

路徑基礎類粒子群最佳化演算法於 求解含凹形節線成本最小成本轉運 問題之研究

A PATH-BASED ANALOGOUS PARTICLE SWARM OPTIMIZATION ALGORITHM FOR MINIMUM COST NETWORK FLOW PROBLEMS WITH CONCAVE ARC COSTS

顏上堯 Shangyao Yan¹
李旺蒼 Wang-Tsang Lee²
施佑林 Yu-Lin Shih³

(95 年 10 月 4 日收稿，95 年 12 月 25 日第一次修改，96 年 8 月 10 日定稿)

摘 要

本研究針對含平方根凹形節線成本之最小成本網路流動問題，以粒子群最佳化演算法之搜尋概念為基礎，並結合遺傳演算法、門檻值接受法與凹形成本網路啟發解法之技術，發展一以路徑為基礎之混合式全域搜尋法，以有效的求解問題。為評估本演算法之求解績效，本研究隨機產生多個網路問題，並以 C++ 語言撰寫所有相關的電腦程式，進行測試分析。測試結果顯示本演算法比新近發展之鄰近搜尋演算法及遺傳演算法更能有效地求解含平方根凹形節線成本之最小成本網路流動問題。

-
1. 國立中央大學土木工程學系教授 (聯絡地址：320 桃園縣中壢市五權里中大路 300 號中央大學土木工程學系；電話：03-4227151 轉 34141；E-mail：t320002@cc.ncu.edu.tw)。
 2. 國立中央大學土木工程學系碩士。
 3. 國立中央大學土木工程學系博士生。

關鍵詞： 凹形節線成本、網路流動問題、粒子群最佳化演算法、遺傳演算法、門檻值接受法

ABSTRACT

In this research, a particle swarm optimization algorithm was employed, coupled with the techniques of a genetic algorithm, and threshold acceptance method and concave cost network heuristics, to develop a path-based global search algorithm for efficiently solving minimum cost network flow problems with square root concave arc costs. To evaluate the proposed algorithm, several network flow problems are randomly generated. C++ is used to code all the necessary programs for the tests. The results indicate that the proposed algorithm is more effective than recently designed local search algorithms and genetic algorithms for solving minimum cost network flow problems with square root concave arc costs.

Key Words: Concave arc cost; Network flow problem; Particle swarm optimization; Genetic algorithm; Threshold accepting

一、緒 論

轉運問題 (transshipment problem) 為運輸界之一知名問題，其意義乃「以最小總成本將人或貨物由各起點運送至各終點，各貨物運送過程中可經過數個轉運點，且各個起點或終點皆可為其他貨物運送之轉運點」。此問題在數學上可定式為最小成本網路流動問題。以往學界或實務界在最小成本網路流動問題的定式上，常將各節線之單位運送成本予以簡化為固定值，即以線性成本來定式運送成本，其主要原因在於簡化後的線性成本較易於求解。然而，此假設常與許多實際貨物之運送成本不相符合，實際上單位運送成本常隨著運送貨物的增加而減少，亦即為經濟學上常見之經濟規模特性。一般而言，高固定成本或低變動成本的運送方式，例如鐵路與水運，通常貨物運送的每單位成本隨貨物量的增加而減少。基本上，由於實務的經濟規模特性，因此貨物運送成本函數常可定式為凹形成本 (Yan 與 Luo^[1,2])。

本研究針對含凹形節線成本的最小成本網路流動問題構建求解演算法，令 $G = (N, A)$ 為一有方向性之網路，其中 N 為所有節點之集合， A 為節線集合。此問題可定式如下：

$$\text{Min} \quad \sum_{ij \in A} f_{ij}(x_{ij}). \quad (1)$$

$$\text{S.T.} \quad \sum_j x_{ij} - \sum_k x_{ki} = b_i, \quad \forall i \in N; \quad (2)$$

$$x_{ij} \geq 0, \quad \forall (i, j) \in A. \quad (3)$$

其中，目標式 (1) 為最小化總成本，節線 ij 的成本函數 $f_{ij}(x_{ij})$ 為凹形成本型態。為簡化問題的複雜度並方便測試，本研究參考 Yan 與 Luo^[2] 及 Yan 等人^[3] 中節線成本的設計方式，假設節線 ij 之運送總成本為 $f_{ij} = c_{ij}\sqrt{x_{ij}}$ ，即單位運送成本為 $f_{ij} = c_{ij} / \sqrt{x_{ij}}$ ，其中 x_{ij} 為節線 ij 之流量， c_{ij} 為節線 ij 之成本參數。請注意，當 $x_{ij} = 0$ ，則此單位運送成本未定義。在實務上，此成本函數型態類似許多運輸問題之成本函數。至於其他類凹形成本函數的課題，可為後續研究。

限制式 (2) 為節點流量守恆限制式，當 i 點為供給點時， $b_i > 0$ ；為需求點時， $b_i < 0$ ；其餘各點（為轉運點） $b_i = 0$ 。限制式 (3) 為節點流量非負限制式。為簡化研究測試，本研究假設各節線流量無上限的限制。實際上，有節線流量上限的問題，可透過網路修正改為無節線流量上限的問題 (Ahuja 等人^[4])。值得注意的是，為確保此最小成本網路流動問題有可行解，本研究設定總供給量等於總需求量，因此後節產生測試的網路時必須滿足 $\sum_i b_i = 0$ 之條件。至於在實務上可能面臨的總供需量不均衡的情形，一般是以加入虛擬節點及虛擬供需量的方式，使總供需量平衡後再求解。

求解上，最小成本網路流動問題及其簡化後之特殊問題，如運輸問題 (transportation problem)、最短路徑問題、最大流量問題等傳統網路規劃問題，多年來學術界已發展多種適合之求解方式 (Ahuja 等人^[4])。然而，含凹形節線成本之最小成本網路流動問題，則因在網路流動規劃問題中屬於 NP-hard 問題 (Garey 與 Johnson^[5])，故難在有限時間內求得大型問題的最佳解。傳統上，求解含凹形節線成本之運輸問題，一般可分為兩大方向：(1) 直接將問題形式簡化為線性問題，以線性規劃的方式求解；(2) 將非線性問題分成數個子問題，以線性近似法求解，如 Zangwill^[6]、Rech 與 Barton^[7]、Gallo 與 Sandi^[8]、Gallo 等人^[9]、Jordan^[10]、Blumenfeld 等人^[11]、Thach^[12]、Guisewite 與 Pardalos^[13]、Yaged^[14]、Larsson 等人^[15]、Balakrishnan 與 Graves^[16]、Amiri 與 Pirkul^[17]、Nourie 與 Guder^[18]、Dukwon 與 Panos^[19]、Suwan 與 Sawased^[20]、Kuhn 與 Baumol^[21] 等。然而，透過簡化後的線性成本，常難以反映實際成本，而不易求得近似最佳解；而以線性近似法求解凹形成本的轉運問題，當問題規模變大時，則傳統整數規劃模式將因問題複雜度增加而難以求解 (Yan 與 Luo^[2])。

近年來，Yan 與 Luo^[1,2] 利用新近組合最佳化的方法，如模擬退火法 (simulated annealing, SA) (Kirkpatrick 等人^[22])、禁制搜尋法 (tabu search) (Glover^[23] 及 Glover 與 Laguna^[24])、門檻值接受法 (threshold accepting, TA) (Dueck 與 Scheuer^[25]) 等鄰近搜尋法，針對單純之含凹形節線成本運輸問題，發展求解演算法，且經過測試及比較傳統啟發解後，發現效果頗佳。繼之，顏上堯等人^[26] 亦曾利用此等鄰近搜尋法，進一步探討含凹形節線成本最小成本轉運問題。然而，此等鄰近搜尋法在求解含凹形節線成本之最小成本網路流動問題上，仍會面臨退化的問題，造成搜尋上的低效率，且是否能有效的搜尋全域以求得最佳解 (或近似最佳解)，往往不得而知。因此 Yan 等人^[3] 進一步利用遺傳演算法 (genetic algorithm) (Goldberg^[27])，發展一求解演算法，並經測試後，發現其求解效果的確較上述鄰近搜尋法

為佳。

粒子群最佳化 (particle swarm optimization, PSO) 演算法是由 Eberhart 與 Kennedy^[28,29] 所提出，主要概念源自於對動物群體行為的研究。粒子群最佳化具有兩個主要的基本概念，其一為藉由觀察人類的決策過程，建立起個體學習與文化傳遞的兩種觀念 (Boyd 與 Richerson^[30])。另一為對於自然界生物的群體行為提出一些簡單的法則，並將其模組化。Reynolds^[31] 提出群體中每個個體行為可藉由跟隨離目標最近的個體移動、朝著目標移動及朝群體中心移動等三種向量方式模組化，並且產生複雜的群體行為。基於上述的兩個基本概念和模擬鳥群飛行的群體行為，Eberhart 與 Kennedy^[28] 提出粒子群最佳化的概念。Eberhart 與 Kennedy^[29] 並提出連續型粒子群最佳化的更新基本法則。其中，各粒子速度依據各粒子本身速度、與個體最佳解的距離及與群體最佳解的距離等三個要素更新，再依據更新後的速度調整目前的位置，以更新每個粒子的搜尋距離與方向。Kennedy 與 Eberhart^[32] 針對離散性最佳化問題提出二進位粒子群最佳化 (binary particle swarm optimization, BPSO) 演算法，其粒子疊代之更新仍依循個體暫時最佳解與群體暫時最佳解之導引方式。之後，Shi 與 Eberhart^[33] 將慣性權重值 (w) 引入基本的更新法則內，並藉由慣性權重值 w 之設計使粒子群在大範圍的搜尋之後，能降低粒子的速度，迫使粒子進入一個較好的區域搜尋，期能以更有效率的方式找到更好的可行解。Shi 與 Eberhart^[34] 另提出三個更新法則，分別為慣性權重法 (inertia weight method)、最大速度法 (max V method) 和收縮係數法 (constriction factor method)。至於其他運用 PSO 之求解概念以求解問題之研究，如 Shi^[35]、Kennedy 與 Spears^[36]、Shi 與 Eberhart^[37]、Salerno^[38]、曾俊傑^[39]、張榮芳^[40]、徐育良^[41]、葉麗雯^[42] 及葉思緯^[43] 等，皆發現效果頗佳。

綜合以往文獻得知，粒子群最佳化具有分散式搜尋、具記憶性、所需元件及變數較少、容易結合其他演算法等特性，因此極具求解組合最佳化問題的潛力。由於過去未曾有利用此演算法求解含凹形節線成本之最小成本網路流動問題，因此本研究針對此一問題，以粒子群最佳化演算法之搜尋觀念為基礎，結合凹形成本網路啟發解法與 GA、TA 等巨集啟發解法之技術，發展有效的演算法，以有效地求解含凹形節線成本之最小成本網路流動問題。本文其餘架構如下，第二節設計求解演算法，第三節進行實驗測試，第四節提出結論與建議。

二、求解演算法設計

本研究以供需節點路徑為基礎，結合二進位粒子群最佳化的演算求解概念，及凹形成本網路啟發解法、GA、TA 等巨集啟發解法之技術，發展一類粒子群最佳化演算法 (analogous particle swarm optimization, APSO)。本研究演算法的設計架構如圖 1 所示，包含初始解產生策略、初始供需節點路徑集合及初始機率值之設定、可行解產生方式、供給點指派順序策略、路徑集合更新策略、機率值更新方式、門檻值個體與全體最佳解更新策略、

鄰近搜尋法之改善策略、菁英保留策略、運算終止機制等要素。本研究演算法的流程如下：首先以隨機與凹形特性兩種方式產生初始可行解（為可行解伸展樹），一可行解為一粒子。接著，產生初始供需節點路徑集合。本研究以兩種方式產生初始供需節點路徑，一為在節線成本參數 c_{ij} 下（即流量 x_{ij} 為 1 下），以標籤修正法 (label correcting method) 求解各供給點到各需求點之最短路徑，作為各供給點至各需求點之基本路徑；另一為依各初始解可行解伸展樹之路段流量限制，及供需節點之供需量限制，以流量從需求點回推至供給點之方式，搜尋此可行解伸展樹中符合供需均衡之供需節點路徑。將上述兩種方式所搜尋到的所有路徑作為粒子群之初始路徑集合及內生變數。之後，紀錄各組之個體暫時最佳解及全體暫時最佳解。再來，利用可行解產生策略並配合流量推擠法 (Yan 與 Young^[44]) 產生可行解，再藉由鄰近搜尋法改善可行解伸展樹以得到較佳解。然後，利用路徑集合更新策略，以確定路徑集合內各路徑在可行解伸展樹上之使用情況，使各路徑之機率值在更新過程中，能依照目標值之優劣進行調整，並能在較優良之可行解伸展樹內，搜尋新的供需節點路徑，並增加至各粒子之路徑集合中，以更新各粒子之供需節點對之路徑集合。之後，依據各可行解之目標值更新全體最佳解、個體最佳解及各路徑之機率值，並將此回合所得到的菁英解（一個或數個較佳解）保留進入下一回合，成為下回合之可行解。最後，重複上述過程直到滿足運算終止條件為止。各要素的具體作法如下：

2.1 初始解產生策略

本演算法的求解過程主要在結合粒子群最佳化之搜尋概念，以多個可行解所組成的群體進行平行搜尋以求解。因此，在初始群體方面需要產生出多個可行解，並期望此等可行解能廣泛的散布於可行解範圍內。本研究針對凹形成本特性及隨機方式提出兩種啟發式可行解產生策略，以產生初始群體，並以機率 P_m 控制初始群體中兩種解所占的比率，例如若 $P_m = 0.6$ 表示初始群體中有 60% 的凹形初始解。此兩種可行解產生策略的作法如下所述：

(一) 針對凹形成本特性產生

本研究參考顏上堯等人^[26]設計符合凹形成本特性的初始解產生策略，主要的做法係先依照供給量大小，由大到小決定供給點的選取順序，然後在每回合以當時的流量值為基礎，從一供給點求一最短路徑樹 (minimum path spanning tree)，並隨機選取一可運送到之需求點，之後將供給量以最短路徑運送至此需求點後，根據新的流量值 x_{ij} ，以成本函數 $c_{ij}/\sqrt{x_{ij}}$ 更新運送經過的節線成本（請注意，當 $x_{ij} = 0$ ，雖然此單位運送成本未定義，但為方便演算法之設計，茲參考 Yan 等人^[3]，令其單位運送成本為 c_{ij} ，即令 $x_{ij} = 1$ ），再進行下一個供給點之運送，並重複此作法直到所有供給量皆運送完成，則有流量的節線集合形成一可行解（可能不為伸展樹）。最後再利用流量推擠法 (Yan 與 Young^[44]) 將可行解轉成一可行解伸展樹，產生一初始解。請注意，利用此作法中需求點的隨機選擇方式，我們可產生多組不同的初始解。

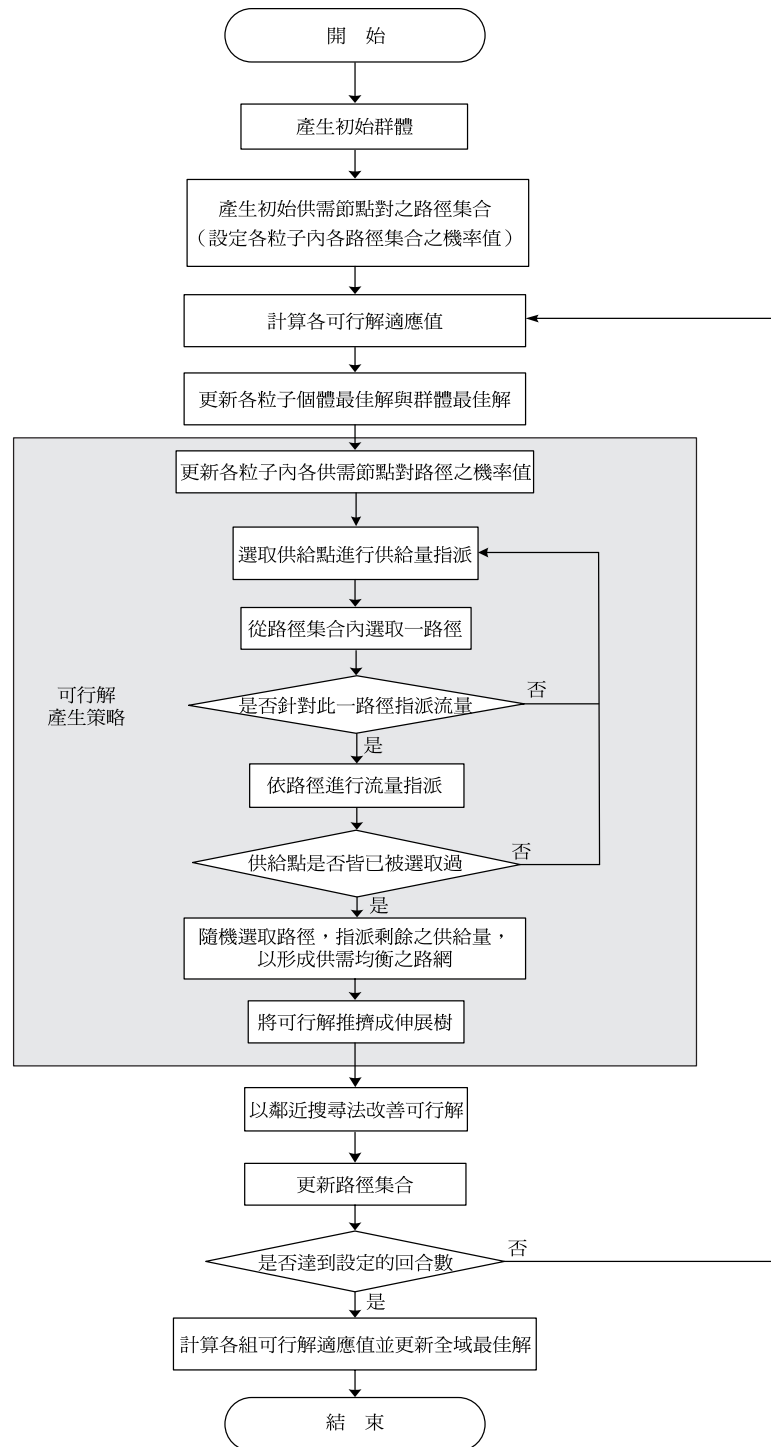


圖 1 演算法流程

(二)以隨機方式產生

本研究以隨機選取供給點與需求點的方式，指派運送量至此供需節點對的最短路徑節線上，當所有供給量皆運送完成時，則有流量的節線集合形成一可行解（可能不為伸展樹）。此隨機初始解產生方式與凹形成本產生方式的最大不同點在於，供給點選取方式係隨機選取，且在每回合從一供給點將供給量以最短路徑運送至需求點後，無須以成本函數 $c_{ij}/\sqrt{x_{ij}}$ 更新節線成本，即可再進行下一個供給點之運送。最後，同樣地利用流量推擠法 (Yan 與 Young^[44]) 將可行解轉成一可行解伸展樹，產生一初始解。此作法中透過供給點與需求點的隨機選取方式可產生多組不同的初始解。

2.2 初始供需節點路徑集合及初始路徑機率值之設定

本研究在初始供需節點路徑集合方面，採用兩種方式產生供需節點對之路徑。第一種為節線成本參數 c_{ij} 下 (即令流量 x_{ij} 為 1 下)，以標籤修正法求解各供給點到各需求點之最短路徑，作為各供給點至各需求點之基本路徑；第二種較為複雜，係依各初始解可行解伸展樹之路段流量限制，及供需節點之供需量限制，以流量從需求點回推至供給點之方式，以搜尋此可行解伸展樹中符合供需均衡之供需節點路徑。各可行解供需路徑的產生作法相同，茲舉較複雜的第二種路徑集合產生方式，詳列步驟如下：

- 步驟 1. 紀錄各供給點供給量 S 、需求點需求量 D 及各節線流量 L 。
- 步驟 2. 若仍有供給點 i 之供給量 $S_i > 0$ 及需求點 j 之需求量 $D_j > 0$ ，則進行步驟 3；否則，結束供需節點路徑之搜尋。
- 步驟 3. 依照目前各節線上之流量，給予各節線之一虛擬成本 C'_{ij} 。若節線上之流量大於零，則令虛擬成本 $C'_{ij} = 0$ ；若節線流量等於零，則令虛擬成本等於原本節線之通過成本 $C'_{ij} = c_{ij}$ 。進行步驟 4。
- 步驟 4. 選取供給點 i 之供給量 $S_i > 0$ ，依照步驟 3 給定之節線成本並以標籤修正法求解此供給點 i 到各需求點之最短路徑。進行步驟 5。
- 步驟 5. 選取需求點 j 之需求量 $D_j > 0$ ，並依步驟 4 中以供給點 i 與需求點 j 為起迄點之路徑，判斷此路徑上最小流量之節線 a 、其流量 L_a 是否大於零？若是，則進行步驟 6；否則，持續進行步驟 5。
- 步驟 6. 令供需節點對最小供給量為 $f = \min\{S_i, D_j, L_a\}$ ，記錄供需點 i 、 j ，運送量 f 及路徑 k ，將此路徑增加至路徑集合內，並更新供給點 i 供給量為 $S_i = S_i - f$ 、需求點 j 需求量為 $D_j = D_j - f$ 及扣除此路徑上各節線 f 之流量。判斷供給點 i 之供給量 S_i 是否大於 0？若是，則回到步驟 5；若否，則回到步驟 2。

此外，本研究參考 Eberhart 與 Kennedy^[28] 以隨機方式給予各粒子內各路徑一介於 -6 到 6 之間的數值，以計算各路徑在求解過程中是否需被指派流量之機率值，其具體的作法容後說明。

2.3 可行解產生策略

本研究以類似初始群體可行解的產生方式，構建各回合的可行解伸展樹，與之前最大的不同處，在於供給點指派之優先順序與供需節點間路徑的挑選方式。在供給點指派優先順序方面，由於本研究發現其會直接影響各路徑被挑選的機會，進而影響到搜尋結果，因此本研究除了以隨機選取和依凹形成本特性等兩種類似初始解之供給點指派方式外，另提出一供給點指派順序策略（將在後文說明），藉此以有效率的引導粒子，收斂至較佳的可行解，以提升求解績效。在供需節點路徑的選擇方面，本研究除了以標籤修正法求解各供給點到各需求點之最短路徑集合外，並依初始解可行解伸展樹搜尋產生供需節點對之路徑集合，再將此集合加入前項最短路徑集合裡，之後依各路徑之機率值判斷是否需指派流量。綜合上述，針對一粒子搜尋可行解的作法如下：首先，利用機率更新公式及全體與個體的暫時最佳解，更新各路徑的機率值。之後，以隨機、凹形特性或供給點指派順序策略之方式挑選供給點，再以此供給點至各需求點路徑的機率值挑選路徑，並進行供需節點間之流量指派，以形成供需均衡之網路。最後，利用流量推擠法 (Yan 與 Young^[44]) 將非伸展樹可行解轉成可行解伸展樹。其具體步驟如下：

- 步驟 1. 利用機率更新公式及全體與個體暫時最佳解，更新各粒子內各路徑機率值。
- 步驟 2. 選擇供給點挑選方式（類似隨機初始解、類似凹形特性初始解或依 2.4 節之供給點指派順序策略）。
- 步驟 3. 選取未被標記過之供給點 i ，其供給量為 S_i ，並標記此一供給點，進行步驟 4；若無未被標記過之供給點，則進行步驟 5。
- 步驟 4. 依機率值判斷挑選之路徑是否指派流量，並紀錄其供需節點對與運送量：
 - 4.1 判斷供給點 i 到各需求點之路徑是否皆被標記過？若是，則回到步驟 3；否則，進行步驟 4.2。
 - 4.2 從供給點 i 到各需求點路徑集合中，以隨機方式選取一路徑之需求點需求量大於 0，且未被標記過之路徑 k ，路徑機率值為 v_k ，並標記此一路徑。
 - 4.3 針對此路徑隨機產生 $0 \sim 1$ 之隨機值 r ，以作為是否針對此路徑指派流量。若 $S(v_k) = \frac{1}{1 + (e^{-v_k})} > r$ ，則進行步驟 4.4；否則，回到步驟 4.1。
 - 4.4 記錄供需點 i 、 j ，運送量 D_j 及路徑 k 。若供給量 $S_i > D_j$ ，則 $S_i = S_i - D_j$ ， $D_j = 0$ ，回到步驟 4.1；若 $S_i \leq D_j$ ，則 $D_j = D_j - S_i$ ， $S_i = 0$ ，回到步驟 3。
- 步驟 5. 判斷所有供給點之供給量是否已經指派完畢？若是，則進行步驟 8；否則，進行步驟 6。
- 步驟 6. 記錄隨機選取之供需節點對與運送量：
 - 6.1 隨機選取供給量 $S_i > 0$ 之供給點 i ；若無 $S_i > 0$ 之供給點 i ，則進行步驟 8。
 - 6.2 由供給點 i 可到達之需求點中，隨機選取一需求點 j ，其剩餘需求量大於 0，並從

此供需節點對 (i, j) 之路徑集合中，隨機選擇一路徑 k ，進行步驟 6.3；若無 $D_j > 0$ 之需求點，則任選可到達之需求點 j ，進行步驟 7。

6.3 記錄供需點 i 、 j ，運送量 D_j 及路徑 k 。若 $S_i > D_j$ ，則 $S_i = S_i - D_j$ ， $D_j = 0$ ，回到步驟 6.2；若 $S_i \leq D_j$ ，則 $D_j = D_j - S_i$ ， $S_i = 0$ ，回到步驟 6.1。

步驟 7. 替換供需節點對與運送量：

7.1 從已指派之供需節點對之紀錄中，找出運送至 j 點之供給點 i' 、運送量為 f 及路徑為 k' 之供需節點對。若供給點 i 之運送量 $S_i < f$ ，則進行步驟 7.2；若 $S_i = f$ ，則進行步驟 7.3；若 $S_i > f$ ，則進行步驟 7.4。

7.2 更新此供需節點對 (i', j) 之運送量為 $f = f - S_i$ ，並紀錄供需節點對 (i, j) 、運送量為 S_i 及其路徑 k' ，回到步驟 6.1。

7.3 更新此供需節點對 (i', j) 之供需節點與運送量，並修改其供需節點為 i 、 j ，運送量為 S_i 及其路徑 k' ，回到步驟 6.1。

7.4 更新步驟 6.1 所選取之供需節點對 (i', j) ，並修改其供需節點為 i 、 j ，運送量為 f 及其路徑 k' ，再隨機選取 i 點可到達之需求點，回到步驟 7.1。

步驟 8. 依供需點運送量計算各路段流量：

8.1 以步驟 4.4、6.3 與 7 中記錄之各供需點對運送量與路徑，累加各路徑上同一路段之流量。

8.2 若節線上流量 F_{ij} 與 F_{ji} 皆大於 0，則 $F_{ij} = F_{ij} - \min(F_{ij}, F_{ji})$ 、 $F_{ji} = F_{ji} - \min(F_{ij}, F_{ji})$ 。

8.3 各路段流量之集合即為可行解。

步驟 9. 使用流量推擠法 (Yan 與 Young^[44]) 將可行解轉換為可行解伸展樹。

2.4 供給點指派順序策略

由於供給點指派之優先順序將會影響到目標值的優劣，因此本研究參考連續型粒子群最佳化之觀念，發展一供給點指派順序策略，藉此以有效率的引導粒子，收斂至較佳的可行解，其作法如下：在初始值的設定上，以隨機方式給予各供給點一介於 0 到 1 之間的數值，表示目前各供給點的位置。在求解的過程中，依式(4)與式(5)更新各供給點目前的速度及位置。之後，再依各供給點的位置由大到小依序排列，以決定供給點指派的優先順序。

$$v_{id}^{new} = w \times v_{id}^{old} + c_1 \times rand_1() \times (g_d - x_{id}^{old}) + c_2 \times rand_2() \times (p_{id} - x_{id}^{old}) \quad (4)$$

$$x_{id}^{new} = x_{id}^{old} + v_{id}^{new} \quad (5)$$

其中，
 i ：第 i 個粒子，即第 i 組可行解；
 d ：第 d 個供給點；
 w ：慣性權重， $0 \leq w \leq 1$ ；

- $v_{id}^{old}, x_{id}^{old}$: 在第 i 個粒子內, 第 d 個供給點目前的速度及位置;
 $v_{id}^{new}, x_{id}^{new}$: 在第 i 個粒子內, 第 d 個供給點更新後的速度及位置;
 p_{id} : 在第 i 個粒子的個體暫時最佳解內, 第 d 個供給點的位置;
 g_d : 在全體暫時最佳解內, 第 d 個供給點的位置;
 c_1, c_2 : 學習因子;
 $rand_1(), rand_2()$: 介於 0 和 1 之間的隨機亂數。

2.5 路徑集合更新策略

在每回合求解的過程中, 以可行解產生策略所形成之均衡網路, 在經由流量推擠及鄰近搜尋法改善後, 可能受到凹形節線特性的影響, 使部分較小流量的節線運送量, 被吸引至其他流量較大的節線上, 進而改變供需節點對的路徑及運送量, 並可能產生新的供需節點路徑。因此, 本研究發展路徑集合更新策略, 利用流量從需求點回推至供給點的方式, 將伸展樹逐一還原成供需節點對路徑, 並藉由供給點供給量、需求點需求量及伸展樹路徑流量三方面的流量限制, 以確保還原的路徑與流量解為此伸展樹之唯一一組路段流量解, 不會因為路徑回推之順序不同, 而還原出不同的路徑與流量解。此可避免形成演算法中不穩定之情況, 並可確認路徑集合內各路徑在伸展樹上之使用情況, 使路徑之機率值能依目標值之優劣有效調整, 以提升整體的求解績效。此外, 藉由此方式亦可在較佳的可行解伸展樹內, 搜尋出新的供需節點對路徑, 以更新各粒子之路徑集合, 進而增加演算法搜尋的廣度。此策略的具體步驟如下:

- 步驟 1. 以每回合可行解伸展樹為基礎, 並計算其目標值。
- 步驟 2. 紀錄各供給點供給量 S 、需求點需求量 D 及各節線流量 L 。
- 步驟 3. 第一階段, 依照第 2.3 節步驟 4.4、6.3 與 7 中記錄之各供需節點對運送量與路徑, 依序確認各供需節點路徑及流量, 以確定供需節點路徑是否為構成伸展樹之路徑。
- 3.1 依序選取供給點為 i 、需求點為 j 及路徑為 k 之供需節點對。若此供需節點對供給點供給量 $S_i > 0$ 及需求點需求量 $D_j > 0$, 則進行步驟 3.2; 否則, 選取下一組供需節點對。若無可選取之供需節點對, 則進行步驟 4。
- 3.2 依照步驟 3.1 選取之供需節點路徑 k , 搜尋其路徑上最小流量之節線 a 、其流量為 L_a 。若 $L_a > 0$, 則進行步驟 3.3; 否則, 進行步驟 3.1。
- 3.3 令供需節點對最小供給量為 $f = \min \{S_i, D_j, L_a\}$, 記錄供需點 i, j , 運送量 f 及路徑 k 為形成伸展樹之路徑, 並更新供給點供給量為 $S_i = S_i - f$ 、需求點需求量為 $D_j = D_j - f$ 及扣除此路徑上各節線 f 的流量。
- 步驟 4. 第二階段, 若網路中還有節線流量 $L > 0$, 則進行步驟 4.1; 否則, 進行步驟 5。
- 4.1 依序選取一供給點 i 之供給量 $S_i > 0$, 進行步驟 4.2。若所有供給點皆已被選取過, 則進行步驟 5。

- 4.2 依序選取一需求點 j 之需求量 $D_j > 0$ ，進行步驟 4.3。若所有需求點皆已被選取過，則回到步驟 4.1。
 - 4.3 判斷供需節點路徑集合中，以供給點 i 與需求點 j 為起迄點之所有路徑，是否存在一路徑且其各節線的流量皆大於零？若是，則紀錄此路徑上最小流量之節線 a 、其流量為 L_a ，並進行步驟 4.4；否則，回到步驟 4.2。
 - 4.4 令供需節點對最小供給量為 $f = \min \{S_i, D_j, L_a\}$ ，記錄供需點 i 、 j ，運送量 f 及路徑 k 為形成伸展樹之路徑，並更新供給點供給量為 $S_i = S_i - f$ 、需求點需求量為 $D_j = D_j - f$ 及扣除此路徑上各節線 f 的流量。判斷 S_i 是否大於 0？若是，則回到步驟 4.2；否則，回到步驟 4.1。
- 步驟 5. 判斷此伸展樹之目標值，是否較個體最佳解之目標值佳且網路中是否仍有節線流量 $L > 0$ 之情況？若是，則進行步驟 6；否則，結束路徑集合更新策略。
- 步驟 6. 第三階段，依照網路中剩餘節線流量搜尋出新的供需節點路徑。
- 6.1 依照目前各節線上之流量，給予各節線之一虛擬成本 C'_{ij} 。若節線上之流量大於零，則令虛擬成本 $C'_{ij} = 0$ ；若節線流量等於零，則令虛擬成本等於原本節線之通過成本 $C'_{ij} = C_{ij}$ 。進行步驟 6.2。
 - 6.2 選取供給量 $S_i > 0$ 之供給點 i ，依照步驟 6.1 給定之節線成本並以標籤修正法求解此供給點 i 到各需求點之最短路徑，進行步驟 6.3；若無可選取之供給點，則進行步驟 7。
 - 6.3 選取需求量 $D_j > 0$ 之需求點 j ，並依步驟 6.2 中以供給點 i 與需求點 j 為起迄點之路徑，判斷此路徑上最小流量 L_a (節線 a) 是否大於零？若是，則進行步驟 6.4；否則，選取另一需求點 j 之需求量 $D_j > 0$ 。
 - 6.4 令供需節點對最小供給量為 $f = \min \{S_i, D_j, L_a\}$ ，記錄供需點 i 、 j ，運送量 f 及路 k 徑為形成伸展樹之路徑，並更新供給點供給量 $S_i = S_i - f$ 、需求點需求量 $D_j = D_j - f$ 及扣除此路徑上各節線 f 流量。判斷供給點 i 之供給量 $S_i > 0$ ？若是，則回到步驟 6.3；否則，回到步驟 6.2。
- 步驟 7. 將步驟 6.4 紀錄之供需節點路徑，增加至路徑集合內，並針對此等路徑及對應之各粒子隨機產生一介於 -6 至 6 之數值，以作為此路徑在各粒子內之初始機率值。

2.6 機率值更新方式

二進位粒子群最佳化 (BPSO) 機率值更新策略的主要目的，為提供粒子群搜索時之方向導引，並藉由疊代求解的過程逐漸突顯出各變數的優劣差異，以提供各粒子進行下一回合搜尋時的方向。值得一提的是，以往在 BPSO 的機率更新作法上，節線數量會隨問題規模逐漸增加而大幅度的增加，使得節線機率值累積或降低的速度過慢，導致不易在設定的運算回合數內有效的突顯各節線間機率值之差異性，進而降低了求解績效。而連續型粒子群最佳化中所設定之慣性權重值 (α) 作法，能藉由慣性權重值控制收斂速度及增加搜尋範

圍，以避免陷入區域最佳解之情況，故本研究利用慣性權重值的作法，並參考各回合與前一回合可行解的適應值，以彈性的調整各回合伸展樹內各路徑的速度（機率值）。另外，由於設定學習因子之作法，可加快各節線速度的收斂，能在設定的運算回合內有效地導引粒子收斂至較佳的可行解區域，以提升求解績效，因此本研究針對凹形成本網路問題的特性，結合慣性權重值與設定學習因子兩種機率更新方式，發展與以往不同的機率值更新方式。其作法如下：將各粒子的路徑集合分成伸展樹與非伸展樹之兩種路徑集合。若為非伸展樹之路徑集合，則依全體與個體最佳解之各變數值，以式 (6) 進行機率值之更新。若為伸展樹之路徑集合，則針對此一回合所產生的新可行解伸展樹，先計算此可行解伸展樹的目標值 F_i^{new} ，再使路徑在更新速度時，除依全體與個體最佳解之各變數值外，並依新的目標值 F_i^{new} 與前一回合目標值 F_i^{old} 之比值，以式 (7) 進行機率值之更新。

$$v_{id}^{new} = v_{id}^{old} + c_3 \times rand_3() \times g_d + c_4 \times rand_4() \times p_{id} \quad (6)$$

$$v_{id}^{new} = \alpha \times v_{id}^{old} + c_3 \times rand_3() (g_d - x_{id}) + c_4 \times rand_4() (p_{id} - x_{id}) + (1 - \alpha) \times \frac{F_i^{old}}{F_i^{new}} \times v_{id}^{old} \quad (7)$$

其中，

i ：第 i 個粒子，即第 i 組可行解；

d ：第 d 個搜尋空間的維度，即表示第 d 條最短路徑；

α ：慣性權重值， $0 \leq \alpha \leq 1$ ；

v_{id}^{new} ：在第 i 個粒子內，第 d 條路徑更新後之速度；

v_{id}^{old} ：在第 i 個粒子內，第 d 條路徑目前之速度；

x_{id} ：在第 i 個粒子內，第 d 條路徑是否在伸展樹內（1：表示在伸展樹內；0：表示不在伸展樹內）；

p_{id} ：在第 i 個粒子的個體暫時最佳解內，第 d 條路徑是否在伸展樹內（1：表示在伸展樹內；0：表示不在伸展樹內）；

g_d ：在全體暫時最佳解內，第 d 條路徑是否在伸展樹內（1：表示在伸展樹內；0：表示不在伸展樹內）；

F_i^{old} ：第 i 組可行解目前的目標值；

F_i^{new} ：第 i 組可行解更新後的目標值；

c_3 、 c_4 ：學習因子；

$rand_3()$ 、 $rand_4()$ ：介於 0 和 1 之間的隨機亂數。

本研究在速度更新上引入連續型粒子群最佳化學習因子的概念，其中給予學習因子較高的數值，雖然能使粒子群快速地收斂至一可行解區域，但是當問題規模增加時，將快速地累積或降低速度，可能容易使粒子群體陷入區域最佳解，而導致整體求解績效不佳。因

此，本研究在速度範圍的限制方式上，以線性遞增的方式，逐漸增加速度的限制範圍。另外，為了避免範圍值過大，而導致速度陷入過高或過低的極端情況，本研究設定速度範圍之最大值為 $V_{\max} = 6$ ，使速度介於 $-6 \sim 6$ 之間（即機率值介於 $0.0025 \sim 0.9975$ 之間），以保留粒子搜尋時的隨機性。綜上所述，單一粒子速度範圍限制的作法如下：首先設定一起始速度值 V 、變動回合數 V_l 、數列長度 V_k 及速度最大值 $V_{\max}$ ，使速度範圍介於 $-V_{\max}$ 到 $V_{\max}$ 之間。之後在求解過程中，若運算回合數經過變動回合數 V_l ，而粒子之個體最佳解目標值仍無改善時，則使速度值 V 增加 Δ_v 值（速度增加幅度 $\Delta_v = \frac{V_{\max} - V}{V_k}$ ），再重複上述過程。

值得注意的是，當速度值 V 增加至 $V_{\max}$ 時，此後均以 $V = V_{\max}$ 進行求解運算，此時作法即與傳統 BPSO 之速度範圍限制的方式相同。

2.7 門檻值個體與全體最佳解更新策略

以往 PSO 在更新個體與全體最佳解的作法上，係當搜尋到的可行解目標值比個體或全體最佳解較佳時，才將此一搜尋到的可行解更新為個體或全體最佳解。本研究嘗試引入 TA 門檻值的觀念，修正個體與全體最佳解之更新方式。茲舉全體最佳解之更新作法如下：首先設定全體最佳解之起始門檻值 T^g 為一初始百分比，當搜尋到之可行解目標值小於全體最佳解 $\times (1 + T^g)$ 時，則更新全體最佳解。經過若干回合運算後，若運算回合經過所設定之變動回合數 (T_l^g) ，而全體之目標值無改善時，則降低起始門檻值 T^g 為 $(T^g - \Delta_g)$ （其中，門檻值下降率 $\Delta_g = \frac{T^g}{T_k^g}$ ， T_k^g 為數列長度），再重複上述過程，直到 T^g 為 0 後，以 $T^g = 0$ 進行全體最佳解的更新（此時即與傳統 PSO 的更新方式相同）。值得注意的是，若最終之全體最佳解比各回合之最佳解差，則以各回合最佳解作為求解結果。至於在個體最佳解的更新上，一樣設定初始門檻值 T^p 、變動回合數 T_l^p 及數列長度 T_k^p 三個參數，其餘更新步驟則與全體最佳解之步驟相同。本研究藉由此一更新方式，能避免快速陷入區域最佳解之情況，以提升演算法求解績效。

2.8 鄰近搜尋法之改善策略

本研究所發展之演算法為散布式搜尋演算法。在搜尋廣度方面，可廣泛的搜尋可行解區域，避免區域性搜尋的限制。然而在搜尋深度方面，因為粒子群最佳化演算法具有隨機性搜尋之特性，可能在尚未搜尋到區域最佳解時，即跳脫至另一可行解區域。因此，本研究在產生可行解後配合顏上堯等人^[26]所發展之鄰近搜尋法，以增加演算法之搜尋深度。此鄰近搜尋法之作法如下：首先以粒子搜尋後的解為目前解（為伸展樹），隨機選取非目前伸展樹的 e 條節線分別加入目前伸展樹中，並分別使用流量推擠法（Yan 與 Young^[44]）以消除迴圈，形成 e 個新的伸展樹，計算 e 個新的目標值。若 e 個目標值中最小值 Fe' 小於目前解的目標值，則更新目前解與目標值。繼續上述步驟，直到 Fe' 大於目前解的目標值，

則表示找到區域最佳解，結束鄰近搜尋步驟。

2.9 菁英策略 (elitist)

本研究引用遺傳演算法之菁英策略觀念並納入於本演算法中，其目的有二：一為保留較佳解不再重新搜尋，可節省運算時間；另一為保留機率值較高的路徑，可有效的將較佳的路徑保留在下一回合之路徑集合中，以形成較優良之可行解伸展樹，對於求解結果具有正向的作用。本研究提出兩種菁英策略如下：

1. 較佳解菁英保留策略：此策略之作法為設定一保留率（回合較佳解保留率， P_s ），將粒子群所搜尋出之可行解依目標值之優劣排序，並將數個較佳解保留至下一回合成為下回合的數組可行解。
2. 較佳路徑菁英保留策略：由於本研究問題具經濟規模的特性（當節線流量越大，單位成本愈低），因此我們將可行解伸展樹還原出來之供需節點路徑數乘以一比率（較佳路徑保留率， P_e ），並從此等構成可行解伸展樹之路徑集合中，保留機率值較大的數對供需節點路徑，至下一回合產生新的可行解時，不再重新搜尋，即依此供需節點對之供給點、需求點、路徑及供需量進行流量指派。

2.10 運算終止機制

一般演算法的運算終止機制會依各問題及求解演算法之特性不同而有所不同，其大致可分為下列三類：

1. 總運算時間：給定一固定時間 S_t ，當求解時間超過 S_t 時，即終止運算。
2. 總運算回合數：給定一固定回合數 S_r ，當求解回合數超過 S_r 時，即終止運算。
3. 未改善回合數：給定一固定容許回合數 S_a ，比較各回合求解之目標值，當暫時最佳目標值經過 S_a 回合皆未改善時，即終止運算。

上述三種方法可避免求解時間過長，而在運算終止時，係以目前所搜尋到最好的目標值，作為最終目標值。經過多次測試的經驗，本研究採用總運算回合數以設計演算法之終止機制。

三、測試分析

為了測試本研究演算法的求解績效，本節針對含凹形節線成本且具方向性之網路，利用電腦產生大量的網路資料以為測試。本研究參考 Yan 等人^[3]所發展的隨機網路產生器，重新產生 10、20、50、100、150 與 300 等節點數規模。其中規模 100、150 與 300 的網路各自配合 0.2、0.5、0.8 三個控制供需點密度的參數 D 值（以亂數產生供需點數量，使其供需點密度近似於 D 值），共設計 9 種網路，而規模 10、20 及 50 的網路則因節點數過少，

且大都能迅速找到最佳解，故在參數測試上予以省略，不做測試。各網路之節線數與網路密度如表 1 所示。

表 1 測試網路之節線數與網路密度

網路節點數	10	20	50	100	150	300
網路節線數	22	46	220	730	1117	2656
網路密度 (%)	22.0	11.5	8.8	7.3	5.0	2.95

本研究使用 C++ 語言撰寫並測試演算法，測試環境為 P4-3.2GHz 之個人電腦。此外，本研究參考顏上堯等人^[26]及 Yan 等人^[3]所發展之四種區域式搜尋演算法與基因演算法 (GA)，以測試比較本研究演算法的求解績效。本節架構如下：3.1 節測試演算法各系統參數對求解之影響；3.2 節比較 APSO、GA 與區域搜尋法之求解績效。

3.1 求解策略與參數測試分析

針對不同類型的問題，本研究演算法內所設定的參數可能會影響求解的品質。因此，經過多次測試的經驗，本研究先提出一組績效不錯之基本參數組一方案 1 (如表 2 所示)，再以方案 1 延伸、設計各種不同參數值範圍的參數組。本研究共提出 52 種方案，並針對不同網路規模及節點密度進行測試，以找出優良的系統參數組合。在測試各種求解參數過程中，本研究比較各方案所求得目標值與最優解目標值的比值，並以誤差百分比表示，其目的在評估各方案在不同網路下的測試績效 (誤差百分比越小表示該方案參數在該網路求解的效果越佳)。本研究針對各組參數方案測試十次，每次終止回合均設定為 500 回合，並以十次測試中最好的一次作為該方案的求解結果。

經本研究多次測試經驗可知，初始群體型態及鄰近搜尋法分別在凹形初始解比率為 0.6 與搜尋範圍為 200 時有較佳之結果。由於此等測試方式與 Yan 等人^[3]相似，為節省篇幅，故在此不加詳述說明。讀者若需更詳細之數據與分析，則可與作者聯繫。以下針對粒子群最佳化演算法中之相關參數作一分析探討，首先針對可行解產生策略中，供給點選擇方式進行測試並提出較佳的挑選方式，以作為本演算法的可行解產生方式，之後分別針對粒子群體個數、供給點指派順序策略、機率值 (速度) 更新公式、速度範圍限制策略、門檻值全體最佳解更新策略、菁英策略及演算法收斂情形等 8 個小節進行參數適用性分析。請注意，以下所有表格中之目標值皆以誤差百分比 (%) 表示 ($= (\text{該參數方案下所得之目標值} / \text{本研究各演算法在各參數方案中所得之最小目標值}) - 1$)，而運算時間單位則皆以秒計。另外，表 3 至表 5 之誤差百分比為在三種不同節點數及三種不同密度下測試結果之平均值，灰底的數字代表該方案中最佳策略的結果。而表 6 至表 14 中，則由於各網路規模的目標值誤差百分比及求解時間為在相同節點數及三種不同密度下測試結果之平均值，故在網路規模一欄中僅以節點數量表示，另灰底數字代表該測試參數下最佳方案的結果。

表 2 方案 1 之系統參數值

求解策略	參數名稱	方案 1	求解策略	參數名稱	方案 1
粒子群體個數	粒子數 (p)	10	門檻值個體 最佳解更新 策略	起始門檻值 (T^p)	0.005
初始群體型態	凹形初始解比率 (p_m)	0.6		數列長度 (T_k^p)	30
供給點選擇策略	慣性權重 (w)	0.8		未變動回合數 (T_1^p)	10
	全體最佳解學習因子 (c_1)	1	門檻值全體 最佳解更新 策略	起始門檻值 (T^g)	0.002
	個體最佳解學習因子 (c_2)	2		數列長度 (T_k^g)	30
機率值 (速度) 更新公式	慣性權重 (α)	0.9		未變動回合數 (T_1^g)	20
	全體最佳解學習因子 (c_3)	2	速度範圍 限制策略	起始速度範圍值 (V)	1
	個體最佳解學習因子 (c_4)	3		數列長度 (V_k)	10
較佳解菁英保留策略	較佳解保留率 (P_s)	0.8		變動回合數 (V_l)	30
較佳路徑菁英保留 策略	較佳路徑保留率 (P_e)	0.7	鄰近搜尋法	搜尋範圍	200

3.1.1 供給點選擇方式

本研究提出三種供給點選擇方式，分別為隨機選擇、依凹形成本特性選擇及供給點指派順序策略，測試結果分列於表 3 及表 4，測試時間則如表 5 所示。

表 3 供給點選擇方式平均目標值之比較

項 目	方 案				
	1	13	18	36	平均
隨機選取	1.51%	1.35%	1.35%	1.37%	1.40%
依凹形成本特性	1.53%	1.47%	1.36%	1.30%	1.42%
供給點指派順序策略	1.40%	1.37%	1.31%	1.31%	1.35%

由表 3 可知，在四組方案測試下，以各網路規模求解十次的平均目標值而言，三種供給點選擇方式，以供給點指派順序策略之作法普遍較佳，而隨機選擇與依凹形成本特性選擇則互有優劣。若以四組方案的平均值而言，則同樣以供給點選擇順序策略的挑選方式最佳，其平均值最低 (1.35%)，隨機選取次之 (1.40%)，依凹形成本特性選取最差 (1.42%)。

表 4 供給點選擇方式最佳目標值之比較

項 目	方 案				
	1	13	18	36	平均
隨機選取	0.94%	0.72%	0.78%	0.71%	0.79%
依凹形成本特性	0.96%	0.98%	0.70%	0.68%	0.83%
供給點指派順序策略	0.80%	0.65%	0.68%	0.72%	0.71%

表 5 供給點選擇方式測試時間之比較

項 目	方 案				
	1	13	18	36	平均
隨機選取	118.91	118.62	121.09	121.01	119.91
依凹形成本特性	119.32	120.42	121.62	121.99	120.84
供給點指派順序策略	114.12	119.15	120.09	121.10	118.62

由表 4 可知，若以各網路規模下之最佳目標值評估，則在四組方案測試下，供給點指派順序策略在方案 1、13 與 18 三個方案中皆為最優，而隨機選擇與依凹形成本特性選擇則互有優劣。若以四組方案的平均值評比，則以供給點指派順序策略具有最佳的求解結果，其平均值最低 (0.71%)，隨機選取次之，依凹形成本特性選取則最差。另外，在求解時間方面，由表 5 可知，三種供給點選擇方式的求解測試時間差距不大，顯示三種選擇方式對於求解時間並無明顯的影響。綜合上述，在平均目標值與最佳目標值的雙重評比下，發現皆以供給點指派順序策略的求解結果最佳，故本研究之後參數分析均以供給點指派順序策略作為測試基礎。

3.1.2 粒子群群體數量

本小節針對粒子群群體數量進行測試，其求解結果與求解時間如表 6 所示。由測試結果可知，求解時間與粒子群體數量成線性正相關，同時隨著網路規模越大，求解時間亦隨之大幅上升。在誤差百分比方面，以方案 3 的求解品質普遍較佳，而方案 1 與方案 2 則互有優劣。若以平均誤差百分比而言，則以方案 3 群體大小為 20 者較佳，其平均誤差百分比為 0.76%，方案 1 (0.80%) 次之，方案 2 (0.90%) 則最差。然而，雖然群體大小為 20 有較佳的求解結果，但所花費的求解時間約為方案 1 的 2 倍，且其平均誤差百分比僅比方案 1 少 0.04%，改善幅度甚有限，因此在求解時間與求解品質的雙重考量下，本研究在粒子群體大小上設為 10。

表 6 粒子數量測試結果

網路規模	項 目	方 案		
		1	2	3
		P = 10	P = 15	P = 20
100	誤差百分比	0.13%	0.01%	0.00%
	求 解 時 間	41.23	58.46	89.16
150	誤差百分比	0.47%	0.53%	0.46%
	求 解 時 間	75.73	112.16	164.54
300	誤差百分比	1.79%	2.15%	1.82%
	求 解 時 間	225.39	368.02	464.67
平均誤差百分比		0.80%	0.90%	0.76%
平均求解時間		114.12	179.55	239.46

3.1.3 供給點指派順序策略相關參數

本節針對供給點位置更新公式中， w 、 c_1 、 c_2 三個重要的參數進行測試分析，其結果分述如下：

(一) 參數 w 之測試結果

在供給點位置的更新公式中，參數 w 主要在控制粒子搜尋的範圍，並且能使粒子避免陷入區域最佳解。在以往 PSO 相關的文獻與本研究多次測試經驗中，發現其值不宜設定過低或超過 0.9，否則容易造成粒子群不易收斂至較佳的可行解區域，致使求解品質降低。本研究將參數 w 範圍設定在 0.6 ~ 0.9 之間以進行測試，所得結果如表 7 所示。

由表 7 可知，方案 24 ($w = 0.9$) 的求解結果最佳，其平均誤差百分比 (0.55%) 最小，並且隨著 w 值的降低，平均目標值有漸變差的趨勢，由此顯示較高的 w 值容易導引粒子群收斂至較佳的可行解空間，因此較易於改善求解品質。此外，在求解時間方面，由表 7 可知在 w 的變動下，各網路規模的求解時間及整體的平均求解時間均無太大的差異，顯示 w 值大小與求解時間的相關性不高。

二、參數 c_1 、 c_2 之測試結果

在以往的 PSO 相關文獻中，給予 c_1 較高的數值，能夠加快粒子群體的收斂速度，但亦相對的限制粒子在其他空間探索的可能性，而容易陷入區域最佳解；若給予 c_2 較高的數值，則可使粒子能有效的搜尋各可行解區域，但其收斂速度較慢。本研究將 c_1 、 c_2 範圍設定在 1 ~ 4 之間，共提出 7 種組合進行測試，所得的結果如表 8 所示。

表 7 參數 w 測試結果

網路規模	項 目	方 案			
		22	23	18	24
		$w = 0.6$	$w = 0.7$	$w = 0.8$	$w = 0.9$
100	誤差百分比	0.07%	0.06%	0.01%	0.06%
	求 解 時 間	44.60	44.27	43.78	44.87
150	誤差百分比	0.18%	0.41%	0.38%	0.35%
	求 解 時 間	80.99	81.42	80.52	80.48
300	誤差百分比	1.69%	1.65%	1.65%	1.25%
	求 解 時 間	244.92	236.22	235.98	242.48
平均誤差百分比		0.65%	0.71%	0.68%	0.55%
平均求解時間		123.50	120.64	120.09	122.61

表 8 參數 c_1 、 c_2 測試結果

網路規模	項 目	方案及參數 $c_1 \times c_2$						
		18	25	26	27	28	29	30
		1×2	2×2	3×2	4×2	1×1	1×3	1×4
100	誤差百分比	0.01%	0.06%	0.06%	0.06%	0.06%	0.06%	0.00%
	求 解 時 間	43.78	44.28	44.93	44.29	44.22	44.80	43.97
150	誤差百分比	0.38%	0.27%	0.35%	0.35%	0.45%	0.44%	0.50%
	求 解 時 間	80.52	81.15	81.00	81.46	79.69	81.51	80.51
300	誤差百分比	1.65%	2.19%	1.87%	1.04%	1.67%	1.75%	1.74%
	求 解 時 間	235.98	239.18	240.47	237.23	241.02	237.96	239.29
平均誤差百分比		0.68%	0.84%	0.76%	0.48%	0.73%	0.75%	0.74%
平均求解時間		120.09	121.54	122.13	120.99	121.64	121.42	121.26

在固定個體最佳解學習因子 $c_2 = 2$ 下，如表 8 中方案 18、25、26 及 27 四個方案，可知以方案 27 ($c_1 = 4$) 的求解結果最佳，其平均誤差百分比為 0.48%，而以方案 25 ($c_1 = 2$) 之平均誤差百分比 0.84% 為最差；而在固定全體最佳解學習因子 $c_1 = 1$ 下，如表 8 中方案 18、28、29、30 等四個方案，可知以方案 18 ($c_2 = 2$) 的求解結果最佳，其平均誤差百分比為 0.68%，而以方案 29 ($c_2 = 3$) 之平均誤差百分比 0.75% 為最差。就整體 c_1 與 c_2 組合而言，

以方案 27 ($c_1 = 4$ 及 $c_2 = 2$) 的求解結果最佳，其平均誤差百分比為 0.48%，而以方案 25 ($c_1 = 2$ 及 $c_2 = 2$) 之平均誤差百分比 0.84% 為最差。此外，在求解時間方面， c_1 、 c_2 值大小與求解時間並無明顯的相關性。

3.1.4 機率值 (速度) 更新公式相關參數

本節針對速度更新公式中，三個重要的參數 α 、 c_3 、 c_4 進行測試分析，其結果分述如下：

(一) 參數 α 之測試結果

參數 α 為慣性權重，與以往的 PSO 中慣性權重不同，其主要目的在參考該回合粒子的適應值，以彈性的調整該回合該伸展樹 (粒子) 之路徑機率，以增加粒子搜索各可行解區域的可能性。在本研究多次測試的經驗中，發現 α 值不宜設的過低或超過 0.9，過低的 α 值易導致路徑的機率值受到各回合適應值之影響，使路徑機率值無法隨著個體與全體最佳解調整。因此，本研究將 α 值設定在 0.6 ~ 0.9 之間，共提出 4 種組合進行測試，結果如表 9 所示。

由表 9 可知，方案 9 ($\alpha = 0.8$) 的求解結果最佳，其平均誤差百分比 (0.78%) 最小。此外，在求解時間方面， α 值大小與求解時間無明顯的相關性。

表 9 參數 α 測試結果

網路規模	項目	方 案			
		7	8	9	1
		$\alpha = 0.6$	$\alpha = 0.7$	$\alpha = 0.8$	$\alpha = 0.9$
100	誤差百分比	0.08%	0.06%	0.13%	0.13%
	求 解 時 間	44.10	44.46	44.35	41.23
150	誤差百分比	0.25%	0.39%	0.52%	0.47%
	求 解 時 間	81.15	81.18	81.16	75.73
300	誤差百分比	2.18%	2.34%	1.68%	1.79%
	求 解 時 間	236.10	233.22	232.92	225.39
平均誤差百分比		0.84%	0.93%	0.78%	0.80%
平均求解時間		120.45	119.62	119.48	114.12

(二) 參數 c_3 、 c_4 之測試結果

在以往的 PSO 相關文獻中，給予 c_3 較高的數值，能夠加快粒子群體的收斂速度，但

亦相對的限制粒子在其他空間探索的可能性，而容易陷入區域最佳解；若給予 c_4 較高的數值，則可使粒子能有效的搜尋各可行解區域，但其收斂速度較慢。本研究將參數 c_3 、 c_4 範圍設定在 1~4 之間，共提出 7 種組合以進行測試，結果如表 10 所示。

表 10 參數 c_3 、 c_4 測試結果

網路規模	項目	方案及參數 $c_3 \times c_4$						
		10	9	11	12	13	14	15
		1×3	2×3	3×3	4×3	2×1	2×2	2×4
100	誤差百分比	0.13%	0.13%	0.06%	0.07%	0.13%	0.06%	0.06%
	求解時間	43.76	44.35	43.78	43.84	43.96	43.97	43.13
150	誤差百分比	0.56%	0.52%	0.26%	0.53%	0.38%	0.39%	0.57%
	求解時間	80.55	81.16	79.73	79.14	80.48	80.71	79.31
300	誤差百分比	2.12%	1.68%	2.60%	2.16%	1.46%	2.62%	1.93%
	求解時間	234.14	232.92	231.99	227.35	233.02	233.97	232.80
平均誤差百分比		0.94%	0.78%	0.97%	0.92%	0.66%	1.02%	0.85%
平均求解時間		119.48	119.48	118.5	116.78	119.15	119.55	118.41

在固定個體最佳解學習因子 $c_4 = 3$ 下，如表 10 中方案 9、10、11、12 等四個方案，以方案 9 ($c_3 = 2$) 的求解結果最佳，其平均誤差百分比為 0.78%，而以方案 11 ($c_3 = 3$) 最差，平均誤差百分比為 0.97%。在固定全體最佳解學習因子 $c_3 = 2$ 下，如表 10 中方案 9、13、14、15 等四個方案，以方案 13 ($c_4 = 1$) 的求解結果最佳，其平均誤差百分比為 0.66%，而以方案 14 ($c_4 = 2$) 最差，平均誤差百分比為 1.02%。就 c_3 與 c_4 組合而言，以方案 13 ($c_3 = 2$ 及 $c_4 = 1$) 的求解結果最佳，其平均誤差百分比為 0.66%，而以方案 14 ($c_3 = 2$ 及 $c_4 = 2$) 最差，平均誤差百分比為 1.02%。此外，在求解時間方面，表 10 顯示 c_3 、 c_4 值大小與求解時間並無明顯的相關性。

3.1.5 速度範圍限制策略相關參數

本研究為避免速度累積或減少過快，而導致速度陷入過高或過低的極端情況，特發展與 BPSO 不同之方法，即透過速度範圍隨著運算回合數的增加而增加的方式，以控制速度累積或減少的變化程度，藉以增加粒子的搜尋空間。結果如表 11 所示。

由表 11 可知，以方案 20 ($V = 1$ 、 $V_k = 10$ 及 $V_l = 10$) 有最佳的求解結果，其平均誤差百分比為 0.59%。由方案 13、16 及 17 三個方案的測試結果可知，當起始速度範圍 V 值增加時，其平均誤差百分比亦有增加之趨勢，表示各路徑之速度在初始求解時就有較大的差

異，導致有些路徑因為機率值較高則容易被挑選較多次；反之，機率值較低的路徑則極不容易被挑選，進而致使搜尋空間受到限制，所以較高起始速度範圍值的求解品質較差。至於在數列長度及變動回合數方面，由方案 13、18、19、20 及 21 的求解結果可知，隨著 V_k 及 V_l 值之增加，其平均誤差百分比並無顯著之改變趨勢。此外，在求解時間方面， V 、 V_k 及 V_l 三個參數值大小與求解時間無明顯的相關性。

表 11 速度範圍限制相關參數測試結果

網路規模	項目	方案及參數 $V \times V_k \times V_l$						
		13	16	17	18	19	20	21
		1×10×30	2×10×30	3×10×30	1×5×30	1×15×30	1×10×10	1×10×20
100	誤差百分比	0.13%	0.00%	0.13%	0.01%	0.06%	0.13%	0.06%
	求解時間	43.96	44.20	44.33	43.78	43.63	44.27	44.37
150	誤差百分比	0.38%	0.42%	0.46%	0.38%	0.60%	0.21%	0.42%
	求解時間	80.48	80.66	81.57	80.52	79.66	80.37	80.67
300	誤差百分比	1.46%	1.62%	1.90%	1.65%	2.01%	1.44%	1.83%
	求解時間	233.02	235.38	235.43	235.98	235.78	243.88	241.20
平均誤差百分比		0.66%	0.68%	0.83%	0.68%	0.89%	0.59%	0.77%
平均求解時間		119.15	120.08	120.44	120.09	119.69	122.84	122.08

3.1.6 門檻值全體最佳解更新策略相關參數

本研究參考以往門檻值接受法 (Yan 與 Luo^[2])，採用線性遞減數列以為門檻值遞減方式。本研究經事先多次測試發現，起始門檻值 T^g 不宜設定太高，故將此參數的範圍設在 0.2% ~ 0.5% 之間，而數列長度 T_k^g 及變動回合數 T_l^g 則分別設定在 10 ~ 30 以及 10 ~ 20 之間，會有較佳之結果。為節省篇幅，此小節僅說明全體最佳解更新策略中各參數的測試結果，如表 12 所示。至於個體最佳解更新策略之結果係以方案 38 ($T^p = 0.002$ 、 $T_k^p = 30$ 及 $T_l^p = 10$) 最佳，其平均誤差百分比為 1.79%，為節省篇幅，故不在此詳述。

在起始門檻值方面，由方案 30、31、32 及 33 的測試結果可知，以方案 30 ($T^g = 0.002$) 的求解結果最佳，其平均誤差百分比為 0.74%，並且隨著起始門檻值的增加，平均誤差百分比有變差之趨勢，表示不宜設定較高的初始門檻值，以避免粒子群體過於廣泛的搜尋可行解區域，而無法有效的導引粒子群集中至較佳的可行解區域。在數列長度及未變動回合數方面，由方案 30、34、35、36 及 37 的求解結果可知，以方案 35 ($T_k^g = 20$ 及 $T_l^g = 20$) 的求解結果最佳，其平均誤差百分比為 0.60%，而以方案 37 ($T_k^g = 30$ 及 $T_l^g = 15$) 最差，其平均誤差百分比為 0.85%。以 T^g 、 T_k^g 及 T_l^g 三個參數組合而言，方案 35 的求解品質最佳，

其平均誤差百分比為 0.60%。此外，在求解時間方面， T^g 、 T_k^g 及 T_I^g 三個參數值大小與求解時間無明顯的相關性。

表 12 門檻值全體最佳解相關參數測試結果

網路規模	項 目	方案及參數 $T^g \times T_k^g \times T_I^g$							
		30	31	32	33	34	35	36	37
		0.002×30×20	0.003×30×20	0.004×30×20	0.005×30×20	0.002×10×20	0.002×20×20	0.002×30×10	0.002×30×15
100	誤差百分比	0.00%	0.13%	0.06%	0.00%	0.06%	0.06%	0.06%	0.13%
	求 解 時 間	43.97	43.69	44.05	44.56	44.42	43.79	44.32	44.05
150	誤差百分比	0.50%	0.39%	0.48%	0.45%	0.42%	0.51%	0.26%	0.50%
	求 解 時 間	80.51	80.80	80.10	81.98	80.04	80.82	79.03	80.35
300	誤差百分比	1.74%	1.81%	1.72%	2.05%	1.83%	1.24%	1.83%	1.93%
	求 解 時 間	239.29	235.93	240.17	235.25	242.33	236.65	239.95	236.96
平均誤差百分比		0.74%	0.78%	0.75%	0.83%	0.77%	0.60%	0.72%	0.85%
平均求解時間		121.26	120.14	121.44	120.60	122.26	120.42	121.10	120.45

3.1.7 菁英策略相關參數

本研究提出兩種菁英策略，分別為較佳解菁英保留策略及較佳路徑菁英保留策略，進行測試，結果如下：

(一)較佳解菁英保留策略

本研究經事先多次測試發現，過高或過低的保留率將使得演算法無法有效地改善下一回合之解，故將此參數的範圍設在 0.3% ~ 0.7% 之間，測試結果如表 13 所示。

由表 13 可知，各網路問題的誤差百分比隨參數 P_s 值遞增而有明顯降低的趨勢，且較高的 P_s 值能有效的減少粒子搜尋數量，故能有效的減少求解時間。例如當 P_s 值等於 0.7 時 (即將上回合中前 70% 較佳的可行解保留至下回合成為部分的可行解)，其平均誤差百分比 (0.49%) 最低，且求解時間 (121.75 秒) 最短；而當 P_s 值等於 0.3 時，其平均誤差百分比 (0.76%) 最大，求解時間亦最長 (183.04 秒)。

(二)較佳路徑菁英保留策略

經事先多次的測試，本研究發現較佳路徑保留率 P_e 的設定值不宜過低或超過 0.9，否則會不易集中較佳路徑群，而降低求解的品質。因此本研究將 P_e 範圍設定在 0.6 ~ 0.9 之間，進行測試，結果如表 14 所示。

表 13 較佳解保留率測試結果

網路規模	項 目	方 案		
		45	46	38
		$P_s = 0.3$	$P_s = 0.5$	$P_s = 0.7$
100	誤差百分比	0.06%	0.00%	0.06%
	求 解 時 間	60.38	52.97	43.99
150	誤差百分比	0.42%	0.33%	0.27%
	求 解 時 間	117.36	99.44	81.00
300	誤差百分比	1.80%	1.52%	1.15%
	求 解 時 間	371.39	313.01	240.25
平均誤差百分比		0.76%	0.62%	0.49%
平均求解時間		183.04	155.14	121.75

表 14 較佳路徑保留率測試結果

網路規模	項 目	方 案			
		47	48	38	49
		$P_e = 0.6$	$P_e = .7$	$P_e = .8$	$P_e = .9$
100	誤差百分比	0.06%	0.06%	0.06%	0.39%
	求 解 時 間	49.64	48.28	43.99	37.35
150	誤差百分比	0.50%	0.19%	0.27%	0.60%
	求 解 時 間	90.61	87.46	81.00	68.60
300	誤差百分比	2.15%	2.03%	1.15%	2.24%
	求 解 時 間	278.66	268.55	240.25	204.39
平均誤差百分比		0.90%	0.76%	0.49%	1.08%
平均求解時間		139.64	134.76	121.75	103.45

由表 14 可知，方案 38 ($P_e = 0.8$) 的求解最佳，平均誤差百分比 (0.49%) 最小，方案 48 ($P_e = 0.7$) 與方案 47 ($P_e = 0.6$) 次之，而方案 49 ($P_e = 0.9$) 最差。此外，在求解時間方面，隨著 P_e 值增加，求解時間呈現遞減的情形，表示設定較高的 P_e 值可有效的節省運算時間。

3.1.8 收斂趨勢

PSO 的求解品質與運算回合數有關，但是過多的運算回合數則難以再改善求解品質且造成求解的時間過長，因此經多次的測試後，本研究設定運算終止回合數為 500 回合。為評估本演算法收斂之情況，本研究測試方案中每一回合皆輸出兩種數值，第一為回合中的最佳目標值 f_g ；第二為回合中的平均目標值 $\bar{f}$ ，並以與本研究在所有測試方案所找到最佳解 F_g 相比，以討論在不同規模網路下的收斂情況。為節省篇幅，以下僅針對方案 1 之三種不同網路與供需節點密度為例，說明 f_g 、 $\bar{f}$ 、 F_g 與運算回合的關係，如圖 2、圖 3 與圖 4 所示。至於其他方案的結果類似，不再贅述。

由圖 2、3 及 4 可知，小型網路問題的收斂速度最快、求解效果亦最佳，其最佳目標值 f_g 約在 20 回合前已快速下降，其後下降變得較為緩慢，約到 360 回合後與 F_g 線重疊。中型路網的收斂速度次之，其最佳目標值 f_g 約在 25 回合前已快速下降，其後則呈現較為

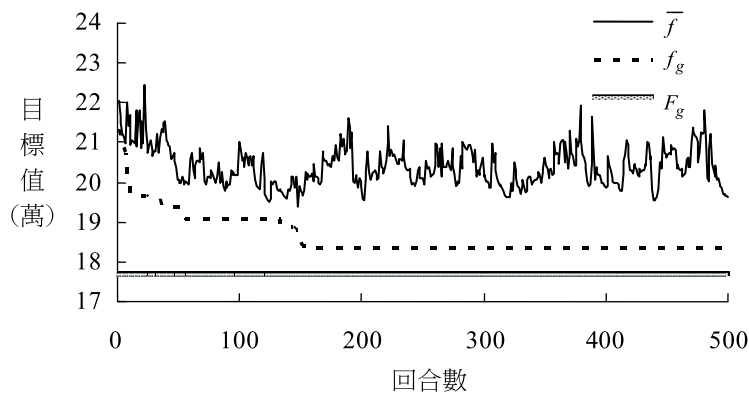


圖 2 大路網高密度 (300_08) 的收斂情形

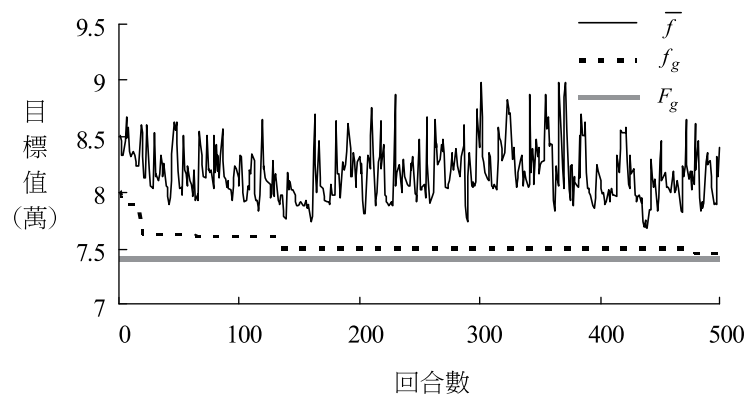


圖 3 中路網中密度 (150_05) 的收斂情形

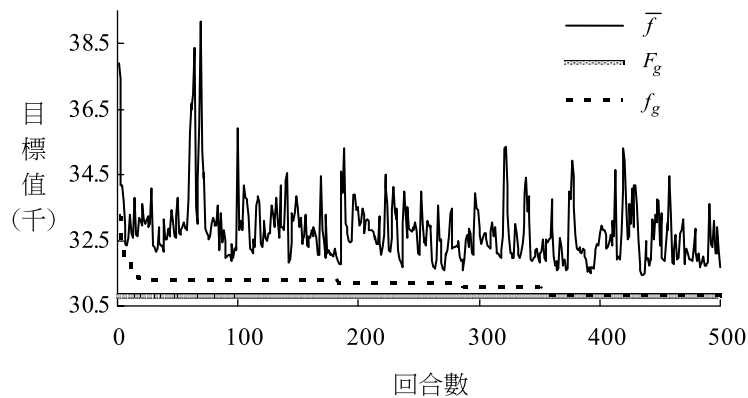


圖 4 小路網低密度 (100_02) 的收斂情形

緩慢的下降趨勢，變動幅度較低，在到達約 475 回合時趨近最佳解 F_g 。至於大型網路的收斂速度則最差，其最佳目標值 f_g 約在 100 回合前明顯下降，約到 160 回合後，即停止收斂，其與 F_g 線仍有一段差距。另外，本演算法在平均目標值 $\bar{f}$ 方面皆會呈現上下大幅度的震盪，其主要原因為本研究可行解的產生方式上，皆重新確認供需節點對與運送量，再從路徑集合內依路徑之機率值挑選路徑以進行運送，並以流量推擠法 (Yan 與 Young^[44]) 形成新的可行解伸展樹，因此各回合所求得的伸展樹，常與前一回合的可行解伸展樹有較大差異，因此目標值的變動幅度亦相對較大。

3.2 APSO、GA 與各區域搜尋法之求解績效比較

在分析探討 APSO 的求解策略與適用參數後，本研究進一步測試比較 APSO 與 GA 及各區域搜尋法之求解績效。本研究參考 Yan 等人^[3]及顏上堯等人^[26]針對本問題所發展的全域與區域搜尋演算法，其中全域演算法為遺傳演算法 (GA)，區域搜尋法則是包括了門檻值接受法 (TA)、大洪水法 (GDA)、結合禁制搜尋策略門檻值接受法 (TTA)、結合禁制搜尋策略大洪水法 (TGDA) 等四種；並以此五種演算法跟本研究之 APSO 作一求解績效之比較。另外，在測試的網路方面，本研究參考 Yan 等人^[3]所發展的隨機網路產生器，針對本研究重新產生 14 個具方向性的網路，並根據實務中演算法之應用方式，對各參數方案均測試十次，再以十次中最佳的一次作為其求解結果，以相同的基準上進行比較。

在各演算法參數部分，本研究 APSO 共有粒子數、 P_m 、 α 、 c_1 、 c_2 、 w 、 c_3 、 c_4 、 V 、 V_k 、 V_l 、 T^g 、 T_k^g 、 T_l^g 、 T^p 、 T_k^p 、 T_l^p 、較佳解保留率、較佳路徑保留率與鄰近搜尋率等 20 種參數，以在 3.1 節 52 組方案中測試結果最佳的三組方案 27、51 與 52 進行測試。而 GA 共有群體大小、優良解複製率、加入群體率、直接搜尋率、突變率、交換節線率、搜尋範圍與輪盤區分組等 8 種參數，並根據 Yan 等人所規劃之 26 組 GA 求解參數，經本研究多次測試後，提出較佳之 3 組參數以進行測試；而各區域搜尋法共有起始門檻值、門檻值數列長度、候選節線數、未改善次數、回合數與禁制串列長度等 6 種參數，參考 Yan 等人^[3]

及顏上堯等人^[26]對各演算法所設定的參數，測試結果如表 15 所示。

表 15 各演算法求解結果

演算法	APSO			GA			TA	TTA	GDA	TGDA	目前最佳 求解方法—方案
方案	27	51	52	GA-1	GA-2	GA-3					
10_05	0.00%	0.00%	0.00%	0.00%	0.00%	0.00%	0.00%	0.00%	0.00%	0.00%	APSO-52 等*
20_05	0.00%	0.00%	0.00%	0.00%	0.00%	0.00%	0.20%	0.20%	0.20%	0.20%	APSO-52 等*
50_02	0.00%	0.00%	0.00%	0.00%	0.00%	0.00%	0.00%	0.00%	0.00%	0.00%	APSO-52 等*
50_05	0.00%	0.00%	0.00%	0.00%	0.00%	0.00%	3.50%	0.00%	4.10%	3.50%	APSO-52 等*
50_08	0.00%	0.00%	0.00%	0.00%	0.00%	0.00%	1.80%	3.00%	0.00%	1.80%	APSO-52 等*
100_02	0.00%	0.00%	0.00%	0.96%	0.22%	1.41%	8.03%	5.44%	1.02%	8.03%	APSO-52 等*
100_05	0.19%	0.18%	0.00%	1.52%	0.19%	0.93%	6.11%	8.31%	4.50%	5.91%	APSO-52 等*
100_08	0.00%	0.00%	0.00%	0.18%	0.96%	1.21%	10.00%	9.50%	0.80%	7.80%	APSO-52 等*
150_02	0.00%	0.00%	0.00%	1.16%	1.16%	0.91%	10.20%	8.70%	8.60%	8.70%	APSO-52 等*
150_05	0.65%	0.44%	0.27%	0.22%	1.54%	2.80%	12.87%	10.07%	7.76%	9.96%	APSO-48
150_08	0.41%	0.61%	0.59%	0.72%	1.17%	0.59%	3.59%	6.60%	5.59%	3.89%	APSO-20
300_02	0.51%	0.65%	0.21%	2.43%	2.50%	0.91%	1.39%	3.46%	3.36%	5.63%	APSO-5
300_05	0.36%	1.37%	0.98%	1.57%	1.24%	1.76%	8.75%	4.76%	1.06%	5.76%	APSO-24
300_08	2.23%	1.77%	2.72%	1.15%	1.09%	0.75%	3.90%	2.71%	7.69%	5.09%	APSO-2
平均	0.31%	0.36%	0.34%	0.71%	0.72%	0.81%	5.02%	4.48%	3.19%	4.73%	
平均時間	80.43	112.25	114.00	303.00	208.80	435.40	23.40	35.16	34.03	34.04	

註 1：灰底數字代表該網路規模下，最佳的求解演算法（暨參數方案）之結果。

註 2：*表示多個求解方案的目標值皆等於目前最佳解。

註 3：各測試例之題目及成果已置於網路上，網址為

http://www.cv.ncu.edu.tw/docs/Concave_Cost_Network_Problem_Instances.rar

經人工運算求證，發現 APSO 及 GA 均可找到 10_05、20_05 兩個小型網路之最佳解。區域搜尋法則在 20_05 的網路測試中，無法有效跳脫區域最佳解，故求解結果皆有 0.2% 的誤差，此可知全域式搜尋演算法 APSO 及 GA 有較佳的搜尋求解機制。整體而言，在目標值誤差百分比平均值方面，APSO 最佳，其值介於 0.31% ~ 0.36% 之間，GA 次之，其值介於 0.71% ~ 0.81% 之間，四個區域搜尋法最差，其值介於 3.19% ~ 5.02% 之間。另外，在求解時間方面，區域搜尋法之平均求解時間（介於 23 ~ 36 秒）明顯比 APSO 及 GA 等全域搜尋法少，而 APSO（介於 80 ~ 114 秒）又明顯較 GA（介於 208 ~ 436 秒）少。因此綜合求解目標值及求解時間，本研究 APSO 優於 GA。在求解時間不是很急迫下（如求解一般規

劃性問題)，APSO 亦明顯優於 TA、GDA、TTA、TGDA 等區域搜尋法。另外，網路規模 10_05 至 150_02 下，APSO 及其他多種求解方案皆可搜尋到目前最佳解，然而當網路規模大於 150_05 之後，其目前最佳解皆由 APSO 求得，表示在搜尋最佳解的能力方面，APSO 明顯優於 GA 以及 TA、GDA、TTA、TGDA 等區域搜尋法。值得一提的是，為使演算時間公平起見，本研究曾嘗試延長區域搜尋法之運算時間至與 APSO 相同，但經多次測試發現，區域搜尋法在求解一段時間後，即無法繼續改善其解，因此原來的終止機制不會違反測試的公平性。

四、結論與建議

本研究以粒子群最佳化演算法的演算觀念為基礎，再輔以 GA、TA 與凹形成本網路啟發解法等相關技術，發展以路徑為基礎之類粒子群最佳化演算法 APSO，以求解實務上常見的含凹形節線成本最小成本轉運問題。測試結果顯示，APSO 在各網路規模中所找到之最佳解，皆較 TA、GDA、TTA、TGDA 等區域搜尋法為佳，且絕大部分較 GA 為佳；而在求解時間方面，APSO 明顯地比 GA 短，而二者皆較區域搜尋法為長。整體而言，在運算時間的差額不太敏感的問題下，APSO 的確優於 GA 及 TA、GDA、TTA、TGDA 等區域搜尋法。值得一提的是，本研究假設目標函數為 $\sum_{ij \in A} c_{ij} \sqrt{x_{ij}}$ 類型的凹形成本函數，對於

其他形式之凹形成本函數，應可以類似本研究演算法流程設計適合的演算法，以進行求解，而具體的演算法設計與適用之系統參數設定，可在未來繼續探討。另外，本研究係根據實務中演算法之應用方式，以多次測試中挑選最佳結果進行求解方案之評估，未來可進一步執行多次的測試後，並以平均值 t 檢定方式，檢驗求解方案的平均績效是否有顯著差異，以評估各求解方案的平均績效。最後，本研究係以路徑為基礎發展類粒子群最佳化演算法，未來可嘗試以節線作為基礎發展混合式全域搜尋法，以有效的求解本研究問題或其他組合最佳化問題。

參考文獻

1. Yan, S. and Luo, S. C., "A Tabu Search-Based Algorithm for Concave Cost Transportation Network Problems", *Journal of the Chinese Institute of Engineers*, Vol. 21, 1998, pp. 327-335.
2. Yan, S. and Luo, S. C., "Probabilistic Local Search Algorithms for Concave Cost Transportation Network Problems", *European Journal of Operational Research*, Vol.117, 1999, pp. 511-521.
3. Yan, S., Juang, D. H., Chen, C. R., and Lai, W. S., "Global and Local Search Algorithms for Concave Cost Transshipment Problems", *Journal of Global Optimization*, Vol. 33, No. 1, 2005, pp. 123-156.

4. Ahuja, R. K., Magnanti, T. L., and Orlin, J. B., *Network Flows, Theory, Algorithms, and Applications*, Prentice Hall, Englewood Cliffs, 1993.
5. Garey, M. R. and Johnson, D. S., *Computers and Intractability: A Guide to the Theory of NP-Completeness*, WH Freeman and Co., San Francisco, 1979.
6. Zangwill, W. I., "Minimum Concave Cost Flows in Certain Networks", *Management Science*, Vol. 14, 1968, pp. 429-450.
7. Rech, P. and Barton, L. G., "A Non-Convex Transportation Algorithm", *Applications of Mathematical Programming Techniques*, E. M. Beale, ed., 1970.
8. Gallo, G. and Sandi, C., "Adjacent Extreme Flows and Application to Min Concave Cost Flow Problems", *Networks*, Vol. 9, 1979, pp. 95-121.
9. Gallo, G., Sandi C., and Sadini, C., "An Algorithm for the Min Concave Cost Flow Problem", *European Journal of Operation Research*, Vol. 4, 1980, pp. 248-255.
10. Jordan, W. C., "Scale Economies on Multi-Commodity Networks", GMR-5579, Operating Systems Research Dept., GM Research Laboratories, 1986.
11. Blumenfeld, D. E., Burns, L. D., Diltz, J. D., and Daganzo, C. F., "Analyzing Trade-offs Between Transportation, Inventory, and Production Costs on Freight Network", *Transportation Research*, Vol. 19B, 1985, pp. 361-380.
12. Thach, P. T., "A Decomposition Method Using a Pricing Mechanism for Min Concave Cost Flow Problems with a Hierarchical Structure", *Mathematical Programming*, Vol. 53, 1992, pp. 339-359.
13. Guisewite, G. M. and Pardalos, P. M., "A Polynomial Time Solvable Concave Network Flow Problems", *Networks*, Vol. 23, 1993, pp. 143-147.
14. Yaged, B., "Minimum Cost Routing for Static Network Models", *Networks*, Vol. 1, 1971, pp. 139-172.
15. Larsson, T., Migdalas, A., and Ronnqvist, M., "A Lagrangian Heuristic for the Capacitated Concave Minimum Cost Network Flow Problem", *European Journal of Operational Research*, Vol. 78, 1994, pp. 116-129.
16. Balakrishnan, A. and Graves S. C., "A Composite Algorithm for a Concave-Cost Network Flow Problem", *Networks*, Vol. 19, 1989, pp. 175-202.
17. Amiri, A. and Pirkul, H., "New Formulation and Relaxation to Solve a Concave Cost Network Flow Problem", *Journal of the Operational Research Society*, Vol. 48, 1997, pp. 278-287.
18. Nourie, F. J. and Guder, F., "A Restricted-Entry Method for a Transportation Problem with Piecewise-Linear Concave Cost", *Computer & Operations Research*, Vol. 21, 1994, pp. 723-733.
19. Dukwon, K. and Panos, M., "Dynamic Slope Scaling and Trust Interval Techniques for Solving Concave Piecewise Linear Network Flow Problems", *Networks*, Vol. 35, 2000, pp. 216-222.
20. Suwan, R. and Sawased, T., "Link Capacity Assignment in Packet-Switched Networks: The Case of Piecewise Linear Concave Cost Function", *IEICE Trans. Commun.*, Vol. E82-B, No. 10, 1999.
21. Kuhn, H. W. and Baumol, W. J., "An Approximate Algorithm for the Fixed-Charge Transportation Problem", *Naval Res. Logistics Quarterly*, Vol. 9, 1962, pp. 1-16.

22. Kirkpatrick, S., Gelatt, C. D., and Vecchi, M. P., "Optimization by Simulated Annealing", *Science*, Vol. 220, 1983, pp. 671-680.
23. Glover, F., "Tabu Search, Part I", *ORSA Journal on Computing*, Vol. 1, No. 3, 1989, pp.190-206.
24. Glover, F. and Laguna, M., *Tabu Search*, Kluwer Academic Publishers, Massachusetts, 1997.
25. Dueck, G. and Scheuer, T., "Threshold Accepting: A General Purpose Optimization Algorithm Appearing Superior to Simulated Annealing", *Journal of Computational Physics*, Vol. 90, 1990, pp.161-175.
26. 顏上堯、陳建榮、湯慶輝，「含凹形節線成本最小成本轉運問題鄰近搜尋法之研究」，**運輸計劃季刊**，第三十三卷，第二期，民國九十三年，頁 277-306。
27. Goldberg, D. E., *Genetic Algorithms in Search, Optimization, and Machine Learning*, Addison-Wesley, Reading MA, 1989.
28. Eberhart, R. C. and Kennedy, J., "Particle Swarm Optimization", *Proceedings of IEEE International Conference on Neural Networks*, Vol. IV, 1995, pp.1942-1948.
29. Eberhart, R. C. and Kennedy, J. "A New Optimizer Using Particle Swarm Theory", *Proceedings of the Sixth International Symposium on Micro Machine and Human Science*, IEEE Service Center, Piscataway, NJ, Nagoya, Japan, 1995, pp. 39-43.
30. Boyd, R. and Richerson, P. J., *Culture and the Evolutionary Process*, the University of Chicago Press, Chicago, 1985.
31. Reynolds, C. "Flocks, Herds and Schools: A Distributed Behavioral Model", *Computer Graphics*, Vol. 21, 1987, pp. 25-34.
32. Kennedy, J. and Eberhart, R. C., "A Discrete Binary Version of the Particle Swarm Optimization", *Proceedings of IEEE International Conference on Neural Networks*, Vol. V, 1997, pp. 4104-4108.
33. Shi, Y. and Eberhart, R. C. "A Modified Particle Swarm Optimizer", *Proceedings of the IEEE International Conference on Evolutionary Computation*, IEEE Press, Piscataway, NJ, 1998, pp. 69-73.
34. Shi, Y. and Eberhart, R. C. "Particle Swarm Optimization: Development, Applications, and Resources", *Proc. Congress on Evolutionary Computation 2001*, IEEE Service Center, Piscataway, NJ, Seoul, Korea, 2001.
35. Shi, Y., "Particle Swarm Optimization", *IEEE Connections*, Vol. 2, 2004, pp. 8-13.
36. Kennedy, J. and Spears, W., "Matching Algorithms to Problems: An Experimental Test of the Particle Swarm and Some Genetic Algorithms on the Multimodal Problem Generator", *IEEE World Congress on Computational Intelligence*, 1998, pp. 74-77.
37. Shi, Y. and Eberhart, R. C., "Parameter Selection in Particle Swarm Optimization", *Evolutionary Programming VII: Proc. EP 98*, Springer-Verlag, New York, 1998, pp. 591-600.
38. Salerno, J., "Using the Particle Swarm Optimization Technique to Train a Recurrent Neural Model", *Proceedings of the Ninth IEEE International Conference on Tools with Artificial Intelligence*, 1997, pp.45-49.
39. 曾俊傑，「一個智慧型指紋辨識系統的設計方法論」，義守大學電機工程研究所碩士論文，

民國八十九年。

40. 張榮芳，「電力用戶負載歸類及整合」，國立中山大學電機工程研究所博士論文，民國九十年。
41. 徐育良，「以粒子群最佳化為基礎之電腦遊戲角色設計之研究」，東海大學資訊工程與科學研究所碩士論文，民國九十一年。
42. 葉麗雯，「供應商產能有限及價格折扣下多產品多供應商最佳化採購決策」，元智大學工業工程與管理研究所碩士論文，民國九十一年。
43. 葉思緯，「應用粒子群最佳化演算法於多目標存貨分類之研究」，元智大學工業工程與管理研究所碩士論文，民國九十二年。
44. Yan, S. and Young, H. F., "A Decision Support Framework for Multi-Fleet Routing and Multi-Stop Flight Scheduling", *Transportation Research*, Vol. 30A, 1996, pp. 379-398.

