Journal of Science and Engineering Technology, Vol. 4, No. 2, pp. 51-62 (2008)

粒子群最佳化演算法改良之研究

李維平　王雅賢　江正文
中原大學資訊管理研究所
中壢市中北路 200 號

摘　要

標準 PSO（standard particle swarm optimization）具有較少參數設定、快速收斂等優點，但粒子移動時僅跟隨 *pbest* 與 *gbest*，使得標準 PSO 有容易落入區域最佳解的弱點。本研究提出一個分群式粒子群演算法的架構，將初始產生的粒子用 K-means 演算法分群劃分搜尋領域後，以實驗法得到較小的 V_{max} 以加強粒子的區域搜尋能力，再經由比較分群各自找到的分群最佳解 $g_k best$，產生全域最佳解，此為分群式粒子群演算法（K-means particle swarm optimization, KPSO）；另外，為確保演算法的收斂性，本研究將文化演算法「知識空間」的概念帶入了 KPSO 中，由知識空間中的粒子來引導主群體粒子前往具良好解答區搜尋，此為文化分群式粒子群演算法（culture K-means particle swarm optimization, CKPSO）。藉由此兩個 PSO 的改良演算法，以期提高粒子搜尋到之全域最佳解的準確度。由研究結果可得知 KPSO 與 CKPSO 在測試函數中的表現，整體來說均能優於過去學者提出之標準 PSO、HPSO（hybrid particle swarm optimization）、FPSO（fuzzy adaptive particle swarm optimization）。

關鍵詞：群體智慧，粒子群最佳化演算法，文化演算法，K-means 演算法

Research on a Modified Particle Swarm Optimization Algorithm

WEI-PING LEE, WANG YA XIAN and CHIANG CHENG WEN
Department of Management Information Systems, Chung Yuan Christian University
200, Chung Pei Rd.,Chung Li, Taiwan 32023, R.O.C

ABSTRACT

According to Particle Swarm Optimization (PSO), all particle were guided to search by *pbest* and *gbest*. For preventing the particle trap into local minima, this article purpose K-main PSO. After swarming off the search region of initial particles with K-main Algorithm, the smaller V_{max} was got by experimentation to enhance searching ability. All $g_k best$ were got by comparing particles in it's own group and the *gbest* was got comparing all $g_k best$. For keeping convergence, this article purpose CKPSO (culture K-means particle swarm optimization). All particle groups were guide to search by particles which were in knowledge space. The experimental results show that KPSO (K-means particle swarm optimization) and CKPSO are effective and gain better performance then SPSO (standard particle swarm optimization), HPSO (hybrid particle swarm optimization) and FPSO (fuzzy adaptive particle swarm optimization).

Key Words: collective intelligence, particle swarm optimization, cultural algorithms, K-means algorithms

一、緒論

粒子群演算法（particle swarm optimization, PSO）是由 Eberhart 和 Kennedy 於 1995 年所提出 [11, 16]，是一種具有群體智慧的方法，也屬於演化式計算中的一個新分支。學者藉由觀察鳥群覓食的社會類爲得到啓發，而將其應用於解決搜尋與最佳化的相關問題。如同一隻在空間中飛行的鳥一樣，稱作「粒子（particle）」，在空間中移動的所有粒子都有一個由目標函數所映射的適應值，每個粒子還有一個速度來決定他們移動的方向與距離，粒子於解空間中的飛行受到兩個因子影響：1. 個體最佳適應值記憶（*pbest*），2. 群體最佳適應值記憶（*gbest*）[17]。PSO 中的每個粒子獨立搜尋，依照此個體最佳適應值記憶去修正下一次的搜尋方向，此稱爲粒子的認知模式（cognition-only model）。同時每個粒子也依照群體最佳適應值來修正下一次粒子的搜尋速度，此謂爲粒子群的社會模式（social-only model）。經過如此疊代（iterations）計算後，PSO 根據粒子群中最佳適應值，計算出問題的最佳解 [12]。

標準 PSO 演算法除了具有通用性，適用於各種最佳化問題的求解 [2-5, 10, 27]，諸如：多目標函數求解、最佳參數搜尋、排程問題…等，而且能夠比其它已成熟的進化演算法（如：遺傳演算法、蟻群演算法）有更好的效能 [9, 12, 13]。目前標準 PSO 演算法與各種改進之 PSO 演算法，大部份著眼於如何更有效地利用一個粒子群在解空間中搜尋最佳解 [1, 8, 11, 20, 24-26]，然而粒子在解空間中搜尋時，僅僅追逐當前的群體最佳適應值記憶，以及個體過去的最佳適應值記憶，如此將限制了粒子的搜尋範圍，同時可能使粒子落入局部最佳解而無法跳脫。因此，本研究提出一分群式 PSO 演算法，於一個訓練世代中，將解空間裡的粒子群以 K-means 演算法分群將搜尋領域切割，以較小的速度分別於解空間中搜尋群內最佳解，如此可擴大粒子的搜尋範圍，再將找到的候選最佳解經比較後，產生較佳的最佳解，期望能提高最佳解的準確度。另外，爲提高粒子於解空間中的搜尋效能以及複雜問題的處理能力，本研究將以文化演算法所使用之「知識空間」的概念來引導粒子搜尋，於達到所設定的疊代數時，淘汰表現較差的粒子，並以知識空間中表現較好的粒子來取代其在空間上的位置。

二、文獻探討

（一）粒子群最佳化演算法

粒子群演算法是一種具有群體智慧的方法，其一開始以初始化方式產生一群隨機的粒子，透過疊代以找到最佳解，在每一次的疊代中，粒子的移動受到自身目前爲止所搜尋到的最佳適應值記憶 *pbest*，以及社交鄰居中目前爲止所搜尋到的最佳適應值記憶 *nbest* 影響，我們也可以把母體中所有的粒子都視爲鄰居，則群體所搜尋到的最佳適應值記憶爲 *gbest*，*gbest* 可視爲 *nbest* 的特例 [14]。粒子在 d 個維度中的速度需受到最大速度 $V_{\max}$ 的控制，$V_{\max}$ 影響了粒子在目前位置與目標之間的搜尋能力。粒子在解空間中依照公式（1）、（2）來改變其位置與速度：

$$V_{id}(t) = V_{id}(t-1) + c_1 \times Rand(\,) \times (P_{id} - X_{id}) + c_2 Rand(\,) \times (P_{gd} - X_{id}) \quad (1)$$

$$X_{id}(t) = X_{id}(t-1) + V_{id}(t) \quad (2)$$

線性遞減權重（inertia weight）的概念於 1998 年由 Shi 和 Eberhart 首先提出 [24, 26]，爲原始的 PSO 演算法加入了線性遞減權重 *w* 這個因子，形成了當前 PSO 的標準版本。公式（3）與（4）爲增加了 *w* 後的新公式，*w* 的設置改善了許多應用的表現，在過去的研究實驗中 *w* 通常被設置由 0.9 線性遞減至 0.4。合適的數值設定將可以提供局部和全域一個平衡的探索（exploitation）及開發（exploration）能力，較大的 *w* 可使粒子具備較大的開發能力，較小的 *w* 能使粒子具備探索能力 [12]，*w* 的加入也使得演算法減低了每回合需小心設定 $V_{\max}$ 的需求。

$$V_{id}(t) = w \times V_{id}(t-1) + c_1 \times Rand() \times (P_{id} - X_{id}) + c_2 \times Rand() \times (P_{gd} - X_{id}) \quad (3)$$

$$X_{id}(t) = X_{id}(t-1) + V_{id}(t) \quad (4)$$

$$w = \frac{(w_{ini} - w_{end}) \times (T_{\max} - t)}{T_{\max}} + w_{end} \quad (5)$$

（二）K-means 演算法

K-means 演算法 [18] 是一個簡單而且有效的統計學分群技巧，相較於其它分群方法，其運算較爲簡單，分群也較爲快速。其分群方式是設立初始中心點，然後利用歐幾里德

距離的概念，將相近的資料點歸類到同一群，然後再重新計算中心點，重覆步驟直到中心點收斂爲止，演算法步驟如下：

Step1：給定一個值 K，K 表示所要分群的數目。

Step2：從 n 個所有資料點中，以亂數的方式選擇 K 個起始中心點，當作初始分群的中心。

Step3：使用歐幾里德距離來分配剩餘的資料點，將資料分到距離它們最近的群集中心。

Step4：重新計算新的中心點。

Step5：假設產生的 K 個新中心點和舊的中心點相同時，即終止分群動作；否則，回到 Step3 繼續分群。

（三）文化演算法

傳統的演化式計算只能隱性的表示和保存知識訊息，而文化演算法提供了一種顯性的機制來獲取、保存和整合演化群體在尋求最佳解的知識和經驗 [22]。文化演算法模擬了人類文化演化的計算模型，如圖 1 所示 [7, 15, 21]。主群體空間是演算法主要進行問題求解運算的主要空間，透過疊代演化和適應值評估來進行求解；知識空間則透過自身的演化策略以進行更新，並對主群體空間的進一步演化做引導。

文化演算法可以看做上下兩層空間框架或模型。下層的主群體空間和上層知識空間各自保存自己的群體，並各自獨立進行演化。一般而言，下層空間定期貢獻精英個體給上層空間，上層空間不斷進行演化自己的精英群體，反過來影響（或控制）下層空間群體，最終形成「雙演化雙促進」機制。文化演算法模型或框架針對不同具體問題可以採用不同內涵的主群體空間和知識空間，具有廣泛的應用前景。

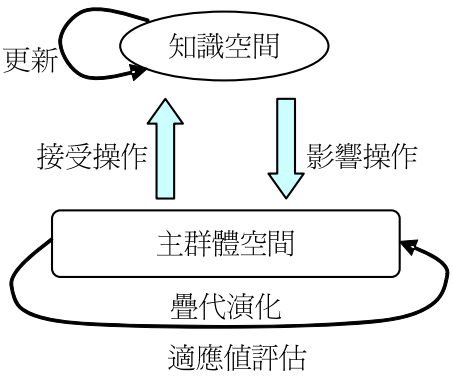

圖 1. 文化演算法架構

三、研究方法

（一）方法一：分群式粒子群演算法（K-means Particle Swarm Optimization, KPSO）

本研究在初始化隨機產生粒子後，隨機選出 K 個點當作初始中心，接下來以 K-means 演算法將空間中其它粒子分群，藉由分群將粒子的搜尋空間做切割，輔以較小的 $V_{\max}$ 使粒子具有較強的區域搜尋能力，由分群後的粒子分頭於解空間中搜尋更佳的適應值。群內粒子的移動只受粒子自身的最佳適應值記憶 *pbest* 以及分群內最佳適應值記憶 $g_k best$ 影響，不受其它分群粒子的影響，分群後粒子在解空間中依照公式（6）（7）來改變其位置與速度；接下來將 K 個分群所得之最佳適應值做比較，由最佳的適應值當做一個疊代的最佳解，隨後粒子繼續跟隨 *pbest* 以及 $g_k best$ 來移動，直到所設定的最大疊代數爲止。

$$V_{Kid}(t) = w \times V_{Kid}(t-1) + c_1 \times Rand(\,) \times (P_{Kid} - X_{Kid}) + c_2 \times Rand(\,) \times (P_{Kgd} - X_{Kid}) \quad (6)$$

$$X_{Kid}(t) = X_{Kid}(t-1) + V_{Kid}(t) \quad (7)$$

（二）方法二：文化分群式粒子群演算法（Culture K-means Particle Swarm Optimization, CKPSO）

本演算法延續上一節所提出之分群式演算法，並加上以文化演算法的知識空間爲基礎所提出之「分群式粒子知識空間」，將不同群粒子在搜尋過程中所得到的知識與經驗，於固定代數時將之保存於知識空間中，如此可以「知識」引導粒子前往具良好解答區搜尋。本研究除了會保留知識空間中表現較優秀的粒子以外，當在達到固定代數時，便以知識空間中表現較優秀的粒子取代分群中表現較差的粒子，以將其它粒子引導至更好的解答區中，其構想如圖 2 所示。

（三）參數設計

1. 測試函數

本研究所提出之分群式 PSO 研究法將對 Sphere 函數（f_1）、Rosenbrock 函數（f_2）、Rastrigrin 函數（f_3）、Griewank 函數（f_4）四個函數做驗證，這些測試函數爲過去學者 [19, 23-25] 最常使用之函數，其優點爲可最測試單目標與多目標函數，同時可以容易看出其最佳解均爲原點或是接近原點。測試函數如式（8）-（11）。

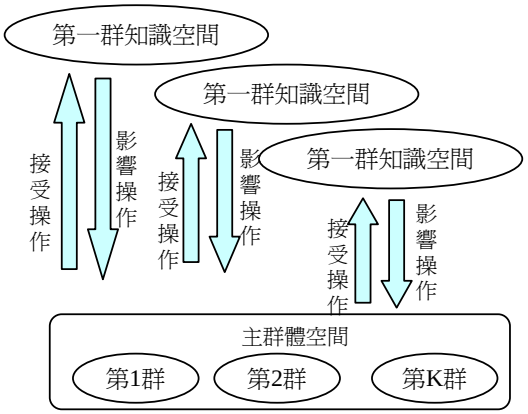

圖 2. 文化分群式粒子群演算法（CKPSO）構想

$$f_1 = \sum_{i=1}^{n} x^2 \qquad (8)$$

$$f_2 = \sum_{i=1}^{n-1} (100(x_{i+1} - x_i^2)^2 + (x_i - 1)^2) \qquad (9)$$

$$f_3 = \sum_{i=1}^{n} (x_i^2 - 10\cos(2\pi x_i) + 10) \qquad (10)$$

$$f_4 = \frac{1}{4000}\sum_{i=1}^{n} (x_i - 100)^2 - \prod_{i=1}^{n} \cos(\frac{x_i - 100}{\sqrt{i}}) + 1 \qquad (11)$$

2. 最大速度 V_{max}

過去的研究中，V_{max} 的設置爲搜尋空間的一半，由於本研究所提出之以 K-means 分群之粒子群演算法，分群後粒子僅跟隨 *pbest* 以及 g_k*best* 移動，切割後粒子的搜尋區域縮小，因此可能需要較小的飛行速度以加強區域搜尋能力。本研究將以實驗法找出適合分群式粒子群演算法的 V_{max}，所需實驗之 V_{max} 值，定爲搜尋空間的 1/2 倍、1/4 倍、1/8 倍、1/16 倍。在 Rastrigin 函數（f_3）中因 1/16 倍的 V_{max} 過小，會使粒子無法移動，因此不予考慮。

3. 其它參數設定

本研究將使用 Angeline [6] 所使用的非對稱式的初始化方式，如表 1。

在線性權重 *w* 與學習因子 C_1 與 C_2 設置方面，由於本研究以標準 PSO 爲主軸發展，因此 *w* 設置由 0.9 遞減至 0.4，學習因子 C_1 與 C_2 設置爲 2，適應值（best fitness）爲所求得測試函數最佳解的最小值，以 f_1 爲例，則爲 min{| f_1|}。

表 1. 測試函數的搜尋空間與初始範圍

函數	搜尋空間	初始範圍
f_1	$-100 \le x_i \le 100$	$50 \le x_i \le 50$
f_2	$-100 \le x_i \le 100$	$15 \le x_i \le 30$
f_3	$-10 \le x_i \le 10$	$2.56 \le x_i \le 5.12$
f_4	$-600 \le x_i \le 600$	$300 \le x_i \le 600$

另外，每一次測試的終止條件則爲達到最大演化代數或當所求得的最佳解在 10^{-4} 以下則停止或進入下一次測試。另外，在效能評估指標（main best fitness）方面，一般可用求解代數、最佳解平均值或 mean square error 等。本文採用與文獻 HPSO [19]、FPSO [23] 中所探討之最佳解平均值的方式，亦即：假設針對測試函數 f_1 測試 n 次，會找到 n 個適應值，求其平均值，作爲其效能評估的指標。最後，本研究針對不同的數學函式，與文獻中所探討之 HPSO [19]、FPSO [23] 相同，分別採用 20、40、80 個粒子，並針對不同的維度來做設定，如表 2 所示。

四、實驗設計

本研究提出的兩種改良後的 PSO 演算法：分群式粒子群演算法（KPSO）與文化分群式粒子群演算法（CKPSO）。分群式粒子群演算法主要以標準 PSO 爲架構，並結合了 K-means 演算法，將初始的粒子分群後，分別跟隨個體最佳適應值記憶 *pbest* 與分群內最佳適應值記憶 g_k*best* 來移動。文化分群式粒子群演算法以分群式粒子群演算法爲主軸，加入了知識空間的概念，以搜尋過程中的經驗來引導分群粒子尋找到更佳的適應值。接下來本研究將實驗所提出之兩種演算法，並與過去學者提出之標準 PSO、HPSO、FPSO 做適應值準度比較。

表 2. 粒子個數與維度、演化代數

粒子數	維度	最大演化代數
20	10	1000
	20	1500
	30	2000
40	10	1000
	20	1500
	30	2000
80	10	1000
	20	1500
	30	2000

（一）實驗一：分群式粒子群演算法（KPSO）

1. KPSO 最大速度 V_{max} 測試

本研究將以實證方式，找出較適用於分群式粒子群演算法的最大速度。最大速度參數如表 3 所示，以搜尋空間的 1/2 倍、1/4 倍、1/8 倍、1/16 倍搜尋速度來做測試，以 K-means 演算法分爲 3 群，對每個測試函數執行 20 次（run=20），結果詳表 4-7。

2. 最大速度 V_{max} 實驗結果分析

KPSO 在 Rosenbrock 函數中，粒子分群搜尋的效果顯而易見，在 10 維的問題當中，當 V_{max} 分別爲 50、25 與 12.5 時，分群式粒子群演算法（KSPO）找到的值已小於標準 PSO 演算法甚多；當粒子數爲 40 與 80 時，KSPO 所搜尋到的平均適應值（mean best fitness, MBF，亦即所求得最佳解的平均值）來說，亦比標準 PSO 來得優秀，且在 V_{max} 爲 12.5 時能找到比標準 PSO 小非常多的適應值。在 Rastrigin 函數中，KPSO 雖不是明顯有效，但也不輸標準 PSO，當 V_{max}=5，維度爲 10，KPSO 所得到的最佳平均解均略勝標準 PSO 一籌。Griewank 函數中，當粒子數目爲 20 時，維度爲 10 與 20 時，KPSO 的適應值優於標準 PSO，僅在維度爲 30 時，KPSO 所搜尋到的平均最佳適應值無法超越標準 PSO。表 8 爲本小節所找出的最佳速度 V_{max}。

表 3. 最大速度測試值

測試函數	V_{max}	V_{max}	V_{max}	V_{max}
f_1	100	50	25	12.5
f_2	100	50	25	12.5
f_3	10	5	2.5	
f_4	600	300	150	75

表 4. KPSO 在 Sphere 函數 V_{max} 測試結果

Popu size	dim	Gene	Mean Best fitness	Mean Best fitness	Mean Best fitness	Mean Best fitness	標準 PSO MBF
			V_{max}=100	V_{max}=50	V_{max}=25	V_{max}=12.5	V_{max}=100
20	10	1000	2.5409E-10	1.33E-10	4.4234E-10	1.2728E-10	0.0000
	20	1500	0.0022	0.0072	0.0006	1.5662E-05	0.0000
	30	2000	19.5204	11.6869	4.1570	0.9369	0.0000
40	10	1000	1.0474E-21	1.2724E-20	2.1187E-21	1.0474E-21	0.0000
	20	1500	1.5986E-08	9.4205E-11	1.8459E-10	1.5730E-10	0.0000
	30	2000	1.2113E-09	9.2127E-11	1.8451E-10	1.7103E-11	0.0000
80	10	1000	9.7916E-26	3.2653E-29	1.5962E-29	2.6078E-30	0.0000
	20	1500	1.3043E-07	1.1102E-08	1.7382E-09	1.9352E-10	0.0000
	30	2000	2.0974E-09	2.5876E-10	8.1292E-13	3.4348E-13	0.0000

表 5. KPSO 在 Rosenbrock 函數 V_{max} 測試結果

Popu size	dim	Gene	Mean Best Fitness	Mean Best Fitness	Mean Best Fitness	Mean Best Fitness	標準 PSO MBF
			V_{max}=100	V_{max}=50	V_{max}=25	V_{max}=12.5	V_{max}=100
20	10	1000	191.9651	24.3296	25.1401	19.0029	96.1715
	20	1500	312.1175	174.1743	101.4575	96.7467	214.6764
	30	2000	655272.4115	105075.8497	4310.7713	634.1855	316.4468
40	10	1000	14.5615	8.0826	4.0246	3.8724	70.2139
	20	1500	67.1390	37.6347	33.6151	27.7528	180.9671
	30	2000	74.1774	68.4878	50.2648	38.4332	299.7061
80	10	1000	11.5364	6.9554	4.9086	4.2982	36.2945
	20	1500	118.6997	30.2394	27.1818	22.5095	87.2802
	30	2000	141.5976	34.4695	35.8396	24.6141	205.5596

表 6. KPSO 在 Rastrigin 函數 V_{max} 測試結果

Popu size	dim	Gene	Mean Best Fitness	Mean Best Fitness	Mean Best Fitness	標準 PSO MBF
			V_{max}=10	V_{max}=5	V_{max}=2	V_{max}=10
20	10	1000	6.7539	5.0529	6.8358	5.5572
	20	1500	24.7154	23.6099	31.1136	22.8892
	30	2000	77.8368	57.3783	65.7125	47.2941
40	10	1000	4.3915	3.5322	4.0086	3.5623
	20	1500	22.4159	16.9705	17.9599	16.3504
	30	2000	47.0128	37.8204	53.5797	38.5250
80	10	1000	2.9354	2.5371	2.5873	2.5379
	20	1500	22.1596	14.1484	18.8576	13.4263
	30	2000	36.3336	35.5704	40.6938	29.3063

表 7. KPSO 在 Griewank 函數 V_{max} 測試結果

Popu size	dim	Gene	Mean Best Fitness	Mean Best Fitness	Mean Best Fitness	Mean Best Fitness	標準 PSO MBF
			V_{max}=600	V_{max}=300	V_{max}=150	V_{max}=75	V_{max}=600
20	10	1000	0.0730	0.0775	0.0686	0.0853	0.0919
	20	1500	0.0654	0.0227	0.0187	0.0242	0.0303
	30	2000	1.5851	0.5190	0.4316	0.3195	0.0182
40	10	1000	0.0638	0.0614	0.0599	0.0561	0.0862
	20	1500	0.0166	0.0101	0.0087	0.0081	0.0286
	30	2000	0.0105	0.0068	0.0047	0.0044	0.0127
80	10	1000	0.0623	0.0647	0.0501	0.0499	0.0760
	20	1500	0.0163	0.0138	0.0145	0.0120	0.0288
	30	2000	0.0059	0.0052	0.0040	0.0034	0.0128

表 8. KPSO 最大速度最適值

測試函數	V_{max}
f_1	12.5
f_2	12.5
f_3	5
f_4	75

3. 分群式粒子群演算法分群數目測試

本小節將延用前小節中所找到的 V_{max}，並進行分群數目的測試。另外，前小節中，Sphere 函數與 Rastrigin 函數在粒子數爲 20 時，可能因粒子分爲三群後，每群粒子個數過少而影響 KPSO 的搜尋效能，因此本小節中，分群數目以分群後每群粒子平均數目以 10 的上下爲限，分群數目如表 9 所示，4 個函數的測試結果詳表 10-13。

4. 分群式粒子群演算法分群數目實驗

Rosenbrock 函數中，整體來說 KPSO 表現優於標準 PSO 許多。由表 11 的數據可看出在問題較不複雜的 10 維時，粒子分爲較多的群數，搜尋效果會比較好，在問題最複雜的 30 維度當粒子數爲 80 時，將粒子分爲 4 群會有較好的表現，若分爲 8 群，會因分群後每群粒子數目較少而效能較低。Rastrigin 函數爲一多峰函數，相較於單峰函數更顯複雜。除了 30 維度的複雜問題外，KPSO 在此表現略勝標準 PSO。Griewank 函數爲一多峰函數，具有無數的區域最佳解。以往的 PSO 演算法及變形演算法，在此函數中皆不易突破，本研究中，粒子數爲 80 時，KPSO 不論爲 2、4 或 8 群表現都優於標準 PSO，且在將粒子分爲 4 群與 8 群時，尤以 30 維度，分爲 8 群時表現最好，其分群搜尋的效果明顯。另外由上述實驗亦可看出當粒子數量夠多時，適當增加分群數量，對於提升演算法在多峰函數的求解效能亦能有正面的幫助。

表 9. KPSO 分群數目

粒子數目	分群數目			
20	2 群	3 群		
40	2 群		4 群	
80	2 群		4 群	8 群

表 10. KPSO 在 Sphere 分群數目實驗結果

Popu size	dim	Gene	Mean Best Fitness	Mean Best Fitness	Mean Best Fitness	Mean Best Fitness	標準 PSO MBF
	V_{max}=12.5		2 群	3 群	4 群	8 群	V_{max}=100
20	10	1000	6.8733E-17	5.4981E-12			0.0000
	20	1500	1.1510E-05	5.1796E-04			0.0000
	30	2000	1.8149E-08	1.7346			0.0000
40	10	1000	1.2545E-26		1.7717E-18		0.0000
	20	1500	5.8294E-13		5.3894E-09		0.0000
	30	2000	8.7016E-14		8.2941E-11		0.0000
80	10	1000	6.73E-31		3.30E-26	3.88E-23	0.0000
	20	1500	7.84E-11		1.70E-09	7.09E-08	0.0000
	30	2000	1.13E-13		1.47E-11	1.38E-09	0.0000

表 11. KPSO 在 Rosenbrock 分群數目實驗結果

Popu size	dim	Gene	Mean Best fitness	Mean Best fitness	Mean Best fitness	Mean Best fitness	標準 PSO MBF
	V_{max}=12.5		2 群	3 群	4 群	8 群	V_{max}=100
20	10	1000	18.4379	18.3613			96.1715
	20	1500	76.1979	97.9085			214.6764
	30	2000	88.1716	469.6004			316.4468
40	10	1000	17.7583		12.8434		70.2139
	20	1500	42.9867		43.7939		180.9671
	30	2000	41.3293		47.3031		299.7061
80	10	1000	5.3499		4.9189	2.8881	36.2945
	20	1500	33.3331		15.5247	16.3792	87.2802
	30	2000	54.9978		31.6140	55.5458	205.5596

表 12. KPSO 在 Rastrigin 分群數目實驗結果

Popu size	dim	Gene	Mean Best fitness	Mean Best fitness	Mean Best fitness	Mean Best fitness	標準 PSO MBF
	V_{max}=5		2 群	3 群	4 群	8 群	V_{max}=10
20	10	1000	5.2420	5.8233			5.5572
	20	1500	22.2884	22.9128			22.8892
	30	2000	50.7647	59.4624			47.2941
40	10	1000	2.9849		3.7651		3.5623
	20	1500	16.2136		19.0573		16.3504
	30	2000	41.5893		44.6793		38.5250
80	10	1000	2.1889		2.5370	2.9851	2.5379
	20	1500	12.0430		12.5764	15.4494	13.4263
	30	2000	28.9489		32.3877	36.1299	29.3063

表 13. KPSO 在 Griewank 分群數目實驗結果

Popu size	dim	Gene	Mean Best fitness	Mean Best fitness	Mean Best fitness	Mean Best fitness	標準 PSO MBF
	V_{max}=75		2 群	3 群	4 群	8 群	V_{max}=600
20	10	1000	0.0811	0.0867			0.0919
	20	1500	0.0149	0.0183			0.0303
	30	2000	0.0171	0.2545			0.0182
40	10	1000	0.0799		0.0641		0.0862
	20	1500	0.0130		0.0072		0.0286
	30	2000	0.0049		0.0047		0.0127
80	10	1000	0.0666		0.0471	0.0466	0.0760
	20	1500	0.0227		0.0071	0.0054	0.0288
	30	2000	0.0080		0.0011	0.0004	0.0128

5. 分群式粒子群演算法實證

在前小節中，我們已得知較適用於 KPSO 的 V_{max} 及分群數目。在本小節中，本研究將以前兩小節中表現最好的 V_{max} 及分群數目來做實證模擬，每個問題各模擬 50 次（run=50），以求得 KPSO 的最佳適應值平均，並與標準 PSO、FPSO、HPSO 做比較。

由表 14~17，分別列出在相同粒子數、維度、迭代數與最大速度情況下，原始 PSO、FPSO 與 HPSO 與 KPSO（KPSO 群數設定爲 2 群）之比較結果。由表 15 可看出 KPSO 在 Rosenbrock 經典優化函數的表現，遠遠優於過去學者所提出的三種方法，而能達到個位數的適應值。無論粒子數目爲 20、40 或是 80，處理問題爲 10 維、20 維、30 維度，KPSO 在此都有良好的表現。推斷是分群的效果，使得以往 PSO 容易落入區域最佳解的弱點，能夠藉由分群後的 $g_k best$ 比較，而得到全域最佳解。Rastrigin 函數中，KPSO 與 FPSO 相較之下其搜尋效能略有輸贏；而混合了基因演算法的 HPSO，在粒子數目爲 20 時，其 100 次的最佳適應值平均是優於本研究所提出的 KPSO。KPSO 在 Griewank 函數中，呈現了分群搜尋的效果。過去在此函數中，因其爲複雜的多峰問題，僅依靠 *pbest* 與 *gbest* 對粒子的吸引不易在此函數上有突破，而 KPSO 在此表現良好，所得之 50 回合最佳適應值平均得以全數打敗標準 PSO 與 FPSO；而在與 HPSO 的比較中，粒子數爲 20 時，KPSO 僅在維度爲 20 時無法打敗 HPSO，在 10 維與 30 維時，均能比 HPSO 找到更好的適應值。

表 14. KPSO 在 Sphere 模擬結果

Test Fun.	Popu size	dim	Gene	最大速度	群數	MBF	原始 PSO	HPSO
f_1	20	10	1000	12.5	2 群	1.0380E-15	0.0000	2.42E-04±2.17E-05
		20	1500		2 群	1.1199E-06	0.0000	0.00212±2.75E-04
		30	2000		2 群	2.3487E-06	0.0000	0.01203±6.33E-04
	40	10	1000	12.5	2 群	2.0429E-26	0.0000	
		20	1500		2 群	1.2197E-15	0.0000	
		30	2000		2 群	4.1150E-13	0.0000	
	80	10	1000	12.5	2 群	4.7594E-31	0.0000	
		20	1500		2 群	4.4449E-11	0.0000	
		30	2000		2 群	1.5539E-14	0.0000	

表 15. KPSO 在 Rosenbrock 分群數目實驗結果

Test Fun.	Popu size	dim	Gene	最大速度	群數	MBF	原始 PSO	FPSO	HPSO
f_2	20	10	1000	12.5	3 群	14.3732	96.1715	66.01409	43.521±16.047
		20	1500		2 群	71.2634	214.6764	108.2865	169.112±21.53
		30	2000		2 群	82.1592	316.4468	183.8037	187.033±22.96
	40	10	1000	12.5	4 群	4.2241	70.2139	48.76523	
		20	1500		2 群	34.2384	180.9671	63.88408	
		30	2000		2 群	56.5729	299.7061	175.0093	
	80	10	1000	12.5	8 群	2.9894	36.2945	15.81645	
		20	1500		4 群	16.9965	87.2802	45.99998	
		30	2000		4 群	34.9967	205.5596	124.4184	

表 16. KPSO 在 Rastrigin 分群數目實驗結果

Test Fun.	Popu size	dim	Gene	最大速度	群數	MBF	原始 PSO	FPSO	HPSO
f_3	20	10	1000	5	2 群	6.2684	5.5572	4.955165	3.0599±0.1535
		20	1500		2 群	20.3607	22.8892	23.27334	11.6590±0.3602
		30	2000		2 群	46.3956	47.2941	48.47555	27.8119±0.8059
	40	10	1000	5	2 群	3.5108	3.5623	3.283368	
		20	1500		2 群	14.1728	16.3504	15.04448	
		30	2000		2 群	38.6850	38.5250	35.20146	
	80	10	1000	5	2 群	1.9502	2.5379	2.328207	
		20	1500		2 群	10.7800	13.4263	10.86099	
		30	2000		2 群	29.2695	29.3063	22.52393	

表 17. KPSO 在 Griewank 分群數目實驗結果

Test Fun.	Popu size	dim	Gene	最大速度	群數	MBF	原始 PSO	FPSO	HPSO
f_4	20	10	1000	75	2 群	0.0750	0.0919	0.091623	0.09078±0.03306
		20	1500		2 群	0.0198	0.0303	0.027275	0.00459±0.3306
		30	2000		2 群	0.0119	0.0182	0.02156	0.09911±0.00106
	40	10	1000	75	4 群	0.0635	0.0862	0.075674	
		20	1500		4 群	0.0132	0.0286	0.031232	
		30	2000		4 群	0.0050	0.0127	0.012198	
	80	10	1000	75	8 群	0.0481	0.0760	0.068323	
		20	1500		8 群	0.0048	0.0288	0.025956	
		30	2000		8 群	0.0015	0.0128	0.014945	

（二）實驗二：文化分群式粒子群演算法（CKPSO）實證

本小節針對第三章差所提出的 CKPSO 加入了「知識空間」的概念，欲藉由知識空間中表現較優秀的粒子來引導其它分群中的粒子，在演化的過程中，每隔 10 代分群中的粒子就將 $g_k best$ 上傳至知識空間中，每一分群的知識空間大小爲其分群母體數的 10%，當達到所設定的 *Influence* 代數時，以知識空間中的粒子取代主群體分群中表現最差的粒子，由粒子在搜尋過程中所產生的「知識」來引導分群後的粒子進入具良好適應值的區域。

1. CKPSO 最大速度 V_{max} 測試實驗分析

文化分群式粒子群演算法（CKPSO）與分群式粒子群演算法（KPSO）相較之下雖無具有絕對優勢，但是 CKPSO 在複雜問題的搜尋能力，在此實驗中則顯著的優於 KPSO。以表 18-21 做一整合分析，可由表中得知搭配不同速度之 KPSO 與 CKPSO 的搜尋能力。

由表 4、表 7 中可看出 Sphere 與 Griewank 函數中，當粒子數爲 20，在 30 個維度的問題處理上，KPSO 會有表現較差的問題，加入了知識空間概念的 CKPSO 則可藉由知識空間的領導，來確保演算法的收斂，如表 18、表 21 所示。

表 18. CKPSO 在 Sphere 函數 V_{max} 測試結果

Popu size	dim	Gene	Mean Best fitness V_{max}=100	Mean Best fitness V_{max}=50	Mean Best fitness V_{max}=25	Mean Best fitness V_{max}=12.5	標準 PSO MBF V_{max}=100
20	10	1000	1.9946E-11	3.0170E-13	2.2377E-13	1.0838E-13	0.0000
	20	1500	1.4487E-04	2.3258E-05	7.5151E-07	2.9120E-06	0.0000
	30	2000	3.8968E-02	1.4500E-02	6.7470E-04	9.8081E-04	0.0000
40	10	1000	1.7582E-21	5.5328E-23	3.3948E-21	2.7742E-21	0.0000
	20	1500	2.1334E-10	5.5268E-11	2.1836E-12	7.2210E-13	0.0000
	30	2000	3.6541E-07	1.7827E-07	2.4459E-08	6.9585E-09	0.0000
80	10	1000	1.8262E-23	2.3087E-24	4.5009E-25	1.5824E-25	0.0000
	20	1500	5.5728E-15	9.8272E-15	2.0240E-15	1.1669E-15	0.0000
	30	2000	3.6154E-10	1.2386E-10	2.0200E-11	1.0915E-11	0.0000

表 19.CKPSO 在 Rosenbrock 函數 V_{max} 測試結果

Popu size	dim	Gene	Mean Best Fitness V_{max}=100	Mean Best Fitness V_{max}=50	Mean Best Fitness V_{max}=25	Mean Best Fitness V_{max}=12.5	標準 PSO MBF V_{max}=100
20	10	1000	33.0891	11.4479	6.3920	16.4536	96.1715
	20	1500	106.0298	90.1269	77.4916	42.3457	214.6764
	30	2000	156.0521	128.9912	136.5957	130.3887	316.4468
40	10	1000	10.4346	4.4533	4.9281	3.2429	70.2139
	20	1500	30.3716	28.0057	27.4756	22.2312	180.9671
	30	2000	34.2607	25.1451	46.6171	36.6389	299.7061
80	10	1000	5.0090	4.6594	3.4197	3.1178	36.2945
	20	1500	20.1164	17.1038	9.5536	14.1915	87.2802
	30	2000	33.6553	26.2295	27.0544	21.6154	205.5596

表 20. CKPSO 在 Rastrigin 函數 V_{max} 測試結果

Popu size	dim	Gene	Mean Best Fitness V_{max}=10	Mean Best Fitness V_{max}=5	Mean Best Fitness V_{max}=2	標準 PSO MBF V_{max}=10
20	10	1000	6.0505	6.5016	6.9613	5.5572
	20	1500	37.8303	21.9978	24.3735	22.8892
	30	2000	39.7365	58.7197	61.8167	47.2941
40	10	1000	5.5220	3.1291	5.0793	3.5623
	20	1500	24.8739	17.1879	23.4171	16.3504
	30	2000	45.3714	36.4747	42.8106	38.5250
80	10	1000	3.5819	2.0844	3.2851	2.5379
	20	1500	19.1032	13.5686	17.4118	13.4263
	30	2000	36.4132	27.0672	38.4054	29.3063

在大部份的情況下，CKPSO 的效能是優於 KPSO，因此 CKPSO 改善了 KPSO 在高維度問題可能會有表現較差的問題。

2. 文化分群式粒子群演算法實證

本小節將以實驗法找出之 CKPSO 最適最大速度與分群數目，對四個測試函數不同粒子數目與不同維度的問題各模擬 50 次，以求得 CKPSO 的最佳適應值平均，並與標準 PSO、FPSO、HPSO 做比較，詳表 22-25。

表 21. CKPSO 在 Griewank 函數 V_{max} 測試結果

Popu size	dim	Gene	Mean Best Fitness V_{max}=600	Mean Best Fitness V_{max}=300	Mean Best Fitness V_{max}=150	Mean Best Fitness V_{max}=75	標準 PSO MBF V_{max}=600
20	10	1000	0.0949	0.0682	0.0905	0.0879	0.0919
	20	1500	0.0328	0.0212	0.0166	0.0222	0.0303
	30	2000	0.0799	0.0578	0.0242	0.0203	0.0182
40	10	1000	0.0710	0.0812	0.0686	0.0761	0.0862
	20	1500	0.0157	0.0124	0.0108	0.0138	0.0286
	30	2000	0.0097	0.0056	0.0069	0.0044	0.0127
80	10	1000	0.0618	0.0546	0.0665	0.0668	0.0760
	20	1500	0.0132	0.0114	0.0108	0.0165	0.0288
	30	2000	0.0025	0.0036	0.0026	0.0037	0.0128

（三）實驗綜合分析

1. Sphere 函數（詳圖 3-5）

由得知當粒子數爲 20 時，CKPSO 的搜尋效果優於 HPSO。另外，與標準 PSO 之比較，因文獻中無法得知確實數字爲何，僅以 0.00001 視之，由數據顯示，CKPSO 的搜尋能力應是不會輸給標準 PSO。當粒子數爲 80 時，維度爲

表 22. CKPSO 在 Sphere 函數模擬結果

Test Fun.	Popu size	dim	Gene	最大速度	群數	MBF	原始 PSO	HPSO
f_1	20	10	1000	12.5	2 群	3.1055E-12	0.0000	2.42E-04±2.17E-05
		20	1500		2 群	9.9716E-06	0.0000	0.00212±2.75E-04
		30	2000		2 群	3.1827E-04	0.0000	0.01203±6.33E-04
	40	10	1000	12.5	2 群	1.2214E-05	0.0000	2.42E-04±2.17E-05
		20	1500		2 群	1.0994E-11	0.0000	0.00212±2.75E-04
		30	2000		2 群	1.1174E-06	0.0000	0.01203±6.33E-04
	80	10	1000	12.5	2 群	1.1174E-20	0.0000	2.42E-04±2.17E-05
		20	1500		2 群	5.8635E-18	0.0000	0.00212±2.75E-04
		30	2000		2 群	2.7866E-04	0.0000	0.01203±6.33E-04

表 23. CKPSO 在 Rosenbrock 函數模擬結果

Test Fun.	Popu size	dim	Gene	最大速度	群數	MBF	原始 PSO	FPSO	HPSO
f_2	20	10	1000	12.5	3 群	16.1275	96.1715	66.01409	43.521±16.047
		20	1500		3 群	53.7278	214.6764	108.2865	169.112±21.53
		30	2000		2 群	78.9310	316.4468	183.8037	187.033±22.96
	40	10	1000	12.5	4 群	5.4852	70.2139	48.76523	
		20	1500		2 群	19.6719	180.9671	63.88408	
		30	2000		2 群	50.6539	299.7061	175.0093	
	80	10	1000	12.5	8 群	3.0401	36.2945	15.81645	
		20	1500		8 群	9.6028	87.2802	45.99998	
		30	2000		4 群	39.4583	205.5596	124.4184	

表 24. CKPSO 在 Rastrigin 函數模擬結果

Test Fun.	Popu size	dim	Gene	最大速度	群數	MBF	原始 PSO	FPSO	HPSO
f_3	20	10	1000	5	2 群	6.7605	5.5572	4.955165	3.0599±0.1535
		20	1500		2 群	21.2332	22.8892	23.27334	11.6590±0.3602
		30	2000		2 群	50.4357	47.2941	48.47555	27.8119±0.8059
	40	10	1000	5	2 群	3.1539	3.5623	3.283368	
		20	1500		2 群	17.1229	16.3504	15.04448	
		30	2000		2 群	37.3144	38.5250	35.20146	
	80	10	1000	5	2 群	2.1441	2.5379	2.328207	
		20	1500		2 群	17.7899	13.4263	10.86099	
		30	2000		2 群	29.1077	29.3063	22.52393	

表 25. CKPSO 在 Griewank 函數模擬結果

Test Fun.	Popu size	dim	Gene	最大速度	群數	MBF	原始 PSO	FPSO	HPSO
f_4	20	10	1000	75	3 群	0.0733	0.0919	0.091623	0.09078±0.03306
		20	1500		2 群	0.0203	0.0303	0.027275	0.00459±0.3306
		30	2000		2 群	0.0134	0.0182	0.02156	0.09911±0.00106
	40	10	1000	75	4 群	0.0640	0.0862	0.075674	
		20	1500		4 群	0.0115	0.0286	0.031232	
		30	2000		4 群	0.0084	0.0127	0.012198	
	80	10	1000	75	8 群	0.0456	0.0760	0.068323	
		20	1500		8 群	0.0040	0.0288	0.025956	
		30	2000		8 群	0.0030	0.0128	0.014945	

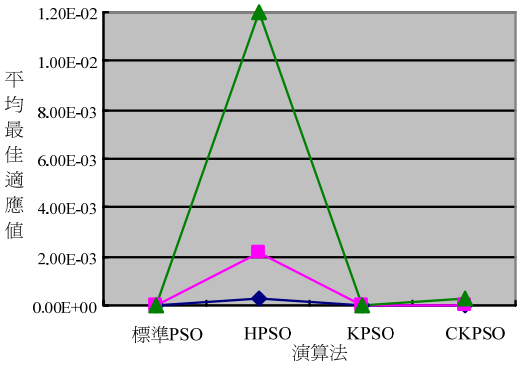

圖 3. Sphere 函數適應值比較圖（粒子數 20）

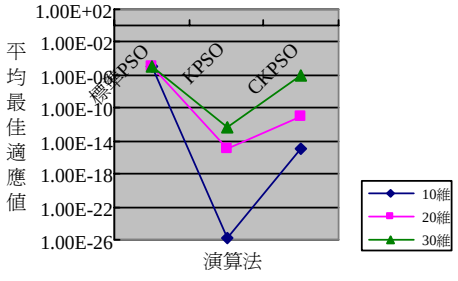

圖 4. Sphere 函數適應值比較圖（粒子數 40）

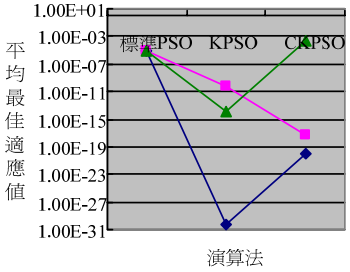

圖 5. Sphere 函數適應值比較圖（粒子數 80）

10 與 20，CKPSO 能找到比 0.00001 更好的適應值，僅在 30 維時，略輸給標準 PSO。

2. Rosenbrock 函數（詳圖 6~8）

KPSO 與 CKPSO 在 Rosenbrock 函數中的效果最爲明顯，且皆打敗過去學者所提出的粒子群改良演算法，分群的效果使得粒子得以將空間中的區域劃分，其中以 CKPSO 具有較好的求解能力與穩定性。

3. Rastrigin 函數（詳圖 9~11）

KPSO 由中看來比 CKPSO 可以找到稍好的解，但是 KPSO 有在高維度可能表現較差的問題。CKPSO 雖未能比 KPSO 找到更好的最佳適應值平均，但是演算法相對穩定，由知識空間引導演算法收斂。而與其它演算法相較之下，CKPSO 在 Rastrigin 函數表現較不突出。

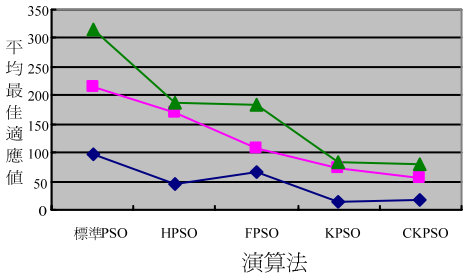

圖 6. Rosenbrock 函數適應值比較圖（粒子數 20）

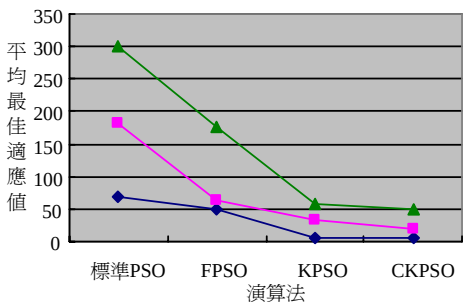

圖 7. Rosenbrock 函數適應值比較圖（粒子數 40）

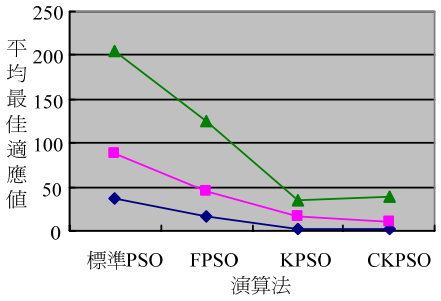

圖 8. Rosenbrock 函數適應值比較圖（粒子數 80）

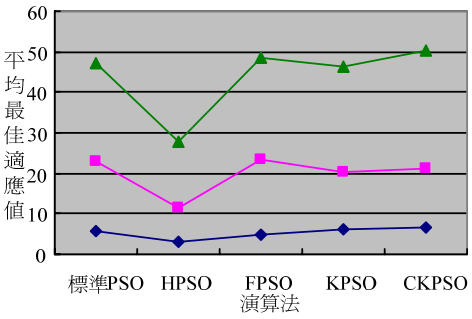

圖 9. Rastrigrin 函數適應值比較圖（粒子數 20）

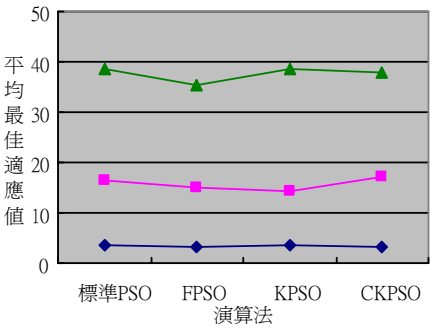

圖 10. Rastrigrin 函數適應值比較圖（粒子數 40）

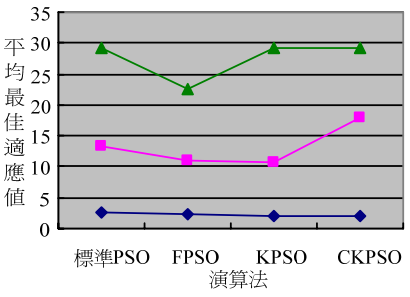

圖 11. Rastrigrin 函數適應值比較圖（粒子數 80）

4. Griewank 函數（詳圖 12~14）

當粒子數爲 20 時，以 Lovbjerg 所提出的 HPSO 效果最好，但 KPSO 與 CKPSO 在此皆能打敗標準 PSO、FPSO。而 KPSO 與 CKPSO 相比則略有輸贏。

五、結論與建議

本研究提出兩種改良自標準 PSO 的演算法：分群式粒子群演算法（KPSO）與文化分群式粒子群演算法（CKPSO）。KPSO 使用 K-means 演算法將粒子做分群以劃

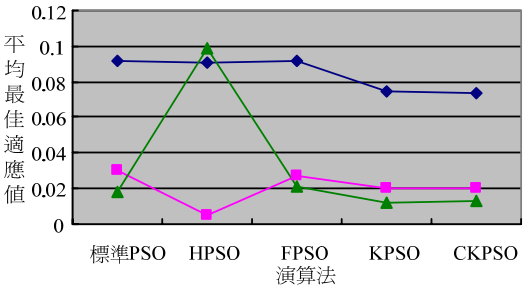

圖 12. Griewank 函數適應值比較圖（粒子數 20）

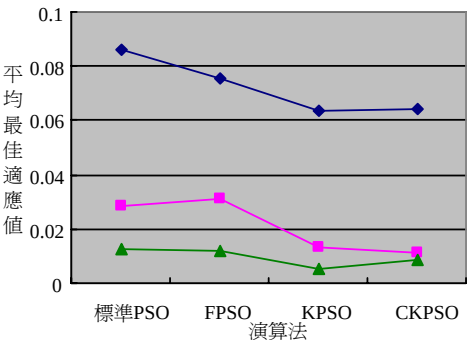

圖 13. Griewank 函數適應值比較圖（粒子數 40）

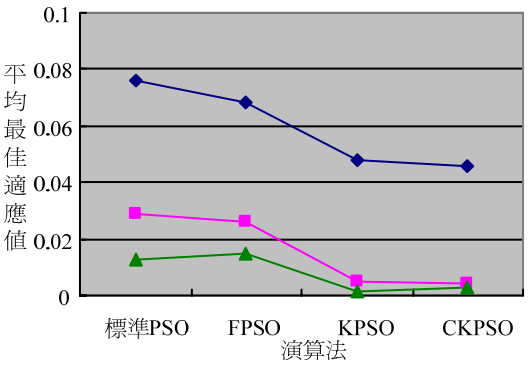

圖 14. Griewank 函數適應值比較圖（粒子數 80）

分搜尋領域，並以實驗的方式找到最適合 KPSO 使用的最大速度 V_{max} 以加強區域搜尋的能力，其分群搜尋的效果使得平均最佳適應值的準度得以提升。雖然 KPSO 在粒子數目較少，而需處理的問題爲較複雜的高維度問題時，會有無法收斂的可能性，但是由前述的分群實驗數據亦可看出當粒子數量夠多時，適當增加分群數量，對於提升演算法在多峰函數的求解效能亦能有正面的幫助。因此，本研究將文化演算法中「知識空間」的概念帶入 KPSO 中，利用知識空間中的粒子來引導主群體中的粒子前往具良好解答區搜尋，此爲 CKPSO；其搜尋效能與 KPSO 相較之下雖無絕對優勢，但 CKPSO 有足夠能力處理複雜的高維問題，較 KPSO 更能確保演算法的收斂。

在四個測試函數中，KPSO 與 CKPSO 也有不錯的表現。KPSO 除了在高維問題中有無法收斂的可能性外，在其它較簡單的問題中則有良好表現；KPSO 在 Sphere 函數、Rosenbrock 函數、Griewank 函數中，極大部份能夠打敗標準 PSO、HPSO、FPSO，僅在 Rastrigin 函數中表現稍顯爲弱。CKPSO 彌補了 KPSO 在高維問題中可能無法有效收斂的弱點，演算法相形穩定，在 Sphere 函數、Rosenbrock 函數、Griewank 函數裡，藉由知識空間中的「知識」引導，使得分群後的粒子能更有效的搜尋，僅在 Rastrigin 函數中表現稍顯微弱，但其與標準 PSO 相較之下仍略有輸贏。

文化分群式粒子群演算法（CKPSO）爲結合了 K-means 分群演算法、文化演算法、及粒子群演算法之改良演算法，分群後的粒子以較小的 V_{max} 飛行搜尋區域最佳解，再經比較分群最佳適應值記憶 $g_k best$ 得到一個疊代內的全域最佳解，另外藉由知識空間的引導，可使粒子移動到良好解答區中搜尋，可確保演算法的收斂，經由實驗證明，爲一個穩定且效能良好的改良式粒子群演算法。

參考文獻

1. 李愛國（民 93），多粒子群協同優化算法，复旦學報，43(1)，923-925。
2. 胡曉輝（民 91），粒子群優化算法介紹，http://icdweb.cc.purdue.edu/~hux。
3. 郭信川、張建仁、劉仁清（民 93），粒子群演算法於最佳化問題之研究，第一屆台灣作業研究學會學術研討會暨 2004 年科技與管理學術研討會，台北。
4. 黃承龍、王良吉、董吉雄（民 94），粒子群最佳化演算法之文獻回顧與研究議題分析，2005 數位內容管理與應用研討會，高雄。
5. 謝曉鋒、張文俊、楊之廉（民 92），微粒群算法綜述，控制與決策，18(5)，129-134。
6. Angeline, P. J. (1998) Evolutionary optimization versus particle swarm optimization: Philosophy and performance differences. Proceedings of the 7th International Conference on Evolutionary Programming VII, San Diego, CA.
7. Boyd, R. and P. J. Richerson (1998) *Culture and Evolutionary Process*, University Of Chicago Press, Chicago.
8. Clerc, M. (1999) The swarm and the queen: towards a deterministic and adaptive particle swarm optimization. Proceedings of the IEEE Congress on Evolutionary Computation, Seoul, Korea.
9. Dorigo, M., V. Maniezzo and A. Colorni (1996) The ant system: Optimization by a colony of cooperating agents. *IEEE Transactions on Systems and Cybernetics - Part* B, 26(1), 29-41.
10. Eberhart, R. C. and Y. Shi (2001) Particle swarm optimization: Developments, application and resources. Proceedings of the 2001 Congress on Evolutionary Computation, Seoul, South Korea.
11. Eberhart, R. C. and J. Kennedy (1995) A new optimizer using particle swarm theory. Proceedings Sixth International Symposium on Micro Machine and Human Science, Nagoya, Japan.
12. Eberhart, R. C. and Y. Shi (1998) Comparison between genetic algorithms and particle swarm optimization. 1998 Annual Conference on Evolutionary Programming, San Diego, CA.
13. Fogel, D and H. G. Beyer (1996) A note on the empirical evaluation of intermediate recombination. Proceedings of the Fifth Annual Conference on Evolutionary Programming V, San Diego, CA.
14. Hu, X., Y. Shi and R. C. Eberhart (2004) Recent advances in particle swarm. Congress on IEEE Evolutionary Computation, Portland, Oregon.
15. Jin, X. and R. G. Reynolds (2000) Using knowledge-based system with hierarchical architecture to guide the search of evolutionary computation. *International Journal on Artificial Intelligence Tools*, 9(1), 27-44.

16. Kennedy, J. and R. C. Eberhart (1995) Particle swarm optimization. Proceedings IEEE International Conference on Neural Networks, Piscataway, NJ.
17. Kennedy, J., R. C. Eberhart and Y. Shi (2001) *Swarm Intelligence*, 287-324. The Morgan Kaufmann Series in Artificial Intelligence, San Francisco, CA.
18. Lloyd, S. (1982) Least squares quantization in PCM. *IEEE Transactions on Information Theory*, 28(2), 129-137.
19. Lovbjerg, M., T. K. Rasmussen and T. Krink (2001) Hybrid particle swarm optimizer with breeding and subpopulations. Proceedings of Genetic and Evolutionary Computation Conference, San Francisco, CA.
20. PSO Tutorial, Retrieved April 16, 2006, from: http://www.swarmintelligence.org/index.php.
21. Reynolds, C. W. (1987) Flock, herds, and schools: A distributed behavioral model. *Computer Graphics*, 21(4), 25-34.
22. Reynolds, R. G. and W. Sverdlik (1994) Problem solving using cultural algorithms. Proceedings of the First IEEE Conference on Evolutionary Computation, Orlando, FL.
23. Shi, Y. and R. C. Eberhart (1998a) Parameter selection in particle swarm optimization. Proceedings of the 7th International Conference on Evolutionary Programming VII, San Diego, California, CA.
24. Shi, Y. and R. C. Eberhart (1998b) A modified particle swarm optimizer. Proceedings of the 1998 IEEE International Conference on Evolutionary Computation, Piscataway, NJ.
25. Shi, Y. and R. C. Eberhart (1999) Empirical study of particle swarm optimization. Proceedings of the IEEE Congress on Evolutionary Computation (CEC), Washington, DC.
26. Shi, Y. and R. C. Eberhart (2001) Fuzzy adaptive particle swarm optimization. Proceeding of the 2001 Congress on Evolutionary Computation, Piscataway, NJ.
27. Srinivasan, D., Loo, W. H. and R. L. Cheu (2003) Traffic incident detection using particle swarm optimization. Proceedings of the 2003 IEEE Swarm Intelligence Symposium, Indianapolis, IN.

收件：96.03.19　修正：96.12.04　接受：97.03.10