Journal of Science and Engineering Technology, Vol. 2, No. 3, pp. 21-28 (2006)

在平行機器架構上擷取高頻項目組

陳垂呈
南台科技大學資訊管理系
台南縣永康市南台街一號

摘　要

在探勘關聯規則的過程中，首先必須找出滿足最小支持度的項目組，稱之爲高頻項目組，是探勘效率上的瓶頸所在，因此，如何提昇擷取高頻項目組的執行效率，即成爲探勘關聯規則最重要的研究主題之一。在本篇論文中，我們在兩個平行機器架構上，分別提出兩個演算法來擷取高頻項目組：一是在一個高度爲 n-1 之完整二元樹的網路架構下，執行次數只須 $m+n$-1 次的平行演算法，n 爲全部項目的個數，m 爲全部交易資料的數目；另一是在一個 n-維超立方體的網路架構下，執行次數只須 $m+n$ 次的平行演算法。而且，兩個演算法都只須讀取每筆交易資料兩次，即可擷取出全部的高頻項目組。

關鍵詞：資料探勘，關聯規則，高頻項目組，完整二元樹，超立方體

Generating Frequent Itemsets in Parallel Machines

CHUI-CHENG CHEN
Department of Information Management, Southern Taiwan University of Technology
1 Nan Tai St., YungKang, Tainan, Taiwan

ABSTRACT

The process of generating frequent itemsets is a bottleneck in data-mining association rules. Therefore, how to improve performance when generating frequent itemsets is one of the most important problems in formulating such rules. In this report, two parallel algorithms are presented for generating frequent itemsets in two parallel machines, respectively. The first task is to develop a parallel algorithm in a full binary tree of height n-1 for generating frequent itemsets, n being the number of all items and m the number of all transactions. The number of execution steps is represented by $m+n$-1. The second task is to develop a parallel algorithm in an n-dimensional hypercube for generating frequent itemsets. The number of execution steps is $m+n$. The algorithms generate frequent itemsets by scanning each transaction only twice.

***Key Words*:** data mining, association rules, frequent itemsets, full binary trees, hypercubes

一、簡介

從交易資料庫中找出關聯規則（association rules），已成爲資料探勘（data mining）最重要的研究主題之一，其目的是在發掘不同項目之間的關聯性，其可提供企業行銷決策的重要參考資訊。

探勘關聯規則首先由 Agrawal 等人所提出 [5]，之後相關的研究也相繼被發表出來 [2-4, 14, 17]，這些研究主要是以序列演算法（serial algorithms）來找出關聯規則，由於交易資料庫包含有大量的交易資料，產生關聯規則必須耗費很大的計算次數，因此有必要發展平行演算法（parallel algorithms）來改善這方面的問題，使用平行演算法來發掘關聯規則的相關研究，也相繼的被發表出來 [1, 6-9, 12-13, 16, 18-19]。

關聯規則的探勘過程中，首先必須擷取出高頻項目組（frequent itemsets），然後再計算高頻項目組可能形成的關聯規則。許多的研究已指出，擷取高頻項目組是探勘關聯規則執行效率上的瓶頸 [2, 4-5, 13, 14]，因此，如何提昇擷取高頻項目組的執行效率，即成爲探勘關聯規則最重要的研究主題之一。在本篇論文中，我們將在兩個平行網路架構上，提出兩個演算法來擷取高頻項目組；首先，在一個高度爲 n-1 之完整二元樹（full binary tree）的網路架構下，我們設計一個執行次數爲 $m+n$-1 的平行演算法，n 表示爲全部項目的個數，m 表示爲全部交易資料的個數，即可有效率地擷取出所有的高頻項目組；再者，在一個 n-維超立方體（hypercubes）的網路架構下，我們設計一個執行次數爲 $m+n$ 的平行演算法，即可有效率地擷取出所有的高頻項目組。在擷取高頻項目組的過程中，以上兩個演算法都只須讀取每筆交易資料兩次，即可擷取出全部的高頻項目組。

本篇論文的架構如下：下一節中，我們說明關聯規則與平行網路架構之相關名詞的定義；第三節中，我們說明在一個完整二元樹的網路架構下，提出一個平行演算法來擷取高頻項目組；第四節中，我們說明在一個超立方體的網路架構下，提出一個平行演算法來擷取高頻項目組；最後，在第五節中做一結論。

二、名詞定義

首先，我們說明一些名詞與關聯規則的定義如下：

- $I=\{i_1, i_2, ..., i_n\}$，是全部項目（items）的集合，共有 n 項。
- $T=\{T_1, T_2, \ldots, T_j, \ldots, T_m\}$，是交易資料庫中全部交易資料（transactions）的集合，共有 m 筆，其中 T_j 爲第 j 筆交易資料，$1\le j\le m$。
- T_j 由 n 位元（bits）所組成，其格式表示成 $T_j=[b_1, b_2, b_3, ..., b_f, ..., b_n]$，$b_f\in\{0, 1\}$，$1\le f\le n$，假如交易資料 T_j 中有出現第 f 項的項目，則 $b_f=1$，否則 $b_f=0$。例如，I={A, B, C, D, E} 表示全部項目的集合，若某一交易資料 T_j={ABD}，其轉換成位元的資料格式爲 [1, 1, 0, 1, 0]。
- $|T_j|$=第 j 筆交易資料所包含的項目個數。
- $mask_i$ 爲 i-位元擷取組合，其格式爲 $[b_1, b_2, b_3, ..., b_f, ..., b_n]$，$b_f\in\{0, 1\}$，$1\le f\le n$，表示 $mask_i$ 中有 i 個位元的值爲“1”，並可對應成由 i 個項目所組成的項目組。

項目組（itemsets）是由一個或以上的項目所組合而成，假設 X、Y 爲項目組，其中 X、$Y\subseteq I$ 且 $X\cap Y=\varnothing$，若 X 與 Y 之間有一關聯規則，則表示爲 $X\rightarrow Y$。有兩個參數 s 與 c 分別表示支持度（support）與信賴度（confidence），以用來決定關聯規則 $X\rightarrow Y$ 是否爲有效規則（strong rules），支持度 s 表示爲：在所有的交易資料集合中，同時包含有 $X\cup Y$ 的比率值，即 s=（同時包含有 $X\cup Y$ 的交易資料數量）/（總交易資料數量），而信賴度 c 表示爲：在包含有 X 的交易資料集合中，也同時包括有 Y 的比率值，即 c=（同時包含有 $X\cup Y$ 的交易資料數量）/（包含有 X 的交易資料數量）。擷取出來的關聯規則，其支持度與信賴度必須大於或等於所指定的最小支持度與最小信賴度，這樣的關聯規則才成立。

關聯規則的探勘過程主要分成以下兩個階段：第一階段先找出滿足最小支持度的項目組，稱之爲高頻項目組（frequent itemsets），若一個項目組包含有 k 個項目，稱之爲 k-項目組（k-itemsets），若某 k-項目組滿足最小支持度，稱之爲高頻 k-項目組（frequent k-itemsets）。第二階段就計算高頻項目組可形成的關聯規則，若滿足最小信賴度，則關聯規則成立。例如 ABC 爲高頻 3-項目組，假如關聯規則 AB→C 滿足最小信賴度，則此關聯規則成立。

在眾多的平行機器（parallel machines）中，完整二元樹（full binary trees）與超立方體（hypercubes）是兩種最常被使用的網路架構之一 [11]。對平行演算法而言，完整二元樹是一種很自然的計算結構，例如 “divide and conquer” 型態的演算法 [10]，其可以很有效率地解決各種應用問題，例如搜尋（searching）、排序（sorting）等。在超立方體的網路架構中，其具有較小的直徑（diameter）、規則性（regularity）、模組化（modularity）、高通訊頻寬（rich

bandwidth）、容錯度（fault tolerance）、容易繞徑（routing）[15]、及可模擬許多的平行計算架構，例如陣列（array）、二元樹（binary trees）等 [11]，使得超立方體成爲一個最受注意與討論的連結網路結構，並且已有利用超立方體架構成功地建置出平行電腦系統，如 Intel iPSC/860 及 NCUBE2。

在本篇論文中，我們將從交易資料庫中，分別在完整二元樹及超立方體的網路架構下，各設計出一個平行演算法來擷取高頻項目組。接下來，我們說明此兩種平行機器之網路架構的拓樸（topology）特徵：

一個高度爲 h 的完整二元樹其定義如下：根結點（root）位於第 0 層（0 level）、第 1 層有 2 個結點（nodes）、第 2 層有 4 個結點、依次於第 h-1 層有 2^{h-1} 個結點，第 h 層有 2^h 個結點，總結點數爲 2^{h+1}-1 個，如圖 1 爲一個高度爲 3 的完整二元樹，其有 15 個結點。因此，對一個高度爲 h 之完整二元樹的網路架構，其資料從根結點往下傳送，只須執行 h 次，即可傳送到全部的結點中。

一個 n-維超立方體（n-dimensional hypercube），其有 2^n 個結點數，每一結點分別以 {0, 1, 2, …, 2^n-1} 的二進位數（binary numbers）來標號。在超立方體中，假如兩個結點的標號有一個位元相異，則此兩結點之間有一條線（edges）相連，兩個結點相異的位元數目，稱爲 *Hamming distance*，圖 2 爲一個 3-維超立方體與各結點的二進位數標號。因此，對一個 n-維超立方體的網路架構，其資料從任一結點往外傳送，只須執行 n 次，即可傳送到全部的結點中。

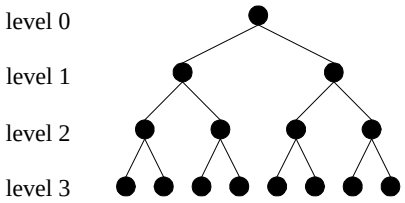

圖 1. 高度爲 3 的完整二元樹

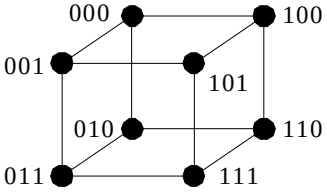

圖 2. 3-維超立方體

三、在完整二元樹之網路架構下擷取高頻項目組

在這一章節中，我們在一個完整二元樹的網路架構下，提出一個平行演算法來擷取高頻項目組。此章節共分爲兩小節如下：第一小節中，我們在一個完整二元樹的網路架構下，提出一個平行演算法來擷取高頻項目組；第二小節中，我們以一實例來說明擷取高頻項目組的過程。

（一）擷取高頻項目組之平行演算法

首先，我們將交易資料轉換成位元的資料格式，假如項目有出現在交易資料中，則相對位元設定爲 “1”，否則設定爲 “0”。因此，讀取之後的每筆交易資料都是以 n 位元的格式來表示，然後依次讀取交易資料來處理。對某一交易資料所包含的項目而言，我們必須計算出其可能形成的所有子項目組，並且加以累積各子項目組出現的次數。我們使用 *or* 布林運算（如表 1）和 *xor* 布林運算（如表 2）、及利用位元擷取組合，來判斷交易資料是否包含某一子項目組。我們執行以下公式來判斷某一 $mask_i$，是否包含於某一交易資料 T_j 中，$1 \le i \le |T_j|$。

$$mask_i \ or \ T_j \ xor \ T_j \qquad (1)$$

假如結果爲 n 個位元都等於 “0”，則表示 $mask_i \subseteq T_j$。

例如，T_1=[0, 1, 1, 0, 1]，$mask_2$=[0, 1, 1, 0, 0]，經由公式（1）的計算，其結果等於 [0, 0, 0, 0, 0]，表示 $mask_2 \subseteq T_1$。

接下來，我們說明要處理一筆交易資料必須產生多少的位元擷取組合。假設一交易資料 T_j 包含有 k 個項目，$1 \le k \le n$，轉換成 n 位元的資料格式，會有 k 個位元爲 “1”，其包含的子項目組類型有 1-項目組、2-項目組、．．．．、k-項目組，因此，我們必須產生的位元擷取組合爲 $mask_1$, $mask_2$, …, $mask_i$, …, $mask_k$，$1 \le i \le k$。由於每個位元擷取組合是由 n 位元所組成，因此各 $mask_i$ 的擷取組合數目爲：

表 1. *or* 布林運算

or	0	0
0	0	1
1	1	1

表 2. *xor* 布林運算

xor	0	1
0	0	1
0	1	0

$mask_1$：有 n 個組合數目。

$mask_2$：有 $n(n\text{-}1)/2$ 個組合數目。

$mask_3$：有 $n(n\text{-}1)(n\text{-}2)/(2\times3)$個組合數目。

．．．

$mask_{k-1}$：有 $n(n\text{-}1)(n\text{-}2)\ldots(n\text{-}k\text{+}2)/(k\text{-}1)! = n!/((n\text{-}k\text{+}1)! \times(k\text{-}1)!)$ 個組合數目。

$mask_k$：有 $n(n\text{-}1)(n\text{-}2)\ldots(n\text{-}k\text{+}1)/k! = n!/((n\text{-}k)! \times k!)$ 個組合數目。

因此，總數爲：

$S(T_j)$ =$mask_1$的組合數目+$mask_2$的組合數目+ … +$mask_k$的組合數目= $\sum_{j=1}^{k} n!/((n-j)!\times j!)$

對一個包含有 k 個項目的交易資料 T_j 而言，我們必須產生 $S(T_j)$ 個位元擷取組合，並且對每個位元擷取組合執行公式（1）的布林運算，以判斷位元擷取組合是否包含於交易資料 T_j 中。因爲一筆交易資料可包含的項目個數最多爲 n 個，若一個結點負責一個位元擷取組合，則必須有 $S_n= \sum_{j=1}^{n} n!/((n-j)!\times j!)$ 個結點，才可處理任一筆包含有不同項目個數的交易資料。

以下我們將證明 S_n 個結點恰好可建構出一個高度爲 n-1 之完整二元樹所需要的結點數。

定理 1：全部項目個數爲 n，則產生的位元擷取組合數目，其恰好可建構出一個高度爲 n-1 之完整二元樹所需要的結點數。

證明：全部項目個數爲 n，任一筆交易資料最多可能包含有 n 個項目，因此，我們必須產生的位元擷取組合爲 $mask_1$, $mask_2$, …, $mask_n$，且由 n 個位元的格式來表示，各位元擷取組合可對應到 “111…1” 到 “000…1”，共有 2^n-1 個組合數目。而一個高度爲 n-1 之完整二元樹的結點數爲 2^n-1 個，故可得證在全部項目個數爲 n，產生的位元擷取組合數目，其恰好可建構出一個高度爲 n-1 之完整二元樹所需要的結點數。 □

在一個完整二元樹之網路架構中的每一個結點，其負責計算之位元擷取組合可分配如下：我們從根結點開始，根據由上而下、由左到右的次序方式，分別以 1, 2, 3, …, S_n 數字來標號各結點，如圖 3。若位元擷取組合 $mask_i$ 等於結點標號的二進位數，則結點負責計算位元擷取組合 $mask_i$ 的出現次數，在圖 4 中顯示高度爲 2 之完整二元樹中各結點所負責計算的位元擷取組合。

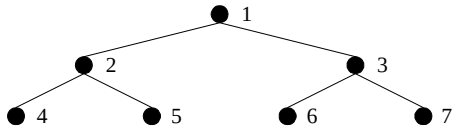

圖 3. 高度爲 2 之完整二元樹各結點的標號

以下我們說明在一個完整二元樹的網路架構下，要擷取所有高頻項目組必須執行的次數。

定理 2：在一個高度爲 n-1 之完整二元樹的網路架構下，執行次數只須 m+n-1 次，即可擷取出所有的高頻項目組，n 爲全部項目的個數，m 爲全部交易資料的數目。

證明：在一個完整二元樹的網路架構下，假設每一個結點擁有獨自的計數器，且初始值爲 0，由根結點依次讀取一筆交易資料 T_j，$1\le i\le m$，然後平行的往下層傳送。每一個結點只處理一種位元擷取組合 $mask_i$，$1\le i\le|T_j|$，利用公式(1)，可快速地計算出所負責的位元擷取組合是否包含於 T_j 中，假如成立，則結點本身的計數器就加 1。由定理 1 得知，必須在一個高度爲 n-1 之完整二元樹的網路架構下，才能處理交易資料 T_j 的每一種位元擷取組合，其執行次數爲：

完整二元樹的高度+1=n-1+1=n

即可計算出交易資料 T_j 形成的所有子項目組。

我們利用一個高度爲 n-1 之完整二元樹的網路架構，處理第一筆交易資料須執行 n 次，即可計算出此交易資料所有的位元擷取組合，然後每執行完一次，即可完成一筆交易資料的計算。因爲還剩餘有 m-1 筆交易資料，所以共須執行 n+m-1 次，即可處理完全部的交易資料，且在每一個結點的計數器中，計算出各自負責之位元擷取組合的出現次數。

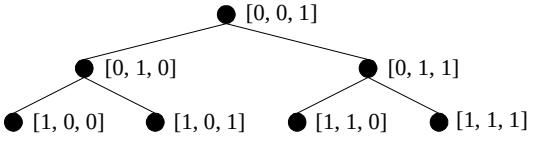

圖 4. 結點所負責計算之位元擷取組合

當處理完所有的交易資料之後，再檢查各個結點的計數器，其值是否大於或等於最小支持度，假如成立，則其位元擷取組合所對應的項目組即爲高頻項目組。故可得證本定理。 □

接下來，我們根據第二節中的定義，說明在一個高度爲 n-1 之完全二元樹的網路架構下，擷取高頻項目組的探勘過程如下：

（1）讀取交易資料並轉換成位元的資料格式 T_j，$1 \le j \le m$。

（2）設定完全二元樹之網路架構中每一結點所負責的位元擷取組合 $mask_i$，並設定結點中之計數器歸零。

（3）從完全二元樹之網路架構的根結點開始，依序讀取交易資料，然後平行地將交易資料往下層的結點傳送，並在每一結點中執行式（1）計算，若結果爲全部位元都等於 “0”，表示 $mask_i \subseteq T_j$，則負責之結點的計數器其計數次數增加 1。

（4）檢查每一結點之計數器所累積的次數，若滿足最小支持度，則其位元擷取組合 $mask_i$ 所對應的項目組爲高頻項目組。

（二）實例說明

我們以表 3 之交易資料庫爲例來說明高頻項目組的擷取過程。假設 I={A, B, C, D, E} 爲全部項目的集合，T={T_1, T_2, T_3, T_4} 爲全部交易資料的集合，最小支持度爲 40%（即最小支持數量爲 1.6）。

根據前一小節所描述的演算法，其擷取高頻項目組的過程如下：因爲全部項目個數爲 5，故需要一個高度爲 4 之完整二元樹的網路架構，如圖 5，其結點數目爲 2^5-1=31，我們在結點旁邊標示其所負責的位元擷取組合及計數器，其中位元擷取組合以相對的十進位數表示之，例如 7, 0 表示擷取組合爲 [0, 0, 1, 1, 1] 及 0 次。首先將各交易資料轉換成位元的資料格式爲 T_1=[1, 0, 1, 1, 0]、T_2=[0, 1, 1, 0, 1]、T_3=[1, 1, 1, 0, 1]、T_4=[0, 1, 0, 0, 1]。假設先讀取交易資料 T_1，要處理完交易資料 T_1 所須要的執行次數爲：此完整二元樹的高度

表 3. 交易資料庫

交易資料編號	交易項目
T_1	ACD
T_2	BCE
T_3	ABCE
T_4	BE

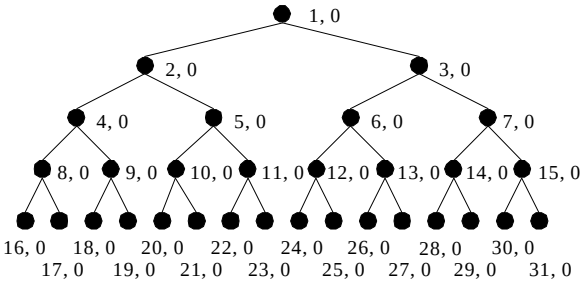

圖 5. 結點旁邊的標示爲擷取組合及計數器的次數

+1=4+1=5 次。因爲從根結點依次地讀取交易資料，且平行地往下層傳送，因此剩餘的 3 筆交易資料，只須執行 3 次即可處理完。因此共須執行 5+3=8 次，即可執行完所需要的全部計算，其結果如圖 6。

再檢查每個結點之計數器的次數，滿足最小支持度者有 [0, 0, 0, 0, 1], [0, 0, 1, 0, 0], [0, 0, 1, 0, 1], [0, 1, 0, 0, 0], [0, 1, 0, 0, 1], [0, 1, 1, 0, 0], [0, 1, 1, 0, 1], [1, 0, 0, 0, 0] 及 [1, 0, 1, 0, 0]，則其對應的項目組即爲高頻項目組。其結果如下：

高頻 1-項目組：A, B, C, E。

高頻 2-項目組：AC, BC, BE, CE。

高頻 3-項目組：BCE。

四、在超立方體之網路架構下擷取高頻項目組

在這一章節中，我們將在一個超立方體的網路架構下，提出一個平行演算法來擷取高頻項目組。此章節共分爲兩小節如下：第一小節中，我們在一個超立方體的網路架構下，提出一個平行演算法來擷取高頻項目組；第二小節中，我們以一實例來說明擷取的過程。

（一）擷取高頻項目組之平行演算法

如同前一章節所描述，我們先將各交易資料轉換成位元的資料格式，轉換之後的每筆交易資料都是以 n 位元的格式

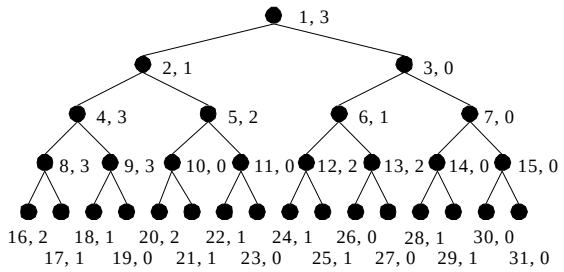

圖 6. 結點之計數器的次數

來表示。然後，依次讀取交易資料來處理，對某一交易資料 T_j 所包含的子項目組而言，可執行公式（1）來判斷某一 i-位元擷取組合，$mask_i$，是否包含於交易資料 T_j 中，$1 \le i \le |T_j|$。

在一個超立方體之網路架構中的每一個結點，其負責計算之位元擷取組合可分配如下：我們可根據超立方體中各結點本身的標號，其定義如第二節中所描述，若位元擷取組合 $mask_i$ 等於結點的標號，則結點負責計算位元擷取組合 $mask_i$ 的出現次數，在圖 7 中，各結點的標號比照圖 2，顯示一個 3-維超立方體中各結點所負責計算的位元擷取組合。

以下我們說明在一個超立方體的網路架構下，要擷取所有高頻項目組必須執行的次數。

定理 3：在一個 n-維超立方體的網路架構下，執行次數只須 $m+n$ 次，即可擷取出所有的高頻項目組，n 爲全部項目的個數，m 爲全部交易資料的數目。

證明：對 n 位元的交易資料而言，我們必須產生 2^n-1 個位元擷取組合，即分別爲 {1, 2, 3, …, 2^n-1} 以二進位數表示其位元組合，才能處理任一筆交易資料所包含的子項目組。在一個 n-維超立方體的網路架構下，共有 2^n 個結點，假設每一個結點負責其二進位數標號所對應的位元擷取組合，並且擁有獨自的計數器，初始值爲 0。

由任一結點依次讀取每筆交易資料 T_j，$1 \le i \le m$，然後平行的往外層傳送，每一個結點只處理一種位元擷取組合，即結點本身的二進位數的標號，可快速地計算出所負責的位元擷取組合是否包含於 T_j 中，假如成立，則結點本身的計數器就加 1。因此，在一個 n-維超立方體中，除了標號爲 n 個位元都爲 “0” 的結點之外，其結點所對應的位元擷取組合，可處理交易資料 T_j 所有的子項目組，其執行次數爲：

n-維超立方體最長的 *hamming distance*+1=n+1

即可計算出交易資料 T_j 包含的所有子項目組。

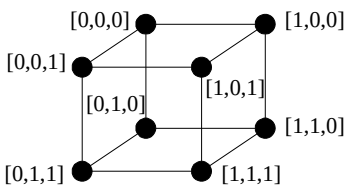

圖 7. 結點負責計算的位元擷取組合

經由以上的說明，我們在一個 n-維超立方體的網路架構下，處理一筆交易資料只須執行 n+1 次，即可計算出此交易資料包含的所有子項目組。若交易資料庫共有 m 筆交易資料，除了第一筆交易資料須要執行 n+1 次，其餘每執行一次，即可處理完一筆交易資料。因此，要處理完全部的交易資料，共須執行 $m+n$ 次，並且在每一個結點的計數器中，計算出各自負責之位元擷取組合的出現次數。

當處理完所有的交易資料之後，再檢查各結點的計數器，假如其值大於或等於最小支持度，則所對應的項目組即爲高頻項目組，故可得證本定理。 □

接下來，我們根據第二節中的定義，說明在一個 n-維超立方體的網路架構下，擷取高頻項目組的探勘過程如下：

（1）讀取交易資料並轉換成位元的資料格式 T_j，$1 \le j \le m$。

（2）根據結點之標號，來設定超立方體之網路架構中每一結點所負責的位元擷取組合 $mask_i$，並設定結點中之計數器歸零。

（3）從超立方體之網路架構中標號二進位數爲 “0” 的結點開始，依序讀取交易資料，然後平行地將交易資料往外層的結點傳送，並在每一結點中執行式（1）計算，若結果爲全部位元都等於 “0”，表示 $mask_i \subseteq T_j$，則負責之結點的計數器其計數次數增加 1。

（4）檢查每一結點之計數器所累積的次數，若滿足最小支持度，則其位元擷取組合 $mask_i$ 所對應的項目組爲高頻項目組。

（二）實例說明

我們仍以第三節的表 3 爲例，因爲全部項目共有 5 個，故需要一個 5-維超立方體的網路架構，如圖 8，我們在結點旁邊標示其標號，以相對的十進位數表示之，同時也相對表示其所負責的位元擷取組合，例如 7 表示 [00111]。

首先將各交易資料轉換成位元的資料格式，然後從任一結點依次地讀取交易資料，且平行的往外傳送，要處理完第一筆交易資料須執行 n+1=5+1=6 次，之後，每執行一次即可完成一筆交易資料的計算，其餘 3 筆交易資料，只須執行 3 次即可處理完。因此，總共須執行 6+3=9 次，即可執行完所須要的全部計算，其結果如圖 9，我們在結點旁邊標示其所負責的位元擷取組合及計數器的次數，例如 7, 0 表示結點爲 [00111] 與計數 0 次。

再檢查每個結點之計數器的次數，滿足最小支持度者有 [0, 0, 0, 0, 1], [0, 0, 1, 0, 0], [0, 0, 1, 0, 1], [0, 1, 0, 0, 0], [0, 1,

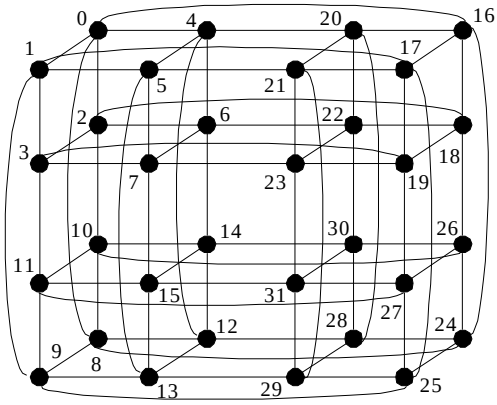

圖 8. 各結點的標號

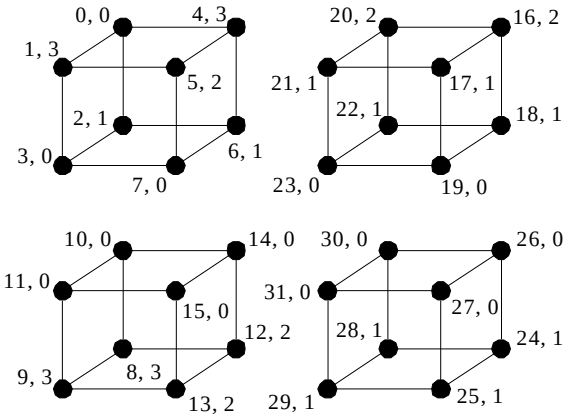

圖 9. 結點旁邊分別標示結點的標號與計數器的次數

0, 0, 1], [0, 1, 1, 0, 0], [0, 1, 1, 0, 1], [1, 0, 0, 0, 0] 及 [1, 0, 1, 0, 0]，則其對應的項目組即爲高頻項目組。其結果如下：

高頻 1-項目組：A, B, C, E。

高頻 2-項目組：AC, BC, BE, CE。

高頻 3-項目組：BCE。

五、結論與未來研究

如何從大量的交易資料中找出項目之間的關聯規則，已成爲資料探勘最重的研究主題之一，在探勘關聯規則過程中，必須先擷取出的高頻項目組，則是執行效率上的瓶頸。在本篇論文中，我們在一個高度爲 *n*-1 之完整二元樹的網路架構下、及在一個 *n*-維超立方體的網路架構下，分別各提出一個平行演算法來擷取出所有的高頻項目組。假設 *n* 爲全部項目的個數，*m* 爲全部交易資料的數目，此兩個平行演算法與傳統序列演算法之間的差異如表 4 所示。

表 4. 平行演算法與序列演算法之效能評估

演算法類別	平行演算法		序列演算法
	高度爲 *n*-1 之完整二元樹	*n*-維超立方體	
結點數	2^n-1	2^n	1
執行次數	*m*+*n*-1	*m*+*n*	(2^n-1)×*m*
閒置結點數	0	1	0
每一筆交易資料讀取次數	2	2	1

目前，本研究僅就在完整二元樹、及在超立方體之網路架構下，分別設計平行演算法來擷取高頻項目組做探討，並分析其執行的次數。對於未來繼續從事之相關研究有：

（1）就交易資料庫的分割（partition）考量，探討擷取高頻項目組之平行演算法的可行性。

（2）當結點發生錯誤（fault）時，探討如何重建（reconfiguration）網路架構來擷取高頻項目組。

（3）當結點數目小於擷取組合數目時，探討在負載平衡下（load balance）擷取高頻項目組之平行演算法的可行性。

（4）在不同的平行機器中，探討擷取高頻項目組之平行演算法的可行性。

參考文獻

1. Agrawal, R. and J. C. Shafer (1996) Parallel mining of association rules. *IEEE Transactions on Knowledge and Data Engineering*, 8, 962-969.
2. Agrawal, R. and R. Srikant (1994) Fast algorithms for mining association rules in large database. Proceedings of the 20th International Conference on Very Large Data Bases, Santiago de Chile, Chile.
3. Agarwal, R. C., C. C. Aggarwal and V. V. V. Prasad (2000) Depth first generation of long patterns. Proceedings of the 6th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, Boston, MA.
4. Agarwal, R. C., C. C. Aggarwal and V. V. V. Prasad (2001) A tree projection algorithm for generation of frequent item sets. *Journal of Parallel and Distributed Computing*, 61, 350-371.
5. Agrawal, R., T. Imielinski and A. Swami (1993) Mining association rules between sets of items in very large database. Proceedings of the ACM SIGMOD Conference on Management of Data, Washington, D.C.

6. Cerin, C., J. S. Gay, G. Le Mahec and M. Koskas (2004) Efficient data-structures and parallel algorithms for association rules discovery. Proceedings of the 15th Mexican International Conference in Computer Science, Colima, Mexico.
7. Cheung, D. W., K. Hu and S. Xia (1998) Asynchronous parallel algorithm for mining association rules on a shared-memory multi-processors. Proceedings of the 10th Annual ACM Symposium on Parallel Algorithms and Architectures (SPAA), Puerto Vallarta, Mexico.
8. Chung, S. M. and C. Luo (2003) Parallel mining of maximal frequent itemsets from databases. Proceedings of 15th IEEE International Conference on Tools with Artificial Intelligence, Sacramento, California, CA.
9. Han, E. H. and G. Karypis (2000) Scalable parallel data mining for association rules. *IEEE Transactions on Knowledge and Data Engineering*, 12(3), 337-352.
10. Horowitz, E. and A. Zorat (1983) Divide-and-conquer for parallel processing. *IEEE Transactions on Computers*, C-32, 582-585.
11. Leighton, T. (1992) *Introduction to Parallel Algorithms and Architectures: Arrays, Trees, Hypercubes*, Morgan Kaufmann, Reading, MA.
12. Meng, X. P., J. Qian and X. Qi (2003) Parallel mining association rules with bit string array in large database. Proceedings of the 2nd International Conference on Machine Learning and Cybernetics, Xi'an, China.
13. Park, J. S., M. S. Chen and P. S. Yu (1995) Efficient parallel mining for association rules. Proceedings of the 4th International Conference on Information and Knowledge Management, Baltimore, Maryland.
14. Park, J. S., M. S. Chen and P. S. Yu (1997) Using a hash-based method with transaction trimming for mining association rules. *IEEE Transactions on Knowledge and Data Engineering*, 9, 813-825.
15. Saad, Y. and M. H. Schultz (1988) Topological properties of hypercubes. *IEEE Transactions on Computers*, 37(7) 867-872.
16. Veloso, A., W. J. Meira and S. Parthasarathy (2003) New parallel algorithms for frequent itemset mining in very large databases. Proceedings of 15th Symposium on Computer Architecture and High Performance Computing, São Paulo, SP, Brazil.
17. Wur, S. Y. and Y. Leu (1999) An effective Boolean algorithm for mining association rules in large databases. Proceedings of the Sixth International Conference on Database Systems for Advanced Applications (DASFAA), Hsinchu, Taiwan.
18. Zaiane, O. R., M. El-Hajj and P. Lu (2001) Fast parallel association rule mining without candidacy generation. Proceedings of the 2001 IEEE International Conference on Data Mining, San Jose, CA.
19. Zaki, M. J., M. Ogihara, S. Parthasarathy and W. Li (1996) Parallel data mining for association rules on shared-memory multi-processors. Proceedings of the 1996 ACM/IEEE Conference on Supercomputing, Pittsburgh, Pennsylvania, PA.

收件：94.07.29　修正：94.09.23　接受：94.11.07