粒子群演算法與基因演算法之比較-以派課問題為例

游峰碩
崑山科技大學資訊管理系
助理教授
fsyu@mail.ksu.edu.tw

王子夏
崑山科技大學資訊管理系
研究生
u9011023@mail.ndhu.edu.tw

吳昀珊
崑山科技大學資訊管理系
研究生
S096002353@stumail.ksu.edu.tw

摘要

粒子群演算法(Particle Swarm Optimization, PSO)模擬生物之群體行為進行分群，是近年來廣為使用之最佳化搜尋方法，其產生群體與計算適應值的步驟與基因演算法(Genetic Algorithm, GA)相當類似，但兩者後期演算步驟與執行效能皆不盡相同。本研究期望透過大專院校之派課問題，比較粒子群演算法與基因演算法於相同的執行成效之下其執行成本的差異性。

關鍵詞：粒子群演算法、基因演算法、派課問題。

Abstract

Particle Swarm Optimization (PSO) is a popular optimize search method that inspire by the swarming of biological populations in recent years. PSO is similar to the Genetic Algorithm (GA) in the sense that both are population-based search method. Since the two methods are supposed to find a solution to given objective function but employ different procedure and computational effort, it's appropriate to compare their implementation. The disadvantage of the GA is its expensive computational cost. This paper attempts to examine that PSO has the same efficiency as the GA but with better computational efficiency. The problem area of the PSO and GA chosen is faculty-course assignment problem (as used in the university).

Keywords: particle swarm optimization, genetic algorithm, course assignment.

1. 前言

近年來各類型最佳化演算法受到的關注日益增加，許多學者利用諸如粒子群演算法(PSO)、基因演算法(GA)與蟻群演算法(Ant Colony Optimization，ACO)等工具用以解決複雜的最佳化問題，其中基因演算法模擬達爾文「物競天擇」的觀念，藉由遺傳運算元(genetic operators)幫助染色體進化，以搜尋問題最佳解;而粒子群演算法模擬生物之群體行為進行分群，並透過粒子間的經驗分享驅使粒子移動，是近十年來廣為使用之最佳化搜尋方法，其產生群體與計算適應值的步驟與基因演算法相當類似，但是兩者後期的演算步驟與執行效能皆不盡相同。因此，本研究希望藉由大專院校派課問題之探討，比較兩著執行成效與所耗費之執行成本的差異性。

本研究所探討之派課問題是指一個科系如何將下學期所欲開設之課程分派給系上之授課老師，其屬於排程問題(Scheduling Problem)的一種，但過去部分研究將其歸於排課問題(Timetabling Problem)之中，少有文獻單一探討此問題。Hultberg[6]從古典的運輸問題 (transportation problem) 觀點來研究教師的派課問題，其對於派課問題只考量如何將指派給教師之不同科目的數量最小化，並且藉證明派課問題是一個強 NP-困難(NP-hard in the strong sense)的問題。文中亦給出了一個使用分支限界(branch-and-bound)的演算法來求解其派課模型。而有些學者從教師以及管理者兩個不同使用者的角度來考量，Ozdemir 等學者[7]使用 0-1 非線性多目標規劃(0-1 nonlinear multiobjective programming)求解派課問題的最佳化，並且採用層級分析法(Analytic Hierarchy Process, AHP)來綜合簡化問題中之多目標函數成為單一函數，有效地求得其問題的最佳解。Parthiban 等學者[10]也採用層級分析法並且提出一個以喜好為基礎之教師派課決策模型。本研究[1][2]過往曾採用基因演算法建構大專院校之派課系統，並可依照系所之需求調整權重值以產生合適的派課結果。

以下各章節內容摘要分述如下。第二節對於基因演算法作相關的文獻探討。第三節對於粒子群演算法作相關的文獻探討。第四節描述本研究之派課問題並且建立所需要之數學模型。第五節對本研究之實驗結果做一歸納與結論。

2. 基因演算法

基因演算法(Genetic Algorithm, GA)由美國密西根大學教授 John H. Holland[5]及其同事、學生於 70 年代左右通過對生物進化過程進行模擬所得出的一種全新的機率最佳化方法，其概念與達爾文進化論的「物競天擇」觀念類似。生物的進化過程本身就是一個關於環境適應性的搜尋過程。因此，基因演算法是一種模擬生物界自然選擇與進化機制所發展出之高度平行、隨機、自適應的隨機搜尋法。基因演算法透過演化的機制隨機地逐步淘汰較差的個體，保留較好的個體，而被保留的個體使用交配(crossover)，突變(mutation)等遺傳運算元來產生優良的下一代，經過世代演化機制達到最佳化的目標。

使用基因演算法，首先必須對問題的解進行編碼成為所謂的「染色體」；常見的編碼方式為採用二位元編碼或是文字編碼二種方式，一個染色體稱為個體(individual)，一個個體代表所探討問題的一個解，而一個染色體由許多個基因(gene)所組成，不同的基因排列代表著此一染色體的不同特徵。本研究所要採用之編碼方式定義如下:對於派課結果之編碼型態，粒子/染色體 X 代表一派課結果，X 由 n 個子粒子/染色體所組成，其中 n 為這學期所決定開設的課程數量，每個課程包含 m 個 0 或 1 的值，其中 m 為教師數量，所以每個子粒子/染色體代表各個課程所排定的派課結果。粒子/染色體編碼如圖 1 所示。

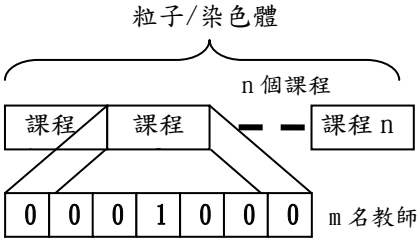

圖 1 粒子/染色體編碼示意圖

令 $C=\{C_1,C_2,\ldots,C_n\}$ 表示所有課程所成之集合，$T=\{T_1,T_2,\ldots T_m\}$ 表示所有教師所成之集合。定義決策變數 x_{ij} 如下：

x_{ij}=1 假如教師 T_j 被指派到課程 C_i；

x_{ij}=0 假如教師 T_j 沒有被指派到課程 C_i； $1\le j\le m$, $1\le i\le n$。

所以對於派課之結果本研究可以用一個矩陣 $X_{n\times m}$ 來表示一個粒子/染色體編碼方式。另外，我們將前置作業所收集之「教師授課喜好表」轉換成喜好度矩陣 $F_{n\times m}$。喜好度矩陣中之每一列(row)代表所有教師對於該課程之喜好度，每一行(column)代表一名教師對於所有課程之喜好度。

每一個染色體會經過評估過程來計算該染色體對所處環境的適應度，適應度越高的染色體其相對映問題解的品質越好。在進化的過程中，整體族群(population)一般會透過遺傳運算元進行最佳化搜尋，為演算法主要的三個運算單位，包含選擇(selection)、交配(crossover)、突變(mutation)等三大部分，其基本的運作流程如圖 2 所示。

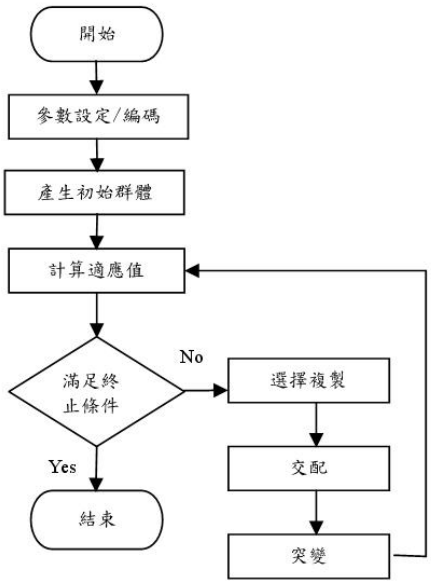

圖 2 基因演算法運作流程

選擇運算元主要在於模擬自然界適者生存的型態，適應度函數越高越能存活至下一演化世代，本研究採用輪盤法則(roulette wheel selection)來決定哪些染色體將會存活至下一代；並且配合菁英保留法(elitist selection)的原則來確保一個世代中最好的一個染色體會存活至下一代。

交配運算元採用單點交配法，但是作法異於傳統單點交配法於基因中隨機選取一個位置做為切割點。我們採用隨機的方式來決定切割點的選取，而切割點的位置在課程與課程的交界處，以避免需要額外的基因修復操作來校訂不合條件的基因，造成演算效率的降低。交配前與交配後染色體的情況如圖 3 所示。

圖 3 交配示意圖

突變運算元的目的在於避免演算法太早收斂造成早熟的現象，這也是避免演算法落入局部最佳解所必須採行之步驟。本研究採用隨機方式選定於染色體中之任意基因位置做為突變的位置。對於選定之位置再隨機決定是否應該突變，如果是則將該基因位置之值做 0，1 的變換，如圖 4 所示。

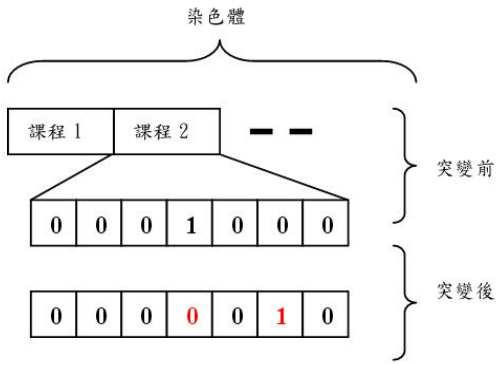

圖 4 突變示意圖

3. 粒子群演算法

粒子群演算法由 Kennedy 和 Eberhart[8]於 1995 年所提出，其主要概念源自於鳥類與魚群之群體行為的觀察。在其最佳化之過程中，每單一粒子會根據其本身的經驗以及整個粒子群之經驗作移動，使粒子能朝全域最佳解之路徑移動，意指每個粒子皆知道群體中哪個粒子離最佳解最近，也知道該粒子之位置，而粒子群之群體行為會驅使所有粒子跟隨離最佳解最近的粒子作移動。粒子群演算法初期隨機產生群體與計算適應值的步驟與基因演算法相當類似，但是兩者不同之處在於粒子群演算法後期會利用粒子間的群體行為與競合關係取代基因演算法的遺傳運算元(genetic operators) [11]。粒子群演算法之運作流程如圖 5 所示。

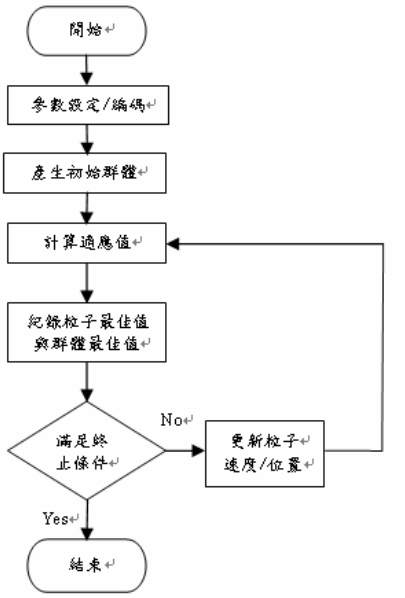

圖 5 粒子群演算法運作流程

由於許多的最佳化問題皆為離散型空間，如部份非連續解的排程問題等，而傳統的粒子群演算法只能針對問題解為連續實數的研究範疇，於是 Kennedy 和 Eberhart[7]於 1997 年提出離散式粒子群演算法(Discrete Particle Swarm Optimization, DPSO)，用以解決非連續性的最佳化問題，證實離散式粒子群演算法於 De Jong[4]之五項最佳化測試方程式中獲得良好的成效。過往亦有數位學者針對此一演算法進行改善，Yang 等學者[11]利用不同的方法更新速率，而提升演算法之執行效率。Al-kazemi 和 Mohan[3]在速率更新部份使用個體最佳值與群體最佳值擇一的方式，代替原本兩者同時使用之方式。

粒子群演算法主要透過粒子間彼此分享經驗，達到更新粒子位置的目的，每個粒子會保存各自目前最好的適應值與位置，稱為粒子最佳值(Particle best value, Pbest);以及目前群體中最好的適應值與位置，稱之為群體最佳值(Globe best value, Gbest)。每個粒子會根據粒子最佳值與群體最佳值更新粒子速度，其粒子移動與速度更新公式如下:

$$V_{ij}^{t+1} = W \times V_{ij}^{t} + c_1 \times r_1 \times (Gbest_{ij} - X_{ij}^{t}) + c_2 \times r_2 \times (Pbest_{ij} - X_{ij}^{t}) \quad (1)$$

$$X_{ij}^{t+1} = X_{ij}^{t} + V_{ij}^{t+1} \quad (2)$$

其中 V_{ij} 為粒子速度，r_1 及 r_2 為介於 0 與 1 之間的隨機變數，c_1 及 c_2 為學習因子，W 為權重值，本研究採用 Shi 和 Eberhart[13]所提出之線性慣量遞減權重(Linear inertia Reduction)，其公式如下:

$$W = W_{\max} - \frac{W_{\max} - W_{\min}}{iter_{\max}} \times iter \quad (3)$$

其中 $W_{\max}$ 為權重最大值，$W_{\min}$ 為權重最小值，$iter$ 為演算法執行代數，而 $iter_{\max}$ 為執行代數最大值。利用線性遞減慣性權重的概念，可使得演算法執行初期有較高的慣性權重值，藉以保持全域搜尋的靈活性，並且隨著執行過程逐漸降低慣性權重值，進而轉入局部搜尋之動作，以加強粒子之局部搜尋能力。

由於粒子群演算法只能針對問題解為連續實數的研究範疇，而傳統離散式粒子群演算法又不適合本研究之粒子編碼，因此對於粒子移動公式本研究將公式(2)替換為公式(4)，其公式如下:

$$\begin{aligned} &if \quad j = \arg\max\ V_{ij}(j = 1,2....m) \quad then\ \ X_{ij}^{t+1} = 1; \\ &\qquad\qquad\qquad\qquad\qquad\quad else\ \ X_{ij}^{t+1} = 0; \end{aligned} \quad (4)$$

根據上式利用此一公式可確保粒子移動後不會違反硬性限制，破壞演算法之粒子結構。

4. 問題描述

本研究所探討的派課問題是指一個科系如何將下學期所欲開設之課程分派給系上之授課老師。派課問題必須考慮到諸如教師喜好，教師專長，行政職務等因素，各因素有其輕重，因此，如同排課問題一般，我們將限制式區分為兩大類來考慮，分別為：硬性限制(Hard Constraint, HC)以及軟性限制(Soft Constraint, SC)。所謂的硬性限制是指因為資源限制，學校規定或班級課程不能衝堂之性質，所必須要滿足的派課條件，是不可避免的條件。而所謂的軟性限制則是問題中必須儘量去滿足的條件。大部份的軟性限制都是人為因素的考量。

4.1. 軟性與硬性限制

本研究所規範之硬性限制與軟性限制列舉如下：

硬性限制:

- HC1　同一門課程只能有一名教師被指派。
- HC2　符合系所之學分數限制
- HC3　所有班級的必修課程必須有指派教師。

軟性限制:

- SC1　被指派之課程的領域必須盡量與教師之專長領域相符合。
- SC2　同一教師被指派之學分數差異度越小越好。
- SC3　同一教師被指派之相異課程數目越少越好。
- SC4　教師的授課偏好應該儘量被滿足。

其中 SC1 軟性限制是為了達到適才適所之目的；而 SC2 軟性限制之設定主要是因為多數的教師不喜歡所被指派的課程太多，此軟性限制可以降低教師的負擔，進而給學生較好的學習品質；而 SC3 軟性限制主要目的在於期望相同課目最好由同一教師擔任。

對於所探討之派課問題所牽涉之軟性限制，我們制訂並且事先收集所需之資料以利將來演算法之運算。這些資料包含「教師授課喜好表」、「課程領域分析表」、「教師專長分析表」等三項表格，分述如下：

「教師授課喜好表」：為一份包含該學期所有課程之課程表，對於一個學期所要開設之課程，每位教師對於課程提供其喜好度，喜好度等級區分為 10 等，其中 10 代表最滿意，1 代表最不滿意。每位教師需提供 10 個科目的喜好度於此表格上，如表 1 所示。

表 1 教師授課喜好表

喜好度 / 科目名稱	教師一	教師二	教師三	…
課程一	5		7	
課程二	4			
課程三		8	2	
課程四	10	3		

「課程領域分析表」：對於全系之課程結構做一綜合性分析並且建構課程之領域結構圖。一個課程可能只屬於某特定領域，但也有可能隸屬於數個以上的領域，表 2 為一課程領域範例，可由表得知課程 1 屬於領域 C，課程

2 屬於領域 A 與 B，而課程 3 則屬於領域 A 與 D。

表 2 課程領域分析表

課程 / 領域	課程一	課程二	課程三	…
領域 A		v	v	
領域 B		v		
領域 C	v			
領域 D			v	

「教師專長分析表」：這是一份由教師所填寫之專長分析表，用以顯示其所屬之專長領域，其目的在於執行派課演算法時做為 SC1 計算之用，如表 3 所示。

表 3 教師授課喜好表

領域 / 教師	領域 A	領域 B	領域 C	領域 D
教師 1	v		v	v
教師 1		v	v	
教師 3		v	v	v
教師 4	v			v

4.2. 適應度函數(fitness function)

為評估粒子/染色體的好壞，本研究訂定一適應度函數，透過適應函數的計算，當粒子/染色體的適應值越佳，表示其所代表的解越能滿足問題的要求。本研究之適應度函數由「派課總體喜好度(Total Average Preference, TAP) 」以及「成本函數(Cost Function)」兩大部分所組合而成，令 X 表示一個粒子/染色體，則其適應度函數定義如下：

$$fitness(X)=\frac{TAP(X)}{1+\text{cost(X)}} \tag{5}$$

由上之公式可以看出，一個相當符合絕大多數教師喜好之派課結果所計算出之適應值會趨近於 1。這是因為對於該粒子/染色體 X ，我們會有 $TAP(X)\to 1$ 以及 $\text{cost}(X)\to 0$。下列小節分述各組成部分的定義。

4.3. 派課總體喜好度(TAP)

「派課總體喜好度」代表全體教師對某一派課結果的平均喜好值。TAP 之計算由派課結果與「教師授課喜好表」矩陣 $F_{n\times m}$ 之轉置相乘而得。令 $I=\{1,\ldots,n\}$ 代表課程數目，$J=\{1,\ldots,m\}$ 代表教師數目，$P=\{P_1,\ldots P_n\}$ 代表喜好值所成的集合，n 代表課程總數，則 TAP 定義如下：

$$TAP(X)=\frac{\sum_{i\in I}\sum_{j\in J}x_{ij}p_{ji}}{\max(P)\cdot n}=\frac{tr(X_{n\times m}F^{T}_{n\times m})}{\max(P)\cdot n} \tag{6}$$

其中 $F^{T}_{n\times m}$ 表示喜好度矩陣 $F_{n\times m}$ 的轉置，$tr(A)$ 表示矩陣 A 的 *trace*，$\max(P)$ 代表集合 P 中之最大值。根據此定義，一個滿意的派課結果(指派給教師之課程皆滿足其最大喜好度) 其派課整體喜好度之值將會趨近於 1。

4.4. 成本函數(Cost Function)

成本函數的責任在於決定一個粒子/染色體的好與壞，讓演算法可以根據此資訊接續著相關操作。對於一個粒子/染色體 X ，定義其 Cost 函數如下：

$$\text{Cost}(X)=\sum_{i=1}^{3}w_i\times d_i(X) \tag{7}$$

其中 w_i 為 $d_i(X)$ 之權重值。採用動態權重值可以幫助系統根據不同的需求來產生合適之課表。

在 Cost 函數中，$d_1(X)$ 代表違反 SC1 之 cost，SC_1 表示被指派之課程的領域必須盡量與教師之專長領域相符合。為了讓派課結果儘量符合適才適所的目的，達到最大的專長符合度 (也就是說不是因為選了就派，而是要考量到是否該教師有該專長領域)。$d_1(X)$ 之設計考量主要是對於派課之結果計算出是否有將適任之教師指派到相關課程。此計算值我們期望它越小越好，才能符合適才適所的目標。

$d_2(X)$ 代表違反 SC2 之 cost，SC2 表示教師被指派之學分數差異度越小越好。亦即是希望每位教師被指派之學分數能夠平均分配，而不至於造成某些教師之授課學分數太多，而有些教師之授課學分數太少。以避免教師對於派課結果有所不滿。

$d_3(X)$ 代表違反 SC3 之 cost，其中 SC3 表示同一教師被指派之相異課程數目越少越好的限制式，亦即派課結果中教師被指派之相異課程數目越多，$d_3(X)$ 之值越大。這裡要說明的是所謂的課組是指由課程名稱相同之課

程所成的集合。例如：數學系負責全校微積分課程的開授，因為欲修課之學生眾多，因此該系開設四門微積分課程以供全校學生選修，如表 4 所示。雖然實質上有四門微積分課程，但其皆屬於同一個課程。因此，將這四門微積分課程歸類於同一課組，此課組即為集合{Calculus_001,… , Calculus_004}。由表 4 為範例可得知對於教師 1，他/她被指派到三門微積分課程，但是實際上這三門課是同一課程，所以在計算時會將此三門課程視為同一課程。

表 4 課組範例表

	教師 / 課程	教師 1	教師 2	…
1	Calculus_001	v		
2	Calculus_002	v		
3	Calculus_003		v	
4	Calculus_004	v		

5.研究成果

本節將模擬一個六名教師、二十門課程之大專院校科系進行派課問題之研究，粒子群演算法與基因演算法之相關參數設定如表 5 所示，由於本研究期望比較粒子群演算法與基因演算法執行效能之差異性，因此固定執行次數與群體大小，以降低因為參數不同所造成的差異。本研究所使用之電腦配備:CPU 為 Intel P4 2.80G、RAM 1.5G。

表 5 演算法參數設定

PSO 演算法		GA 演算法	
執行代數	10000	演化代數	10000
粒子數	60	染色體數	60
學習因子 1	2	交配率	0.5
學習因子 2	2	突變率	0.01

根據上述設定，以本研究之派課模型對於粒子群演算法與基因演算法各執行 100 次運算，其運算結果及變異數分析結果如下:

表 6 演算法執行結果

	PSO 演算法	GA 演算法
執行次數	100	100
平均適應值	0.336871	0.317551
最佳適應值	0.410959	0.397609
最差適應值	0.275067	0.272840
平均執行時間	52(sec)	164(sec)

表 7 變異數分析結果

模式	平方和	自由度	平均平方和	顯著性
迴歸	4.1350	1	4.135	0.000
殘差	45.865	198	0.232	
總和	50.000	199		

預測變數:執行結果
依變數:演算法類型

根據研究結果可得知粒子群演算法之執行結果較基因演算法為優，並且以執行結果與演算法類型進行變異數分析可得知，不同演算法類型對於執行結果具有高度顯著性，所以使用粒子群演算法或是基因演算法對於所產生派課結果之優劣具有相當大的影響。而下圖為粒子群演算法與基因演算法各執行 100 次運算之平均適應值成長曲線圖。

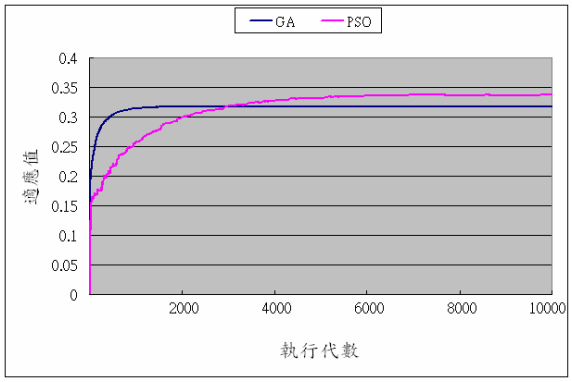

圖 6 平均適應值成長曲線

透過圖 6 可得知粒子群演算法收斂速度較慢，約在 5000 代開始收斂;而基因演算法收斂速度較快，約在 1000 代開始收斂，所以 3000 代之前基因演算法之執行成效明顯優於粒子群演算法，但是根據表 6 可觀察出基因演算法之平均執行時間高於粒子群演算法，因此本研究將兩者的執行時間進行正規化，所得出之平均適應值演化曲線圖如圖 7 所示。

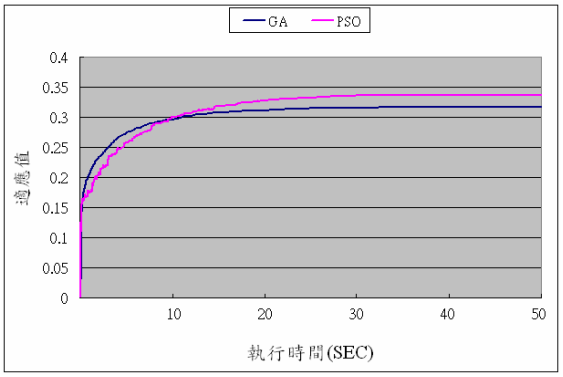

圖 7 時間正規化後之適應值成長曲線

根據圖 7 可得知將粒子群演算法與基因演算法之執行時間進行正規化之後，基因演算法之收斂速度明顯緩慢許多，兩者初期之執行成效經由變異數分析結果如表 8 所示，我們可得知不同演算法對於執行初期不具有顯著性。

表 8 變異數分析結果

模式	平方和	自由度	平均平方和	顯著性
迴歸	0.196	1	0.196	0.379
殘差	49.804	198	0252	
總和	50.000	199		

預測變數:執行結果
依變數:演算法類型

根據研究成果，基因演算法對於派課問題之執行成效在執行初期時遠優於粒子群演算法，其收斂速度與平均適應值皆具有顯著水準，但在相同的執行成本上兩者之執行成效並無明顯差異，而配合平均執行時間可得知基因演算法所耗費之成本較粒子群演算法來的高。後期兩者之比較上對於執行成效而言，粒子群演算法具有明顯差異，其不論平均適應值、最佳適應值與最差適應值相較於基因演算法皆有顯著優勢。因此，根據研究成果可得知以相同的執行成本，粒子群演算法之效能較佳;但若單看執行代數，基因演算法在執行初期之效能較佳，而後期粒子群演算法具有較突出之執行成果。

6. 結論

本研究期望藉由大專院校之派課問題，比較演算法的執行成效與執行成本的差異性，透過研究成果可得知粒子群演算法與基因演算法皆可用於求解派課問題，而在兩者之執行成效上，基因演算法在不考慮其高執行成本的條件下具有較佳之收斂速度，但是執行代數增加後，粒子群演算法所產生之派課結果具有較佳之適應值，若是加入執行成本的考量，更可觀察出粒子群演算法其執行效能與基因演算法之明顯差異。未來我們希望能夠探討更多不同類型之最佳化演算法，以進行執行效能之比較，觀察求解相同的排程問題時，其執行成果之優劣與成長曲線的趨勢。

參考文獻

[1] 游峰碩,王子夏,運用基因演算法於大專院校教師派課問題之研究，2007 年資訊科技國際研討會，2007。

[2] 游峰碩,王子夏, 運用基因演算法建構大專院校派課系統之研究，台灣作業研究學會 2007 年學術研討會，2007。

[3] B. Al-kazemi and C. K. Mohan, “Multi-phase discrete particle swarm optimization”, in Proc. Intl. Workshop Frontiers in Evolutionary Algorithms, 2002

[4] De Jong, K. A., “An analysis of the behavior of a class of genetic adaptive systems”, Doctoral dissertation, University of Michigan, 1975.

[5] Holland, J., Adaptation in Natural and Artificial System, University of Michigan Press, 1975.

[6] Hultberg, T. and Cardoso, D., The teacher assignment problem: A special case of the fixed charge transportation problem, European journal of operational research, vol.101, pp. 463-473, 1997.

[7] J. Kennedy and R. Eberhart, “A discrete binary version of the particle swarm optimization”, in Proc. IEEE Conf. Syst., Man, and Cybernetics, Orlando, FA, pp4104-4109, 1997.

[8] J. Kennedy and R. Eberhart, “Particle swarm optimization”, in Proc. IEEE Intl. Conf. Neural Networks, vol. 4, pp.1942-1948, 1995.

[9] Ozdemir, M. S., and Gasimov, R. N. The analytic hierarchy process and multiobjective 0-1 faculty course assignment, European Journal of Operational Research, 157, pp. 398-408, 2004.

[10] Parthiban, P., Ganesh, K., Narayanan, S. Dhanalakshmi, R., Preferences Based Decision-making Model (PDM) for Faculty Course Assignment Problem, In Proceedings of Engineering Management Conference, IEEE International, pp. 1338-1341, 2004.

[11] S. Yang, M. Wang, and L. Jiao,“A quantum particle swarm optimization”, in Cong. Evolutionary Computing, Vol1.1, pp.320-324, 2004

[12] Y. Shi and R. Eberhart, “A modified particle swarm optimizer”, in Proc. IEEE World Cong. On Computational Intelligence, pp96-73, 1998.

[13] Y. Shi, and R. Eberhart, “Parameter Selection in Particle Swarm Optimization,” V. W. Porto, N. Saravanan, D. Waagen, and A. E. Eiben (eds), Lecture Notes in Computer Science, 1447, Evolutionary Programming VII, Springer, Berlin, pp. 591−600, 1998.