮了顧客交易的行為，在執行速度上仍比沒有考慮顧客交易的行為的 IncWTP 演算法有效率，且所使用的記憶體空間也比較少。

七、誌謝

這篇論文的研究成果是國科會計劃(NSC 96-2221-E-130-005 和 NSC 96-2221-E-130-017)的一部份。我們在此感謝國科會經費支持這個計劃。

八、參考文獻

[1] M. S. Chen, X. M. Huang and I. Y. Lin. “Capturing User Access Patterns in the Web for Data Mining.” Proceedings of the IEEE International Conference on Tools with Artificial Intelligence, pp. 345-348, 1999.

[2] R. Cooley, B. Mobasher, and J. Srivastava, “Web Mining: Information and Pattern Discovery on the World Wide Web”, Proceedings of IEEE International Conference on Tools with Artificial Intelligence, 1997.

[3] M. S. Chen, J. S. Park and P. S. Yu. “Efficient Data Mining for Path Traversal Patterns in a Web Environment.” IEEE Transaction on Knowledge and Data Engineering, Vol. 10, No. 2, pp. 209-221, 1998.

[4] H. Cheng, X. Yan, and J. Han, “IncSpan: Incremental Mining of Sequential Patterns in Large Database”, Proceedings of 2004 International Conference on Knowledge Discovery and Data Mining (KDD'04), Seattle, WA, Aug. 2004.

[5] C.I. Ezeife and Min Chen, “Incremental Mining of Web Sequential Patterns Using PLWAP Tree on Tolerance MinSupport”, proceedings of the IEEE 8th International Database Engineering and Applications Symposium, Coimbra, Portugal, July 7th to 9th, 2004, pp. 465-479.

[6] Maged EL-Sayed, Carolina Ruiz, and Elke A. Rundensteiner, “FS-Miner: Efficient and Incremental Mining of Frequent Sequence Patterns in Web logs”, Proceedings of 6th ACM International Workshop on Web Information and Data Management, pp.128-135, 2004.

[7] J. Han, J. Pei, Y. Yin: Mining Frequent Patterns without Candidate Generation. ACM SIGMOD, pages 1-12, 2000.

[8] Y.S. Lee and S.J. Yen, "Mining Web Transaction Patterns in an Electronic Commerce Environment," Lecture Notes in Computer Science (LNCS): Advances in Web and Network Technologies, and Information Management, Vol. 4537, pp. 74-85, June 2007.

[9] Y.S. Lee, S.J. Yen and M.C. Hsieh, "A Lattice-Based Framework for Interactively and Incrementally Mining Web Traversal Patterns," International Journal of Web Information Systems (IJWIS), Vol. 1, No. 4, pp. 197-207, December 2005

[10] Y.S. Lee, S.J. Yen, G.H. Tu and M.C. Hsieh, “Mining Traveling and Purchasing Behaviors of Customers in Electronic Commerce Environment”, Proceedings of IEEE International Conference on e-Technology, e-Commerce and e-Service, pp. 227-230, 2004.

[11] J. Pei, J. Han, B. Mortazavi-Asl and H.Zhu. “Mining Access Patterns Efficiently from Web Logs.” Proceedings of the Pacific-Asia Conference on Knowledge Discovery and Data Mining, pp. 396-407, 2000.

[12] S. Parthasarathy, M.J. Zaki, M. Ogihara, S. Dwarkadas, “Incremental and Interactive Sequence Mining”, Proceedings of International Conference on Information and Knowledge Management, pp. 251-258, 1999.

[13] Y. Xiao and M. H. Dunham. “Efficient Mining of Traversal Patterns.” IEEE Transaction on Data and Knowledge Engineering, Vol. 39, No. 2, pp. 191-214, 2001.

[14] Y. Xiao, J. F. Yao and G. Yang, “Discovering Frequent Embedded Subtree Patterns from Large Databases of Unordered Labeled Trees,” International Journal of Data Warehousing and Mining, Volume 1, Issue 2, pp.70-92, 2005.

[15] C. H. Yun and M. S. Chen. “Using Pattern-Join and Purchase-Combination

for Mining Web Transaction Patterns in an Electronic Commerce Environment." Proceedings of the COMPSAC, pp. 99-104, 2000.

[16] S. J. Yen. "An Efficient Approach for Analyzing User Behaviors in a Web-Based Training Environment." International Journal of Distance Education Technologies, Vol. 1, No. 4, pp.55-71, 2003.

[17] S.J. Yen and Y.S. Lee, "An Incremental Data Mining Algorithm for Discovering Web Access Patterns," International Journal of Business Intelligence and Data Mining, 2006.

[18] Y. S. Lee, S. J. Yen, S. S. Lin and Y. C. Liu, "Integrating Multidimensional Association Rule Mining into Classification," Proceedings of International Conference on Informatics, Cybernetics, and Systems, pp. 831-836, December 2003.