運輸計劃季刊
第三十九卷 第一期
民國九十九年三月
頁 99 ～ 頁 132

Transportation Planning Journal
Vo1. 39 No. 1
March 2010
PP. 99～132

調適型導引螞蟻演算法求解時窗收卸貨問題之研究[1]

SOLVE PICK UP AND DELIVERY PROBLEM WITH TIME WINDOWS VIA A GUIDED ADAPTIVE ANT COLONY SYSTEM

李泰琳 Tai-Lin Li [2]
張　靖 Ching Chang [3]

(97 年 5 月 21 收稿，98 年 4 月 16 日第一次修改，
98 年 11 月 10 日第二次修改，99 年 3 月 8 日定稿)

摘　要

本研究主要結合導引區域搜尋技術(guided local search, GLS) 與李泰琳等人提出的調適型螞蟻演算法(adaptive ant colony system, AACS)，設計調適型導引螞蟻演算法(guided adaptive ant colony system, GAACS) 求解時窗收卸貨問題(pickup and delivery problem with time windows , PDPTW)。首先，引用李泰琳等人所提出之問題規模精簡策略與關連式旅行成本網絡結構，將 PDPTW 轉換為無時窗的近似收卸貨問題 (similar pickup and delivery problem, SPDP)，其優點為在計算過程中不需要考慮時窗限制；其次，提出 GAACS 演算法整合的觀念。演算法績效測試為應用國際標準題庫 VRPTW 例題，配合 Lau 與 Liang (2002)方法進行修改，產生具有標竿解之 PDPTW 例題 18 題進行測試，問題規模皆為 100 個作業，結果顯示 GAACS 可以在

1. 本研究獲國科會計畫補助(NSC 97-2218-E-266 -001)，謹此致謝。
2. 醒吾技術學院行銷與流通管理系助理教授；中華大學科技管理所博士 (聯絡地址：244 臺北縣林口鄉粉寮路 1 段 101 號醒吾技術學院行銷與流通管理系；電話：0922145691；E-mail：096040@mail.hwc.edu.tw)。
3. 中華大學運輸科技與物流管理學系副教授兼系主任 (聯絡地址：30012 新竹市香山區五福路 2 段 707 號中華大學運輸科技與物流管理學系；電話：03-5186598；E-mail：ching@chu.edu.tw)。

25.9 分鐘平均時間，求得與標竿解平均差異僅 1.8%之近似最佳解。

關鍵詞：螞蟻演算法；導引區域搜尋法；時間窗收卸貨問題

ABSTRACT

The Pick Up and Delivery Problem with Time Windows (PDPTW) was solved via the proposed Guided Adaptive Ant Colony System (GAACS), which was integrated by the Guided Local Search (GLS) and Adaptive Ant Colony System (AACS) proposed by Lee Tai-Lin, etc. However, in order to solve the PDPTW more efficiently and effectively, a PDPTW was transferred to be a new similar PDP (SPDP) without time window via the Time Window Partitioning and Discretization Strategy. In order to show the contribution of the GAACS, 18 Solomon benchmark VRP problems were transferred to be PDPTW problems via the method developed by Lau and Liang. According to the computation results, we obtained the average percentages of errors of the best published solutions among18 PDPTW test instances, with 1.80% and an average computation time of 25.9 minutes. This shows that our proposed GAACS method can solve PDPTW accurately in a reasonable time.

Key Words: *Ant colony optimization; Guided local search; Pickup and delivery problem with time windows*

一、前 言

現今提供物流服務的汽車貨運業者眾多，該型態作業多為點對點運輸，客戶提供貨物資訊，要求貨運公司在指定時間區間派車至某處收貨再送至另一處，業者為提升競爭力，必須提供快速、穩定、正確，且客製化的服務，因此時窗限制收卸貨問題 (pickup and delivery problem with time windows, PDPTW) 的研究將益發重要。PDPTW 已知屬於 NP-hard 問題 [1]，運算時間會隨著問題規模增加呈指數成長，大規模問題極不易在合理時間內求得精確解。因此，本研究在進行問題求解之前，將延用作者過去所提出的 PDPTW 前置處理作業 [2]，將問題簡化為無時窗限制的近似收卸貨問題 (similar pickup and delivery problem, SPDP) 後再進行求解，可以大幅減少求解過程中所有與時間相關的運算過程，大幅降低求解時間。

演算法方面，PDPTW 隨著問題規模漸趨龐大、求解品質要求、以及求解的時效性，結構簡單的啟發式演算法 (heuristics method) 已漸漸無法滿足求解效率與品質之需求。近年進行相關研究之學者們多採用人工智慧概念、自然界生態法則、與生物習性觀察等結果，發展巨集啟發式演算法 (meta-heuristics method)，亦已成為目前求解高複雜度組合最佳化問題的主流方法，且持續有新演算法被提出。螞蟻群落最佳化演算法 (ant colony optimization, ACO) 是近年新興的巨集啟發式演算法 [3]，已陸續成功地應用於求解多種複雜的組合最佳化問題 (combinatorial optimization problems)，但應用於求解 PDPTW 之研究

成果則相對稀少。近十五年來，隨著對求解機制執行細節的不同，已有許多種改良式螞蟻演算法陸續被提出，這些改良的方法被統稱為 ACO [4]，其中李泰琳等[5]提出的調適型螞蟻演算法 (adaptive ant colony system, AACS) 求解旅行推銷員問題 (traveling salesman problem, TSP) 時的求解績效相當優異，因此本研究繼續延伸 AACS，作為設計求解 PDPTW 巨集啟發式演算法的基礎。

AACS 是以群體為基礎所發展出的演算法，屬於廣度搜尋策略方法，將求解過程所求得的品質不錯的可行解，透過費洛蒙激素記錄在路段上。然而許多研究指出 [6-10]，這種僅具廣度搜尋策略的方法，必須搭配有力的區域搜尋方法，增加深度搜尋能力，才能有效提升其解題績效，因此配合強而有力的區域搜尋技術，乃是提升 AACS 效能的重要因素之一。近年以區域搜尋為基礎所發展出較著名的巨集啟發式演算法有 1995 年 Tsang 與 Voudouris 提出的導引式區域搜尋法 (guided local search, GLS) [11]，以及 1997 年時 Mladenović 與 Hansen 提出的變動鄰域搜尋法 (variable neighborhood search, VNS) [12]，皆可有效提升傳統區域搜尋演算法的求解績效。NVS 的執行概念為在區域搜尋過程中，有系統地更換不同大小的鄰域機制，避免單一鄰域機制陷入局部最佳解困境，相較於其他以區域搜尋為基礎的巨集啟發式方法，VNS 並非僅遵循單一個特定的交換程序，而是逐漸擴大其鄰域搜尋的範圍，與 AACS 相同屬於廣度搜尋策略的方法，不易提升 AACS 深度搜尋的能力。GLS 是一種以懲罰為基礎的方法，能夠讓演算機制或電腦程式思考如何避免落入局部最佳解之束縛，導引傳統區域搜尋演算法朝向存在全域最佳解的空間趨近，能更有效率地執行尋優過程。反觀 ACO 演算法中螞蟻選擇前進方向時，雖需要同時考慮路段費洛蒙濃度與路段意念函數，但是路段意念函數值為路段成本的倒數，在求解過程中始終是固定不變的，不似路段費洛蒙濃度會隨著演算過程累積接資訊，逐漸突顯出最佳路徑資訊。因此，若導入 GLS 懲罰解特徵的機制於 ACO 演算法中 (在車輛途程問題中，將各個路段設定為特徵)，藉由懲罰來增加部分較差或曾經行走過的路段的成本，即可使路段意念函數也能累積可行解資訊，且與路段費洛蒙濃度所累積的資訊並不相同，因為路段費洛蒙濃度累積的是較佳、較建議行走的路段，而導入 GLS 概念的路段意念函數則是引導螞蟻避開較差、較不建議行走的路段。再者 GLS 是以區域搜尋為基礎所發展出的巨集啟發式演算法，可彌補 AACS 深度搜尋能力不足的缺失。

鑑於上述原因，本研究選擇將 GLS 與 AACS 巨集啟發式演算法結合，設計兼具搜尋深度與廣度的 GAACS 演算法，求解消除 PDPTW 中所有時窗限制後轉換所得的 SPDP。由於 SPDP 與 PDPTW 之最適解與可行解完全相同，且無需考慮時窗限制，因此可有效降低求解時間，進而提升求解效率與品質。

二、文獻回顧

本節內容首先詳述 SPDP 之問題定義與數學規劃模式，關於 PDPTW 之問題定義與數

學規劃模式，請參考 Savelsbergh 與 Solomon 所提出之研究 (Savelsbergh 與 Solomon) [13]，本文中將不再贅述。其次回顧過去學者在 PDPTW 方面所提出的演算法優劣與應用領域，最後針對近年新興之巨集式啟發式演算法發展現況進行回顧，以輔助本研究能夠設計有效提升演算法求解績效的改善策略。

2.1 近似收卸貨問題 (SPDP)

SPDP 是由 PDPTW 應用時窗分割策略 [14]，與李泰琳等人 [2] 所提出之整數化策略及問題規模精簡策略，將時間窗限制消除後轉換所得，與 PDP 極為相似。後續小節詳述 SPSP 問題定義與數學規劃模式，符號定義於表 1。

表 1　SPDP 相關符號定義

N	訂單集合，$n=\|N\|$，表示訂單數量，集合中的元素為 J_u 。
J_u	表示編號為 u 的作業點，當 $u=0$ 時為場站；$u=1,\ldots,n$ 時表示收貨作業點；$u=n+1,\ldots,2n$ 為卸貨作業點，其中收貨作業點 J_u 所對應的卸貨作業點為 J_{u+n} 。該變數為集合變數，集合中的元素為作業點所分割出的子作業。
M	車輛集合，$m=\|M\|$，表示可使用的車輛數， $k\in M$ 。
Q	車容量限制。
P	使用車輛的固定成本，相較於各車輛旅行距離而言，是一個較旅行成本相對大的常數值。
R_k	車輛 k 的行駛路線。
R	收、卸貨指派與路線規劃結果，亦為所有車輛行駛路線的集合。 $R:=\bigcup_{k\in M}R_k$
O	所有車輛路線的起、迄地點。
m	被指派的車輛數。
u	變數下標，表示作業點編號。
v	變數下標，表示作業點編號。
q_u	作業點 u 運載量，$1\leqq u\leqq n$ 時，$q_u\geqq 0$，反之 $q_u\leqq 0$，另外 $q_0=0$。
$f(u)$	作業點 u 所分割出的子作業總數。
$J_{u,i}$	作業點 u 所分割出的第 i 個子作業，$i=0,1,\ldots, f(u)$ 。
$c_{ui,vj}$	子作業 J_{ui} 與子作業 J_{vj} 間的旅行成本。
Z_{ui}^k	若車輛 k 有服務 J_{ui} ，則 $Z_{ui}^k=1$，反之 $Z_{ui}^k=0$，另由於場站為車輛的起迄點，因此 $Z_0^k=1$。
$X_{ui,vj}^k$	若車輛 k 從 J_{ui} 行駛至 J_{vj} (行經該路段)，則 $X_{ui,vj}^k=1$，反之 $X_{ui,vj}^k=0$。
y_u	車輛離開作業點 u 時的負載量， $y_0=0$ 。

時窗分割策略最早是由 Appelgren [14] 求解船運排班問題時所提出，求得的近似最佳解品質極佳。分割方法將具有連續時窗的作業點，依固定數量或是固定時段，分割為一群具有不同時窗的數個作業點 (文中後續稱之為子作業)，並各自視為一個獨立的決策變數。李泰琳等人 [2] 的研究中有相當詳盡的敘述，本文中將不再贅述。

2.1.1 SPDP 問題定義

PDP 中所包括的所有作業皆僅需服務一次，而 SPDP 問題中，屬於同一作業點所分割出的子作業們，僅能選擇其中一個服務僅一次，因此 SPDP 與 PDP 間的差異，僅在於針對訂單所下的定義與服務方式不同而已，分別敘述於後，相關符號已定義於表 1。

1. 訂單定義間的差異：PDP 中的一個訂單 (i)，僅包括 1 個收貨作業 (N_i) 與 1 個卸貨作業 (N_{i+n})，然而 SPDP 中的一個訂單 (u)，是包括 1 個收貨作業集合 (J_u) 與 1 個卸貨作業集合 (J_{u+n})，收、卸貨集合中所包含的元素數量並不一定相等，每個元素皆代表一個獨立的子作業。
2. 訂單服務方式差異：一般 PDP 問題中，每個收貨作業與卸貨作業都必須僅服務一次，但是 SPDP 中屬於相同收貨作業集合的收貨子作業們，只能選其中一個服務僅一次，其原因在於屬於相同收貨作業集合 (J_u) 中的收貨子作業們 ($J_{u,j}$)，其實都是由原 PDPTW 中訂單(u)的收貨作業分割所得。

綜上所述，過去學者針對 PDP 所設計之演算法，將可以很容易應用於求解 SPDP 問題，僅需修改訂單變數定義，及加入一個作業步驟 (當子作業 J_{ui} 被服務時，將其所屬作業集合 (J_u) 中的所有子作業皆標註成已服務即可)。因此 SPDP 與 PDP 極為相似，問題定義如下：

(1) 每項訂單只能且必須要指派給一輛車服務。

(2) 每項訂單只有一個收貨作業地點與一個卸貨作業地點，且載貨量與卸貨量必須相等。

(3) 一個訂單 (u) 包括 1 個收貨作業集合 (J_u) 與 1 個卸貨作業集合 (J_{u+n})，收、卸貨集合中所包含的元素數量並不一定相等，每個元素皆代表一個獨立的子作業。

(4) 收貨作業集合中的第 j 個元素，即為收貨子作業 ($J_{u,j}$)，同一收貨集合中每一個收貨子作業的服務需求，除服務時窗限制之外，其餘完全相同 (包含載貨量、地點與收貨耗時)；以此類推，卸貨作業集合中的第 j 個元素，即為卸貨子作業 ($J_{u+n,j}$)，且同一卸貨作業集合中每一個元素，亦皆為一個獨立的卸貨子作業，服務需求方面也是除了服務時窗之外，其餘皆完全相同 (包含卸貨量、地點與卸貨耗時)。

(5) 車輛 k 行駛路線 (R_k) 的起、迄點必須為單一場站 (O)，k =1,2,…,m。

(6) 訂單 u 指派給車輛 k 進行服務，則車輛 k 的行駛路線 (R_k)，必須包括訂單 u 的收貨作業集合 (J_u) 中僅一個收貨子作業 ($J_{u,j}$)，與卸貨作業集合 (J_{u+n}) 中僅一個卸貨子作業 ($J_{u+n,j}$)，且行駛路線 (R_k) 中，收貨子作業 ($J_{u,j}$) 的服務順序必須在卸貨子作業 ($J_{u+n,j}$) 之前。

(7) 車輛 k 的載貨量無論何時皆不能超過車容量上限 (Q)。

(8) PDP 問題的可行解 (R)，為所有車輛路線 (R_k，k =1,2,…,m) 的集合。

(9) 子作業點完成後立即離開，並前往下一個子作業點。

(10)當子作業 J_{ui} 被服務時，其所屬作業集合 (J_u) 中的所有子作業皆視為已服務。

2.1.2 SPDP 數學規劃模式

SPDP 不允許服務延遲，且車輛服務各作業點時皆採用嚴謹離開時間法則 [15]，意即無論到達子作業的時間點為何時，皆將該子作業的最晚完成時間，視為服務完成後的離開時間。因此目標式中僅需要考慮最小化的使用車輛數與路徑成本，如式 (1) 所示，只要 P (車輛固定成本方式) 夠大就可以滿足最少車輛數為目標式主要目標，最短旅行成本為次要目標，SPDP 數學規劃模式中使用的符號定義於表 1。作業中服務等待時間最小化之目標，採用時窗分割技術進行控制，並未加入目標式中，當分割時窗時使用的 δ 值愈小，作業中的服務閒置時間也將愈少。

$$Min \quad \sum_{k=1}^{m}\sum_{u=0}^{2n}\sum_{i=0}^{f(u)}\sum_{\substack{v=0\\ v\neq u}}^{2n}\sum_{j=0}^{f(v)} c_{ui,vj} X_{ui,vj}^{k} + P\cdot\sum_{k=1}^{m}\sum_{u=1}^{2n}\sum_{i=0}^{f(u)} X_{0,ui}^{k} \tag{1}$$

SPDP 所使用的子作業旅行成本矩陣中存在旅行時間的路段，必定滿足 8 項原則，分別條列如下，因此 SPDP 的求解過程中，僅需考量不連續作業間的服務順序關係以及車容量限制條件。數學規劃模式限制式詳列於下，模式中使用的集合、決策變數與參數定義於表 1。

原則 1： 各作業地點必須僅服務一次。

原則 2： 滿足子作業服務時窗限制。

原則 3： 同一訂單的收貨作業點必須在卸貨作業點前被服務。

原則 4： 任兩連續服務作業所裝載的貨物量，不得超過車容量限制。

原則 5： 車輛離開與回到場站時必須為空車。

原則 6： 若某一收貨作業僅可由場站直達，則該收貨作業能直接前往的卸貨作業，必定且唯一是其相對應的卸貨作業。

原則 7： 若某一卸貨作業僅能前往場站 (回場站)，則可直達該卸貨作業的收貨作業，必定且唯一是與其相對應的收貨作業 (原則 7 與原則 6 相呼應)。

原則 8： 子作業點應符合流量守恆定理。

$$\sum_{k=1}^{m}\sum_{i=0}^{f(u)} Z_{ui}^{k} = 1 \text{，} u=1,2,\ldots,2n \tag{2}$$

$$\sum_{i=0}^{f(u)} Z_{ui}^{k} - \sum_{j=0}^{f(u+n)} Z_{u+n\ j}^{k} = 0 \text{ ，} u=1,2,\ldots,n \text{ ，} k=1,2,\ldots,m \tag{3}$$

$$\sum_{k=1}^{m}\sum_{i=0}^{f(u)} l_{ui} \times Z_{ui}^{k} - \sum_{k=1}^{m}\sum_{j=0}^{f(u+n)} l_{u+n\ j} \times Z_{u+n\ j}^{k} \le 0 \text{ ，} u=1,2,\ldots,n \tag{4}$$

$$\sum_{k=1}^{m}\sum_{i=0}^{f(u)}\sum_{v=0}^{2n}\sum_{j=0}^{f(v)} X_{ui,vj}^{k} = 1 \text{ ，} u=1,2,\ldots,2n \text{ ，}$$

$$\text{其中}\begin{cases} \text{若 } u=1,2,\ldots,n \text{ ，則} v \neq 0 \text{且} v \neq u \\ \text{若 } u=n+1,n+2,\ldots,2n \text{ ，則} v \neq u-n \text{且} \mathrm{v} \neq u \end{cases} \tag{5}$$

$$\sum_{k=1}^{m}\sum_{u=0}^{2n}\sum_{i=0}^{f(u)}\sum_{j=0}^{f(v)} X_{ui,vj}^{k} = 1 \text{ ，} v=1,2,\ldots,2n \text{ ，}$$

$$\text{其中}\begin{cases} \text{若 } v=1,2,\ldots,n \text{ ，則} u \neq v+n \text{且} u \neq v \\ \text{若 } v=n+1,n+2,\ldots,2n \text{ ，則} u \neq \mathrm{v} \text{且} u \neq 0 \end{cases} \tag{6}$$

$$\sum_{v=0}^{2n}\sum_{j=0}^{f(v)} X_{vj,ui}^{k} = Z_{ui}^{k} = \sum_{v'=0}^{2n}\sum_{j'=0}^{f(v')} X_{ui,v'j'}^{k} \text{ ，} k=1,2,\ldots,m \text{ ，} u=1,2,\ldots,2n \text{ ，} \tag{7}$$

$$\text{其中}\begin{cases} \text{若 } u=1,2,\ldots,n \text{ ，則} v \neq v+n \text{ ，} v \neq u \text{ ，且} v' \neq 0 \text{ ，} v' \neq u \\ \text{若 } u=n+1,n+2,\ldots,2n \text{ ，則} v \neq 0 \text{ ，} v \neq u \text{ ，且} v' \neq u-n \text{ ，} v' \neq u \end{cases}$$

$$\sum_{u=1}^{n}\sum_{i=0}^{f(u)} X_{0,ui}^{k} = \sum_{v=n+1}^{2n}\sum_{j=0}^{f(v)} X_{vj,0}^{k} \text{ ，} k=1,2,\ldots,m \tag{8}$$

$$\sum_{u=1}^{n}\sum_{i=0}^{f(u)} X_{0,ui}^{k} \le 1 \text{ ，} k=1,2,\ldots,m \tag{9}$$

$$y_v = y_v\left(1 - \sum_{i=0}^{f(u)}\sum_{j=0}^{f(v)} X_{ui,vj}^{k}\right) + \sum_{i=0}^{f(u)}\sum_{j=0}^{f(v)} X_{ui,vj}^{k}\left(q_v + y_u\right) \text{ ，} k=1,2\ldots,m \text{ ，} \tag{10}$$

$v=0,1,\ldots,2n$ ，$u=0,1,\ldots,2n$ ，

$$\text{其中}\begin{cases} \text{若 } v=1,2,\ldots,n \text{ ，則} u \neq v+n, \text{且} u \neq v \\ \text{若 } v=n+1,n+2,\ldots,2n \text{ ，則} u \neq 0, \text{且} u \neq v \end{cases}$$

$$0 \le y_u \le Q \text{ ，} u=1,2,\ldots,2n \tag{11}$$

限制式 (2) 控制每個作業點僅能被一輛車服務一次，式 (3) 控制每一個訂單的收送貨作

業必須由同一輛車完成服務，式 (4) 控制收卸貨作業的服務順序，式 (5) (6) 限制每一個作業點只能有一輛車駛離與到達，式 (7) 為流量守恆現制式，式 (8) 控制場站為車輛的起迄點，式 (9) 則限制車輛僅能離開與回到場站至多一次，式 (10) 與 (11) 則為車容量限制。

2.2 PDPTW 演算法回顧

PDPTW 相關研究的解法上，可以區分為精確解演算法與啟發式演算法兩種。可以求出精確解 (exact solution) 的辭書搜尋法、分枝定限法、分枝截面法等求解效率皆不佳，且僅止於求解較小型的問題 [16-19]，因此多數學者皆已朝向巨集啟發式演算法 (meta-heuristics) 的方向發展 [20-22]，原因在於巨集啟發式演算法，能夠跳脫求解過程中陷入局部最佳解的困境，與改善傳統啟發式演算法的效率。

2.2.1 PDPTW 精確解演算法

Psarafis 學者針對單一車種 PDPTW 問題，提出複雜度為 $O(n^2 3^n)$ 的動態可成式演算法 [23-25]，求取精確解，目標函數為求取最小的顧客服務不便 (例如服務延遲)，但僅能求解很小規模的問題，最大需求量必須小於 10。

Sexton 等 [25,26] 也針對相同的問題，提出組合式演算法求取精確解，該方法將單一車種 PDPTW 問題中，有關於路線規劃方面的問題，轉換為整數規劃問題；而關於指派作業方面的問題，則轉換為線性問題，再利用演算法依序個別求解，其可以在 18 秒內有效求解出需求規模介於 7~20 之間的 PDPTW 問題。

Lu 探討多車種 PDP 問題 (multiple vehicle pickup and delivery problem, MPDP) 的精確解[19]，研究中設計混合型線性整數規劃問題模式來表示 MPDP 問題，模式特色為僅使用單一二元變數表示路徑走向，不用增加額外的變數來表示收、卸貨節點間的配對關係、服務車輛編號、以及各客戶之收、卸貨服務須同車處理的限制。該模式可以在 3 小時 CPU 時間 ([Unix 系統] SUN Fire 4800 server) 內，解出所羅門國際題庫之 VRP 題型中，C 型問題 (5 輛車 17 個顧客)，以及 R 型問題 (5 輛車 25 個顧客) 之精確解。

綜上所述，目前所發展可應用於求解 PDPTW 問題精確解之演算法，仍無法有效降低求解時間，且在合理時間內可求得精確解之問題規模，亦不超過 30 個顧客。

2.2.2 PDPTW 啟發式演算法

Dumas 等學者提出 Column Generation 方案與 10 個可以改善路線成本的方法 [17]，設計數種改善解組合，求解考量車容量、時窗、服務優先順序、與需求合併的 VRPPDTW 單場站問題，組合式改善法雖可提升求解的品質，但求解時間亦相對增加。

Nanry 與 Barnes 提出反應式禁忌演算法 (reactive tabu search)，以最小旅行成本為車輛指派與路徑規劃目標，配合違反時窗與車輛超載兩種懲罰成本函數機制，考量與違反時窗限制條件，求解 m-PDPTW 問題 [27]，求解問題規模為 100 個節點的問題，與 Kohl [18] 所

求得的近似最佳解相比較，平均差距僅 0.01%，且求解時間平均減少 65%。

Lau 與 Liang 發展兩階段啟發式演算法求解 PDPTW 問題 [16]，第一階段利用類似掃描法的方式，針對收貨點進行路線分群，然後再將卸貨點加入對應群中最適合的路線位置，第二階段亦配合利用禁忌搜尋法進行路線改善作業，測試需求為 100 對左右規模的問題，可求得不錯的近似解。

Bent 與 Van Hentenryck 亦提出兩階段式演算法 [28]，研究中提及降低使用的車輛數，是降低總旅行成本的關鍵之一，因此第一階段主要目標為減少車輛總數，第二階段為減少總路線成本，同時也再次減少車輛數。演算法核心則改為模擬退火法 (simulated annealing, SA) 與大型鄰域搜尋法 (large neighborhood search, LNS)，該演算法改善了國際題庫中 200 與 600 個顧客標竿解達 47%與 76%，為求解績效極佳的啟發式演算法之一，但求解問題規模較大時，求解時間亦相對增加。

Mitrovic-Minic 與 Laporte [29] 則認為車輛在服務客戶的過程中，其閒置時間的分布狀況，會影響新增需求指派結果，亦與整體面的最佳指派結果息息相關，研究中將車輛總服務時間依據服務時間、服務顧客數或是固定時段，區分為數個時區，在各個時區中，依據該時區的時間區間，與時區中所服務客戶的時窗限制，設計 4 種等候策略進行求解，結果顯示研究中所設計的高階變動式等待策略 (advanced dynamic waiting strategy)，較傳統等待方式 (服務完成後立即離開並前進到下一個作業地點)，平均可減少 4.34% ~ 5.3%總距離，以及減少 3.19% ~ 5.94%的使用車輛數。

Dantzig-Wolf 分解法 [30]，近似於 Lagrangian 鬆弛法，是一種可以求得近似最佳解的方式，該研究中藉由鬆弛 PDPTW 中較複雜的限制式，將原問題簡化後再求解。應用類似鬆弛法的尚有 Kolen 等學者提出特定區域鬆弛法 (state space relaxation) [31]，針對動態問題進行求解。Desrosiers 等學者[32]則提出時間窗限制式鬆弛法，首先鬆弛問題中所有包含時窗的限制式，然後進行求解，藉由每次求解回合中最後產生的不可行解，作為下一求解回合中時窗分割的依據，然後再利用網絡鬆弛法 (network relaxation methods) 求解鬆弛後的問題，如此反覆求解。該種啟發式演算法求解績效較 Dantzig-Wolf 分解法差，但優於 Lagrangian 鬆弛法。

2.3 巨集啟發式演算法回顧

巨集啟發式演算法能夠跳脫求解過程中陷入局部最佳解的困境，並可改善傳統啟發式演算法的效率 [33-37]，近年進行 PDPTW 相關研究之學者們，多採用發展巨集啟發式演算法 (meta-heuristics method) 進行求解。調適型螞蟻演算法 (AACS) 是本文作者們近年提出的巨集啟發式演算法，求解 TSP 國際題庫績效卓越，因此本研究繼續延伸 AACS 結合 GLS，設計兼具搜尋深度與廣度的 GAACS 演算法，應用於 PDPTW 求解。選擇 GLS 的動機已詳述於第一節前言中，本節不多贅述，後續將簡單回顧 AACS 與 GLS 演算法。

2.3.1 導引式區域搜尋法 (GLS)

GLS 由 Voudouris 與 Tsang 所提出 [38]，其概念源自於 GENET 之類神經網路方法 [39]，與 Tabu Search 極為相似 [40]。GLS 是一種以懲罰為基礎的方法，搜尋過程中，GLS 利用解特徵來描繪解的特性，並應用懲罰的概念，藉由懲罰特徵來獲得擴增目標函數，導引後續可行解的搜尋方向，避開可行解較差的鄰域與曾經搜尋過的鄰域，朝向可能存在全域最佳解的空間趨近，能更有效率地執行尋優過程。擴增目標函數如式(12)所示，相關符號定義於表 2，並將專有名詞條列解釋於後。

$$h(s) = g(s) + \lambda \cdot \sum_{i=1}^{M} p_i \cdot I_i(s) \tag{12}$$

1. 特徵(I_j)

GLS 根據問題的特性，為可行解定義「特徵」，在車輛途程問題中，通常是以節線作為特徵，一條節線就視為是一個特徵。

2. 擴增目標函數 (h)

擴增目標函數是在原問題的目標函式 (g) 中，為每一個特徵附加一個 λ 權重的特徵懲罰值 (p_i)，產生新的目標函數 (h)，引導演算法在解空間中的搜尋方向。新的目標函數中附加的懲罰值參數值，會隨著每回合搜尋到的區域最佳解而改變，λ 用來控制特徵懲罰值在擴增目標函數中的影響程度。

3. λ 權重值

λ 權重值主要目的，為控制特徵懲罰值在擴增目標函數中的影響程度，設定方式可以是在演算過程中始終不會改變的固定定值 [41]，也有學者依問題特性隨著演算過程，以遞增或遞減方式變動 [42]，也有學者設計變動方式隨著目前最佳者品質而改變 [43]，在各回合的搜尋過程中，擴增目標函數中的 λ 係數是固定的，但是在進入不同回合時，λ 值則會隨前一回合的局部最佳解而變，計算公式說明於式 (13)。不同的設定方式下，或求解不同型態的問題，適合的參數設定亦不盡相同，但值得注意的是，λ 權重值是影響 GLS 演算法績效的重要關鍵之一。

$$\lambda = \frac{a \cdot g(s^*)}{\sum_{i=1}^{M} Ii(s^*)} \tag{13}$$

其中 s^*為前一回合之局部最佳解；a 為事先設定之參數。可知 λ 值主要受到前回合局部最佳解之原始目標函數值與特徵出現次數的影響。

4. 特徵成本(c_j)

GLS 為每一個特徵設定特徵成本，在車輛途程問題中，通常是以節線成本當作特徵成本。

5. 效用函數與特徵懲罰值

懲罰值更新計算方式由效用函數 (*util*) 控制 (如式 (14))，求解過程中，當搜尋過程陷入區域最佳解時，意即當區域搜尋找到新的局部最佳解 s 時，GLS 會根據式 (14) 計算出各特徵之效用值 ($util_i$)，然後找到具有最大效用值的特徵 *Ii**，並改變該特徵的懲罰值，即令 *pi* = pi* + 1*，(具有最大效用值的特徵可能不只一個，所有滿足極大化效用函數的解特徵所對應的懲罰值參數，皆要累加)，其他特徵的懲罰值則維持不變。故懲罰值 *pi* 並非固定值。其目的在於可藉由懲罰該特徵，來導引區域搜尋離開該局部最佳解，而能繼續朝其他空間來搜尋。

$$util(s, f_i) = I_i(s) \cdot c_i / (1 + p_i) \tag{14}$$

s 表示本回合找到的局部最佳解；$util_i$ (*s*)表示在 *s* 時，第 *i* 項特徵 (*Ii*) 的效用值函數；c_i 為特徵 (*Ii*) 的特徵成本；p_i 為特徵 (*Ii*) 的特徵懲罰成本。

由式(14)可知，若局部最佳解 (*s*) 未具有某特徵時，該特徵之效用值為 0，因為 $I_i(s) = 0$；反之，若具有某特徵時，則其效用值會受到特徵成本 (c_i) 與特徵懲罰成本 (p_i) 這兩個因素的影響。

(1) 若特徵 (*Ii*) 的成本 c_i 越高，其效用值越高；

(2) 若特徵 (*Ii*) 的懲罰值 p_i 越大，其效用值則越低。

表 2　GLS 演算法符號定義表

g	問題原始目標函式，*g*(*s*)為可行解 *s* 的目標成本。
h	附加懲罰值後的新目標函式，*h*(*s*)為可行解 *s* 的新目標成本。
S	可行解集合，集合中所包括的可行解元素以 *s* 表示。
λ	附加懲罰值對原目標函式影響力參數。
M	特徵數量。
p_i	特徵 *i* 所對應的懲罰值。
$I_i(s)$	特徵函數，若可行解 *s* 中存在特徵 *i* 時，$I_i(s) = 1$，若否，則 $I_i(s) = 0$。
c_i	特徵 *i* 的特徵成本。

GLS 應用領域相當廣泛，尤其是適合求解組合優選問題，目前已成功應用於旅行推銷員問題 (TSP) [44,45]、二次指派問題 (quadratic assignment problem, QAP) [46]、局部限制滿意問題 (partial constrain satisfaction) [47,48]，工作人員排班問題 (workforce scheduling problem) [11,49]、工作站排班問題 (job shop scheduling problem) [50]、三維裝箱問題 (three-dimensional bin-packing problem) [51]、廣播接收頻率指派問題 (radio link frequency assignment problem) [52]、函數最佳化問題 (function optimization problem) [38]，與資源分派問題 (resource

allocation) [53] 等，亦可以簡單與多種啟發式演算法配合使用，例如 2-Opt 交換法 [54]， 節點重置 (relocate) 與節點交換 (exchange) 等 [55]。

學者 Kilby 等應用 GLS 配合 4 種路徑改善方法 [40]，求解時間窗與車容量限制的 VRP 問題，並應用所羅門題庫中 C1、C2、R1、R2、CR1 與 CR2 類型的 VRPTW 問題共 56 題進行測試，結果顯示 C1 與 C2 類型問題由於較易求解，所以 GLS 的表現與其他區域搜尋演算法 [56,57] 的績效皆很優秀；R2 與 CR2 的結果特別突出，但在 R1 類型的問題則最為耗時。

Voudouris 與 Tsang [45] 結合 GLS 與快速區域搜尋法 (fast local search, FLS) 求解 TSP，特徵設定方式與學者 Kilby 等 1997 年研究中所使用的方式雷同 [40]，測試結果顯示在 GLS 架構中，使用 2-Opt 為基礎之 FLS，比 Iterated Lin-Kernighan 交換法及基因-區域搜尋法具有更高的解題績效。

Kilby 等 [58] 將 GLS 法應用於時間窗限制車輛路線問題 (VRPTW)，其區域搜尋採用了 2-Opt、Relocate、Exchange 及 Cross 等方法。測試結果發現當車輛路線較長時，GLS 比其他文獻的方法表現較佳；但當路線較短時，GLS 則表現稍差。此結果顯示 GLS 似乎較適用於求解長路線的例題。而該結論亦與學者 Kilby 等過去研究結果[40] 相符合。

Voudouris 與 Tsang 以 TSP 問題為例 [43]，說明如何選定特徵，其特徵係指某些特定的節線。例如：若不希望 TSP 解的節線長度太大，可選擇某些很長的節線作為特徵，並設定其特徵成本為該節線之長度。

Zhong 與 Cole [59] 應用 GLS 法求解時間窗限制之回程取貨車輛路線問題(VRPBTW)，研究中考慮了有顧客優先順序及無顧客優先順序兩種情況。其解題架構為：首先構建一個起始解 (允許其為不可行解)，並提出一個提高可行性的 section planning 方法，然後使用 GLS 改善該起始解的可行性與目標值。此 GLS 法在部分 VRPBTW 例題求解上已突破目前已知標竿解。

2.3.2 調適型螞蟻演算法

調適型螞蟻演算法 (AACS) 為李泰琳、張靖所提出 [5]，研究中詳細探討前述螞蟻演算法的求解機制與各機制的優缺點，發現各方法彼此間的主要差異雖在於路徑費洛蒙濃度更新方式之不同，然而其費洛蒙計算公式所使用的參數組合 (α, β)，在整個執行過程中卻始終固定不變。然而，(α, β) 參數組合分別控制節線長度與費洛蒙值對路徑選擇的影響程度，故為影響求解效率與品質的重要因素之一。因此，若在求解過程中僅使用固定的參數值組合，可能會因為初期產生之可行解品質不佳，而累積了偏誤的可行解資訊，導致無法藉由累積費洛蒙的方式搜尋到更佳的解；此外，演算後期也較不易跳脫局部最佳解的束縛。有鑑於此，研究中針對螞蟻演算法的費洛蒙控制參數進行深入研究與分析，並導入調適型 (adaptive) 的執行概念於螞蟻演算法的參數控制機制當中，以提升螞蟻演算法的解題績效。

研究中應用調適型 (adaptive) 之概念 [5]，針對 ACS 演算法設計變動式參數設定機制 (AACS)，以及兩種改良式的路徑費洛蒙更新機制：路徑費洛蒙即時更新機制 (MACS1 機

制) 與路徑費洛蒙全域更新機制 (MACS2 機制)，再配合區域搜尋法效率改善技術，構建求解 TSP 問題的演算法測試架構，並自 TSP 的國際標竿題庫中，選擇了 24 個節點規模在 500 點以內之例題進行測試，以驗證調適型螞蟻演算法 (adaptive ant colony system, AACS) 的解題績效。

測試結果顯示，AACS 可有效提升 ACS 的解題績效平均達 0.54%，與題庫最佳解間的差距平均為 0.46%；MACS1 較原 ACS 演算法中所使用的更新機制，可以有效提升原 ACS 的解題績效平均達 0.44%，當求解之問題規模小於 400 時，與題庫最佳解間平均僅有 0.19% 的差距；MACS2 較原 ACS 演算法中所使用的更新機制，可以有效提升原 ACS 的解題績效平均達 0.42%，與題庫最佳解間的差距平均為 0.47%。

三、調適型導引螞蟻演算法

AACS 屬於廣度搜尋方法，然而這種僅具廣度搜尋策略的方法，須搭配有力的區域搜尋方法，增加深度搜尋能力，才能有效提升其解題績效。因此，本研究延伸 Tsang 與 Voudouris 提出的導引式區域搜尋法 (GLS)，導引 AACS 演算法中螞蟻搜尋路徑的方向，意即將 GLS 導引的概念結合於 AACS 演算法中，設計導引式調適型螞蟻演算法 (guided adaptive ant colony system, GAACS)，求解架構如圖 1，與 AACS 間的差異及求解機制條列於下。

1. GAACS 演算法中，構建每隻螞蟻可行路徑的「路徑構建機制」，與 AACS 雷同，唯一的差異僅在於意念函數的設計方式，導入了 GLS 懲罰解特徵的機制，文中後續稱之為「導引式路徑構建機制」，詳述於 3.1 小節。
2. GAACS 演算法中，更新路段費洛蒙濃度的機制稱為「導引式費洛蒙更新機制」，與 AACS 雷同，可細分為即時更新與全域更新兩種，詳述於 3.2 小節。
3. GAACS 演算法中所搭配的區域搜尋法，為 Tsang 與 Voudouris 提出的導引式區域搜尋法 (GLS) [11,49]，但為了能增加跳脫區域最佳解的機會，卻又不希望增加演算法複雜性，因此再配合使用巨集啟發式演算法中採用確定性接受暫劣解機制的門檻接受法，設計兼具搜尋深度與廣度的可行解改善作業，詳述於 3.3 小節。
4. GAACS 演算法中所選擇的特徵為路段，特徵數量等於總路段數，特徵成本設定為路段成本，特徵懲罰值更新方式亦沿用 Tsang 與 Voudouris 所提出的效用值函數公式 (式 14) [11,49]。
5. GAACS 演算法的停止法則有三，其一為求解回合數結束時，演算法即停止；其二為求解時間結束時，演算法即停止；其三為演算過程中求得之最適解次數，持續最大回合數的 1/5 次皆沒有改變時，亦終止演算法。

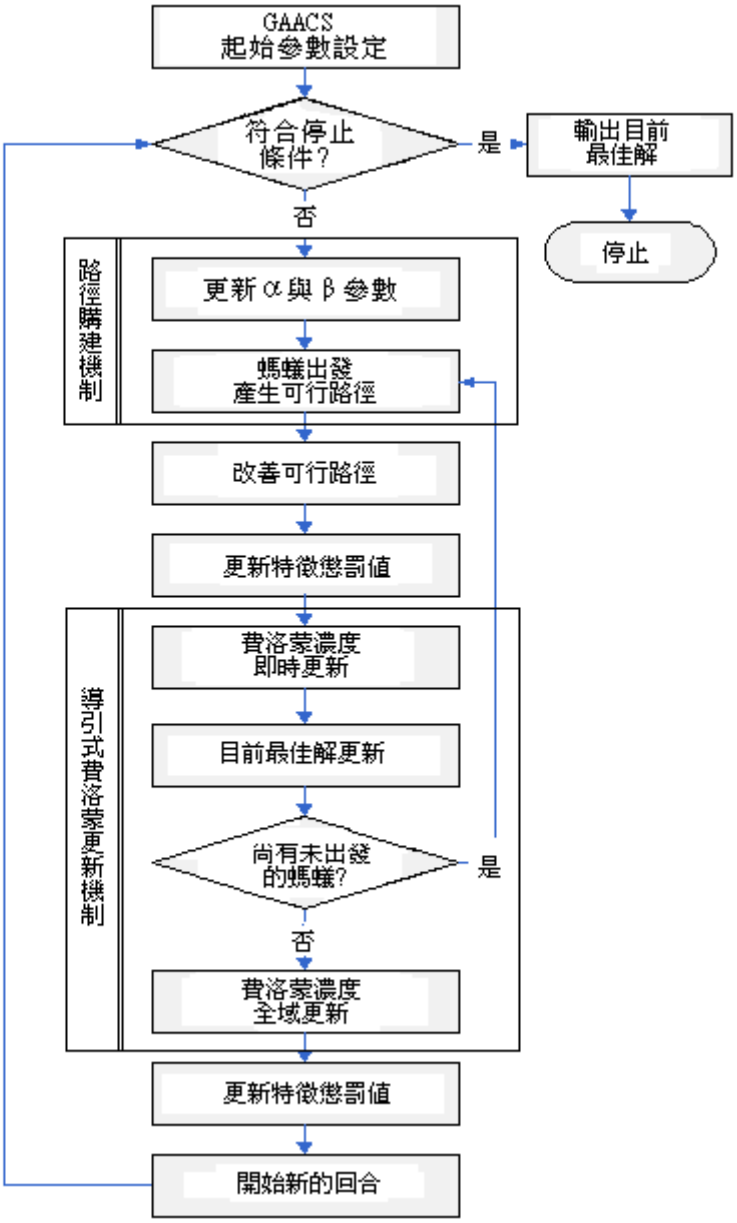

圖 1　GAACS 演算法架構圖

3.1 導引式路徑構建機制

GAACS 演算法與 AACS 演算法所使用的路徑構建機制，有兩個主要的差異，其一為記錄旅行路段距離的旅行成本矩陣；其二為螞蟻構建路徑方式。

(一) 演算法使用的成本表差異

本研究採用關連式旅行成本網絡結構表記錄旅行路段的成本，僅記錄每一個子作業與其他子作業間存在可行路段時段的連結關係，可大幅減少啟發式演算法求解過程中，每次搜尋回合中需重覆考慮子作業數量，且螞蟻搜尋可能繼續服務的作業過程中，亦不需考慮與服務時窗相關的限制條件，意即任何與作業服務時間窗相關的運算步驟皆不會產生，又可再進一步降低運算時間。關連式旅行網絡結構表，有別於傳統正方矩陣之上三角或下三角旅行成本矩陣，請參考 Fabian 與 Perez 之論文[1]，文中有詳盡的介紹，本篇則不再多做贅述。

(二) 螞蟻構建路徑方式的差異

AACS 演算法中的路徑構建方式，是以鄰近點插入法 (nearest inserting method) 為基礎設計螞蟻前進機制，求解 TSP 問題時無須選擇作業加入路徑中的最適位置，加入的作業皆是加入路徑最尾端，如此反覆作業，直到螞蟻行經所有作業點，即完成一條可行路徑的構建作業。但是此方法並不適用於求解 SPDP，因為 SPDP 的每一項需求作業，皆包括一組收貨作業與卸貨作業，因此構建可行路徑過程中，將子作業加入路徑時，必須同時考慮兩個子作業插入的位置，且可能加入路徑中的任何位置，並不一定是是加入路徑最尾端，因此，本研究修改 AACS 中螞蟻搜尋前進作業點之機制，改以最小成本插入法為基礎，設計為適合用來求解 SPDP 問題的可行路徑構建作業。茲將 GAACS 演算法之意念函數、螞蟻前進機率值計算方式與參數變動方式，條列詳述於下，相關符號定義於表 3。

1. GAACS 演算法意念函數

AACS 演算法中螞蟻構建路徑時，選擇最適合加入路徑中的作業時，主要取決於各作業的前進機率值，需要同時考慮路段費洛蒙濃度與路段意念函數，但是路段意念函數值為路段成本的倒數，在求解過程中始終是固定不變的，不似路段費洛蒙濃度會隨著演算過程累積資訊，逐漸突顯出最佳路徑資訊。

再者，從 GLS 的效用值函數公式 (式 14) 與擴增目標函數 (式 12) 明顯可以發現，可行解中路段成本愈高，且未被懲罰過的路段，愈容易被懲罰，而被懲罰的路段則會增加擴增目標函數的成本，造成後續搜尋可行解的過程中，將會盡可能的避免選擇到被懲罰過的路段，即可以避免將成本較高的路段加入可行解中。

有鑑於此，本研究導入 GLS 懲罰解特徵的導引概念，延伸至 AACS 演算法的前進機制中的意念函數，藉由懲罰機制，來增加部分較差或曾經行走過的路段成本，即可使路段意念函數也能累積可行解資訊，且與路段費洛蒙濃度所累積的資訊並不相同，因為，路段費洛蒙濃度累積的是較佳的建議行走路段，而路段意念函數則是記錄較差且建議避開的路段。

GAACS 演算法的前進機率值計算方式與 AACS 雷同，唯一的差異僅在於意念函數的設計方式，設計概念如式 (15)，詳細計算公式請參考式 (16) 與式 (17) 所示，加入 GLS 中

表 3　符號定義表

符　號	定　　義
n	需求作業數量。
$\Delta\eta_I^l(t)$	第 t 回合時將需求 l 的收、卸貨作業點同時且依序插入節線 I 後，意念函數的變量。
Δd_I^l	需求 l 收卸貨作業點同時且依序插入節線 I 後，路徑成本的變量。
Δp_I^l	需求 l 收卸貨作業點同時且依序插入節線 I 後，特徵懲罰值變量。
$\Delta\eta_{IJ}^l(t)$	第 t 回合時將需求 l 的收、卸貨作業點依序插入節線 I 與 J 後，意念函數的變量。
Δd_{IJ}^l	需求 l 收卸貨作業點依序插入節線 I 與 J 後，路徑成本的變量。
Δp_{IJ}^l	需求 l 收卸貨作業點依序插入節線 I 與 J 後，特徵懲罰值的變量。
$i1$ 與 $i2$	$i1$ 與 $i2$ 表示節線 I 的兩個端點，服務順序 $i1$ 在 $i2$ 之前，相同的，$j1$ 與 $j2$ 表示節線 J 的兩個端點，服務順序 $j1$ 在 $j2$ 之前。
J_l	編號 l 的需求作業的收貨作業。
J_{l+n}	編號 l 的需求作業的卸貨作業。
S	螞蟻 k 當下能選擇加入路徑中的所有需求作業的集合(滿足作業點服務時間窗限制、容量限制與總作業時間限制的需求作業)。
$P_{I,J}^k(t)$	螞蟻 k 在第 t 回合時將作業點 l 插入節線 I，且將作業點 $l+n$ 插入節線 J 的適合度機率分配。
$P_I^k(t)$	螞蟻 k 在第 t 回合時，將作業點 l 與作業點 $l+n$ 同時插入節線 I 的適合度機率分配。

懲罰的概念，使得成本愈高的路段前進機率值愈小，以避免將成本較高的路段加入可行解中。

$$\Delta\eta = 1/(\Delta d + \lambda \cdot \Delta p) \tag{15}$$

其中　$\Delta\eta$ 為 GAACS 演算法意念函數

Δd 為路段成本總變量

Δp 為特徵懲罰值總變量

總變量指的是 (新路徑成本－舊路徑成本) 的成本變量。應用最小成本插入法求解 SPDP 問題時，可能將需求 l (包括收貨作業 J_l 與卸貨作業 J_{l+n}) 同時插入一條路徑 (節線) 中，也可能個別插入不同路徑 (節線) 中，不同的插入法，$\Delta\eta$ 、Δd 與 Δp 的計算公式亦稍有差異。

(1) 需求 l 同時插入一條路徑 (節線 i)

$$\Delta\eta_I^l(t) = 1/(\Delta d_I^l + \lambda \cdot \Delta p_I^l) \tag{16}$$

其中 $\Delta d_I^l = d_{i1,l} + d_{l,l+n} + d_{l+n,i2} - d_{i1,i2}$

$\Delta p_I^l = p_{i1,l} + p_{l,l+n} + p_{l+n,i2} - p_{i1,i2}$

(2) 需求 l 插入不同路徑(節線 i 與 節線 j)

$$\Delta\eta_{IJ}^l(t) = 1/(\Delta d_{IJ}^l + \lambda \cdot \Delta p_{IJ}^l) \tag{17}$$

其中 $\Delta d_{IJ}^l = d_{i1,l} + d_{l,i2} - d_{i1,i2} + d_{j1,l+n} + d_{l+n,j2} - d_{j1,j2}$

$\Delta p_{IJ}^l = p_{i1,l} + p_{l,i2} - p_{i1,i2} + p_{j1,l+n} + p_{l+n,j2} - p_{j1,j2}$

2. GAACS 演算法前進機率值

SPDP 問題中選擇適合加入路徑中的作業時，需同時考慮收貨作業以及卸貨作業的合適程度。然而收、卸貨作業是否以「連續服務」的方式加入路徑中，與計算路段意念函數時相同，在適合度計算方式，亦會稍有不同，如式 (18) 與 (19) 所示，使用符號定義列於表 3。

(1) 需求 l 插入一條路徑 (節線 i)

兩個作業點以連續服務的方式加入現有路徑中，只可能在現有路徑中選擇任一條節線 i 插入該需求作業，且收貨作業必須在卸貨作業前完成，因此本研究另外以 $P_I^k(t)$ 表示螞蟻 k 在第 t 回合時將作業點 l 與作業點 $l+n$ 同時插入節線 I 的適合度機率分配，計算方式如式 (18) 所示。

$$P_I^k(t) = \frac{(\Delta\tau_I^l(t))^\alpha (\Delta\eta_I^l(t))^\beta}{\sum_{h\in S}[(\Delta\tau_I^h(t))^\alpha (\Delta\eta_I^h(t))^\beta]} \tag{18}$$

其中 $\Delta\tau_I^l(t) = \tau_{i1,l}(t) + \tau_{l,l+n}(t) + \tau_{l+n,i2}(t) - \tau_{i1,i2}(t)$

(2) 需求 l 插入不同路徑 (節線 i 與節線 j)

收、卸貨作業以不連續服務的方式，加入螞蟻 k 當下所行走的路徑中，將收貨作業 l 插入節線 i，且將卸貨作業 $l+n$ 插入節線 j，計算方式如式 (19) 所示。

$$P_{IJ}^k(t) = \frac{(\Delta\tau_{IJ}^l(t))^\alpha (\Delta\eta_{IJ}^l(t))^\beta}{\sum_{h\in S}[(\Delta\tau_{IJ}^h(t))^\alpha (\Delta\eta_{IJ}^h(t))^\beta]} \tag{19}$$

其中 $\Delta\tau_{IJ}^{l}(t)=\tau_{t1,l}(t)+\tau_{l,i2}(t)-\tau_{i1,i2}(t)+\tau_{j1,l+n}(t)+\tau_{l+n,j2}(t)-\tau_{j1,j2}(t)$

計算作業點插入機率值時，要特別注意，並非是計算 SPDP 問題各個子作業的插入機率值，而是針對作業點來計算 (原 PDPTW 問題中的作業點)，所以儘管 SPDP 中的子作業決策變數倍增於原 PDPTW 問題中作業點決策變數，卻完全不會增加機率值的計算步驟。

3. 參數 α 與參數 β 的變動機制

GAACS 中參數 α 與參數 β 的設計方式，採用 AACS 中最佳的 α 參數變動策略[2]，求解過程中 β 參數值為固定值，而 α 參數值則隨著求解回合數遞增或遞減，最適參數值測試資料詳列於 4.1 小節。

3.2 導引式費洛蒙更新機制

GAACS 演算法中使用傳統旅行成本矩陣，記錄作業點間的路段費洛蒙濃度與成本，請注意！是記錄作業點之間，不是子作業點之間。演算法中並不記錄子作業點之間的旅行成本，僅利用關連式旅行網絡結構表，記錄子作業點之間的連結關係，因此，儘管 SPDP 問題中的路段決策變數，倍增於原 PDPTW 問題的路段決策變數，但並不會比求解 PDPTW 問題時增加額外的計算過程。

1. GAACS 費洛蒙全域更新機制

近年所提出的各式改良式螞蟻演算法所使用的費洛蒙全域更新方式雖不盡相同，但是有一個共通點，就是一個可行解累積在該可行解所包含路段上的費洛蒙濃度值，無論該路段長度如何，累積值皆是相同的。

本研究將 GLS 演算法，應用特徵懲罰值增加較差路段成本的概念，整合於 AACS 演算法中，設計「導引費洛蒙更新機制」，如式 (20) 所示，使得成本較高與懲罰值較大的路段的費洛蒙濃度增量，小於成本較低與懲罰值較小路段的費洛蒙濃度增量，希望可以較快累積路網上正確的費洛蒙資訊，以正確突顯出哪些路段的費洛蒙濃度較高，有效縮短求解時間。

$$\tau_{ij}(t+1)=(1-\rho)\tau_{ij}(t)+(\rho-w(t))\tau_{ij}^{\ b}(t)+w(t)\tau_{ij}^{\ gb} \tag{20}$$

其中

$$\Delta\tau_{ij}^{\ gb}=\frac{1}{L^{gb}+(\ \lambda\cdot I_{ij}(s^{gb})\cdot p_{ij}\ )}$$

$$\Delta\tau_{ij}^{\ b}=\frac{1}{L^{b}+(\ \lambda\cdot I_{ij}(s^{b})\cdot p_{ij}\ )}$$

同 AACS 般，採變化型的權重分配方式，其中 $(\rho-w(t))$ 與 $(w(t))$ 分別代表反覆最佳解與目前最佳解在第 t 次反覆時所使用的權重值。變化方式為求解初期給予反覆最佳解較高之權重 $(\rho-w(t))$，以增加搜尋區域的廣度，隨著反覆次數的增加，漸漸給予全域最佳解

路徑較高的權重值 ($w(t)$)，同時漸漸降低反覆最佳解所占之權重 (ρ-$w(t)$)，以增加搜尋解的深度與品質，如此並不會增加新的參數，同時反覆最佳解與目前最佳解所使用的權重設定方式亦較具彈性。

2. GAACS 費洛蒙區域更新機制

AACS 演算法中的費洛蒙即時更新策略 MACS1 相較於 ACS，就解最佳解的平均品質改善績效而言，改善程度平均為 0.44%，較 MACS2 機制 (0.42%) 優，但是 t-test 檢定結果顯示約有顯著改善問題的比率 (54%)，卻差於 MACS2 機制 (58%)，且求解時間的增量(求解時間平均增加約 57.3 秒) 亦倍增於 MACS2 (求解時間平均增加約 25.2 秒)。

就邊際效益而言，MACS1 機制的改善績效並不如 MACS2 機制高，因此在 GAACS 演算法中，為縮短求解時間，改採用原 ACS 演算法中的更新機制，如式(21)所示，且由於 GAACS 費洛蒙全域更新機制中，已整合 GLS 的導引概念，因此 GAACS 費洛蒙區域更新機制則不再多此一舉。但須注意的是，式(21)中 τ_0 的計算方式，將改為利用最小成本插入法所求得的可行解目標值，其值在演算過程中是固定的。

$$\tau_{ij}(t) = (1-\xi)\tau_{ij}(t) + \xi\tau_0 \quad \text{其中} \quad 0 < \xi < 1$$

3.3 導引式路徑改善法

本研究求解問題屬 PDPTW 型態，每次執行作業交換時，皆需考慮一組收、卸貨作業，以及服務時間窗、車容量與服務順序的限制，為避免交換法複雜度太高而降低了求解效率，因此本研究採用路徑間與路徑中節點交換法，配合巨集式啟發式演算法中的門檻接受法，設計導引式路線改善法的演算核心。導引式路線改善法架構如圖 2 所示，包括路徑間 1-0 節點交換法、路徑間 1-1 節點交換法、與路徑中節點交換法。求解 SPDP 問題的方式與作業步驟，分別條列詳述於後。

1. 路徑間 1-0 節點交換法

路徑間 1-0 節點交換法目的旨在減少車輛數，Bent 等學者研究中指出，減少車輛數(路徑數)是減少總路徑成本的關鍵要素之一 [28]，且進行 1-0 節點交換法時，修改目標式如式 (22) 時，較僅考慮較少車輛數與較低路徑成本的目標式而言，更可有效降低車輛數。因此，本研究執行 1-0 交換法時，目標式改以減少指派車輛數為主要目標，加入考慮每條路徑中服務的作業 (節點) 數 ($-\sum_{r\in\sigma}|r|^2$)，將目標式 (22) 修改為式 (23)，僅供路徑間 1-0 節點交換法使用。

$$Min \quad \left(|\sigma|, -\sum_{r\in\sigma}|r|^2, \sum_{r\in\sigma}t(r)\right) \tag{22}$$

其中 σ：車輛集合；|σ|：使用的車輛數；

$|r|$：第 r 條路徑中服務的作業 (節點) 數；

$t(r)$：第 r 條路徑的路徑成本。

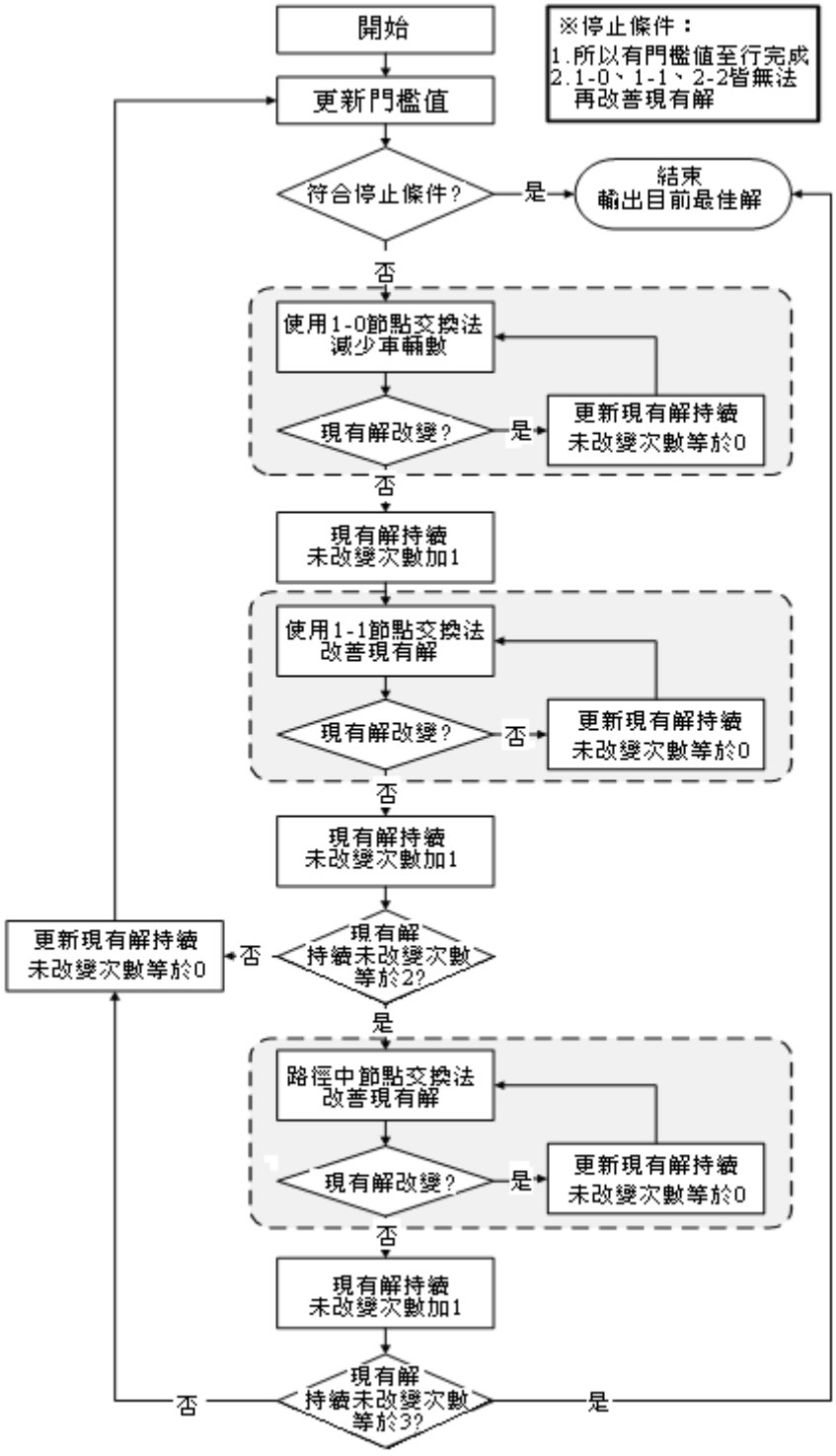

圖 2　導引式區域搜尋法改善 SPDP 問題可行解架構圖

$$Min \qquad \sum_{k=1}^{m}\sum_{u=1}^{2n}\sum_{i=0}^{f(u)} X_{0,ui}^{k} - \sum_{r\in\sigma}|r|^{2} \tag{23}$$

2. 路徑間 1-1 節點交換法

1-1 交換法是將目前路徑中的一組需求 (作業)，與其他路徑 (標的路徑) 中的一組需求 (作業) 交換。本研究為避免執行效率低落，以及避免搜尋過程快速落入區域最佳解，因此路徑間 1-1 節點交換法，並非選擇所有需求 (作業) 中成本增量最小的那一個進行交換，而是只針對標的路徑中成本增量最小的需求 (作業) 進行交換，而交換的必要條件必須同時滿足車容量限制，以及成本增量必須小於或等於門檻值，如此反覆執行至目前路徑中的需求 (作業) 皆考慮過後，即換下一條標的路徑重複執行相同作業。

3. 路徑中節點交換法

路徑中節點交換法，將目前路徑中的二組需求 (作業) 的服務順序重置。目的在於改善現有路徑的成本，以及增加 1-0 交換與 1-1 交換產生新可行解的機會。

四、調適型導引螞蟻演算法求解績效測試

4.1 演算法參數測試實驗設計

本研究利用學者 Lau 與 Liang [16] 以及李泰琳等人 [2] 提出的方法，將 Solomon 國際標準題庫中的 VRPTW 問題轉化，產生 18 題問題規模皆為 100 個節點，且具有相同標竿解的 SPDP 問題後求解，問題轉換方式詳述於李泰琳等人 [2] 研究中。

GAACS 演算法中需設定的參數眾多，最適參數設定值與最適值測試方式及結果，各別條列於下。

(一) 求解回合數與每回合螞蟻數量參數設定

1. 每回合螞蟻數量

Dorigo 等人 [3,60] 研究中指出，每一求解回合中使用的螞蟻數量不宜過多，原因在於每一隻螞蟻的路徑皆用來更新路徑費洛蒙濃度，最佳路徑之資訊容易被路徑不佳的螞蟻混淆。作者過去研究 [5] 應用 AACS 配合 25 隻螞蟻反覆 200 回合，求解 400 節點以內之 TSP 問題，績效優異。學者 Dorigo 等人 [3,60] 與 Gambardella 及 Dorigo 之研究 [61,62] 應用 ACO 求解數百節點之 TSP 問題，使用螞蟻數亦介於 15~30 之間。PDPTW 的複雜度雖較 TSP 高，但求解 100 節點之問題，規模較小，每回合使用 25 隻螞蟻 (總節點數的 1/4) 確已足夠，因此，每回合使用的螞蟻數即設定為 25 隻。

2. 求解回合數

應用螞蟻演算法求解 PDPTW 問題與 TSP 問題時，皆是利用路網上所累積的歷史較佳可行解資訊，搜尋新的近似最佳解。李泰琳等人 [5] 過去研究，應用 AACS 配合 25 隻螞蟻反覆 200 回合，求解 400 節點以內之 TSP 問題，績效優異。因此，求解 100 節點之 PDPTW 問題，規模較小，每回合固定使用 25 隻螞蟻，求解反覆回合數設定為 200 回合，累積在路網上的較佳可行解資訊應已足夠，故合理設定最大求解回合數為 200 回合。

(二) 導引式路徑構建機制相關參數設定

導引式路徑構建機制計算式如式 (18) 與式 (19)，式中需設定的參數值有 α、β，與計算 GLS 中擴增函數時須設定的參數 λ。

1. λ 權重值

λ 權重值是影響 GLS 求解績效的關鍵，λ 值較小時，雖可增加各個解空間的搜尋深度，但演算法求解時間則相對較長，若 λ 值偏大，則搜尋鄰域雖較廣，但搜尋到的區域最佳解品質卻較差，因此測試適當的參數值，對提升演算法績效是相當重要的 [45]。

學者 Kilby 等 [40] 求解有時間窗與車容量限制的 VRP 問題，測試所得的最適 λ 參數值為 0.2，Kilby 等後續研究 [58]，再將 GLS 法應用於時間窗限制車輛路線問題 (VRPTW)；同年 Voudouris 與 Tsang [45] 結合 GLS 與 FLS 求解旅行推銷員問題 (TSP)，λ 參數建議值為 0.3；Mills 研究中針對不同類型問題，λ 權重變化有相當詳細的測試，並使用固定與遞增兩種方式設計 λ 權重值，測試範圍從 0.1 至 100 [42]；隔年 Voudouris 與 Tsang [43] 以 TSP 問題為例，採用式 (2.19) 的設定方式產生 λ 權重值，其中參數 a 之值，對 2-Opt 而言，建議為 $1/8 \le a \le 1/2$。但目前尚無學者們針對 PDPTW 問題，測試適用的 λ 權重參數值。

綜觀過去學者求解 VRP、VRPTW、與 TSP 等 [40,42,43,45,58] 車輛途程問題，建議使用的 λ 權重值皆偏小 (≤ 0.3)，而螞蟻演算法又是屬於深度搜尋能力較差的演算法，因此配合使用的 λ 權重值不宜偏大。本研究測試 λ 權重值之範圍為 $0 < \lambda \le 0.5$，由於參數組合數量龐大，因此僅測試 0.01、0.1、0.3、0.5 等四種參數值，測試時採用固定式 λ 權重值，即 λ 值在各回合的搜尋過程中始終是固定的，而後續求解時亦同。

四組 λ 權重與其他參數綜合測試共 96 組參數值，每一問題皆測試 30 次取平均值，平均而言，λ 權重值等於 0.1 時的績效表現最佳，亦印證了前述對 λ 權重參數值設定之推論。由於參數測試之數據資料過多，文中僅篩選其中績效表現最佳的 10 組參數組合，如表 4 所示，完整測試數據歡迎來信索取。

2. α 參數與 β 參數值設定

α 參數與 β 參數值設定方式，採用 AACS 中最佳的 α 參數變動策略。但由於 AACS 建議的最適參數值為求解 TSP 問題的最適值，應用於 GAACS 演算法中求解 PDPTW 問題時，可能適合度稍差，因此以過去文獻中針對 ACS 所建議的最適 α 參數設定值 ($\alpha = 0.1$)，與 β

參數設定值 ($\beta = 2$)，為基礎 [3,60]，重新設計數種變動組合，測試適用於 GAACS 演算法的 α 參數與 β 參數。變動方式條列於下：

(1) α 以 0.1 為中間值，設定 0.01 ~ 0.2 變動區間；β 以 2 為中間值，令 β 值為 1、2、3，設計 3 種變動方式：($\alpha = 0.2 \sim 0.01, \beta = 1$)、($\alpha = 0.2 \sim 0.01, \beta = 2$) 與 ($\alpha = 0.2 \sim 0.01, \beta = 3$)。

(2) 設定參數 α 等於 β，並隨求解回合逐漸遞減至 0.1，β 值等於過去文獻中所建議的最適設定值 ($\alpha = 2 \sim 0.1, \beta = 2$)。

(3) 設定參數 α 等於 β，並隨求解回合逐漸遞減至 0.2，β 值等於過去文獻中所建議的最適設定值 ($\alpha = 2 \sim 0.2, \beta = 2$)。

(4) 設定參數 α 等於 β，並隨求解回合逐漸遞減至 0.01，β 值等於過去文獻中所建議的最適設定值 ($\alpha = 2 \sim 0.01, \beta = 2$)。

(三) 導引式費洛蒙全域更新機制相關參數設定

導引式費洛蒙全域更新機制如式 (20) 所示，需設定的參數包括 ρ 與 $w(t)$，其中 ρ 值設定方式，採用各式改良式 ACO 與 AACS 皆建議的最適參數值 $\rho = 0.1$；而 $w(t)$ 則以 AACS 路徑費洛蒙全域更新機制 (MACS2) 所建議之設定值為基礎，測試 $w(t)$ 等於 5、7、9、與 11 等 4 種數值。其原因為 MACS2 中的權重變化長度值 $w(t)$ 等於 5 與 9 時，求解 TSP 問題的較佳解平均績效普遍較佳，而長度值為 9 時所求得之近似最佳解品質亦明顯較佳，與題庫標竿解差距也最小。

(四) 其他重要參數設定

1. 費洛蒙即時更新機制採用原 AACS 的設計，需設定的參數值包括 ξ 與 τ_0，皆採用 AACS 建議的最適參數值，$\xi = 0.1$，而 τ_0 為利用最小成本插入法所求得的可行解目標值，其值在演算過程中是固定的 。
2. 導引式路徑改善法中使用的門檻值，採用表 3.13 測試結果。起始門檻值為回合最佳解的 1.5%，門檻長度值為 30。
3. 演算法的主要停止條件，為每一回合所得之近似最佳解持續總反覆次數的 1/5 次皆未見改善時，即停止演算法繼續執行並輸出目前最佳解，而執行時間與反覆次數則依序作為次要停止條件，每一問題的求解時間以 60 分鐘為上限。

綜和上述參數設定方式，需測試的參數組合總共有 96 種。另由於 GAACS 演算法中每隻螞蟻產生起始路徑的方式存在隨機的變因，即使使用完全相同的參數設定值求解，每次求解的結果亦不盡相同，因此每一問題皆測試 30 次取平均值。

問題測試之目的旨在區分不同參數組合的求解績效，並非求取品質極佳之近似最佳解，因此求解反覆回合數減半，僅反覆 100 次，每回合使用之螞蟻數量亦減少為 10 隻，

求解時間改以 30 分鐘為上限，並以每一問題求解 30 次後的近似解平均值與標竿解之間的差異百分比進行分析比較。

本研究將平均求解績效較佳的前 10 組參數組合列於表 4，可以看出，近似最佳解平均值與標竿解之間的平均差異均小於 1.87%，而平均求解績效最佳的參數組合為編號 59 ($\alpha = 2\sim0.01,\ \beta = 2,\ \lambda = 0.1,\ W(t) = 9$)，近似最佳解平均值與標竿解之間的平均差異僅 1.80%。研究後續將利用此組參數，完整執行 GAACS 演算法，測試 GAACS 演算法的求解績效。

表 4　較適參數組合測試結果表

參數 / 題目	*α= 2-0.01*	*α= 2-0.01*	*α= 2-0.01*	*α= 2-0.01*	*α= 2-0.2*	*α= 0.2-0.01*	*α= 0.2-0.01*	*α= 0.2-0.01*	*α= 2-0.2*	*α= 2-0.1*
	β=2	*β=2*	*β=2*	*β=2*	*β=2*	*β=3*	*β=1*	*β=2*	*β=2*	*β=2*
	λ=0.1	*λ=0.3*	*λ=0.1*	*λ=0.3*	*λ=0.1*	*λ=0.01*	*λ=0.1*	*λ=0.3*	*λ=0.5*	*λ=0.5*
	W(t)=9	*W(t)=9*	*W(t)=7*	*W(t)=7*	*W(t)=9*	*W(t)=9*	*W(t)=9*	*W(t)=9*	*W(t)=9*	*W(t)=9*
R103_1	0.31%	0.40%	0.33%	0.40%	0.45%	0.36%	0.14%	0.55%	0.38%	0.31%
R103_2	0.60%	0.58%	0.62%	0.58%	0.53%	0.35%	0.51%	0.32%	0.58%	0.44%
R103_3	0.57%	0.45%	0.59%	0.45%	0.43%	0.40%	0.41%	0.50%	0.41%	0.46%
R112_1	2.32%	2.48%	2.34%	2.49%	2.41%	2.43%	2.42%	2.56%	2.54%	2.45%
R112_2	2.56%	2.31%	2.58%	2.32%	2.46%	2.25%	2.25%	2.37%	2.45%	2.38%
R112_3	2.61%	2.62%	2.61%	2.65%	2.70%	2.51%	2.61%	2.61%	2.56%	2.51%
R208_1	1.10%	1.10%	1.10%	1.12%	1.15%	1.25%	1.25%	1.27%	1.37%	1.36%
R208_2	1.10%	1.10%	1.10%	1.14%	1.17%	1.31%	1.10%	1.31%	1.13%	1.15%
R208_3	1.10%	1.10%	1.10%	1.23%	1.16%	1.16%	1.10%	1.24%	1.41%	1.40%
R210_1	3.22%	3.01%	3.27%	3.05%	3.35%	3.70%	3.55%	3.41%	3.87%	3.56%
R210_2	3.18%	3.25%	3.23%	3.30%	3.25%	3.65%	3.51%	3.85%	3.22%	3.12%
R210_3	2.70%	2.65%	2.74%	2.65%	2.84%	2.65%	2.74%	2.65%	2.85%	2.68%
RC204_1	2.78%	2.55%	2.92%	2.55%	2.86%	2.55%	3.16%	2.55%	2.64%	2.63%
RC204_2	3.04%	3.34%	3.08%	3.35%	3.21%	3.19%	2.95%	3.17%	3.01%	3.15%
RC204_3	3.18%	3.23%	3.27%	3.25%	3.28%	3.18%	3.49%	3.20%	3.35%	3.25%
RC205_1	1.19%	1.21%	1.19%	1.23%	0.91%	1.26%	1.52%	1.15%	0.74%	1.64%
RC205_2	0.39%	0.69%	0.41%	0.71%	0.39%	0.42%	0.39%	0.36%	0.41%	0.35%
RC205_3	0.48%	0.66%	0.50%	0.68%	0.61%	0.74%	0.45%	0.54%	0.75%	0.84%
平均%	1.80%	1.82%	1.83%	1.84%	1.84%	1.85%	1.86%	1.87%	1.87%	1.87%

4.2 演算法最適參數分析

本章節中利用表 4 中的參數組測試結果，分析較適合的參數 α 與 β 組合與參數值 (表 5)、MACS2 機制中較適合的權重變化長度值 $w(t)$ (表 6) 與 GLS 懲罰值權重 λ 的較適設定值 (表 7)。

表 5 中數值為各個 α 與 β 參數組合，配合不同的權重變化長度值 $w(t)$ 與 GLS 懲罰值權重 λ，所求得之近似最佳解與標竿解間的平均差異百分比，例如第 2 欄中問題 R103_1 之近似最佳解平均值與標竿解間差異 0.561%，意即為 α 與 β 參數組合 ($\alpha = 0.2$~0.01, $\beta = 1$)，配合其他參數設定值 ($w(t) = 5, 7, 9, 11$; $\lambda = 0.01, 0.1, 0.3, 0.5$) 所產生的所有參數組求解 R103_1 的平均值。表 5 中的其他數據值計算方式，以此類推。

表 5　GAACS 適合的 α 與 β 參數組合

題目名稱	α 與 β 參數組合					
	$\alpha = 0.2$~0.01	$\alpha = 0.2$~0.01	$\alpha = 0.2$~0.01	$\alpha = 2$~0.01	$\alpha = 2$~0.1	$\alpha = 2$~0.2
	$\beta = 1$	$\beta = 2$	$\beta = 3$	$\beta = 2$	$\beta = 2$	$\beta = 2$
R103_1	0.561%	0.656%	0.500%	0.502%	0.586%	0.404%
R103_2	0.727%	0.670%	0.672%	0.669%	0.730%	0.667%
R103_3	0.532%	0.670%	0.636%	0.613%	0.668%	0.506%
R112_1	2.637%	2.697%	2.688%	2.622%	2.724%	2.528%
R112_2	2.552%	2.460%	2.580%	2.590%	2.569%	2.445%
R112_3	2.680%	2.674%	2.554%	2.780%	2.662%	2.664%
R208_1	1.254%	1.224%	1.251%	1.176%	1.281%	1.267%
R208_2	1.180%	1.286%	1.324%	1.208%	1.238%	1.209%
R208_3	1.210%	1.211%	1.202%	1.195%	1.348%	1.351%
R210_1	3.646%	3.730%	3.643%	3.470%	3.591%	3.624%
R210_2	3.803%	3.839%	3.636%	3.549%	3.513%	3.312%
R210_3	2.805%	2.813%	2.756%	2.990%	2.872%	2.874%
RC204_1	3.423%	3.603%	3.152%	3.119%	3.510%	2.987%
RC204_2	3.190%	3.190%	3.213%	3.306%	3.269%	3.139%
RC204_3	3.396%	3.387%	3.378%	3.338%	3.307%	3.425%
RC205_1	1.371%	1.536%	1.565%	1.052%	1.600%	1.266%
RC205_2	0.407%	0.422%	0.436%	0.485%	0.414%	0.453%
RC205_3	0.814%	0.882%	0.937%	0.629%	1.186%	0.822%
平均	2.010%	2.053%	2.007%	1.961%	2.059%	1.941%

表 5 分析結果可以看出，α 與 β 參數組合以 $(\alpha = 2\sim0.2, \beta = 2)$ 的平均求解績效最佳，$(\alpha = 2\sim0.01, \beta = 2)$ 次之，然而表 4 中平均求解績效最佳的前 4 組 α 與 β 參數組合皆為 $(\alpha = 2\sim0.01, \beta = 2)$，因此本研究建議使用 α 與 β 參數組合 $(\alpha = 2\sim0.01, \beta = 2)$ 配合 GAACS 演算法求解 SPDP 問題。

應用 α 變動策略求解其它規模及其他類型之 PDP 問題時，則建議以 $(\alpha = 2\sim0.2, \beta = 2)$ 與 $(\alpha = 2\sim0.01, \beta = 2)$ 兩組參數設定值為基礎，來設計需測試的參數值，應可減少後續相關研究需測試的參數範圍與規模。

表 6 中數值為各個權重變化長度值 $w(t)$，配合不同的 α 與 β 參數組合及 GLS 懲罰值權重 λ，所求得之近似最佳解與標竿解間的平均差異百分比，第 2 欄中問題 R103_1 之近似最佳解平均值與標竿解間差異 0.57%，為權重變化長度值 $w(t) = 5$，配合其他 24 組參數設定值 (6 組 α 與 β 參數組合及 4 組 λ 值) 所產生的所有參數組所求得的平均值。表 6 中其他數據值計算方式，以此類推。

表 6　GAACS 中 MACS2 機制適合的權重變化長度值 $w(t)$

題目名稱	權重變化長度值 $w(t)$			
	5	7	9	11
R103_1	0.570%	0.502%	0.477%	0.592%
R103_2	0.711%	0.667%	0.652%	0.727%
R103_3	0.633%	0.576%	0.550%	0.659%
R112_1	2.675%	2.625%	2.613%	2.685%
R112_2	2.577%	2.487%	2.464%	2.602%
R112_3	2.705%	2.635%	2.620%	2.716%
R208_1	1.265%	1.217%	1.201%	1.285%
R208_2	1.267%	1.207%	1.195%	1.294%
R208_3	1.282%	1.210%	1.185%	1.334%
R210_1	3.658%	3.580%	3.550%	3.681%
R210_2	3.656%	3.569%	3.525%	3.685%
R210_3	2.884%	2.813%	2.787%	2.923%
RC204_1	3.392%	3.182%	3.083%	3.538%
RC204_2	3.273%	3.177%	3.146%	3.276%
RC204_3	3.425%	3.323%	3.269%	3.470%
RC205_1	1.443%	1.358%	1.332%	1.461%
RC205_2	0.464%	0.405%	0.395%	0.481%
RC205_3	0.898%	0.856%	0.843%	0.916%
平均	2.043%	1.966%	***1.938***%	2.074%

從表 6 分析結果可以看出，$w(t)=9$ 的平均求解績效最佳，$w(t)=7$ 次之，96 組測試參數組合中求解績效最佳的前 10 組中，有 8 組為 $w(t)=9$，績效最佳的亦為 $w(t)=9$，因此本研究建議使用 $w(t)=9$ 配合 GAACS 演算法求解 SPDP 問題。

表 7 中數值為各個懲罰值權重 λ，配合不同的 α 與 β 參數組合及 MACS2 之權重變化長度值 $w(t)$，所求得之近似最佳解與標竿解間的平均差異百分比，第 2 欄中問題 R103_1 之近似最佳解平均值，與標竿解間差異 0.541%，為權重變化長度值 $\lambda=0.01$，配合其他 24 組參數設定值 (6 組 α 與 β 參數組合及 4 組 $w(t)$ 值) 所產生的所有參數組所求得的平均值。表 7 中的其他數據值計算方式，以此類推。

表 7　GAACS 適合的 GLS 懲罰值權重 λ

題目名稱	GLS 懲罰值權重 λ			
	0.01	0.1	0.3	0.5
R103_1	0.541%	0.522%	0.580%	0.495%
R103_2	0.630%	0.670%	0.714%	0.743%
R103_3	0.582%	0.596%	0.601%	0.638%
R112_1	2.634%	2.624%	2.650%	2.690%
R112_2	2.547%	2.513%	2.494%	2.577%
R112_3	2.742%	2.646%	2.662%	2.627%
R208_1	1.246%	1.206%	1.237%	1.280%
R208_2	1.270%	1.232%	1.234%	1.228%
R208_3	1.223%	1.224%	1.289%	1.275%
R210_1	3.636%	3.515%	3.562%	3.757%
R210_2	3.810%	3.399%	3.645%	3.581%
R210_3	2.955%	2.803%	2.822%	2.826%
RC204_1	3.306%	3.179%	3.258%	3.452%
RC204_2	3.199%	3.217%	3.230%	3.225%
RC204_3	3.377%	3.332%	3.383%	3.396%
RC205_1	1.448%	1.444%	1.447%	1.254%
RC205_2	0.407%	0.438%	0.504%	0.395%
RC205_3	0.930%	0.911%	0.854%	0.818%
平均	2.027%	***1.971***%	2.009%	2.014%

從表 7 分析結果可以看出，$\lambda = 0.1$ 的平均求解績效最佳，$\lambda = 0.3$ 次之，96 組測試參數組合中，平均求解績效最佳的前 5 組中，有 3 組為 $\lambda = 0.1$，績效最佳的亦為 $\lambda = 0.1$，因此本研究建議使用 $\lambda = 0.1$ 配合 GAACS 演算法求解 SPDP 問題。

4.3 演算法求解績效分析

本章節旨在測試 GAACS 求解 SPDP 問題近似最佳解的績效，GAACS 演算法中所使用的參數設定值，為前述建議之最佳參數組合 ($\alpha = 2 \sim 0.01, \beta = 2, \lambda = 0.1, W(t) = 9$)，求解時間每一題以 60 分鐘為上限，每一題求解 30 次，求解數據結果詳列於表 8。

表 8　GAACS 求解績效分析結果表

參數	α 值	β 值	λ 值	$W(t)$值	回合數	螞蟻數
	2 ~ 0.01	2	0.1	9	200	25
欄名 / 題目	標竿解 (車輛) (A)	近似最佳解 (車輛) (B)	近似最佳解平均值 (平均車輛) (C)	求得近似最佳解時間 (分)	近似最佳解差異% (B-A)/A%	近似最佳解平均差% (C-A)/A%
PDPTWR103_30_1	1292.68 (13)	1292.68 (13)	1296.66 (13)	10.48	0.00%	0.31%
PDPTWR103_30_2	1292.68 (13)	1292.68 (13)	1300.44 (13)	11.27	0.00%	0.60%
PDPTWR103_30_3	1292.68 (13)	1292.68 (13)	1300.03 (13)	10.00	0.00%	0.57%
PDPTWR112_30_1	982.14 (9)	1003.77 (9)	1004.94 (9)	18.78	2.20%	2.32%
PDPTWR112_30_2	982.14 (9)	1003.77 (9)	1007.30 (9)	16.38	2.20%	2.56%
PDPTWR112_30_3	982.14 (9)	1003.77 (9)	1007.76 (9)	17.90	2.20%	2.61%
PDPTWR208_30_1	726.82 (2)	734.848 (2)	734.88 (2)	38.66	1.10%	1.11%
PDPTWR208_30_2	726.82 (2)	734.848 (2)	734.98 (2)	43.80	1.10%	1.12%
PDPTWR208_30_3	726.82 (2)	734.848 (2)	734.86 (2)	45.90	1.10%	1.11%
PDPTWR210_30_1	939.37 (3)	964.22 (3)	969.60 (3.167)	28.32	2.65%	3.22%
PDPTWR210_30_2	939.37 (3)	964.22 (3)	969.21 (3.067)	29.76	2.65%	3.18%

表 8　GAACS 求解績效分析結果表（續）

參數	α 值	β 值	λ 值	$W(t)$ 值	回合數	螞蟻數
	2 ~ 0.01	2	0.1	9	200	25
欄名 / 題目	標竿解 (車輛) (A)	近似最佳解 (車輛) (B)	近似最佳解平均值 (平均車輛) (C)	求得近似最佳解時間 (分)	近似最佳解差異% (B-A)/A%	近似最佳解平均差% (C-A)/A%
PDPTWR210_30_3	939.37 (3)	964.22 (3)	964.70 (3)	29.21	2.65%	2.70%
PDPTWRC204_30_1	798.46 (3)	818.663 (3)	820.63 (3.03)	44.59	2.53%	2.78%
PDPTWRC204_30_2	798.46 (3)	818.663 (3)	822.73 (3.1)	42.69	2.53%	3.04%
PDPTWRC204_30_3	798.46 (3)	818.663 (3)	823.86 (3.267)	38.18	2.53%	3.18%
PDPTWRC205_30_1	1297.65 (4)	1302.2 (4)	1313.06 (4.2)	14.89	0.35%	1.19%
PDPTWRC205_30_2	1297.65 (4)	1302.2 (4)	1302.65 (4)	14.14	0.35%	0.39%
PDPTWRC205_30_3	1297.65 (4)	1302.2 (4)	1303.89 (4)	12.24	0.35%	0.48%
平均值				25.95	1.47%	1.80%

本研究提出的 GAACS 求解 PDPTW 時，可在 25.9 分鐘平均時間內，求得與標竿解平均差異僅 1.47%之近似最佳解，其中 16%的問題可求得相同標竿解；33.3%的問題與標竿解平均差異小於 0.35%；50%的問題與標竿解平均差異小於 1.11%，該研究成果對後續學者進行 PDPTW 相關研究應有不少助益。

五、結論與建議

5.1 結論

PDPTW 問題已證實屬於 NP-hard 問題，大規模問題無法在合理時間內求得精確解，過去學者多發展啟發式演算法求解，本研究則應用改良式螞蟻演算法，結合新興的導引區域搜尋技術 (GLS)，設計調適型導引螞蟻演算法 (GAACS) 求解 SPDP 問題。研究中所使用的改良式螞蟻演算法，為詳細分析近年較著名的數項改良式螞蟻演算法的優缺點後，所設計的調適型螞蟻演算法 (AACS) [5]。

螞蟻演算法相較於其他組合優選演算法雖花費較長的時間，然而卻可以找出較佳的求解品質。近年使用螞蟻演算法求解 PDP 相關問題之文獻極少，Doerner 等 [63] 學者應用螞蟻演算法，求解 512 個需求的收送貨問題 (PDP)，研究中顯示螞蟻數量愈多，得到的近似最佳解品質愈高，但求解時間亦相對拉長，求解時間介於 33 分鐘至 772 分鐘之間。

本研究利用國際標準題庫 VRPTW 例題進行修改，產生具有標竿解之 PDPTW 例題 18 題進行測試，問題規模皆為 100 個節點。PDPTW 的問題複雜度較 VRPTW 高，若應用求解 PDPTW 所得的解品質與求解 VRPTW 時的解品質相比較，實過於嚴苛，然而測試結果顯示，本研究將 PDPTW 問題轉換為 SPDP 問題後求解，確實可以在較短的時間內求出高品質的近似最佳解，大幅減少求解時間。相關研究結果與測試數據分析結果條列如下：

1. GAACS 可求出與標竿解間平均差異僅 1.47%的近似最佳解，近似最佳解平均值與標竿解間的差異僅 1.8%，其中 R103 題型皆可求得標竿解相同品質的近似最佳解。
2. GAACS 求解問題規模 100 個節點的問題時，求得近似最佳解所需平均時間僅 25.95 分鐘。
3. 本研究方法所得結果，平均 16%的問題可求得相同標竿解；33.3%的問題與標竿解平均差異小於 0.35%；50%的問題與標竿解平均差異小於 1.11%。
4. 近似最佳解品質合理可接受，求解時間亦相對減少很多，顯示將 GLS 導入螞蟻演算法確實可以提高求解績效。研究成果對後續學者進行螞蟻演算法應用與 PDPTW 求解等相關研究，應可提供相當的貢獻。

5.2 建議

1. 本研究所設計的 AACS 啟發式演算法，求解 TSP 問題的績效確實優於 ACS 演算法，而以 AACS 為基礎，配合 GLS 所設計的 GAACS 啟發式演算法，求解 PDPTW 問題時，亦可在 25.95 分鐘的平均時間下，求出與標竿解間平均差異僅 1.47%的近似最佳解。但是 AACS 與 GAACS 兩啟發式演算法求解其他路徑規劃問題的適用性與求解績效尚屬未知，建議後續研究可進行探討。
2. ACO 是近年新興的演算法，已陸續成功地應用於求解多種複雜組合最佳化問題，本研究亦應用 ACO 中的 AACS 為基礎，設計 GAACS 求解 SPDP 問題，然而 ACO 本身屬於需要較長求解時間的演算法，PDPTW 問題的複雜度亦較高，相對所需的求解時間也較長，即使轉換為 SPDP 問題後，每隻螞蟻求得可行解的時間已大幅降低，但由於 ACO 需要足夠的求解回合與螞蟻數量，因此求解時間卻又相對增加，建議後續學者應用 ACO 演算法求解 SPDP 問題時，可配合平行處理技術，降低求解時間。

參考文獻

1. Fabian, J. and Perez, L., “A Meta-Heuristic Applied for a Topologic Pickup and Delivery Problem with Time Windows Constraints”, *Lecture Notes in Computer Science*, Vol. 3516, 2005,

pp. 924-928.

2. 李泰琳、張 靖，「應用時窗分割與整數化策略簡化時窗收卸貨問題之研究」，**運輸學刊**，第 20 卷，第 3 期，民國 97 年，頁 313-350。
3. Dorigo, M. and Gambardella, L. M., “Ant Colony System: A Cooperative Learning Approach to the Traveling Salesman Problem”, *IEEE Transactions on Evolutionary Computation*, Vol. 1, No. 1, 1997, pp.53-66.
4. Dorigo, M., “Optimization, Learning, and Natural Algorithms”, Ph.D. Thesis, Dipartimento di Elettronica e Informatica, Politecnico di Milano, Italy, 1992.
5. 李泰琳、張 靖、卓裕仁，「調適型螞蟻演算法應用於旅行推銷員問題之研究」，**運輸學刊**，第 19 卷，第 1 期，民國 96 年，頁 89-120。
6. Bullnheimer, B., Hartl, R. F.,and Strauss, C., “A New Rank Based Version of the Ant System - A Computational Study”, *Central European Journal for Operations Research and Economics*, Vol. 7, No. 1, 1999, pp. 25-38.
7. Le Louarn, F. X., Gendreau, M., and Potvin, J. Y., “GENI Ants for the Traveling Salesman Problem”, *Annals of Operations Research*, Vol. 131, 2004, pp. 187-201.
8. Stützle, T. and Hoos, H. H., “The MAX-MIN Ant System and Local Search for the Travelling Salesman Problem”, in Bäck, T., Michalewicz, Z., Yao, X. (editors), Proceedings for the IEEE International Conference on Evolutionary Computation (ICEC’97), IEEE press, Piscataway, USA, 1997, pp. 309-314.
9. Stützle, T. and Hoos, H. H., “Improvements on the Ant System: Introducing the MAX-MIN Ant System”, In R. F. Albrecht Q. D. Smith, N. C. Steele (editors), *Artificial Neural Network and Genetic Algorithms*, Springer Verlag, New York, 1998, pp. 245-249.
10. Stützle, T. and Hoos, H. H., “MAX-MIN Ant System and Local Search for Combinatorial Optimization Problems”, *Meta-Heuristics: Advances and Trends in Local Search Paradigms for Optimization*, 1999, pp. 313-329.
11. Tsang, E and Voudouris, C., “Fast Local Search and Guided Local Search and Their Application to British Telecom's Workforce Scheduling Problem”, Technical Report CSM-246, Department of Computer Science, University of Essex, 1995.
12. Mladenović N. and Hansen, P., “Variable Neighborhood Search”, *Computers and Operations Research*, Vol. 24, 1997, pp. 1097-1100.
13. Savelsbergh, M. W. P. and Solomon, M., “The General Pickup and Delivery Problem”, *Transportation Science*, Vol. 29, No. 1, 1995, pp.17-29.
14. Appelgren, L. H., “A Column Generation Algorithm for a Ship Scheduling Problem”, *Transportation Science*, Vol. 3, No. 1, 1969, pp. 53-68.
15. Wang, X., “Algorithms and Strategies for Dynamic Carrier Fleet Operations: Applications to Local Trucking Operations”, Ph.D. Dissertation, Civil Engineering, University of Calllifornia Irvine, 2001.
16. Lau, H. C. and Liang, Z., “Pickup and Delivery with Time Windows - Algorithms and Test Case Generation”, *International Journal on Artificial Intelligence Tools*, Vol. 11, No. 3, 2002, pp.

455-472.

17. Dumas, Y., Desrosiers J., and Soumis F., “The Pickup and Delivery Problem with Time Windows”, *European Journal of Operational Research*, Vol. 54, No. 1, 1991, pp. 7-22.
18. Kohl, N.,“Exact Methods for Time Constrained Routing and Related Scheduling Problems”, Ph.D. Dissertation, Institute of Mathematical Modeling, Technical University of Denmark,1995.
19. Lu, Q. and Dessouly, M., “An Exact Algorithm for the Multiple Vehicle Pickup and Delivery Problem”, *Transportation Science*, Vol. 38, No. 4, 2004, pp. 503-514.
20. 黃信翔、王晉元，「解決具時間窗限制的提送貨問題」，國立交通大學運輸科技與管理學系碩士論文，民國 94 年。
21. 彭冠儒、陳正芳，「混合送貨、收貨的車輛途程問題」，2002 年產業電子化運籌管理學術暨實務研討會，長庚大學，民國 91 年，頁 150-157。
22. 朱經武、劉曄、洪秀華，「同時考慮以自有車輛收送或或委託貨運公司服務之啟發式演算法」，**海運學報**，第 13 期，民國 93 年 10 月，頁 147-162。
23. Psarafis, H., “A Dynamic Programming Solution to the Single Vehicle Many to Many Immediate Request Dial-a-Ride Problem”, *Transportation Science*, Vol.14, No. 2, 1980, pp.130-54.
24. Psarafis, H., “An Exact Algorithm for the Single Vehicle Many-to-Many Immediate Request Dial-a-Ride Problem”, *Transportation Science*, Vol. 17, No. 4, 1983, pp. 351-361.
25. Sexton, T. R. and Bodin, L. D., “Optimizing Single Vehicle Many-to-Many Dial-a-Ride Problem with Desired Delivery Time: I Scheduling”, *Transportation Science*, Vol. 19, No. 4, 1985, pp. 378-410.
26. Sexton, T. R. and Bodin, L. D., “Optimizing Single Vehicle Many-to-Many Dial-a-Ride Problem with Desired Delivery Time: II Routing”, *Transportation Science*, Vol. 19, No. 4, 1985, pp .411-435.
27. Nanry, W. P. and Barnes, J. W., “Solving the Pickup and Delivery Problem with Time Windows Using Reactive Tabu Search”, *Transportation Research Part B*, Vol. 34, 2000, pp. 107-121.
28. Bent, R. and Van Hentenryck, P., “A Two-Stage Hybrid Algorithm for Pickup and Delivery Vehicle Routing Problems with Time Windows”, *Computers and Operations Research*, Vol. 33, No. 4, 2006, pp. 875-893.
29. Mitrovic-Minic, S. and Laporte, G., “Waiting Strategies for The Dynamic Pickup and Delivery Problem with Time Windows”, *Transportation Research Part B*, Vol. 38, 2004, pp. 635-655.
30. Dror, M., “Note on the Complexity of the Shortest Path Models for Column Generations in the VRPTW”, *Operations Research*, Vol. 42, No. 5, 1994, pp. 977-978.
31. Kolen, A. W. J., Rinnooy Kan, A. G. H., and Trienekens, H. W. J. M., “Vehicle Routing with Time Windows”, *Operations Research*, Vol. 35, No. 2, 1987, pp. 266-273.
32. Desrosiers, J., Dumas, Y., Solomon, M. M., and Sournis, E., “Time Constrained Routing and Scheduling”, *Handbooks in Operations Research and Management Science*, Vol. 8, 1995, pp. 35-139.
33. 梅明德、謝浩明，「時窗限制動態車輛路線問題之線上型路線建立啟發式解法」，**運輸學**

刊，第 13 卷，第 2 期，民國 90 年，頁 73-111。

34. 韓復華、王國琛，「巨集式解法在求解大規模旅行推銷員問題之應用」，**運輸學刊**，第 14 卷，第 2 期，民國 91 年，頁 1-14。

35. 林志鴻，「宅配業車輛路線規劃問題之模式建立與求解」，**運輸學刊**，第 17 卷，第 1 期，民國 94 年，頁 65-94。

36. 韓復華、卓裕仁、陳國清，「五種巨集啟發式解法在 VRP 問題上之應用與比較」，中華民國第四屆運輸網路研討會論文集，國立成功大學，民國 88 年，頁 72-82。

37. 陳家和、丁慶榮，「應用螞蟻演算法於時窗限制車輛途程問題之研究」，**運輸學刊**，第 17 卷，第 3 期，民國 94 年，頁 261-280。

38. Voudouris, C. and Tsang, E., “Function Optimization Using Guided Local Search”, Technical Report CSM-249, Department of Computer Science, University of Essex, 1995.

39. Davenport, A., “GENET: A Connectionist Architecture for Constraint Satisfaction”, Ph. D. Thesis, Department of Computer Science, University of Essex, 1997.

40. Kilby, P., Prosser, P., and Shaw, P., “Guided Local Search for the Vehicle Routing Problem”, 2ND International Conference on Metaheuristics – MIC97, Sophia, Antipolis, France, 1997, pp. 1-10

41. Mester, D. and Bräysy, O., “Active Guided Evolution Strategies for the Large Scale Vehicle Routing Problem with Time Windows”, *Computers & Operations Research*, Vol. 32, 2005, pp. 1593-1614.

42. Mills, P., “Extensions To Guided Local Search”, Ph. D. Thesis, Department of Computer Science, University of Essex, 2002.

43. Voudouris, C. and Tsang, E. P. K., “Guided Local Search”, in F. Glover (ed.), *Handbook of Metaheuristics*, Kluwer, Springer, 2003.

44. Voudouris, C. and Tsang, E., “Guided Local Search and Its Application to the Travelling Salesman Problem”, *European Journal of Operational Research*, Vol. 132, No. 2, 1998, pp. 80-110.

45. Voudouris, C. and Tsang, E., “Theory and Methodology Guided Local Search and Its Application to the Traveling Salesman Problem”, E*uropean Journal of Operational Reasearch*, Vol. 113, 1999, pp. 469-499.

46. Voudouris, C. and Tsang, E., “Guided Local Search”, Technical Report CSM-247, Department of Computer Science, University of Essex, 1995.

47. Voudouris, C. and Tsang, E., “Partial Constraint Satisfaction Problems and Guided Local Search”, Technical Report CSM-250, Department of Computer Science, University of Essex, 1995.

48. Voudouris, C. and Tsang, E., “Partial Constraint Satisfaction Problems and Guided Local Search”, Proceedings of 2nd International Conference on Practical Application of Constraint Technology (PACT'96), London, 1996, pp. 337-356.

49. Tsang, E. and Voudouris, C., “Fast Local Search Guided Local Search and Their Application to British Telecom's Workforce Scheduling Problem”, *Operations Research Letters*, Vol. 20, No. 3,

1997, pp. 119-127.

50. Balas, E. and Vazacopoulos, A., “A Guided Local Search with Shifting Bottleneck for Job Shop Scheduling”, *Management Science*, Vol. 44, No. 2, 1998, pp. 262-275.
51. Faroe, O., Pisinger, D., and Zachariasen, M., “Guided Local Search for the Three-Dimensional Bin Packing Problem”, *INFORMS Journal on Computing*, Vol. 15, 2003, pp. 267-283.
52. Voudouris, C. and Tsang, E., “Solving the Radio Link Frequency Assignment Problem Using Guided Local Search”, Proceedings of NATO Symposium on Radio Length Frequency Assignment, Sharing and Conservation Systems (Aerospace), Aalborg, Demark, October 1998.
53. Voudouris, C., “Guided Local Search for Combinatorial Problems”, Ph. D. Thesis, University of Essex, 1997.
54. Lin, S., “Computer Solutions for the Traveling Salesman Problem”, *Bell Systems Technology Journal*, Vol. 44, 1965, pp. 2245-2269.
55. Savelsbergh, M. W. P., *Computer Aided Routing*, Centrum Voor Wiskunde en Informatica, Amsterdam, 1988.
56. Rochat, Y. and Taillard, E. D., “Probabilistic Diversification and Intensification in Local Search for Vehicle Routing”, *Journal of Heuristics*, Vol. 1, 1995, pp. 147-167.
57. Taillard, É., Badeau, P., Gendreau, M., Guertain, F., and Potvin, J. Y., “A New Neighbourhood Structure for the Vehicle Routing Problem with Time Windows”, Technical Report CRT-95-66,Centre de Recherche sur les Transports, University of Montreal, 1995.
58. Kilby, P., Prosser, P., and Shaw, P., “Guided Local Search for the Vehicle Routing Problem with Time Windows”, *META-HEURISTICS Advances and Trends in Local Search Paradigms for Optimization*, Kluwer Academic Publishers, Boston, 1999.
59. Zhong, Y. and Cole, M. H., “A Vehicle Routing Problem with Backhauls and Time Windows: A Guided Local Search Solution”, *Transportation Research Part E*, Vol. 41, 2005, pp. 131-144.
60. Dorigo, M. and Gambardella, L. M., “Ant Colonies for the Traveling Salesman Problem”, *BioSystems*, Vol. 43, 1997, pp. 73-81.
61. Gambardella, L. M. and Dorigo, M., “Solving Symmetric and Asymmetric TSPs by Ant Colonies”, In Proceedings of the IEEE International Conference on Evolutionary Computation, Nayoya University, Japan, 1996, pp. 622-627.
62. Gambardella, L. M. and Dorigo, M., “Ant-Q: A Reinforcement Learning Approach to the Traveling Salesman Problem”, In Proceedings of the Eleventh International Conference on Machine Learning, Morgan Kaufmann, 1995, pp. 252-260.
63. Doerner, K., Hartl, R. F., and Reimann, M., “Ant Colony Optimization Applied to the Pickup and Delivery Problem”, Working Paper, 2000.