行政院國家科學委員會專題研究計畫　期中進度報告

資料串流管理系統及串流資料探勘之研發(1/3)

計畫類別：個別型計畫
計畫編號：NSC92-2213-E-009-123-
執行期間：92 年 08 月 01 日至 93 年 07 月 31 日
執行單位：國立交通大學資訊工程學系

計畫主持人：李素瑛

報告類型：精簡報告

處理方式：本計畫可公開查詢

中　華　民　國 93 年 6 月 11 日

計劃中文摘要

關鍵字：資料串流、串流模式、連續性查詢、串流管理、串流探勘

目前有許多新的應用環境所產生的資料，例如電信公司通聯紀錄、電腦網路封包、網站網頁點擊串流、電力感測網路、交通感測網路與其他感測網路資料等，是一種持續不斷並且數量龐大的有序資料，我們稱之為「資料串流」。傳統的資料庫管理系統，由於並非針對「資料串流」所設計；因此，不僅不適用，而且其功能也相當的不足。因此，我們需要新的「資料串流管理系統」來管理這些持續性的大量資料串流。在這些應用環境所產生的連續性資料串流中，除了可以提供基本的資料之外，往往也隱藏了許多有用的資訊與知識，可以被用來做更進階的資料挖掘或是提供預測等的豐富應用。

資料串流模式的資料特性，可以用「你只能看一次來形容」。對於在這種資料密集的應用環境中，將資料載入傳統的資料庫系統中，再執行傳統的查詢等的資料庫操作的方式並不適用於資料串流的模式。這是因為在傳統的資料庫系統中，使用者欲執行某個查詢時，資料庫系統針對目前現有的資料，執行查詢規劃，再將結果回傳給使用者。而在資料串流系統中，因為主記憶體或是主磁碟容量有限，所以系統無法將全部的資料儲存在記憶體中。因此，我們需要採用全新的模式，並設計全新的方法以掌控串流資料，以及完成使用者的傳統查詢與新型態的查詢，在此我們稱其為「連續型查詢」。

連續型查詢是當使用者於資料串流管理系統中註冊了某種查詢之後，系統將持續對於新進的串流資料進行評估，也會持續地將查詢的結果回傳給使用者；因此，查詢結果可能會隨著時間不同而產生不同的結果。隨著具有大量且高速特性的資料串流不斷地湧入資料串流管理系統，系統的設計將面對許多新的議題與挑戰，例如無限記憶體的空間需求與會暫停運算的查詢。在設計串流資料管理系統時，我們必須使用許多的新技術與方法，包括近似的查詢回覆、滑動窗、批次作業、取樣與概要，以及解決會暫停運算的查詢演算法等。因此，我們計劃第一年的目標是建構資料串流管理系統與系統架構與資料串流模式。從宏觀的角度來探究各種可能的資料串流模式，以及資料串流相關的各種議題，並加以歸納，以架構出廣義的且可延伸的資料串流模式。此外，我們將統合不同資料串流應用的特殊性，並探討在廣義的資料串流管理系統下，傳統查詢與新型態的各種查詢，以及達成這些查詢的方法與子系統架構。根據目前所提出的系統，建構出一個完整的串流資料管理系統的系統架構。

計劃英文摘要

Keywords: data stream, streaming model, continuous query, stream management, stream mining.

Nowadays, many data-intensive applications like telecommunication networks, Web service networks, power sensor network, traffic sensor network, generate large amount of data continuously in real time. These transient data streams cannot be modeled as persistent relations. Traditional database management systems, targeting at persistent data, are becoming inadequate in supporting the functionalities and requirements of modeling this new class of modern data streams.

The unique characteristic of data streams is “you only get one look”. If data streams are simply loaded into traditional DBMS’s, some queries can not be answered. For example, some queries having aggregate operator like AVERAGE cannot be evaluated since it fails to see the end of the stream input. Consequently, we need a totally different approach in the design of systems which can model the streams to take and answer general queries. The model and the architecture of the data stream management system need to be redesigned. Besides, continuous queries in which new results corresponding to one query should be returned to users as new data keep on arriving.

As the rapid and unbounded streams flood in the data stream management systems, we are facing many new challenges, including unbounded memory requirements and blocking in querying processing. New techniques such as approximate answering, sliding window, batch-processing, sampling, synopses, and blocking resolving must be developed. In the first year, we will study the data stream models in current streaming data applications, investigate the modeling and query issues of streaming data, and then construct a general data model of streaming data and an expandable data stream management system (DSMS). The proposed general DSMS will handle traditional queries, and new continuous queries efficiently with the newly designed query processor and core streaming data extraction modules.

報告內容

一. 前言

目前有許多新的應用環境所產生的資料，例如電信公司通聯紀錄、電腦網路封包、網站網頁點擊串流、電力感測網路、交通感測網路與其他感測網路資料等，是一種持續不斷並且數量龐大的有序資料，我們稱之為「資料串流」。傳統的資料庫管理系統，由於並非針對「資料串流」所設計；因此，不僅不適用，而且其功能也相當的不足。因此，我們需要新的「資料串流管理系統」來管理這些持續性的大量資料串流。在這些應用環境所產生的連續性資料串流中，除了可以提供基本的資料之外，往往也隱藏了許多有用的資訊與知識，可以被用來做更進階的資料挖掘或是提供預測等的豐富應用。

在一個資料串流的環境（streaming environment）中，資料是以連續性的方式產生（由於此項特性，我們也可以將串流資料的大小視為是無限大），並且可能是以極高速的方式進入串流資料管理系統中。因此，在設計串流資料管理系統時，由於實際上系統所可以擁有的記憶體並不是無限的，所以我們必須設計出可儲存在主記憶體的摘要型資料結構來有效率地儲存具有無限容量特性的串流型資料。而且，在處理串流型資料時，由於資料是持續不斷地產生，所以並不容許我們對已處理過的資料，做第二次的串流資料處理，這也就是串流型資料最大的特徵—「你只能看一次來形容」[4]。

因此，傳統的資料庫管理系統不適用於串流型資料的主要原因，就是因為此種串流資料的特殊性質，在此我們可以用「你只能看一次來形容」來描述串流資料與其他資料之間的最大差異之處。因此，建構串流資料管理系統的困難度將較傳統的資料庫管理系統高出許多，但是同時串流資料管理系統在未來的發展性將更值得我們去期待。

二. 研究目的

本計劃（第一年）的目的是建構資料串流管理系統（DSMS）的系統架構（architecture）與資料串流的模式（modeling）。從宏觀的角度，探討各種可能的資料串流模式，以及與資料串流相關的各種議題，並加以歸納，以架構出廣義的並且具可延伸性的資料串流模式。此外，我們將統合並參考不同資料串流應用的特殊性，以進一步地探討在廣義的資料串流管理系統（General DSMS）下，有哪些傳統資料庫查詢與新型態的查詢可以適用在此串流環境中，以及如何達成這些查詢的方法與子系統架構。並根據目前提出的系統，建構出一個更完整的資料串流管理系統架構。同時，我們將根據整理的資料串流模型，設計資料串流管理

系統中串流資料的模擬（streaming data simulation），以提供系統做進一步的分析與其他進階功能的測試。

在資料串流管理系統的系統架構完成之後，系統將不僅止於提供簡易的串流基本資料的查詢運算，我們還可以進一步對串流資料進行知識發掘（knowledge discovery）。這是因為在資料串流的應用中，對於某些具有關鍵任務的應用而言，能在線上（on-line）－也就是當串流資料到達時－就立即進行有用或是可能有意義的樣式探勘（pattern mining），是比單純的資料查詢，更能發揮出資料串流管理系統的價值。舉例說明，在線上交易中，及時找出可能發生金融詐騙的交易事件，或是在應用系統發生異常前，線上及時對資料串流進行探勘，以達到事前預警的功能等。此外，資料串流管理系統所需的概要或是近似結果，有些是可以藉由「傳統」資料探勘的技術來完成的。舉例說明如下：

資料探勘的技術可以找出關聯規則（association rules）或是循序樣式（sequential patterns）等的頻繁樣式（frequent patterns）或是特殊樣式（outliers）；或是找出決策樹（decision tree），以及可以將資料摘要化（summarization）或是找出資料特徵（characterization），也可以對未知的資料作群聚分析（clustering），或者是提供趨勢分析（trend analysis）或預測（prediction）等的功能。如果能將將這些功能整合到資料串流管理系統中，我們將可預期此系統效能將得到更大的發揮。

此外，由於資料串流具有高速的資料到達速率（high speed arrival rates）的資料特徵，這通常代表的意思是說，串流資料通常只能以最原始的資料型態（primitive）出現。為了提供多樣化的線上分析，若是僅依靠原始資料的樣式分析，所得之結果的可用性可以會受到較大的限制。所幸在資料探勘的技術中，有一種技術可進行多維度的資料整合分析（multi-dimensional data analysis），若是能整合到資料串流管理系統中，定能提高其串流資料的線上決策分析的品質。

但是，傳統的資料探勘技術在面對資料串流的特殊資料特性（例如「只能看一次」或是無止境的接收串流資料等）時，將會面臨到可能無法直接整合的問題。這是因為之前所開發出來的資料探勘相關技術都需要掃描資料庫至少兩次以上，而且對系統在記憶體使用上並沒有任何的限制。這些技術上的特性都將導致傳統資料探勘的技術不再適用於串流性（streaming）的環境中。

基於上述的研究動機與目的，在第一年的計劃中我們整合了三種資料串流的資料模式－標的物窗模式（landmark window model）、滑動窗模式（sliding window model）與遞減窗模式（damped window model）－並根據不同性質的應用需求，動態地調整這些模式的使用時機。透過廣泛的資料串流模式的探討之後，我們認

為三種模式都有其存在的必要性。在充分了解三種模式的差異性之後，我們首先嘗試整合資料探勘技術中的變異偵測（change detection）到以標的模式為基礎的資料串流中，並提出具有可有效處理串流資料特性的廣義系統架構與方法來解決這個子問題。我們期許這個廣義的串流資料管理系統的架構可以適用於各種應用中。

三. 文獻探討

資料串流管理系統之系統架構（DSMS System Architecture）

目前最有名的串流資料管理系統整合開發計劃，首推史丹佛大學的 STREAM [19] 計劃，該計劃是希望建構出一個通用的資料串流管理系統。STREAM 的目標是建構出一個完整的資料串流管理系統所需的查詢語言（extended SQL 與 algebra）、並配置運算子與概要、並以 XML 或是 GUI 的方式來表示查詢規劃，並嘗試以視覺化的方式來呈現，並進一步加速傳統的記憶體配置，以及佇列管理等。除此之外，STREAM 著重於宣告式的 SQL，並強調語意上的精確性以及容許產生近似的結果。另外，STREAM 將中間暫時產生的結果儲存在佇列記憶體中，並透過運算子分享與資源管理進行多筆查詢處理的最佳化，但是其查詢語言較不易表達，但是系統的負擔可以透過近似處理的做法來減輕。Tapestry [6] 系統主要是針對只能持續附加（append-only）的電子郵件與 BBS 訊息進行內容的過濾，該系統並運用有限且指定的 SQL 語言子集合，來加速查詢的執行，最後輸出的結果也是以持續附加的形式呈現。Tribeca [20] 系統主要是用來做網路監測之用。Tribeca 也是使用受限的 SQL 查詢語言在網路封包串流的處理上。OpenCQ [17] 系統與 NiagaraCQ [7] 系統則是處理在廣域網路上的持續性的串流資料，並提供連續性查詢的功能。OpenCQ 使用遞增視界更新（incremental view maintenance）的技術在加速查詢處理上，而 NiagaraCQ 則是使用群組（group）的觀念（也就是提出群組連續性查詢的技術），來擴充查詢數量的能力。

資料串流探勘（Data Stream Mining）

到目前為此，有關資料串流探勘的研究包含了頻繁樣式探勘（frequent pattern mining）[5, 11, 12, 18, 21]、群聚分析 [2, 13]、分類 [9, 15, 22]、迴歸分析 [8] 以及變化偵測與探勘 [1, 10, 11] 等。在此計劃的前期規劃中，除了通用資料串流管理系統系統架構的開發之外，我們將主要針對頻繁樣式與其變化樣式－例如頻繁封閉樣式（frequent closed patterns）與最大頻繁樣式（maximal frequent patterns）等－作深入探討研究。

最早將頻繁樣式探勘與資料串流結合在一起的學者是 Manku 與 Motwani [18]。他們提出了第一個以 Apriori 特性 [3] 為基礎的一次資料串流掃描演算法 Lossy Counting，而所謂的 Apriori 特性是說：如果長度為 k 的樣式在資料庫中不是頻繁樣式時，則包含此長度為 k 的樣式而且長度為 k+1 的樣式也不可能會是頻繁樣式。Lossy Counting 使用一個特殊的陣列表示法來描述在雜湊樹（hash tree）中的遊歷順序，此雜湊樹 [3] 在非串流型資料的頻繁樣式探勘中是一種經常被使用到的樣式計數（pattern counting）技術。Teng 等人 [21] 提出一個以線性迴歸為基礎技術的探勘演算法，稱作 FTP-DS，這個頻繁樣式的探勘技術是以滑動窗模型當作其資料串流運作模式，而 Lossy Counting 則是以標的物模型為其基礎的資料串流模式。Chang 與 Lee 等人 [5] 則提出一個可適用於遞減模型中的頻繁樣式探勘演算法，稱作 estDec。在 estDec 中，每筆交易一開始的時候都被設定了一個加權值，然後隨著資料串流處理時間的前進，每筆交易被賦予的加權值會逐漸遞減。也就是說，愈早出現的資料的重要性愈低。Giannella 等人 [12] 提出一個以頻繁樣式樹（FP-tree；Frequent Pattern-tree）[14] 為基礎的頻繁樣式探勘演算法，稱為 FP-stream，這個演算法也是採用遞減模式。FP-stream 從記憶體中讀取一個區塊的資料後，先用 FP-growth 演算法找出在該區塊中的區域頻繁樣式，將這些樣式儲存在另一個樹狀結構中之後，再將先前已產生的頻繁樣式樹從記憶體中移除。

四. 研究方法

在此計劃中，我們將資料串流模型分成三大類：**標的物窗模型**、**滑動窗模型**以及**遞減窗模型**。我們將個別描述如下，但是在此之前，我們先說明在一般的情況下，處理資料串流的基本單位，也就是上述的「資料窗（window）」或簡稱「窗」：單筆資料 D_i 或是一個資料區塊（data block）B_i，其中一個交易區塊指的是在一個特定時間點（timestamp）所收集到的多筆交易所組成的串流處理單位（i = 1, 2, 3,…）。在此計劃中，我們所使用的基本單位是一個資料區塊。

(a) 標的物窗模型（landmark window model；LWM）：在這個模型中，串流資料是從某個特定的時間點 T_k 開始處理，一值處理到最近的一個時間點 T_N。也就是說，如果 $T_k = 1$，則表示是所處理的資料串流是全部的資料。我們可以用 WM[T_k] | $k \geq 1$ 來表示這個模型（WM = Window Model）。

(b) 滑動窗模型（sliding window model；SWM）：在此模型中，系統只對最近收到的 w 個資料窗有興趣；因此，假設現在的時間點是 T_N，則此模型可以表示成 WM[$T_N - w + 1 \rightarrow T_N$]。要注意的是在此模型中，最重要的問題就在於當新的資料窗進入系統時，要如何有效地將目前最早進入系統中的資料窗的相關資料移除。

(c) 遞減窗模型（damped window model；DWM）：在此模型中，我們是假設最近的資料窗的重要性是大過之前的資料窗。通常我們可以使用下列公式來設定，當然也可以使用別種公式（這是決定於此模型所使用的應用領域）。

$$avg_{new} = avg_{old} * p + T_N * (1-p)$$，其中 $0 < p < 1$

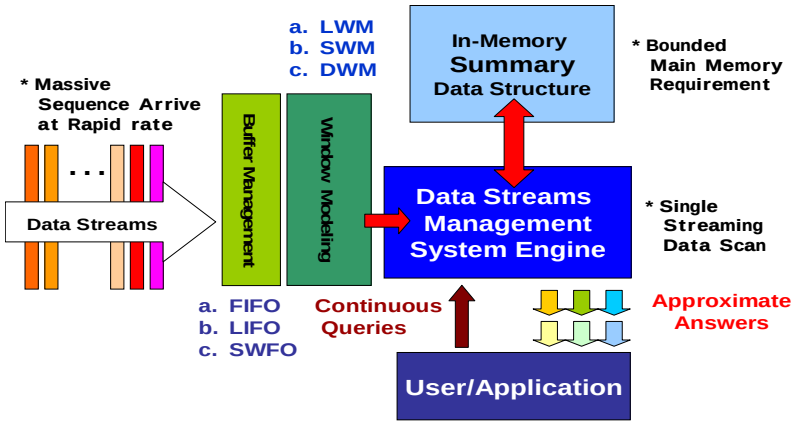
接下來，我們將討論為了要有效率地處理資料串流的產生速率，所設計的緩衝記憶體設計（buffer management）。在此計劃中，我們設計了三種機制來管理資料串流的分佈與產生速率。

第一種緩衝記憶體管理機制是**先進先出**（First-In First Out；FIFO）：此機制是假設最新產生的串流型資料的重要性是大過於先前產生但尚未處理到的串流型資料，而最簡單的實作方式就是使用一個佇列結構（queue）。

第二種緩衝記憶體管理機制是**先進後出**（Last-In First-Out；LIFO）：此模型是假設先前儲存在緩衝記憶體中但尚未被處理到的資料的重要性是大於目前的準備進入緩衝記憶體的資料。原因是因為這種做法可以降低系統運算上的負荷，避免過多的記憶體搬移操作所產生的系統問題。最簡單的實作方法就是使用一個堆疊結構（stack）。

第三種緩衝記憶體機制是**小型資料窗優先移除**（Small-Window First-Out；SWFO）：此機制是根據在每一個時間點進入緩衝記憶體中的資料窗的大小來設定移除的優先權，基本原則是資料量愈小的資料窗愈先被移除。這個模型是假設在單位時間內，如果產生的資料量愈大，就代表在這個時間點所產生的資料愈具有代表性，或是愈具有其特殊意義。

根據上述之資料窗模型與緩衝記憶體管理機制，我們可以得到如下圖所示之料串流管理系統的通用架構圖。



當使用者註冊了或輸入某個持續查詢（continuous query）之後，在資料串流管理系統中的查詢翻譯器會分析其查詢的內涵，並進一步地將此查詢最佳化的動作。而資料串流進入系統時，會先由串流緩衝記憶體接收，並根據不同的應用來採用不同的緩衝記憶體管理機制（目前系統中共有三種機制：FIFO、LIFO 以及 SWFO）。然後，再根據不同的應用需求，使用不同的串流型資料的處理模型（目前系統中共有三種資料窗模型：LWM、SWM 與 DWM）。對於某些具有超高速的資料串流而言，資料可能需要穿越緩衝記憶體，直接進入資料串流管理系統。在將經過查詢最佳化之後的查詢運算，送至系統的核心模組。接下來，系統核心將使用目前的串流資料，並根據系統已接受的持續查詢的功能需求，在記憶體中建立一個可儲存在主記憶體中的摘要資料結構（in-memory summary data structure），並隨時根據現在的記憶體剩餘容量來更新其摘要結構的內容，並將不適用的資料移除。最後，系統所使用的演算法與工作記憶體空間，配合某些持續性查詢的需求，在查詢執行運算之後，將定期將相關近似結果回傳給該使用者與相關應用程式。

五. 結果與討論

在目前為止的具體成果如下：

1. 資料串流與資料串流管理系統的定義、意義、相關方法以及應用領域的文獻蒐集與整理。
2. 三種主要資料串流模型（標的物窗模型、滑動窗模型與遞減窗模型）的架構討論與報告。
3. 串流應用系統的需求分析表設計。
4. 系統輸出近似結果的錯誤率上限之證明。
5. 摘要資料結構的空間使用率上限之證明。
6. 整合資料串流模型（在此是使用標的物窗模型）與串流變動偵測與探勘議題的研究論文發表 [16]。

參考文獻

1. C.C. Aggarwal. A Framework for Diagnosing Changes in Evolving Data Streams. In *ACM SIGMOD*, 2003.
2. C. C. Aggarwal, J. Han, J. Wang, and P.S. Yu. A Framework for Clustering Evolving Data Streams. In *Proc. of the 29th VLDB conference*, 2003.

3. R. Agrawal and R. Srikant. Fast Algorithms for Mining Association Rules. In *Conf. of the 20th VLDB conference*, pages 487-499, 1994.
4. B. Babcock, S. Babu, M. Datar, R. Motwani, and J. Widom. Models and Issues in Data Stream Systems. In *Proc. of the 2002 ACM Symposium on Principles of Database Systems* (*PODS* 2002), ACM Press, 2002.9.
5. J. Chang and W. Lee. Finding Recent Frequent Itemsets Adaptively over Online Data Streams. In *Proc. of the 9th ACM SIGKDD International Conference & Data Mining* (*KDD-2003*), 2003.
6. S. Chaudhuri and R. Motwani. On Sampling and Relational Operators. *Bulletin of the Technical Committee on Data Engineering*, Vol. 22, pp. 35-40, 1999.
7. J. Chen, D. J. DeWitt, F. Tian, and Y. Wang. NiagraCQ: A Scalable Continuous Query System for Internet Databases. *Proc. of the 2000 ACM SIGMOD Intl. Conf. on Management of Data*, pp. 379-390, 2000.
8. Y. Chen, G. Dong, J. Han, B. W. Wah, and J. Wang. Multi-Dimensional Regression Analysis of Time-Series Data Streams. In *Proceedings of 2002 International Conference on Very Large Data Bases* (*VLDB'02*), Hong Kong, China, Aug. 2002.11.
9. P. Domingos and G. Hulten. Mining High-Speed Data Streams. In *Proc. of the ACM Conference on Knowledge and Data Discovery* (*SIGKDD*), 2000.
10. G. Dong, J. Han, L.V.S. Lakshmanan, J. Pei, H. Wang and P.S. Yu. Online Mining of Changes from Data Streams: Research Problems and Preliminary Results. In P*roceedings of the 2003 ACM SIGMOD Workshop on Management and Processing of Data Streams*, June 2003.
11. V. Ganti, J. Gehrke, and R. Ramakrishnan. Mining Data Streams under Block Evolution. *SIGKDD Exploration*, 3(2):1-10, Jan. 2002.
12. C. Giannella, J. Han, J. Pei, X. Yan and P. S. Yu. Mining Frequent Patterns in Data Streams at Multiple Time Granularities. In *Proc. of the NSF Workshop on Next Generation Data Mining*, 2002
13. S. Guha, N. Mishra, R. Motwani, and L. O'Callaghan. Clustering Data Streams. In *Proc. of the Annual Symp. on Foundations of Computer Science* (*FOCS*), 2000.
14. J. Han, J. Pei, and Y. Yin. Mining Frequent Patterns without Candidate Generation. In *Proc. of 2000 ACM SIGMOD*, pages 1-12, 2000.
15. G. Hulten, L. Spencer, and P. Domingos. Mining Time-Changing Data Streams. In *Proc. of the ACM Conference on Knowledge Discovery and Data Mining* (*SIGKDD*), 2001.
16. H. F. Li and S. Y. Lee. Single-Pass Algorithms for Mining Frequency Change Patterns with Limited Space in Evolving Append-only and Dynamic Transaction

Data Streams. In *IEEE International Conference on e-Technology, e-Commerce and e-Service* (*EEE-04*), Mar. 2004.

17. L. Liu, C. Pu, and W. Tang. Continual Queries for Internet Scale Event-driven Information Delivery. *IEEE Trans. on Knowledge and Data Engineering*, Vol. 11, No. 4, pp. 583-590, 1999.
18. G. S. Manku and R. Motwani. Approximate Frequency Counts Over Data Streams. In *Proc. of the 28th VLDB conference*, 2002.
19. Stanford Stream Data Management (STREAM) Project. http://www-db.stanford.edu/stream
20. M. Sullivan. Tribeca: A Stream Database Manager for Network Traffic Analysis. *Proc. of the 1996 Intl. Conf. on Very Large Data Bases*, pp. 594, 1996.
21. W.G. Teng, M.-S. Chen, and P. S. Yu. A Regression-Based Temporal Pattern Mining Scheme for Data Streams. In *Proc. of the 29th VLDB Conference*, 2003.
22. H. Wang, W. Fan, P. S. Yu, and J. Han. Mining Concept-Drifting Data Streams using Ensemble Classifiers. In *ACM SIGKDD*, 2003.