結合基因演算法與先驗演算法用於規則探勘

詹詠翔
國立高雄應用科技大學
ysjhankuas@gmail.com

王皓廷
國立高雄應用科技大學
hank79124@hotmail.com

謝欽旭
國立高雄應用科技大學
csshieh@kuas.edu.tw

李財福
國立高雄應用科技大學
tflee@kuas.edu.tw

摘要

在資料探勘中，分類規則法為具有指標性的規則表示方法之一。其主要研究在於可靠的資料中找出具有意義的資訊並藉此歸納出規則，推測未來事情發生的可能性。資料探勘領域中，目前已有許多技術用於解決分類規則，其中，基因演算法與先驗演算法為本論文主要參考的兩種方法，但這兩大演算法用於關聯規則探勘尚有待加強之處。故此，在本篇研究中將結合這兩種演算法之特性，以基因演算法為主要架構，再藉由融入先驗演算法提升整體探勘效率。

關鍵字:基因演算法、先驗演算法、資料探勘、規則探勘。

1. 前言

在資料探勘中，分類規則法為具有指標性的規則表示方法之一。其主要研究在於可靠的資料中找出具有意義的資訊並藉此歸納出規則，推測未來事情發生的可能性。資料探勘領域中，目前已有許多技術用於解決分類規則，其中，基因演算法與先驗演算法為本論文主要參考的兩種方法，但這兩大演算法用於關聯規則探勘尚有待加強之處。故此，在本篇研究中將結合這兩種演算法之特性，以基因演算法為主要架構，再藉由融入先驗演算法提升整體探勘效率。

在基因演算法中，染色體進行交配與突變完成演化式計算。其演算法表現在分類規則上先對染色體編碼表示成規則，進而透過演化式計算找到規則解。而傳統先驗演算法，則是對條規則做逐一的計算，能夠在不遺漏任何規則之情況下找出規則解。若比較這兩種演算法，基因演算法能以較高的效率找出規則解，但在規則的找尋上則會遺失掉些許規則；反之，先驗演算法雖無規則的遺漏問題，但所使用的計算量將遠超出基因演算法。因此在本論文中提出結合上述兩種方法提出一個新的架構藉此達到更加快速有效率的探勘方法。

2. 文獻探討

現今的社會上在人們在做任何事情幾乎都伴隨著資料的紀錄。從到餐廳吃飯、在百貨公司購物、租屋或購屋等，資料的紀錄已經隨著我們的食衣住行育樂，對如此龐大的資料如何將背後隱藏的有用資訊從資料中挖掘出，資料探勘因此孕育而生，其中流程以圖 1 作簡略的說明[1]。

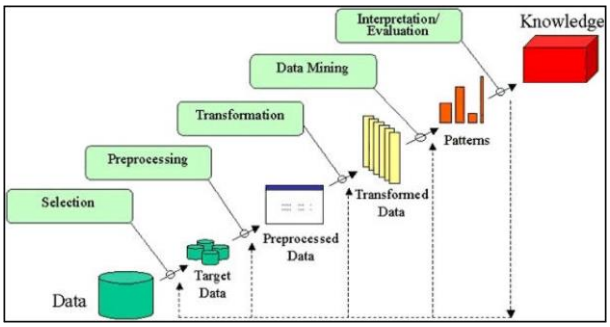

圖 1: 資料庫資料探勘流程[1]。

1. 探勘資料選擇(selection):選擇資料庫中有利於探勘的資料，建立探勘資料集。
2. 資料預先處理(preprocessing):將探勘資料集中的雜訊資料預先處理。
3. 轉換(transformation):將資料做轉換，轉換為利於探勘的資料形式。
4. 資料探勘(data mining):從資料中提攜出有用資訊，常用的方法如:決策樹、關聯規則、回歸分析等。
5. 說明與評估(interpretation/evaluation):經過資料探勘後，評估資訊的可靠度，並以直觀的方式呈現。

在資料探勘中，關聯規則為資料探勘中常用的一種方法，此方法將資料庫中的資料依其之間的關係性建立關聯規則。關聯規則最早由 Agrawal, Imielinski 及 Swami 在 1993 年所提出，從資料中找尋各個項目之間的關係，且須符合兩項門檻定義，定義分別為最小支持度(Minimum Support)與最小信賴度(Minimum Confidence)，關聯規則的表式法則表示成 X→Y，令 I 為所有商品項目集合 I={i1,...,im}，D 為交易資料庫集合

D={ d1,...,dm}，T 為交易資料庫中的項目集合 T={t1,...,tm}，X 與 Y 是所有商品項目集合的商品 X⊂I,Y⊂I 且 X∩Y=∅。

以關聯規則 X→Y 舉例表示支持度與信賴度的公式如下:

$$\text{支持度}(X \cap Y) = \frac{X \cap Y}{D}$$

公式 1:支持度公式

$$\text{信賴度}(X \rightarrow Y) = \frac{|X \cap Y|}{\text{包含 X 的交易紀錄}}$$

公式 2:信賴度公式

其中關聯規則以先驗演算法最具有公信力，是由 Agrawal 等人所提出的一種經典的演算法[2]用於購物籃分析，其演算法透過反覆地對頻繁資料集合進行掃描，進而找出所有頻繁資料集合，通常會訂定最小支持度與最小信賴度篩選出有用的關聯規則如圖 2 所示[3]。演算法精神主要透過頻繁資料集合內所有的資料進行組合，產生新的頻繁資料集，並且以最小支持度與最小信賴度驗證新的頻繁資料集，直到所有的頻繁資料集都掃描過，但為產生大量的頻繁資料集，必須付出龐大的系統資源。

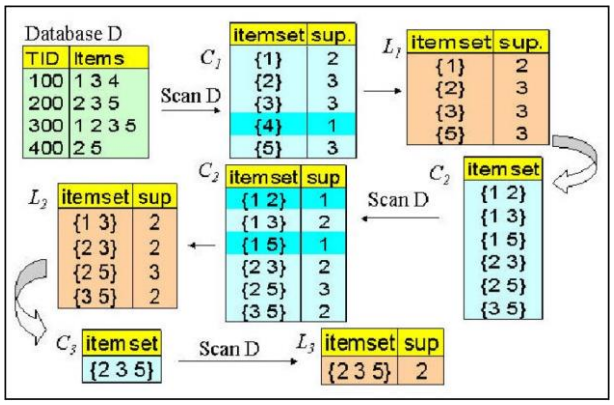

圖 2:先驗演算法挖掘所有的頻繁項目集[3]。

雖然先驗演算法能以無誤差的方式找出所有關聯規則，但消耗系統資源過多，因此 Han 等人提出 FP-growth 演算法[4]克服產生過多頻繁資料集，FP-growth 演算法將各個項目以樹狀結構表示 FP-tree，此方法大量的減少頻繁資料集，且對資料庫只需兩次掃描即可以完成，解決關聯規則資料量過大的問題目前已經有多種解決演算法，例如:蟻群演算法、基因演算法、改良先驗演算法等。本論文是以基因演算法結合先驗演算法兩者特性的混合性演算法，在第三章實驗方法會有詳細的解釋。

基因演算法於 1975 年由 John H. Holland 所提出，主要特色是模仿大自然物競天擇演化的現象，將資料以像基因的方式編碼，編碼過後形成數條染色體，染色體會經由交配、突變、修剪等方式達到繁衍出較優秀的下一代，在資料探勘中，基因演算法也已經被廣泛的應用[5] [6] [7]，提出各種編碼與演化方式。

在一般傳統關聯規則探勘上，所使用的方法必須依賴最小支持度與最小信賴度的全域搜尋，例如 Apriori 演算法，此類方法在規則找尋上非常耗時，為解決此問題 ARMGA(Association Rules Mining with Genetic Algorithm)因此孕育而生，ARMGA 是由 Yan 等學者所提出的方法，ARMGA 目的在於能像一般的遺傳演算法做有效的全域性搜尋，實現在規則探勘上，找出有效的規則。ARMGA 架構可參考下圖 3。

虛擬碼一開始給定一商品項目集 s，藉由 s 集合作為種子初始化染色體，初始化完成後 pop 作為染色體族群。以設定機率 ps 挑選染色體，被挑選上的染色體以 c 作表示回傳 TRUE，沒被挑選上則為 FALSE，接著設定一機率 pc 做為交配率，對被挑選上的染色體進行交配，再來定一突變率 pm 對染色體進行突變，最後回傳染色體族群 pop。

```
Population ARMGA(s,ps,pc,pm)
begin
    i ← 0;
    pop[i] ← initialize(s);
    while not terminate(pop[i])do
    begin
        pop[i+1] ← ∅;
        pop_temp ← ∅;
        FOR ∀c ∈ pop[i] do
            If select(c,ps) then
                pop[i+1] ← pop[i+1] ∪ c;
        pop_temp ← crossover(pop[i+1],pc);
            for ∀c ∈ pop_temp do
                pop[i+1] ← (pop[i+1] - c) ∪ mutate(c,pm);
            i ← i+1;
        end
        return pop[i];
end
```

圖 3:先驗演算法挖掘所有的頻繁資料集[8]。

在 ARMGA 演算法中使用"Mushroom"資料庫做為實驗用資料，並且與 Apriori 演算法比較結果。Apriori 演算法的規則長度固定為 3，支持度為 0%，信賴度為 90%，總共得到 445978 條規則。比較 ARMGA 演算法所找到規則有等於或以上 90%的規則與 Apriori 演算法規則相似，若是把 Apriori 演算法支持度提高到 60%則只找到 8 條規則。從結果上來說 ARMGA 演算法可以是一種有效的全域性搜尋演算法。

3. 實驗架構與方法

本論文分別用以三種演算法實現規則探勘進行分析與比較，Apriori 演算法、GA 演算法與本論文提出的混合式演算法。Apriori 演算法已於第二章做架構與流程說明，並且在第四章將結果做比較，因此本章將節著重於基因演算法與混合式演算法的詳細說明。

3.1 實驗架構

探討文獻發現 GA 演算法與 Apriori 演算法結合，往往因為忌於 Apriori 演算法過多的程式計算，因此以往結合方法只利用了 Apriori 演算法一開始的步驟，產生頻繁資料集做為染色體編碼的項目依據。

在本篇論文中有別於過往的做法在於，在 GA 演算法執行的每個步驟加上了一個短程的信賴度計算，計算不同的項目順序組合是否能夠找出更優秀的染色體，在此稱作短程的 Apriori。演算法結合流程如圖 4。在結合演算法流程中，GA 演算法部分主要參考由 Yan 等學者提出的 ARMGA 演算法，主要差別在於產生頻繁資料集與短程 Apriori，在細節部分每個步驟同樣有些許不同，例如適應函數的訂定、編碼的方式、交配率與突變率的訂定等。

3.2 研究方法

本篇論文中，染色體編碼方式採二進制編碼，每條染色體皆代表一條規則，每個基因代表商品項目，根據商品項目集隨機不重複挑選基因，在此混合演算法中則利用頻繁資料集挑選基因，可視染色體內項目皆為 1，沒被隨機挑選的商品項目為 0，沒被挑選上的商品項目也無法編入染色體。染色體長度根據頻繁資料集隨機決定，信賴度上必須要至少兩項商品項目才可計算，因此最短規則為 2 個商品項目組成，規則最長則是依據頻繁項目資料集總數，假設頻繁資料集和有 10 項商品項目，規則最長可能為 10，此外規則的表示法具有因果關係，在此則需要一個變數決定規則的先行條件，此變數是根據染色體長度亂數決定變數大小。

適應函數在此是識別染色體好壞的評估值，以往規則的強度是以支持度與信賴度兩值做為評價，在此染色體的適應函數主要也是以支持度與信賴度所組成。假設有一染色體 X，在此先預設定兩值 MinSUP 與 MinCONF 做為分母，主要目的在於避免規則找尋時，發生支持度與信賴度其中一值過低，導致搜尋無法全域性搜尋，因此透過 MinSUP 與 MinCONF 設定會影響得到規則結果。

$$fitness(X) = \frac{SUP(X)}{MinSUP} + \frac{CONF(X)}{MinCONF}$$

公式 3:結合式演算法適應函數

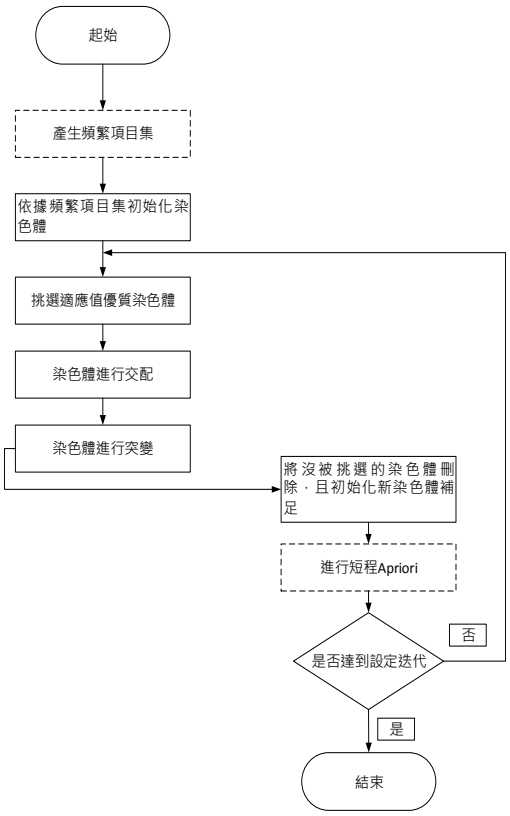

圖 4: 結合式演算法流程圖

本篇論文挑選機制有別於過往的機率式隨機挑選，在此編碼結束後並依照適應函數大小由大到小做排序，且設定一區間 s 挑選適應函數前幾名不做重新編碼，排序完成後將挑選前面數個染色體兩兩做交配，接著將剩下染色體挑選數個做突變，其餘則重新編碼。

交配機制在 ARMGA 演算法中，必須透過一個交配率變數對選擇染色體有一定機率的進行交配機制，在本論文方法中，交配機制首先必須訂定一區間 c，c 值將決定排名前幾名染色體進行交配機制，因以染色體兩兩進行交配，所以 c 值的訂定必須為雙數。在本論文採雙點交配機制，首先判斷交配的兩個染色體長度，根據較短的染色體訂定兩變數cp_1與cp_2，此雙點以不重複亂數決定交配區間，以下圖 5 為說明。

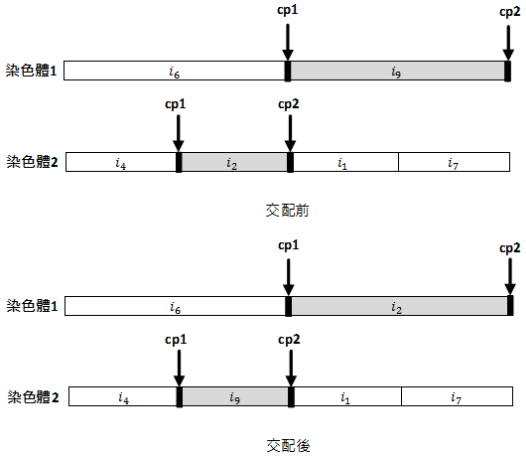

圖 4: 結合式演算法交配狀況 1

若交配完後染色體具有重複基因，則刪除後面重複基因。如圖 6。若發現交配後染色體長度為 2，且剩下兩項相同基因，則還原成原來交配前基因如圖 7。

在本論文突變機制中，在交配後所進行的機制，且同一迭代上突變機制與交配機制並不會發生在同一條染色體，突變機制是利用交配後所剩餘的染色體利用一突變機制區間 m 決定對前幾條染色體進行突變。在此突變機制是採用兩點變數mp_1與mp_2決定突變區間，此區間最大不超過染色體長度，區間最小長度為一個基因，接著以亂數且不重複原本基因挑選新的基因。在此機制中若發生染色體基因為所有商品項目所組成基因則不進行突變機制，因為無論如何挑選基因皆會重複，以圖 7 所示。

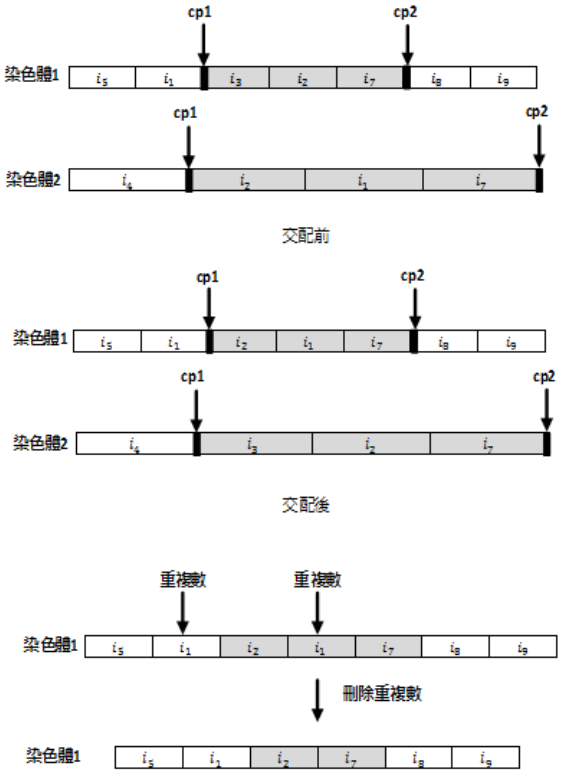

圖 5: 結合式演算法交配狀況 2

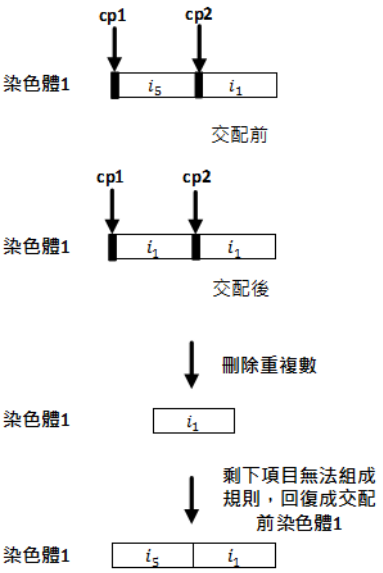

圖 6: 結合式演算法交配狀況 3

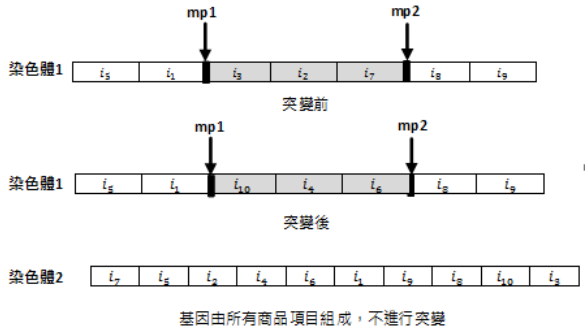

圖 7: 結合式演算法突變機制

本篇論文中加入了兩個方法，第一個是利用商品項目集合所產生的頻繁資料集，主要目的在替除掉支持度低的商品項目，此方法必須先訂定一個最小支持度，在此以 mc 為最小支持度變數，通過最小支持度之項目組成頻繁資料集。

第二個方法，短程的 Apriori 方法屬於本論文所加入新方法，此方法透過變化染色體中決定先行條件變數 k，找尋適應函數較佳染色體。在傳統的 Apriori 演算法中，每一次的掃描需對高頻項目集中，所有商品項目做所有不同的組合找尋規則，在此不會對基因重新排列，且重新排列計算適應函數將耗費相當大的計算成本，因此短程的 Apriori 方法透過變化先行條件變數 k，期望能夠從不變化基因的方式找尋更好的規則。首先須訂定一區間 a 決定染色體排名前幾名進行短程的 Apriori 方法，在此對染色體中所有變數 k 可能發生的位置做適應函數的計算，如果得到更高的適應函數則做 k 值更新。

假設有一染色體 C 由商品項目i_1、i_2、i_3、i_4所組成，變數 k 可能發生值有三個，短程的 Apriori 方法則會對每個 k 值狀況計算適應函數，更新最高適應函數與相對應 k 值，如下圖 8，透過此方法找尋適應函數最高 C3。

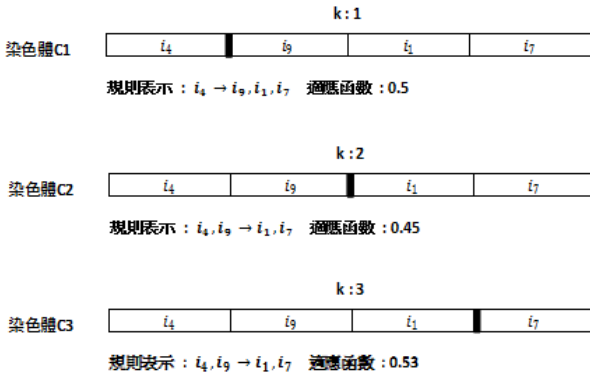

圖 8: 短程的 Apriori 方法

4.實驗結果

在基因演算法中能夠以較少的耗時找到規則，但在規則的品質上卻不如 Apriori 演算法來的可靠，因此本論文目的在於能夠兩利相權取其重，既能避免如 Apriori 演算法耗時，找到規則也能比基因演算法結果可靠，提供一個適合規則探勘的好方法。本章節將探討五個方法在同一資料庫進行規則探勘的結果，五個方法分別為 Apriori 演算法(AP)，基因演算法(GA1)，基因演算法加入頻繁資料集方法 (GA2)，基因演算法加入短程的 Apriori 方法(GA3)，結合短程的 Apriori 方法、頻繁資料集方法與基因演算法(GA4)。

本實驗所採用的資料庫是 ExtendedBakery 所提供的麵包店 5000 筆交易資料集[9]，觀察各個演算法在每一迭代中消耗時間， GA 演算法配合不同機制在適應函數上的表現，並且觀察調整各個區間變數所造成的影響結果為何，實驗環境如下表 1，實驗參數設定如下表 2

表 1: 實驗環境

實驗環境	
作業系統	Windows7(32bit)
CPU	Pentium(R)Dual-Core
記憶體	4GB DDR2
實驗程式開發環境	Devcpp 5.8.2.0

表 2: 實驗參數設定

演算法參數設定	
Apriori 最小支持度	0.03
Apriori 最小信賴度	0.7
GA 迭代次數	1000
染色體數	50
Minsup	0.03
Minconf	0.7
選擇區間(s)	1~25
交配區間(c)	1~20
突變區間(m)	21~25
短程的 apriori 區間(a)	1~5

Apriori 演算法在此資料庫的時間消耗為 60.93 秒(sec)如圖 9，找尋到規則有 32 條。使用同樣資料庫在 GA 的四種不同方法，執行 20 次平均結果如下圖 10，從圖上可以發現若加入頻繁資料集方法，適應函數 GA4 與 GA2 上據有明

顯較快速的爬升，若只加入短程的 Apriori(GA3)，GA3 與 GA1 在適應函數表現上，則是從迭代 63 開始 GA3 一直優於 GA1，若同時加入頻繁資料集方法與短程的 Apriori 在 GA4 中，則是在所有方法中適應函數表現最好，如圖 11，整合方法 GA4 比基因驗算法 GA1 最大差距有 43.6%的適應函數。

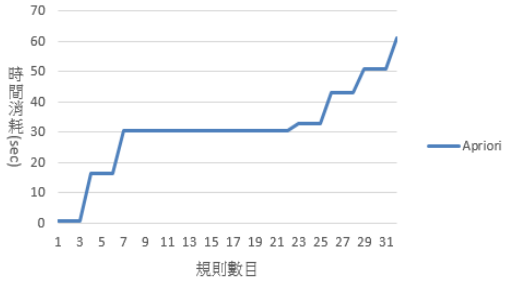

圖 9: Apriori 演算法

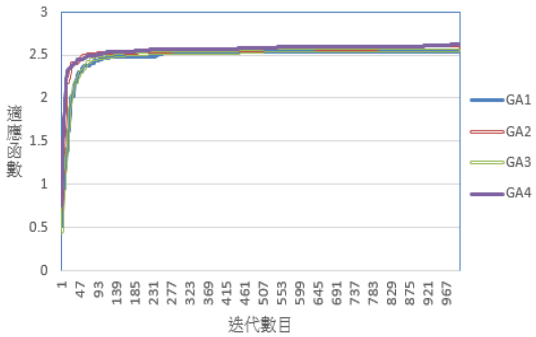

圖 10: GA 四種方法(a)

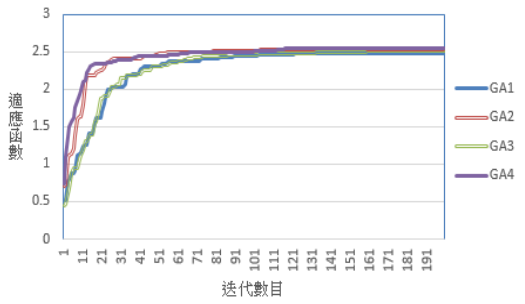

圖 11: GA 四種方法(b)

在第 500 迭代與第 1000 迭代的平均結果如下表 3，GA1 在所有適應函數表現最差，GA4 在所有適應函數表現最佳，且在 GA4 的適應函數比 GA1 的適應函數多了 0.083 適應函數值，如圖 12。且 GA4 找到具有 Apriori 演算法平均找到規則 29.3 條，透過此適應函數的設計加上整合式的方法，可找到通過 Apriori 演算法最小支持度與最小信賴度的規則有 91.562%。

表 3: 實驗參數設定

第 500 迭代適應函數	
基因演算法(GA1)	2.535244
基因演算法加入頻繁資料集方法(GA2)	2.552355
基因演算法加入短程的 Apriori 方法(GA3)	2.538129
結合短程的 Apriori 方法、頻繁資料集方法與基因演算法(GA4)	2.584145
第 1000 迭代適應函數	
基因演算法(GA1)	2.538809
基因演算法加入頻繁資料集方法(GA2)	2.583181
基因演算法加入短程的 Apriori 方法(GA3)	2.557958
結合短程的 Apriori 方法、頻繁資料集方法與基因演算法(GA4)	2.621825

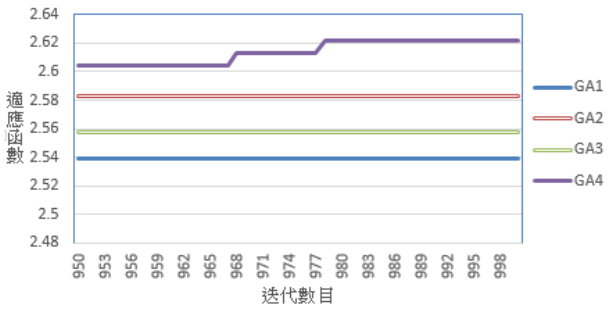

圖 12: GA 四種方法(c)

從適應函數的結果可以發現，加入短程的 Apriori 在規則搜尋上的確有助於找尋更好的規則，但在時間上的消耗勢必需要有所犧牲，如下表 4 比較可以發現，最耗時的 Apriori 演算法(AP)總共耗時 55.429(sec)，基因演算法(GA1)耗時 42.325(sec)，基因演算法加入頻繁資料集方法(GA2)耗時 28.883(sec)，基因演算法加入短程的 Apriori 方法(GA3)耗時 43.941(sec)，結合短程的 Apriori 方法、頻繁資料集方法與基因演算法(GA4)耗時 30.341(sec)。

可以發現 GA1 與 GA3 耗時比較上，GA3 只比 GA1 多了 0.036%的耗時，若比較 GA2 與 GA4，GA4 則多了 0.048%的耗時，在結果上發現，加入短程的 Apriori 在耗時上的犧牲在接受範圍內，若 GA3 與 AP 耗時比較上，GA3 省去 20%的時

間，若再加上頻繁資料集方法(GA4)與 AP 耗時比較，GA4 則可以省去 45%的時間。

表 4: 實驗參數設定

演算法程式執行時間	
Apriori 演算法(AP)	55.429(sec)
基因演算法(GA1)	42.325(sec)
基因演算法加入頻繁資料集方法 (GA2)	28.883(sec)
基因演算法加入短程的 Apriori 方法(GA3)	43.941(sec)
結合短程的 Apriori 方法、頻繁資料集方法與基因演算法(GA4)	30.341(sec)

結論

本論文提出一個方法可從基因演算法的演化過程中，做具有短程的區域搜尋，稱作短程 Apriori 方法，希望藉此可以增加基因演算法在規則探勘更加有效率，且在計算成本上是可以被接受的，在此進行了各種方法上的比較可以發現，加入短程 Apriori 方法在 5000 筆的交易資料集當中，只比基因演算法多了 0.036%的耗時，且在適應函數的表現上的確優於基因演算法。本論文將方法結合後，在實驗結果可以發現結合式演算法相較於傳統 Apriori 演算法省去了 45%的耗時，並且在規則找尋上 Apriori 演算法平均有 91.5%規則相似結合式演算法，在串流式的資料當中，本論文提出的結合方法與基因演算法比較，最大適應函數有增加了 43.6%適應函數，達到比基因演算法更有效率且比 Apriori 演算法更省計算成本。

致謝

感謝科技部 MOST 104-2221-E-151-068-及 MOST 103-2622-E-151-017-CC3 對本研究的支持與補助。

參考文獻

[1] U. Fayyad, G. Piatetsky-Shapiro and P. Symth , “Overview of Data Mining and Knowledge Discovery”, Knowledge Discovery and Data Mining, AAAI press,1996, pp. 1-36.

[2] R. Agrawal and R. Srikant , “Fast algorithm for mining association rules”, in Proc. 1994 Int. Conf. Very Large Data Bases (VLDB’94), 1994, pp. 487-499.

[3] 廖原豐，“因果關聯規則挖掘”，國立中央大學資訊管理研究所碩士論文，2006。

[4] J. Han,J.Pei and Y.Yin , “Mining Frequent Patterns without Candidate Generation”, in Proc. 2000 ACM-SIGMOD Int. Conf. Management of Data(SIGMOD’00), 2000, pp. 1-12.

[5] A. A. Freitas, “A Survey of Evolutionary Algorithm s for DataMining and Knowledge Discovery”,Advances in EvolutionaryComputing Natural Computing Series, 2003.

[6] M. R. Kumar and D. K. Iyakutti, “Application of genetic algorithms for the prioritization of association rules” IJCA Special Issue on Artificial Intelligence Techniques-Novel Approaches and Practical Applications, 2011,pp. 1 – 3.

[7] Yang Xu,Mingming Zeng,Quanhui Liu, and Xiaofeng Wang “ A Genetic Algorithm Based Multilevel Association Rules Mining for Big Datasets” Hindawi Publishing Corporation Mathematical Problems in Engineering Volume, Article ID 867149, 9 pages 2014.

[8] Xiaowei Yan, Chengqi Zhang, and Shichao Zhang,"ARMGA:Identifying interesting association rules with genetic algorithms", Expert Systems with Applications 36,2009

[9]ExtendedBakery, https://wiki.csc.calpoly.edu/datasets/wiki/ExtendedBakery