漸進式區塊深度優先關聯法則探勘之研究

游坤明
中華大學資訊工程學系

王子健
中華大學資訊管理學系

王冠傑
中華大學資訊管理學系

摘要

有鑑於傳統關聯法則之探勘方法，需要耗費大量時間來完成資料之探勘，過去雖有學者提出漸進式探勘架構，不過仍然無法避免舊有資料庫重複掃瞄。因此本論文提出一個運用項目資料結構與區塊深度優先之探勘策略，只需對交易資料庫進行一次掃瞄，建立探勘程序使用之資料結構，可避免反覆掃瞄資料庫，並且在產生關聯法則時，只需要針對必要項目進行比對。此外針對漸進式資料之動態資料庫，透過本演算法所提出的漸進探勘機制，利用過去探勘所記錄之資訊，可以避免對舊有資料進行重複掃瞄完成資料探勘。本論文並針對傳統演算法，利用實際的資料進行探勘效能之實驗，並且進行效能比較與分析。透過實驗結果顯示，本論文提出之演算法可以節省大量的探勘時間。

關鍵字：資料探勘、關聯式法則、漸進式探勘

A Study of Incremental Block Depth First Search Technique on Association Rule Mining

Kun-Ming Yu
Department of Computer Science and Information Engineering, Chung Hua University

Tzu-Chien Wang
Department of Information Management, Chung Hua University

Kuan-Chieh Wang
Department of Information Management, Chung Hua University

Abstract

Data mining technique and application has received a lot of attention in the past decade. And finding out the association rules among data is one of the hot topics of data mining. By applying data mining technique, we can get valuable information from large size of raw data efficiently. But with the evolution of computer technology, the data grow constantly in time and the time spent in finding the valuable information is growth sharply. Therefore, how to design an efficient data mining scheme is extremely important. This paper focuses on the important issue and proposes an I-BDFS (Incremental Block Depth First Search) algorithm to resolve the problem. In I-BDFS algorithm, the raw data only needed to be scanned once instead of reduplicate of database scanned in previous algorithms. The proposed algorithm also can quickly generate large itemset by necessary intersection item, so the algorithm can save lots of execution time when mining. Moreover, with the help of designed structure in I-BDFS, the proposed algorithm need only to mine the necessary specific patterns to save scanning and comparison time. At last, in the paper, we conduct several experiments with real data to evaluate the performance of I-BDFS as well as some traditional algorithms. And the experimental results show that I-BDFS algorithm indeed has better performance compared with those traditional algorithms.

Key words: Data Mining, Association Rules, Incremental

壹、緒論

隨著資訊科技快速發展，除了電腦軟硬體的效能不斷提升、數位資料儲存空間的大量增加，連同網際網路的蓬勃發展，讓使用者可以輕易且便利地儲存大量的資料，使得過去只能以紙張紀錄的資料，逐漸轉變以數位的型態來儲存。同時，伴隨而來多元化的資訊技術應用，除了帶給人們更多的便利，還為企業帶來的無限商機，舉凡從企業內部流程的改造、到對外的供應鏈關係管理，都因為資訊科技有了重大的改變。

其中一個關鍵的基礎在於儲存媒介的改變。透過電腦儲存媒介與資料庫系統的運用，資料的儲存與呈現，從傳統的紙本開始大量以數位的方式儲存於電腦當中。舉凡個人身份資料、企業的交易資料、股市交易資料，全都可以儲存於電腦系統當中，並且透過資料庫系統的搜尋與檢索，使用者可以方便地使用所需要的資料，使得資料庫系統受到廣泛的應用。本論文所探討的資料探勘，就是資料庫眾多應用的技術之一。

資料探勘是為了發現有意義之模式或規則，以自動或半自動的方式，對大量資料進行探勘、分析進行的流程 [3]。從大量且散亂的資料中挖掘、萃取出有用的資訊，提供使用者能夠根據挖掘的資訊來制定決策或解決問題。資料探勘有許多不同的應用，也可應用於不同的領域，企業組織可以運用資料探勘來分析顧客的消費行為，進而對顧客進行有效的產品廣告促銷；政府機關可以藉由資料探勘技術的輔助，對民眾的生活習慣進行分析來提供更便民的行政服務。使用者還可以運用資料探勘技術來協助醫療臨床數據的分析[6,7,15]，來產生有用的醫療資訊。

資料探勘的領域當中，依照探勘的技術方法大致上可以分成以下六類[3]：分類（Classification）[14]、分群（Clustering）、回歸（Regression）、關聯法則（Association rule）[8,21]、因果（Sequence）、時間序列（Time series）[9]。本論文所要探討的主題就是資料探勘領域中的「關聯法則」為主要的研究範圍。關聯法則是資料探勘中相當重要的技術之一，旨在找出資料之間的關聯性。關聯法則的運用也相當廣泛，最典型的應用就是賣場的購物車分析。目前在關聯法則經常被討論的，就是在大量資料的探勘工作下執行效率的問題。在一般的情況下，關聯法則的探勘方式是採用透過項目組合的方式，然後不斷檢查掃瞄資料，來判斷項目之間的關聯性。雖然這樣可以產生正確有用的結果，不過隨著資料量不斷成長，電腦系統掃瞄資料的工作的負擔也會越加沈重，導致每次探勘都必須耗費越來越冗長的時間，才能完成工作。如果每次探勘都需要耗費冗長的時間才能取得資訊，就無法滿足使用者即時性的需求，不符合現在環境的需要。

貳、文獻探討

關聯性法則的模型是由Agrawal[8]等學者在1993年提出的一個模型，它用來呈現資料中特定項目同時發生的關聯性，也就是說在發生項目A的情況下，同時也會發生項目B

的可能性。如果要找出項目之間的關聯，必須透過將資料庫中的資料項目進行組合、驗證，然後產生關聯法則。

關聯法則模型首先假設資料存放於資料庫DB，DB裡面存放著過去的交易紀錄，每個交易記錄以TID作為辨識的欄位，並且每筆交易記錄會記錄一筆或多筆的資料項目Item1,Item2,…,ItemN，由個別項目組成的項目集合稱為itemset，itemset I則是該交易紀錄中所有項目的集合，所有的項目都會包含於I中。資料探勘的一開始就是從眾多的項目集合I中，找出經常發生的項目集，用意在於判斷項目之間關聯性是否足夠頻繁，當發生的次數夠多，就代表此項目集可能具有某些存在的意義。為此，在探勘過程當中必須設定一個門檻值，來判斷某一個項目集合發生的頻率符合是否門檻值的依據。一般而言，我們會以支持度（Support）來稱呼門檻值。當某項目集發生的次數高於支持度，就稱該項目集為高頻項目集（Frequent Pattern）。所以，在交易記錄T的項目集合I當中，同時存在A、B兩個項目，如果交易記錄T所在資料庫DB當中發現的頻率高於支持度的設定，那麼就會判定候選項目AB成為高頻項目集，項目A與B具有關聯性。然後透過條件機率來檢驗此高頻項目的信賴度（Confidence）檢驗，也就是說在出現A項目的情況下，B項目同時也會發生的機率，我們以Support（A∩B）／Support（A）來表示，在符合支持度之後也符合信賴度，那麼這兩個項目的關聯，就可以成為有意義的關聯法則。

關聯法則探勘的工作大致上可以分成兩個階段，第一個階段是從大量的資料中尋找符合支持度的高頻項目集合（Frequent Patterns），第二的階段就是將第一個階段產生的高頻項目集合，進行信賴度驗證的工作。由於第一個階段必須利用繁複的工作去掃瞄資料，並且透過不斷組合、刪減、驗證來產生高頻項目集。因此第一階段成為探勘關聯法則最費時的地方，而第二個階段是利用第一階段的結果，推演出關聯法則。所以在改善關聯法則探勘效能的議題上，主要也是針對改善第一個部分，如何有效產生高頻項目集的工作上。

過去有Agrawal [8]等學者對關聯法則探勘進行單獨的說明，並且其有學者[18,20]對各種關聯法則演算法進行較詳細的整理與效能的比較。關聯法則探勘演算法中最耳熟能詳且影響深遠的就是Apriori演算法。Apriori演算法透過k-itemsets產生長度k+1的候選項目集，並且透過掃瞄資料庫產生長度k+1的高頻項目集。然而，該演算法最為人詬病的就是，在探勘的過程中，產生多少數量的候選項目集，就必須反覆掃瞄資料庫多少次，才能得到探勘的結果，造成效能的低落。因此有許多演算法根據Apriori演算法進行改進，以減少產生候選項目集的數量，得到縮短探勘時間的目的。不過由於Apriori-like演算法仍然擺脫不了候選項目集過多且必須重複掃瞄資料庫的缺點，因此有學者提出直接採取額外的探勘資料結構搜尋與不同的探勘策略，來進行更有效率的探勘工作如FP-Growth [16, 17]演算法，期望得到更好的探勘效能。但是FP-Growth演算法由於面臨大量資料的探勘，可能產生記憶體無法一次承載大量資料，引發虛擬記憶體的不斷存取，使得探勘效能降低，於是Han等人於2004提出Parallel以及Projection的觀念來[17] 來有效降低探勘過程中記憶體之需求。R. Dass[13]根據過去之文獻[19]，提出改良之區塊優先搜尋之探勘演算法BDFS(b)，以有效之探勘策略減少磁碟機之使用。即使面對大量資料，仍可以有效率地完成探勘工作。

此外，由於現在資料會隨著時間不斷增加，為了維持探勘結果的正確性，使用者就必須重複進行資料探勘的工作。在這樣的情況下，有學者提出漸進式的關聯法則探勘架構[11,12]，透過儲存過去探勘的資訊，減少舊有資料重複掃瞄的工作，進而節省舊有資料掃瞄所需要的時間。然而由於上述演算法在漸進探勘的過程中，仍然必須反覆掃瞄資料庫，因此探勘效能仍然十分低落。更糟的是，面對漸進資料探勘，項目集可能在高頻與非高頻屬性之間跳動，造成該類項目集在漸進探勘的過程中，必須被反覆計算，造成時間上的浪費。針對這樣的問題已有學者[1]提出準大項目集的觀念，以另外訂定低於最小支持度的門檻值，藉由保留高頻與準高頻項目集，來減少項目集在高頻與非高頻間跳動的問題。

莊文宗[4] 在2005根據FP-Growth演算法提出改良之漸進探勘演算法IDFPBT，採用該演算法少量掃瞄資料庫之優點，在面對漸進資料之探勘，更可以避免舊有資料之重複掃瞄。大幅提昇探勘之效能。不過該演算法仍然沒有考量記憶體與探勘資料量之問題，同時面對漸進資料的探勘，可能發生資料在高頻與非高頻之間發生跳動，使得該演算必須付出大量時間來進行資料結構之維護，造成探勘效能下降的可能。

參、研究方法

對於關聯法則探勘，本研究根據過去的演算法[13]進行演算架構的改良，稱為漸進式區塊深度優先關聯法則探勘演算法(Incremental Block Depth First Search, I-BDFS)演算法。I-BDFS演算法的探勘流程，依照探勘類型可以分為初次探勘或漸進式探勘方法。詳細的探勘流程，如圖1所示。

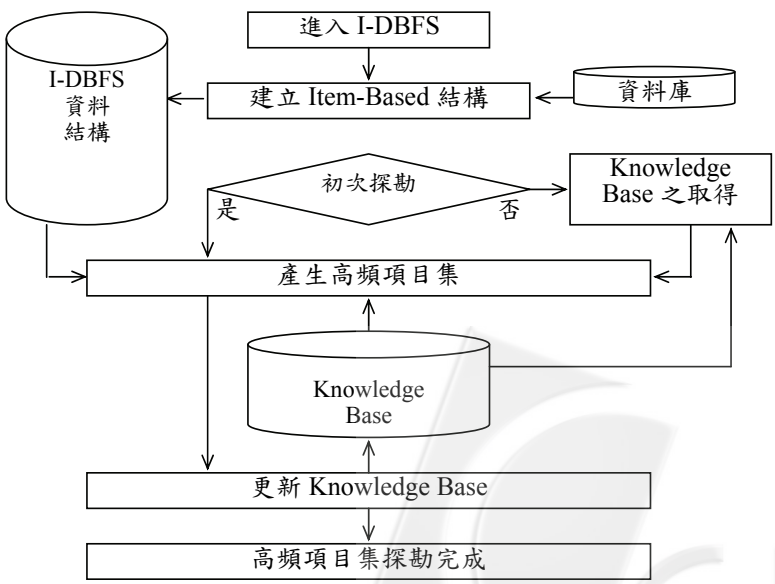

圖1：I-DBFS探勘流程

一、儲存探勘資料之資料結構

在正式產生高頻項目的工作開始之前，I-DBFS演算法會先對資料庫進行一次掃瞄，建立記錄長度為1和2項目集在資料庫中發生次數的二維矩陣M中，如圖2，以及儲存項目集名稱、項目發生次數和該項目TID List的資料結構，如表1所示。

透過二維矩陣，演算法可以快速找出一階與二階項目集發生的次數，此時配合使用者設定支持度的篩選，就可以直接得到一階與二階的高頻項目集合（Frequent Patterns），並且記錄於F(1)與F(2)中，F(n)則代表長度為n的高頻項目集合。

透過項目資料結構，演算法可以快速得到項目集的發生次數以及TID List 資訊，只要將項目集的子集合透過該TID List的比對，就可以快速得到結果，而不用反覆掃瞄資料庫或資料結構。例如要得到項目集abc的資訊，我們利用其子項目集ab與bc的TID List的比對(見表5)，就可以得到項目集abc的次數為3、TID List為1, 8, 11。

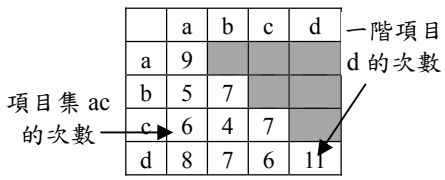

圖2：二維矩陣M

表1：項目資料結構

Item	count	TID List
a	9	1, 2, 3, 6, 7, 8, 10, 11, 12
b	7	1, 4, 5, 6, 7, 8, 11
c	7	1, 2, 4, 8, 10, 11, 12
d	11	1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11

二、區塊深度優先之探勘策略

進入高頻項目探勘，I-DBFS演算法採用區塊深度優先的探勘策略，並且依照探勘的資料長度與工作內容可以切割成不同的探勘深度狀態（Mining Depth State）。透過前處理我們可以得到一階與二階的高頻項目集，並且運用二階高頻項目集我們可以產生三階候選項目並放入GP（Global Pool）中，我們將到目前為止的工作深度狀態定義為S_0（State 0）。如果GP中存在候選項目集，就進行深度狀態S_1的區塊探勘工作。假設GP存在數量龐大的三階候選項目集，I-DBFS先取出部分的三階候選項目集放入區塊中進行次數探勘工作，然後產生有計算價值的四階候選項目集。如果有產生四階候選項目集就進入S_2的區塊探勘工作，對四階候選項目進行檢查與候選項目集產生的工作，直到無法繼續下去為止。接著回到S_1繼續從GP抓取候選項目集進行項目集的探勘與產生的工作。

依照探勘資料的特性可能產生不同程度的探勘深度狀態，關於探勘過程可能產生的探

勘狀態，以下圖3所示，依照探勘策略會依序產生b_0、b_1、b_2、b_3、b_4、b_5、…的區塊探勘工作，同時依序對應於S_0、S_1、S_2、S_1、S_2、S_2、…不同的探勘深度狀態，流程如圖3所示。

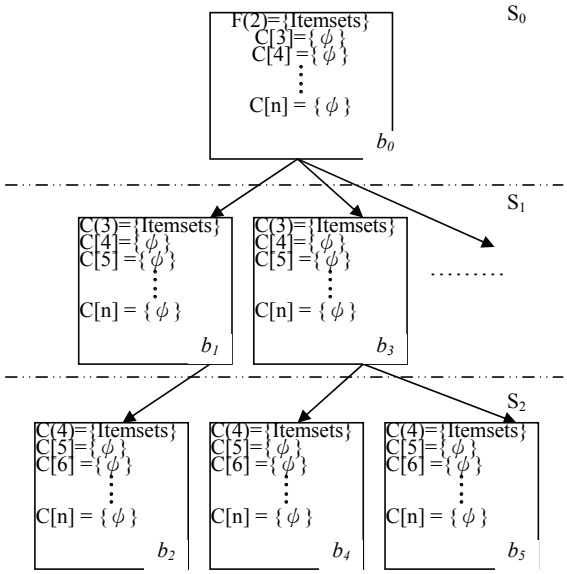

圖3：區塊深度優先探勘策略架構圖

三、區塊內項目集之處理工作

區塊中項目集之搜尋、組合與判定的工作會透過GP與BS（Border Set）緩衝區來進行。BS存放區塊高頻項目集組合產生的更高階項目集合，同時會記錄該項目集合次子集合高頻項目集的數量；GP裡存放等待計數的候選項目集。對於區塊項目的處理架構如圖4所示。

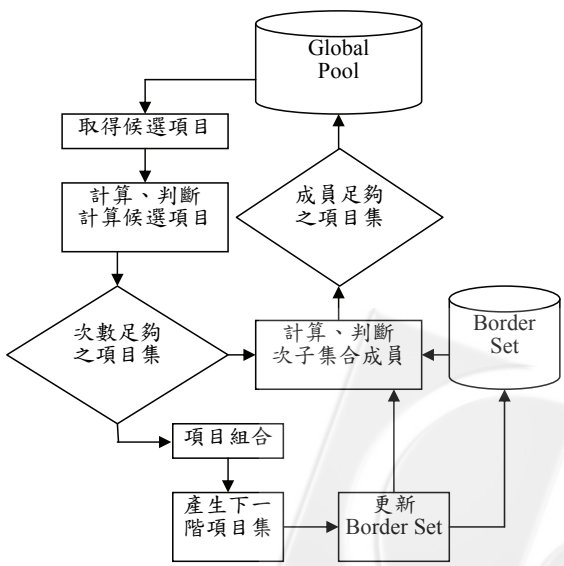

圖4：區塊工作架構圖

區塊作業的開始會先從GP抓取候選項目集，透過項目資料結構的TID List比對得到區塊中各個候選項目集的發生次數後，將不符合支持度門檻值的項目集進行刪減。接著將留下來的高頻項目集與BS中的項目集進行比對，判斷這些高頻項目集是否是BS中項目集的子集合之一，並且更新該項目集的次高頻子集合數量。當BS中某一項目集的次高頻項目子集合的數量等於該項目集的長度時，就將該項目集從BS移到GP當中成為有計數價值的候選項目集。

在區塊中項目集處理的工作中，如果可以藉由區塊中的高頻項目集不斷產生更高階的項目集，同時區塊的項目集處理工作就會持續下去，直到無法產生更高頻的項目集，此一區塊的工作就到一段落，並且依照相同的方式進行下一次的區塊工作。

四、項目集之判斷工作

關聯法則中，存在於眾多資料裡的項目集可以分成兩個種類，高頻與非高頻項目集，而I-DBFS演算法因為採用準大項目集的漸進探勘策略，因此可以將項目集分成三類：高頻、準高頻與非高頻項目集三種。同時依照項目集在於舊有、新進資料庫存在的狀況，可能存在以下九種情形，如表2所示，項目集發生情況一的情形就表示該項目集屬於新進資料的高頻項目，同時也屬於舊有資料的高頻項目集；發生情況二的情形就表示該項目集屬於新進資料的高頻項目，而屬於舊有資料的準高頻項目集；其他情形則依此類推。如果本次探勘作業屬於初次探勘，或者是不存在舊有探勘資料，將只會發生情況三、六與九的項目情形。

表2：項目集可能存在於新舊資料庫之狀況

		新進資料		
		高頻	準高頻	非高頻
舊資料	高頻	情況一	情況四	情況七
	準高頻	情況二	情況五	情況八
	非高頻	情況三	情況六	情況九

判斷項目集在更新資料中類型的工作中，根據該項目集存在於新舊資料庫的情形，可以進行不同方式的計數策略。關於項目集的情形與計數方法如圖5所示。在大部分的情況下，透過知識庫（Knowledge Base, KB，稍後將對KB進行說明）的使用，演算法只要透過簡單的步驟，就可以判斷該項目集屬於更新資料的高頻、準高頻或是非高頻項目集。只有少部分情況需要透過TID List的比對，才能得到該項目集的次數，不過由於不用重複掃瞄資料庫，因此依然能夠快速地完成工作。

在情況一、二、四、五、七、八的情形下，由於該項目集在舊有資料庫屬於高頻與準高頻項目集，因此會將該項目集的名稱、次數與TID List記錄於KB中，因此得到項目集在新進資料的資訊後，演算法只要透過KB的使用，利用簡單的次數加總就可以得到該項目集在更新資料庫發生的次數，也就可以快速判斷項目集的類型。

在情況九的情形下，由於該項目集在無論在舊有資料或新進資料都不屬於高頻或準高頻項目集，因此它在更新的資料中，也一定不會成為高頻、準高頻項目集，因此演算法就沒有從KB取出該項目集並且計算其次數的必要。除非本次探勘不是初次探勘同時本次探勘所設定的支持度低於以往探勘設定的潛在支持度，在這樣的情況下，則有可能因為支持度降低，使得該項目集提升為準高頻甚至是高頻項目集。在這樣的情況，就必須從KB中取出該項目集的兩個次子項目集進行TID List的比對來得到該項目集的資訊。

比較特殊的狀況為情況三與六，該項目集在這兩個情況下表示它並不屬於舊有資料的高頻或準高頻項目集，可是卻屬於新進資料中的高頻、準高頻項目集。在可能的情況下，該項目集可能因為新進資料的高發生率，使得該項目集成為更新後的高頻或準高頻項目集。不過由於該項目集在舊有資料庫屬於非高頻項目集，為了節省空間，因此它的項目資訊不會儲存於KB之中。如果要得到該項目集在舊有資料庫的資訊，就必須透過其次子項目集合來得到該項目的資訊。

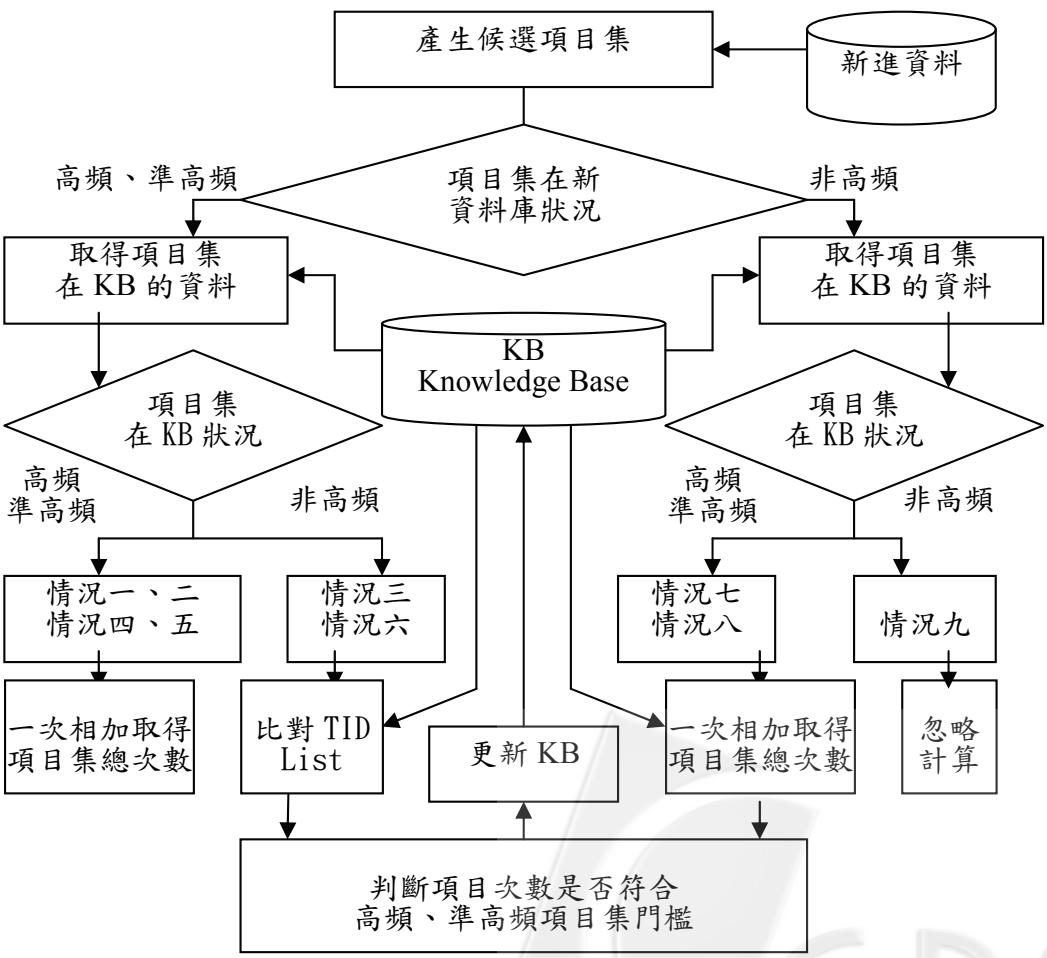

圖5：項目集計數策略架構

五、知識庫(Knowledge Base)

在I-BDFS演算法探勘的工作中，會先從資料庫中讀出知識庫（Knowledge Base, KB）資訊以幫助探勘作業的進行，如果為初次探勘，則內容設為ψ (空集合)。此外隨著探勘的進行會將必要的資訊進行保留並且更新KB之中。KB內含的資訊包含過去探勘項目所有的一階項目以及隨著探勘進行產生的高頻、準高頻項目集的名稱、次數、與TID List，內容如圖6所示。

透過儲存這些資訊，I-DBFS演算法可以有效避免以下探勘工作所耗費的時間。

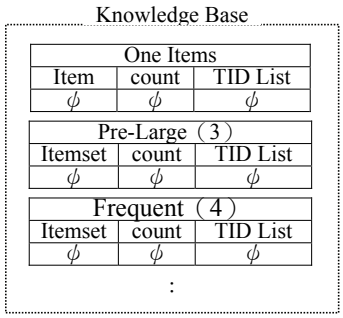

圖6：Knowledge Base內容架構

（一）重複TID List比對的工作：

隨著探勘的深度增加，項目集的長度會越來越長。過去雖然運用項目資料結構可以對項目集所擁有的一階子項目集進行TID List可以避免其他不必要項目的掃瞄，不過隨著項目集的長度越長，必須比對的次數就會越來越多次，長度為n的項目集，就必須對其n個一階子項目集進行n次的TID List比對。此外當演算法必須計算長度 n的候選項目集的次數時，就表示其所屬k個k-1階的子項目集合都屬於高頻項目集，也就代表過去也必須對該子集合進行相同一階子集合的TID List比對的工作。這就表示在計算更高階項目集的過程中，進行了重複TID List比對的工作。雖然可以在記憶體中快速完成TID List的比對，不過仍是不必要的計算工作。

I-BDFS演算法透過KB記錄高頻、準高頻項目集的資訊，要產生計算k候選項目集的次數，只要針對其長度k-1的兩個次子項目集合進行一次TID List比對的工作，就可以得到該項目集的資訊，省去反覆一階項目比對的工作。

（二）舊有資料庫的重複掃瞄：

透過運用KB的漸進式探勘，在進行漸進資料的探勘時，由於KB以利於探勘的格式記錄了舊有一階項目的資訊，因此在再次探勘的過程中，演算法可以不用對舊有資料進行再一次的掃瞄。並且進行資料轉換的工作。

（三）非必要新、舊資料項目集的運算：

運用KB記錄舊有高頻、準高頻的特性，在對新進資料進行探勘時，透過KB的幫

助，得知目標項目集在舊資料庫中的存在情形判斷，是否有必要對該筆新進項目集進行更深入的組合與探勘。反之，也可以在新進資料探勘的過程中，是否因為資料的增加而失去高頻項目集的水準。

六、I-BDFS演算法

I-BDFS演算法探勘的過程會使用變數來幫助探勘的進行，以下對演算法所使用的輸入與輸出所使用的變數進行說明。DB：存放舊有探勘資料的資料庫；db：存放新進資料之資料庫；KB：過去探勘過程得到的高頻項目集資訊；PL：探勘過程得到的準高頻項目集；b：使用者設定之區塊大小；min_sup：使用者設定高頻項目集之最低支持度；may_sup：使用者設定準高頻項目集之支持度。

I-BDFS Algorithm

Input: DB, db
Output: The frequent patterns
Description:
1. Pre-Process ()
2. generate 3-itemsets from 2-itemsets in PL and put them into global pool which's number of immediate pre-large subsets = 3
3. update KB
4. while (global pool is not empty)
5. block_manipulation(KB, block of candidate patterns)
6. end of while
7. output all itemsets in PL which hava support >= min_sup
8. store KB in database
9. end of algorithm

Pre-Process Function

Input: DB, db, KB
Output: I-table、PL、KB
Description:
1. if DB is not null then
2. Load KB
3. else
4. Initiate KB as null
5. end of if
6. Scan db once to get all 1-item and its' support.
7. Create matrix M, which stores support of all 1-item and 2-itemsets.
8 Create item structure of I-table contain item name, item support and tid list.
9. Search M to get frequent patterns of PL whose support >= may_support
10. update KB(all 1-items in I-table , pre-large 2-itemsets in PL)
11. return

Block_Manipulation(candidate patterns) Function

Input: KB, block of candidate patterns
Output: Any length of pre-large patterns
Description:
1. For each candidate patterns in b
2. get support by intersecting it's own two immediate subsets' tid_list from I-table
3. if sum of the itemset support got in step2 and in KB >= may_sup then
4. move this itemset to PL
5. else
6. prun this itemset from block
7. end of if
8. end of for
9. generate pattern of next higher length from newly got PL patterns in step 3 to 7
10. put patterns got in step9 into border set
11. check the immediate pre-large sub-itemsets number in border set with PL
12. for each itemset in border set has a number of immediate pre-large subsets equals to it's length then
13. move it from border set to global pool with decreasing order by length
14. end of for
15. update KB(new got patterns in block process)
16. return

Update_KB (pre-large patterns) function

Input: pre-large patterns
Output:Updated KB
Description:
1. For each pattern
2. if pattern not exists in KB then
3. add this pattern in KB
4. else
5. update pattern's support and tid_list
6. end of if
7. return

（一）初次進行探勘演算推導

設定資料探勘的高頻支持度為交易筆數的33.3%，準高頻為交易筆數的25%，每個區塊能夠處理的資料量為4個項目集。以下開始初次探勘的流程。舊有交易資料內容如表3：

表3：探勘交易資料表

TID	Item	TID	Item
1	a b c d e	7	a b d
2	a c d e	8	a b c d
3	a d e	9	d e
4	b c d e	10	a c d e
5	b d e	11	a b c d e
6	a b d	12	a c e

將掃瞄原始資料庫一次，計算所有一階項目的名稱、次數以及所擁有的TID list，並且整理出記載兩兩項目次數的二維矩陣M，如圖7，以及Item-Based基礎的資料結構，存放於I-Table_1，如表4：

	a	b	c	d	e
a	9				
b	5	7			
c	6	4	7		
d	8	7	6	11	
e	6	4	6	8	9

圖7、項目次數之二維矩陣

表4：一階項目資訊I-Table_1

Item	count	TID List
a	9	1, 2, 3, 6, 7, 8, 10, 11, 12
b	7	1, 4, 5, 6, 7, 8, 11
c	7	1, 2, 4, 8, 10, 11, 12
d	11	1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11
e	9	1, 2, 3, 4, 5, 9, 10, 11, 12

目前為止總交易筆數為12筆，因此在高頻支持度33.3%和準高頻支持度25%的水準下，每個項目集發生的次數分別要在3.996和3筆以上，才能夠成為高頻項目及準高頻項目。在完整掃瞄資料庫的一階項目後，除了建立探勘需要的資料結構外，透過矩陣M同時可以得到一、二階準高頻項目PL（1）與PL（2）。接著透過I-Table_1產生PL（2）中二階準高頻項目集的TID List，並且記錄於I-Table_2中，如表5所示：

表5：二階項目資訊I-Table_2

Itemset	count	TID List
ab	5	1, 6, 7, 8, 11
ac	6	1, 2, 8, 10, 11,12
ad	8	1, 2, 3, 6, 7, 8, 10, 11
ae	6	1, 2, 3, 10, 11, 12
bc	4	1, 4, 8, 11
bd	7	1, 4, 5, 6, 7, 8, 11
be	4	1, 4, 5, 11
cd	6	1, 2, 4, 8, 10, 11
ce	6	1, 2, 4, 10, 11, 12
de	8	1, 2, 3, 4, 5, 9, 10, 11

透過PL（2）可以產生三階項目集，項目集ab與ac可以組成的項目集abc，將abc丟入BS（3）中，並且將項目集abc的子項目集數量設定為1。接著項目集ab與ad可以組成的項目集abd，將abd丟入BS中，並且將項目集abd的子項目集數量設定為1。而項目集bc與ac可以組成的項目集abc已存在於BS(3)中，因此直接將項目集abc的子項目集數量加1，…依此類推。組合完畢之BS中的各個項目集，如表6：

表6：目前border set BS（3）內容

Itemset	次子集合數量
abc	3
abd	3
abe	3
acd	3
ace	3
ade	3
bcd	3
bce	3
bde	3
cde	3

其中{ abc, abd, abe, acd, ace, ade, bcd, bce, bde, cde }的次子集合數量都等於項目集的長度，因此移至GP成為候選項目集，等待分配至區塊（block）進行計數的工作。

由於設定每個區塊只能處理四個項目集，因此先從GP挑選四個項目集{ abc, abd, abe, acd }，並且丟入區塊b來進行探勘的工作。

第一步，利用I-Table_2分別對abc, abd, abe, acd進行計數的工作。以abc為例，從I-Table_2找到項目集ab與bc，利用ab與bc兩個項目集的TID List進行比對的工作，而得到abc的次數為3，TID list為{ 1, 8, 11 }。依此類推，得到區塊中所有項目集的資訊，如表7所示：

表7：區塊b目前的項目集資訊

Itemset	count	TID List
abc	3	1, 8, 11
abd	5	1, 6, 7, 8, 11
abe	2	1, 11
acd	5	1, 2, 8, 10, 11

其中abe低於準高頻支持度的門檻值，因此將abe移除，而abc, abd, acd都高於準高頻項目的支持度，因此將這些項目集記錄於PL（3）當中，成為準高頻項目集。然後再將這些項目集進行組合，產生更高階的項目集，並且計算該項目集此子集合的數量，並且放入BS（4）當中。由於區塊b產生的新的項目集abcd的此子集合數量小於它的長度4，因此無法從BS（4）中移到GP候選項目集。此區塊在無法產生更高頻候選項目集的情形下，就開始進行下一個區塊的探勘，並且進行相同的區塊處理工作。

當GP內沒有候選項目集、區塊也沒有項目集可以進行計算，透過組合與計數的探勘工作就到此結束，而挖掘高頻、準高頻項目集的工作就近乎完成了。透過從之前的探勘程序，得到了各階層的準高頻項目集，如表8所示：

表8：各階層準高頻項目集

階層	準高頻項目集內容（次數）
PL（1）	a（9）, b（7）, c（7）, d（11）, e（8）
PL（2）	ab（5）, ac（6）, ad（8）, ae（6）, bc（4）, bd（7）, be（4）, cd（6）, ce（8）, de（8）
PL（3）	abc（3）, abd（5）, acd（5）, ace（5）, ade（5）, bcd（4）, bce（3）, bde（4）, cde（5）
PL（4）	abcd（3）, acde（4）, bcde（3）

現在透過簡單篩選的動作，只要PL中任何項目集的次數高於高頻支持度3.996次，就將這些項目集從PL中移至F而成為高頻項目集。篩選之後，可以得到最終的高頻項目集，如表9所示，產生高頻項目集的工作也告一段落。最後只要依照高頻、準高頻項目集的長度將KB保存於資料庫中即可。

表9：各階層高頻項目集

階層	高頻項目集內容（次數）
F（1）	a（9）, b（7）, c（7）, d（11）, e（8）
F（2）	ab（5）, ac（6）, ad（8）, ae（6）, bc（4）, bd（7）, be（4）, cd（6）, ce（8）, de（8）
F（3）	abd（5）, acd（5）, ace（5）, ade（5）, bcd（4）, bde（4）, cde（5）
F（4）	acde（4）

（二）漸進資料之探勘演算推導

在一段時間之後，有資料新增至資料庫中，資料內容如表10所示，於是必須開始更新的探勘工作，以下開始進行當有新增資料時，進行的探勘步驟，本次探勘所使用的高頻、準高頻支持度與初次探勘的支持度相同，維持在33.3%與25%，不過由於增加了12筆交易資料，高頻、準高頻的門檻值分別提升成為7.992與6筆。

表10：新進交易資料

新進交易資料			
TID	Item	TID	Item
13	a b c d e	19	a b d
14	a c d e	20	a b c d
15	a b e	21	d e
16	b c d e	22	a c d e
17	b d e	23	b c d e
18	a b d e	24	b e f

由於並非初次探勘，因此在正式進入探勘程序之前，會先將前一次探勘存放於資料庫的探勘資訊取出，根據其階層類別分別存放於KB中。KB建置完畢後開始正式的探勘作業。

根據前處理產生新進資料的二維矩陣和資料結構I-Talbe_1，我們快速得到六個一階項目項目{ a（7）, b（9）, c（6）, d（10）, e（10）, f（1）}，接著分別對這些個項目和KB進行的搜尋，如果某一項目存在於KB中，就將該項目的次數相加，例如項目a的次數就是9+7=16次；如果某一項目不存在於KB中，那麼就將該項目的次數加0，例如項目f的次數就是1+0=1。然後對該項目相加後的次數進行判斷，如果次數高於準高頻支持度次數6，那麼就將該項目移入一階準高頻項目PL(1)中。上述工作進行完畢後，我們可以得到一階準高頻項目PL(1)= { a（7）, b（9）, c（6）, d（10）, e（10）}。根據PL（1）各個項目的組合，與二維矩陣的使用，我們可以快速產生二階項目集及所屬發生次數，記錄於I-Table_2。

以上述之方式，配合KB檢查I-Table中項目之次數，篩選出高頻、準高頻項目集，並且產生下一階之候選項目集，記錄於下一階層之I-Table中。判斷候選項目集之工作，一樣採取區塊深度優先之探勘策略，只是在項目集判斷的過程，會加入上述之KB運用技術，來得到更新資料後某項目集是否屬於高頻項目集。當GP裡沒有項目集、區塊也沒有項目集可以作業，我們無法從準高頻項目產生更高階的項目集時，透過組合與計數的探勘工作就到此結束。

到目前為止，我們得到了各階層的準高頻項目集的新增資料。接著，將I-table_1或PL中的準高頻資料附加於KB中的項目集資料之後，並且更新項目集次數。如果存在於I-table或PL中的項目集同時存在於KB之中，那麼就將該項目集的次數相加，並且將I-table或PL該項目的TID list附加於KB該項目的TID list之後；如果存在於I-table_1或PL中卻不存在於KB之中，那麼就直接將該項目新增至KB之中。最後將存在於長度在2以上的任何項目集進行篩選的動作，只要項目集的次數低於準高頻支持度6次，就將這些項目集從KB中移除，結果如表11所示：

表11：更新後的KB

階層	項目集名稱（次數）
KB（1）	a（16）, b（16）, c（13）, d（21）, e（18）, f（1）
KB（2）	ab（10）, ac（10）, ad（13）, ae（11）, bc（8）, bd（14）, be（11）, cd（12）, ce（11）, de（15）
KB（3）	abd（8）, acd（8）, ace（8）, ade（8）, bcd（8）, bce（6）, bde（9）, cde（10）
KB（4）	acde（7）, bcde（6）

最後再進行一次篩選，只要將KB中任何的項目集高於高頻支持度7.992，就將該項目集移至F而成為高頻項目集，內容如表12，就是我們所關心的結果。然後將KB依照其階層分別將項目集名稱、次數、TID list存放於資料庫，以利下次探勘使用。

表12：更新資料的高頻項目集合

階層	項目集名稱（次數）
F（1）	a（16）, b（16）,c（13）, d（21）, e（18）, f（1）
F（2）	ab（10）, ac（10）, ad（13）, ae（11）, bc（8）, bd（14）, be（11）, cd（12）, ce（11）, de（15）
F（3）	abd（8）, acd（8）, ace（8）, ade（8）, bcd（8）, bde（9）, cde（10）
F（4）	無

肆、實驗評估

本研究針對關聯法則高頻項目集的探勘，提出了I-BDFS演算法，並且運用I-BDFS演算模式實際開發出高頻項目集的演算模組。本節將利用醫療生化的檢驗資料為實驗數據，對過去的演算法與I-BDFS演算法進行實際探勘的工作，並且統計各演算法進行資料探勘所耗費的時間，並且加以分析與比較，以驗證本論文提出的I-BDFS演算法，的確能夠有效提升資料探勘的效率。本研究進行之實驗平台如下：

- 硬體環境

  中央處理器：Intel Pentium 4 1.5 GHz

  記憶體：512 MB DDR RAM

  磁碟存取：160 GB IDE Hard Disk

- 軟體環境

  作業系統：Microsoft Windows 2003 Enterprise Edition

  資料庫系統：Microsoft SQL 2000 Server with service pack 4

  軟體開發工具：Microsoft Visual Basic .NET

本研究使用的資料是與北部某地區教學醫院合作，利用該醫院於民國90年度的醫生門診資料進行探勘。探勘採用的資料是從醫生進行門診診斷所記錄而來。在一個病患的看診中，醫生可能根據病患的症狀、各樣生化檢驗資料，對其下達多種症狀的診斷，而這些診斷中並沒有前後順序。門診診斷資料記錄了病例號碼、看診日期、看診順序、診斷順序、診斷代碼、診斷說明、醫師代碼、病患性別、科別等看診資料。實驗資料之數量如表13所示：

表13：實驗資料之數量

月份	單月資料量（筆）	累積資料量（筆）	月份	單月資料量（筆）	累積資料量（筆）
一	67204	67204	七	76685	522295
二	65997	133201	八	78148	600443
三	78535	211736	九	67850	668293
四	74667	286403	十	78156	746449
五	81564	367967	十一	74323	820772
六	77643	445610	十二	74453	895225

由於本論文主要探討的議題是關聯法則高頻項目集探勘演算法的改進，並且經過探勘程序產生的資料關聯性與分析的工作，必須透過醫院的專家來進行探討，才能說明探勘的結果對於醫療研究上的貢獻。因此本論文將依照資料探勘的幾個應用時機或模式，在探勘結果正確無誤的前提下，進行演算法的效能評估與特性分析。為了達到上述目的，本論文設計了以下三個實驗：

一、尋找適當探勘支持度之前置實驗

關聯法則探勘必須針對資料特性採用適當的支持度，當支持度門檻過高，可能會造成無法產生關聯法則之情況；當支持度門檻過低，則可能造成過多無意義法則之產生。因此，在進行各個演算法效能比較之前，會先針對探勘的資料進行不同支持度水準下的預先探勘，然後根據探勘的結果來設定適當的探勘支持度。

此外由於本論文提出之演算法使用準高頻項目集之漸進式探勘策略，為發揮演算法最佳之探勘效能，將在相同高頻支持度水準之下，進行不同準高頻項目集支持度探勘之實驗，並且根據探勘的效能找出適合的準高頻項目集支持度。

本實驗首先對本論文所使用的實驗資料，分別在支持度4.5%、3.0%、1.5%、1.0%、0.5%、0.3%、0.1%的水準進行探勘，並且得到高頻項目內容如表14所示：

表14：不同支持度，高頻項目集之數量

月份 / 高頻	高頻項目集數量											
	1	2	3	4	5	6	7	8	9	10	11	12
	4.5%											
一階	6	6	6	6	6	6	6	6	6	6	6	6
	3.0%											
一階	13	12	12	11	10	10	10	10	10	10	10	10
	1.5%											
一階	31	33	35	35	35	34	33	33	34	32	32	33
二階	2	1	1	1	1	1	1	1	1	1	1	1
	1.0%											
一階	49	48	48	49	48	48	47	48	48	49	48	48
二階	5	4	4	4	4	4	3	3	3	2	2	2
	0.5%											
一階	86	90	89	89	91	93	96	97	100	100	101	101
二階	30	30	32	31	30	30	28	28	28	28	29	29
三階	1	1	1	1	1	1	1	1	1	2	2	2
	0.3%											
一階	131	133	137	140	143	145	149	150	150	150	150	149
二階	56	54	51	51	49	47	46	48	44	45	47	47
三階	3	5	5	5	5	5	5	6	6	6	5	5
四階	0	0	1	1	1	1	1	0	0	1	1	1
	0.1%											
一階	258	268	263	271	269	270	275	275	276	277	276	278
二階	255	241	248	273	259	261	262	254	252	246	249	254
三階	22	21	26	27	26	26	25	24	22	21	23	24
四階	1	1	1	1	1	1	1	1	1	1	1	1

由實驗數據得知，在支持度1.5%的水準之下，才有較多的高頻項目集產生，也才能貼近實際的探勘環境。因此本研究將針對支持度在低於1.5%的水準下，進行下一個實驗探勘效能之實驗。

此外，本論文使用準高頻項目集來提升漸進探勘的效能，因此在進行資料探勘前，必須找出適合的準高頻項目集支持度。但是探勘目標資料屬性的不同，準高頻項目集支持度的訂定對於探勘效能會有不同的效果，因此準高頻項目集支持度的訂定也就沒有一定的標準。因此，在此將會進行訂定準高頻項目集支持度之前置實驗。實驗的方法是在高頻支持度1.0%的水準下，分別以高頻支持度的100%、80%、60%與40%當作準高頻項目集支持度，並且以本論文提出之演算法來進行漸進資料的探勘。最後從中找出最適當的準高頻項目集支持度。實驗的數據如圖8所示。

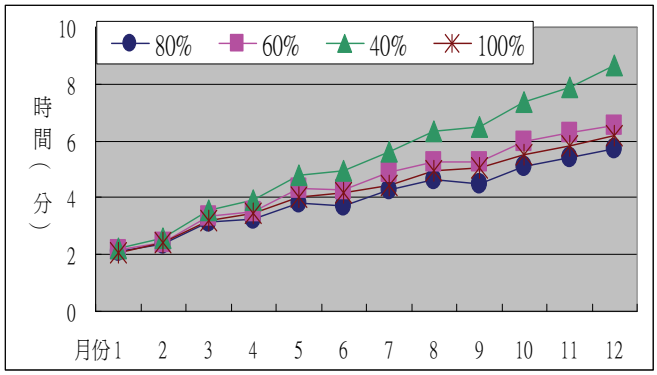

圖8：不同準高頻支持度之探勘效能

從實驗數據得知，將準高頻項目集支持度設定在高頻項目集支持度的80%，可以得到較好的探勘效能，因此在以下的實驗，I-BDFS演算法的準高頻項目集支持度將設定為高頻支持度之80%。其中將準高頻項目集支持度與高頻項目集支持度設為相同時，也就如同不使用準高頻項目集一樣。在這樣的情況下，就會有項目集隨著漸進資料的增加，而在高頻與非高頻項目集間不斷跳動。因此會對這些項目集產生計算的工作，造成時間增加。而如果將準高頻項目集之支持度設定在60%（含）以下，可能由於準高頻項目集的門檻較低，造成必須計算的項目集增加，使得處理準高頻項目集的工作增加，因此即使節省了在高頻與非高頻之前跳動的項目集的計算，不過由於必須處理過多的準高頻項目集，造成探勘時間增加的反效果。

二、探勘資料量改變對於演算法效能之影響

在不同資料量下，I-BDFS執行效率之變化是一個相當重要的問題，因此，本節將討論探勘資料量改變對於不同演算法效能之影響，實驗步驟為，根據支持度在1.5%、1.0%與0.5%的水準下，分別將BDFS（b）、IDFPBT與本論文提出之I-BDFS演算法進行三次

完整不同支持度的探勘，並且記錄探勘過程所耗費的時間。對於支持度固定、資料逐月增加的情況下，將不同演算法花費的探勘時間以折線圖呈現，方便探討在相同的支持度水平下三種演算法在效率上的差異。以下將支持度分別在1.5%、1.0%與0.5%的比較圖分別呈現於圖9、圖10、圖11。

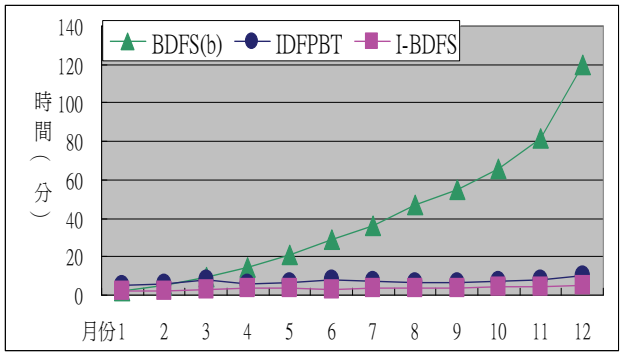

圖9：支持度1.5%，漸進探勘效能之比較

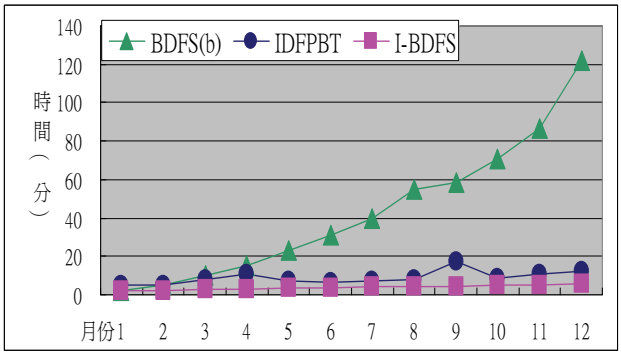

圖10：支持度1.0%，漸進探勘效能之比較

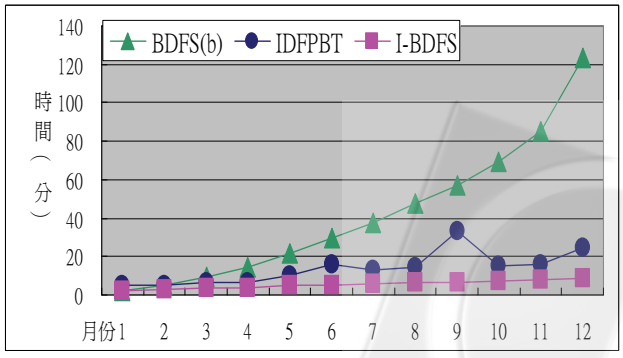

圖11：支持度0.5%，漸進探勘效能之比較

從圖9、圖10、圖11中，我們可以發現，由於BDFS(b)演算法不具備漸進式探勘的能力，因此當進行漸進式探勘工作時，由於資料量越來越多，每一次所要處理的資料就越加龐大，使得探勘的時間會急速增加，因此本研究僅針對具有漸進式探勘功能的IDFPBT與I-BDFS演算進行以0.3%、0.1%低之支持度水準，來測試演算法在低支持度之探勘效能。兩個演算法探勘所耗費之時間比較如圖12、圖13所示。

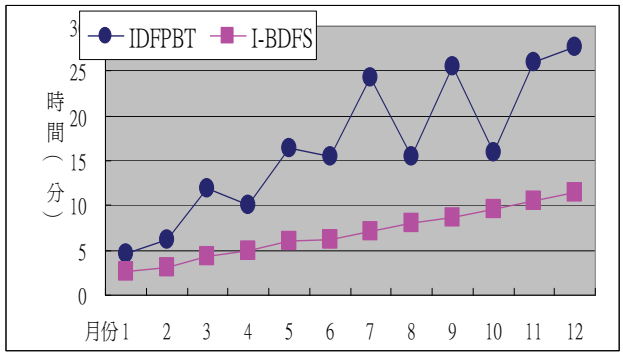

圖12：支持度0.3%，漸進探勘效能之比較

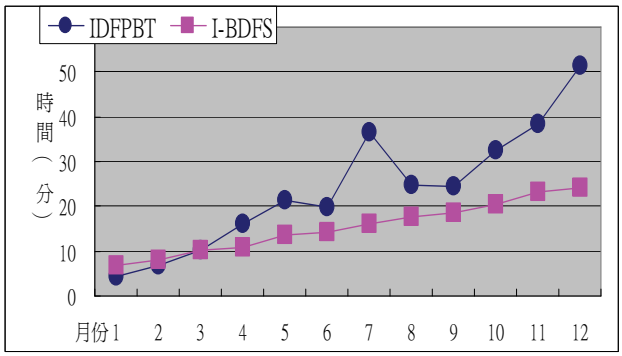

圖13：支持度0.1%，漸進探勘效能之比較

由圖12、圖13中，我們可以發現在相同的支持度水準之下，當資料量逐漸增加，無論是IDFPBT或是I-BDFS探勘所需要時間也會越來越長。但IDFPBT演算法雖然具備漸進探勘的功能，不過在漸進資料探勘的過程中，由於必須因應項目資料屬性的跳動，不斷維護其樹狀結構，造成探勘時間較長且不穩定，反觀本研究提出之I-BDFS演算法，能夠以穩定且有效率的方式完成探勘的工作，由此可以證明本論文提出的演算法具備相當良好的效能。

三、支持度改變對於演算法效能之影響

漸進資料探勘是以有效率的方式對漸進的資料進行探勘的工作，而改變支持度對舊

有資料重複探勘，屬於應用資料探勘技術可能發生的工作。透過漸進資料探勘技術的應用，將過去探勘資訊有效記錄與使用，將使得重複探勘的工作更有效率。反之，如果不採用漸進式探勘技術來重複探勘舊有資料，重複探勘的過程就如同進行全新探勘工作一般，必須對舊有資料進行完整的掃瞄與計算來得到探勘的結果。

本實驗是在固定的資料量下，改變支持度水準，對過去舊有的資料進行重新探勘的工作。這個實驗的進行類別與結果可以分為三個部分，在原本探勘的支持度為1%的情況下，重新探勘後的支持度：

一、支持度提高0.5%成為1.5%；

二、支持度不變，維持1%；

三、支持度降低0.5%成為0.5%。

對於上述實驗數據根據不同資料量，對於支持度改變探勘所需要的時間，以長條圖來表示之，如圖14所示。

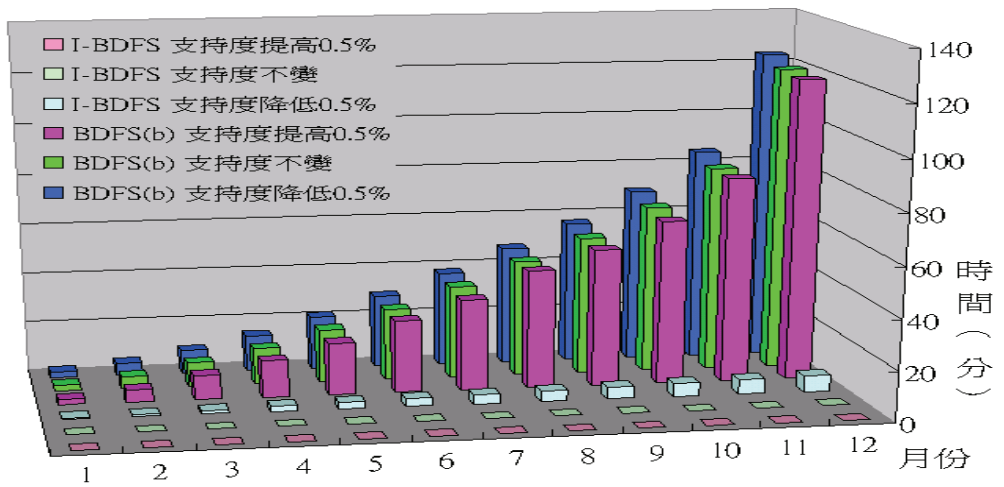

圖14：改變支持度，對舊有資料重複探勘結果

由圖13我們可以明確的看出來，I-BDFS在針對舊有資料重複探勘的所需要的時間都相當的少，即使是資料量不斷提升，I-BDFS仍然保持相當優異的探勘效率。會有如此顯著的差異是因為DBFS（b）的演算法在資料探勘完之後，並不會對過去的資料進行保存與有效的利用。反之，演算法I-BDFS因為具備漸進式的演算架構，重複探勘的過程中，如果支持度提高、沒有改變或者是支持度下降的幅度並沒有低於準高頻項目集的門檻值，I-BDFS只要針對過去探勘保存的KB進行簡單支持度篩選的動作，就可以直接得到改變後的高頻項目集。即使重新探勘的支持度低於之前準高頻項目集的支持度，I-BDFS也只要透過KB，針對因為降低支持度增加的高頻項目集進行判斷，而不用重新掃瞄完整資料庫，因此可以達到相當優異的探勘效率。

伍、結論

本論文提出一個漸進式關聯法則探勘演算法I-DBFS，I-DBFS演算法能夠除了能夠有效率地探勘出高頻項目集合，不用像過去演算法必須重複探勘資料庫，才能得到高頻項目集合。而是利用一次掃瞄資料庫，以Item-Based的資料結構建立I-Table，並且運用條件滿足機制，當項目集的子集合都是高頻項目集才需要進行驗證，只需要少量的測試，不像傳統演算法繁複且大量掃瞄資料庫的測試機制，所以可以節省掉大量時間。此外由於簡單地運用項目為主的資料結構除了可以加速項目產生的速度外，還可以省去其他演算法採用樹狀結構演算法，在將資料掃瞄至樹狀結構時，必須不斷透過搬移或其他方法來維持良好樹狀結構，也可以避免樹狀結構在漸進資料探勘的過程中，因為項目集在高頻與非高頻之間跳動，造成額外資料結構維護花費的時間。此外在面對資料量不斷增加，為了維持關聯法則地有效性，所產生的二次探勘工作，本演算法透過有效的資料保存與運用的方法，可以大量省去重複探勘舊有資料的時間。最後透過實驗的證明，本論文提出的演算法I-BDFS與過去非漸進式演算法相比，在面對支持度改變或者是資料量增加的情況，隨著資料量的增加，演算法可以有效縮短數倍的的探勘時間。即使與近來提出之漸進式探勘演算法相比，仍然擁有更好的演算效率以及穩定性。由此可證明本論文所提出之演算法確實能夠達到增進探勘效能，節省資料探勘時間花費的目標。

參考文獻

1. 王慶堯，2000，利用準大項目集之漸進式資料挖掘，義守大學資訊工程所碩士論文。
2. 江俊彥，2001，應用分群法提升關聯法則效率之研究，國立屏東科技大學資訊管理學系碩士畢業論文。
3. 高淑珍，2004，應用資料探勘於顧客回應模式之研究—以國內A壽險公司為例，國立成功大學企業管理學系博士論文。
4. 莊文宗，2005，運用二元樹於漸進式關聯法則探勘之研究，中華大學資訊管理學系碩士畢業論文。
5. 游坤明、莊文宗、蕭偉呈，2004，『分群方式技術與資料探勘應用於肝功能檢驗與疾病關係之研究』，資通技術管理與應用會議，42頁。
6. 游坤明、盧展皓，2002，『大量資料之關聯式法則的快速發掘與應用—以醫院門診資料為例』，第八屆海峽兩岸資訊管理研討會，294～298頁。
7. 游坤明、盧展皓、張煥禎、林年茂、謝泉發，2002，『大量資料之關聯法則的快速發掘與應用—以醫院門診為例』，第八屆海峽兩岸資訊管理策略發展會議，297～298頁。
8. Agrawal, R. and Srikant, R. "Fast algorithms for mining association rules in large database," *Proceedings of 20th International conference on Very Large Database*, 1994,

pp. 487-499.

9. Agrawal, R. and Srikant, R. "Mining Sequential Patterns," *Proceedings of 11th International conference on Data Engineering*, 1995, pp. 3-14.
10. Chen, Ming-Syan, Han, Jiawei, and Yu, P. S. "Data mining: an overview from a database perspective," *IEEE Transactions on Knowledge and Data Engineering* (8:6), December 1996, pp. 866-883.
11. Cheung, D. W., Han, Jiawei, Ng, V. T. and Wong, C. Y. "Maintenance of discovered association rules in large databases: an incremental updating technique," *Proceedings of the Twelfth International Conference on Data Engineering*, 1996, pp. 106-114.
12. Cheung, D. W., Lee, S. D., Kao, B. and Wong, C. Y. "A general Incremental Technique for Maintaining Discovered Association Rules," *Proceedings of the fifth International Conference on Data Engineering*, 1997, pp. 185-194.
13. Dass, R. and Mahanti, A. "An Efficient Technique for Frequent Pattern Mining in Real-Time Business Applications," *Proceedings of the 38th Hawaii International Conference on System Sciences*, 2005, pp. 76a-76a.
14. Gorodetsk, Vladimir, Karasaeyv, Oleg and Samoilov, Vladimir "Multi-agent Technology for Distributed Data Mining and Classification," *Proceedings of the IEEE/WIC International Conference on Intelligent Agent Technology*, 2003, pp. 438-441.
15. Han, Nam "Data Analysis and Mining in the Life Sciences," *SIGMOD Record* (30:3), 2001, pp. 76-85.
16. Han, Jiawei, Pei, Jian and Yin, Yiwen "Mining frequent patterns without candidate generation," *Proceedings of the 2000 ACM SIGMOD International Conference* on Management of Data, 2000, pp. 1-12.
17. Han, Jiawei, Pei, Jian and Yin, Yiwen "Mining Frequent Patterns without Candidate Generation: A Frequent-Pattern Tree Approach," *Data Mining and Knowledge Discovery* (8:1), 2004, pp. 53-87.
18. Hipp, J., Guntzer, U. and Nakhaeizadeh, G. "Algorithms for Association Rule Mining - A general Survey and Comparison," *Proceedings of ACM SIGKDD Explorations Newsletter*, vol. 2, Issue 1, 2000, pp. 58-64.
19. Mahanti, A., Ghosh, S. and Pal, A. K. "A High Performance Limited-Memory dmissible and Real Time Search Algorithm for Networks," University of Maryland at College Park, MD 20742, Maryland, College Park, *Computer Science Technical Report Series* CS-TR-2858 UMIACS-TR-92-34, 1992.
20. Su, J. H. and Lin, W. Y. "CBW: An Efficient Algorithm for Frequent Itmeset Mining," *Proceedings of the 37th Hawaii International Conference on System Science*, 2004, pp. 9.
21. Park, Jong Soo, Chan, Ming-Syan and Yu, Philip S. "An Effective Hash-Based Algorithm for Mining Association Rules," *ACM SIGMOD Record* (24:2), 1995, pp. 175-186.