運用分群數擾動策略之差分自動分群演算法

李維平
中原大學資管
研究所 助理教授
wplee@cycu.edu.tw

陳慎微
中原大學資管
研究所 研究生
g9794015@cycu.edu.tw

摘要

傳統的分割式分群演算法必須預先知道分群數，本研究提出以分群數擾動策略輔助之差分自動分群演算法(ACDE-O)，可在演化過程中自動調整最佳分群數，運用分群數的擾動策略可避免一般自動分群演算法會受到初始群數解好壞的影響，透過階段性過程中廣泛的搜尋，可避免陷入區域群數最佳解的情況發生。經由與 ACDE 演算法的比較證實，本研究所提出的演算法具有一定程度的有效性與可靠性，在初始群數缺乏多樣性的情形下，仍然有辦法探索到較佳的分群數。

關鍵詞：差分演算法(Differential Evolution), 分割式分群演算法 (Partitional Clustering), K-means 演算法

Abstract

In this paper, an improved Differential Evolution algorithm (ACDE-O) with cluster number oscillation for automatic crisp clustering has been presented. The proposed algorithm needs no prior knowledge of the number of clusters of the data. Rather, it finds the optimal number of clusters on the processing with stable and fast convergence, cluster number oscillation mechanism will search more possible cluster number in case of bad initial cluster number caused bad clusters. Superiority of the proposed algorithm is demonstrated by comparing it with one recently developed partitional clustering algorithm. Experimental results over three real life datasets and the performance of proposed algorithm is mostly better than the other one.

Keywords: Differential Evolution, Partitional Clustering, K-means

1. 緒論

分群 (Clustering) 在資料探勘這項技術領域中是一個非常重要的研究，自 1960 年開始了相關理論與演算法的發展，不同於分類 (Classification) 技術是針對已知的類別標籤來區隔資料，分群是解決非監督式(Unsupervised)問題，依據資料的特性，將相似度高的資料分為一群。一個好的分群結果，是指群內資料相似度高，群和群之間的相似度低[1]。分群技術的應用領域包含資料探勘、影像分類 (Image classification) [2,3]、生物資訊 (Bioinformatics) [4]及網頁搜尋 (Web searching) [5,6]等等，為了配合不同的問題需求與應用上的限制，不同的分群方法也隨之產生。

分群演算法較普遍應用的有階層式分群演算法 (Hierarchical Clustering Algorithm) 和分割式分群演算法 (Partitioning Clustering Algorithm)[7]，兩者皆以距離為基礎來衡量資料間的差異，其餘還有密度基準法 (Density-based method) 和網格基準法 (Grid-based method)[1]等等。階層式分群演算法可分為聚合法 (Agglomerative Algorithm) 和分裂法 (Divisive Algorithm)[8]兩種，階層式分群演算法的優點是無須預先知道分群數為何，可是一但資料的筆數變龐大，其運算成本也會隨之增加，和分割式分群演算法相比之下，較無法在有限時間內得到一個適當解，且資料點一但被分到某群後則不可再做變動與調整，演化 (evolution) 過程的不彈性亦會影響到分群結果[1]。分割式分群演算法不同於階層式分群演算法，需要預先知道分群的數目為何，但是演化過程比較有彈性，可以隨時調整資料點，直到趨近於最佳解為止，最著名的方法如 K-means 演算法。

分群亦可根據欲分割資料的特性分為軟式分群 (Fuzzy Clustering) 和硬式 (Crisp Clustering) 分群兩種。軟式分群又稱模糊分群

，可以把資料點重複分給不同的群，針對重疊性高的資料集有較好的分群效果。硬式分群則是把資料點分給特定的一群，一個資料點只能隸屬於某一群，不得重複[9]。

分群問題會隨著資料集的增加，使問題複雜度成指數型成長且無法在有限時間內找出一個合理的解，在某些特定的目標函式下，當分群數超過 3 群時亦屬 NP-hard 問題[1]。目前針對 NP-hard 或是組合最佳化問題，其求解的方法仍以近似解演算法 (Approximation Algorithm) 為主[10]，其中的演化式演算法是一項熱門的研究標的，較著名的有基因演算法 (Genetic Algorithm，GA) 與粒子群演算法 (Particle Swarm Clustering，PSO)。基因演算法與粒子群演算法同屬於演化式演算法 (Evolutionary Algorithm)，其共同的優點是區域搜尋 (Local Search) 能力佳且演化過程穩定，相較於同樣以區域搜尋著稱的模擬退火法 (Simulated Annealing)，基因演算法與粒子群演算法是以群智能的運算架構來運算，可以平行處理多個可行解，而模擬退火法只能處理一個可行解。

以分割式分群演算法的架構求解分群問題有一個比較大的限制，就是必須要事先給予一個指定的分群數，否則便無法進行求解，有鑑於此，近年來已有一些研究開始提出自動分群演化的機制[2,3,11-18]。其中，差分演算法 (Differential Evolution，DE) 是目前演化式演算法裡被熱門研究的標的，主要是藉由向量間彼此的差異進行向量的突變，在搜尋空間上透過不斷的演化尋找近似最佳解，其特點就在於收斂速度快、有效率、控制參數少且實作容易[19,20]，已有研究證實其求解效能優於基因演算法與粒子群演算法[21,22]。然而，差分演算法依舊會有陷入區域最佳解與演化過程不穩定的隱憂，目前為止，針對差分演算法於國際期刊上發表的自動分群機制並不多，且針對標竿性的分群問題，其實驗結果也並未全然得到最佳解，故仍舊有許多值得探討與實驗的空間。

根據以上的論點，本研究希望能達到以下的目的：

1. 提出一自動分群演算法，以差分演算法為改良標的，不需要預先給定分群數即可準確的在演化過程中找出該資料集的最佳分群數。
2. 改良現有差分分群演算法的機制與架構，改善分群數易陷入區域最佳解 (Local Optimal) 與收斂不穩定的情況，以突破現有自動分群研究的成效。

2. 文獻探討

2.1 分群技術

2.1.1 硬式分群定義

以 A.K. Jain 等人對硬式分群的描述[9]轉換成數學式來表示，假設一個資料集 *S* 有 *n* 筆資料$\{x_1,x_2,\ldots,x_n\}$，每筆資料皆擁有 *d* 個屬性(或稱維度)，則此資料即可表示承矩陣 X_{nxd}，若將資料集分成 *K* 群，則群中心可表示成 $C = \{c_1,c_2,\ldots,ck\}$。

$$C_i \neq \emptyset \text{ for } i = 1, \ldots, K \quad (1)$$

$$C_i \cap C_j = \emptyset \text{ for } i = 1, \ldots, K; j = 1, \ldots, K \text{ and } i \neq j \quad (2)$$

$$\bigcup_{i=1}^{K} C_i = S \quad (3)$$

公式(1)表示任一群不可為空集合，也就是群內至少要有一筆以上的資料，不過，群內有兩筆以上的資料才是有意義的群。公式(2)表示任兩群互斥，即資料不可同時存在於兩群以上的群集中，公式(3)則表示每一群的資料加總起來，必須剛好等於資料集的總數，也就是說，任一筆資料必定隸屬於某一特定群集。

2.1.2 K-means 演算法

K-means (或稱 Hard C-means) 演算法為分割式演算法，是由 MacQueen 於 1967 年所提出的分群演算法[23]，此方法具有理論簡單、實作容易、時間複雜度僅有 *O(n)* 的優點，故廣泛地應用於學術界與業界。K-means 的分群執行步驟為：

Step1：隨機選擇 K 個資料點作為群中心。

Step2：計算所有資料點與各群群中心的距離。

Step3：根據 Step2 的結果將資料點分派給距離最近的群。

Step4：重新計算各群的群中心。

Step5：重複 Step2 至 Step4 直到各群的群中心不再變動為止。

K-means 於操作上相當容易，但也具有以下缺點：

一、 使用者須預先告知分群數為何。
二、 初始解的好壞會影響分群結果。

演化式演算法一直以來都有著會陷入區域最佳解的詬病，故後續都會有研究提出改善其缺失的方法。其中，KGA 演算法是由 Bandyopadhyayb 等學者於 2002 年所提出[24]，以基因演算法的架構為基礎做改良，目的是為了避免 K-means 容易陷入區域最佳解與初始解影響求解品質的問題，但仍然無法大幅的改善分群的穩定性，且執行效率因受限於基因演算法收斂時間較長而導致成效不佳，另外，分群數需預先告知的問題也依然沒有解決。

2.2 差分演算法

差分演算法是以族群為基礎的演化式演算法，是由 Storn 和 Price 兩位學者於 1996 年所提出[25]，其演算法的概念是在搜尋空間中，每個搜尋向量皆代表一個可行解，稱之為解向量，解向量各自擁有自己的方向和大小，解向量的移動是透過彼此向量間的差距，使其在搜尋空間中能不斷的往最佳解方向進行收斂。相較於基因演算法和粒子群演算法，差分演算法在全域搜尋上有較顯著的成效，另外還具有收斂速度快、需調整的參數少以及無論初始解的好壞都能找到全域最佳解的優點[26]。以下為差分演算法的流程圖和演算機制介紹：

一、 突變 (Mutation)
隨機選取三個向量 $X_{r1,G}$、$X_{r2,G}$、$X_{r3,G}$，透過突變因子(Mutation weighting factor,F)取得合成向量(Donor Vector)$V_{i,G+1}$。其運算公式如下: $V_{i,G+1} = X_{r1,G} + F(X_{r2,G} - X_{r3,G})$(4)

二、 重組 (Recombination)
將合成向量與群體中所挑選出來的目標向量 $X_{i,G}$ 透過交配率(Crossover Rate,*CR*)的選擇進行交配並產生試驗向量(Trial Vector)。公式如下：

$$u_{i,j,G+1} = \begin{cases} v_{i,j,G+1} \text{ if rand} \le CR \\ x_{i.j,G+1} \text{ if rand} > CR \end{cases} \quad (5)$$

三、 選擇 (Selection)
透過適應值函數 (fitness function) 來評估目標向量和試驗向量的適應值，選擇適應值較佳的一方進入下一世代，成為下一世代的目標向量。公式如下：

$$X_{i,G+1} = \begin{cases} u_{i,G+1} \text{ if } F(u_{i,G+1}) \le F(x_{i,G}) \\ x_{i,G} \text{ , otherwise} \end{cases} \quad (6)$$

不同於基因演算法，差分演算法是先突變、後選擇，而基因演算法的交配機制則和差分演算法的重組機制有著異曲同工之妙。差分演算法的相關分群研究直到 2004 年以後才開始慢慢增加，Paterlinia 和 Krink 兩位學者於 2004 年[22]和 2006 年[21]分別發表了一篇文章，其內容是比較具指標性的演化式演算法於分割式分群上的成效差異，其中包含了基因演算法、粒子群演算法以及差分演算法，實驗結果顯示，差分演算法在硬式分群的效能上優於基因演算法和粒子群演算法。2005 年由 Omran 等學者所提出的差分演算法[27]應用於影像分割上有不錯的成效，其差分演算法架構在

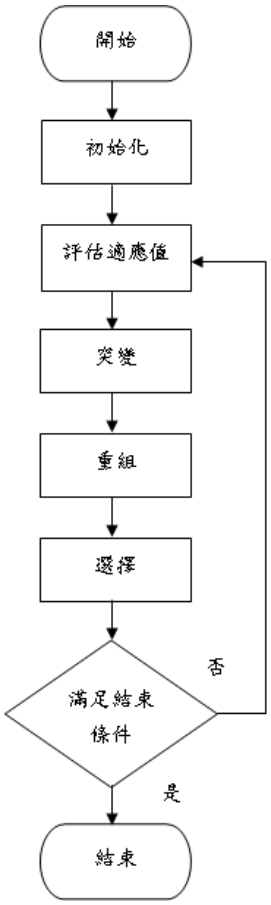

圖 1 差分演算法流程圖

K-means 分群法的概念上，實驗結果也證實了差分演算法在分割式分群上的效能優於其他著名的演化式演算法。直到 2006 年，Das 等學

者進一步提出了自動差分分群演算法[11]，應用於影像分割上，改善了傳統切割式分群法需預先知道分群數的缺點，使其研究成果更適用於現實環境中，爾後也陸續發表了相關自動分群的研究。

然而，相較於基因演算法和粒子群演算法，差分演算法是屬近年才開始應用於自動分群演算法的演化式演算法，其相關研究明顯不足於基因演算法和粒子群演算法，因此還有很大的研究空間。差分演算法是利用其解向量彼此間的差異來引導搜尋過程，容易造成在演化過程中收斂不穩定的結果。因此，本研究嘗試加入分群數擾動的機制，改善差分演算法在收斂過程中的不穩定性，以提升整體演化的效能。

2.3 分群效度指標

分群效度指標是用來評估動態分群結果好壞的方法。最普遍被用來評估的標準有 (1) 緊密度 (Cohesion)：群內的資料特性相近，距離相近。(2) 分離度 (Separation)：群與群之間的資料特定相異，即有相當程度的距離[28]。一個好的分群是指群內資料相似度越高越好，而群間資料相似度越低越好。分群效度指標的評估方法各不相同，針對不同的資料特性而有所調整，故沒有一個分群效度指標能完整的應用與評估所有的資料集。以下將介紹常見的分群效度指標：

一、 Dunn's index (DI) [29]

DI 指標是由 J.C. Dunn 於 1973 年所提出，用來計算分群結果的緊密程度與分離程度。

$$DI(K) = \min_{i\in K}\left\{\min_{j\in K, i\neq j}\left\{\frac{\min_{x\in C_i, y\in C_j} \mathrm{d}(x,y)}{\max_{\mathrm{k}\in \mathrm{K}}\left\{\max_{x,y\in C_k} \mathrm{d}(x,y)\right\}}\right\}\right\} \quad (7)$$

二、 Davies-Bouldin index (DB) [30]

DB 指標是由 D.L. Davies 與 D.W. Bouldin 於 1979 年所提出，其方法是評估組內緊密程度與組間分離程度之間的比率，當不同群集間資料的相異程度越大而群內資料的相異程度越小時，則代表分群效果越佳。

$$DB(K) = \frac{1}{K}\sum_{i=1}^{K} R_{i,qt} \quad (8)$$

$$R_{i,qt} = \max_{j\in K, j\neq i}\left\{\frac{S_{i,q}+S_{j,q}}{d_{ij,t}}\right\} \quad (9)$$

$$S_{i,q} = \left[\frac{1}{N_i}\sum_{x\in C_i}\|x - c_i\|_2^q\right]^{1/q} \quad (10)$$

三、 I Index (I) [31]

I 指標是由 Maulik 和 Bandyopadhyay 兩位學者在 2002 年所提出，主要是計算最大群間距離與平均群內資料和各群間距離的比率，較大的優勢是該指標的運算成本較其他指標來的低。

$$I(K) = \left(\frac{1}{K} \times \frac{E_1}{E_K} \times D_K\right)^P \quad (11)$$

$$E_K = \sum_{k=1}^{K}\sum_{j=1}^{\mathrm{n}} u_{kj}\left\|x_j - z_k\right\| \quad (12)$$

$$D_K = \max_{i,j=1}^{K}\left\|z_i - z_j\right\| \quad (13)$$

四、 Xie-Beni Index(XB)[32]

XB 指標是由 Xie 和 Beni 兩位學者於 1991 年所提出，*XB* 擅長處理模糊分群的評估，以 FCM 演算法驗證後，效果不錯。

$$XB_m = \frac{\sum_{i=1}^{c}\sum_{j=1}^{n} \mathrm{u}_{ij}^2\left\|\overrightarrow{\mathrm{X}_j} - \overrightarrow{\mathrm{V}_i}\right\|^2}{n \times \min_{i\neq j}\left\|\overrightarrow{\mathrm{V}_i} - \overrightarrow{\mathrm{V}_j}\right\|^2} \quad (14)$$

本研究是針對硬式分群架構發展出自動分群演算法，故分群效度指標以硬式分群效度指標為評估基準，2002 年由 Maulik 和 Bandyopadhyay 兩位學者所提出的 index *I*，在 2009 年證實其成效優於其他 8 個評估指標(包含最常用的 *DI* 與 *DB*)[34]，故本研究會採用 index *I* 做為評估動態分群結果的指標，評估效度值為越大越好。運算過程中，隨著分群數和資料量的增加，其運算成本也會跟著增加。

2.4 自動分群相關研究

基因演算法由於發展已久且實作容易，是最常被應用在分群議題上的演化式演算法，早期由 Bandyopadhyay 和 Maulik 在 2002 年以基因演算法提出 GUCK 自動分群演算法[2]，其染色體的編碼是由資料點隨機產生的實數與一些代表非群中心的代號(#)所組成，分群架構以 K-means 為主，實驗結果顯示，針對影像分割的資料集有良好的分割成效。而 Bandyopadhyay 又於 2009 年提出以混合模糊理論的基因分群演算法，可辨別密度分布不同的圖形，改善 GUCK 在分群績效上的表現[16]。粒子群演算法於分群問題上的應用有 Handl 和 Knowles 在 2004 年所提出，搭配多目標最佳化架構的自動分群演算法[18]，運用於文件分群上，而後較備受注目的研究是由 Omran 等人於 2005 年所提出的 DCPSO 自動分群演算法

[3]，結合了 Binary PSO[33]與 K-means 演算法，針對影像分割有不錯的效果。

差分自動分群演算法最早是由 Das 等人於 2006 年所提出[11]，以軟式分群的架構結合差分演算法，應用於影像分割問題上，後來又在 2008 年提出硬式分群架構的差分自動分群演算法 ACDE[12]，實驗顯示其成效優於前段所提的 GUCK 和 DCPSO 演算法，其向量結構採實數設計，以真實資料當作群中心，另外設計一群數遮罩來判斷啟用群中心為何，接著以 K-means 分群架構為主進行距離運算，判斷資料歸屬於何群。後期的差分自動分群演算法大多採用 Das 的向量設計，差別就在於參數上的設計與調整、分群架構與機制上的變化、運用不同的分群效度指標或是修正分群目標式等。

差分演算法不同於基因演算法和粒子群演算法，雖然透過向量差異可快速的收斂並深入探索解空間，但基因演算法採全域最佳解為標的進行探索，粒子群演算法有區域和全域最佳解互相分享資訊，兩者在演化過程的穩定性上較優於差分演算法。因此，不同於目前自動分群演算法研究的改良機制，本研究擬提出一分群數擾動機制，針對最佳解向量(領導向量)的群數解，透過階段性、一定範圍的群數擾動，更進一步的優化該領導向量，目的為提升探索最佳群數的效能，透過廣泛的搜尋，改善以往演化過程容易陷入區域群數解的問題，希望藉此提升整體求解效能的穩定性。

3. 研究方法

目前以差分演算法為架構所提出的自動分群演算法中，改良標的以參數修正最多，其他還有搭配多目標架構、修正目標函數等方法，然而，根據本研究所整理的相關文獻，尚未發現有研究針對解空間中的全域最佳向量解做資訊引入或修正嘗試。有鑑於此，本研究提出一分群數擾動機制，透過階段性、範圍由大至小程度的擾動策略，針對每一代最佳向量解，亦即領導向量，給予其群數解擾動的機會，再透過分群效度指標的評估後，若比原群數解更好則取代之，反之則否。引入分群數擾動機制的目的在於避免分群結果會受到初始群數解好壞的影響，一般的自動分群演算法都是針對解向量的內容，也就是群中心，做相關修正策略，然而針對群數解卻少有研究提出類似的機制，除此之外，分群數擾動機制會提升演化過程的收斂速度，但搭配差分演算法原有的突變(F)與交換因子(CR)，可預防快速收斂的過程中陷入區域最佳解的狀況發生。

3.1 ACDE-O 演算法介紹

以下簡稱本研究提出的演算法為 ACDE-O (Automatic clustering differential evolution with cluster number oscillation

K_{min}	$K_{min}+1$	…	…	…	…	…	K_{max}

圖 2 ACDE-O 群數向量示意圖

$Vec_i,1$	$Vec_i,2$	…	…	…	Vec_i,K_{max}	Index_value	Cluster_num

圖 3 ACDE-O 解向量示意圖

$Vec_i,1$	$Vec_i,2$	…	…	…	Vec_i,K_{max}	0.25	2
$Vec_n,1$	$Vec_n,2$	…	…	…	Vec_n,K_{max}	0.111113	3

圖 4 ACDE-O 解向量示意圖

method)。ACDE-O 採用分群數擾動策略，在差分演化過程結束後，針對該代的領導向量群數給予再次擾動的機會，根據分群效度指標的重新評估後，選擇保留最佳的群數解。

首先，在分群初始化階段，本研究設計了群數向量與解向量來表示分群資料，如圖 2 與圖 3。群數向量的長度是 K_{min} 到 K_{max}，K_{min} 表示最小分群數，K_{max} 表示最大分群數，一般而言，最小分群數至少大於等於 2 才是有意義的分群，最大分群數則看使用者的需要而定，群數向量的作用類似是一個籤筒，每個解向量究竟是要分幾群，就端看從群數向量隨機得到的群數為多少，另外，解向量的長度為 $K_{max}\times$ 資料維度+2，後面 2 個長度會用來存放該解向量的分群效度值與分群數，其餘的位置則是隨機從資料集中挑選資料存放。

啟用群中心即是透過隨機挑選群數向量裡的值後，再隨機從解向量前 $K_{max}\times$ 資料維度個分量裡挑選資料點當作群中心，以圖 4 為例，Vec_i 透過群數向量的隨機抽取後得到分群數為 2，因此再隨機從分量裡決定 2 個群中心資料點，如圖所示為第 2 與第 3 分量，分群後的評估效度指標與分群數也會記錄下來，每條解向量都會透過上述的程序依序決定分群數，過程皆為隨機指派，故有可能其中幾條解向量會重複選到同一分群數，但依據各向量不同分量裡的演化結果，其分群結果就不一定會相同。

在分群演化階段分為三大步驟，首先是要依據先前初始階段得到的分群數和群中心點來進行分群，在文獻探討的部分有討論到，分割式分群問題的領域中，目前判定資料相似度的方法大多以計算兩點間的尤拉距離為主，按照尤拉距離公式的運算，計算資料點與所有群中心的距離，最後將資料點指派給距離最近的群中心。接下來，將母體透過差分演算法的三項機制進行演化，過程中會依序更新各自解向量的區域最佳解與最佳分群數，等母體演化完畢後即可依據分群效度值來判斷全域最佳解是哪一條向量，該向量亦會被選定為該代的領導向量。

最後，啟用分群數擾動機制，將領導向量的群數值取出並給予擾動，擾動的程度會隨著演化代數而遞減，前 33%個世代給予± 3 的擾動範圍，中間 33%到 66%的世代給予± 2 的擾動範圍，最後 33%的世代給予± 1 的擾動範圍，透過初始較大範圍的群數擾動可以增加群數的多樣性，除了有機會快速收斂至最佳群數之外，還可以避免因為初始群數解好壞影響到分群結果的狀況發生，擾動後的群數解需要再次進行分群效度指標的評估，若結果優於原群數解則取代之，反之則不允取代。

3.2 分群數擾動策略

文獻[12]所設計的向量表示，在群數遮罩部分是以機率值來判定群數多寡與群中心的位置，完成之後便以這樣的初始結果進行後續的演化，演化過程中不再調整各解向量的群數與群中心位置，這裡的群中心位置是指位於哪一個分量中，如圖 5，該解向量的群中心位置根據啟用群中心的機率值大於 0.5 所採用，所以結果會是第 2 分量和第 4 分量位置的資料點為群中心。文獻[12]的向量設計潛在著一個問題，那就是分群結果會受到初始群數解的影響，以 Iris 資料集為例，最佳分群數為 3 群，但若是在初始過程中，沒有任何一條解向量被分配到演化群數為 3 的群數，那麼，無論經過多少代的演化，都沒有辦法收斂至群數為 3 的分群結果。

有鑑於此，本研究提出了分群數擾動策略，目的為改善分群結果受到初始群數好壞的影響。分群數擾動策略為階段性的、擾動範圍由大至小程度的修正機制，針對全域領導向量的群數值做預先設定範圍內的擾動，將擾動後的結果再次分群，根據分群效度值的評估結果來判定是否取代原先的分群數。挑選領導向量做為群數擾動的目的在於評估成本與效益考量，由於每一次擾動都需要再做一次分群和評估，執行分群會增加演算法本身的運算成本，而執行評估除了運算成本增加外，其評估次數的增加也意味著演算法演進成本的增加。另外，分群結果是根據演化結束後最好的分群效度值來判定好與否，所以不需考慮整體解向量的平均成效，只要考慮一個最佳解即可。因此，本研究認為針對領導向量做群數擾動不僅可達到初始目的，且不會額外增加太多的運算和評估成本，下面將詳細說明分群數擾動機制的運作流程與方法。

圖 6 為 ACDE-O 的解向量示意圖，該圖是呈現母體有 5 條解向量，最大分群數為 5，資料維度為 3 的 2 維陣列架構，每條解向量的最後 2 個位置是存放分群後的分群效度值和當時分配到的群數值。以 *I* 分群效度指標為例，分群效度值越大代表分群結果越好，故圖 6 可以看出當代的領導向量為向量 4，其分群效度值為 0.201，分群數為 2。圖 6 這樣的向量架構若是碰到 Iris 這類最佳群數解為 3 或是 2、4、5

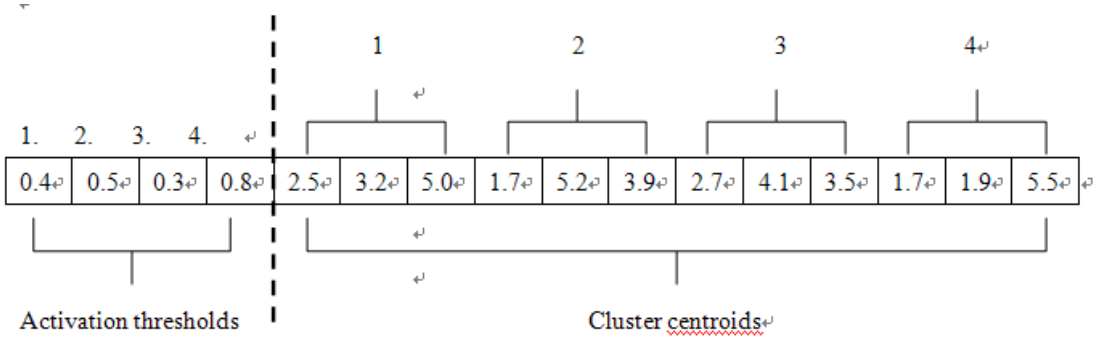

圖 5 ACDE 解向量示意圖

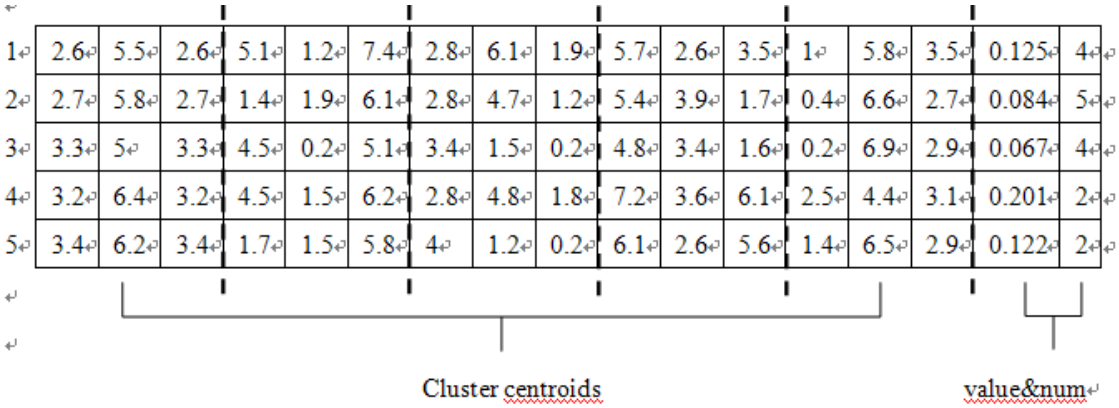

圖 6 ACDE-O 解向量示意圖

以外最佳群數的測試集就會遇到問題，因為初始群數設定好之後就固定不再調整，所以永遠也不會收斂到分群數為 3 或是其他分群數解的分群結果。

而分群數擾動機制此時就會針對向量 4 的群數解做擾動，擾動的依據是按照以下規則：

CASE1 : if ((iter / MAXIT) < 0.33) then new_num = old_num ± 3

當演化代數佔最大演化代數的比例小於 33%，則給予領導向量群數解±3 的擾動範圍值。

CASE2 : if (((iter / MAXIT) ≥ 0.33) and ((iter / MAXIT) < 0.66)) then new_num = old_num ± 2

當演化代數佔最大演化代數的比例大於 33%且小於 66%時，則給予領導向量群數解±2 的擾動範圍值。

CASE3 : if ((iter / MAXIT) ≥ 0.66) then new_num = old_num ± 1

當演化代數佔最大演化代數的比例大於 66%，則給予領導向量群數解±1 的擾動範圍值。

本研究的群數擾動範圍設定在 1 到 3 之間，除非初始的最大分群數和最小分群數的差距非常大，否則擾動範圍不宜太大，以免無法達到快速收斂的目的。至於擾動的持續時間也依據擾動範圍，針對最大演化代數做了平均 3 段的設定，初期透過較大的擾動範圍達到廣泛的探索群數目的，中期則收斂至標準差為 2 的擾動範圍，直到後期就不再大幅度的擾動，因為後期差不多都已經開發到最佳群數解，大幅度的擾動只會造成評估成本的浪費，圖 7 為分群數擾動策略的流程。

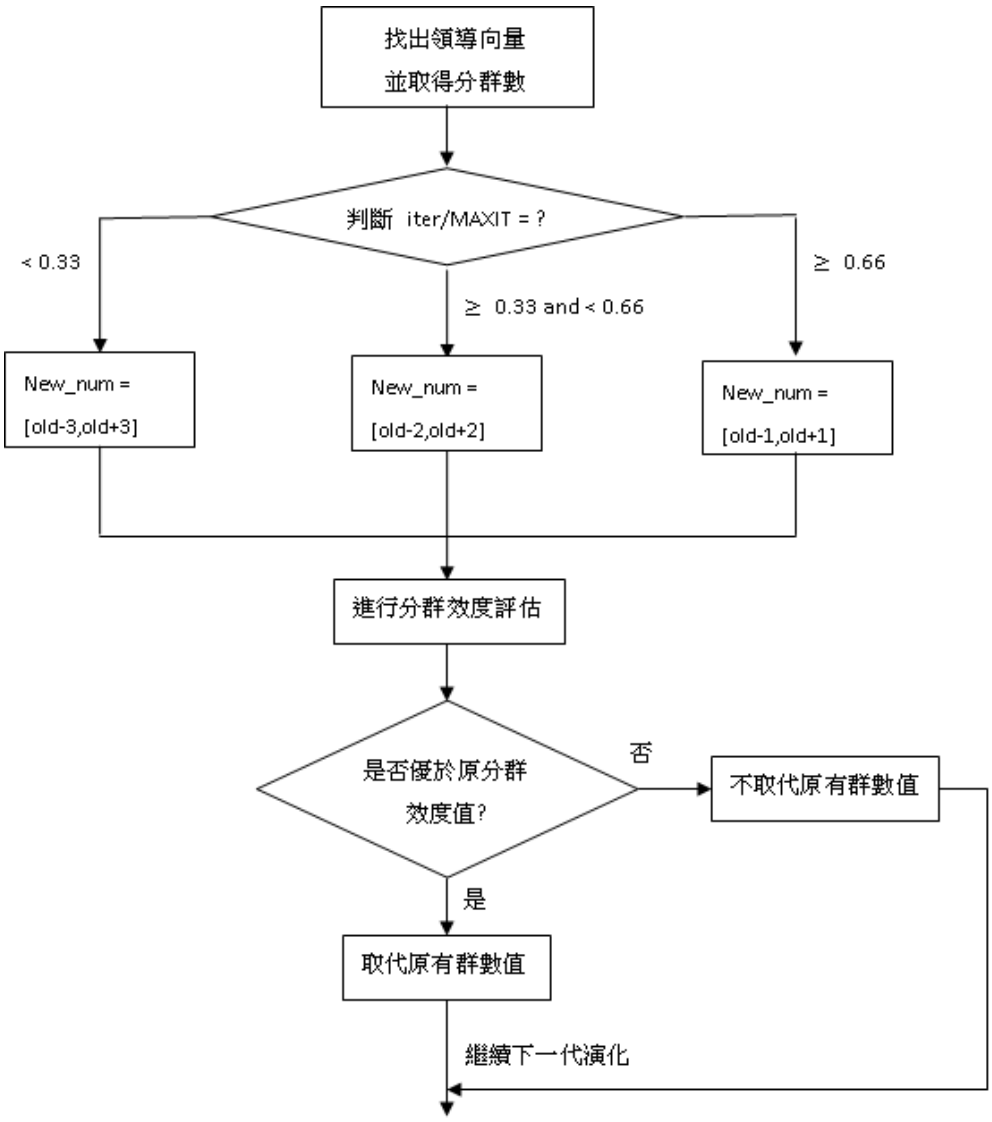

圖 7　分群數擾動策略流程圖

3.3 ACDE-O 演算法虛擬碼

Step1： 初始群數向量與內容向量

Step2： 根據隨機挑選群數向量的群數，找出各內容向量的群中心點

Step3 : For iter=1 to MAXIT

3.1 計算各資料點與所有群中心的距離
3.2 分配各資料點至距離較近的群中心
3.3 檢驗分群結果是否符合硬式分群的條件，若否，則透過懲罰函數給予該群極小的分群效度值，意味該群為失敗的分群結果
3.4 根據差分演算法的 3 項機制和分群效度指標計算，改變解向量
3.5 啟用分群數擾動策略再次更動領導向量的群數解，若評估效度指標後有更佳的分群結果則取代之。

Step4： 分群效度指標最大的向量解即為本次實驗最佳解

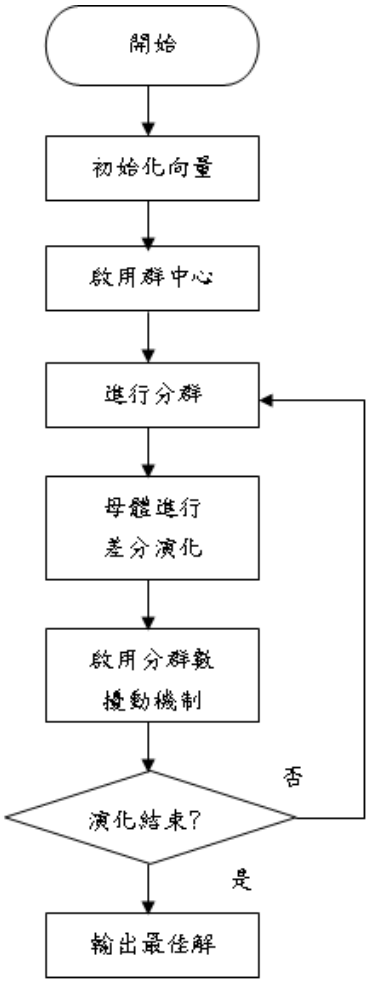

圖 8　ACDE-O 演算法流程圖

4. 實驗結果與分析

4.1 測試資料集

表 1 資料集說明

資料集	資料量	維度	群數	各群筆數
Iris	150	4	3	50,50,50
Cancer	683	9	2	239,444
Liver	345	6	2	145,200

4.2 實驗參數設定

表 2 參數設定

參數	設定值
向量數	20
F	0.5 × (1+rand(0,1))
CR	0.5 × (MAXIT - iter) / MAXIT
K_{max}	20
K_{min}	2

本研究的參數設定採用 ACDE 的原設定，以完整驗證出實驗的有效性與比較後的差異性。向量數由 ACDE 原本的 10 × 資料維度個數量調整成只有 20 個，目的在於強調本研究所提出分群數擾動策略可以在向量數不多而

導致分群數缺乏多樣性的情形下，還能依據分群數擾動策略自行尋找其他可能更優異的分群數，讓分群結果不易受到初始分群數好壞的影響。

4.3 實驗數據與分析

表 3 演算法精確度比較

資料集	演算法	平均群數	標準差
Iris	ACDE	2.2667	0.5208
	ACDE-O	2.0667	0.2537
Cancer	ACDE	3.6667	1.4933
	ACDE-O	2.8667	0.8193
Liver	ACDE	2.9	0.8847
	ACDE-O	2.4	0.6747

表 4 演算法精確度比較

資料集	演算法	I	標準差
Iris	ACDE	0.1520	0.0974
	ACDE-O	0.2407	0.0352
Cancer	ACDE	0.1062	0.072
	ACDE-O	0.1516	0.0788
Liver	ACDE	0.1508	0.0798
	ACDE-O	0.2035	0.0734

實驗結果表 3 和表 4 是執行 400 代，30 次後的平均數據。由實驗數據總體來看，ACDE-O 在平均群數上較 ACDE 來的接近正確群數，標準差也比 ACDE 小，從分群效度指標也可看出 ACDE-O 較 ACDE 的值來的大，亦代表分群效果比較好，其標準差也比較小，證實分群數擾動策略的確在向量數變少、分群數缺乏多樣性的情況下，能有效的達到分群數探索的效果、增加分群數的多樣性，以確保演化過程中能收斂至近似最佳分群數解，提升整體的分群效率。

圖九至十一是 ACDE 與 ACDE-O 兩者在演化過程中的收斂情形，以 Liver 為例，可看出 ACDE 大約在 25 代開始就陷入了停滯，停留在分群數為 3 的群數解上面，這是因為初始群數裡的區域最佳群數就是 3，但反觀 ACDE-O 則是約在 25 代之後，靠分群數擾動機制跳脫出分群數為 3 的區域解，往分群數為 2 的群數解做收斂。

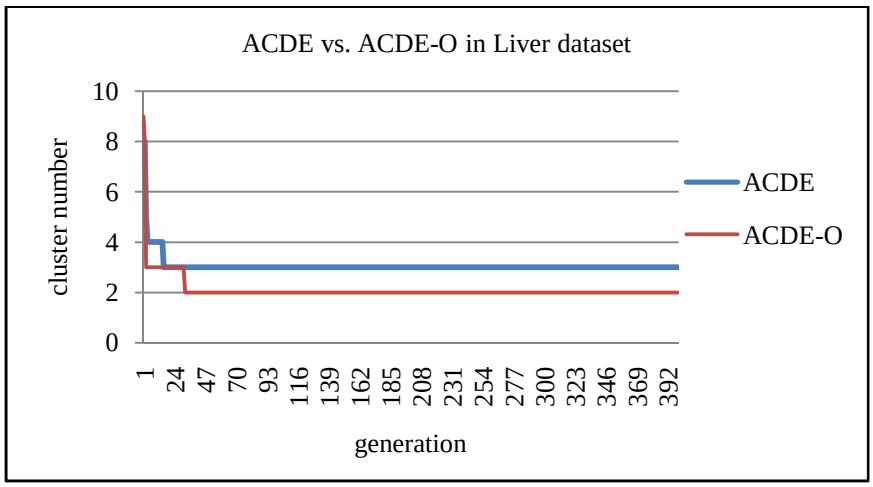

圖 9 Liver 資料集收斂比較圖

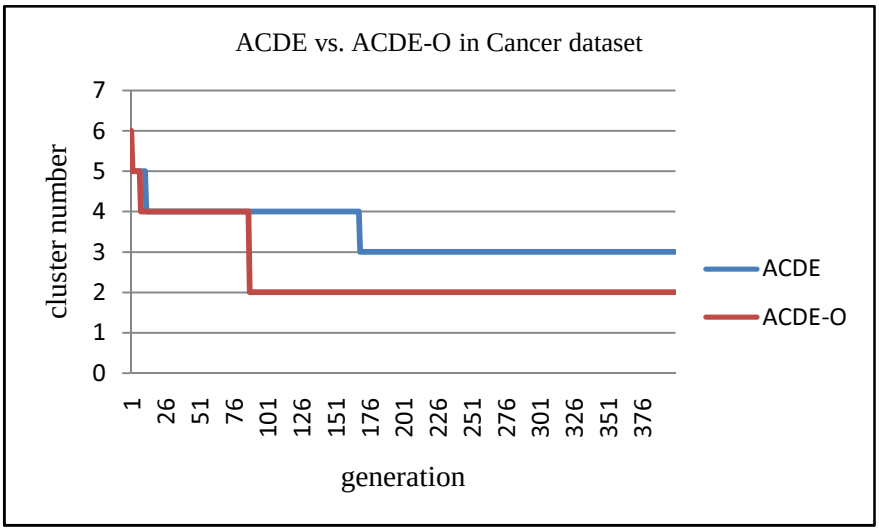

圖 10　Cancer 資料集收斂比較圖

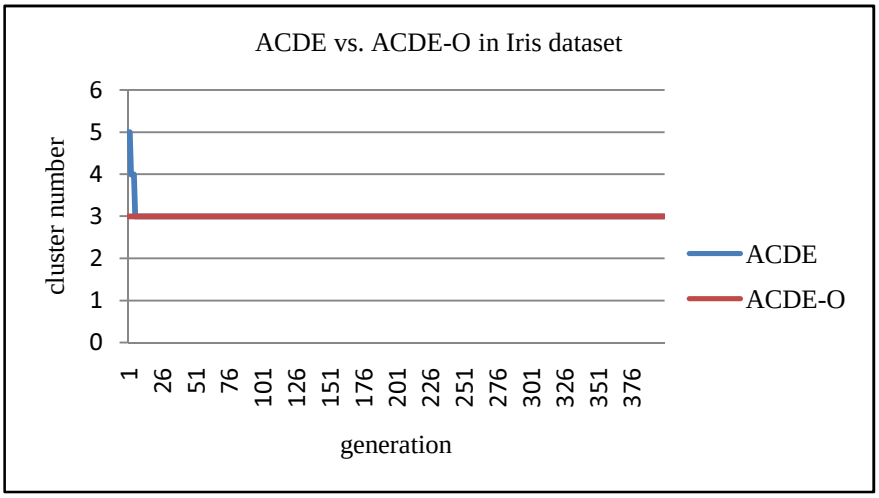

圖 11　Iris 資料集收斂比較圖

5. 結論

透過本研究所提出之分群數擾動策略，可成功的解決一般自動分群演算法遇到分群數缺乏多樣性時，陷入區域最佳群數解的問題。然而目前為止，自動分群演算法相較於一般分群演算法尚未能改善的問題在於，自動分群演算法的求解精確度較差，這是因為自動分群需要解決分群數的問題和群中心解的問題。因此，有許多學者嘗試以多目標的架構或是以模糊分群的架構來解決自動分群問題，希望能藉此提升自動分群演算法的精確度。

本研究所提出的分群擾動策略可應用在上述兩者分群架構或是其他架構的自動分群演算法上面，其理論簡單，實作上也非常容易。本篇研究並未探討差分演算法的突變(F)與交換(CR)因子的變化如何影響自動分群的成效，而這也是本研究在未來將繼續研究的議題。

參考文獻

[1] J. Han and K. Micheline, Data Mining Concepts and Techniques. Morgan Kauffman, 2001.

[2] S. Bandyopadhyay, U. Maulik, “Genetic clustering for automatic evolution of clusters and application to image classification,” Pattern Recognition., vol. 35, no. 6, pp. 1197-1208, Jun. 2002.

[3] M. Omran, A. Salman, and A. Engelbrecht, ”Dynamic clustering using particle swarm optimization with application in unsupervised image classification,” in Proc. 5th World Enformatika Conf. (ICCI), Prague, Czech Republic, 2005.

[4] Y. Chen, K.D. Reilly, A.P. Sprague and Z.

Guan, SEQOPTICS: a protein sequence clustering system, BMC Bioinform 7 (Suppl 4) (2006), p. S10.

[5] R. Krishnapuram, A. Joshi and L.Yi, "A fuzzy relative of the k-medoids algorithm with application to web document and snippet clustering," in IEEE International Fuzzy Systems Conferences, Seoul, Korea, pp. 1281-1286, 1999.

[6] B. Mobasher, R. Cooley and J. Srivatstava, "Creating adaptive web sites through usage-based clustering of URLs," in Knowledge and Data Enginnnring Workshop,1999.

[7] G. Gautam, and B.B. Chaudhuri, "A Novel Genetic Algorithm for Automatic Clustering." Pattern Recognition Letters, Vol. 25, 2004, pp. 173-187.

[8] T.S. Chen, C.C. Lin, Y.H. Chiu and R.C. Chen, "Combined Density- and Constraint-based Algorithm for Clustering," In Proceedings of 2006 International Conference on Intelligent Systems and Knowledge Engineering, 2006.

[9] A.K. Jain, M.N. Murty, and P.J. Flynn, "Data clustering: A review," ACM Comput. Surv., vol.31, no. 3, pp. 264-323, Sep. 1999.

[10] H.-P. Schwefel, Numerical Optimization of Computer Models. Chichester: Wiley, 1981.

[11] S. Das, A. Konar, U.K. Chakraborty, "Automatic Fuzzy Segmentation of Images with Differential Evolution", 2006 IEEE Congress on Evolutionary Computation , Sheraton Vancouver Wall Centre Hotel, Vancouver, BC, Canada July 16-21, 2006.

[12] S. Das, A. Abraham, A. Konar, "Automatic Clustering Using an Improved Differential Evolution Algorithm," IEEE Trans. Syst, Man Cybernetics, Part A, vol. 38 no. 1, pp. 218-237, 2008.

[13] Y. Chen, C. Tang, J. Zhu, C. Li, S. Qiao, R. Li, J. Wu, "Clustering Without Prior Knowledge Based on Gene Expression Programming," Proceedings of the Third International Conference on Natural Computation, Vol. 3, pp. 451-455, 2007.

[14] S. Das, A. Abraham, and A. Konar, "Automatic kernel clustering with a multi-elitist particle swarm optimization algorithm," Pattern Recognition Letters, vol. 29, Issue 5, pp. 688-699, 2008.

[15] Y. Liu, M. Ye, J. Peng, H. Wu, "Finding the Optimal Number of Clusters Using Genetic Algorithms," Cybernetics and Intelligent Systems, pp. 1325-1330, 2008.

[16] S. Saha, S. Bandyopadhyay, "A new point symmetry based fuzzy genetic clustering technique for automatic evolution of clusters," Information Sciences: an International Journal, vol. 2179, Issue. 19, pp. 3230-3246, Sep. 2009.

[17] D. Kundu, K. Suresh, S. Ghosh, S. Das, A. Abraham, Y. Badr, "Automatic Clustering Using a Synergy of Genetic Algorithm and Multi-objective Differential Evolution," Proceedings of the HAIS, 4th International Conference, pp. 177-186, 2009.

[18] J. Handl and J. Knowles. Multiobjective clustering with automatic determination of the number of clusters. Technical Report TR-COMPSYSBIO-2004-02, UMIST, Manchester, UK, 2004.

[19] R. Storn and K. Price. "Differential evolution-A simple and efficient heuristic for global optimization over continuous spaces," J.Glob.Optim., vol. 11, no. 4, pp.341-359, Dec. 1997.

[20] M.M. Ali, and A. Torn, "Population set-based global optimization algorithms:some modications and numerical studies." Comput. Oper: Res., vol.31, issue 10, pp. 1703-1725, Sep. 2004.

[21] S. Paterlinia and T. Krink, "Differential evolution and particle swarm optimization in partional clustering," Comput. Stat. Data Anal., vol.50, no.5, pp. 1220-1247, Mar. 2006.

[22] S. Paterlini and T. Krink, "High performance clustering with differential evolution," in Proc. IEEE Congr. on Evolutionary Computation (CEC'2004), pp. 2004–2011, 2004.

[23] J. MacQueen, "Some Methods for Classification and Analysis of Multivariate Observations," Proc. Fifth Berkeley Symp. Math. Statistics and Probability, vol. 1, pp. 281-296, 1967.

[24] S. Bandyopadhyay and U. Maulik, "An Evolutionary Technique Based on K-Means Algorithm for Optimal Clustering in RN," Information Sciences-Applications: An Int'l J., vol. 146, pp. 221-237, Oct. 2002.

[25] R. Storn and K. Price, "Minimizing the real function of the ICEC'96 contest by differential evolution," in Proc. IEEE Conf. Evolutionary

Computation Nagoya, Japan, 1996, pp. 842-844.

[26] R. Storn, K. Price, “Differential evolution – A Simple and Efficient Heuristic for Global Optimization over Continuous Spaces”, Journal of Global Optimization, vol. 11, Issue. 4, pp. 341-359, 1997.

[27] M. Omran, A.P. Engelbrecht, A. Salman, “Differential Evolution Methods for Unsupervised Image Classification,” Proceedings of the Seventh Congress on Evolutionary Computation (CEC-2005), Edinburgh, Scotland, IEEE Press, 2005.

[28] M. Halkidi, Y. Batistakis, M. Vazigiannis, “On clustering validation techniques,” J. Intell. Inform. Syst. (JIIS), vol. 17, (2-3), pp. 107-145, 2003.

[29] J.C. Dunn, “Well seperated clusters and optimal fuzzy partitions,” J. Cybern., vol. 4, pp. 95-104, 1974.

[30] D.L. Davies and D.W. Bouldin, “A cluster separation measure,” IEEE Trans. On Pattern Analysis and Machine Intelligence, Vol. 1, No. 4, pp. 224-227, 1979.

[31] U.Maulik and S. Bandyopadhyay, “Performance evaluation of some clustering algorithms and validity indices,” IEEE Trans. Pattern Anal. Mach.Intell., vol. 24, no. 12, pp. 1650–1654, Dec. 2002.

[32] X.L. Xie and G. Beni, “A Validity Measure for Fuzzy Clustering,” IEEE Trans. Pattern Analysis and Machine Intelligence, vol. 13, pp. 841-847, 1991.

[33] J. Kennedy and R. Eberhart, “A discrete binary version of the particle swarm algorithm,” in Proc. IEEE Int. Conf. Systems, Man, Cybernetics, Computational Cybernetics, Simulation, vol. 5, 1997, pp. 4104–4108.

[34] S. Saha, S. Bandyopadhyay, Performance Evaluation of Some Symmetry-Based Cluster Validity Indexes, Sys. Man, and Cybernetics, Part C: App. And Reviews, Vol. 39, Issue 4, pp. 420-425, July 2009.