通用啟發式演算法於背包問題之比較研究

陳明華[1]　周郁文[2]
嶺東科技大學 資訊管理研究所[1,2]
mhc@teamail.ltu.edu.tw[1]
vivian198715@yahoo.com.tw[2]

摘要

本研究希望透過多種演算法之比較分析，以找出較佳演算法。因此，選擇通用啟發式演算法中的基因演算法、粒子群演算法、量子遺傳算法此三種演算法應用於背包問題進行比較分析，以找出較佳演算法。

其中，量子遺傳算法是將量子計算與基因演算法相結合而衍生的新演算法。該演算法採用量子位元方式產生染色體，並透過量子旋轉門的改變進行種群的演化。因此將量子遺傳算法與基因演算法、粒子群演算法進行比較，以驗證量子遺傳算法是否能找到更好適應值。

首先將基因演算法與粒子群演算法，依照最基本的演算法進行改良，將兩種演算法各發展出六種不同的演算程序，並應用於背包問題，進行比較並找出較佳的演算程序。之後將較佳之基因演算程序、粒子群演算程序與量子遺傳算法依照不同物品數進行比較分析，其結果發現量子遺傳算法性能優於基因演算法與粒子群演算法。

關鍵詞：基因演算法、量子遺傳算法、粒子群演算法、背包問題。

1. 前言

通用啟發式演算法(meta-heuristic algorithm)一般用以求解組合性最佳化(combinatorial optimization)問題。通用啟發式演算法的概念是由觀察自然界所獲得的想法，而常用通用啟發式演算法有以下六種：模擬退火法(simulated annealing，SA)、禁忌搜尋法(tabu search，TS)、基因演算法(genetic algorithms，GA)、粒子群最佳化演算法(particle swarm optimization，PSO)、蟻群最佳化演算法(ant colony optimization，ACO)、量子遺傳算法(Quantum-inspired genetic algorithm，QGA)[4]。

其中 QGA 是將量子計算與 GA 相結合而衍生出新的混合式演算法，因此本研究將探討此新演算法其性能是否優於 GA。另外，因為 PSO 與 GA 類似，因此一併選入進行比較，以找出較佳演算法。

2. 研究理論與方法

本節將針對所使用之演算法進行簡單介紹，其詳細理論請參考各演算法之文獻。

2.1 基因演算法

基因演算法(Genetic Algorithms, GA)，為 John Holland 所提出的，主要是根據生物學家達爾文的物種進化論而發展出的演算法，也就是利用基因遺傳的原則－基因有選擇(Selection)、交配(Crossover)及突變(Mutation)的能力[1]。GA 根據問題所定義的適應值(Fitness)作為指標，再透過基因運算：選擇、交配及突變進行適當調整搜尋方向及區域，以找到問題的最佳解或近似最佳解[3]。

本研究依照原始 GA 進行改良，其 GA_1 是在選擇的部分將父母隨機選擇改為父輪盤、母隨機方式進行測試。GA_2 突變的部分則是在演化前半段使用大突變。GA_3 交配與突變部分則是以自適應方式進行。GA_4 交配分部則由原始的隨機單點切入改為隨機雙點切入。GA_5 交配分部使用自適應方式，突變則用多基因隨機突變加上自適應，表 1 為本研究所歸類出的 GA 之方法。

表 1 本研究所使用之 GA 方法

代號	方法名稱	選擇	交配	突變
GA_0	最基本 GA	父母隨機	隨機單點切	單基因隨機突變
GA_1	基本 GA	父輪盤母隨機	隨機單點切	單基因隨機突變
GA_2	大突變 GA	父輪盤母隨機	隨機單點切	演化前半段使用大突變
GA_3	自適應 GA	父輪盤母隨機	自適應單點切	自適應
GA_4	雙切點交配 GA	父輪盤母隨機	隨機雙點切	單基因隨機突變
GA_5	自適應多基因突變 GA	父輪盤母隨機	自適應單點切	多基因隨機突變+自適應

2.2 粒子群最佳化演算法

粒子群最佳化演算法(Particle Swarm Optimization，PSO)，由 James Kennedy 和 Russell Eberhart 兩位學者於 1995 年時所提出，是一種以群體為基礎(Population-based)的最佳化搜尋技術，是

一種模擬鳥群覓食的社會行為所衍生的技術[5,6]。PSO 與 GA 類似，同樣都是先產生一組初始解，再經過進化的方式來取得最佳值，不同的地方為 PSO 沒有交配(Crossover)及突變(Mutation)，是屬於單向的訊息流動，整個搜尋更新過程是跟隨當前最佳解的機制。因此，與 GA 相比較之下，PSO 能更快的收斂於最佳解處[2]。

本研究依照原始 PSO 進行改良，其 PSO_1 在權重部分使用自適應權重法。PSO_2 在權重部分使用隨機權重法。PSO_3 則是將學習因子的部分修改為異步變化學習因子。PSO_4 為二階粒子群演算法。PSO_5 為加入交配機率的混合式 PSO，其表 2 為本研究所歸類出的 PSO 之方法。

表 2 本研究所使用之 PSO 方法

代號	方法名稱	類別
PSO_0	最最基本 PSO	基本 PSO
PSO_1	自適應權重法 PSO	權重改進
PSO_2	隨機權重法 PSO	
PSO_3	異步變化學習因子 PSO	改變學習因子
PSO_4	二階 PSO	二階
PSO_5	基於雜交 PSO	混合

2.3 量子遺傳算法

量子遺傳算法 (Quantum-inspired genetic algorithm，QGA)由 Narayanan 等學者於 1996 年提出，概念為將量子計算與 GA 相結合而衍生的新演算法[8]。

2.3.1 量子比特編碼

在 QGA 中，染色體的基因是用量子位元(Q-bit)或機率幅表示，一個 Q-bit 可表示為：

$$|\psi\rangle = \alpha|0\rangle + \beta|1\rangle \quad (1)$$

其中 α、β 分別為 0 或 1 的機率幅，且需滿足 $|\alpha|^2+|\beta|^2=1$ 的條件。因此一個長度為 m 的量子染色體可表示為：

$$\begin{bmatrix} \alpha_1 & \alpha_2 & \dots & \alpha_m \\ \beta_1 & \beta_2 & \dots & \beta_m \end{bmatrix} \quad (2)$$

其中 $|\alpha|^2+|\beta|^2=1$，$i=1,2,\dots,m$。

2.3.2 量子門更新操作

Q-bit 透過量子門旋轉進行演化，因此量子門的設計是 QGA 實現的關鍵，設計的好壞，會直接影響 QGA 的性能。而本研究的量子門旋轉角度採用 Kuk-Hyun Han 等學者所提出的設計。其量子旋轉門更新如下[7]：

$$\begin{bmatrix} \alpha_i' \\ \beta_i' \end{bmatrix} = \begin{bmatrix} \cos(\theta_i) & -\sin(\theta_i) \\ \sin(\theta_i) & \cos(\theta_i) \end{bmatrix} \begin{bmatrix} \alpha_i \\ \beta_i \end{bmatrix} \quad (3)$$

其中，(α_i,β_i)為染色體中第 i 個 Q-bit，$\theta_i = s(\alpha_i,\beta_i)\cdot\triangle\theta_i$，而 (α_i',β_i') 值為通過量子旋轉門更新後的新基因，$s(\alpha_i,\beta_i)$和$\triangle\theta_i$為已知設定值，可由表 3 查詢旋轉角度。本研究在這部分的參數值採用 Kuk-Hyun Han 等學者研究中所設計的角度[7]。其相關詳細 QGA 之理論請參考文獻[7]。

表 3 旋轉角調整表

x_i	b_i	$f(x)\geqq f(b)$	$\triangle\theta_i$	$s(\alpha_i,\beta_i)$			
				$\alpha_i\beta_i>0$	$\alpha_i\beta_i<0$	$\alpha_i=0$	$\beta_i=0$
0	0	false	0	0	0	0	0
0	0	true	0	0	0	0	0
0	1	false	0	0	0	0	0
0	1	true	0.05π	-1	+1	±1	0
1	0	false	0.01π	-1	+1	±1	0
1	0	true	0.025π	+1	-1	0	±1
1	1	false	0.005π	+1	-1	0	±1
1	1	true	0.025π	+1	-1	0	±1

3. 研究問題與參數設定

所謂的背包問題(Knapsack Problem)，是指有一個背包和 m 個物品，其中第 i 個物品的重量為 $w_i(w_1,w_2,\dots,w_i,\dots,w_m)$，其所對應的價值為 $p_i(p_1,p_2,\dots,p_i,\dots,p_m)$，而背包能裝載的總重量為 C，如何在不超過總重量的限制下，該如何選擇物品才能使背包中物品的總價值最大?

在選擇裝入背包的物品時，對每種物品 i 只有兩種選擇，即放入背包或不放入背包。且不能將物品 i 裝入背包多次，也不能只裝入部分物品 i。其公式為：

$$\max f = \sum_{i=1}^{m} p_i x_i$$

$$\text{subject to} \quad \sum_{i=1}^{m} w_i x_i \leq C$$

$$\text{where} \quad x_i = \begin{cases} 0 & \text{，不選取該物品} \\ 1 & \text{，選取該物品} \end{cases} \quad (4)$$

本研究針對不同的物品數量進行測試，將物品個數 m 分別設為 100、250 和 500 個進行測試，其物品重量 w_i 為 1~10 之間均勻分布的隨機數，而物品價值 $p_i=w_i+5$，背包平均總重量為 $C = \frac{1}{2}\sum_{i=1}^{m} w_i$。

首先，先將物品以隨機方式產生並計算出相對價值及背包總重量。當 m=100 時，C=281.5；當 m=250 時，C=736.5；當 m=500 時，C=1365.5。將這三組參數分別代入GA、PSO及QGA中求背包問題的解。

3.1 GA 與 PSO 之參數設定

本研究在 GA 與 PSO 參數設計上，為參考其他學者文獻並經過測試設計而成，其參數值如表 4 與表 5。

表 4 各種 GA 之參數設定

演算法代號		GA_0	GA_1	GA_2	GA_3	GA_4	GA_5
已知條件參數設定	交配機率 1	0.9	0.9	0.9	0.9	0.9	0.9
	交配機率 2	—	—	—	0.5	—	0.5
	突變機率 1	0.04	0.04	0.04	0.04	0.04	0.04
	突變機率 2	—	—	—	0.02	—	0.02
	密集因子	—	—	0.6	—	—	—
	大突變機率	—	—	0.2	—	—	—

表 5 各種 PSO 之參數設定

演算法代號		PSO_0	PSO_1	PSO_2	PSO_3	PSO_4	PSO_5
已知條件參數設定	學習因子 1	2	2	2	—	1	2
	學習因子 1 最大值	—	—	—	2.5	—	—
	學習因子 1 最小值	—	—	—	0.5	—	—
	學習因子 2	2	2	2	—	1	2
	學習因子 2 最大值	—	—	—	0.5	—	—
	學習因子 2 最小值	—	—	—	2.5	—	—
	慣性權重(w)	0.5	—	—	0.9	0.7	0.7
	最大權重	—	0.9	—	—	—	—
	隨機權重平均值的最大值	—	—	0.8	—	—	—
	最小權重	—	0.4	—	—	—	—
	隨機權重平均值的最小值	—	—	0.5	—	—	—
	隨機權重方差	—	—	0.2	—	—	—
	交配機率	—	—	—	—	—	0.9
	交配池之大小比例	—	—	—	—	—	0.2

3.2 QGA 之染色體設定

QGA 中，一個量子染色體即可攜帶多個狀態的信息，能帶來豐富的種群，進而保持群體的多樣性。而本研究針對染色體的部分，分別針對不同的染色體數進行測試，各執行 50 次，並記錄每次的執行結果，並選出 50 次結果中，最優的適應值與最差的適應值及 50 次結果平均後之適應值。藉由此測試，以瞭解染色體增加，其適應值變化為何。

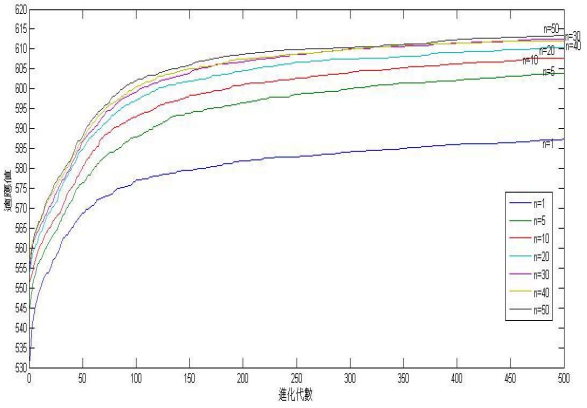

圖 1 QGA 染色體測試

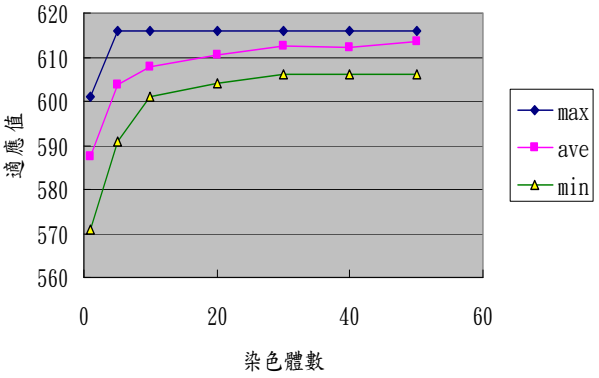

圖 2 各染色體之適應值比較

由圖 1、圖 2 中可看出，當染色體數 1~30 時平均適應值有逐漸成長的趨勢，但當染色體數 40 時，其平均適應值會略為下降，且就算在增加染色體數，適應值也沒有明顯的升高，因此本研究在染色體數參數設為 30。

4. 結果與討論

首先，將上述背包問題公式分別代入 GA 與 PSO 之各演算程序中，並各選出較佳的兩種演算程序，之後在與 GQA 相比較。

4.1 各種 GA 演算程序之比較

同樣的演算法，設計出的不同演算程序，其適應值差異並不大，但還是可以找出適應值較佳之演算程序。

由圖 3 中可明顯看出 GA_5 的適應值為最佳，其次為 GA_3，而 GA_0 適應值為最差，因此將選擇 GA_5 與 GA_3 此兩種演算程序再與 QGA、PSO 相比較。

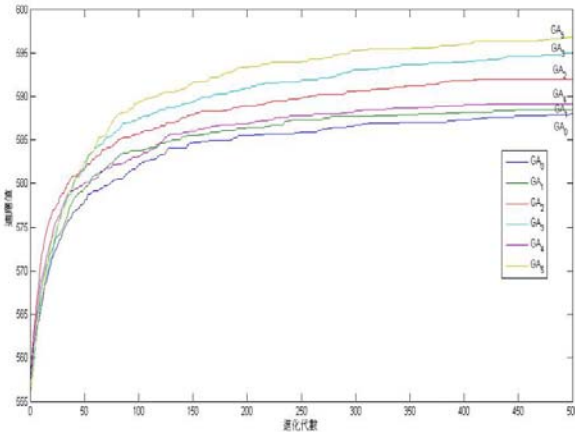

圖 3 各基因演算程序之適應值比較

4.2 各種 PSO 演算程序之比較

由圖 4 中可明顯看出 PSO_0 的適應值為最佳，其次為 PSO_4，而 PSO_3 適應值為最差，因此將選擇 PSO_0 與 PSO_4 此兩種演算程序再與 GA、QGA 相比較。

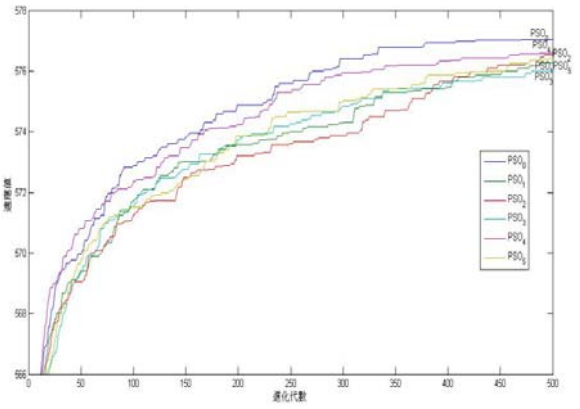

圖 4 各 PSO 演算程序之適應值比較

4.3 GA、QGA 與 PSO 之比較

將 GA、QGA 與 PSO 代入背包問題，各執行 50 次，並記錄每次執行結果之適應值，其表 6 為當物品數為 100 時，執行 50 次中最大及最小適應值，和 50 次平均之適應值。由圖 5 可看出一開始各演算法所找到的適應值差不多，而 PSO 到 50 代後可明顯看出其適應值變化並不明顯；GA 演化到 150 代後可明顯看出其適應值變化並不明顯，由圖中可發現，相同的演算法所求得的適應值差距並不大。在 QGA 的部分則可明顯看出所找到之適應值明顯優於 GA、與 PSO。在執行時間的部分由圖 6 可看出 PSO 的執行時間最久，而 GA 的執行時間最快。

表 6 當 m=100 時，GA、QGA 與 PSO 之適應值

物品個數	執行結果		GA_3	GA_5	QGA	PSO_0	PSO_7
100	適應值	最優	601	606	616	586	585
		平均	595	596	611	577	576
		最差	586	587	606	571	571
	所需時間(秒/次數)		1.73	1.88	17.35	78.09	58.74

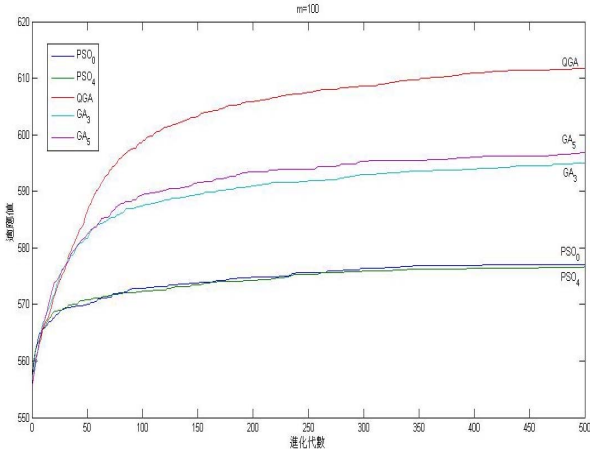

圖 5 當 m=100 時，各演算法之適應值

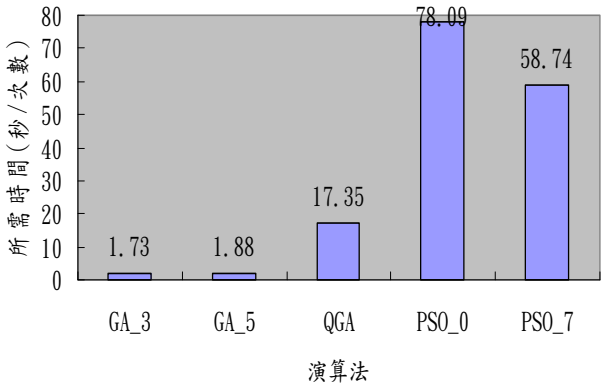

圖 6 當 m=100 時，各演算法執行時間

表 7 為當物品數為 250 時，執行 50 次中最大及最小適應值，和 50 次平均之適應值。由圖 7 可看出一開始 PSO 所找到的適應值較佳，但演化 50

代後就趨於平緩了；而 GA 雖然一開始適應值最差，但演化至 25 代後，其適應值便高於 PSO 所找尋到的最佳適應值。其 QGA 適應值優於其他演算法。在執行時間的部分由圖 8 可看出 PSO 的執行時間最久，而 GA 的執行時間最快。

表 7 當 m=250 時，GA、QGA 與 PSO 之適應值

物品個數	執行結果		GA_3	GA_5	QGA	PSO_0	PSO_7
250	適應值	最優	1506	1496	1536	1461	1458
		平均	1481	1477	1521	1436	1441
		最差	1461	1460	1491	1426	1426
	所需時間(秒/次數)		2.14	2.68	42.34	192.2	130.7

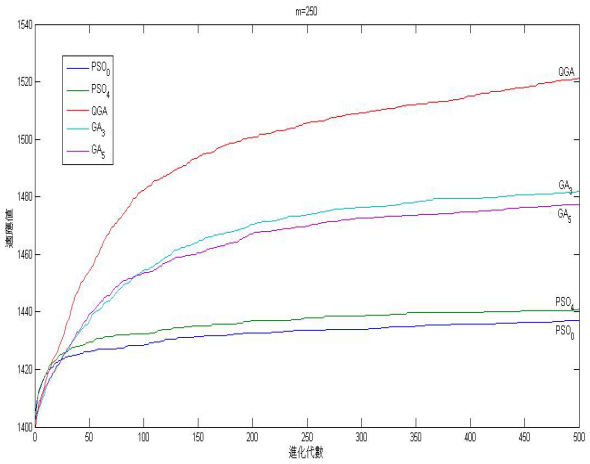

圖 7 當 m=250 時，各演算法之適應值

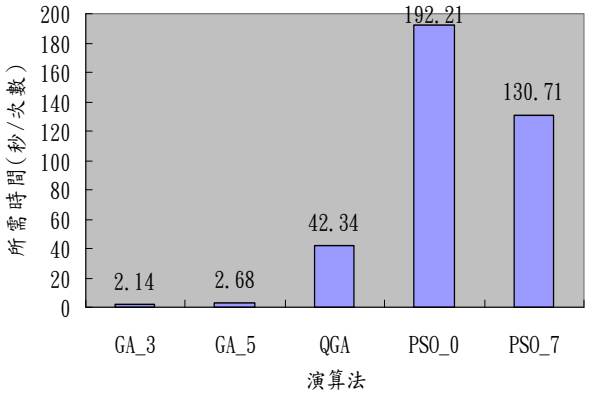

圖 8 當 m=250 時，各演算法執行時間

表 8 為當物品數為 500 時，執行 50 次中最大及最小適應值，和 50 次平均之適應值。由圖 9 可看出一開始 PSO 所找到的適應值較佳，但演化 50 代後就趨於平緩了；而 GA 雖然一開始適應值最差，但演化至 75 代後，其適應值便高於 PSO 所找尋到的最佳適應值。其 QGA 適應值優於其他演算法。在執行時間的部分由圖 10 可看出 PSO 的執行時間最久，而 GA 的執行時間最快。

表 8 當 m=500 時，GA、QGA 與 PSO 之適應值

物品個數	執行結果		GA_3	GA_5	QGA	PSO_0	PSO_7
500	適應值	最優	2855	2840	2915	2775	2800
		平均	2817	2807	2876	5742	2759
		最差	2780	2764	2824	2724	2727
	所需時間(秒/次數)		2.74	3.63	93.92	384.7	321.7

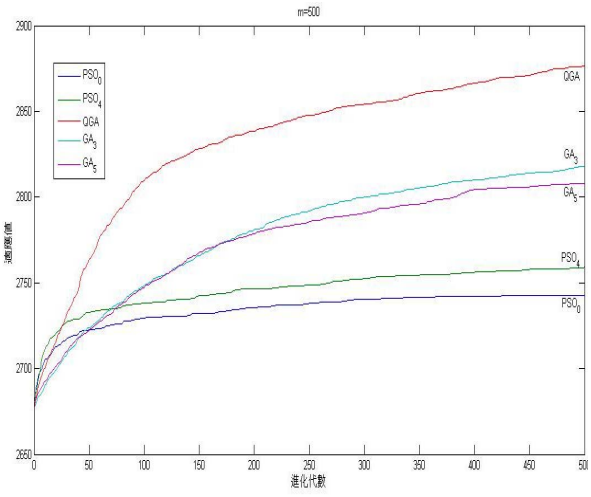

圖 9 當 m=500 時，各演算法之適應值

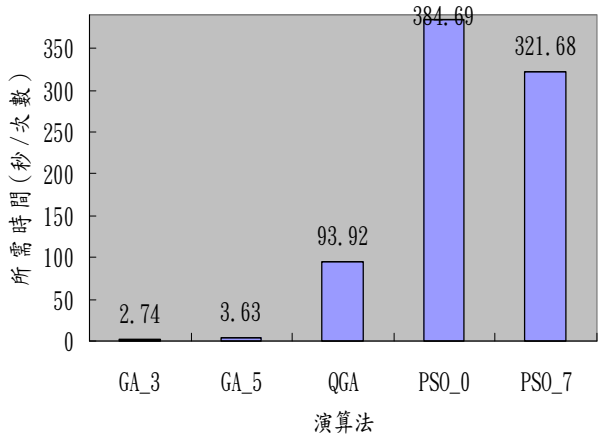

圖 10 當 m=500 時，各演算法執行時間

接著將此三種物品所得到的結果加總後平均再進行比較，其結果如表 9。藉由圖 11 可明顯看出，執行 50 次後，所得到的最大、最小及平均適應值，皆以 QGA 所得的值為最優。由圖 12 中可看出其執行時間以 PSO 為最久，GA 為最快。雖然 QGA 執行速度不如 GA，但所得之適應值卻優於 GA，能找出更優的近似最佳解。

表 9 總平均後 GA、QGA 與 PSO 之適應值

物品個數	執行結果		GA_3	GA_5	QGA	PSO_0	PSO_7
總平均	適應值	最優	1654	1647	1689	1607	1614
		平均	1631	1627	1669	1585	1592
		最差	1609	1603	1640	1573	1574
	所需時間(秒/次數)		2.2	2.73	51.2	218.3	170.4

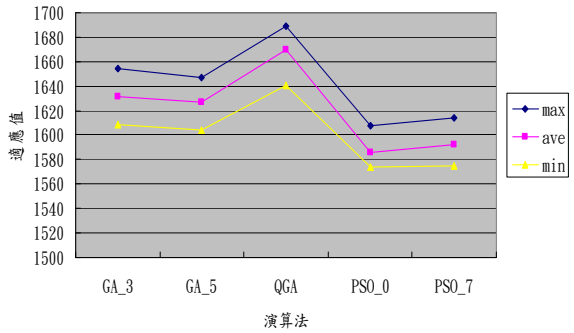

圖 11 總平均後各演算法之適應值

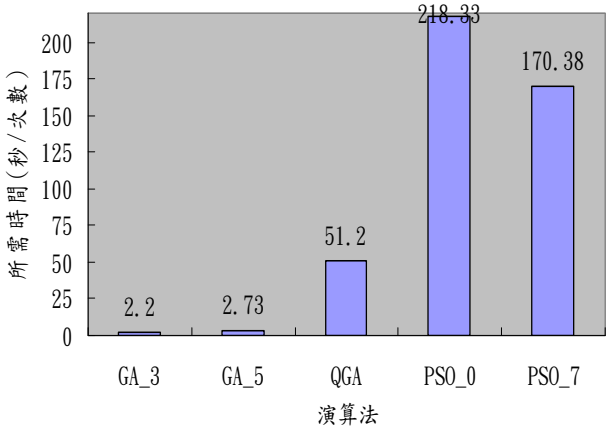

圖 12 總平均後各演算法執行時間

5. 結論與建議

就研究結果，經分析討論後，提出以下結論與建議：

(1) 當 QGA 的種群規模較小時，其適應值與 GA、PSO 差距不多，但若將種群規模提高，則可發現所求得的適應值遠高於 GA、PSO。

(2) 執行速度的部分，QGA 的執行時間會比 GA 久，經程式剖析後發現量子門旋轉的部分要花較長的時間計算搜尋。因此後續研究的部分可針對量子門旋轉進行修改，以提升 QGA 收斂速度。

(3) 本研究在量子旋轉門的部分，$\triangle\theta$ 值未進行測試，後續研究可針對此部分進行測試，評估是否能找到更好的適應值。

(4) 由研究結果中發現 GA 收斂速度為最快，因此後續研究可將 GA 的交配與突變套入 QGA 中，以研究是否能加快收斂速度並提高性能。

參考文獻

[1] 周鵬程，「遺傳算法原理與應用－活用 Matlab(修定三版)」，全華圖書股份有限公司，2007 年 8 月。

[2] 郭信川、張建仁、劉清祥，「粒子群演算法於最佳化問題之研究」，科技與管理學術研討會 pp419-432，2004。

[3] 黃意純，「應用基因演算法於機台組態配置之研究」，南華大學資訊管理學系碩士論文，2009。

[4] 廖慶榮，「作業研究」，華泰文化出版，2009 年 2 版。

[5] Eberhart, R.C. and Kennedy, J., “A new optimizer using particle swarm theory.” Proc. Sixth International Symposium on Micro Machine and Human Science, Nagoya, Japan, pp.39-43, 1995.

[6] Kennedy, J. and Eberhart, R.C, “Particle swarm optimization.” Proc. IEEE International Conference on Neural Networks (Perth, Australia), IEEE Service Center, Piscataway, NJ, pp. IV：1942-1948, 1995.

[7] Kuk-Hyun Han, Chi-Ho Lee, Jong-Hwan Kim, “Parallel Quantum-inspired Genetic Algorithm for Combinatorial Optimization Problem.” Proceedings of the 2001 IEEE congress on Evolutionary Computation Seoul, Korea., pp1422-1429, 2001.

[8] Narayanan A, Moore M., “Quantum inspired genetic algorithms.” Proceedings of the 1996 IEEE International Conference on Evolutionary Computation (ICEC96). USA:IEEE Press,.pp61-66, 1996.