在移動視窗模式中發掘資料流前 K 個封閉循序樣式

蔡秀滿
明新科技大學資訊工程系
e-mail：pauray@must.edu.tw

摘要

循序樣式探勘(sequential pattern mining)是資料探勘領域中一項重要的研究。隨著各種相關應用的興起，例如網頁點選串流探勘、網路入侵偵測、以及線上交易分析等，我們所要處理的資料不再是靜態的資料，而是一連串即時且連續的動態資料流(dynamic data stream)。傳統循序樣式探勘的方法在處理資料的過程中，經常需要多次讀取資料庫。然而在高速資料流的環境中，我們只允許資料被讀取一次，並且必須在最短的時間內產生分析的結果。本論文考慮在移動視窗模式(sliding window model)的資料流環境中，設計一個有效率的方法來發掘資料流前 *k* 個封閉循序樣式(top-*k* closed sequential patterns)。由於設定適當的支持度不是一件容易的事，因此先前相關的研究提出讓使用者設定“最小長度”來取代“最小支持度”，目的是發掘長度大於或等於最小長度的封閉循序樣式。然而，這種方式的探勘結果會遺失許多長度較小，但具有高支持度的循序樣式之重要資訊。由於長度愈長的循序樣式，其支持度相對地也愈小，應用上的價值將會降低，因此本論文提出以“最大長度”來取代“最小支持度”，發掘出所有長度小於或等於最大長度的封閉循序樣式。

關鍵詞：資料探勘、資料流探勘、移動視窗、循序樣式、封閉循序樣式

Abstract

Sequential pattern mining is an important research topic in the data mining community. With the emergence of new applications, the data we need to process is not again static, but the continuous dynamic data stream. Examples include network traffic analysis, Web click stream mining and on-line transaction analysis. In the process of mining association rules, traditional methods need to read the database more than once. However, due to the consideration of performance and storage constraints, on-line data stream mining algorithms are restricted to making only one pass over the data. In this paper, we design an efficient algorithm for mining top-k closed sequential patterns over a stream sliding window. Because finding an appropriate support threshold is not easy, previous related researches use “minimum length” to replace “minimum support”. That is to discover closed sequential patterns of length no less than the minimum length. However, the method will lose information about sequential patterns with short length but high support. Thus, we use “maximum length” in this paper instead.

Keywords: data mining, data stream mining, sliding window, sequential pattern, closed sequential pattern

1. 前言

近年來，資料探勘(data mining)已廣泛地被應用在許多不同的領域[1,6,10,15]。例如，在大型交易資料庫中發掘相關物品的關連法則(association rules)[2]，發掘和使用者購物時間相關的循序樣式(sequential patterns)[3]，以及分析在網際網路中的使用者行為[4,21]等。這些傳統的資料探勘應用所使用的資料，主要是來自於靜態的儲存設備，也就是說資料的更新是定期進行，而不是一種動態持續的方式。

然而隨著各種新興應用的崛起，例如網路流量分析、網頁點選串流探勘、網路入侵偵測以及線上交易分析等，我們所要處理的資料不再是靜態的資料，而是一連串即時且連續的動態資料流(dynamic data stream)[8,9,11,15]。由於資料流是以極快速的方式，大量且持續產生的一連串資料，因此資料流探勘必須滿足下列三個要求[23]：(1)資料流中的每一個資料項目最多只能被檢視一次。(2)雖然在資料流中新資料不斷的產生，但是資料流探勘所使用的記憶體是有限的。(3)最新的資料流探勘的結果必須要能夠配合使用者的要求快速地產生。為了達到上述的需求，資料流探勘通常會允許分析的結果具有某種程度的誤差，以達到縮短處理時間的目的。

在資料探勘的領域裡，一項重要的應用是在大型交易資料庫中發掘循序樣式(sequential

patterns)[3,12,17,22]。例如，顧客購買書本 B1 和 B5 之後，下一次會購買 B3，再接下來會購買 B2 和 B4。項目(item) 的集合稱之為“項目集”(itemset)。具體地說，一個“序列”(sequence) 是 itemsets 的有序串列(order list)。Sequence S 被表示為 $<s_1\ s_2 \ldots s_n>$，其中 s_k 是一個 itemset，$1 \le k \le n$。Sequence $<a_1 a_2 \ldots a_n>$ 被定義為“**包含在** sequence $<b_1 b_2 \ldots b_m>$ 中”，假如存在整數 $i_1 < i_2 < \ldots < i_n$ 滿足 $a_1 \subseteq b_{i1}$，$a_2 \subseteq b_{i2}$，…，$a_n \subseteq b_{in}$。在一組 sequences 中，如果 sequence S 不被包含在其它 sequences，則稱它為 **maximal**。每一個 maximal sequence 代表一個“循序樣式”(sequential pattern)，滿足最小支持度 (minimum support) 的 sequence 稱之為 frequent sequence。一旦所有的 frequent sequences 被發掘之後，sequential patterns 的產生將變得非常容易。

傳統的循序樣式探勘(sequential pattern mining)有下列兩個重要的問題：第一個問題是使用者必須自行設定“最小支持度”(minimum support)。支持度的設定必須考慮資料的特性，經過不斷的測試和調整之後才能得到適當的值。然而在實際應用上，使用者要找到最佳的最小支持度是一件非常不容易的事。第二個問題是探勘的結果產生了大量的 sequential sequences，提高了應用上的困難度。

在傳統的關連法則探勘[2]上也有類似的情況發生，包括“最小支持度”的設定以及產生大量 frequent itemsets 的問題。Pasquier et al.[16] 最早提出頻繁封閉項目集(frequent closed itemsets)的觀念，用來解決產生大量 frequent itemsets 的問題。一個 frequent closed itemset 的支持度必定大於任何它的 proper superset 的支持度。frequent closed itemsets 的數量不僅遠低於 frequent itemsets，而且它可以保持結果的完整性(completeness)，也就是說，可以從 frequent closed itemsets 還原所有的 frequent itemsets。

在循序樣式探勘的研究方面，Yan et al.[20] 也提出類似的觀念來解決產生大量sequential sequences 的問題。他們設計了一個稱為 CloSpan 的演算法來發掘封閉的循序樣式 (closed sequential patterns)。一個 closed sequential pattern 的支持度必定大於任何它的 superpattern 的支持度。Closed sequential patterns 的數量遠低於 sequential patterns，而且可以從 closed sequential patterns 還原所有的 sequential patterns。

在“最小支持度”設定的問題上，Han et al.[13,19] 提出 mining top-k frequent closed itemsets of length no less than min_l 的想法。也就是說，使用“最小長度”(min_l) 來取代 minimum support threshold，並且根據使用者所設定的“min_l”，發掘前 k 個長度大於或等於 min_l 的 frequent closed itemsets。他們使用一種混合 top-down 和 bottom-up FP-Tree 的搜尋技術，設計一個有效率的演算法(稱之為 TFP)來解決這個問題。他們也將“最小長度”的觀念應用在 sequential pattern mining，提出 TSP 演算法 [18] 來發掘 top-k closed sequential patterns。

在資料流的環境中發掘 sequential patterns 的研究已逐漸受到重視[5,7,14]。Chen et al.[5] 提出在多資料流的環境中發掘 sequential patterns 的方法。他們以PrefixSpan 演算法為基礎，設計一個稱為 MILE 的演算法，有效地執行資料流探勘的工作。MILE 演算法利用現有 patterns 的特性，避免掉許多不必要的資料掃描，以提升發掘新的 patterns 的效率。這篇論文所考慮的資料是由 events 所組成的類別性資料流 (categorical data stream)，和 PrefixSpan 演算法應用在交易資料(transactional data)上有所差異。

Ho et al.[14] 提出一個 single-pass 演算法來發掘資料流中的 sequential patterns，稱之為 IncSPAN。傳統的移動視窗模式是保留所有最近的N筆交易來進行探勘，而他們則對 sequence 的資料提出了新的移動視窗的概念，針對每一個顧客保留他們各別最近的N筆交易。他們使用 bit-sequence 的表示方式來儲存資料，配合 sequence tree 的建立，有效地降低執行時所需要的時間和記憶體。

在資料流的探勘環境中，發掘 sequential patterns 也存在著傳統關連法則探勘所遇到的兩個問題：如何設定適當的“最小支持度”以及如何精簡 frequent sequences 的數目。先前相關的資料流探勘的研究，尚未提出有效的方法能夠同時來解決這兩個問題。在本論文中，我們考慮在移動視窗模式(sliding window model)的資料流環境中，設計解決這兩個問題的方法。Tzvetkov et al.[18] 的方法是根據使用者所設定的“最小長度”來發掘長度大於或等於最小長度的封閉循序樣式(closed sequential patterns)。他們所考慮的環境不是資料流的環境，而且探勘的結果可能會遺失許多長度較

小，但具有高支持度的 sequential patterns 之重要資訊。由於長度愈長的 sequential patterns，其支持度相對的也愈小，應用上的價值將會降低。因此，在資料流的環境中探勘 sequential patterns 的技術仍有許多可以改進的空間。本論文的研究目的是在移動視窗的模式中設計一個 "one-pass" 演算法，以 "最大長度"(maximum length) 來取代 "最小支持度"，有效率地從交易資料流中發掘出前 *k* 個長度小於或等於最大長度的 closed sequential patterns。

2. one-pass 演算法之設計

在這一節，我們將在移動視窗模式(sliding window model) 的資料流環境中，設計一個有效率的方法來發掘資料流中前 k 個封閉循序樣式(top-k closed sequential patterns)。研究內容主要包含三個部份：以序列(sequence)中所包含的項目集(itemsets)個數當作序列的長度定義、使用 "最大長度"(maximum length)來取代 "最小支持度"(minimum support)以及設計一個有效率的演算法來發掘資料流中前 *k* 個封閉循序樣式。

2.1 以序列中所包含的項目集個數當作序列的長度定義

Tzvetkov et al.[18] 提出使用 "最小長度" (*min_l*) 來取代 "最小支持度" 的方法，根據使用者所設定的 "*min_l*" 來發掘前 *k* 個長度大於或等於 *min_l* 的 closed sequential patterns。他們定義 sequence *S* 的長度為在 *S* 中的 items 之個數，然而以這種方式定義序列的長度可能會產生不符預期的結果。以表1為例，考慮下面的例子：

例一：發掘 top-2 closed sequential patterns with *min_l* = 2
Tzvetkov et al.[30] 的方法得到的結果是: {<(b c d)>: 3, <(d)(e)>:3, <(e f)>:3}，它表示 closed sequential patterns <(b c d)> 的支持個數為 3，<(d)(e)> 的支持個數為 3，<(e f)> 的支持個數為 3。sequential pattern mining 的目的是希望找出項目集出現的先後關係，雖然 <(b c d)> 和 <(e f)> 符合長度大於或等於2的條件，但是這兩個序列中都只包含一個項目集，和 sequential pattern mining 的實際意涵有落差。

上述問題主要是因為 sequence 的長度定義所造成的。我們認為 sequence 的長度應該維持最早提出 sequential pattern mining 的研究[3]中所作的定義。也就是說，sequence *S* 的長度應該定義為 "在 *S* 中的 itemsets 之個數"。Tzvetkov et al.[18] 改變原始 sequence 長度定義的原因，可能是為了方便 search tree 的建立，但是很明顯地，它會產生上述例子在探勘的結果中，有些 closed sequential patterns 只包含一個 itemset 的情況。因此在本論文中，我們的研究將以序列中所包含的 itemsets 之個數當作序列的長度定義。以例一來說，改變長度的定義之後，我們得到的結果是: {<(d)(e)>:3, <(d)(e f)>:2, <(e f)(g)>:2, <(b c d)(e)>:2, <(d)(f)(g)>:2, <(d)(e)(g)>:2}。這個結果比原來例一的結果更具意義。

表1 交易資料庫

ID	Sequence
1	<(a b c d) (e f) (g)>
2	<(a)(b c d)(k n)>
3	<(d)(e f m)(g)>
4	<(e f k)(a b)(c)>
5	<(b c d)(e)>

2.2 使用 "最大長度" 來取代 "最小支持度"

我們發現，Tzvetkov et al.[18] 使用 "最小長度" (*min_l*) 來取代 "最小支持度" 的方法也會產生許多問題。以表1為例，考慮下面的例子：

例二：發掘 top-2 closed sequential patterns with *min_l* = 3
以 Tzvetkov et al.[18] 的方法得到的結果是: {<(bcd)>:3, <(d)(e f)>:2, <(e f)(g)>:2, <(b c d)(e)>:2, <(d)(f)(g)>:2, <(d)(e)(g)>:2}。若定義序列的長度為序列中所包含的 itemsets 之個數，則得到的結果為: {<(d)(f)(g)>:2, <(d)(e)(g)>:2}。在這兩個結果中，我們都看不到 <(d)(e)>:3 的資訊，但根據定義，事實上 <(d)(e)> 也是 closed sequential pattern，若單純從結果來看，則可能會誤解 <(d)(e)> 的支持個數為2，與事實情況不符。

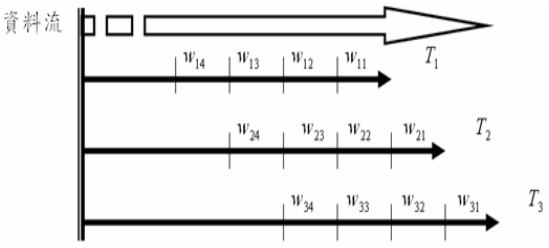
造成上述問題的主要原因是因為執行的結果只列出長度大於或等於 "最小長度" 的 closed sequential patterns，導致在結果中失去了長度小於 "最小長度" 但具有高支持度的 closed sequential patterns 之重要資訊。假設 *X*

是出現在結果中的 closed sequential patterns，對於任何 X 的 sub-sequence X'，若 X' 亦為 closed sequential pattern(i.e., X' 的支持個數大於 X)，則 X' 也應該出現在結果中，如此才不會遺失重要的資訊。因此在本論文中，我們除了以序列中所包含的 itemsets 個數當作序列的長度定義，也將以“最大長度”(*max_l*) 來取代“最小支持度”，以發掘前 k 個長度小於或等於“最大長度”的 closed sequential pattern。k 值可由使用者根據需要自行設定，並且自動排除只包含一個 itemset 的 sequence (亦即內定*min_l* = 2)。以表1為例，考慮下面的例子：

例三：發掘 top-2 closed sequential patterns with *max_l* = 3
得到的結果是: {<(d)(e)>:3, <(d)(e f)>:2, <(e f)(g)>:2, <(b c d)(e)>:2, <(d)(f)(g)>:2, <(d)(e)(g)>:2}。這個結果比例二的結果更具意義。

2.3 發掘 top-*k* closed sequential patterns with *max_l = l* 的演算法之設計

以交易資料流的循序樣式探勘來說，Ho et al.[14] 所考慮的移動視窗模式(sliding window model) 是保留每個顧客最近的 N 次交易。這種定義方式可能會造成同一個顧客相鄰兩次的交易時間間隔太長，因而降低了 sequential pattern 的應用價值。在本論文中，我們所使用的移動視窗模式如圖一所示。假設使用者所設定的視窗個數為 n，每一個視窗所包含的時間為 t，則我們所考慮的範圍是以現在的時間點為基準，往前 $n \times t$ 的時間內之所有交易。



圖一： 移動視窗模式

資料流中的資料是一筆一筆的原始交易記錄(如表 2)，為了快速產生探勘的結果，資料流中的每一筆交易將只能被讀取一次，因此，相關的資訊必須在第一次讀取時被記錄下來，如此才能處理後續的查詢。假設目前的時間點為 T_i，移動視窗為 $w_{i1}, w_{i2}, \ldots, w_{in}$。我們定義 $TS_i(X)$為包含 itemset X 的交易之相關資訊所成的集合。例如在 $TS_i(X)$中的某一個元素 y_j，它表示編號為 y 的顧客，在第 j 次的交易中包含項目集 X。$TC_i(X)$則表示 $TS_i(X)$中的元素個數。在不影響下面討論的情況下，我們將暫時省略參數 i。

表 2 交易資料流

視窗	顧客編號	項目集
w_{11}	1	abcd
	2	a
	3	d
	4	efk
	5	bcd
	1	ef
w_{12}	1	g
	2	bcd
	4	ab
	5	e
w_{13}	2	kn
	3	efm
	4	c
	3	g

長度為 m 的 sequence 稱之為 m-sequence，它包含 m 個 itemsets。我們將根據下列幾點推論，設計一個發掘 top-k closed sequential patterns with *max_l = l* 的演算法。

首先，我們從 1-sequence 開始考慮，接著考慮 2-sequence、3-sequence，依此類推下去。雖然在我們的研究中，將排除探勘結果包含 1-sequence 的情況，不過，因為 1-sequence 是產生 2-sequence 的基礎，所以我們仍然必須從 1-sequence 開始考慮。包含 n 個 item 的 itemset 稱之為 n-itemset。當我們掃描如表 2 的資料內容之後，即可得到各 item 的 TS 和 TC 值。我們將 support count 大於 0 的 itemsets 稱之為“potential closed 1-sequences”。

推論(1)：假設 L_{n-1} 為所有 potential closed (n-1)-itemsets 所成的集合，X 為一個 potential closed (n-1)-itemset，X[1],X[2],…,X[n-1]代表 X 中的 n-1 個 items，且 X[1]<X[2]<…<X[n-1]。令 Ln 為 potential closed n-itemsets 所成的集合，則

L_n={{Xp[1],Xp[2],…,Xp[n-1],Xq[n-1]}|
Xp∈Ln-1 and Xq∈Ln-1 and Xp[1]=Xq[1], Xp[2]=Xq[2], …, and Xp[n-2]=Xq[n-2], and

Xp[*n*-1]<Xq[*n*-1]}
potential closed n-itemset {Xp[1],Xp[2],…,Xp[*n*-1],Xq[*n*-1]} 的 TS 值可以由下列的運算產生：
TS({Xp[1],Xp[2],…,Xp[*n*-1],Xq[*n*-1]})
= TS(Xp)∩TS(Xq)
出現在交集結果中的元素 y_j，必須滿足存在一個元素 $v_a \in$ Xp，以及一個元素 $w_b \in$ Xq，使得 $y=v=w$，$j=a=b$。

我們可以利用推論(1)所產生的 1-sequences 來產生 potential top-k closed 2-sequences。TS(*S*)為包含 sequence *S* 的顧客之相關資訊所成的集合。假設 *S* 為一個 *m*-sequence，在 TS(*S*)中的某一個元素 $X_{a[1],a[2],...,a[m]}$，它代表編號為 *X* 的顧客，在第 $a[j]$次的交易中包含 *S* 的第 *j* 個 itemset ($1 \le i \le m$)。TC(*S*)則表示 TS(*S*)所包含的元素個數。我們稱目前所找到的前 *k* 個 closed sequences 為 "potential top-*k* closed sequential patterns"。若 sequence *S* 不是 potential top-*k* closed sequential patterns，則 *S* 的任何 super sequence *S'* 也絕對不可能為 top-*k* closed sequential patterns。這是因為 *S'* 的 support 小於或等於 *S* 的 support，既然 *S* 不是 potential top-*k* closed sequential patterns，則 *S'* 也不可能是 potential top-*k* closed sequential patterns。

推論(2)：假設 LS_1 為所有 potential closed 1-sequences 所成的集合，CP_2 為 potential closed 2-sequences 所成的集合，S[1] 代表某一個 potential closed 1-sequence *S* 所包含的 itemset，則
CP_2={<Sp[1],Sq[1]>| Sp∈LS_1 and Sq∈LS_1 }
potential closed 2-sequence <Sp[1],Sq[1]> 的 TS 值可以由下列的運算產生：
TS(<Sp[1],Sq[1]>) = TS(Sp)∩TS(Sq)
出現在交集結果中的元素 $X_{a[1],a[2]}$，必須滿足存在一個元素 $Y_{b[1]} \in$Sp，以及一個元素 $Z_{c[1]} \in$Sq，使得 $X=Y=Z$，$a[1]=b[1]$，$a[2]=c[1]$，且 $b[1]<c[1]$。我們可以從 CP_2 中找到目前的 potential top-*k* closed sequences。

推論(3)：根據推論(2)，我們可以利用已產生的 potential top-*k* closed (*m*-1)-sequences 來產生 potential closed *m*-sequences。令 P 為目前的 potential top-*k* closed sequences。假設 P_{m-1} 為 P 中所有 potential closed (*m*-1)-sequences 所成的集合，*S* 為一 potential closed (*m*-1)-sequences，S[1],S[2],…,S[*m*-1]代表 *S* 中的 *m*-1 個 closed itemsets。令 P_m 為 potential closed *m*-sequences 所成的集合，則

P_m={<Sp[1],Sp[2],…,Sp[*m*-1],Sq[*m*-1]>|
Sp∈P_{m-1} and Sq∈P_{m-1} and Sp[1]=Sq[1], Sp[2]=Sq[2], …, and Sp[*m*-2]=Sq[*m*-2]}

potential closed *m*-sequence <Sp[1],Sp[2],…,Sp[*m*-1],Sq[*m*-1]> 的 TS 值可以由下列的運算產生：

TS(<Sp[1],Sp[2],…,Sp[*m*-1],Sq[*m*-1]>)
= TS(Sp)∩TS(Sq)

出現在交集中的元素 $X_{a[1],a[2],...,a[m]}$，必須滿足存在一個元素 $Y_{b[1],b[2],...,b[m-1]} \in$Sp，以及一個元素 $Z_{c[1],c[2],...,c[m-1]} \in$ Sq，使得 $X=Y=Z$，$a[1]=b[1]$，$a[i]=b[i]=c[i-1]$ ($2 \le i \le m-1$)，$a[m]=c[m-1]$且 $b[m-1] < c[m-1]$。一旦 potential closed *m*-sequences 產生之後，再根據它的 TC 值和目前 P 的內容，即可判斷它是否為 potential top-*k* closed sequence，進而更新 P 的內容。

根據上述的推論，我們設計一個發掘交易資料流中 top-*k* closed sequential patterns with $max\_l = l$ 的 one-pass 演算法，主要含下列四個步驟：

步驟(1)：假設目前的時間點為 T_i，scan 視窗 w_{ij} ($1 \le j \le$ n) 所包含的交易資料一次，計算每一個 item 的 TS_i 和 TC_i 值。

步驟(2)：從推論(1)以及步驟(1)所得到的結果，找出所有的 potential closed 1-sequences。

步驟(3)：根據推論(2)，產生 potential top-*k* closed 2-sequences，即為 potential top-*k* closed sequences P 的初始值。

步驟(4)：根據推論(3)，產生 candidate closed *m*-sequences ($3 \le m \le l$)，從TC值即可判斷它是否為 potential top-*k* closed sequences，再進一步更新目前 P 的內容。

根據上述所設計的演算法，可以很容易地進行 top-*k* closed sequential patterns 的維護。以圖一為例，假設我們在時間點 T_1 得到在視窗 w_{1j} ($1 \le j \le 4$) 的所 top-*k* closed sequential patterns。在時間點 T_2，我們只需要掃描視窗 w_{21} 中的資料一次，就可以很容易的更新每一個 item 的 TS 和 TC 值。一旦得到每一個

item 的 TS_2 和 TC_2 的值，即可產生所有在時間點 T_2 的 closed itemsets。再根據演算法中的步驟(3)和步驟(4)，即可找出所有 top-*k* closed sequential patterns with *max_l* = *l*。

3 範例

假設使用者所設定的視窗個數 n = 4，每一個視窗所包含的時間 t = 5分鐘，表2為時間點T_1時各視窗的交易資料。假設 k =3，根據2.3節 one-pass 演算法的步驟，發掘 top-3 closed sequential patterns with max_*l* = 3 的進行過程如下：

在步驟 1，假設目前的時間點為 T_1，掃描表 2 所有視窗中包含的交易資料一次，計算每一個 item 的 TS 和 TC 值。表 3 是只包含一個 item 的 1-sequence 中，TC 值在前三名者。在表 3 中，TC 值最小者為 2。

表3 包含一個 item 的 potential closed 1-sequence

closed 1-sequence	TS	TC
<a>	$\{1_1,2_1,4_2\}$	3
<b>	$\{1_1,2_2,4_2,5_1\}$	4
<c>	$\{1_1,2_2,4_3,5_1\}$	4
<d>	$\{1_1,2_2,3_1,5_1\}$	4
<e>	$\{1_2,3_2,4_1,5_2\}$	4
<f>	$\{1_2,3_2,4_1\}$	3
<g>	$\{1_3,3_3\}$	2
<k>	$\{2_3,4_1\}$	2

在步驟 2，我們根據推論(1)和步驟(1)所得到的結果，分別得到包含兩個和三個 item 的 1-sequence 且 TC 值不小於 2 者，如表 4 和表 5 所示。

表4 包含兩個 item 的 potential closed 1-sequence

closed 1-sequence	TS	TC
<ab>	$\{1_1,4_2\}$	2
<bc>	$\{1_1,2_2,5_1\}$	3
<bd>	$\{1_1,2_2,5_1\}$	3
<cd>	$\{1_1,2_2,5_1\}$	3
<ef>	$\{1_2,3_2,4_1\}$	3

表 5 包含三個 item 的 potential closed 1-sequence

closed 1-sequence	TS	TC
<bcd>	$\{1_1,2_2,5_1\}$	3

在步驟 3，我們根據推論(2)，產生 potential top-3 closed 2-sequences，如表 6 所示，即 P 的初始值。

表 6 potential top-3 closed 2-sequences

closed 2-sequence	TS	TC
<a,c>	$\{2_{(1,2)},4_{(2,3)}\}$	2
<d,ef>	$\{1_{(1,2)},3_{(1,2)}\}$	2
<bcd,e>	$\{1_{(1,2)},5_{(1,2)}\}$	2
<ef,g>	$\{1_{(2,3)},3_{(2,3)}\}$	2
<b,e>	$\{1_{(1,2)},5_{(1,2)}\}$	2
<c,e>	$\{1_{(1,2)},5_{(1,2)}\}$	2
<d,e>	$\{1_{(1,2)},3_{(1,2)},5_{(1,2)}\}$	3
<d,f>	$\{1_{(1,2)},3_{(1,2)}\}$	2
<d,g>	$\{1_{(1,3)},3_{(1,3)}\}$	2
<e,g>	$\{1_{(2,3)},3_{(2,3)}\}$	2
<f,g>	$\{1_{(2,3)},3_{(2,3)}\}$	2

在步驟 4，根據 P 的初始值產生 TC 值不小於 2 的 closed 3-sequence，如表 7 所示。因此，最後所產生的 top-3 closed sequential patterns with max_*l* = 3 為 表 6 和表 7 中的 sequences（假設不考慮 1-sequence 的情況）。

表 7 TC 值不小於 2 的 closed 3-sequence

closed 3-sequence	TS	TC
<d,e,g>	$\{1_{(1,2,3)},3_{(1,2,3)}\}$	2
<d,f,g>	$\{1_{(1,2,3)},3_{(1,2,3)}\}$	2

4 結論

本論文考慮在移動視窗模式(sliding window model)的資料流環境中，設計一個 "one-pass" 演算法，以 "最大長度"(maximum length) 來取代 "最小支持度"，有效率地從交易資料流中發掘出前 k 個長度小於或等於最大長度的封閉循序樣式。我們所提出的 one-pass 演算法只需要掃描資料一次，並且在產生 candidate closed *m*-sequences 的同時，可以馬上計算出 support count，有助於在資料流的環境中維護循序樣式。目前我們已經得到一些重要的初步成果，未來我們將持續研究，完成一個以我們的方法為基礎的資料流探勘系統，以評估系統的執行效能。

參考文獻

[1] Agrawal, R., Imielinski, T. and Swami, A., "Database Mining: A Performance Perspective," ***IEEE Transactions on***

Knowledge and Data Engineering, pp. 914-925, 1993.

[2] Agrawal, R. and Srikant, R., "Fast Algorithms for Mining Association Rules," ***Proceedings of the VLDB Conference***, pp. 487-499, 1994.

[3] Agrawal, R. and Srikant, R., "Mining Sequential Patterns," ***Proceedings of IEEE International Conference on Data Engineering***, pp. 3-14, 1995.

[4] Ceri, S., Klemettinen, M., Lanzi, P. and Milano, P., "A Tool for Extracting XML Association Rules from XML Documents," ***Proceedings of the International Conference on Tools with Artificial Intelligence***, 2002.

[5] Chen, G.., Wu, X. and Zhu, X., "Sequential Pattern Mining in Multiple Streams," ***Proceedings of the IEEE International Conference on Data Mining***, 2005.

[6] Cheng, H., Yan, X. and Han, J., "IncSpan: Incremental mining of sequential patterns in large database," ***Proceedings of the 10th ACM SIGKDD Int'l Conf. on Knowledge Discovery and Data Mining***, 2004.

[7] Chi, Y., Wang, H., Yu, P.S. and Muntz, R.R., "Moment: Maintaining Closed Frequent Itemsets over a Stream Sliding Window," ***Proceedings of the IEEE International Conference on Data Mining***, 2004.

[8] Gaber, M.M., Zaslavsky, A. and Krishnaswamy, S., "Mining Data Streams: A Review," ***SIGMOD Record***, pp. 18-26, 2005.

[9] Ghanem, T.M., Hammad, M.A., Mokbel, M.F., Aref, W.G.. and Elmagarmid, A.K., "Incremental Evaluation of Sliding-Window Queries over Data Streams," ***IEEE Transactions on Knowledge and Data Engineering***, pp. 57-72, 2007.

[10] Giannella, C., Han, J., Pei, J., Yan, X. and Yu, P.S., "Mining Frequent Patterns in Data Streams at Multiple Time Granularities," in Kargupta, H., Joshi, A., Sivakumar, K. and Yesha, Y. (eds.), ***Next Generation Data Mining***, AAAI/MIT, pp. 191-210, 2003.

[11] Golab, L. and Özsu, M.T., "Issues in Data Stream Management," ***SIGMOD Record***, 32 (2), pp. 5-14, 2003.

[12] Han, J., Pei, J. and Yin, Y., "Mining Frequent Patterns without Candidate Generation," ***Proceedings of ACM SIGMOD***, pp. 1-12, 2000.

[13] Han, J., Wang, J., Lu, Y. and Tzvetkov, P. "Mining Top-K Frequent Closed Patterns without Minimum Support," ***Proceedings of the IEEE International Conference on Data Mining***, 2002.

[14] Ho, C.C., Li, H.F., Kuo, F.F. and Lee, S.Y., "Incremental Mining of Sequential Patterns over a Stream Sliding Window," ***Proceedings of the IEEE International Conference on Data Mining***, 2006.

[15] Jiang, N. and Gruenwald, L, "Research Issues in Data Stream Association Rule Mining," ***SIGMOD Record***, 35 (1), pp. 14-19, 2006.

[16] Pasquier, N., Bastide, Y., Taouil, R. and Lakhal, L., "Discovering Frequent Closed Itemsets for Association Rules," ***Proceedings of the Database Theory***, pp. 398-416, 1999.

[17] Pei, J., Han, J., Mortazavi-Asl, B., Wang, J., Pinto, H., Chen, Q., Dayal, U. and Hsu, M.C., "Mining Sequential Patterns by Pattern-Growth: The PrefixSpan Approach," ***IEEE Transactions on Knowledge and Data Engineering***, 16(11), pp. 1424-1440, 2004.

[18] Tzvetkov, P., Yan, X. and Han, J., "TSP: Mining Top-K Closed Sequential Patterns," ***Proceedings of the IEEE International Conference on Data Mining***, 2003.

[19] Wang, J., Han, J., Lu, Y. and Tzvetkov, P., "TFP: An Efficient Algorithm for Mining Top-K Frequent Closed Itemsets," ***IEEE Transactions on Knowledge and Data Engineering***, 17(5), pp. 652-664, 2005.

[20] Yan, X., Han, J. and Afshar, R., "CloSpan: Mining Closed Sequential Patterns in Large Datasets," ***Proceedings of the SIAM***

International Conference on Data Mining, pp. 166-177, 2003.

[21] Yang, Q. and Zhang, H.H., "Integrating Web Prefetching and Caching Using Prediction Models," ***World Wide Web***, pp. 299-321, 2001.

[22] Zaki, M.J. and Hsiao, C.J., "HARM: An Efficient Algorithm for Closed Itemset Mining," ***Proceedings of the SIAM International Conference on Data Mining***, pp. 457-473, 2002.

[23] Zhu, Y. and Shasha, D., "StartStream: Statistical Monitoring of Thousands of Data Streams in Real Time," ***Proceedings of the VLDB Conference***, pp. 358-369, 2003.