Journal of Science and Engineering Technology, Vol. 2, No. 4, pp. 83-91 (2006)

有效率探勘關聯規則之演算法

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

在大型資料庫中利用關聯規則（association rules）表示產品項目之間的關聯性，是資料探勘（data mining）最常使用的技術之一。在本篇論文中，我們提出兩個有效率的演算法分別來擷取關聯規則及包含有項目數量的關聯規則：一是以布林運算爲基礎，根據 Apriori 演算法的執行步驟來擷取關聯規則；二是以布林運算爲基礎，修改陳彥良等所提出之 MQA-1 演算法，從包裹資料庫中擷取包含有項目數量的關聯規則。從實驗評估中顯示，我們所提出之演算法的執行效率分別優於由 Wur 與 Leu 所提出之演算法及 MQA-1 演算法。

關鍵詞：資料探勘，關聯規則，布林運算

Efficient Algorithms for Mining Association Rules

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

ABSTRACT

The association rule is one of the most popular technologies to find the associations between items in large databases. In this paper, we present two efficient algorithms for mining association rules and quantitative association rules, respectively. One is to mine association rules with the Boolean computation according to the Apriori algorithm. The other is to mine association rules including the quantities of items with the Boolean computation by modifying the MQA-1 algorithm proposed by Chen et al. in bag databases. The experiments show that the performances of both algorithms are faster than the algorithm proposed by Wur and Leu's algorithm and the MQA-1 algorithm, respectively.

***Key Words*:** data mining, association rules, boolean computations

一、簡介

近幾年來，資料探勘（data mining）技術已廣泛地運用在資料庫領域中，其目的是從大量的交易資料中，找出潛在有用的資訊與知識，以支援企業的行銷決策。經由資料探勘可擷取出產品項目（items）之間的關聯性，並以關聯規則（association rules）的形式表示之，在眾多探勘關聯規則的方法中，由 Agrawal 與 Srikant 所提出的 Apriori 演算法 [3] 是最具代表性的方法之一，而 Wur 與 Leu 提出一個布林演算法（Boolean algorithm）[17]，並且證明其演算法的執行效率優於 Apriori 演算法。在本篇論文中，我們將根據 Apriori 演算法之執行步驟，提出一個以布林運算爲基礎的演算法來探勘關聯規則，並實驗顯示將可提升 Wur 與 Leu 所提出之布林演算法的執行效率。

傳統探勘關聯規則往往都只是記錄產品項目是否購買，而並未考量項目的購買數量，但在現實的交易資料中，都會記錄有購買的項目與其數量，對於一資料庫其交易資料若記錄有包含項目之購買數量，即稱之爲包裹資料庫（bag databases）。因此，若能從包裹資料庫中找出包含有項目數量的關聯規則，對企業在考量項目該以多少的項目數量來搭配做銷售時，將可提供相當有用的參考資訊。陳彥良等曾經提出 MQA-1（mining quantitative association rules with 1 comparator）演算法來擷取包含有項目數量的關聯規則 [1]，在本篇論文中，我們將修改 MQA-1 演算法，提出一個以布林運算爲基礎的演算法，擷取包含有項目數量的關聯規則。

本篇論文的架構如下：下一節中，我們介紹探勘關聯規則的相關研究；第三節中，我們根據 Apriori 演算法的執行步驟，提出一個以布林運算爲基礎的方法來探勘關聯規則；第四節中，我們修改 MQA-1 演算法，提出一個以布林運算爲基礎的方法，探勘包含有項目數量的關聯規則；第五節中，我們實驗評估所提出之演算法的執行效能；最後，我們在第六節中做一結論。

二、相關研究

從大量的交易資料中探勘產品項目之間的關聯規則，由 Agrawal et al. 首先提出 [2]，之後相關的研究也相繼的被發表出來 [3-10, 12, 15-17]。在探勘關聯規則的方法中又以 Apriori 演算法 [3] 最具代表性，而後續的研究中，分別提出許多的不同的資料結構或是資料儲存方式 [4-10, 15, 16]，並搭配設計對應的演算法，以提升探勘關聯規則的執行效能。在以上相關研究中，所提出之演算法都是以字元運算的方式來進行探勘計算 [3-10, 12, 15-16]，而 Wur 與 Leu 提出一個布林運算爲基礎的方法來探勘關聯規則 [17]，並實驗顯示其執行效率優於 Apriori 演算法。接下來，我們說明關聯規則的相關定義及 Apriori 演算法的執行步驟。

假設 I 是交易資料庫中所有項目的集合，每一筆交易資料 T 是由一些項目所形成的集合，且 $T \subseteq I$，在項目集合 X 與 Y 之間有一關聯規則被表示成 $X \rightarrow Y$，其中 X 稱之爲前項目組（antecedent），Y 稱之爲後項目組（consequent），X、$Y \subseteq I$ 且 $X \cap Y = \varnothing$。有兩個參數 s 與 c 分別爲支持度（support）與信賴度（confidence），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，如此的關聯規則才有意義。

關聯規則的探勘過程主要分成兩個階段：在第一階段中，先找出滿足最小支持度的項目組，這些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）。在第二階段中，以最小信賴度爲條件，計算高頻項目組所形成的關聯規則，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，A、B、C$\in I$，若關聯規則 AB$\rightarrow$C 滿足最小信賴度，則此關聯規則成立。

以下我們說明 Apriori 演算法探勘關聯規則的步驟 [3]：

1. 找出高頻 (k-1)-項目組，k>1。
2. 由步驟 1 中找出任兩個有 k-2 項目相同的高頻 (k-1)-項目組，組合成 k-項目組。
3. 判斷由步驟 2 所找出的 k-項目組，其所有包括的 (k-1)-項目組之子集合是否都出現在（1）中，假如成立就保留此 k-項目組，否則就刪除。
4. 再檢查由步驟 3 所擷取的 k-項目組是否滿足最小支持度，假如符合就成爲高頻 k-項目組，否則就刪除。
5. 計算高頻 k-項目組所形成的關聯規則，若滿足最小信賴

度，則關聯規則成立。

6. 跳至步驟 1 找高頻 (*k*+1)-項目組，直到無法產生高頻項目組爲止。

以上演算法所擷取出的高頻項目組，都視每一項目其數量爲 1，因此所找出的關聯規則，其項目所包含的數量也就都爲 1。但是，在現實交易資料中都會記錄購買的項目與其數量，對於一交易資料庫其包含之交易資料記錄有項目的購買數量，稱之爲包裹資料庫 [1]，目前已有許多相關研究在探討如何擷取包含有項目數量的關聯規則 [1, 11, 13, 14]。其中陳彥良等 [1] 曾經修改 Apriori 演算法，提出一個 MQA-1 演算法，從包裹資料庫中找出包含有項目數量的關聯規則，例如一關聯規則 2A→3B 成立，表示購買 2 個單位的 A，也會同時購買 3 個單位的 B。

MQA-1 演算法在探勘包含有項目數量之關聯規則的過程中，其設定以下的條件，做爲判斷某一項目組 *X* 是否被包含在交易資料 *T* 的準則：

X 中的任一項目皆出現在 *T* 中，且在 *T* 中之購買數量必須大於或等於在 *X* 中的購買數量。

例如若 *X*={2A3B}={AABBB}，*T*=3A3B={AAABBB}，則 *T* 包含 *X*。以下我們說明 MQA-1 演算法探勘包含有項目數量之關聯規則的步驟：

1. 找出所有 1-項目組的支持度，若滿足最小支持度者，即成爲高頻 1-項目組，若爲∅，則停止執行。
2. 找出高頻 (*k*-1) 項目組，*k* > 1。
3. 由步驟 2 中找出任兩個有 *k*-2 項目組相同的高頻 (*k*-1)-項目組，包括高頻 (*k*-1)-項目組與本身，組合成 *k*-項目組。
4. 判斷由步驟 3 所找出的 *k*-項目組，其所有包含的 (*k*-1)-項目組之子集合，是否都出現在步驟 2 中，假如成立就保留，否則就刪除。
5. 再檢查由步驟 4 所找出的 *k*-項目組，若滿足最小支持度，即成爲高頻 *k*-項目組，否則就刪除。
6. 對高頻 *k*-項目組，列出所有可能的關聯規則，若滿足最小信賴度者，即成爲有效的關聯規則。
7. 跳至步驟 2 找高頻 (*k*+1)-項目組，直到無法產生高頻項目組爲止。

在本篇論文中，我們將提出兩個以布林運算爲基礎的演算法，分別來探勘關聯規則及從包裹資料庫中探勘包含有項目數量的關聯規則。

三、探勘關聯規則

Wur 與 Leu 曾經提出一個以布林運算爲基礎的方法，稱之爲布林演算法，來探勘關聯規則 [17]，並證明其演算法的執行效率優於 Apriori 演算法。在此一章節中，我們將以布林運算爲基礎，根據 Apriori 演算法的執行步驟，提出一個有效率的演算法來探勘關聯規則。此章節共分爲兩小節：第一小節中，我們設計一個以布林運算爲基礎的演算法來探勘關聯規則；第二小節中，我們以一實例來說明演算法的探勘過程。

（一）以布林運算爲基礎探勘關聯規則之演算法

在說明如何利用布林運算爲基礎來探勘關聯規則之前，我們首先定義以下名詞：

- $I=\{i_1, i_2, ..., i_n\}$，是全部項目（items）的集合，共有 *n* 項。
- $T=\{T_1, T_2, ..., T_j, ..., T_m\}$，爲交易資料庫（transaction database）中全部消費者之交易資料的集合，共 *m* 筆，其中 T_j 爲消費者 *j* 的交易資料，$1 \le j \le m$。
- T_j 是由 *n* 位元（bits）所組成，其格式表示成 $T_j=[b_1, b_2, b_3, \ldots, b_f, \ldots, b_n]$，$b_f \in \{0, 1\}$，$1 \le f \le n$，假如 T_j 中有出現第 *f* 項的項目，則 $b_f=1$，否則 $b_f=0$。
- $Itemset_k$ 表示某一 *k*-項目組，其格式由 *n* 位元所組成，表示成 $Itemset_k=[b_1, b_2, b_3, \ldots, b_f, \ldots, b_n]$，$b_f \in \{0, 1\}$，$1 \le f \le n$，且有 *k* 個項目其值爲 “1”，$k \ge 0$。
- F_k 表示某一高頻 *k*-項目組，其格式定義與 $Itemset_k$ 相同，$F_k=[b_1, b_2, b_3, \ldots, b_n]$。

我們利用 *or* 布林運算，如表 1，可以很有效率地計算出兩項目組之間位元的聯集，而利用 *xor* 布林運算，如表 2，可以很有效率地計算出兩項目組之間相異的位元。

表 1. *or* 布林運算

or	0	1
0	0	1
1	1	1

表 2. *xor* 布林運算

xor	0	1
0	0	1
1	1	0

首先，我們將每一筆交易資料轉換成位元的資料格式，然後根據 Apriori 演算法的執行步驟、及利用 *or* 和 *xor* 等布林運算，設計一個方法，稱之爲 Boolean-Apriori 演算法，來探勘關聯規則，其探勘過程說明如下：

1. 將交易資料轉換成位元的資料格式。
2. 找出 F_{k-1}，$k > 1$。
3. 任意兩個 F_{k-1} 做 *or* 布林運算，假如結果爲 $Itemset_k$，即有 k 個項目其值爲 “1”，且非重複者，就保留此 $Itemset_k$，否則就刪除 [17]。
4. 判斷由步驟 3 所找出的 $Itemset_k$，其包含的 $Itemset_{k-1}$ 之子集合，是否都出現在步驟 1 中，可將 $Itemset_k$ 與步驟 2 中所有各 F_{k-1} 做執行以下布林運算：

   $$F_{k-1}\ xor\ Itemset_k \qquad (1)$$

   計數結果爲 $Itemset_1$ 的數目是否等於 k，假如成立就保留此 $Itemset_k$，否則就刪除。
5. 將步驟 4 擷取出的 $Itemset_k$ 執行以下的布林運算：

   $$Itemset_k\ or\ T_j\ xor\ T_j,\ (1 \le j \le m) \qquad (2)$$

   假如結果爲 $Itemset_0$，則表示 $Itemset_k \subseteq T_j$，掃瞄所有交易資料後，判斷出現的次數是否滿足最小支持度，假如符合就成爲 F_k，否則就刪除。
6. 對擷取出的 F_k，設定前項目組爲 X，則其後項目組 Y，可由以下的布林運算找出：

   $$Y = F_k\ xor\ X \qquad (3)$$

   計算關聯規則 $X \rightarrow Y$ 的信賴度，若滿足最小信賴度，則此關聯規則成立。
7. 跳至步驟 2 找 F_{k+1}，直到無法產生高頻項目組爲止。

在 Boolean-Apriori 演算法之步驟 4 及步驟 5 的運算中，可避免如 Wur 與 Leu 所描述之方法 [17]：判斷 $Itemset_k$ 是否爲高頻項目組，就須掃瞄所有交易資料一遍。我們利用 Apriori 演算法的執行步驟，先判斷 $Itemset_k$ 之所有子集合 $Itemset_{k-1}$ 是否都爲高頻項目組，惟有所有子集合 $Itemset_{k-1}$ 都爲高頻項目組，$Itemset_k$ 才有可能成爲高頻項目組，如此將可以減少掃瞄所有交易資料的次數。

（二）實例說明

我們以下面一個例子來說明前一小節所描述之 Boolean-Apriori 演算法其探勘關聯規則的過程。表 3 爲交易資料庫中有 4 位消費者的交易資料，其中 I={A, B, C, D, E} 表示全部項目所形成的集合，T={T_1, T_2, T_3, T_4} 表示全部交易資料所形成的集合，設定最小支持度爲 40%（即最小支持數量爲 1.6），最小信賴度爲 70%。

首先將各交易資料轉換成位元的資料格式：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]。擷取高頻項目組的過程說明如圖 1，無 4-項目組。

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

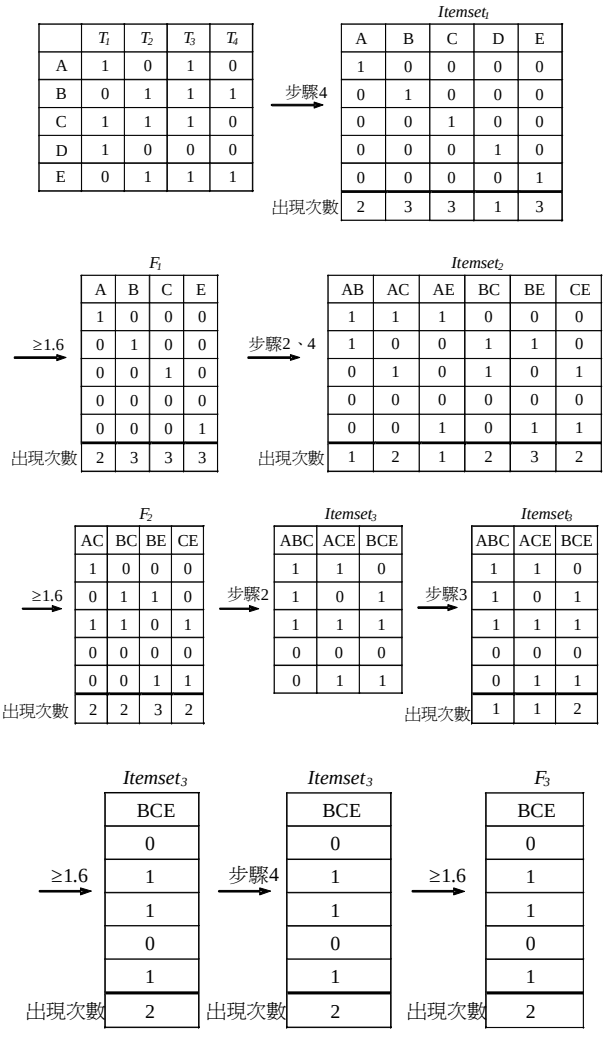

圖 1. 擷取高頻項目組之過程

經由以上探勘的過程，可分別找出的高頻 3-項目組有：BCE，高頻 2-項目組有：AC, BC, BE, CE。我們以高頻 BCE-項目組爲例，利用演算法的步驟 6，計算可形成的關聯規則且滿足最小信賴度者有：BC→E 與 EC→B。對其餘高頻項目組執行相同的計算，即可找出其他的關聯規則。

四、探勘包含有項目數量之關聯規則

在現實交易所記錄的資料中，往往會記錄消費者購買的產品項目與其數量，因此若能找出包含有項目數量的關聯規則，對企業擬定項目該以多少的數量來搭配銷售，必定可以提供相當有用的資訊。在 MQA-1 演算法中，交易資料若包含之項目的購買數量大於 1，則在探勘的過程中先將此項目分解，使得每一項目的數量都爲 1，並視爲不同的項目，例如某一交易資料 T_j={2A3BC}，在計算的過程中，分解爲 T_j={AABBBC}，每一項目的數量都爲 1，且第 1 個 A 與第 2 個 A 視爲不同的項目，再利用與 Apriori 演算法相近之方法來擷取高頻項目組，然後在計算高頻項目組所形成的關聯規則時，再來合併原本相同的項目。

在此一章節中，我們將以包裹資料庫其交易資料包含有項目與其購買數量爲探勘的資料來源，利用布林運算爲基礎，修改陳彥良等所提出之 MQA-1 演算法 [1]，探勘包含有項目數量的關聯規則。此章節共分爲兩小節：第一小節中，我們提出一個布林演算法，探勘包含有項目數量的關聯規則；第二小節中，我們以一實例來說明探勘的過程。

（一）探勘包含有項目數量之關聯規則的布林演算法

在說明如何以布林運算爲基礎，探勘包含有項目數量的關聯規則之前，我們先定義以下名詞：

- I={$i_1, i_2, ..., i_k, ..., i_a$}，是全部產品項目（items）的集合，共有 a 項，其中 $1 \le k \le a$。
- T={$T_1, T_2, ..., T_j, ..., T_m$}，爲包裹交易資料庫中全部消費者之交易資料的集合，共 m 筆，其中 T_j 爲消費者 j 的交易資料，$1 \le j \le m$。
- Q_{ik}=max{T_j 中第 i_k 項目的數量，$1 \le j \le m$}，即表示在所有的交易資料中，出現第 i_k 項目的最大數量。
- T_j 由 n 位元（bits）所組成，n 的大小可由以下式子決定：

$$n = Q_{i1} + Q_{i2} + \ldots + Q_{ia} \qquad (4)$$

其格式表示成 T_j=[$b_{i1}, b_{i2}, \ldots, b_{ik}, \ldots, b_{ia}$]，其中 b_{ik}=[$b_1, b_2, \ldots, b_f, \ldots, b_{Qik}$]，$b_f \in \{0, 1\}$，$1 \le f \le Q_{ik}$，表示第 i_k 項目的位元個數有 Q_{ik} 個，若某一交易資料第 i_k 項目有 c 個，則表示前面 c 個位元其值爲 “1”，其餘爲 “0”，$0 \le c \le Q_{ik}$。

- $Itemset_k$ 表示某一 k-項目組，其格式由 n 位元所組成，表示成 $Itemset_k$=[$b_1, b_2, b_3, \ldots, b_n$]，且有 k 個項目其值爲 “1”，$k \ge 0$。
- F_k 表示某一高頻 k-項目組，其格式定義與 $Itemset_k$ 相同，F_k=[$b_1, b_2, b_3, \ldots, b_n$]。

在探勘的過程中，我們仍然根據 MQA-1 演算法的設定條件，即判斷某一項目組是否被包含在交易資料中，其此一項目組之任一項目皆出現在交易資料中，且在交易資料中之購買次數必須大於或等於在此一項目組中的購買次數。我們先將交易資料轉換成位元的資料格式，並將數量爲 s 的項目，s>1，拆成 s 個數量爲 1 的項目，並且在計算的過程中，視爲不同的項目。我們利用 *or*、*xor* 布林運算（如表 1 及表 2）、及修改 MQA-1 演算法，設計一個方法，稱之爲 Boolean-MQA-1 演算法，來探勘包含有項目數量的關聯規則，其擷取的過程說明如下：

1. 將交易資料轉換成位元的資料格式。
2. 找出 F_{k-1}，k>1，若爲∅，則停止執行。
3. 任意兩個 F_{k-1} 做 *or* 布林運算，假如結果爲 $Itemset_k$，即有 k 個項目其值爲 “1”，且非重複者，就保留此 $Itemset_k$，否則就刪除。
4. 判斷由步驟 3 所找出的 $Itemset_k$，其包含的 $Itemset_{k-1}$ 之子集合，是否都出現在步驟 2 中，可將 $Itemset_k$ 與步驟 1 中所有各 F_{k-1} 執行以下的布林運算：

$$Itemset_k \ xor \ F_{k-1} \qquad (5)$$

計數結果爲 $Itemset_1$ 的數目是否等於 k，假如成立就保留此 $Itemset_k$，否則就刪除。

5. 將步驟 4 擷取出的 $Itemset_k$ 與交易資料 T_j（$1 \le j \le m$）執行以下的布林運算：

$$Itemset_k \ or \ T_j \ xor \ T_j \qquad (6)$$

假如結果爲 $Itemset_0$，則表示 $Itemset_k \subseteq T_j$，掃瞄所有交易資料之後，判斷出現的次數是否滿足最小支持度，假如符合就成爲 F_k，否則就刪除。

6. 對擷取出的 F_k，設定前項目組爲 X，則後項目組 Y，可由

以下運算找出：

$Y=F_k$ *xor* X，且 $X \cap Y=\varnothing$，即原先相同的項目不能分別包含於 X 和 Y 中。

假如關聯規則 $X \rightarrow Y$ 滿足最小信賴度，則再檢查 X 或 Y 中是否有原先相同的項目，若有則將之合併，即可找出包含有項目數量的關聯規則。

7. 跳至步驟 2 找 F_{k+1}，直到無法產生高頻項目組爲止。

在包裹資料庫中，MQA-1 演算法雖然修改 Aprior 演算法來找出包含有項目數量的關聯規則，但仍然使用 Apriori 演算法的計算方式。在以上所提出之 Boolean-MQA-1 演算法中，我們以較有效率之布林運算爲基礎所設計出的方法，將可以更有效率地找出包含有項目數量的關聯規則。

（二）實例說明

我們以下面一個例子來說明如何利用前一小節所描述的 Boolean-MQA-1 演算法，來擷取包含有項目數量之關聯規則的探勘過程。表 4 爲在包裹資料庫中有 4 位消費者的交易資料，其中 I={A, B, C} 表示全部項目所形成的集合，T={T_1, T_2, T_3, T_4} 表示全部交易資料所形成的集合，設定最小支持度爲 40%（即最小支持數量爲 1.6），最小信賴度爲 70%。

首先計算 n 的值爲：$n=max${A, A}+max{2B, 2B, B}+max{2C, C, C, C}=5。因此將各消費者之交易資料轉換成位元的資料格式，可表示成如表 5。

擷取包含有項目數量之高頻項目組的探勘過程說明如圖 2，無 4-項目組。

表 4. 包裹資料庫

交易資料編號	交易項目
T_1	A2C
T_2	2BC
T_3	A2BC
T_4	BC

表 5. 轉換成位元格式之包裹資料庫

	T_1	T_2	T_3	T_4
A	1	0	1	0
B	0	1	1	1
B	0	1	1	0
C	1	1	1	1
C	1	0	0	0

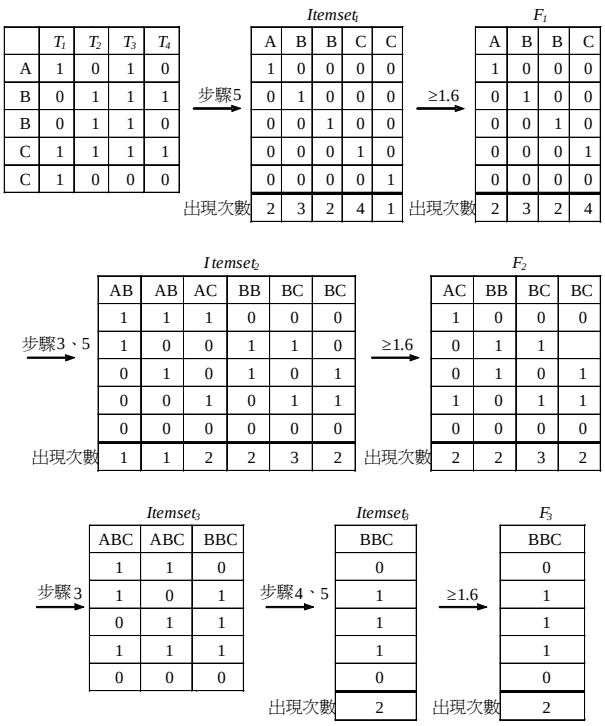

圖 2. 擷取包含有項目數量之高頻項目組的過程

經由以上探勘的過程，可分別找出的高頻 3-項目組有：BBC，高頻 2-項目組有：AC, BB, BC, BC。我們以高頻 3-項目組 BBC 爲例，利用步驟 6 計算其可能形成的關聯規則，且滿足最小信賴度者有：2B→C。對其餘高頻項目組執行相同的計算，即可找出其他包含有項目數量的關聯規則。

五、效能評估

在此一章節中，我們實驗評估前面章節所描述之演算法的執行效能，其實驗平台說明如表 6，交易資料由 IBM Data Mining 網站（http://www.almaden.ibm.com/）下載資料模擬程式，以產生評估實驗中所需要的交易資料。

爲了評估演算法的執行效能，我們共產生 10 個分別包含有 1000 筆交易資料的資料庫，然後依次累加其交易資料的數量分別成爲 1000, 2000, 3000, …, 10000，其所代表的資

表 6. 實驗平台

CPU	CPU-Pentium III 850
Main memory	256 Mbytes
作業系統	Windows 2000 Sever
使用語言	C#

料庫分別以 $D_1, D_2, D_3, D_4, D_5, D_6, D_7, D_8, D_9, D_{10}$ 表示之，如表 7。資料庫中的主要參數值其意義分別爲：*n* 代表項目的數量、*ntran* 爲交易資料的數量、*np* 爲型樣組合的數量、*tl* 爲交易資料的平均項目個數、*pl* 爲高頻項目組的平均長度，其餘參數以預設值表示之。以下我們分別評估 Wur 與 Leu 所提出之布林演算法 [17] 和第三節所描述之 Boolean-Apriori 演算法的執行效能。

在圖 3 中，我們以資料庫 D_1 爲探勘的資料來源，分別評估在不同最小支持度的條件下，兩個演算法的執行時間。在圖 4 中，我們固定最小支持度爲 0.5，分別評估以資料庫 $D_1, D_2, D_3, D_4, D_5, D_6, D_7, D_8, D_9, D_{10}$ 爲探勘的資料來源，兩個演算法的執行時間。

再者，我們考量消費者之交易資料中包含有購買的項目與其的數量，並以表 7 之 10 個資料庫爲例，在不失一般性的條件下，我們設定項目之購買數量爲 1 到 4 之間，然後以亂數隨機產生每筆交易資料所包含之項目所購買的數量，形成包裹資料庫，如表 8，並分別評估 MQA-1 演算法和第四章節所描述之 Boolean-MQA-1 演算法的執行效能。

表 7. 資料庫與其參數

參數 / 資料庫	*n*	*ntran*	*np*	*tl*	*pl*
D_1	100	1000	10000	10	4
D_2	100	2000	10000	10	4
D_3	100	3000	10000	10	4
D_4	100	4000	10000	10	4
D_5	100	5000	10000	10	4
D_6	100	6000	10000	10	4
D_7	100	7000	10000	10	4
D_8	100	8000	10000	10	4
D_9	100	9000	10000	10	4
D_{10}	100	10000	10000	10	4

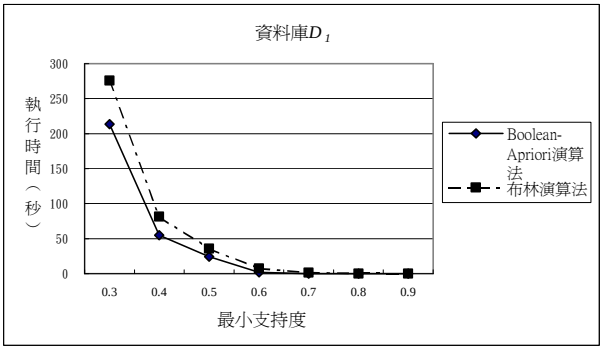

圖 3. 不同最小支持度的執行時間

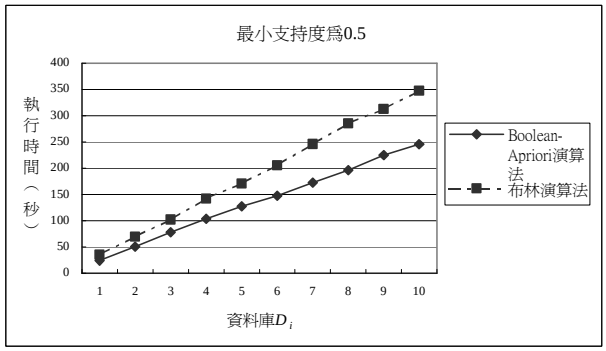

圖 4. 不同交易資料數量的執行時間

表 8. 包裹資料庫與其參數

參數 / 資料庫	*n*	*ntran*	*np*	*tl*	*pl*	項目購買數量
BD_1	100	1000	10000	10	4	1-4
BD_2	100	2000	10000	10	4	1-4
BD_3	100	3000	10000	10	4	1-4
BD_4	100	4000	10000	10	4	1-4
BD_5	100	5000	10000	10	4	1-4
BD_6	100	6000	10000	10	4	1-4
BD_7	100	7000	10000	10	4	1-4
BD_8	100	8000	10000	10	4	1-4
BD_9	100	9000	10000	10	4	1-4
BD_{10}	100	10000	10000	10	4	1-4

在圖 5 中，我們以包裹資料庫 BD_1 爲探勘的資料來源，分別評估在不同最小支持度的條件下，兩個演算法的執行時間。在圖 6 中，我們固定最小支持度爲 0.5，分別評估以包裹資料庫 $BD_1, BD_2, BD_3, BD_4, BD_5, BD_6, BD_7, BD_8, BD_9, BD_{10}$ 爲探勘的資料來源，兩個演算法的執行時間。

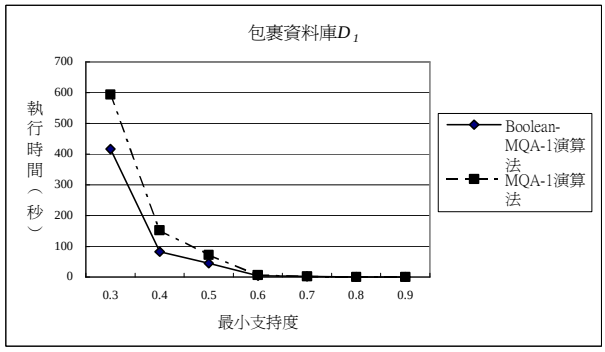

圖 5. 包含項目數量之不同最小支持度的執行時間

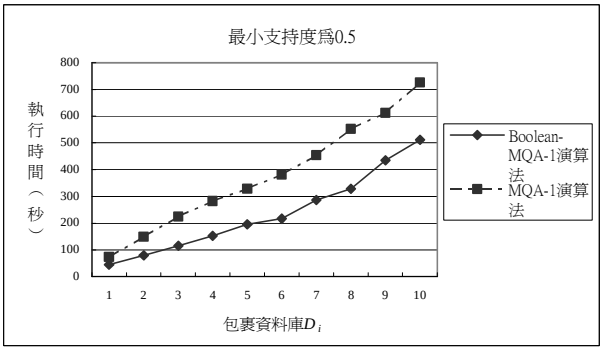

圖 6. 包含項目數量之不同交易資料數量的執行時間

從以上實驗評估中顯示，我們所提出之 Boolean-Apriori 演算法及 Boolean-MQA-1 演算法的執行效能，分別優於 Wur 與 Leu 所提出的布林演算法 [17] 及 MQA-1 演算法 [1]，可以更有效率地探勘關聯規則、及包含有項目數量的關聯規則。

六、結論

關聯規則是資料探勘技術中最常被用來表示項目之間關聯性的形式之一，如何提升探勘關聯規則的執行效能，是目前資料探勘技術中最重要的研究主題之一。在本篇論文中，我們提出兩個以布林運算爲基礎的演算法，可以分別有效率地擷取出關聯規則、及包含有項目數量的關聯規則。從實驗評估中顯示，我們所設計之 Boolean-Apriori 演算法及 Boolean-MQA-1 演算法的執行效能，分別優於 Wur 與 Leu 所提出的布林演算法 [17] 及陳彥良等所提出的 MQA-1 演算法 [1]。在表 9 中，我們說明所設計之 Boolean-Apriori 演算法與 Wur 與 Leu 所提出的布林演算法之間的差異。在表 10 中，我們說明所設計之 Boolean-MQA-1 演算法與 MQA-1 演算法之間的差異。

與我們分別就所需記憶體空間和執行時間，來探討布林運算與傳統字元運算之間的差異。在探關聯規則之前，需將交易資料所包含的項目先進行編碼，其中布林運算是將項目轉換成位元（bits）的形式，因此對有出現的項目而言，此轉換方式所佔用的記憶體空間是最小，只需一個位元，但由於須要記錄有未出現的項目，因此相較於傳統字元的轉換，其記錄項目的個數其數量也較多。在擷取高頻項目組的過程中，需要花費大量的時間於項目組與交易資料之間的比對。假設全部的項目個數爲 n，交易資料長度爲 t（即包含項目的個數），項目組的長度爲 i，對於字元運算的方式而言，其比對項目組是否出現在交易資料中所需花費的時間複雜度爲 $O(t+i)$，對布林運算的方式而言，其比對項目組是否出現在交易資料中所需花費的時間複雜度爲 $O(n/b)$，其中 b 爲機器可支援計算一次布林運算所佔用的位元數（本實驗評估所使用的平台爲 32 bits），我們以表 11 來說明布林運算和傳統字元運算之間的差異。因此，就以上兩種運算方法的整體平均執行效能而言，布林運算仍較優於傳統字元運算。

表 9. Boolean-Apriori 演算法和布林演算法

	布林演算法	Boolean-Apriori
運算方式	以布林運算爲基礎	同左
搜尋項目組 $itemset_i$ 方式，t>1	直接掃瞄交易資料庫	首先利用布林運算，計算項目組 $itemset_i$ 是否包含有全部其子項目組 $itemset_{i-1}$，若不是則刪除，否則保留再掃瞄交易
執行效能	較差	較佳
輸出形式	關聯規則	同左

表 10. Boolean-MQA-1 演算法和 MQA-1 演算法

	MQA-1 演算法	Boolean-MQA-1
運算方式	以傳統字元運算爲基礎	以布林運算爲基礎
搜尋項目組 $itemset_i$ 方式，i>1	採用類似 Apriori 演算法的方法	採用類似 Bollean_Apriori 演算法的方法
執行效能	較差	較佳
輸出形式	數量關聯規則	同右

表 11. 傳統字元運算和布林運算

	傳統字元運算	布林運算
運算方式	以傳統字元運算爲基礎	以布林運算爲基礎
項目轉換方式	出現的項目以字元或數字來編碼	以位元值爲 “1” 來表示項目的出現（位元初始值都爲 “0”）
當 t 趨近於 n 時所佔用記憶體空間	較大	較小
當 t<<n 時所佔用記憶體空間	較小	較大
當 b 趨近於 n 時，比對項目組是否出現在交易資料中所需執行的時間	較大	較小

參考文獻

1. 陳彥良、凌俊青、許秉瑜（民 90），在包裹式資料庫中挖掘數量關聯規則，資訊管理學報，7(2)，215-229。
2. Agrawal, R., T. Imielinski and A. Swami (1993) Mining association rules between sets of items in large database. Proceedings of the ACM SIGMOD Conference on Management of Data, Washington, D.C.
3. 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.
4. Bodon, F. and L. Rónyai (2003) Trie: An alternative data structure for data mining algorithms. *Mathematical and Computer Modelling*, 38, 739-751.
5. Coenen, F., P. Leng and S. Ahmed (2004) Data structure for association rule mining- T-trees and P-trees. *IEEE Transactions on Knowledge and Data Engineering*, 16(6), 774-778.
6. Da Silva Camargo, S. and P. Martins Engel (2002) MiRABIT: A new algorithm for mining association rules. Proceedings of the XII International Conference of the Chilean Computer Science Society (SCCC'02), Copiapó, Atacama, Chile.
7. Holt, J. D. and S. M. Chung (2002) Mining association rules using inverted hashing and pruning. *Information Processing Letters*, 83, 211-220.
8. Li, Z. C., P. L. He and M. Lei (2005) A high efficient AprioriTid algorithm for mining association rule. Proceedings of the Fourth International Conference on Machine Learning and Cybernetics, Guangzhou, China.
9. Liu, P. Q., Z. Z. Li and Y. L. Zhao (2004) Effective algorithm of mining frequent itemsets for association rules. Proceedings of the Third International Conference on Machine Learning and Cybernetics, Shanghai, China.
10. 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(5), 813-825.
11. Ruckert, U., L. Richter and S. Kramer (2004) Quantitative association rules based on half-spaces: An optimization approach. Proceedings of the Fourth IEEE International Conference on Data Mining (ICDM 2004), Brighton, UK.
12. Srikant, R. and R. Agrawal (1995) Mining generalized association rules. Proceedings of the 21th International Conference on Very Large Data Bases, Zurich, Switzerland.
13. Srikant, R. and R. Agrawal (1996) Mining quantitative association rules in large relational tables. Proceedings of the 1996 ACM SIGMOD Conference on Management of Data, Montreal, Canada.
14. Tsai, P. S. M. and C. M. Chen (2001) Mining quantitative association rules in a large database of sales transactions. *Journal of Information Science and Engineering*, 7, 667-681.
15. Tsay, Y. J. and J. Y. Chiang (2005) CBAR: An efficient method for mining association rules. *Knowledge-Based Systems*, 18, 99-105.
16. Tsay, Y. J. and Y. W. Chang-Chien (2004) An efficient cluster and decomposition algorithm for mining association rules. *Information Sciences*, 160, 161-171.
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.

收件：94.11.23　修正：95.03.30　接受：95.06.07