行政院國家科學委員會專題研究計畫　成果報告

演化式演算法應用於資料探勘之研究(2/2)

計畫類別：個別型計畫
計畫編號：NSC93-2213-E-009-028-
執行期間：93 年 08 月 01 日至 94 年 07 月 31 日
執行單位：國立交通大學工業工程與管理學系(所)

計畫主持人：沙永傑

計畫參與人員：劉正祥

報告類型：完整報告
報告附件：出席國際會議研究心得報告及發表論文
處理方式：本計畫可公開查詢

中　華　民　國 94 年 9 月 29 日

目錄

頁次

表目錄

頁次

圖目錄

頁次

中文摘要

在知識發掘與資料探勘的領域中，分類技術被視為一個重要的研究議題。分類的目的在於定義每一個類別的特徵，透過訓練組的資料，建立一個判斷類別歸屬的模型，將未歸類的資料分門別類。

一般來說，資料類型可分成兩大類：數值型、類別型。在真實生活中，數值型資料之存在是相當普遍的，但是大多數的分類預測方法，只能處理類別型資料。針對數值型資料大多數研究會在資料前置處理（preprocessing）過程中將其離散成類別型資料，之後再利用資料探勘工具萃取分類規則。然而任何離散數值型資料的方法，都會造成原先隱藏於資料當中的資訊流失。因此本研究希望能發展一套分類預測方法，能在不離散數值型資料情況下進行分類規則的萃取。

本研究共分成兩個階段，第一階段將以基因演算法（GA）為主要研究工具，希望藉由基因演算法的高效率與彈性，設計出能同時處理數值型與類別型資料的分類預測方法。第二階段為針對螞蟻理論（Ant System）設計出能同時處理數值型與類別型資料的分類預測方法，並比較兩種演化演算法在分類績效上的差異。

關鍵詞：資料探勘、離散化、基因演算法、螞蟻理論

英文摘要

Classification is one of the important issues in knowledge discovery and data mining. The goal of classification is to define the characteristics for each class in order to predict if a previously unknown object either belongs to the class or not.

In general, attribute data type can divide into two groups: numerical and categorical. Numerical attributes are very common in real-world application. There exist a large number of classification algorithms, which handle categorical attributes only. Therefore, the process of the descretization is an essential task for data preprocessing in knowledge discovery in databases (KDD). But during the discretization of numerical attribute, some information hidden in the data set can be lost, we will proposed classification algorithms can extract classified rules with numerical and categorical attributes simultaneously.

The first step of this project is based genetic algorithm to develop more effective classifier (GA-Classifier), which handle numerical and categorical attributes simultaneously than traditional classification algorithm—Decision Tree (C4.5). The second step of this project is based a novel evolutionary algorithm (Ant System, AS) to develop more effective and efficient classifier (Ant-Classifier) than GA-Classifier and Decision Trees.

Keywords: Data mining, Descretization, Genetic algorithm, Ant colony optimization

前言與動機

為因應環境的變遷與商業競爭日益激烈，許多企業在面臨資訊科技發展的潮流下，均希望透過資訊科技的力量為企業帶來更多的競爭優勢。然而當企業引進資訊技術來有效率地收集資料時，卻發現無法有效率地從所收集數量龐大的資料中，發掘出有用的知識與規則。因此企業的焦點逐漸轉變到如何有效的利用所收集到的資料獲取有用的資訊。所以資料探勘的技術就逐漸被學術界與業界所重視。常見的資料探勘的定義有以下數種。

> Cabena(1997)定義資料探勘是將先前不知道，有效的資訊從龐大資料庫中萃取出的過程，並提供給決策者作為決策依據。
>
> Hall(1995)定義資料探勘乃是針對大量的資料，以全自動或半自動的方式進行分析，找出有意義的關係或規則。
>
> Berry(1997)定義資料探勘結合許多不同的技術，如資料視覺化（Data Visualization）、機器學習（Machine Learning）、統計（Statistics）以及資料庫（Databases）以便從龐大資料量中萃取以規則形式或其他模式所表達的知識。

資料探勘可以應用的領域幾乎涵蓋了各行各業，例如：生產製造、財務投資、信用卡交易、服務業、...、等等。基本上，資料探勘是知識發現（Knowledge Discovery in Databases, KDD）當中的一個步驟。知識發現大致可分成五大步驟[Bruha, 2000]：

1. 瞭解資料探勘所要應用的領域以及熟悉相關知識，並選擇所要使用的資料探勘技術。
2. 進行目標資料的收集。
3. 進行資料前置處理。針對目標資料當中不一致或是遺漏值進行必要的處理。由於所收集的目標資料當中，屬性的資料類型可分成兩大類：數值型、類別型。在資料前置處理過程中會將把數值型資料進行離散化，以轉換成類別型資料，以方便資料探勘工具的使用。
4. 將經過資料前置處理後的資料，以資料探勘工具進行知識萃取。
5. 資料後置處理。透過專家來驗證所萃取出的知識，並將有用的知識納入現有的決策系統。

資料探勘的相關作業應用可區分成五種[Collard, 2001]：資料特徵描述（Description）、分類（Classification）、關連分析（Association Discovery）、順序樣式分析（Sequential Pattern Analysis）、迴歸（Regression）。

在本計劃中主要在探討資料探勘中的分類作業。在資料探勘的領域中，分類被視為一個重要的研究議題。分類的目的在於定義每一個類別的特徵，透過訓練組的資料，建立一個判斷類別歸屬的模型，將未歸類的資料分門別類。分類作業所萃取出的知識，其

表達形式通常為 IF-THEN 規則，表達如下所示。

IF <先決條件式> THEN <類別歸屬>

先決條件式由數個條件子(terms)所組成，每一個條件子利用邏輯運算符號(AND)連結。每一個條件子由三項資訊組成：屬性、比較運算符號、屬性值，例如：<性別=男性>。類別歸屬用來預測符合先決條件式的資料其應屬類別值。從使用者的觀點而言，IF-THEN 的知識表達形式比較簡單易懂，較能讓知識使用者所接受。

一般來說，進行分類作業時多數的分類預測方法，針對數值型資料會在資料前置處理（preprocessing）過程中將其離散成類別型資料，之後再利用資料探勘工具萃取分類規則。然而任何離散數值型資料的方法，都會造成原先隱藏於資料當中的資訊流失。因此本研究希望能發展一套分類預測方法，能在減少資訊流失的前提下離散數值型資料，進行分類規則的萃取。

研究目的

本計劃目的在於利用演化式演算法來發展分類預測方法，能同時處理數值型與類別型資料，在減少資訊流失的前提下離散數值型資料，並進行分類規則的萃取。本計劃共分成兩個階段，第一階段將以基因演算法（GA）為主要研究工具，希望藉由基因演算法的高效率與彈性，設計出分類績效較傳統分類工具—決策樹(Decision Tree, DT)好的分類器。第二階段將根據相同概念以螞蟻理論（ACO）設計出能同時處理數值型與類別型資料的分類器，並比較兩種演化演算法在分類績效上的差異。

文獻探討

目前已有許多機器學習(Machine Learning, ML)工具被應用於分類技術，如：決策樹、類神經網路、基因演算法。決策樹是目前最常被使用於分類作業的工具，其優點在於所產生的規則容易被人們所接受與解釋，缺點在於無法偵測與利用有交互作用的屬性[Clare, 2000]。同時決策樹針對數值型資料必須在知識發現（Knowledge Discovery in Databases, KDD）的資料前置處理（preprocessing）過程中使用離散工具(C4.5 Discretization)將其離散成類別型資料，之後再利用演算法(C4.5)進行規則萃取。然而任何離散數值型資料的方法，都會造成原先隱藏於資料當中的資訊流失[Bruha, 2000]，所以應在不離散數值型資料情況下進行規則萃取。另外，類神經網路雖能同時處理數值型資料與類別型資料，但是其計算過程被視為黑箱(black box)且輸出的結果難以被人所解釋。至於基因演算法具有輸出結果容易被解釋以及有全域搜尋最佳解能力的優點，但缺點是計算時間較長。因此本計劃將基於演化演算法發展有效率的分類預測方法，能在不離散數值型資料情況下進行分類規則的萃取，具同時處理數值型與類別型資料的能力。

在過去有許多研究文獻利用基因演算法發展分類器[Congdon, 2000][Bandar, 1999][Lopes and Pozo, 2001][Fu and Mae, 2001][Pozo and Hasse, 2000][Shin and Lee, 2002][Bruha, 2000][Noda et al., 1999][Fidelis et al., 2000]，相關文獻證明利用基因演算法進行分類其分類正確率優於決策樹。表一彙整近年來使用基因演算法求解分類問題之研究，並說明各研究中針對數值型資料處理方式以及允許建構於規則當中的比較運算符號與邏輯運算符號。從表一可以了解利用基因演算法應用於分類問題時，早期對數值型資料均是加以離散化，同時允許建構於規則內的比較運算符號只限等於（＝），邏輯運算符號亦只限交集（AND）。近年來逐漸有學者發表可以同時處理數值型資料與類別型資料的基因演算法，同時所能建構於規則當中的比較運算符號也較以往來的多。至於邏輯運算符號還是只允許 AND 符號。

表一 基因演算法應用於分類技術研究之彙整

作者	年代	數值型資料處理方式	比較運算符號		邏輯運算符號	備註
Noda et al.	1999	無數值型資料	=		AND	--
Bruha et al.	2000	離散化	=		AND	--
Congdon	2000	離散化	＝		AND	--
Fidelis et al.	2000	不離散化	數值型資料	類別型資料	AND	--
			≧、＜	＝、≠		
Pozo and Hasse	2000	不離散化	數值型資料	類別型資料	AND	--
			≦、≧	＝		
Shin and Lee	2002	不離散化	數值型資料	類別型資料	AND	限制條件子數目為 5 個
			＜、≧	＜、≧		

經由表一可以了解使用基因演算法所能產生的分類知識表達形式(IF-THEN)其變化性較少，因此第一年計劃將針對基因演算法發展一套有效率、符號變化性較多的分類器，並能同時處理數值型與類別型資料。

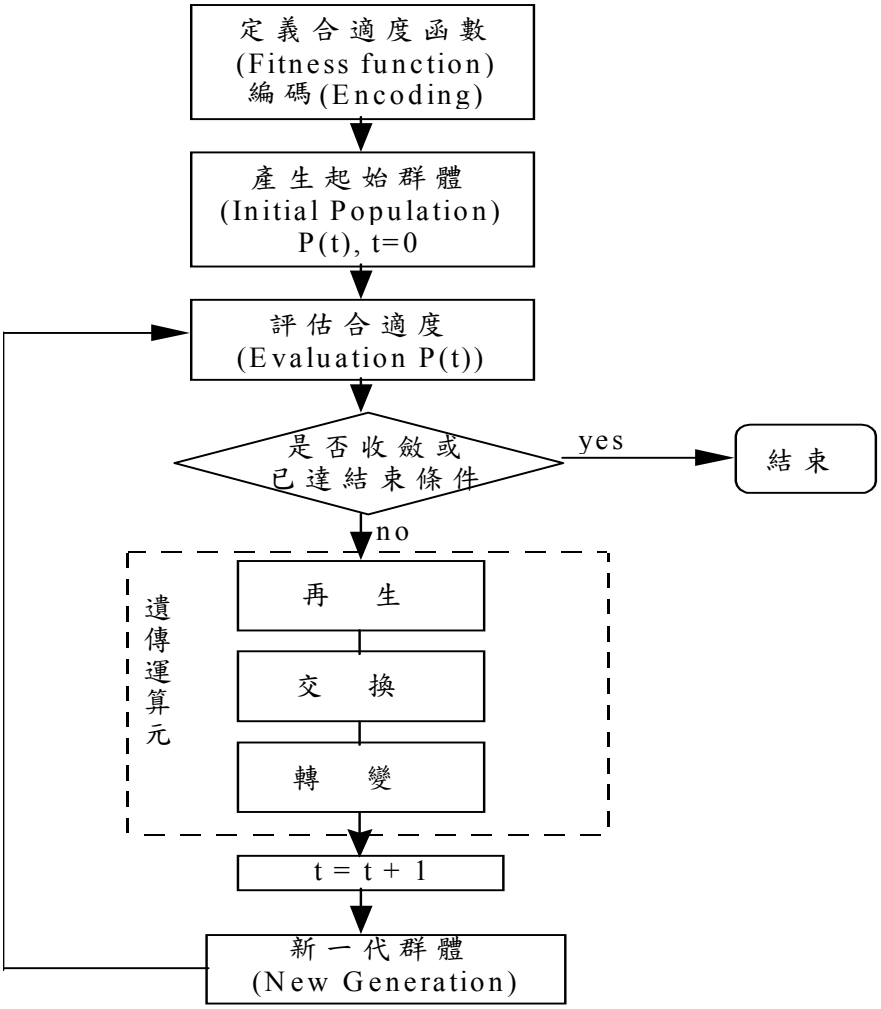
基因演算法（genetic algorithm, GA）最早由 John Holland 等人在 1975 年首度發表，但直到 1980 年代後才逐漸有較多理論與應用之發展。GA 係將所有搜尋的參數轉換成另一種有限長度的表示式子（字串），再利用遺傳運算元產生新的下一代，配合評估的標準使子代具有比母代更好的表現。應用 GA 求解最佳化問題，須將問題目標轉化成對應的函數，稱為合適度函數（fitness function）；合適度函數代表系統對環境的適應能力，相當於系統的性能指標。GA 會將每一代中合適度高的解依據機率複製到下一代，再經

過演化(evolution)、突變(mutation)與交配(crossover)等等運算去產生合適度更高的下一代，持續此反覆過程便可以逐步找到近似最佳解。這種演算的方式正與大自然的生物生存特性相似，產生更具適應能力的下一代以求在外部環境下能延續族群。

演算法中一開始會產生一個起始集合，稱為母體（Population），集合內包含有 N 個染色體（Chromosome），每一個染色體由許多的基因（Gene）所組成，每一個基因就代表一個自變數。因此，每一個染色體就代表一組解。GA 會使用適應函數（Fitness function）來評估每一個解的品質。在每一個世代中，GA 會根據解的品質選取適應函數較佳的染色體放入交配池進行複製，再經由交配及突變的過程以產生下一世代的染色體。在運算的過程中，有些參數必須是先設定，如：族群大小（Population Size）、交配機率（Crossover Rate）、突變機率（Mutation Rate）、適應函數之設計、終止條件（Stopping Conditions）。其中，族群大小的設定會影響演化的結果，族群太小則收斂較早，較難達到預期成果；族群太大則會消耗較長的計算時間。交配機率太高則會需要較長的計算時間；交配機率太低，則演化收斂較快。突變機率太高則會演化過程與隨機搜尋一樣；交配機率太低，則演化過程中將少有新物種進入族群。適應函數的設計必須根據求解的題型來設計，它必須具有反應出不同染色體間品質的差異，也必須能將表現不佳的染色體淘汰地能力。

圖一為 GA 應用在搜尋最佳解問題的執行流程[王培珍, 1996]，以下針對其流程做說明：

(1) 定義合適度函數／編碼。
(2) 產生起始解。
(3) 評估目標值。
(4) 如果已達結束條件則停止，反之則執行步驟(5)。
(5) 遺傳運算元之運算。
(6) 產生新群體後，執行步驟(3)。

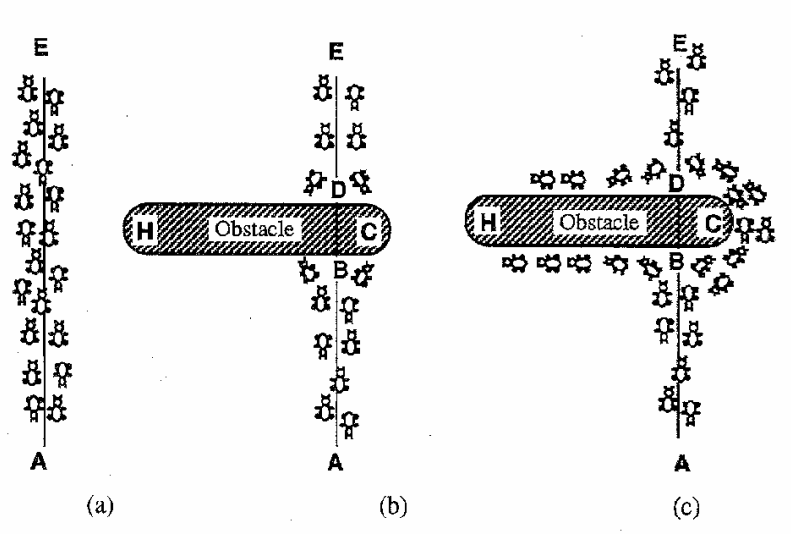


圖一 基因演算法流程圖（王培珍, 1996）

螞蟻理論是由 M. Dorigo 在 1996 首次發表應用求解旅行者推銷問題（Traveling Salesman Problem, TSP），此理論目前已應用於其他 NP-Hard 問題，如：非對稱式旅行者推銷問題（Asymmetric Traveling Salesman Problem, ATSP）、二次規劃問題（Quadratic Assignment Problem, QAP）、零工式工廠排程問題（Job Shop Scheduling Problem, JSP）以及車輛途程問題（Vehicle Routing Problem, VRP）、資料探勘(Data Mining, DM)等等。

螞蟻理論主要是根據大自然界中螞蟻尋找食物方法而演化而成。自然界的螞蟻是接近全盲的一種生物，在尋找食物的過程中螞蟻之間是利用費洛蒙進行溝通。螞蟻在行進中會遺留費洛蒙給其他螞蟻作為路徑選擇的依據，隨著越來越多螞蟻走過相同路徑時，此一路徑的費洛蒙量也隨之增加，相對地其他路徑上的費洛蒙物質量就會逐漸被蒸發，費洛蒙量越多則螞蟻選擇此一路徑機會就越高，相對地其他路徑被選擇機會就會減少，到最後多數螞蟻搜尋食物的路徑就會收斂至單一的搜尋路徑。此一路徑即為螞蟻巢穴與食物來源之間最短路徑。以圖二為例，E 為螞蟻巢穴所在地，A 為食物所在的地點。螞蟻不停地在 AE 路徑上搬運食物 (圖二 (a))。如果在 AE 路徑上放置一個障礙物，則螞蟻在搬運的過程必須從障礙物的兩側選擇單一側通過，以到達其目的地 (食物的地點 A

或巢穴 E) 。在障礙物放置初期，兩側的路徑上並沒有任何費洛蒙的殘留物，所以兩側路徑中任一側被螞蟻選擇通過的機率是相同的 (圖二(b))。由於路徑 ACE 的距離較短，所需的時間亦較少，因此路徑上所留下的費洛蒙被蒸發的量較少。當後面有螞蟻要再通過時，路徑 ACE 上殘留較多的費洛蒙，所以螞蟻選擇路徑 ACE 的機率較高。久而久之大部份的螞蟻都會選擇此一較短的路徑通過 (圖二 (c))。



圖二 在現實世界裡螞蟻的行為模式 (Dorigo, 1996)

目前為止，已有許多學者提出數種不同的螞蟻演算法，除了所要求解的問題不同外，主要還是在討論如何設計不同的產生問題解以及費洛蒙更新的方法。在這個章節將以 TSP 問題為例，簡略介紹本計畫內所使用的螞蟻系統 (Ant System, AS)。在做進一步介紹之前，先詳列各數學符號的定義：

1. $\tau_{ij}(t)$：在第 t 次循環中，存在於節點 i 及節點 j 之間費洛蒙量。

2. η_{ij}：除了費洛蒙之外，提供給螞蟻選擇行走路徑之其它資訊。在 TSP 問題中可以將它定義為 $\frac{1}{d_{ij}}$，其中 d_{ij} 表城市 i 和城市 j 之間的距離。

3. ρ：每個循環間費洛蒙蒸發的比率。

4. $p_{ij}^{k}(t)$：在第 t 次循環內，螞蟻 k 行進至節點 i 時，選擇前往節點 j 的機率。

5. N_i^k：當螞蟻 k 行進到節點 i 時，可選擇作為下一步目的地的節點集合。

6. t：第幾次循環。

7. m_{ant}：每個循環所使用的螞蟻數量。
8. α：螞蟻隨機選擇路徑時賦予費洛蒙的權重。
9. β：螞蟻隨機選擇路徑時賦予其他資訊的權重。

螞蟻系統（Ant System, AS）是最早被提出的蟻群演算法之一，系統內前進某一節點的機率如公式所示。當螞蟻於節點 i 時，選擇往節點 j 前進的機率和路徑 ij 上的費洛蒙量及其他($[\tau_{ij}(t)]^{\alpha}[\eta_{ij}]^{\beta}$)有關。當$[\tau_{ij}(t)]^{\alpha}[\eta_{ij}]^{\beta}$愈大，則往節點 j 前進的機率就愈大。其中α、β分別為對τ_{ij}及對η_{ij}的權重，當α愈大，表示比較以τ_{ij}的大小來選擇路徑；當β愈大，表示比較以η_{ij}大小選擇路徑。

$$p_{ij}^{k}(t)=\frac{[\tau_{ij}(t)]^{\alpha}[\eta_{ij}]^{\beta}}{\sum_{l\in N_i^k}[\tau_{il}(t)]^{\alpha}[\eta_{il}]^{\beta}} \quad , \quad j\in N_i^k$$

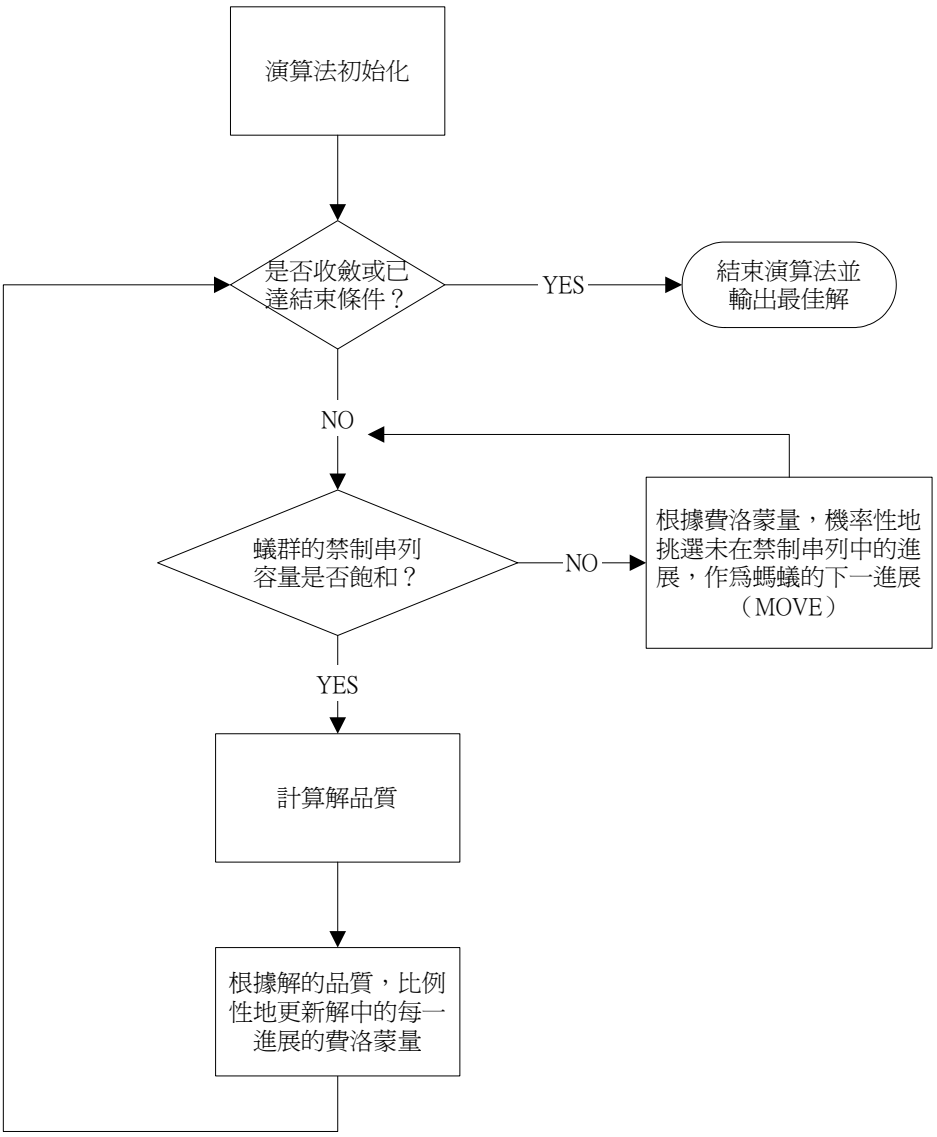
每經過一個循環，螞蟻系統會更新所有的路徑上所遺留下的費洛蒙。此次循環內，螞蟻所走的路徑愈短，增加的費洛蒙就愈多；越長，則增加的費洛蒙越少。更新的方式如公式所示。

$$\tau_{ij}(t+1)=(1-\rho)\cdot\tau_{ij}(t)+\sum_{k=1}^{m_{ant}}\Delta\tau_{ij}^{k}(t)$$

$$\Delta\tau_{ij}^{k}(t)=\begin{cases}1/f^{k}(t) & if\ arc\ (i,j)\ is\ used\ by\ ant\ k\\ 0 & otherwise\end{cases}$$

圖三為 Ant System 應用在搜尋最佳解問題的執行流程，以下針對其流程做說明：

(1) 演算法初始化。
(2) 根據費洛蒙量機率性挑選未在禁制串列中的進展。
(3) 如果蟻群禁制串列以飽和則進行步驟(4)，反之則執行步驟(2)。
(4) 計算解品質。
(5) 根據解品質，比例性更新費洛蒙。
(6) 如果已達結束條件則停止，反之則執行步驟(2)。



圖三 Ant System 應用在搜尋最佳解問題之執行流程

其中首次應用螞蟻理論於資料分類的是 Parpinelli 等人，不過 Parpinelli 等人針對數值型資料乃是利用離散工具(C4.5 Discretisation)將其離散成類別型資料，再進行規則萃取，允許使用的運算符號只限等於(＝)以及交集(AND)。因此在本計畫第二年度將根據之前相同的概念，針對螞蟻發展一套有效率、符號變化性較多的分類器，並能同時處理數值型與類別型資料。

研究方法

第一年計劃將以基因演算法作為主要研究工具，發展以基因演算法為基礎之分類器（GA-Classifier）。此一分類器針對不同類型的資料具有不同的處理方式。針對數值型資

料，GA-Classifier 將不執行離散化步驟，希望能保有原始資料內所隱藏的資訊；針對數值型資料將可擁有小於等於（≦）以及大於等於（≧）的比較運算符號，而類別型資料將可擁有小於等於（≦）、大於等於（≧）、等於（＝）以及不等於（≠）的比較運算符號；至於邏輯運算符號部分將可擁有交集（AND）與聯集（OR）符號的選擇。希望透過較多的比較運算符號與邏輯運算符號的選擇，讓多個規則能結合成單一規則，減少產生規則的數目，以方便決策者使用。表二內描述 GA-Classifier 所具有的功能。

表二 GA-Classifier 功能說明

演算法	數值型資料處理方式	比較運算符號		邏輯運算符號
GA-Classifier	不離散化	數值型資料	類別型資料	AND、OR
		≦、≧	≦、≧ ＝、≠	

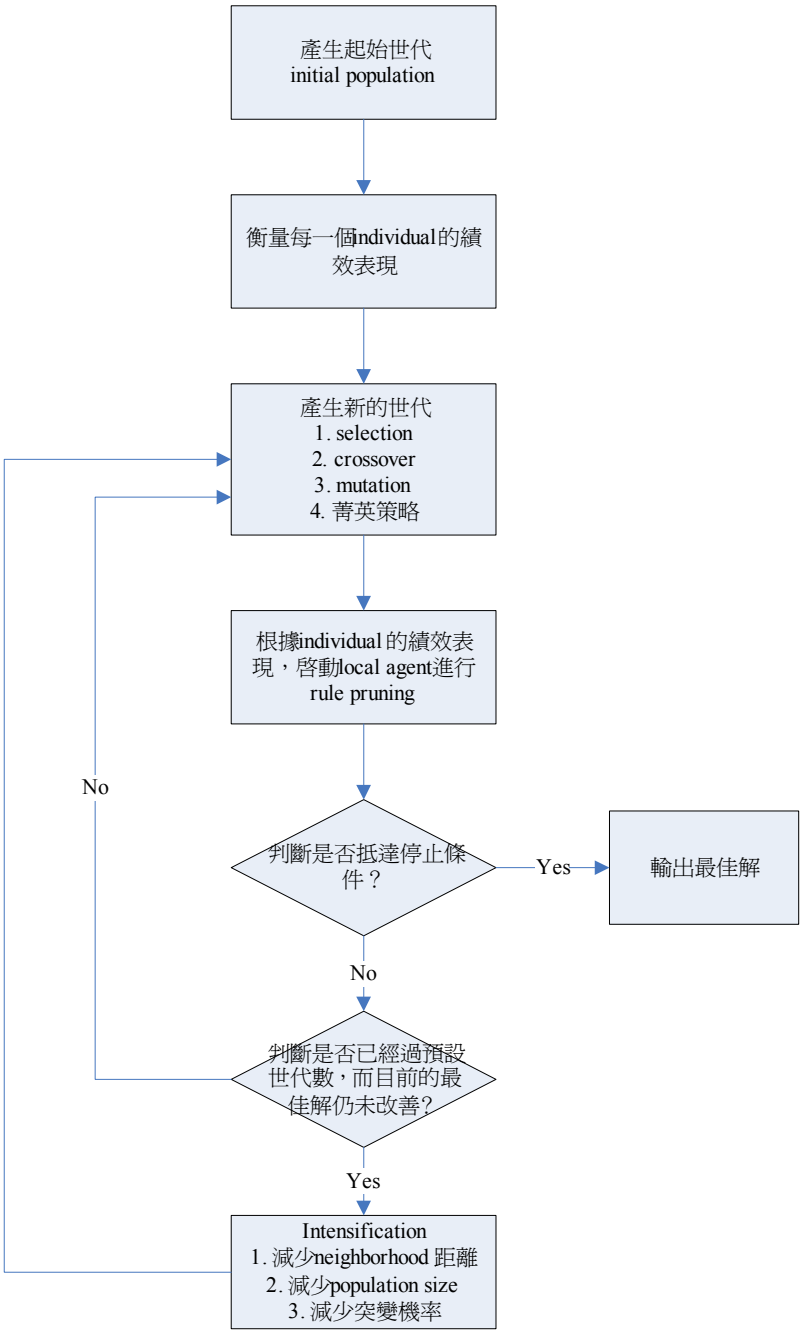
在 GA-Classifier 內針對連續型屬性產生規則子的方式，先隨機產生一個基點（base point）以及運算符號，並在 GA-Classifier 設定之搜尋範圍（search domain）隨機產生一個區間值（interval），將基點加上區間值即為該規則子之界限值（bound value），例：variable(x) ≦ base point + interval。

圖四表示我們所發展的 GA-Classifier 的執行流程。GA-Classifier 的第一個步驟將隨機產生一個起始世代，起始世代內包含 *IPopSize* 個個體，每一個個體代表一條 IF-THEN 規則。針對這 30 個個體衡量其績效表現，根據其績效表現產生下一個新的世代。接下來針對新世代內的的個體，進行規格修剪（rule pruning），去除無意義的規則子，以減少資料收集與判斷成本。判斷是否以達到停止條件，若有，則輸出目前之最佳解；若無，則進行下一步驟。在下一步驟中，若目前的最佳解經過預設的世代數，卻仍未有改善時，則會進行強化步驟（Intensification）。在強化過程中，世代內個體數目（*PopSize*）將會減少；針對連續型資料的搜尋範圍（search domain）亦會縮小；針對基因演算法內的突變機率（*PMut*）也會減少。遞減公式如下所示，其中 ρ^{red} 表示一個使用者自訂遞減參數；$\Delta PopSize$ 表示此一世代內所應遞減之染色體數目；*IPMut* 表示初始設定之突變機率；Nb^{red} 表示目前已遞減的次數。

$$Search\_domain^{new} = \frac{Search\_domain^{old}}{\rho^{red}}$$

$$PopSize^{new} = PopSize^{old} - \Delta PopSize$$

$$PMut = IPMut * EXP(-Nb^{red})$$



圖四 GA-Classifier 之執行流程

本計劃主要利用爪哇語言(JAVA)設計 GA-Classifier，並將 GA-Classifier 分類績效與傳統分類工具決策樹進行比較。

第二年計劃將以蟻群演算法作為主要研究工具，設計一個 Ant-Classifier。此一分類器具有與 GA-Classifier 相同的功能，如表三所示。針對數值型資料，GA-Classifier 將不執行離散化步驟，希望能保有原始資料內所隱藏的資訊；針對數值型資料將可擁有小於等於(≦)以及大於等於(≧)的比較運算符號，而類別型資料將可擁有小於等於(≦)、

大於等於（≧）、等於（＝）以及不等於（≠）的比較運算符號；至於邏輯運算符號部分將可擁有交集（AND）與聯集（OR）符號的選擇。希望透過較多的比較運算符號與邏輯運算符號的選擇，讓多個規則能結合成單一規則，減少產生規則的數目，以方便決策者使用。

表三 Ant-Classifier 功能說明

演算法	數值型資料處理方式	比較運算符號		邏輯運算符號
Ant-Classifier	不離散化	數值型資料	類別型資料	AND、OR
		≦、≧	≦、≧ ＝、≠	

圖五描述了 Ant-Classifier 演算法的內容，每一隻螞蟻所行走的路徑代表一條 IF-THEN 規則。在 Ant-Classifier 內針對連續型屬性產生規則子的方式，先隨機產生一個基點（base point）以及運算符號，並在 Ant-Classifier 設定之搜尋範圍（search domain）隨機產生一個區間值（interval），將基點加上區間值即為該規則子之界限值（bound value），例：variable(x) ≦ base point + interval。Ant-Classifier 評判個體間的績效表現，所使用的績效指標為 Q，其指標內容如表四所示。另外，Ant-Classifier 演算法中亦會進行規格修剪（rule pruning），去除無意義的規則子，以減少資料收集與判斷成本。另外，本計畫在 Ant-Classifier 中加入交配（Crossover）與突變（Mutation）運算，用以增加解演化的多樣性。在突變的過程中，針對連續型屬性，Ant-Classifier 根據突變機率調整界限值，調整方式如下所示。其中 $\Delta(T,R)$ 表示區間值調整量；R 表示搜尋範圍；γ 表示隨機變數從[0..1]；T 表示預設最大的世代數；b 是一個系統參數，用以決定非線性程度。當 $\Delta(T,R)$ 產生完畢後，將原先的區間值加上 $\Delta(T,R)$，即成為新的界限值。

$$\Delta(T,R) = \pm R * (1-\gamma^{(1-T)b})$$

表四 績效指標內容說明

Actual Class \ Predicted Class	Yes	No
Yes	TP: True Positive	FN: False Negative
No	FP: False Positive	TN: True Negative

Ant-Classifier 與 GA-Classifier 不同之處在於：(1) Ant-Classifier 具有正向回饋（positive feedback）之功能，它在演化的過程中所根據演化結果增加或減少特定之解空間發展的機率；(2) 它指派特定數量的 local ants 進行規則的修剪。

本計劃將利用爪哇語言(JAVA)設計一個分類績效較傳統分類工具決策樹準確的 Ant-Classifier，並將 Ant-Classifier 分類績效與 GA-Classifier 以及傳統分類工具決策樹進行比較。

TrainingSet={all training cases};
DiscoveredRuleList=[];
WHILE (TrainingSet > Max_uncovered_cases)
t=1;
j=1;
Create N regions;
Initialize all regions with the same amount of pheromone;
REPEAT
(1) Send "G" global ants for Crossover and Mutation;
a. Crossover:
1. Using normal crossover to exchange each categorical attributes.
2. Using crossover to exchange the base point of each continuous attributes.
b. Mutation:
1. Using general mutation to modify each categorical attribute.
2. Using mutation to randomly adding or subtracting a value to each continuous attribute with a mutation probability. Mutation step size defined as following:
$\Delta(T,R) = \pm R * (1 - \gamma^{(1-T)b})$

(2) The trail value of the newly created child regions is assigned a trail value lying between the values of original parent regions that proportional by crossover point;
(3) Using G newly regions to update the G weakest regions;
(4) Select regions as per normalized trail value of regions and send "L" local ants to do rule pruning;
(5) Calculate each Function Value (Q) of regions;

$$Q = \frac{TP}{TP+FN} * \frac{TN}{FP+TN} \qquad ,Q \in [0,1]$$

(6) Update trail value; $\tau_i(t+1) = \tau_i(t) + \tau_i(t) * Q$
(7) Pheromone evaporates for each trail; $\tau_i(t+1) = \tau_i(t) * \rho$
(8) Sort regions according to trail value;
(9) Choose the best rule R_t among all rules R_N constructed by all the ants according to trail value;
(10) If (R_t is equal ro R_{t-1}) Then
j=j+1;
Else
j=1;
End If
(11) t=t+1;
UNTIL (j>No_rules_coverge or t>No_generation)
Choose the best rule R_{best} among all rules R_t constructed by all the ants;
Add rule R_{best} to DiscoveredRuleList;
TrainingSet= TrainingSet -{set of cases correctly covered by R_{best}};
END WHILE

圖五 Ant-Classifier 演算法之內容

結果與討論

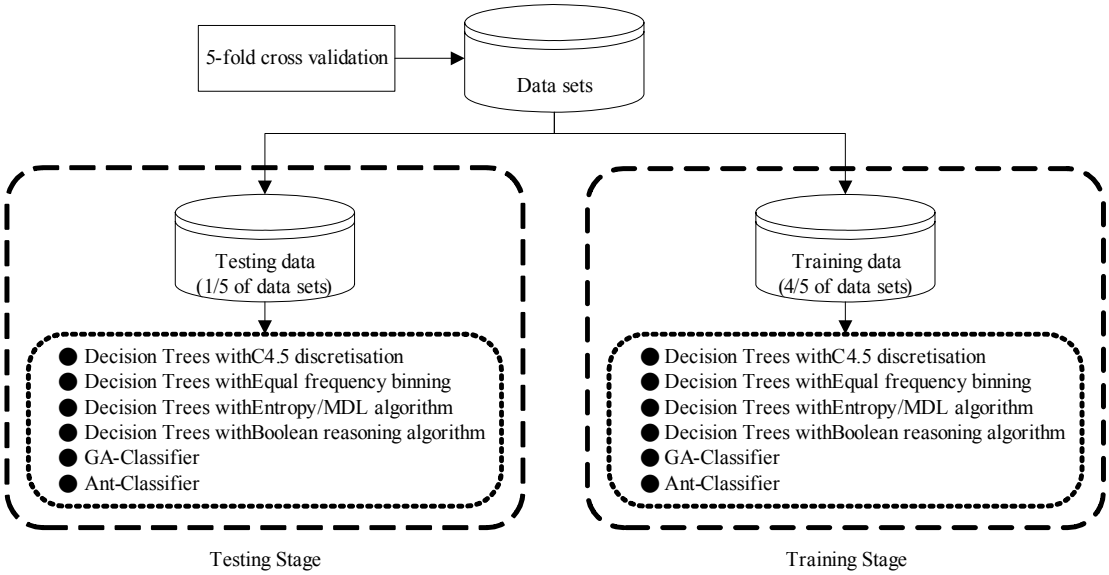
本計劃主要利用爪哇語言(JAVA)設計一個分類績效較傳統分類工具決策樹(Decision Tree)準確的 GA-Classifier 以及 Ant-Classifier。為了進一步驗證 GA-Classifier 以及 Ant-Classifier 分類的能力，本計劃首先採用四種常見的離散化技術（C4.5 discretization、Boolean reasoning algorithm、Entropy/MDL algorithm、Equal frequency binning）搭配決策樹使用，並使用三個標準資料庫 Cleveland、Australian、Iris 進行測試，以了解不同的分類器的分類結果。表五為三個標準資料庫的屬性資料說明。

在 GA-Classifier 內參數設定如下：*IPopSize*=30, *FPopSize*=10, *ΔPopSzie*=5, *IPMut*=0.9, ρ^{red}=2。在 Ant-Classifier 內參數設定如下：*N*=50, *G-Ants*=20, *L-Ants*=5, *No_generation*=100, *No_rules_coverge*=10, *R*=0.5, *ρ*=0.9。

本計畫針對每一個標準資料庫使用 Five-Folds Cross-Validation 方法測試不同分類器的分類結果，結果如圖六所示。表六列示各項離散化技術的平均分類錯誤率。從表六中，我們可以得知使用不同離散化技術進行連續型資料的離散，離散後資料經由決策樹分析後會得到不同分類結果。針對 Cleveland 資料庫，使用 Ant-Classifier 的分類表現最佳；針對 Australian 資料庫，使用 Ant-Classifier 的分類表現最佳；針對 Iris 資料庫，使用 GA-Classifier 的分類表現最佳。這也說明了使用不同的離散化技術搭配決策樹會造成不同程度的資訊流失，使用柔性演算法才能保有原有資料隱藏的資訊，達到較佳的分類結果。總和來說，GA-Classifier 以及 Ant-Classifier 的績效表現均比其他方法來的好，其中 Ant-Classifier 的整體表現略佳於 GA-Classifier。未來的研究方向，由於基因演算法與蟻群演算法在運算過程中需要較多的時間，因此本計畫將朝向提昇 GA-Classifier 以及 Ant-Classifier 的運算效率進行研究。

表五 資料庫說明

資料庫	類別型屬性數	連續型屬性數	資料筆數
Cleveland	7	6	303
Australian	8	6	690
Iris	0	4	150



圖六 實驗架構圖

表六 分類錯誤率

離散化技術 \ 資料庫		Cleveland	Australian	Iris
Ant-Classifier		***41.7%***	***11.5%***	2.6%
GA-Classifier		42.1%	12.6%	***2.4%***
Decision Trees	C4.5 discretisation	43.9%	13.8%	6.0%
	Equal frequency binning	47.2%	12.2%	5.3%
	Entropy/MDL algorithm	44.2%	14.9%	6.7%
	Boolean reasoning algorithm	44.6%	15.1%	2.7%

參考文獻

Bandar, Z, H. Al-Attar and D. McLean, "Genetic algorithm based multiple decision tree induction," *Proceedings of 6th International Conference on Neural Information Processing*, Piscataway, NJ, USA, pp.429-434 (1999).

Berry, M. J. A. and L. Gordon, *Data Mining Techniques for Marketing, Sales, and Customer Support*, John Wiley and Sons, New York, NY (1997).

Bruha, I., P. Kralik and P. Berka, "Genetic learner: Discretization and fuzzification of numerical attributes," *Intelligent Data Analysis*, vol:4, no:5, pp.445-460 (2000).

Cabena, P., P. O. Hadjinian, R. Stadler, J. Vehees and A. Zanasi, *Discoverying Data Mining from Concept to Implementation*, Prentice Hall, New York, NY (1997).

Collard, M, D. Francisci, "Evolutionary data mining: an overview of genetic-based algorithms," *Proceedings of 8th International Conference on Emerging Technologies and Factory Automation,* Piscataway, NJ, USA, pp.3-9 (2001).

Congdon, C. B., "Classification of epidemiological data: a comparison of genetic algorithm and decision tree approaches," *Proceedings of the 2000 Congress on Evolutionary Computation*, Piscataway, NJ, USA, pp.442-449 (2000).

Dorigo, M., V. Maniezzo and A. Colorni, "Ant system: optimization by a colony of cooperating agents," IEEE Transactions on Systems, Man, and Cybernetics—Part B:Cybernetics, vol:26, no:1, pp.29-41 (1996).

Fidelis, M. V., H. S. Lopes and A. A. Freitas, "Discovering comprehensible classification rules with a genetic algorithm," *Proceedings of the 2000 Congress on Evolutionary Computation*, Piscataway, NJ, USA, pp.805-810 (2000).

Fu, Z. and F. Mae, "A computational study of using genetic algorithms to develop intelligent decision trees," *Proceedings of the 2001 Congress on Evolutionary Computation,* Piscataway, NJ, USA, pp.1382-1387 (2001).

Hall, C., "The devil's in the details: techniques, tools, and application for database mining and knowledge discovery part II," *Intelligent Software Strategies*, vol:6, no:9, pp.1-16 (1995).

Lopes, F. M. and A. T. R. Pozo, "Genetic algorithm restricted by tabu lists in data mining," *21st International Conference of the Chilean Computer Science Society,* Los Alamitos, CA, USA, pp.178-185 (2001).

Noda, E, A. A. Freitas and H. S. Lopes, "Discovering interesting prediction rules with a genetic algorithm," *Proceedings of the 1999 Congress on Evolutionary Computation*, Piscataway, NJ, USA, pp.1322-1329 (1999).

Parpinelli, R. S., H. S. Lopes and A. A. Freitas, "Data mining with an ant colony optimization algorithm," *IEEE Transactions on Evolutionary Computation,* vol:6, no:4, pp.321-332 (2002).

Pozo, A. R. and M. Hasse, “A genetic classifier tool,” *Proceedings 20th International Conference of the Chilean Computer Science Society,* Los Alamitos, CA, USA, pp.14-23 (2000).

Shin, K. S. and Y. J. Lee, “A genetic algorithm application in bankruptcy prediction modeling,” *Expert Systems with Applications*, vol:23, no:3, pp.321-328 (2002).

王培珍，應用遺傳演算法與模擬在動態排程問題之探討，中原大學工業工程研究所碩士論文，1996.

計畫成果自評

本計畫主要利用爪哇語言(JAVA)設計分類績效較傳統分類工具決策樹準確的 GA-Classifier 以及 Ant-Classifier，此二分類器分別是以兩種柔性演算法（Genetic Algorithm and Ant System）所發展而成。這兩種分類器將能在不離散數值型資料狀況下，同時處理數值型與類別型資料，從資料中萃取出有用的 IF-THEN 規則，協助管理者進行決策。目前計畫已執行完畢，計畫結果也已達到預期目標。兩個以柔性演算法為基礎之分類器的分類結果均較傳統離散化技術搭配決策樹來得好，其中 Ant-Classifier 表現更比 GA-Classifier 好。本計畫相關執行人員，針對此計畫成果自評滿意，此一研究成果之學術價值非常適合在學術期刊發表，目前已開始著手撰寫學術論文。另外透過執行此項國科會計劃，也使相關參與人員充分了解目前業界相當受重視的資料探勘技術以及柔性演算法，培植參與人員未來的就業的實力，同時參與人員的程式撰寫能力也有顯著地提升。

附錄一　Ant-Classifier 程式碼

```
import java.io.*;
import java.util.*;
import java.sql.*;

public class ant
{
 public static void main(String[] args) throws java.io.IOException
 {

    /* ------------------------------- parameter setting ------------------------------------------------------------------------------ */
    String
SourcePath,TestPath,temp,key,sql,sql_TP,sql_FP,sql_FN,sql_TN,sql_cut,sql_update,FieldName,sql_select,temp_term1,temp_term2,sql_opti
mal,sql_rules;
    int
i,k=0,k1=0,k2=0,l,AttributeCount,DynamicAttribute,NoRegion=50,j,crossover_point=0,selection_point1=0,selection_point2=0,pheromone_
point=0,b=1,a=0,c=0,d=0,p[]=new int[6],ant_L=5,f,h,g,pruning_point=0,term_count=0,temp_ID=0,remaining_records=0,end_count=0;
/*variable p 代表 pruning point*/
    int MaxCategory[];
    Connection MyConn,MyConn1,MyConn2;
    Statement s,s2,s3,s_optimal,s_rules,s_cut;
    Random random;
    double RandomValue,RandomValue1,CategoryValue,step_size,R=0.5,m_rate_c=0.6,m_rate_n=0.6,global_Q,temp_Q=0,initial_Q;
    ResultSet rs,rs1,rs2,rs_optimal,rs_rules,rs_cut;
    float Q,TP=0,FP=0,FN=0,TN=0,update_p,iteration=1,total_pheromone=0,cdf[]=new float[51],global_solution=0;
    boolean check=false,check1=false,check2=false;;
    /* ------------------------------------------------------------------------------------------------------------------------------- */

    /* ------------------------------- 輸入原始資料檔 ------------------------------------------------------------------------------ */
    InputStreamReader stdin=new InputStreamReader(System.in);
    BufferedReader bufin=new BufferedReader(stdin);
    System.out.print("請輸入資料檔之絕對路徑：");
    SourcePath=bufin.readLine();
    File SourceFile=new File(SourcePath);
    /* ------------------------------------------------------------------------------------------------------------------------------- */

    /* ------------------------------- 了解資料結構 ------------------------------------------------------------------------------ */
    System.out.print("請輸入資料檔中之屬性個數：");
    AttributeCount=Integer.parseInt(bufin.readLine());
    System.out.print("請輸入資料檔中之數值屬性個數：");
    DynamicAttribute=Integer.parseInt(bufin.readLine());
    MaxCategory=new int[(AttributeCount+DynamicAttribute+2)];                    /*條件式結果亦要變動，尚未定義結果是否要
變動*/
    for (i=((DynamicAttribute*2)+1);i<=(AttributeCount+DynamicAttribute+1);i++)      /* 因爲要跟 cleveland.candidate 欄位序號相對
應 */
    {
      System.out.print("請輸入資料檔中之第"+(i-(DynamicAttribute*2))+"個類別屬性之最大類別碼：");
      MaxCategory[i]=Integer.parseInt(bufin.readLine());
    }
    /* ------------------------------------------------------------------------------------------------------------------------------- */

    String term[][]=new String[AttributeCount+DynamicAttribute+2][2];
    double delta_Q[]=new double[AttributeCount+DynamicAttribute+1];
    String optimal_term[]=new String[AttributeCount+DynamicAttribute+1];
```

```
  /* ------------------------------ 建立資料庫與原始資料表與結果資料庫 ------------------------------------------------------------------ */
  try
  {
    Class.forName("org.gjt.mm.mysql.Driver");
    MyConn=DriverManager.getConnection("jdbc:mysql://localhost/MySQL","bmw0818","gilber");
    s=MyConn.createStatement();
    k=s.executeUpdate("create database cleveland");
    sql="create table cleveland.source (";
    for (l=1;l<=DynamicAttribute;l++)                                    /*限制兩個以上的數值屬性*/
    {
      FieldName="a"+l+" DOUBLE,";
      sql=sql+FieldName;
    }
    for (l=DynamicAttribute+1;l<=AttributeCount;l++)
    {
      FieldName="a"+l+" TINYINT,";
      sql=sql+FieldName;
    }
    FieldName="a"+(AttributeCount+1)+" TINYINT";
    sql=sql+FieldName;
    sql=sql+")";
    k=s.executeUpdate(sql);
    System.out.println("原始資料庫建立成功");

    /* ---------------------------將資料輸入至資料庫---------------------- */
    BufferedReader Source=new BufferedReader(new FileReader(SourceFile));
    while ((temp=Source.readLine())!=null)
    {
      sql="";
      sql="insert cleveland.source values ("+temp+")";
      k=s.executeUpdate(sql);
      remaining_records=remaining_records+1;                          /*計算原始資料筆數*/
    }

    /* ---------------------------------------------------------------------------- */

    /* ---------------------------建立規則資料庫---------------------------------- */
    sql="create table cleveland.rules (";
    for (l=1;l<=(DynamicAttribute*2);l++)
    {
      FieldName="a"+l+" FLOAT DEFAULT 99,";
      sql=sql+FieldName;
    }
    for (l=(DynamicAttribute*2)+1;l<=(AttributeCount+DynamicAttribute);l++)
    {
      FieldName="a"+l+" TINYINT DEFAULT 99,";
      sql=sql+FieldName;
    }
    FieldName="a"+(AttributeCount+DynamicAttribute+1)+" TINYINT DEFAULT 99,";
    sql=sql+FieldName;
    sql=sql+"Pheromone FLOAT DEFAULT 99, Q FLOAT DEFAULT 99, ID INT PRIMARY KEY AUTO_INCREMENT, CAL CHAR(1)
DEFAULT 'F')";
    k=s.executeUpdate(sql);
    System.out.println("規則資料庫建立成功");
    /* ---------------------------------------------------------------------------- */

    /* ---------------------------建立起始解資料庫---------------------------------- */
```

```
    sql="create table cleveland.candidate (";
    for (l=1;l<=(DynamicAttribute*2);l++)                                    /*限制兩個以上的數值屬性*/
    {
      FieldName="a"+l+" FLOAT DEFAULT 99,";
      sql=sql+FieldName;
    }
    for (l=(DynamicAttribute*2)+1;l<=(AttributeCount+DynamicAttribute);l++)
    {
      FieldName="a"+l+" TINYINT DEFAULT 99,";
      sql=sql+FieldName;
    }
    FieldName="a"+(AttributeCount+DynamicAttribute+1)+" TINYINT DEFAULT 99,";
    sql=sql+FieldName;
    sql=sql+"Pheromone FLOAT DEFAULT 99, Q FLOAT DEFAULT 99, ID INT PRIMARY KEY AUTO_INCREMENT, CAL CHAR(1)
DEFAULT 'F')";
    k=s.executeUpdate(sql);
    System.out.println("候選件資料庫建立成功");
    /* ---------------------------------------------------------------------------- */

    int runs;
    while (remaining_records>10 && end_count<2)                  /**--------------------Ant-Miner 演算法開始啓動----------------**/
    {
    check1=false;
    /* --------------------------將起始解輸入至起始解資料庫---------- */
    random=new Random();
    for (i=1;i<=NoRegion;i++)                                /*50 個候選解集合*/
    {
      sql="insert cleveland.candidate values (";
      sql_TP="select count(*) from cleveland.source where ";
      sql_TN="select count(*) from cleveland.source where (";
      for (l=1;l<=(DynamicAttribute);l++)
      {
            if (random.nextDouble()<0.5)                                    /*以 0.5 機率產生起始解中的條件項*/
            {
          RandomValue=random.nextDouble();
          if (RandomValue<=0.5)
          {
            RandomValue=random.nextDouble();
            sql=sql+RandomValue+",-0.5,";
            sql_TP=sql_TP+"a"+l+"<"+RandomValue+" and "+"a"+l+">"+(RandomValue-0.5)+" and ";
            sql_TN=sql_TN+"a"+l+">"+RandomValue+" or "+"a"+l+"<"+(RandomValue-0.5)+" or ";
          }
          else
          {
            RandomValue=random.nextDouble();
            sql=sql+RandomValue+",+0.5,";
            sql_TP=sql_TP+"a"+l+"<"+(RandomValue+0.5)+" and "+"a"+l+">"+RandomValue+" and ";
            sql_TN=sql_TN+"a"+l+">"+(RandomValue+0.5)+" or "+"a"+l+"<"+RandomValue+" or ";
          }
          check1=true;
        }
        else
        {
          sql=sql+"99,99,";
        }
      }
      for (l=(DynamicAttribute*2)+1;l<=(AttributeCount+DynamicAttribute);l++)
      {
```

```
        if (random.nextDouble()<0.5)                                    /*以 0.5 機率產生起始解中的條件項*/
        {
          RandomValue=random.nextDouble()*(MaxCategory[l]+1);                    /*限定最大亂數產生值*/
          CategoryValue=Math.floor(RandomValue);
          sql=sql+CategoryValue+",";
          sql_TP=sql_TP+"a"+(l-DynamicAttribute)+"="+CategoryValue+" and ";
          if (l==(AttributeCount+DynamicAttribute))
          {
            sql_TN=sql_TN+"a"+(l-DynamicAttribute)+"<>"+CategoryValue+") and ";
          }
          else
          {
            sql_TN=sql_TN+"a"+(l-DynamicAttribute)+"<>"+CategoryValue+" or ";
          }
          check1=true;
        }
        else
        {
          sql=sql+"99,";
        }
      }

      if (sql_TN.substring((sql_TN.length()-4),sql_TN.length()).equals(" or "))
      {
        sql_TN=sql_TN.substring(0,(sql_TN.length()-4));
        sql_TN=sql_TN+") and ";
      }

      RandomValue=random.nextDouble()*(MaxCategory[AttributeCount+DynamicAttribute+1]+1);                    /*隨機產生預測值
*/
      CategoryValue=Math.floor(RandomValue);

      if (check1==false)
      {
            Q=0;
      }
      else
      {
        sql_FP=sql_TP+"a"+(AttributeCount+1)+"<>"+CategoryValue;
        rs=s.executeQuery(sql_FP);
        rs.first();
        FP=rs.getFloat(1);

        sql_TP=sql_TP+"a"+(AttributeCount+1)+"="+CategoryValue;
        rs=s.executeQuery(sql_TP);
        rs.first();
        TP=rs.getFloat(1);

        sql_FN=sql_TN+"a"+(AttributeCount+1)+"="+CategoryValue;
        rs=s.executeQuery(sql_FN);
        rs.first();
        FN=rs.getFloat(1);

        sql_TN=sql_TN+"a"+(AttributeCount+1)+"<>"+CategoryValue;
        rs=s.executeQuery(sql_TN);
        rs.first();
        TN=rs.getFloat(1);
        if ((TP==0 && FN==0) || (TN==0 && FP==0))
```

```
            {
               Q=0;
            }
            else
            {
               Q=(TP/(TP+FN))*(TN/(FP+TN));
            }
         }
         sql=sql+CategoryValue+","+(1+Q)+","+Q+",null,'T')";
         k=s.executeUpdate(sql);
      }

      MyConn2=DriverManager.getConnection("jdbc:mysql://localhost/cleveland","bmw0818","gilber");     /*專供寫入規則資料庫之用*/

       /*---------------------------------------------------------------------------- */

       /*----------------------------------演化過程開始
------------------------------------------------------------------------------------------------------------------------ */
      for (iteration=1;iteration<=100;iteration++)
      {
      /*-----------------------------------------------------------剔除表現較差之解------*/
      sql_cut="select ID from cleveland.candidate order by Q limit 20";    /*--前二十小資料*/
      rs=s.executeQuery(sql_cut);
      sql_cut="delete from cleveland.candidate where ID IN (";
      while (rs.next())
      {
         if (rs.isLast()==true)
         {
            sql_cut=sql_cut+rs.getString(1);
         }
         else
         {
            sql_cut=sql_cut+rs.getString(1)+",";
         }
      }
      sql_cut=sql_cut+")";
      k=s.executeUpdate(sql_cut);
     /*-------------------------------------------------------------------------------------------*/

      MyConn1=DriverManager.getConnection("jdbc:mysql://localhost/cleveland","bmw0818","gilber");
      s=MyConn1.createStatement();
      sql_select="select * from candidate";
      rs=s.executeQuery(sql_select);
      rs1=s.executeQuery(sql_select);

      for (l=1;l<=20;l++)                                          /*產生 20 個新候選解*/
      {
         selection_point1=(int) Math.floor(random.nextDouble()*30)+1;
         selection_point2=(int) Math.floor(random.nextDouble()*30)+1;
         while (selection_point2==selection_point1)
         {
            selection_point2=(int) (Math.floor(random.nextDouble()*30)+1);
         }
         crossover_point=(int) (Math.floor(random.nextDouble()*(AttributeCount-1))+1);
         pheromone_point=crossover_point;
         if (crossover_point<=DynamicAttribute)
         {
            crossover_point=crossover_point*2;
```

```
}
else
{
   crossover_point=crossover_point+DynamicAttribute;
}
rs.moveToInsertRow();
rs1.absolute(selection_point1);
update_p=rs1.getFloat((AttributeCount+DynamicAttribute+2));

/*------------------------------作 crossover & mutation ----------------------*/
for (j=1;j<=crossover_point;j++)
{
   if (j<=DynamicAttribute*2)
   {
      if (j%2==0 && random.nextDouble()<=m_rate_n && rs1.getFloat(j-1)!=99)
      {
         step_size=R*(1-Math.pow(random.nextDouble(),(1-(iteration/100))*b));        /*每一世代演化 100 次*/
         rs.updateFloat(j,(float) (rs1.getFloat(j)+step_size));
      }
      else
      {
         if ((rs1.getFloat(j))!=99)
         {
            rs.updateFloat(j,rs1.getFloat(j));
         }
         else
         {
            if (random.nextDouble()<=m_rate_n && j%2!=0)
            {
                  check2=true;
               rs.updateFloat(j,random.nextFloat());
               if (random.nextDouble()<=0.5)
               {
                  rs.updateFloat(j+1,(-1)*random.nextFloat());
               }
               else
               {
                  rs.updateFloat(j+1,random.nextFloat());
               }
            }
            else
            {
                  if (check2!=true)
               rs.updateFloat(j,99);
            }
         }
         if (j%2==0)
         check2=false;
      }
   }
   else
   {
      if (random.nextDouble()<=m_rate_c)
      {
         CategoryValue=Math.floor(random.nextDouble()*(MaxCategory[j]+1));                  /*限定最大亂數產生值*/
         while (CategoryValue==rs1.getFloat(j))
         {
            CategoryValue=Math.floor(random.nextDouble()*(MaxCategory[j]+1));
```

```
            }
            rs.updateFloat(j,(float) CategoryValue);
        }
        else
        {
            if ((rs1.getFloat(j))!=99)
            {
                rs.updateFloat(j,rs1.getFloat(j));
            }
            else
            {
                rs.updateFloat(j,99);
            }
        }
    }
}
rs1.absolute(selection_point2);
for (j=(crossover_point+1);j<=(AttributeCount+DynamicAttribute+1);j++)
{
    if (j<=DynamicAttribute*2)
    {
        if (j%2==0 && random.nextDouble()<=m_rate_n && rs1.getFloat(j-1)!=99)
        {
            step_size=R*(1-Math.pow(random.nextDouble(),(1-(iteration/100))*b));        /*每一世代演化 100 次*/
            rs.updateFloat(j,(float) (rs1.getFloat(j)+step_size));
        }
        else
        {
            if ((rs1.getFloat(j))!=99)
            {
                rs.updateFloat(j,rs1.getFloat(j));
            }
            else
            {
                if (random.nextDouble()<=m_rate_n && j%2!=0)
                {
                    check2=true;
                    rs.updateFloat(j,random.nextFloat());
                    if (random.nextDouble()<=0.5)
                    {
                        rs.updateFloat(j+1,(-1)*random.nextFloat());
                    }
                    else
                    {
                        rs.updateFloat(j+1,random.nextFloat());
                    }
                }
                else
                {
                    if (check2!=true)
                    rs.updateFloat(j,99);
                }
            }
            if (j%2==0)
            check2=false;
        }
    }
    else
```

```
            {
              if (random.nextDouble()<=m_rate_c)
              {
                CategoryValue=Math.floor(random.nextDouble()*(MaxCategory[j]+1));                    /*限定最大亂數產生值*/
                while (CategoryValue==rs1.getFloat(j))
                {
                  CategoryValue=Math.floor(random.nextDouble()*(MaxCategory[j]+1));
                }
                rs.updateFloat(j,(float) CategoryValue);
              }
              else
              {
                if ((rs1.getFloat(j))!=99)
                {
                  rs.updateFloat(j,rs1.getFloat(j));
                }
                else
                {
                  rs.updateFloat(j,99);
                }
              }
            }
          }
          update_p=update_p*((float) pheromone_point/(float)
AttributeCount)+rs1.getFloat((AttributeCount+DynamicAttribute+2))*(1-((float) pheromone_point/(float) AttributeCount));
          rs.updateFloat((AttributeCount+DynamicAttribute+2),update_p);
          rs.updateString((AttributeCount+DynamicAttribute+5),"F");
          rs.insertRow();
          /* ------------------- crossover & mutation ending ---------------------------------------- */
        }
        /* ----------------------計算新解之 Q 值,並更新所有解之 Pheromone & prunung---------------------------------- */
        rs.close();
        s=MyConn.createStatement();
        sql_select="select sum(Pheromone) from cleveland.candidate";
        rs=s.executeQuery(sql_select);
        rs.first();
        total_pheromone=rs.getFloat(1);
        sql_select="select * from cleveland.candidate";
        rs=s.executeQuery(sql_select);
        s2=MyConn.createStatement();
        s3=MyConn.createStatement();

        /* ---------------------------------決定哪一筆資料需進行 pruning------------ */
        cdf[0]=0;
        d=1;
        while (rs.next())
        {
          cdf[d]=cdf[d-1]+rs.getFloat(AttributeCount+DynamicAttribute+2);
          //System.out.println(d+"-"+cdf[d]);
          d=d+1;
        }
        for (d=1;d<=ant_L;d++)
        {
          p[0]=0;
          f=1;
          check=false;
          while (check==false)
          {
```

```
            RandomValue1=random.nextDouble()*cdf[50];
                //System.out.println(RandomValue1);
                for (f=1;f<=(NoRegion-1);f++)
                {
                  if (RandomValue1>cdf[f] && RandomValue1<cdf[f+1] && f!=p[d-1])
              {
                    p[d]=f;
                    check=true;
                    //System.out.println(f);
                  }
                }
          }
        }
        /*--------------------------------決定哪一筆資料需進行 pruning-------end----*/
        rs.beforeFirst();

        for (d=1;d<=NoRegion;d++)                                    /*while (rs.next())*/
        {
          rs.next();
          temp_ID=rs.getInt(22);
//          System.out.println(temp_ID);
          if (d==p[1] || d==p[2] || d==p[3] || d==p[4] || d==p[5])     /*doing rule pruning*/
          {
            if (rs.getString(AttributeCount+DynamicAttribute+5).equals("F"))                    /*需要計算 Q 值,再進行
pruning,並更新 pheromone*/
            {
              for (a=1;a<=(AttributeCount+DynamicAttribute+1);a++)     /*------------------將資料讀入陣列---------term[0][*]不使用---*/
              {
                if (rs.getFloat(a)!=99)
                {
                  term[a][0]=String.valueOf(rs.getFloat(a));
                  term[a][1]="T";
                }
                else
                {
                  term[a][0]="99";
                  term[a][1]="F";
                }
              }                                                          /*------------------將資料讀入陣列--------------------------*/


              /*判斷剩餘條件項數目*/
              term_count=0;
              for (g=1;g<DynamicAttribute*2;g=g+2)
              {
                if (term[g][1].equals("T"))
                {
                  term_count=term_count+1;
                }
              }
              for (g=(DynamicAttribute*2+1);g<=AttributeCount+DynamicAttribute;g++)
              {
                if (term[g][1].equals("T"))
                {
                  term_count=term_count+1;
                }
              }
```

```
            if (term_count>1)
            {
            /*我覺得應該要從此處進行重複 pruning 程序*/
            sql_TP="select count(*) from cleveland.source where ";
            sql_TN="select count(*) from cleveland.source where (";
            c=0;
            for (a=1;a<(DynamicAttribute*2);a=a+2)
            {
               if (term[a][1].equals("T"))
               {
                  if (Float.parseFloat(term[a+1][0])<=0)
                  {
                        sql_TP=sql_TP+"a"+(a-c)+"<"+term[a][0]+" and
"+"a"+(a-c)+">"+(Float.parseFloat(term[a][0])+Float.parseFloat(term[a+1][0]))+" and ";
                        sql_TN=sql_TN+"a"+(a-c)+">"+term[a][0]+" or
"+"a"+(a-c)+"<"+(Float.parseFloat(term[a][0])+Float.parseFloat(term[a+1][0]))+" or ";
                  }
                  else
                  {
                        sql_TP=sql_TP+"a"+(a-c)+">"+term[a][0]+" and
"+"a"+(a-c)+"<"+(Float.parseFloat(term[a][0])+Float.parseFloat(term[a+1][0]))+" and ";
                        sql_TN=sql_TN+"a"+(a-c)+"<"+term[a][0]+" or
"+"a"+(a-c)+">"+(Float.parseFloat(term[a][0])+Float.parseFloat(term[a+1][0]))+" or ";
                  }
               }
               c=c+1;
            }
            for (a=(DynamicAttribute*2)+1;a<=(AttributeCount+DynamicAttribute);a++)
            {
               if (term[a][1].equals("T"))
               {
                  sql_TP=sql_TP+"a"+(a-DynamicAttribute)+"="+term[a][0]+" and ";
                  sql_TN=sql_TN+"a"+(a-DynamicAttribute)+"<>"+term[a][0]+" or ";
               }
            }
            if (sql_TN.substring((sql_TN.length()-4),sql_TN.length()).equals(" or "))
            {
               sql_TN=sql_TN.substring(0,(sql_TN.length()-4));
            }
            sql_TN=sql_TN+") and ";
            sql_FN=sql_TN+"a"+(AttributeCount+1)+"="+term[AttributeCount+DynamicAttribute+1][0];
            sql_TN=sql_TN+"a"+(AttributeCount+1)+"<>"+term[AttributeCount+DynamicAttribute+1][0];
            sql_FP=sql_TP+"a"+(AttributeCount+1)+"<>"+term[AttributeCount+DynamicAttribute+1][0];
            sql_TP=sql_TP+"a"+(AttributeCount+1)+"="+term[AttributeCount+DynamicAttribute+1][0];

            rs2=s2.executeQuery(sql_FP);
            rs2.first();
            FP=rs2.getFloat(1);
            rs2=s2.executeQuery(sql_FN);
            rs2.first();
            FN=rs2.getFloat(1);
            rs2=s2.executeQuery(sql_TP);
            rs2.first();
            TP=rs2.getFloat(1);
            rs2=s2.executeQuery(sql_TN);
            rs2.first();
            TN=rs2.getFloat(1);
```

```
if ((TP==0 && FN==0) || (TN==0 && FP==0))
{
   Q=0;
}
else
{
   Q=(TP/(TP+FN))*(TN/(FP+TN));
}

//          Q=(TP/(TP+FN))*(TN/(FP+TN));
initial_Q=Q;
sql_update="update cleveland.candidate set Q="+Q+",CAL='T' where ID="+rs.getFloat(AttributeCount+DynamicAttribute+4);
k1=s3.executeUpdate(sql_update);

/*判斷剩餘條件項數目*/
term_count=0;
for (g=1;g<DynamicAttribute*2;g=g+2)
{
   if (term[g][1].equals("T"))
   {
      term_count=term_count+1;
   }
}
for (g=(DynamicAttribute*2+1);g<=AttributeCount+DynamicAttribute;g++)
{
   if (term[g][1].equals("T"))
   {
      term_count=term_count+1;
   }
}

if (term_count>=2)
{
temp_Q=Q;
do
{
   global_Q=-100;
   Arrays.fill(delta_Q,0,AttributeCount+DynamicAttribute+1,-2);
   for (h=1;h<=(AttributeCount+DynamicAttribute);h++)
   {
      temp_term1="";
      temp_term2="";
      if (h<=DynamicAttribute*2)
      {
         if (h%2==1)
         {
            if (term[h][1].equals("T"))
            {
                temp_term1=term[h][0];
                temp_term2=term[h+1][0];
                term[h][0]="99";
                term[h][1]="F";
                term[h+1][0]="99";
                term[h+1][1]="F";

             sql_TP="select count(*) from cleveland.source where ";
                sql_TN="select count(*) from cleveland.source where (";
                c=0;
```

```
                        for (a=1;a<(DynamicAttribute*2);a=a+2)
                        {
                           if (term[a][1].equals("T"))
                           {
                              if (Float.parseFloat(term[a+1][0])<=0)
                              {
                                 sql_TP=sql_TP+"a"+(a-c)+"<"+term[a][0]+" and
"+"a"+(a-c)+">"+(Float.parseFloat(term[a][0])+Float.parseFloat(term[a+1][0]))+" and ";
                                    sql_TN=sql_TN+"a"+(a-c)+">"+term[a][0]+" or
"+"a"+(a-c)+"<"+(Float.parseFloat(term[a][0])+Float.parseFloat(term[a+1][0]))+" or ";
                              }
                              else
                              {
                                 sql_TP=sql_TP+"a"+(a-c)+">"+term[a][0]+" and
"+"a"+(a-c)+"<"+(Float.parseFloat(term[a][0])+Float.parseFloat(term[a+1][0]))+" and ";
                                    sql_TN=sql_TN+"a"+(a-c)+"<"+term[a][0]+" or
"+"a"+(a-c)+">"+(Float.parseFloat(term[a][0])+Float.parseFloat(term[a+1][0]))+" or ";
                              }
                           }
                           c=c+1;
                        }
                        for (a=(DynamicAttribute*2)+1;a<=(AttributeCount+DynamicAttribute);a++)
                        {
                           if (term[a][1].equals("T"))
                           {
                              sql_TP=sql_TP+"a"+(a-DynamicAttribute)+"="+term[a][0]+" and ";
                              sql_TN=sql_TN+"a"+(a-DynamicAttribute)+"<>"+term[a][0]+" or ";
                           }
                        }
                        if (sql_TN.substring((sql_TN.length()-4),sql_TN.length()).equals(" or "))
                        {
                           sql_TN=sql_TN.substring(0,(sql_TN.length()-4));
                        }
                        sql_TN=sql_TN+") and ";
                        sql_FN=sql_TN+"a"+(AttributeCount+1)+"="+term[AttributeCount+DynamicAttribute+1][0];
                        sql_TN=sql_TN+"a"+(AttributeCount+1)+"<>"+term[AttributeCount+DynamicAttribute+1][0];
                        sql_FP=sql_TP+"a"+(AttributeCount+1)+"<>"+term[AttributeCount+DynamicAttribute+1][0];
                        sql_TP=sql_TP+"a"+(AttributeCount+1)+"="+term[AttributeCount+DynamicAttribute+1][0];
//                        System.out.println(sql_FP);
                        rs2=s2.executeQuery(sql_FP);
                        rs2.first();
                        FP=rs2.getFloat(1);
//                        System.out.println(sql_FN);
                        rs2=s2.executeQuery(sql_FN);
                        rs2.first();
                        FN=rs2.getFloat(1);
//                        System.out.println(sql_TP);
                        rs2=s2.executeQuery(sql_TP);
                        rs2.first();
                        TP=rs2.getFloat(1);
//                        System.out.println(sql_TN);
                        rs2=s2.executeQuery(sql_TN);
                        rs2.first();
                        TN=rs2.getFloat(1);

                        if ((TP==0 && FN==0) || (TN==0 && FP==0))
                        {
                                Q=0;
```

```
                    }
                    else
                    {
                       Q=(TP/(TP+FN))*(TN/(FP+TN));
                    }

                    term[h][0]=temp_term1;
                    term[h+1][0]=temp_term2;
                    term[h][1]="T";
                    term[h+1][1]="T";
                    delta_Q[h]=Q-temp_Q;
                    if (delta_Q[h]>global_Q)
                    {
                        global_Q=delta_Q[h];
                    }
                 }
              }
           }
           else
           {
              if (term[h][1].equals("T"))
              {
                 temp_term1=term[h][0];
                 term[h][0]="99";
                 term[h][1]="F";

                sql_TP="select count(*) from cleveland.source where ";
                 sql_TN="select count(*) from cleveland.source where (";
                 c=0;
                 for (a=1;a<(DynamicAttribute*2);a=a+2)
                 {
                    if (term[a][1].equals("T"))
                    {
                       if (Float.parseFloat(term[a+1][0])<=0)
                       {
                          sql_TP=sql_TP+"a"+(a-c)+"<"+term[a][0]+" and
"+"a"+(a-c)+">"+(Float.parseFloat(term[a][0])+Float.parseFloat(term[a+1][0]))+" and ";
                          sql_TN=sql_TN+"a"+(a-c)+">"+term[a][0]+" or
"+"a"+(a-c)+"<"+(Float.parseFloat(term[a][0])+Float.parseFloat(term[a+1][0]))+" or ";
                       }
                       else
                       {
                          sql_TP=sql_TP+"a"+(a-c)+">"+term[a][0]+" and
"+"a"+(a-c)+"<"+(Float.parseFloat(term[a][0])+Float.parseFloat(term[a+1][0]))+" and ";
                          sql_TN=sql_TN+"a"+(a-c)+"<"+term[a][0]+" or
"+"a"+(a-c)+">"+(Float.parseFloat(term[a][0])+Float.parseFloat(term[a+1][0]))+" or ";
                       }
                    }
                    c=c+1;
                 }
                 for (a=(DynamicAttribute*2)+1;a<=(AttributeCount+DynamicAttribute);a++)
                 {
                    if (term[a][1].equals("T"))
                    {
                       sql_TP=sql_TP+"a"+(a-DynamicAttribute)+"="+term[a][0]+" and ";
                       sql_TN=sql_TN+"a"+(a-DynamicAttribute)+"<>"+term[a][0]+" or ";
                    }
                 }
```

```
                    if (sql_TN.substring((sql_TN.length()-4),sql_TN.length()).equals(" or "))
                    {
                       sql_TN=sql_TN.substring(0,(sql_TN.length()-4));
                    }
                    sql_TN=sql_TN+") and ";
                    sql_FN=sql_TN+"a"+(AttributeCount+1)+"="+term[AttributeCount+DynamicAttribute+1][0];
                    sql_TN=sql_TN+"a"+(AttributeCount+1)+"<>"+term[AttributeCount+DynamicAttribute+1][0];
                    sql_FP=sql_TP+"a"+(AttributeCount+1)+"<>"+term[AttributeCount+DynamicAttribute+1][0];
                    sql_TP=sql_TP+"a"+(AttributeCount+1)+"="+term[AttributeCount+DynamicAttribute+1][0];
//                    System.out.println(sql_FP);
                    rs2=s2.executeQuery(sql_FP);
                    rs2.first();
                    FP=rs2.getFloat(1);
//                    System.out.println(sql_FN);
                    rs2=s2.executeQuery(sql_FN);
                    rs2.first();
                    FN=rs2.getFloat(1);
//                    System.out.println(sql_TP);
                    rs2=s2.executeQuery(sql_TP);
                    rs2.first();
                    TP=rs2.getFloat(1);
//                    System.out.println(sql_TN);
                    rs2=s2.executeQuery(sql_TN);
                    rs2.first();
                    TN=rs2.getFloat(1);

                    if ((TP==0 && FN==0) || (TN==0 && FP==0))
                    {
                            Q=0;
                    }
                       else
                    {
                       Q=(TP/(TP+FN))*(TN/(FP+TN));
                    }

                    term[h][0]=temp_term1;
                    term[h][1]="T";
                    delta_Q[h]=Q-temp_Q;
                    if (delta_Q[h]>global_Q)
                    {
                        global_Q=delta_Q[h];
                    }
                 }
              }
           }
           if (global_Q>0)
           {
//                pruning_point=Arrays.binarySearch(delta_Q,global_Q);
              for (g=1;g<=AttributeCount+DynamicAttribute;g++)
              {
                if (global_Q==delta_Q[g])
                {
                   pruning_point=g;
                }
              }

//                System.out.println(global_Q);
```

```
                    if (pruning_point<0)
                    {
                        System.out.println("error");
                        for (g=1;g<=18;g++)
                        {
                        System.out.println(delta_Q[g]);
                        }
                        System.out.println(pruning_point);
                    }
                    if (pruning_point<DynamicAttribute*2)
                    {
                        term[pruning_point][0]="99";
                        term[pruning_point][1]="F";
                        term[pruning_point+1][0]="99";
                        term[pruning_point+1][1]="F";
                    }
                    else
                    {
                        term[pruning_point][0]="99";
                        term[pruning_point][1]="F";
                    }
                    temp_Q=temp_Q+global_Q;
                }
                /*判斷剩餘條件項數目*/
                term_count=0;
                for (g=1;g<DynamicAttribute*2;g=g+2)
                {
                  if (term[g][1].equals("T"))
                  {
                        term_count=term_count+1;
                  }
                }
                for (g=(DynamicAttribute*2+1);g<=AttributeCount+DynamicAttribute;g++)
                {
                  if (term[g][1].equals("T"))
                  {
                        term_count=term_count+1;
                  }
                }

            } while (global_Q>0 && term_count>=2);


            if (temp_Q!=initial_Q)
            {
                sql_update="update cleveland.candidate set ";
                for (g=1;g<=(AttributeCount+DynamicAttribute);g++)
                {
//                      if (term[g][1].equals("T"))                                     /*因爲要將捨棄的條件項輸回原條件中,故不能
使用 if*/
//                      {
                        sql_update=sql_update+"a"+g+"="+term[g][0]+",";
//                     }
                }

sql_update=sql_update+"Q="+temp_Q+",Pheromone="+(rs.getFloat(AttributeCount+DynamicAttribute+2)+rs.getFloat(AttributeCount+Dy
namicAttribute+2)*temp_Q)+",CAL='T'"+" where ID="+temp_ID;
                k1=s3.executeUpdate(sql_update);
```

```
            }
            }

        }
        else
        {
            sql_update="update cleveland.candidate set Q=0 where ID="+temp_ID;
            k1=s3.executeUpdate(sql_update);

        }

        }
        else                                                                    /*直接進行
pruning,並更新 pheromone*/
        {
          for (a=1;a<=(AttributeCount+DynamicAttribute+1);a++)     /*------------------將資料讀入陣列---------term[0][*]不使用---*/
          {
            if (rs.getFloat(a)!=99)
            {
              term[a][0]=String.valueOf(rs.getFloat(a));
              term[a][1]="T";
            }
            else
            {
              term[a][0]="99";
              term[a][1]="F";
            }
          }                                                                  /*------------------將資料讀入陣列--------------------------*/


          /*判斷剩餘條件項數目*/
          term_count=0;
          for (g=1;g<DynamicAttribute*2;g=g+2)
          {
            if (term[g][1].equals("T"))
            {
              term_count=term_count+1;
            }
          }
          for (g=(DynamicAttribute*2+1);g<=AttributeCount+DynamicAttribute;g++)
          {
            if (term[g][1].equals("T"))
            {
              term_count=term_count+1;
            }
          }

          if (term_count>=2)
          {
          temp_Q=rs.getFloat(AttributeCount+DynamicAttribute+3);
          do
          {
            global_Q=-100;
            Arrays.fill(delta_Q,0,AttributeCount+DynamicAttribute+1,-2);
            for (h=1;h<=(AttributeCount+DynamicAttribute);h++)
            {
              temp_term1="";
              temp_term2="";
```

```
                if (h<=DynamicAttribute*2)
                {
                   if (h%2==1)
                   {
                      if (term[h][1].equals("T"))
                      {
                          temp_term1=term[h][0];
                          temp_term2=term[h+1][0];
                          term[h][0]="99";
                          term[h][1]="F";
                          term[h+1][0]="99";
                          term[h+1][1]="F";


                       sql_TP="select count(*) from cleveland.source where ";
                          sql_TN="select count(*) from cleveland.source where (";
                          c=0;
                          for (a=1;a<(DynamicAttribute*2);a=a+2)
                          {
                             if (term[a][1].equals("T"))
                             {
                                if (Float.parseFloat(term[a+1][0])<=0)
                                {
                                   sql_TP=sql_TP+"a"+(a-c)+"<"+term[a][0]+" and
"+"a"+(a-c)+">"+(Float.parseFloat(term[a][0])+Float.parseFloat(term[a+1][0]))+" and ";
                                      sql_TN=sql_TN+"a"+(a-c)+">"+term[a][0]+" or
"+"a"+(a-c)+"<"+(Float.parseFloat(term[a][0])+Float.parseFloat(term[a+1][0]))+" or ";
                                }
                                else
                                {
                                   sql_TP=sql_TP+"a"+(a-c)+">"+term[a][0]+" and
"+"a"+(a-c)+"<"+(Float.parseFloat(term[a][0])+Float.parseFloat(term[a+1][0]))+" and ";
                                      sql_TN=sql_TN+"a"+(a-c)+"<"+term[a][0]+" or
"+"a"+(a-c)+">"+(Float.parseFloat(term[a][0])+Float.parseFloat(term[a+1][0]))+" or ";
                                }
                             }
                             c=c+1;
                          }
                          for (a=(DynamicAttribute*2)+1;a<=(AttributeCount+DynamicAttribute);a++)
                          {
                             if (term[a][1].equals("T"))
                             {
                                sql_TP=sql_TP+"a"+(a-DynamicAttribute)+"="+term[a][0]+" and ";
                                sql_TN=sql_TN+"a"+(a-DynamicAttribute)+"<>"+term[a][0]+" or ";
                             }
                          }
                          if (sql_TN.substring((sql_TN.length()-4),sql_TN.length()).equals(" or "))
                          {
                             sql_TN=sql_TN.substring(0,(sql_TN.length()-4));
                          }
                          sql_TN=sql_TN+") and ";
                          sql_FN=sql_TN+"a"+(AttributeCount+1)+"="+term[AttributeCount+DynamicAttribute+1][0];
                          sql_TN=sql_TN+"a"+(AttributeCount+1)+"<>"+term[AttributeCount+DynamicAttribute+1][0];
                          sql_FP=sql_TP+"a"+(AttributeCount+1)+"<>"+term[AttributeCount+DynamicAttribute+1][0];
                          sql_TP=sql_TP+"a"+(AttributeCount+1)+"="+term[AttributeCount+DynamicAttribute+1][0];
//                          System.out.println(sql_FP);
                          rs2=s2.executeQuery(sql_FP);
                          rs2.first();
```

```
                        FP=rs2.getFloat(1);
//                        System.out.println(sql_FN);
                        rs2=s2.executeQuery(sql_FN);
                        rs2.first();
                        FN=rs2.getFloat(1);
//                        System.out.println(sql_TP);
                        rs2=s2.executeQuery(sql_TP);
                        rs2.first();
                        TP=rs2.getFloat(1);
//                        System.out.println(sql_TN);
                        rs2=s2.executeQuery(sql_TN);
                        rs2.first();
                        TN=rs2.getFloat(1);

                        if ((TP==0 && FN==0) || (TN==0 && FP==0))
                        {
                           Q=0;
                        }
                        else
                        {
                           Q=(TP/(TP+FN))*(TN/(FP+TN));
                        }

                        term[h][0]=temp_term1;
                        term[h+1][0]=temp_term2;
                        term[h][1]="T";
                        term[h+1][1]="T";
                        delta_Q[h]=Q-temp_Q;
                        if (delta_Q[h]>global_Q)
                        {
                            global_Q=delta_Q[h];
                        }
                     }
                  }
               }
               else
               {
                  if (term[h][1].equals("T"))
                  {
                      temp_term1=term[h][0];
                      term[h][0]="99";
                      term[h][1]="F";

                    sql_TP="select count(*) from cleveland.source where ";
                      sql_TN="select count(*) from cleveland.source where (";
                      c=0;
                      for (a=1;a<(DynamicAttribute*2);a=a+2)
                      {
                        if (term[a][1].equals("T"))
                        {
                          if (Float.parseFloat(term[a+1][0])<=0)
                          {
                             sql_TP=sql_TP+"a"+(a-c)+"<"+term[a][0]+" and
"+"a"+(a-c)+">"+(Float.parseFloat(term[a][0])+Float.parseFloat(term[a+1][0]))+" and ";
                        sql_TN=sql_TN+"a"+(a-c)+">"+term[a][0]+" or
"+"a"+(a-c)+"<"+(Float.parseFloat(term[a][0])+Float.parseFloat(term[a+1][0]))+" or ";
                          }
                          else
```

```
                                {
                                    sql_TP=sql_TP+"a"+(a-c)+">"+term[a][0]+" and
"+"a"+(a-c)+"<"+(Float.parseFloat(term[a][0])+Float.parseFloat(term[a+1][0]))+" and ";
                                sql_TN=sql_TN+"a"+(a-c)+"<"+term[a][0]+" or
"+"a"+(a-c)+">"+(Float.parseFloat(term[a][0])+Float.parseFloat(term[a+1][0]))+" or ";
                                }
                              }
                              c=c+1;
                            }
                            for (a=(DynamicAttribute*2)+1;a<=(AttributeCount+DynamicAttribute);a++)
                            {
                              if (term[a][1].equals("T"))
                              {
                                sql_TP=sql_TP+"a"+(a-DynamicAttribute)+"="+term[a][0]+" and ";
                                sql_TN=sql_TN+"a"+(a-DynamicAttribute)+"<>"+term[a][0]+" or ";
                              }
                            }
                            if (sql_TN.substring((sql_TN.length()-4),sql_TN.length()).equals(" or "))
                            {
                              sql_TN=sql_TN.substring(0,(sql_TN.length()-4));
                            }
                            sql_TN=sql_TN+") and ";
                            sql_FN=sql_TN+"a"+(AttributeCount+1)+"="+term[AttributeCount+DynamicAttribute+1][0];
                            sql_TN=sql_TN+"a"+(AttributeCount+1)+"<>"+term[AttributeCount+DynamicAttribute+1][0];
                            sql_FP=sql_TP+"a"+(AttributeCount+1)+"<>"+term[AttributeCount+DynamicAttribute+1][0];
                            sql_TP=sql_TP+"a"+(AttributeCount+1)+"="+term[AttributeCount+DynamicAttribute+1][0];
                            rs2=s2.executeQuery(sql_FP);
                            rs2.first();
                            FP=rs2.getFloat(1);
                            rs2=s2.executeQuery(sql_FN);
                            rs2.first();
                            FN=rs2.getFloat(1);
                            rs2=s2.executeQuery(sql_TP);
                            rs2.first();
                            TP=rs2.getFloat(1);
                            rs2=s2.executeQuery(sql_TN);
                            rs2.first();
                            TN=rs2.getFloat(1);

                            if ((TP==0 && FN==0) || (TN==0 && FP==0))
                            {
                                    Q=0;
                            }
                            else
                            {
                              Q=(TP/(TP+FN))*(TN/(FP+TN));
                            }

                            term[h][0]=temp_term1;
                            term[h][1]="T";
                            delta_Q[h]=Q-temp_Q;
                            if (delta_Q[h]>global_Q)
                            {
                                global_Q=delta_Q[h];
                            }
                        }
                    }
                }
```

```
                if (global_Q>0)
                {
//                    pruning_point=Arrays.binarySearch(delta_Q,global_Q);
                    for (g=1;g<=AttributeCount+DynamicAttribute;g++)
                    {
                      if (global_Q==delta_Q[g])
                      {
                        pruning_point=g;
                      }
                    }

                    if (pruning_point<0)
                    {
                        System.out.print("error");
                    }
                    if (pruning_point<DynamicAttribute*2)
                    {
                        term[pruning_point][0]="99";
                        term[pruning_point][1]="F";
                        term[pruning_point+1][0]="99";
                        term[pruning_point+1][1]="F";
                    }
                    else
                    {
                        term[pruning_point][0]="99";
                        term[pruning_point][1]="F";
                    }
                    temp_Q=temp_Q+global_Q;
                }
                term_count=0;
                for (g=1;g<DynamicAttribute*2;g=g+2)
                {
                  if (term[g][1].equals("T"))
                  {
                        term_count=term_count+1;
                  }
                }
                for (g=(DynamicAttribute*2+1);g<=AttributeCount+DynamicAttribute;g++)
                {
                  if (term[g][1].equals("T"))
                  {
                        term_count=term_count+1;
                  }
                }

            } while (global_Q>0 && term_count>=2);


            if (temp_Q!=rs.getFloat(AttributeCount+DynamicAttribute+3))
            {
                sql_update="update cleveland.candidate set ";
                for (g=1;g<=(AttributeCount+DynamicAttribute);g++)
                {
//                    if (term[g][1].equals("T"))                    /*因爲要將捨棄的條件項輸回條件式中,故不能用 if*/
//                    {
                    sql_update=sql_update+"a"+g+"="+term[g][0]+",";
//                    }
                }
```

```
sql_update=sql_update+"Q="+temp_Q+",Pheromone="+(rs.getFloat(AttributeCount+DynamicAttribute+2)+rs.getFloat(AttributeCount+Dy
namicAttribute+2)*temp_Q)+",CAL='T'"+" where ID="+temp_ID;
                    k1=s3.executeUpdate(sql_update);
                }
                }

            }
          }
          else                                                                                                    /*don't to do rule
pruning*/
          {
            if (rs.getString(AttributeCount+DynamicAttribute+5).equals("F"))
            {
              for (a=1;a<=(AttributeCount+DynamicAttribute+1);a++)     /*------------------將資料讀入陣列---------term[0][*]不使用---*/
              {
                if (rs.getFloat(a)!=99)
                {
                  term[a][0]=String.valueOf(rs.getFloat(a));
                  term[a][1]="T";
                }
                else
                {
                  term[a][0]="99";
                  term[a][1]="F";
                }
              }
              /*判斷剩餘條件項數目*/
              term_count=0;
              for (g=1;g<DynamicAttribute*2;g=g+2)
              {
                if (term[g][1].equals("T"))
                {
                  term_count=term_count+1;
                }
              }
              for (g=(DynamicAttribute*2+1);g<=AttributeCount+DynamicAttribute;g++)
              {
                if (term[g][1].equals("T"))
                {
                  term_count=term_count+1;
                }
              }

              if (term_count>1)
              {
                                                                  /*------------------將資料讀入陣列------------*/
              sql_TP="select count(*) from cleveland.source where ";
              sql_TN="select count(*) from cleveland.source where (";
              c=0;
              for (a=1;a<(DynamicAttribute*2);a=a+2)
              {
                if (term[a][1].equals("T"))
                {
                  if (Float.parseFloat(term[a+1][0])<=0)
                  {
                        sql_TP=sql_TP+"a"+(a-c)+"<"+term[a][0]+" and
"+"a"+(a-c)+">"+(Float.parseFloat(term[a][0])+Float.parseFloat(term[a+1][0]))+" and ";
```

```
                    sql_TN=sql_TN+"a"+(a-c)+">"+term[a][0]+" or
"+"a"+(a-c)+"<"+(Float.parseFloat(term[a][0])+Float.parseFloat(term[a+1][0]))+" or ";
                }
                else
                {
                    sql_TP=sql_TP+"a"+(a-c)+">"+term[a][0]+" and
"+"a"+(a-c)+"<"+(Float.parseFloat(term[a][0])+Float.parseFloat(term[a+1][0]))+" and ";
                    sql_TN=sql_TN+"a"+(a-c)+"<"+term[a][0]+" or
"+"a"+(a-c)+">"+(Float.parseFloat(term[a][0])+Float.parseFloat(term[a+1][0]))+" or ";
                }
              }
              c=c+1;
            }
            for (a=(DynamicAttribute*2)+1;a<=(AttributeCount+DynamicAttribute);a++)
            {
              if (term[a][1].equals("T"))
              {
                sql_TP=sql_TP+"a"+(a-DynamicAttribute)+"="+term[a][0]+" and ";
                sql_TN=sql_TN+"a"+(a-DynamicAttribute)+"<>"+term[a][0]+" or ";
              }
            }
            if (sql_TN.substring((sql_TN.length()-4),sql_TN.length()).equals(" or "))
            {
              sql_TN=sql_TN.substring(0,(sql_TN.length()-4));
            }
            sql_TN=sql_TN+") and ";
            sql_FN=sql_TN+"a"+(AttributeCount+1)+"="+term[AttributeCount+DynamicAttribute+1][0];
            sql_TN=sql_TN+"a"+(AttributeCount+1)+"<>"+term[AttributeCount+DynamicAttribute+1][0];
            sql_FP=sql_TP+"a"+(AttributeCount+1)+"<>"+term[AttributeCount+DynamicAttribute+1][0];
            sql_TP=sql_TP+"a"+(AttributeCount+1)+"="+term[AttributeCount+DynamicAttribute+1][0];

            rs2=s2.executeQuery(sql_FP);
            rs2.first();
            FP=rs2.getFloat(1);
            rs2=s2.executeQuery(sql_FN);
            rs2.first();
            FN=rs2.getFloat(1);
            rs2=s2.executeQuery(sql_TP);
            rs2.first();
            TP=rs2.getFloat(1);
            rs2=s2.executeQuery(sql_TN);
            rs2.first();
            TN=rs2.getFloat(1);

            if ((TP==0 && FN==0) || (TN==0 && FP==0))
            {
                Q=0;
            }
            else
            {
              Q=(TP/(TP+FN))*(TN/(FP+TN));
            }

//              Q=(TP/(TP+FN))*(TN/(FP+TN));
            sql_update="update cleveland.candidate set
Q="+Q+",Pheromone="+(rs.getFloat(AttributeCount+DynamicAttribute+2)+rs.getFloat(AttributeCount+DynamicAttribute+2)*Q)+",CAL=
'T' where ID="+rs.getFloat(AttributeCount+DynamicAttribute+4);
            k1=s3.executeUpdate(sql_update);
```

```
            }
            else
            {
                sql_update="update cleveland.candidate set Q=0,CAl='T' where ID="+temp_ID;
                k1=s3.executeUpdate(sql_update);

            }

            }
            else
            {
               sql_update="update cleveland.candidate set
Pheromone="+(rs.getFloat(AttributeCount+DynamicAttribute+2)+rs.getFloat(AttributeCount+DynamicAttribute+2)*rs.getFloat(AttributeC
ount+DynamicAttribute+3))+" where ID="+rs.getFloat(AttributeCount+DynamicAttribute+4);
               k1=s3.executeUpdate(sql_update);
            }
         }
      }
      /*-------------------------更新 Pheromone 結束-----------------------------------------------*/

/*-------------------------------------------------------------------------------------------------------------------------------------------------------------------
---- */
      }     /**演化結束,100 個世代*/


      /**----------------------------挑選好的候選解----------------------------**/
      sql_optimal="select * from cleveland.candidate order by Q DESC limit 1";
      s_optimal=MyConn.createStatement();
      rs_optimal=s_optimal.executeQuery(sql_optimal);
      rs_optimal.first();
//       System.out.println(rs_optimal.getFloat(19));
      sql_rules="select * from cleveland.rules";
      s_rules=MyConn2.createStatement();
      rs_rules=s_rules.executeQuery(sql_rules);
      rs_rules.first();
      rs_rules.moveToInsertRow();
      for (k2=1;k2<=DynamicAttribute+AttributeCount+3;k2++)
      {
         rs_rules.updateFloat(k2,rs_optimal.getFloat(k2));
      }
      rs_rules.insertRow();
      /**----------------------------挑選好的候選解---end-------------------------**/

      /**----------------------------刪除在原始資料庫中符合最佳解之資料筆----------------------------**/
      sql_cut="delete from cleveland.source where ";
      s_cut=MyConn2.createStatement();

      c=0;
      for (k2=1;k2<(DynamicAttribute*2);k2=k2+2)
      {
         if (rs_optimal.getFloat(k2)!=99)
         {
            if (rs_optimal.getFloat(k2+1)<=0)
            {
               sql_cut=sql_cut+"a"+(k2-c)+"<"+rs_optimal.getFloat(k2)+" and
"+"a"+(k2-c)+">"+(rs_optimal.getFloat(k2)+rs_optimal.getFloat(k2+1))+" and ";
            }
```

```
            else
            {
               sql_cut=sql_cut+"a"+(k2-c)+">"+rs_optimal.getFloat(k2)+" and
"+"a"+(k2-c)+"<"+(rs_optimal.getFloat(k2)+rs_optimal.getFloat(k2+1))+" and ";
            }
          }
          c=c+1;
        }
        for (k2=(DynamicAttribute*2)+1;k2<=(AttributeCount+DynamicAttribute);k2++)
        {
          if (rs_optimal.getFloat(k2)!=99)
          {
            sql_cut=sql_cut+"a"+(k2-DynamicAttribute)+"="+rs_optimal.getFloat(k2)+" and ";
          }
        }
//       sql_cut=sql_cut+"a"+(AttributeCount+1)+"="+rs_optimal.getFloat(AttributeCount+DynamicAttribute+1);   /*1006.修改之*/
        sql_cut=sql_cut+"a"+(AttributeCount+1)+"<="+MaxCategory[AttributeCount+DynamicAttribute+1];        /*1003 修改之*/
        k=s_cut.executeUpdate(sql_cut);
        System.out.println(k);
        if (k==0)
        {
          end_count=end_count+1;
        }
        else
        {
          end_count=0;
        }
        remaining_records=remaining_records-k;
        System.out.println("剩餘未被規則分類之資料筆數:"+remaining_records);

        sql_cut="delete from cleveland.candidate";
        k=s_cut.executeUpdate(sql_cut);

        /**----------------------------刪除在原始資料庫中符合最佳解之資料筆---end----------------------**/
        }     /*----------------------------執行演化迴圈------------end-------------------*/
        System.out.println("Ant-Miner Algorithm runing finished !!!");

    }
    catch(ClassNotFoundException e)
    {
      System.out.print("找不到連線類別檔案");
    }
    catch(SQLException e)
    {
      System.out.print("SQL 語法無法執行："+e);
    }
     /* ------------------------------------------------------------------------------------------------------------------------------------------------------ */

    /**-------------------------計算分類正確率-------------------------**/
    System.out.println("-------------------------------------");
    System.out.println("計算分類預測正確率,請稍後!!");
    TestPath=SourcePath.substring(0,(SourcePath.length()-4))+"_T1.csv";
    System.out.println(TestPath);
//    File TestFile=new File(TestPath);
    /**-------------------------計算分類正確率-------end----------------**/

  }
}
```

出席國際學術會議心得報告

1. 研究之構想與重要性

本計劃主要透過柔性演算法發展分類器，我們在發展的過程中發現樹狀結構對於資料分析有顯著地貢獻。透過樹狀結構將資料分群分類，分別針對每一小群資料進行分析的結果將會比針對全體資料分析還要來得好。因此，本計畫將樹狀結構的概念導入一個新興的機器學習工具—案例式推理（Case-based Reasoning, CBR）。我們希望透過結合樹狀結構以及案例式推理系統，以提高案例式推理系統在連續型數值預測的效率以及預測誤差。本計畫並將所發展的新的案例式推理系統應用於工業工程領域內常見之問題—交期指派（Due Date Assignment, DDA）。

交期指派是現場管理決策的要務之一，決定合適的訂單交期並將產品準時送交客戶，將可有效地提昇客戶服務水準並強化競爭優勢。然而，交期指派確是一個困難的決策。一般而言，訂單的交期均由生管人員利用交期指派法則來預測訂單的完工時間。目前已有相當多以迴歸為基礎之交期指派法則(Regression-based DDA Method)發表於學術期刊，其中比較常見的法則有：Total Work Content (TWK)、Common Slack (SLK), Number of Operations (NOP)、Jobs in System (JIS)、Jobs in Queue (JIQ)、Operation Flowtime Sampling (OFS)、Congestion and Operation Flowtime Sampling (COFS)、…、等等。以迴歸為基礎之交期指派法則有著使用容易且易於瞭解的優點，所以已經廣泛地被使用於實際生產環境中。然而，以迴歸為基礎之指派法則的缺點在於其預測準確度不高。因此，目前有許多學者使用機器學習的工具發展交期指派法則。本計畫針對一個新興機器學習工具—案例式推理（Case-Based Reasoning，CBR）進行研究與改善。案例式推理主要根據過去的求解經驗來解決問題。本計畫為了提高案例式推理對連續型數值的預測能力與效率，透過結合樹狀結構發展一個新的案例標示方法—變異縮減標示法（Variation Reduction-indexing approach）。本論文使用零工式生產環境（Job Shop Environment）來驗證變異縮減標示法的效益，研究結果顯示案例式推理搭配變異縮減標示法使用，將能有效地降低案例式推理的預測誤差，並且能顯著地縮短案例擷取時間，意即此一新的案例式推理系統的表現明顯優於傳統交期指派方法。

2. 研究成果與出席心得

本計畫將此研究構想撰寫成一篇研討會論文（The Variation Reduction-Indexing Approach for Case-Based Reasoning to Predict the Due Dates in a Dynamic Job Shop），並投稿於 2005 IIE Annual Conference。IIE Annual Conference 是工業工程領域中最大且最重要的研討會，屬於全球性研討會。每年都有來自世界各國工業工程相關領域的專家報名參與此一盛會，會議中彼此都會進行交流並相互切磋。今年度的 IIE Annual Conference 於美國亞特蘭大市舉行。本計畫所撰寫之論文於 94 年 5 月 16 日在 Production Planning 8

場次中發表，沙永傑教授並擔任該場次之主持人。本計畫參與人員透過參與此一國際性研討會，學習到許多機器學習以及柔性演算法的相關知識與技術，也拓展了參與人員的國際觀以及提昇參與人員的英文能力。總結來說，此次赴美參與 IIE 研討會收穫甚多，在此特別感謝國科會提供此一機會讓參與人員增廣見聞。

2005 IIE Annual Conference 發表之論文

The Variation Reduction-Indexing Approach for Case-Based Reasoning to Predict the Due Dates in a Dynamic Job Shop

Cheng-Hsiang Liu, D. Y. Sha
Department of Industrial Engineering and Management, National Chiao Tung University
1001 Ta Hsueh Road, Hsinchu, Taiwan (R.O.C.) 30050

ABSTRACT

In this study, a novel case indexing approach is proposed for case based reasoning (CBR). This new approach, called the variation reduction-indexing (VR-indexing) approach, is a modified form of the inductive learning-indexing approach and is applied especially for CBR in numeric prediction. The VR-indexing approach clusters similar cases in the memory by inducting a tree-shaped structure, in order to improve the efficiency and effectiveness of case retrieval. This study applies CBR with VR-indexing approach (VR-CBR) for solving the due date assignment problem in a dynamic job shop environment in order to investigate whether VR-CBR expected benefits can be observed in practice. The results of the experiments show that our proposed VR-CBR can indeed more accurately predict the job due date than the other methods presently in use.

Keywords: Case-based reasoning; Numeric prediction; Due date assignment

1. INTRODUCTION

Case-based reasoning (CBR) is a general problem solving method with a simple and appealing definition [5] that emphasizes the finding of appropriate past experiences as a solution to new problems. The central tasks that CBR methods deal with are the identification of the current problem situation, finding a past case similar to the new one, use that case to suggest a solution to the current problem, evaluate the proposed solution and update the system by learning from this experience [1,6]. In doing these tasks, the most basic problem in CBR is the retrieval and selection of relevant cases, since the remaining operations of adaptation and evaluation will succeed only if the past cases are the relevant ones [8]. The retrieval of relevant cases is closely related and dependent upon the indexing approach used. The indexes organize and label cases in the case base with respect to the aim of deciding under what circumstances the cases may be useful. In this study, a modified form of inductive learning-indexing (IL-indexing) approach is proposed, called the variation reduction-indexing (VR-indexing) approach. It inherits the advantages and characteristics of the IL-indexing approach to assist the CBR method in indexing and retrieving cases for improving the effectiveness of numeric prediction.

In order to evaluate the effectiveness of the VR-CBR in numeric prediction, the VR-CBR is applied to a due date assignment problem in a dynamic job shop environment to investigate whether VR-CBR expected benefits can be observed in practice. The due date assignment is an important task in shop floor control. Assigning exact due dates and timely delivering the goods to the customer will enhance customer's satisfaction as well as provide a competitive advantage. Simulation results show that our proposed VR-CBR is significantly better than

the existing conventional regression-based due date assignment methods and the commonly used machine learning tools in reducing the prediction error.

2. RELEVANT LITERATURE

In order to describe the process of developing the VR-indexing approach to CBR for due date assignment in a dynamic job shop environment, it will be helpful to first discuss the following two areas as background: case indexing and due date assignment.

2.1 Case indexing

The most basic problem in CBR is the retrieval and selection of relevant cases. The retrieval of relevant cases is closely related to and dependent upon the indexing approach used. There are five approaches for case indexing [12]: checklist-based indexing, difference-based indexing, similarity and explanation-based generalization methods, inductive learning methods, and explanation-based techniques. Among these approaches, the inductive learning-indexing (IL-indexing) approaches are widely used (e.g., in Cognitive system's ReMind) and commonly use variants of the ID3 algorithm used for rule induction [12]. The IL-indexing approach clusters cases that are similar to one another and figures out which category best matches the new situation. It then selects the most similar items in that category and adapts it as the new solution. The traditional induction algorithms such as ID3 and C4.5 determine which features do the best job in discriminating cases and then generate a tree-shape classification structure to organize and index the cases in the case memory. However, these algorithms are not suited to index the class that takes on a continuous numeric value.

2.2 Due date assignment

With the current emphasis on the just-in-time (JIT) production philosophy, it is crucial to meet the target job due date. To date, many regression-based due date assignment methods have been proposed, including Constant Allowance (CON), Total Work Content (TWK), Common Slack (SLK), Random Allowance (RAN), Number of Operations (NOP), Jobs in System (JIS), Jobs in Queue (JIQ), Operation Flowtime Sampling (OFS), and Congestion and Operation Flowtime Sampling (COFS), and the advantages of these classical approaches are easy to comprehend and practice. In recent years, many artificial intelligent and machine learning tools have been used for decision support and generating forecasts. Philipoom et al. [9,10] considered a new procedure for internally setting due dates, namely, neural network prediction, in a simple flow shop. Chang et al. [2] and Chiu et al. [3] explored case-based reasoning in the due date assignment problem of the wafer fabrication factory, the experimental results have shown that the CBR approach is very effective and comparable with a neural network approach.

Our purpose here is to begin the development of a novel case indexing approach for CBR for numeric prediction and attempt to see whether the due date setting performance of CBR with this novel case indexing approach can outperform regression-based due date assignment methods that are commonly used in research and in practice, neural networks, and regular CBR.

3. VR-INDEXING CBR

3.1 VR-indexing approach

The new case indexing approach for CBR is called the variation reduction-indexing (VR-indexing) approach. The basic idea behind building the index of cases for predicting the numeric value is inspired by the concept of model trees algorithm [11]. The VR-indexing approach involves two distinct processes. First, a decision tree induction algorithm is used to build a tree for classifying the cases. Instead of maximizing the information gain at each interior node, a splitting criterion is used that minimizes the intra-subset variation in the class values down each branch [11]. The splitting criterion is standard deviation reduction (*SDR*) that given in [11] and is calculated as follows,

$$SDR = sd(T) - \sum_i \frac{|T_i|}{|T|} \times sd(T_i) \quad \text{.....(1)}$$

where T_i and $sd(T_i)$ denote the subset of cases that have the ith outcome of the potential test, and the standard deviation of the target values of cases in T_i respectively. After examining all possible tests, the attribute which maximizes the *SDR* is chosen for splitting at that node. Splitting in the tree ends when the target class values of all the cases that reach a node vary only very slightly, or if only a few cases remain [11]. In this study, we do not split nodes if the branch cannot contain more than 5% of the entire case base of examples. Neither do we split them if the standard deviation of the class values of the examples at the node is less than 5% of the standard deviation of the class values of the entire case base of examples [11]. Second, after the tree is built, each case in the case base will be classified into a leaf. Each leaf in the tree denotes a unique class, which means a particular index. Each case in the case base is indexed according to the leaf into which the case falls.

3.2 Case retrieval and adaptation

When the query case arrives, a class of cases that are similar to the case in question is retrieved based on the feature values of the query one to make the routing decision at each node in the tree-shape classification structure. On this subset of cases, the *k*-NN is performed to find the *k* best matches. The similarity between an old case (say $Case_p$) and a given query case (say $Case_q$) can be measured using the standard Euclidean distance metric. After calculating the similarity between a query case ($Case_q$) and each case in the retrieved class, the *k* cases that are similar to the query case are identified. The expected target value (TV_t) of the query case is derived by Jo et al. [4] and is obtained by Equation (2),

$$E\left(TV_t \middle| \{S_{tb}\}_{b=1,\ldots,n}\right) = \sum_{b=1}^{n} \left(\frac{S_{tb}}{\sum_{i=1}^{n} S_{ti}} \right) \times TV_b \quad \text{.....(2)}$$

Where n, S_{tb}, and TV_b denote the number of cases selected to generate the forecast, the similarity between the query case t and the selected case b, and the flowtime of selected case b for the subject application respectively. Among these, the similarity between the query case and selected case is captured by inversing the distance between them.

The advantages of VR-indexing approach are: (1) it can index the cases at the stage of case retrieval for predicting a “class” that takes on a continuous numeric value, rather than a discrete category into which an

example falls; (2) the VR-indexing approach divides the original case base into several subsets of cases. Retrieving a case from a subset of cases is more efficient than retrieving it from the overall case base.

4. EXPERIMENTAL DESIGN

In this section, we have applied the VR-CBR to solve the due date assignment problem in a dynamic job shop environment, to investigate whether VR-CBR expected benefits are observed in practice.

4.1 The job shop model

In order to evaluate the effectiveness of VR-CBR in solving the due date assignment problem, a suitable shop model needs to be defined. This research used a 10×10 benchmark problem from Lawrence [7]. This test model has ten jobs, each with ten operations and ten machines. In this study, the probability of each product being chosen to be released into the shop is equal. Job inter-arrival times were also selected from a negative exponential distribution, but with a mean 76.5. This resulted in a shop utilization of 90 percent, which represents a heavy shop load. SPT is the dispatching rule that is being used in our study.

4.2 Data collection

This study collected a lot of data using a simulation experiment in a virtual job shop. It is necessary to guarantee statistical independence among the cases before the test is performed. To insure this, once the simulation has reached the steady state, only one in every 50 outputs from the shop simulation is randomly selected to be included in the sample of 15 000 jobs as the case sets. The warm-up period for the shop is the time interval from the start of the simulation to the completion of the first 10 000 jobs. In this study, 48 characteristics and actual flowtime were observed, and are listed in Table 1. Among the 15 000 case points, 5000 points were randomly selected as the raw training data, and the remaining ones were used as the testing data to check the due date predictability of due date assignment methods. Many features were gathered for the collected job. Some features influence the flowtime, while others do not. This experiment acquired influential features by means of screening with a stepwise regression procedure for raw training data in order to select statistically significant input features. Once the influential features were identified, the case of each job in raw training data was represented by these and their actual flowtimes, to arrive in the training data.

4.3 Due date assignment method

For comparison with our proposed VR-CBR, regular case-based reasoning (CBR), back-propagation neural networks (BPN), regression model (REG), jobs in queue (JIQ), and total work content (TWK) due date assignment methods were chosen. For training the VR-CBR and CBR, four fifths of the training data are used as the case base, and the remaining ones are used as the training data for the CBR systems. The REG rule constructed a unique multivariate linear model, based on the information of influential features.

Table 1. Complete data for each collected job

Factor	Information
B1,…, B5	Processing times for operating in 1st, …, 5th bottleneck machine for job *i*
TW	Sum of processing times for job *i*
M1QL,…, M10QL	Sum of the jobs presently in queue on machine 1,…, 10
M1WL,…, M10WL	Sum of the remaining processing time on the machine 1,…, 10 for all the jobs in the shop
NJ1,…, NJ10	Work in process of job 1,…, 10 in the shop
J1R,…, J10R	Average flow time (three lots) of job 1,…, 10, which had most recently completed jobs
SRT	Sum of the remaining processing time for all jobs in the shop
WIP	Work in process in the shop
FT	Actual flowtime in the system of job *i*

5. EXPERIMENT RESULTS

In summary, the procedure described above was followed to generate 5000 data points from the shop as the training data in the virtual job shop. Those data points were used to construct the due date assignment methods of CBR, BPN, REG, JIQ, TWK, and to model the VR-CBR. Once all the methods were constructed during the training phase, as described above, they were tested using the additional 10 000 data points generated specifically for that purpose from the same shop. The 10 000 testing data were formed into 10 batches of 1000 testing data each. The root mean squared error (RMSE) is used consistently to measure the performances of all the due date assignment methods.

The results of the experiment are summarized in Table 2. Each item in Table 2 is an average of the ten batches of the experiment. The results in Table 2 demonstrate that the VR-CBR appears to be the best. Table 3 represents the results of the paired *t*-tests. The due date assignment methods in Table 3 are listed in descending order of performance. They are grouped into homogeneous subsets that are indicated by underline if the difference between the means of performance measure of two methods in the subset is not significantly beyond the prescribed α=0.003 level. The VR-CBR is significantly better than the other methods. In addition, the differences between the CBR and BPN are not significant, but the both methods are significantly better than the REG, JIQ, and TWK methods.

Table 2. The RMSE for each method

Category	Methods	RMSE
Machine learning tools	VR-CBR	***783.89***
	CBR	846.00
	BPN	832.90
Regression-based DDA methods	REG	882.29
	JIQ	931.30
	TWK	960.89

Table 3. Comparison of due date assignment methods

Due date assignment method					
VR-CBR	BPN	CBR	REG	JIQ	TWK

6. CONCLUSIONS

In this study, we presented a new case indexing approach of CBR for numeric prediction, called the variation reduction-indexing approach. This paper examined the effectiveness of VR-CBR for solving the due date assignment problem in a dynamic job shop environment. When the new job arrives, the VR-CBR works as

follows: first, using the tree-shape classification structure it retrieves a class of cases. On this subset of cases, the *k*-NN is performed to find the relevant cases. The VR-CBR is compared with the CBR, BPN, REG, JIQ, and TWK methods with respect to RMSE. The results indicated that the VR-CBR exhibits a performance that is superior to the CBR and other methods. The paired-*t* tests also reinforce the fact that the VR-CBR is the overall best. Future studies might want to focus on investigating whether further improvement can be made by a better design of the VR-CBR method. In addition, using the fuzzy concepts in expressing the feature value would be another worthwhile research topic. Lastly, we would like to explore more applications of VR-CBR in other manufacturing areas.

REFERENCES

[1] Aamodt, A., and Plaza, E., 1994, "Case-based reasoning: foundational issues, methodological variations and system approaches," AI Communications, 7(1), 39-59.

[2] Chang, P.-C., Hsieh, J.-C., and Liao, T.W., 2001, "A case-based reasoning approach for due-date assignment in a wafer fabrication factory," Proceedings of the 4th International Conference on Case-Based Reasoning, Springer-Verlag, Berlin, Germany, 648-659.

[3] Chiu, C.-C., Chang, P.-C., and Chiu, N.-H., 2003, "A case-based expert support system for due-date assignment in a wafer fabrication factory," Journal of Intelligent Manufacturing, 14(3-4), 287-296.

[4] Jo, H., Han, I., and Lee, H., 1997, "Bankruptcy prediction using case-based reasoning, neural networks and discriminant analysis," Expert Systems with Applications, 13(2), 97-108.

[5] Kim, S.H., and Shin, S.W., 2000, "Identifying the impact of decision variables for nonlinear classification tasks," Expert System with Applications, 18(3), 201-214.

[6] Kolodner, J., 1993, Case-Based Reasoning, Morgan Kaufmann, San Mateo, CA.

[7] Lawrence, S., 1984, Resource constrained project scheduling: an experimental investigation of heuristics scheduling techniques, Graduate School of Industrial Administration, Carnegie Mellon University, Pittsburgh.

[8] Lopez de Mantaras, R., 2001, "Case-based reasoning," Lecture Notes in Computer Science 2049, Springer-Verlag, Heidelberg, 127-145.

[9] Philipoom, P.R., Rees, L.P., and Wiegmann, L., 1994, "Using Artificial Neural Networks to Determine Internally-Set Due Date Assignment for Shop Scheduling," Decision Sciences, 25(5/6), 825-847.

[10] Philipoom, P.R., Wiegmann, L., and Rees, L.P., 1997, "Cost-based due-date assignment with the use of classical and neural-network approaches," Naval Research Logistics, 44(1), 21-46.

[11] Wang, Y., and Witten, I.H., 1997, "Induction of model trees for predicting continuous classes," Proceedings of the poster papers of the European Conference on Machine Learning, University of Economics, Faculty of Informatics and Statistics, Prague, 128-137.

[12] Watson, I., and Marir, F., 1994, "Case-based reasoning: a review," The Knowledge Engineering Review, 9(4), 355-381.