PCP：正相關與封閉項目集之關聯規則演算法

黃仁鵬

南臺科技大學資訊管理系

jehuang@stust.edu.tw

摘 要

探勘交易資料庫中項目間的關聯性，有助於組織的決策支援；其中探勘技術的演算法，關係著資料探勘效能及資源有效的利用。在相關的研究中，其所產生的關聯規則最常為人詬病的問題之一，就是過濾的機制不佳，產生過多的規則，致使決策者淹沒在繁多的規則中，因此提出有效的方法精簡關聯規則是本研究的主要方向。本研究所提出的 PCP(Positive Correlation and Closed Itemsets by Phase)演算法是以 GRA(Gradation Reduction Approaches)演算法為主軸並加入封閉項目集(closed itemset)概念。雖然 GRA 演算法的階段縮減交易長度機制，已大量減少非高頻型樣的數量，但仍存在著高頻項目集不夠精簡及關聯規則太多等問題，導致使用者不能精準的解讀和利用所產生的關聯規則。本研究利用刪除負相關項目集來縮減每一階段的高頻項目集，所探勘出的高頻項目集數量小但更具代表性，並在每一階段長度產生的高頻項目集中，找出包含前一階段長度的封閉項目集，並刪除被包含的前一階段長度的高頻項目集，最後以高效率探勘出精簡的關聯規則。在現實生活中的資料庫容量通常都是大於記憶體容量，為了解決此問題，本研究也以 PCP 演算法為主題提出另一修正版 PCP-M (Positive Correlation and Closed Itemsets by Phase - Modified Version)演算法，PCP-M 演算法採用資料庫分割方式執行探勘任務，每個子資料庫僅需進行四次實體 I/O 動作，不隨著高頻項目集的長度增長而增加實體 I/O 的次數，以避免耗費過多的 I/O 時間，也可有效提高執行效率與實用性。

關鍵詞：正相關項目集、封閉項目集、資料探勘、關聯規則

PCP: Mining Association Rules Algorithm with Positive Correlation and Closed Itemsets

Jen-Peng Huang

Department of Information Management, Southern Taiwan University of Science and Technology

Abstract

The correlation among the itemsets of data mining in the transactions intensified the effect of decision supporting system in the organization. The algorithm of data mining technology plays the key roles for the effectiveness of data mining and the utilities of the resources. One of the most defective problems of the previous researches for the correlative data mining algorithm has low effectiveness for system filtering. Too many regulations which have been generated by the decision supporting systems cause the poor efficiency. To present a simple and effective correlation algorithm is the main objective of this research. A Positive Correlation and Closed Itemsets by Phase (PCP) algorithm was presented in this research to modify Gradation Reduction Approaches (GRA) algorithm by adding concept of closed itemsets and positive correlation itemsets. Although

Received: Aug. 13, 2015; first revised: Dec. 15, 2015; accepted: Jan., 2016.

Corresponding author: J -P. Huang

GRA algorithm has gradually reduced a great deal of works in the transactions of database, and can reduce a great number of infrequent itemsets, the decision makers are still confused by too many frequent itemsets and association rules which are generated by GRA algorithm when they want to make decisions. The PCP algorithm use the concept of closed itemsets and positive correlation itemsets to reduce the number of association rules and the mining results will be meaningful. The size of the databases in the real world is always greater than the size of the memory. In order to solve this problem, we propose a modifying algorithm - PCP-M (Positive Correlation and Closed Itemsets by Phase - Modified Version); it divides a large database into many sub-databases and mines association rules from those sub-databases. The PCP-M algorithm only scans database four times and will not be affect by the length of frequent itemsets. The PCP-M algorithm avoids wasting a lot of I/O time and increases the efficiency and the practicability in application.

Keywords: Positive Correlation Itemsets, Closed Itemsets, Data Mining, Association Rule

壹、前言

科技時代提供了豐富的資訊和新興的技術，各行各業在此浪潮下，企業需不斷的在眾多競爭者中謀求致勝之道。而有關提供企業提升競爭力的各項解決方案課題隨之應運而生，為組織內部提供策略方案，並為企業帶來更高的利潤貢獻。其中資料探勘(Data Mining)在此風潮下扮演著重要的影響力，資料探勘可由大量資料中擷取出有價值的知識，其能協助企業決策者獲得知識並創造商機，故引起廣泛的重視，也造就了更多新研究的發展。探勘技術的演算法之優劣，直接關係著資料探勘效能及記憶體資源有效的利用。過往關於關聯規則(Association Rule)技術之演算法所努力的方向，不外乎是朝著改善演算法效能及記憶體使用率為目標。關聯規則廣泛地應用在各相關領域，它的結果具體的呈現項目之間的相關性。部份研究產生的關聯規則中最為人所垢病的問題，就是過濾機制不完善，導致產生過多的規則，致使決策者淹沒在瑣碎的規則中。本篇論文以改善項目集過濾機制及高頻項目集數量，進而精簡關聯規則為研究方向。因而提出以效能上表現優異的 GRA (Gradation Reduction Approaches)演算法(黃仁鵬、藍國誠，2006)為主軸，加入過濾負相關項目集及找出封閉項目集，進一步減少產生項目集的數量及精簡關聯規則。

貳、文獻探討

一、Apriori 演算法

早期探勘高頻項目集最著名的方法是由 Agrawal and Srikant(1994)所提出的 Apriori 演算法，該演算法為關聯規則中位居代表性的研究。Apriori 演算法將探勘過程分為兩個階段：階段一：首先掃描資料庫，針對每一項目累計其在資料庫中出現的次數，產生長度為 1 的候選項目集(Candidate Itemsets)。再找出候選項目集中滿足最小支持度(Minimum Support)的項目，稱之為高頻 1-項目集(Frequent 1-Itemsets)。階段二：由兩個 k-高頻項目集匹配組合成(k+1)-候選項目集，其中必須兩個項目集中有 k 個項目為相同的。再一次掃描資料庫計算候選項目集的支持度，其中滿足最小支持度的項目集即為高頻(k+1)-項目集(Frequent (k+1)-Itemsets)。重複以上過程，直到無法找到更長的高頻項目集為。最後依據產生的高頻項目集推導出有用的關聯規則。Apriori 演算法的優點是結構簡單，沒有複雜的資料結構。但是它的缺點是必須多次的重複掃描資料庫以及產生大量的侯選項目集，而導致效能不佳，尤其在挖掘長交易記錄時此項缺陷更顯突出。

二、FP-growth 演算法

FP-growth 演算法(Han, Pei & Yin, 2000)已完全擺脫產生候選項目集的概念，而是將每筆交易中的高頻

項目集壓縮到樹狀結構中，探勘全程僅須掃描資料庫二次，第一次掃描交易資料並對計數項目出現次數，去除非高頻的項目後，每筆交易依項目支持度大小降冪排序。接著再掃描資料庫第二次，依序將交易資料建成一顆 FP-tree 的樹狀結構，利用相同字首共享樹中同一路徑的原則，並於每個樹節點(Nodes)記錄出現次數及資訊，以及利用 Header Table 來指向 FP-tree 中每個項目的第一個節點，並且樹中每個相同的項目都會建立一個橫向的連結關係。

由於 FP-tree 的結構緊密，並且可以利用 Header Table 來找出 conditional pattern tree，進而以遞迴方式找出所有的高頻項目。因此是一個非常有效率且很節省記憶體的演算法。FP-tree 演算法雖已改善掃描資料庫的次數及避免產生大量的候選項目集，但因建立資料結構耗費時間，為其主要的缺點。

三、NNIL 演算法

NNIL 演算法(賈坤芳、劉家銘 ,2001)中提出兩個重點研究，第一，以線上挖掘演算法，使用者可任意更動門檻值，且於短時間內回應所有的規則，其目的是為解決 Apriori 演算法缺乏彈性和耗時的缺點。第二，利用負相關的項目集來預先儲存項目集絡(Itemset Lattice)，提供使用者線上挖掘關聯規則，還可避免因產生負相關的項目集，而導致產生並非真正相關的規則。

NNIL 演算法運用預先處理式的線上挖掘方法，在使用者作資訊挖掘之前，事先將所需的資訊儲存下來，方便之後使用者更動門檻值時，可以直接從這些儲存下來的資訊挖掘出相關的規則，避免挖掘過程重複的計算。如此一來，每當使用者更動門檻值後，就可以避免重複掃描資料庫。

四、A-close 演算法

Pasquier 等人於 1999 年提出 A-close 演算法(Pasquier, Bastide, Taouil,& Lakhal, 1999)，其首開探勘高頻封閉項目集(Frequent Closed Itemsets)的新觀念，提供給後來者一個改善探勘過程的方向。A-close 演算法在不致遺失任何資料的限制下，利用一種封閉的機制來找出高頻封閉項目集，進而減少產生關聯規則的數量。

A-close 演算法中定義高頻封閉項目集為一個項目集 I 的支持度與 I 的封閉項目集的支持度相同，且此封閉項目集的最大高頻項目集的集合與最大的高頻封閉項目集的集合是完全相同的，意即：一個高頻封閉項目集為一高頻項目集，且此高頻封閉項目集中不會存在另一個大項目集(Superset)的支持度是與此高頻封閉項目集的支持度是相同的。

五、CLOSET 演算法

CLOSET 演算法(Pei, Han, &Mao,2000)是植基於 FP-tree 上的演算法，主要是用來探勘出高頻封閉項目集(Frequent Closed Itemsets)，而探勘封閉項目集的目的是為了改善使用者必須在大量的高頻項目集與規則下篩選出有用的其中一個規則。其探勘的的技術有三項：1. 以壓縮的 FP-tree 結構建立資料，無須產生候選項目集，自 FP-tree 的結構中的高頻型樣樹探勘出封閉項目集(Closed Itemsets)。2. 以單一字首路徑壓縮技術來快速的找出高頻封閉項目集。3. 探勘一個以分割為基礎的投影機制來探勘大型資料庫。透過 FP-tree 找出每個結點的 conditional pattern base 中的所有的項目集，再以該結點的支持度為依據，計算這些項目出現的次數，假使項目出現的次數小於該結點的支持度者，則不列入該封閉項目集中的項目，到此即產生所要的該支持度的封閉項目集。

六、GDA 演算法

GDA 演算法是由(黃仁鵬、熊浩志 ,2005)所提出，其主要是改進演算法 IDA(黃仁鵬、熊浩志、郭煌政 (2004)的兩項缺點。在 GDA 演算法的特色是以階段長度拆解，及採用與 IDA 演算法不同的遮罩模式來產生子項目集，並於每階段完成後釋放記憶體空間，對記憶體的利用大有助益。以下將概述 GDA 演算法的優點：1. 利用遮罩的方式將交易資料依長度升冪排序後，分別拆解產生高頻項目集 L2、 L3、 L4…，

如果無法再產生高頻項目集就停止拆解，而非全部拆解，如此可以節省拆解時間。階段拆解能避免對於交易長度的限制及對記憶體利用率的改善。2. 無須產生候選項目集，避免將時間浪費在無關的項目集上。3. 在每個階段適時的釋放非高頻項目集所佔的記憶體空間，因此所須的記憶體空間不會一直上升，將有效利用記憶體。4. 當進行到某個長度發現沒有高頻項目集時，則即可忽略後續的資料，提前完成探勘的工作。 GDA 演算法在執行過程中，並無完全有效的過濾產生非必要項目集的機制，以致仍將產生大量的項目集，影響其探勘的效能。

七、GRA 演算法

GRA (Gradation Reduction Approaches)演算法是由黃仁鵬、藍國誠(2005)所提出，是為改進 GDA 演算法(黃仁鵬、熊浩志 ,2005)的兩大問題，其一為 GDA 演算法在探勘交易長度較長時效能不佳，其二為在某階段所產生的大量非必要項目集時，將導致記憶體使用率不良。為改進以上兩項缺點，GRA 提出階段縮減交易長度的機制，將可大量減少產生非必要的項目集，較適用於探勘交易長度較長的資料庫。GRA 演算法仍延用 GDA 演算法中的階段拆解方法，無須產生候選項目集，其結構簡單，僅須歷經四次資料庫的 I/O，即可完成探勘的任務。列舉 GRA 演算法優點如下：1. 無須產生候選項目。2. 利用遮罩的方式將交易資料階段拆解的程序來產生高頻項目集，能避免對於交易長度的限制及對記憶體利用率的改善。3. 在進行每個階段長度項目集拆解前，利用階段產生的高頻項目集所找出的高頻項目來縮減下一階段的交易資料庫。階段縮減交易長度機制，可大量減少產生非高頻項目集的數量。4. 合併相同的交易，避免重複處理相同的交易資料，提升效能。5. 以前序階段拆解過濾非必要項目集，提升效能。6. 提出以分割子資料庫的方式適用於大型資料庫的探勘。GRA 演算法在效能上的表現相當優越，對於記憶體的利用率也有顯著的改善效益。

參、研究方法

PCP(Positive Correlation and Closed Itemsets by Phase)演算法主要是以 GRA 演算法(黃仁鵬、藍國誠，2006)探勘的方法為主軸，且本文所提出的 PCP 演算法仍會保留 GRA 演算法快速產生項目集的核心概念，並加入項目集過濾的機制。雖然，GRA 演算法在執行過程中，已針對交易長度逐步縮減，同時，也將更新後相同的交易記錄進行計數累計的處理，可避免不斷地重複拆解相同的交易記錄。然而，諸如此類的作法，仍有可能在項目間的關聯性低時，將會因無法有效節制高頻項目的數量，而可能在每一階段都必須儲存一些不重要或不必要的高頻項目集，導致耗費記憶體支援及最後產生的關聯性規則多而無當，因此 PCP 演算法加入了過濾負相關項目集及非封閉項目集的機制，進一步的減少產生非必要的項目集。而 PCP 演算法將繼續延用 GRA 每階段產生新項目集的遮罩技術，並利用前一階段的高頻項目集中所含有的高頻項目資訊，來刪除該階段資料庫中非高頻的項目，同時，在項目集通過該階段的支持度門檻值後，緊接著計算該項目的實際出現機率，刪除實際出現機率低於期望出現機率的項目集，保留更具價值的項目集成為正相關高頻項目集，接下來計算前一階段高頻項目集中項目的次數，刪除低於下一階段長度減一的項目，成為下一階段的資料庫，將更新後的相同的交易記錄進行計數累計的處理，可避免不斷地重複拆解相同的交易記錄，經由上述的階段縮減機制後，將可使資料庫逐漸縮小，同時，也能使 PCP 演算法僅產生最有價值的高頻項目集。

一、PCP 演算法流程圖

本研究所提出之 PCP 演算法的流程圖，如圖 1 所示：

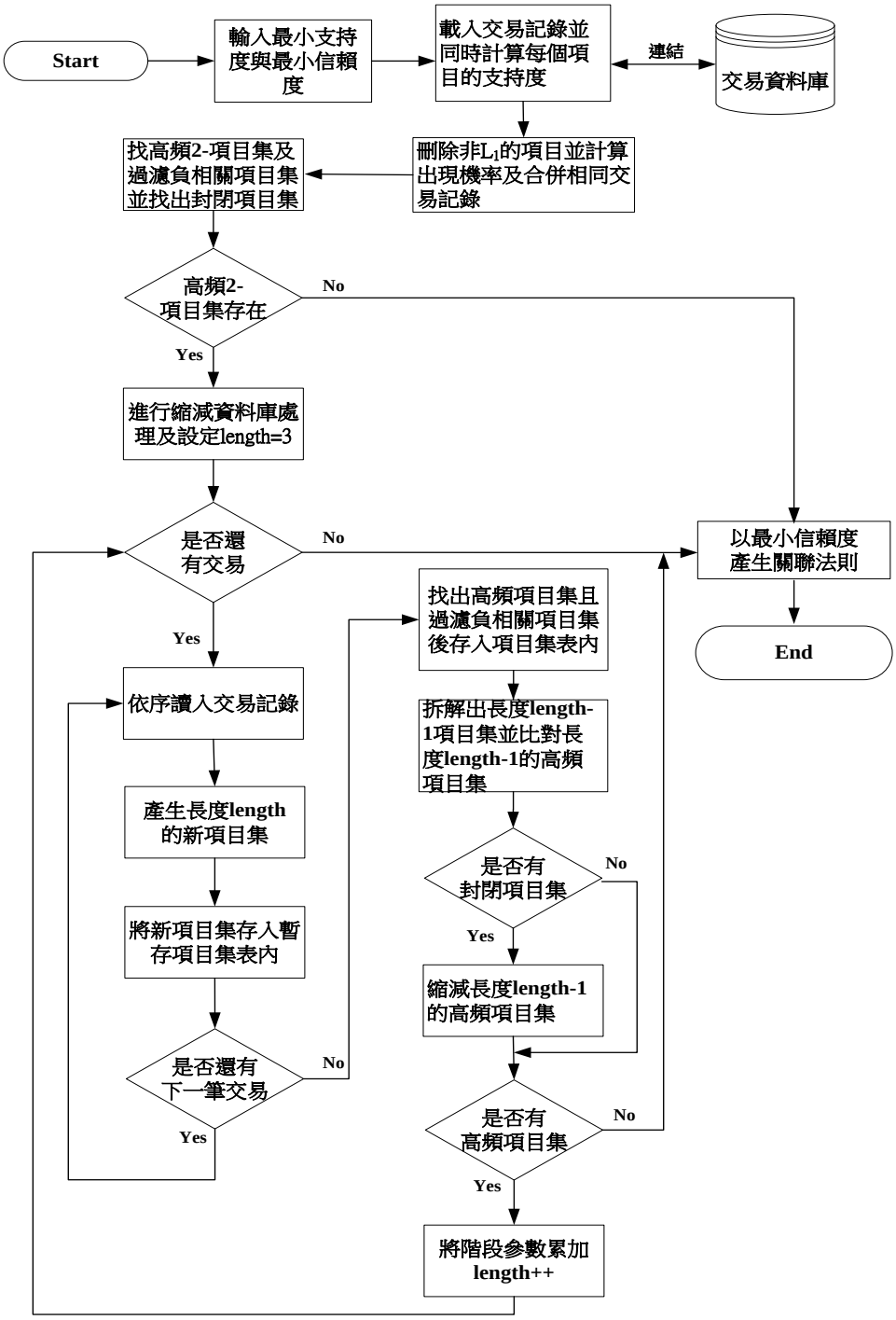

圖 1　PCP 演算法流程圖

二、PCP 演算法虛擬碼

本研究所提出 PCP 演算法的虛擬碼，如圖 2 所示：

```
Algorithm PCP
Input : DBName ,        //資料庫名稱
        min_Sup ,       //最小支持度
        min_Conf ,      //最小信賴度
Output :Rules           //關聯規則

get data from DB and save in DBdata;                    //讀取資料庫內容存入 DBdata 中
scan DBdata to get L1 and delete not L1 item in DB[0]; //去除非 L1 項目,並計算出現機率
save L1 into L[0] and Large1;                           //將 L1 儲存至 L[0]及 Large1
scan DB[0] to get L2 and delete not frequent item and not correlation to save in L[1];
                                                        //找出 L2 存入 L[1]
let L[0] compare with L[1] and delete closed items in L[0];
int length=3;                                           //設定目前欲探勘的階段長度 length=3

for (i=2; i<maxLen; i++) {                              //i 代表該階段，由 2 開始到最長的交易長度

      for (j=0; j<DB[length-1].length; j++) {           //依序讀入該階段的交易記錄
            gd.setBase( DB[length-2][j] , i             //設定要拆解的項目集長度
            while (tmp = gd.getNext() != null) {        //當還有下一筆資料時
                  temp.add(tmp);                        //取得該階段高頻項目集且正相關且
                                                        //不存在前一階段負相關的高頻項目集
            }
        }
      }
      L[length-1] = temp;
      let L[length-1] compare with L[length-2] and delete closed items in L[length-1];
                                                        //比對是否包含前一階段高頻項目集
                                                        //是則過濾前一階段階段高頻項目集
      find frequent itemsets from temp;                 //由 temp 中找出高頻項目
      if not exist any frequent itemsets then break;    //若無任何高頻項目則離開
      DB[length-1]=Prun_DB( DB[length-2] , L [length-1] , length );   //修剪資料庫
      length++;                                         //累加長度，即進行下一階段探勘
  }

get Rules from L[];                                     //由 L[]中的高頻項目集找關聯規則存入 Rule
return Rules;
```

圖 2　PCP 演算法虛擬碼

三、資料壓縮

PCP 演算法為善用記憶體的空間與配合項目集表(Itemset Table)存取的型態，因此讀取資料庫的交易轉換成以 char[]的資料型態存入記憶體時，藉以對資料進行壓縮，以減少記憶體的負擔。而壓縮的做法為依序讀取每筆交易的同時記錄每個項目的出現次數。而且在累計的同時，順便把原本項目的字串轉換為字元的型態，可以有效的節省記憶體。

四、階段縮減交易長度及合併之處理

在首次讀取完所有交易資料庫後，隨即計算單一項目的支持度，將通過最小支持度的項目篩選出來，並扣除交易資料中非高頻的項目，將可事先去除不具重要性的項目，更進一步的縮短交易的長度，有助於在接下來拆解過程的簡化與降低所需儲存的空間。此種處理方式類似於 FP-growth 中首先刪除資料庫中

不過最小支持度的項目。唯 FP-growth 另外再將通過單一項目支持度的每筆交易資料重新依照項目的支持度由大至小排序，而 PCP 演算法並不須要如此處理。其次在階段得到的高頻項目集(項目集長度 n>1)中，計數其中項目的出現次數，若項目的出現次數小於下階段長度減一或不存在的項目，則不可能產生下一階段的項目集，則可事先去除此不必要的項目，另亦可刪去該交易長度小於下階段需探勘的項目集長度的記錄，更進一步的縮短交易的筆數，並合併相同記錄的交易，有助於在接下來拆解過程的簡化與降低所需儲存的空間。

五、 以前一階段高頻項目集過濾非必要項目集之程序說明

本研究為了能更有效修剪每一階段的資料庫，將繼續延用 GRA 演算法(黃仁鵬、藍國誠 ,2006)中對項目集的拆解方法，而拆解方法源自於 GDA 演算法(黃仁鵬、熊浩志 ,2005)中的階段拆解產生新項目集的概念，其做法就是利用階段拆解將交易拆解出所設定的長度之子項目集，利用上階段的高頻項目集當作下階段產生新項目集的方法，過濾每一階段長度的非必要項目集。在進行拆解前，先設定三個參數，第一個是要拆解的交易記錄(char[])，第二個是目前的階段長度(int)，亦即要拆解成多長的子項目集，第三個是上階段的高頻項目集資訊。具備這三個資訊，將可以拆解出此階段可能為高頻的子項目集。在每階段產生項目集時，將利用前階段的高頻項目集作為檢查資訊。例如，產生長度為 n 的項目集時，假使 (n-1)個項目為在前階段的高頻項目集，則產生以該前(n-1)個項目為前序的所有新項目集；反之，若該前(n-1)個項目在前階段為非高頻時，即不產生以該前(n-1)個項目為前序的所有新項目集。本研究利用前階段的高頻項目集資訊作為下階段項目集的檢查資訊，能有效避免產生非必要的項目集，同時，亦可提昇效能及記憶體的使用率。

首先舉例說明階段拆解過濾的程序。假設有一筆交易記錄的內容為{ABCE}，若將產生長度為 3 的子項目集時，設定遮罩的數量為 3，且設定第一個遮罩為 1110，此遮罩與{ABCE}進行 AND 的運算後，可產生項目集 ABC，接著，移動遮罩的位置，得到 1101，所取得項目集為 ABE，其餘的項目集皆以上述方法進行拆解後，另外產生了 ACE、BCE。拆解的方式及過程如表 1 所示。

其次，將繼續說明利用前一階段的高頻項目集做為下一階段過濾依據的條件。假設有一筆交易記錄的內容為{ABCDE}，若將產生長度為 4 的子項目集時，設定遮罩的數量為 4，且設定第一個遮罩為 11110，此遮罩與{ABCDE}進行 AND 的運算後，可產生項目集 ABCD，取其項目集的前序得到 ABC，由於前序 ABC 為前一階段長度的高頻項目集，因此可產生含有該前序的項目集，所以產生的項目集 ABCD 為有效的，接著仍以 ABC 為前序，移動遮罩位置後得到 11101，此遮罩與{ABCDE}進行 AND 的運算後，可產生項目集 ABCE。至此交易記錄{ABCDE}已完成以 11110 為遮罩拆解子項目集的程序。接著將移動遮罩位置得到 11011，依上述的方法，得到子項目集 ABDE，其中 ABDE 的前序為 ABD，由於前序 ABD 並非是前一階段長度的高頻項目集，因此停止以該前序來產生項目集的工作。依據上述的規則持續移動遮罩的位置，直到不能再產生下一個遮罩為止。處理完成這些步驟即完成交易記錄{ABCDE}的拆解子項目的動作，所有過程如圖 3 所示。

表 1　階段拆解長度 3 之項目集說明表

交易內容	遮罩	子項目集
ABCE	1110	ABC
ABCE	1101	ABE
ABCE	1011	ACE
ABCE	0111	BCE

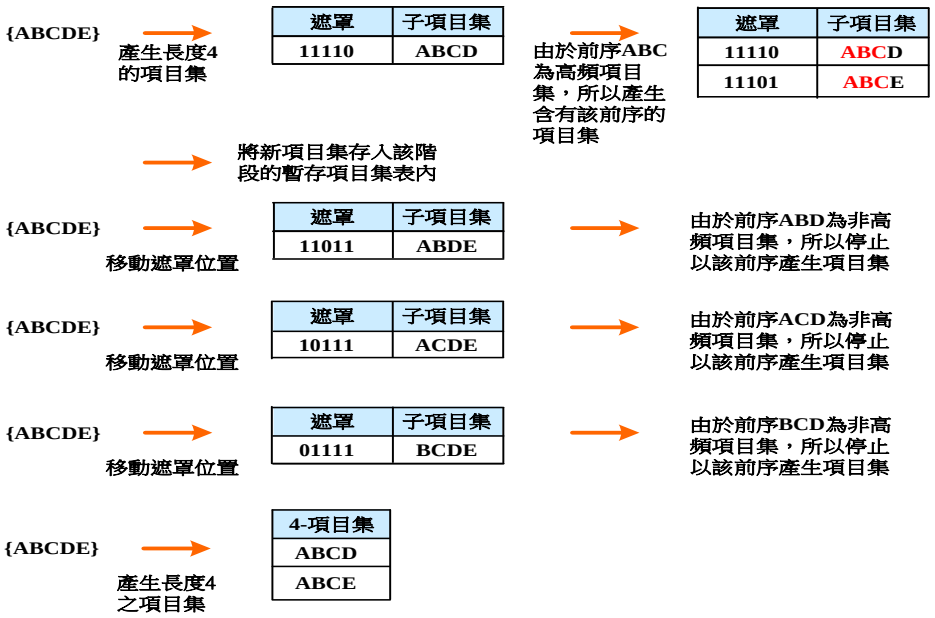

圖 3 拆解子項目集程序圖

六、過濾負相關高頻項目集之程序說明

本研究為了進一步檢查正相關高頻項目集(項目長度 n>2)中是否包含前一階段的負相關項目集，將採用遮罩拆解的方式將交易拆解出所設定長度的子項目集，利用前一階段的負相關項目集做為判斷該被拆解的項目集是否為負相關項目集的依據，來過濾每一階段長度的負相關項目集。在進行拆解前，先設定二個參數，第一個是要拆解的交易記錄(char[])，第二個是要拆解成多長的子項目集，具備這二個資訊，將可以拆解出此所需要的子項目集。

首先舉例說明拆解過濾的程序。假設有二組項目集的內容為 ABC 及 ABE，欲檢查其是否包含負相關 L2(NL2)。故將產生長度為 2 的子項目集，設定遮罩的數量為 2，且設定遮罩的初始值為 110，此遮罩與 ABC 進行 AND 的運算後，可產生項目集 AB，接著，移動遮罩的位置，得到 101，所取得項目集為 AC，再移動遮罩的位置，得到 011，會取得項目集 BC；ABE 項目集亦以同樣的方法進行拆解後，另外產生了 AB、AE 及 BE。接著將繼續檢查的處理，檢查 ABC 所產生的子項目集中是否為 NL2，若否則保留，反之若是則過濾。拆解的方式及過濾過程如圖 4 所示。

七、階段過濾負相關項目集之處理程序說明

在首次讀取完所有交易資料庫後，計算通過最小支持度的單一項目之出現機率，其計算公式為單一項目的出現次數除以交易資料庫總筆數。此單一項目的出現機率將是後續探勘中計算項目集的期望出現機率之依據。

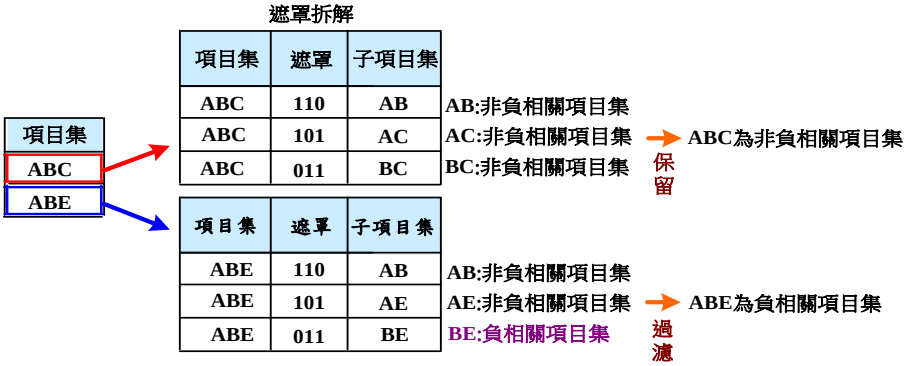

圖 4 拆解子項目集程序圖

本論文在每個階段皆會進行兩種負相關項目集的過濾，第一種是出現機率的過濾，第二種是前一階段的負相關項目集的過濾。以下將就兩種過濾的方法說明如下：

在探勘的過程中，將陸續產生 L_n (n>1)的高頻項目集，該 L_n 中的所有項目集雖已滿足最小支持度門檻的檢查，仍須再經過另兩項過濾負相關項目集的檢查，才是所要的正相關高頻項目集，其中第一項是檢查出現機率，計算其項目集的實際出現機率與期望出現機率，實際出現機率為該項目集在該階段資料庫出現的次數除以該階段資料庫的總筆數，而期望出現機率為該項目集中每個項目的出現機率相乘，如果處理的項目集長度 n>1 時，其項目集的實際出現機率小於期望出現機率時，則捨去，但將此項目集的記錄保留在負相關高頻 n-項目集 (NLn)中，做為下一階段判斷是否含有負相關項目集時所用。第二項檢查是前一階段負相關項目集的過濾，若處理的長度 n>2 時，依序將 L_n 中的項目一一拆解為數個 C_{n-1} 子項目集，若任何一個 C_{n-1} 子項目存在於在 NL_{n-1} 中，則將該項目從 L_n 中刪除，將之存入 NL_n 中，最後通過三項檢查即成為真正的正相關高頻項目集。

Lemma 1：假設 L_k 為正相關高頻 k-項目集，但如果任一 L_k 中的項目中的任何一個 C_{k-1} 子項目存在負相關項目集時，則不予保留該項目，因為產生的 L_k 項目是無意義的，因此移除該項目。

說明如下：基於一種向下封閉(Downward Closure)的特性限制—任何非高頻的(k-1)-項目集都不可能是高頻 k-項目集 Lk 中項目的子項目集。因此，如果任何一個候選 k-項目集的(k-1)-子項目不在高頻 (k-1)-項目集中，則該候選 k-項目集也不可能為高頻，可自候選項目集中刪除。假設 L3 項目集為 abc，其 L2 子項目集為 ab、ac 及 bc，如果已知 ac 為負相關項目集，亦即 abc 已無意義，故不可能成為正相關高頻項目集。

八、過濾高頻項目集之處理程序說明

本研究為了能更進一步修剪每一階段的資料庫，採用 GRA 演算法(黃仁鵬、藍國誠 ,2006)中有效過濾每一階段長度的高頻項目集之機制。在每一階段的所有高頻項目集中，仍可提前獲得下一階段絕對為非高頻項目的資訊，即在所有高頻項目集中，如果某項目出現的次數低於下一階段的長度減一時，即表示該項目將無法在下一階段組合出新項目集，若能事先自資料庫刪除該項目，將更有助於修剪資料庫，且也可避免產生非必要的項目集，提昇記憶體的使用率，亦可省下非必要的探勘時間。

Lemma 2：如果 L_{n-1} 中的項目集的元素出現少於 n-1 次，則不可能為 L_n 中的元素，因此可以刪除，其中，L 為高頻項目集，n 為項目集的項目數量，亦即是項目集的長度。

說明如下：L_n 中的有 n-1 個 L_{n-1}，因此 L_n 中各個元素必須出現 n-1 次。反之如果 L_{n-1} 中的元素出現少於 n-1 次，則不可能為 L_n 中的元素。例如：假設 L_4 的一高頻項目集為 abcd，其 L_3 子項目集為 abc、abd、acd、bcd，可得知 a、b、c、d 在 L_3 子項目集內各出現 3 次。假設 L_3 子項目集為 abc、abd、acd，亦即缺少一個項目集 bcd 時，將無法產生長度為 4 的高頻項目集 abcd，也就是說，abcd 不可能為 L_4，因為少了一個 bcd。

九、過濾前一階段之高頻項目集之處理

為了精簡所產生的高頻項目集，本研究在每個階段加入了找出封閉項目集的方法，逐步地減少所產生的高頻項目集，亦可釋放出記憶體的使用。首先在產生高頻 n-項目集後，隨即拆解出 n-1 項目集，並比對高頻 n-1 項目集，假使高頻 n-1 項目集為高頻 n-項目集的子項目集且支持度相同，則可將高頻 n-1 項目集自高頻項目集表中刪除，達到減少不必要的項目集數量及節省記憶體的目的，但不致有資料遺失的問題。在此舉一範例說明處理的過程。假設已知的高頻 2-項目集有 AC、AF、FM、CF 及 CM，它們的支持度分別是 3、4、4、4 及 3，而目前階段產生的高頻 3-項目集有 ACF 及 CFM，將這些高頻項目集拆解成長度 2 的項目集，則產生了 AC、AF、CF、CM 及 FM 的項目集，以這些項目集比對前一階段的高頻 2-項目集，結果比對得到 AC、CM 為高頻 2-項目集且支持度相同，所以得到 ACF 及 CFM 分別是 AC 及 CM 的封閉項目集。因此可將高頻 2-項目集中的 AC 及 CM 項目集刪除，其中因 AF、FM 與 CF 並未符合比對的條件，因此保留。到此即完成過濾前一階段的高頻項目集的程序。階段高頻項目集與被過濾掉的高頻項目集如圖 5 所示。

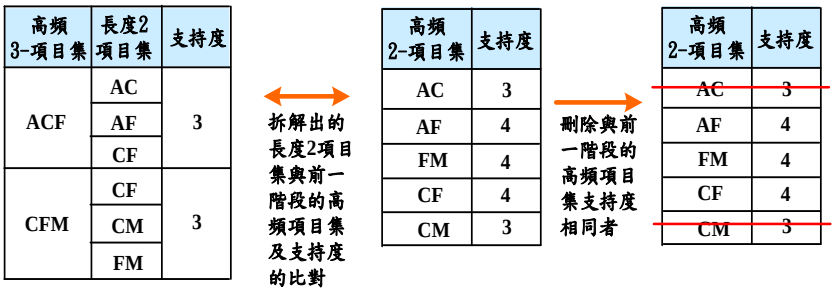

圖 5 階段高頻項目集與上一階高頻項目集圖

十、PCP-M 演算法實例說明

由於目前在實務資料中，資料庫的資料量通常是遠超過記憶體的空間容量，為了解決這問題，GRA 演算法(黃仁鵬、藍國誠 ,2006)中以該演算法為主要架構提出另一修訂版 GRA-M 演算法(Gradation Reduction Approaches - Modified Version)(黃仁鵬、藍國誠 ,2006)，在資料庫部份採用切割方式，即將大型資料庫切割成適當的子資料庫，而每個子資料庫僅需進行四次實體 I/O 動作，即可完成探勘任務，將可有效降低非必要時間成本的浪費，以提昇執行效能。本研究也以 GRA-M 演算法為藍本，提出 PCP-M 演算法(Positive Correlation and Closed Itemsets by Phase - Modified Version)以期達到在大型資料庫中執行本演算法，而能突破有限記憶體空間的問題。PCP-M 演算法的流程圖如圖 6 所示，圖 6 中框線所標示之處，即 PCP-M 演算法與 PCP 演算法相異之處。PCP-M 演算法虛擬碼如圖 7 所示。並提供範例來說明 PCP-M 演算法的流程。

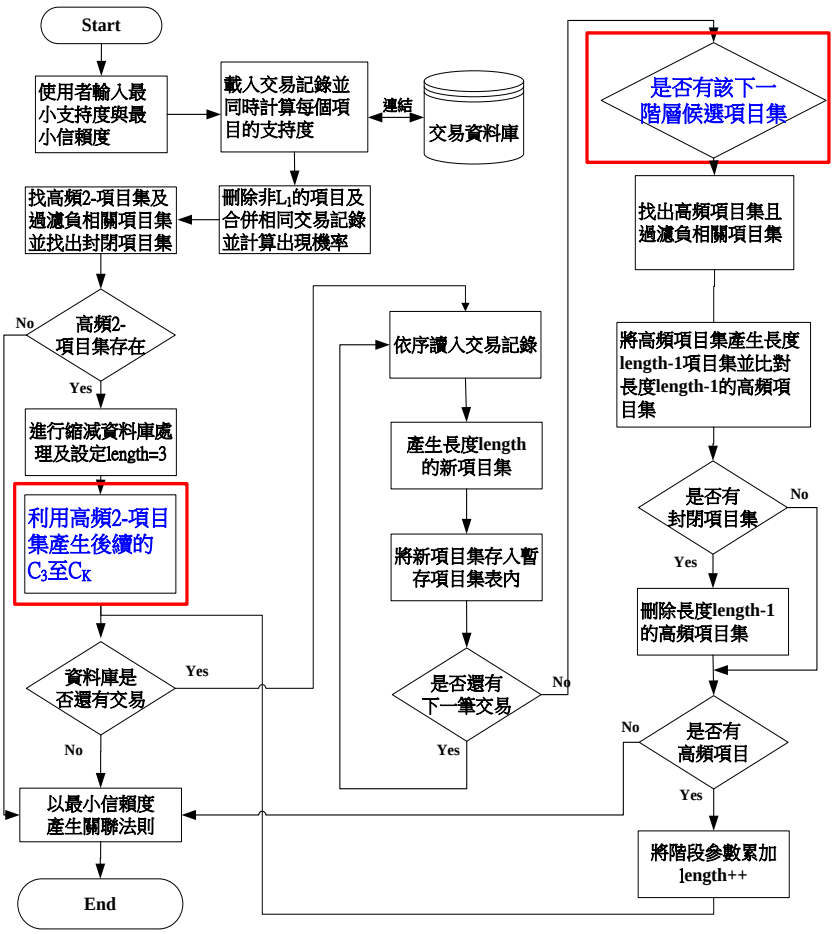

圖 6 PCP-M 演算法流程圖

```
Algorithm PCP-M                    //Java-like Pseudo-code
Input: DBName,                     //資料庫名稱
       min_sup,                    //最小支持度
       min_conf,                   //最小信賴度
Output:Rules                       //關聯規則
fetch data from Database and save in Dbdata;
scan Dbdata to get L1 and save in Large1 and L[0];   //Large1保留L1的出現機率
Delete non-L1 items in Dbdata and save in DB[0];
int maxLen = DataBase.get_maxLen();          //maxLen的初始值是DB[0]中記錄的最大長度
scan DB[0] to get L2 and save in L[1];
let L[0] compare with L[1] and delete closed itemsets in L[0];
                                             //過濾包含在封閉項目集中的L1
delete infrequent items in DB[0] and save in DB[1];
Int length=3,C_lenght=3;                     //C_length為階段長度
Int s=0;
C[0]=L2;
for (int i=0; i<maxLen-1; i++){              //以Candidate_Gen產生所有的候選項目集
      C[i+1]=Candidate_Gen(C[i],C_length);
      C_length++;
}
for (i=2; i<C_length; i++) {                 //階段拆解
      if (C[i]==null) break;
      for (j=0; j<DB[length-1].length; j++) {    //載入階段資料庫
            gd.setBase(DB[length-2][j], I, C[length-2]);//gd是GraDec class的object
            while (tmp = gd.getNext()!=null) {
                  temp.add(tmp);                 //將拆解項目存入
            }
      }
}
L[length-1]=Get_LargeItemsets(temp);         //找出滿足min_sup且正相關
let L[length-2] compare with L[length-1] and delete closed itemsets in L[length-2];
                                             //過濾包含在封閉項目集中的L[length-2]
if there exist no frequent itemsets then break;
DB[length-1]=Prun_DB(DB[length-2], C[length-1], length);//修剪交易記錄

length++;

}

get Rules from L[];

return Rules;
```

圖 7　PCP-M 演算法虛擬碼

肆、 效能評估

一、實驗環境

CPU： Intel Celeron 1.10 GHz

RAM： 512MB DDR RAM

OS： Microsoft Windows XP

資料庫來源： 由 IBM Data Generator[8]所產生的資料

開發工具： J2SDK 1.4.2

二、 資料庫參數說明

本研究實驗為求公正，實驗資料庫是由 IBM Data Generator 所產生；另外，本實驗所使用到的相關設定參數，如下說明：

T：每筆交易的平均長度。 (對應到的參數 -tlen)

I ：潛在高頻項目集的平均長度。 (對應到的參數 -patlen)

N：資料庫中商品項目的種類。 (對應到的參數 -nitems)

D：資料庫的資料筆數。 (對應到的參數 -ntrans)

而其它未加以說明的參數均使用預設值。

由於真實資料庫來源不易取得，因此，本研究採用微軟公司的 Microsoft SQL Server 2000 所附之 FoodMart(微軟公司，Microsoft Analysis services 的範本倉儲資料庫)FoodMart 跨國連鎖零售商店之範例資料庫來模擬真實交易資料庫的環境，且經過彙整後，再分別隨機抽取 5,000、10,000、15,000、20,000 與 25,000 筆交易記錄進行實驗模擬，以測得各演算法在模擬真實資料庫中的執行效能。

三、 測試的演算法

本研究提出一個新的演算法 PCP 主要是從 GRA 演算法的階段產生高頻項目集之核心概念，加上負相關項目集的過濾，再加上探勘封閉項目集的概念，因此以具有相同探勘封閉項目集概念的 CLOSET 演算法為本研究要比較的演算法。本論文以建構 FP-tree 為資料結構，並由 FP-tree 探勘出 Frequent Closed Itemsets 的概念，實作出 CLOSET 演算法(Pei et al.,2000)，因此要與原作者的數據做一個比較。如圖 8 即為原始文獻中的效能比較圖。針對所實作比較的演算法其公正性作說明，也就是針對 CLOSET 演算法來做驗證。驗證的方式則是以 Han 等所發表的文獻(Pei et al.,2000)中提到的數據來做比例的推算，圖 8 即為其中出現的數據圖。圖 8 是在資料庫為 T25I20N10KD100K 且 min_sup 為 0.7%、0.9%、1.1%、1.3%及 1.5%的情況下進行測試，測試環境為 Pentium 233MHz CPU、128MB RAM、在 Windows NT 平台以 Visual C++ 6.0 撰寫。而本研究則是以 J2SDK 1.4.2 撰寫此演算法，且在 Intel Celeron 1.1GHz、512GB RAM 及 Windows XP 環境下進行測試，也以 IBM data generator 產生一樣參數的模擬資料庫，測試結果如圖 9 所示。我們可將圖 8 及圖 9 的結果作比較，由於使用的語言及測試環境都不一樣，而本研究中實作之 CLOSET 演算法已儘可能達到其該有的執行水準。雖然這種比較方式可能不夠客觀，但是至少可看出本研究在實作上，確實已儘可能以最佳方式來實作 CLOSET 演算法，並無低估方式實作他人演算法的顧慮。因此 CLOSET 演算法在後續各實驗中所測出的數據仍有一定的公正性及可信度。

四、 實驗設計

本研究共設計九項實驗，分別用來測試 PCP 演算法與 CLOSET 演算法在不同的參數下對演算法結果的影響及 PCP-M 演算法在大型資料庫的效能。其中的參數包含最小支持度門檻值及第二小節中的四個參

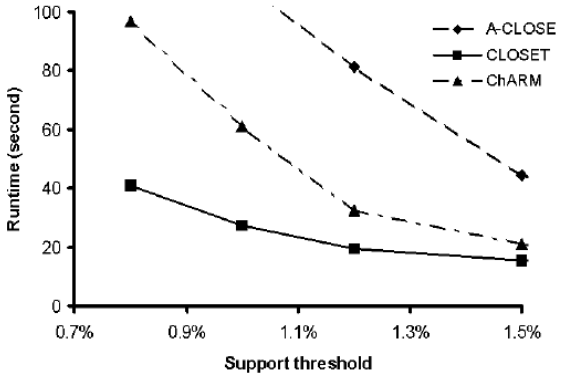

圖 8 原始文獻的 CLOSET 演算法之效能(資料來源 Han(Pei et al. ,2000))

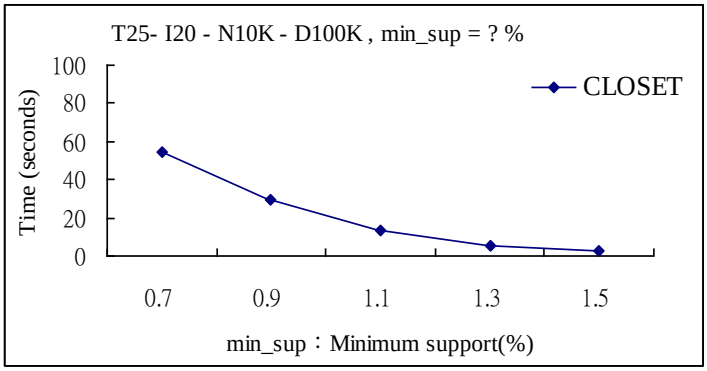

圖 9　本論文實作 CLOSET 演算法之效能

數。九項實驗分別如下：

實驗一：在不同最小支持度門檻值下，對演算法效能的影響

實驗二：在不同平均交易長度下，對演算法效能的影響

實驗三：在不同潛在高頻項目集平均長度下，對演算法效能的影響

實驗四：在不同項目種類下，對演算法效能的影響

實驗五：在不同資料筆數下，對演算法效能的影響

實驗六：以 FoodMart 模擬資料庫，在不同資料筆數下，對演算法效能的影響

實驗七：在不同資料筆數下，對演算法記憶體使用量之差異比較

實驗八：在大型資料庫下，對 PCP-M 演算法效能的影響

實驗九：以 retail 真實資料庫，在不同最小支持度門檻值下，對演算法效能的影響

實驗十：以 BMS-POS 真實資料庫，在不同最小支持度門檻值下，對演算法效能的影響

每個實驗的測試方法只變動所要測試的參數，而其它的參數都固定不變。

在每個實驗的測試方法裡，本研究只變動所要測試的參數且將固定其它的參數不變；另外，當演算法的效能差異太大時，可能會分為兩部份來測試，第一階段所有的演算法都會進行比較，第二階段則是進階測試，也就是將第一階段中效能較不好的演算法予以排除。每個實驗中的「執行時間」計算，皆是從資料庫載入完畢後開始，到找出所有的高頻項目集及個別出現的次數為止，也就是只測量各演算法核心所須花費的時間，並去除掉每個演算法載入資料庫的時間(因為載入的時間都一樣)。

五、 實驗分析

(一)實驗一：在不同最小支持度門檻值下，對各演算法效能的影響

在實驗一裡，將測試在不同最小支持度門檻值下，對 PCP 與 CLOSET 兩演算法效能的影響，同樣的，固定其它參數不變(T3-I2-N1K-D200K)，實驗結果如圖 10 所示：

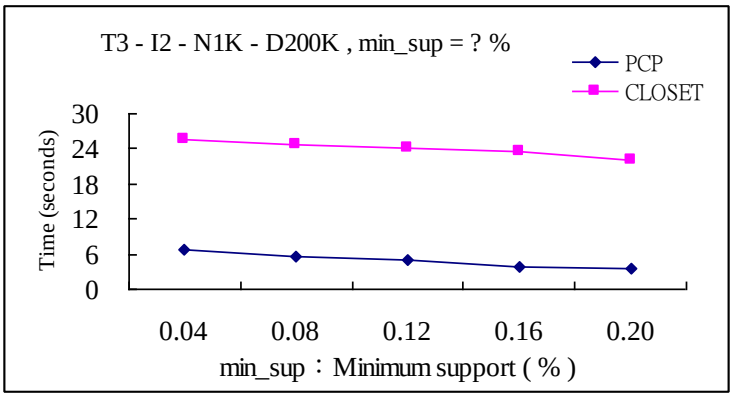

圖 10　最小支持度門檻值對效能的影響圖

由圖 10 可得知 PCP 與 CLOSET 演算法花費的時間成本皆隨著最小支持度的上升而平緩下降。而 PCP 演算法的效能穩定且較佳的原因，主要在於 PCP 演算法利用階段縮減機制的運作，可避免產生大量非必要的項目集，所以 PCP 演算法比 CLOSET 演算法呈現較佳的效能。

(二) 實驗二：在不同平均交易長度下，對演算法效能的影響

在實驗二中，將比較在不同的平均交易長度(參數 T)且最小支持度門檻值為 0.1%下，對 PCP 與 CLOSET 兩演算法的效能作比較。同樣的，其他參數固定不變(I2-N5K-D200K)，僅變動參數 T 的設定值，實驗結果如圖 11 所示：

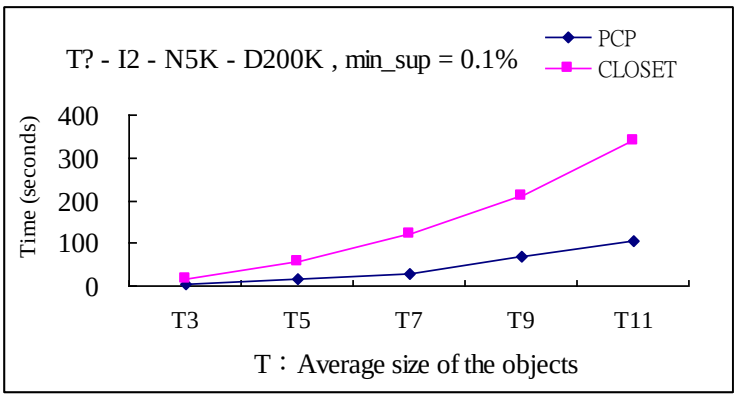

圖 11 平均交易長度對演算法效能的影響圖

由圖 11 可得知當平均交易長度(參數 T)愈長時，PCP 與 CLOSET 演算法所須花費的時間也隨之上升。因 CLOSET 演算法是以 FP-tree 架構為基礎，在平均交易長度增長時，所耗費的時間也隨之增加。反觀 PCP 演算法在平均交易長度增長時，效能呈現平緩上揚的情況。隨著平均交易長度的增長，更形拉開兩演算法效能的幅度差距。所以，PCP 演算法的效能在不同平均交易長度下優於 CLOSET 演算法。

(三) 實驗三：在不同潛在高頻項目集平均長度下，對演算法效能的影響

在實驗三中，將比較在不同的潛在高頻項目集平均長度(參數 I)且最小支持度門檻值為 0.3%下，對 PCP 與 CLOSET 兩演算法的效能作比較。同樣的，其他參數固定不變(T13-N5K-D200K)，僅變動參數 I 的設定值，實驗結果如圖 12 所示：

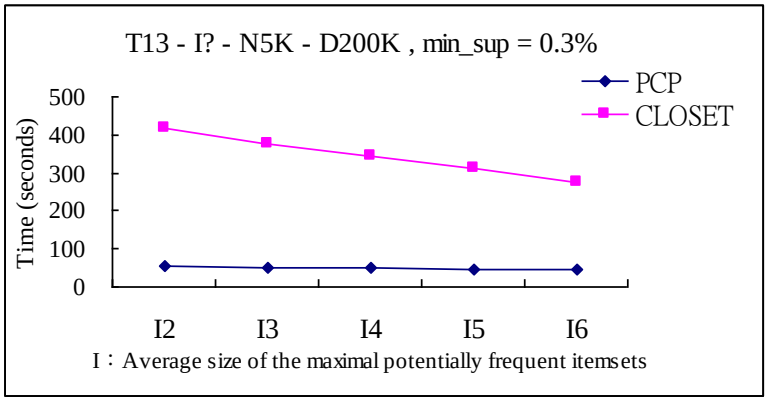

圖 12 潛在高頻項目集平均長度對效能的影響圖

由圖 12 可得知當潛在高頻項目集平均長度(參數 I)增長時，PCP 與 CLOSET 演算法的執行時間也隨之下降，因潛在高頻項目集平均長度增長時，其資料的緊密度也愈高，因此 CLOSET 演算法的執行時間也隨之下降，但相較 PCP 演算法在潛在高頻項目集平均長度增長時，在執行時間的耗用上的影響極小。所以，PCP 演算法的效能表現在不同潛在高頻項目集平均長度下優於 CLOSET 演算法。

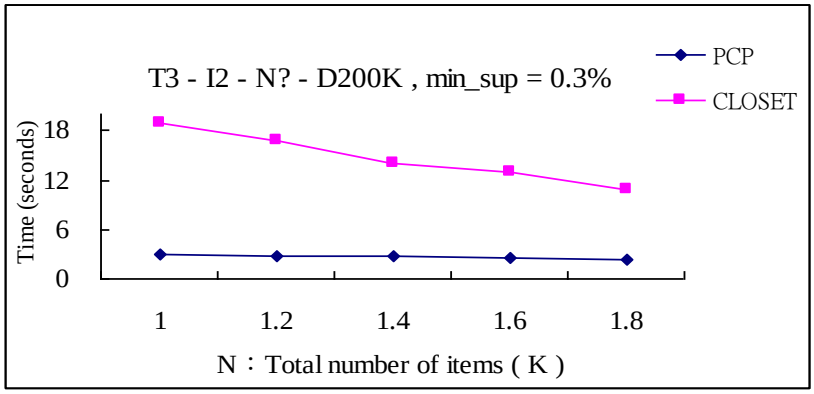

圖 13　項目種類對效能的影響圖

(四)實驗四：在不同項目種類下，對演算法效能的影響

在實驗四中，將比較在不同的項目種類(參數 N)且最小支持度門檻值為 0.3%下，對 PCP 與 CLOSET 兩演算法的效能作比較。同樣的，其他參數固定不變(T3-I2-D200K)，僅變動參數 N 的設定值，實驗結果如圖 13 所示。

由圖 13 可得知當項目種類(參數 N)愈多時，PCP 與 CLOSET 演算法探勘的時間將隨之減少，此乃因項目種類多時，每筆交易中出現的次數也相對的降低，使得相同的支持度下可過濾更多的項目，致探勘的速度較快之故。其中，PCP 演算法即使在項目種類較少的情況下，仍可避免產生大量的非必要項目集，因而在執行時間的耗用上影響極小，所以，PCP 演算法在不同的項目種類下較 CLOSET 演算的執行效能佳且穩定。

(五)實驗五：在不同資料筆數下，對演算法效能的影響

在實驗五中，將比較在不同的資料筆數(參數 D)且最小支持度門檻值為 0.3%下，對 PCP 與 CLOSET 兩演算法的效能作比較。同樣的，其他參數固定不變(T10-I5-N10K)，僅變動參數 D 的設定值，實驗結果如圖 14 所示。在進階實驗部份，將變動參數 D 的設定值，產生五個資料量較大的資料庫，以測試在資料筆數多時，對各演算法的執行效能的影響，實驗結果如圖 15 所示。

由圖 14 可得知當資料筆數(參數 D)愈多時，PCP 與 CLOSET 演算法探勘的時間是呈現遞增情況，資料筆數增加對於 PCP 演算法而言，只有些微的影響，而相較於 CLOSET 演算法而言，當資料筆數愈多時，其所花費的執行時間會隨之大幅的上升。所以，PCP 演算法在不同的資料筆數下較 CLOSET 演算的執行效能佳且穩定。

由圖 15 的進階實驗結果得知當資料筆數愈多時，PCP 與 CLOSET 演算法須花費的時間成本也隨之增加。而 CLOSET 演算法的建樹所花費的時間成本也就愈大，甚至因無法載入大型資料庫，而發生記憶體不足的情況。而 PCP 演算法因階段縮減機制的運作，即使在資料筆數較多時，仍能有效避免大量非必要項目集的產生，因此，在執行效能方面，PCP 演算法較 CLOSET 演算法要來得穩定且佳。

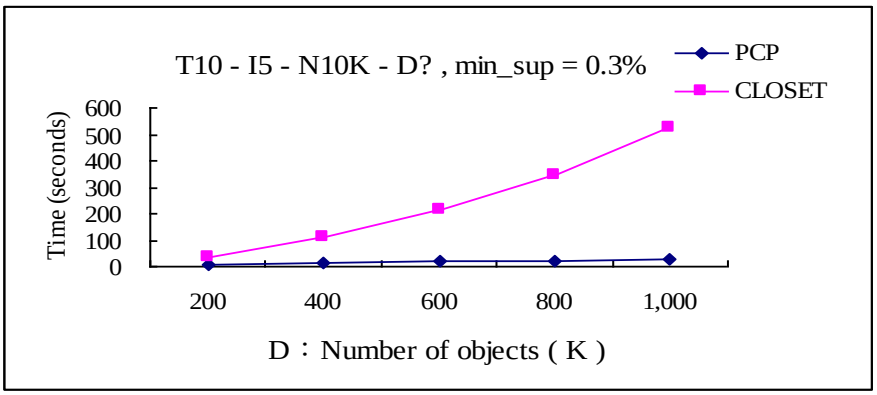

圖 14　資料筆數對效能的影響圖

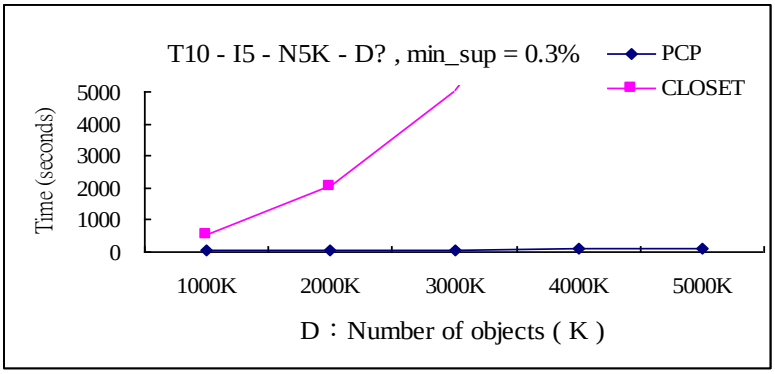

圖 15　資料筆數對效能的影響圖(進階實驗)

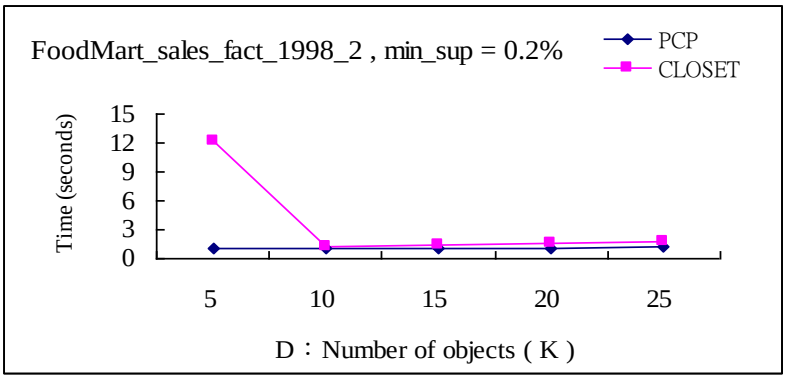

圖 16　資料庫筆數對效能的影響圖

(六)實驗六：以 FoodMart 模擬資料庫，在不同資料筆數下，對演算法效能的影響

在實驗六中，將使用 FoodMart(微軟公司，Microsoft Analysis services 的範本倉儲資料庫 FoodMart)的 sales_fact_1998_2 資料庫進行測試，其商品項目種類有 1,560 種，經過彙整後，交易記錄有 2,689 筆。本研究將隨機抽取 5,000、10,000、15,000、20,000 及 25,000 筆交易進行模擬實驗，由於 FoodMart 資料庫的項目及筆數較少且經過隨機抽取後，其資料的緊密性將會較高。比較在不同資料筆數且最小支持度門檻值為 0.2%下，對 PCP 與 CLOSET 演算法的效能作比較，實驗結果如圖 16 所示。

由圖 16 可得知當資料庫筆數愈多時，因資料的緊密性高，而 CLOSET 演算法是以 FP-tree 為基礎架構，因 FP-tree 的資料結構較適合緊密性高的資料，所以執行效能相當好。而 PCP 演算法因利用階段縮減機制的運作，可有效的在每一階層降低大量非必要項目集的產生，因而 PCP 演算法在資料緊密性較高的資料庫中的執行時間，亦能與 CLOSET 演算法以平緩穩定的狀態成長。在圖 16 中當資料庫筆數在 5K 時，因資料筆數愈少支持度門檻值將愈低，通過支持度的項目則愈多，所以 CLOSET 演算法需處理較多的項目集愈多，以致其執行時間呈現急速上升的情形，相較於 PCP 演算法無論資料筆數的變化，其執行時間皆能保持穩定的狀態，且在資料庫筆數增多時，其效能表現優於 CLOSET 演算法的效能將更明顯。

(七)實驗七：在不同資料筆數下，對演算法記憶體使用量之差異比較

在實驗七中，將比較在不同的資料筆數(參數 D)且最小支持度門檻值為 0.3%下，對 PCP 與 CLOSET 兩演算法所須耗用的記憶體空間作比較。同樣的，其他參數固定不變(T10-I5-N10K)，僅變動參數 D 的設定值，實驗結果如圖 17 所示。

由圖 17 可得知當資料筆數(參數 D)愈多時，PCP 與 CLOSET 演算法所產生的項目集將愈多，所需的記憶體使用量是呈現遞增情況，但 CLOSET 演算法耗用的記憶體數量大於 PCP 演算法，且隨著資料庫筆數的增加，CLOSET 演算法耗用的記憶體數量上升幅度更多。而 PCP 演算法在資料筆數增加時，會階段縮減交易的資料，避免產生大量不必要的項目集，因而記憶體的耗用是平緩微增的情況。所以在資料筆數不同時，PCP 演算法較 CLOSET 演算更節省記憶體的使用。

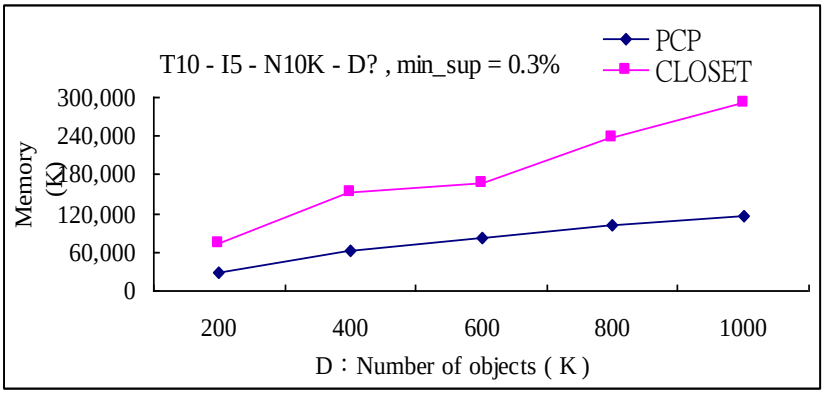

圖 17　資料筆數對演算法記憶體使用量差異比較圖

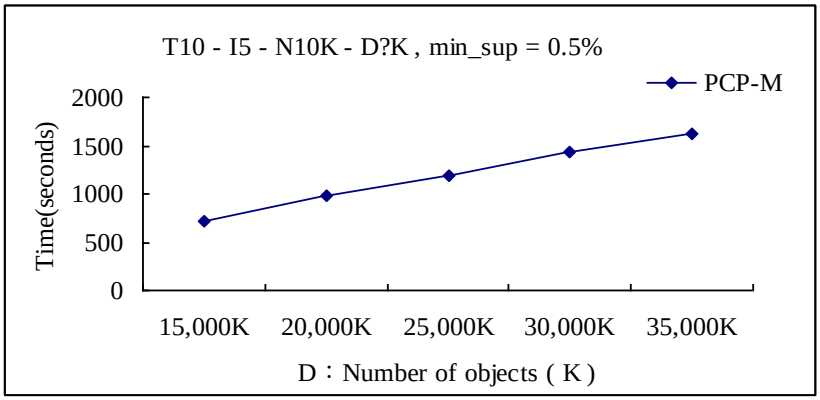

圖 18　資料筆數對 PCP_M 演算法效能影響圖

(八)實驗八：在大型資料庫下，對 PCP-M 演算法效能的影響

在實驗八中，欲測試當大型資料庫的容量大於記憶體的容量，而無法一次將資料庫讀入記憶體時，對 PCP-M 演算法執行效能的影響。因由於在實驗五的進階實驗結果得知 CLOSET 演算法在資料庫筆數 4000K 時，即已出現記憶體不足的情形，以致無法繼續進行更多資料筆數的實驗，因而本實驗唯以 PCP-M 演算法來進行實驗。本實驗使用 IBM[13]產生器產生出 15,000K、20,000K、25,000K、30,000K 及 35,000K 等不同資料筆數，並固定其它參數為 T10-I5-N10K，測試在不同資料筆數且最小支持度設為 0.5%時，對 PCP-M 演算法執行效能的影響，實驗結果如圖 18 所示。

由圖 18 可得知在記憶體受限的情況下，本研究所提出的 PCP-M 演算法仍能穩定及有效率的執行探勘工作，主要原因為 PCP-M 僅需進行四次實體 I/O 動作，也就是第一次找尋高頻 1-項目集、第二次刪除非高頻項目、第三次找尋高頻 2-項目集及第四次直接進行探勘產生高頻項目集等；而 PCP-M 演算法因階段縮減機制及階段產生高頻項目集的運作，即使在資料筆數較多時，仍能避免產生大量非必要的項目集，減少浪費非必要的時間成本。

(九)實驗九：以 retail 真實資料庫，在不同最小支持度下，對演算法效能的影響

在實驗九中，將使用 retail(http://fimi.cs.helsinki.fi/data/.)的資料庫進行測試，資料來自於比利時的零售商店的銷售記錄，銷售商品項目種類有 16,470 種，經過挑選後，其交易記錄有 88,163 筆。將比較在不同的最小支持度門檻值下，對 PCP 與 CLOSET 兩演算法在效能上作比較。實驗結果如圖 19 所示。

由圖 19 可得知 PCP 與 CLOSET 演算法花費的時間成本皆隨著最小支持度的上升而下降。而當支持度門檻值愈小時，CLOSET 演算法發揮了 FP-tree 在資料結構的優勢，所以在效能上的表現比 PCP 演算法的效能好。而當支持度門檻值愈大時，PCP 演算法利用階段縮減機制的運作，可避免產生大量非必要的項目集，所以 PCP 演算法比 CLOSET 演算法呈現較佳的效能。

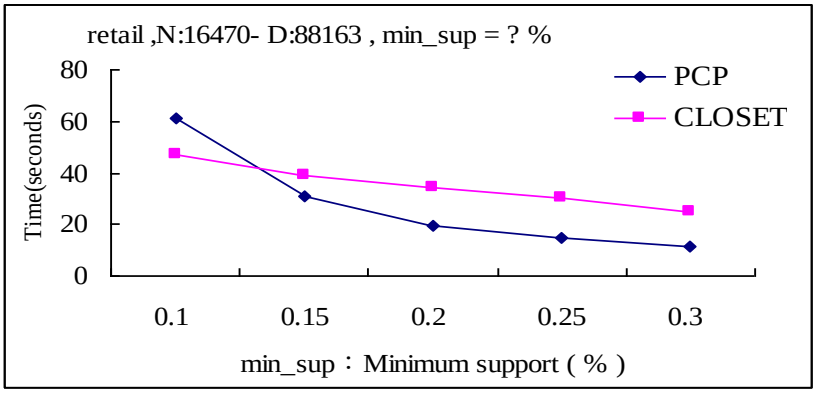

圖 19　最小支持度門檻值對效能的影響圖

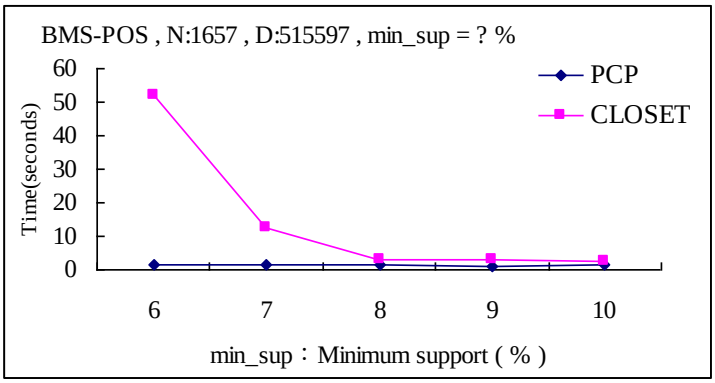

圖 20 最小支持度門檻值對效能的影響圖

(十)實驗十：以 BMS-POS 真實資料庫，在不同最小支持度下，對演算法效能的影響

在實驗十中，將使用 BMS-POS(http://fimi.cs.helsinki.fi/data/.)真實資料庫來進行測試，該資料庫的項目種類有 1,657 種，且交易記錄有 515,597 筆，針對此真實資料庫在不同支持度下各個演算法的影響進行模擬實驗，此真實資料的平均交易長度較長，將比較在不同的最小支持度門檻值下，對 PCP 與 CLOSET 兩演算法在效能上作比較。實驗結果如圖 20 所示。

由圖 20 中可看出兩個演算法的花費時間都隨著支持度的上升而下降， CLOSET 演算法因需一直遞迴建立 conditional FP-tree，所以在支持度愈小時其效能更明顯的不佳，而本研究所提出之 PCP 演算法因本身的過濾機制，可避免產生大量非必要候選項目集，其效能確實比 CLOSET 演算法優越。

伍、 結論

本研究所提出之 PCP(Positive Correlation and Closed Itemsets by Phase)演算法，在產生項目集的方法上，乃利用階段拆解的前序項目集是否是前一階段的高頻項目集，若成立則繼續以該前序項目集來拆解出其他的子項目集，否則終止以該前序項目集來拆解出子項目集的程序，避免產生過多不必要的項目集數量，降低在拆解不必要的項目集所需的時間。其次在每一階段縮減資料庫的處理中，計數階段產生的高頻項目集中每個項目出現的次數，將小於下一階段長度減 1 的項目自資料庫中刪除，除了減少產生不必要的項目集外，也節省了記憶體不必要的耗用。其三採用過濾每階段高頻項目集為負相關項目集的方法，以刪減相關程度不高的項目集，進一步減少項目集數量。其四是找出封閉項目集，以階段的高頻項目集比對前一階段的高頻項目集，若前者的子項目集與後者相同且支持度也相同時，即可刪除前一階段的高頻項目集，進一步的減少高頻項目集的數量及精簡關聯規則。總結來說 PCP 演算法有以下優點：

一、 利用刪減負相關項目集，進一步過濾不重要的項目集。

二、 利用階段拆解的特性，每一階的高頻項目集只須與上一階的高頻項目集做封閉項目集(Closed Itemsets)的比對，減少大量比對的時間。

三、 由縮減後的高頻封閉項目集推導出精簡的關聯規則。

未來可以再更進一步的研究方向為：

一、 由於實務上的資料庫取得不易，正相關的資料特性適用於資料項目間相關性高的資料庫。若能取得實際應用中的資料檔案，將有助於取得的規則比一般的關聯規則準確度更高，在預測上更準確，挖掘更出多隱含、有價值的資訊。再者，對演算法的關聯規則而言，也因現實生活中資料的緊密度比起 IBM generator 的資料更高。在規則的精簡上會有更突出的表現。

二、 加利用正相關與封閉項目的特性運用在順序性資料之探勘，探討連續資料項目間的相關程度，提升探勘有價值的規則。譬如統計學、生物學、醫學等方面的探勘，進一步發揮資料探勘的技術並帶給人們更多的貢獻。

參考文獻

沈清正、陳仕昇、高鴻斌、張元哲、陳家仁、黃琮盛與陳彥良 (2002)。資料間隱含關係的挖掘與展望，*資訊管理學報，9*(S)，75-99。

黃仁鵬、熊浩志與郭煌政 (2004/12/4)。**直覺拆解之關聯法則演算法-IDA**。第十屆資訊管理暨實務研討會，台灣，台中。

黃仁鵬、熊浩志 (2005)。**快速資料探勘演算法與相關應用**(碩士論文)。南臺科技大學資訊管理研究所，台南市。

黃仁鵬、藍國誠 (2006)。高效率探勘關聯規則之演算法-GRA，*電子商務學報，8*(4)，469-498。

賈坤芳、劉家銘 (2001)。**利用負相關線上挖掘關聯式規則**(碩士論文)。國立中興大學資訊科學研究所，台中市。

微軟公司 (2000)。**Microsoft Analysis services 的範本倉儲資料庫 Food Mart**。

Agrawal ,R., .Imielinski ,T, and Swami, A. (1993). *Mining Association Rules Between Sets of Items in Large Databases,* In proc. of the ACM SIGMOD Conference on Management of Data, 207-216.

Agrawal, R., and Srikant, R. (1994). *Fast algorithms for mining association rules,* Proceedings of 1994 International Conference on Very Large Data Bases, 487-499.

Brin, S., Motwani, R., and Silverstein, C. (1997). *Beyond market baskets: generalizing association rules to correlations*, Proc. of the ACM SIGMOD, 265-276.

Han, J., Pei, J., and Yin, Y. (2000). *Mining Frequent Patterns without Candidate Generation*, Proc. ACM SIGMOD Int. Conf. on Management of Data, 1-12.

Kim, W.-Y., Lee, Y.-K., and Han, J. (2004). *CCMine: Efficient Mining of Confidence-Closed Correlated Patterns,* Proc. 2004 Pacific-Asia Conf. on Knowledge Discovery and Data Mining (PAKDD'04), 569-579.

Lancaster, H. O. (1969). *The Chi-squared Distribution,* New York: John Wiley & Sons.

Lin D. and Kedem, Z. M. (1998). *Pincer Search: A New Algorithm for Discovering the Maximum Frequent Set*, Proceedings of the 6th International Conference on Extending Database Technology: Advances in Database Technology, 103-119.

Meo, R. (2000). Theory of dependence values, *ACM Trans. Database System*, *25*(3), 380–406.

Park, J. S., Chen, M. S. and Yu, P. S. (1995). *An Effective hash-based Algorithm for Mining Association Rules*, Proceedings of the ACM SIGMOD Conference on Management of Data－SIGMOD'95, 175-186, .

Pasquier, N., Bastide, Y., Taouil, R., and Lakhal, L. (1999). *Discovering frequent closed itemsets for association rules,* In Proc. 7th Int. Conf. Database Theory (ICDT'99), 398–416.

Pei, J., Han, J., and Mao, R. (2000). CLOSET: An efficient algorithm for mining frequent closed itemsets, retrieved from: https://www.cs.sfu.ca/~jpei/publications/dmkd00.pdf

Seno, M. and Karypis, G. (2001). *LPMiner: An Algorithm for Finding Frequent Itemsets Using Length-Decreasing Support Constraint*, Proceedings of the 2001 IEEE International Conference on Data Mining(ICDM), 505-512.

Wang, J., Han, J., and Pei, J. (2003). CLOSET+: Searching for the Best Strategies for Mining Frequent Closed Itemsets, Proc. ACM SIGKDD Int. Conf. on Knowledge Discovery and Data Mining (KDD'03), 236-245.

Zaki, M. J. and Hsiao, C.-J. (2002). *CHARM: An efficient algorithm for closed itemset mining*, Proceedings of the 2002 SIAM International Conference on Data Mining.