第二章 時間窗車輛路線問題文獻回顧

本章介紹歷年來，國際間著名學者研究 VRPTW 問題之相關文獻整理，包括 VRPTW 的問題定義；相關時間窗文獻蒐集；構成 VRPTW 之充要條件-時間可行性之檢驗；構建式啟發式解法；路線改善法；綜合型構件/改善法；巨集啟發式解法等。

2.1 時間窗車輛路線問題定義

以下是根據 Solomon(1983)[29]之硬時窗車輛路線問題之數學模式，第一目標為總車輛數最少，第二目標為給定總車輛數求總路線距離最短。

Object1 min K (1) ⇒ 求總車輛數 K最少

Object2 min $\sum_i \sum_j \sum_k c_{ijk} x_{ijk}$ (2) ⇒ 給定K 求總路線距離最小

subject to :

$\sum_i q_i y_{ik} \le V_k \quad k = 1,..., K$ (3) ⇒ 容量限制

$$\sum_k y_{ik} = \begin{cases} K & i = 0 \\ 1 & i = 1,..., n \end{cases}$$ (4) ⇒ 每點恰經過一次

$\sum_i x_{ijk} = y_{jk} \quad j = 0,..., n \quad k = 1,..., K$ (5) ⇒ 流量守恆

$\sum_j x_{ijk} = y_{ik} \quad i = 0,..., n \quad k = 1,..., K$ (6)

$b_j \ge b_i + s_i + t_{ij} - (1 - x_{ijk})T \quad i, j = 1,...n \quad k = 1,..., K$ (7) ⇒ 時間順序

$t^k_{0,f} \ge b_i + s_i + t_{i0} - (1 - x_{i0k})T \quad i = 1,..., n \quad k = 1,..., K$ (8)

$b_j \ge t^k_{0,s} + t_{0j} - (1 - x_{0jk})T \quad j = 1,..., n \quad k = 1,..., K$ (9)

$e_i \le b_i \le l_i \quad i = 1,..., n$ (10) ⇒ 時間窗限制

$e_0 \le t^k_{0,p} \le l_0 \quad p = s, f \quad k = 1,..., K$ (11) 路線時限限制

$b_i \ge 0 \quad i = 0,.., n$ (12)

$y_{ik} = (0, 1) \quad i = 1,..., n \quad k = 1,..., K$ (13) ⇒ 二元變數

$x_{ijk} = (0, 1) \quad i, j = 0,...n \quad k = 1,..., K$ (14) ⇒ 二元變數

變數定義：K：代表所有車輛之集合，下標為 k

q_i：顧客 i 之需求量

V_k：車輛 k 之容量

c_{ijk}：代表車輛 k 從顧客 i 到 j 的行駛成本

y_{ik}：0-1 整數變數；當車輛 k 服務顧客 i 時其為 1，否則為 0

x_{ijk}：0-1 整數變數；當車輛 k 從顧客 i 到顧客 j 時其為 1，否則為 0

b_i：代表開始服務顧客 i 時間

s_i：代表顧客 i 所需服務時間

t_{ij}：代表顧客 i 到顧客 j 所需的旅行時間

T：為一大的常數，通常以所有路線中最長的路線之服務時間為代表

e_i：代表顧客 i 時窗之下界

l_i：代表顧客 i 時窗之上界

$t^k_{0,s}$：代表車輛 k 從場站出發之時間

$t^k_{0,p}$：代表車輛 k 抵達場站之時間

e_0：路線時限之下界

l_0：路線時限之上界

(1)、(2)式為 VRPTW 之目標函數；(3)式保證每條路線的需求不會超過車輛容量；(4)式說明每條路線起迄點皆為場站(depot)，且每個顧客點恰被一輛車服務一次；(5)(6)式限制每個顧客點恰被一條路線進入與離開一次；(7)-(9)保證抵達任意兩顧客的時間不會矛盾，且(7)式為破除子迴路限制式；(10)式說明運達顧客點時間不能違反時間窗限制；(11)說明每條路線不能違反路線時限。

假設由顧客 i 運送到顧客 j，則開始服務顧客 j 的時間 $b_j = \max[e_j, b_i + s_i + t_{ij}]$，即當車輛在時間窗下界 e_j 之前抵達顧客點 j 時($b_i + s_i + t_{ij} < e_j$)，須等到時間下界方可開始服務，而產生等待時間 $w_j = e_j - (b_i + s_i + t_{ij})$。

由於 VRPTW 之計算複雜度屬於 NP-hard，當問題規模擴大時無法保證在有限時間內求得最佳解，故大部份學者致力於啟發式解法的發展。構建式啟發法較具代表性的有節省法(Savings Heuristics)、插入法(Insertion Heuristics)等。路線改善法有節線交換法如 k-opt、or-opt，節點交換法如λ-interchange。而新近發展之啟發式解法主要著重在於能夠跳脫區域最佳解，達到更深度的搜尋，以及更進一步擴大解空間的搜尋範圍，以達到廣度搜尋的效果，此類解法最具代表性的為禁制搜尋法(TS, Tabu Search)。而有部份學者致力於精確解解法之研究如 Desrochers et al.(1992)[7] 以變數產生法(column generation)解出 100 點之例題（Solomon(1983)[29]測試例題中的 7 題，其中 2 題為顧客點均勻分布形態，5 題為顧客點群聚形態）；Fisher et al. (1997)[11]亦分別以拉式鬆弛法(Lagragian relaxation)及 K-樹法(K-tree approach)解出 100 點之例題（皆為顧客點群聚形態）。

2.2 時間窗車輛路線問題解法回顧

2.2.1 時間可行性之檢驗

VRPTW 問題中的顧客點具有時間窗的限制，當顧客插入一路線中，必須逐一檢查此顧客以及其後受到影響之顧客是否違反時窗上界，以決定此顧客插入此位置之時間可行性(time feasibility)，Solomon(1983)[29]提出檢查時間可行性之充分必要條件，茲簡要說明如下：

如圖 2.1 所示，假設將顧客 u 插入路線 i 中的第 p 點之前，即路線 i(0, 1, 2, …, p-1, p, …, m)中 p 和 p-1 之間，其中 0=m=depot，1≤p≤m。b_p 為開始服務顧客 p 之時間，b_p^n 為插入顧客後，開始服務顧客 p 之新時間。w_p 為在顧客 p 之等待時間。因此插入顧客 u 之後，顧客 p 開始服務時間之後推時間項(push forward)為：

$PF_p = b_p^n - b_p \geq 0$，

$PF_{r+1} = \max\{0, PF_r - w_{r+1}\}$，　$p \leq r \leq m-1$

時間可行性之充分且必要條件為：

$b_u \le l_u$　且　$b_r + PF_r \le l_r$，　$p \le r \le m$

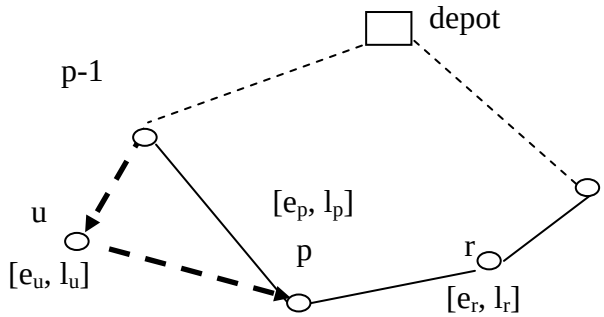


圖 2.1　時間可行性示意圖

對插入點 u 及之後各點 i_r 逐一檢查時間可行性，直到路線迄點。但在檢查過程中，若遇到下列二種情形即可停止：(1)遇到 $PF_r = 0$ 時，$p \le r \le m$，提前確定時間可行。(2)或在顧客 r 違反時間窗上界，提前確定時間不可行，$p \le r \le m$。

2.2.2 構建式啟發式解法

1. 節省法

屬於循序構建啟發式解法。Solomon(1983)[29]延伸 Clarke and Wright(1964)[2]原運用在 VRP 之節省法，將其應用在 VRPTW 問題上。此法根據成本節省值大小依序將兩顧客點之節線加入路線中，直到所有顧客都分配到路線中為止。開始構建路線時，將第一條節線加入路線後，選擇時間上界較早的顧客點為第一點，以決定路線方向。節省值公式為 $s_{ij} = c_{i0} + c_{0j} - Qc_{ij}$ 其中 Q 為路線形狀參數（設為 1 或 2），分為 Gaskell(1967)及 Yellow(1970)所提出。每一節線加入的步驟皆須檢查是否違反容量及時間可行性。此外，由於時間窗使得路線之合併需考慮順序的問題，如圖 2.2 所示：

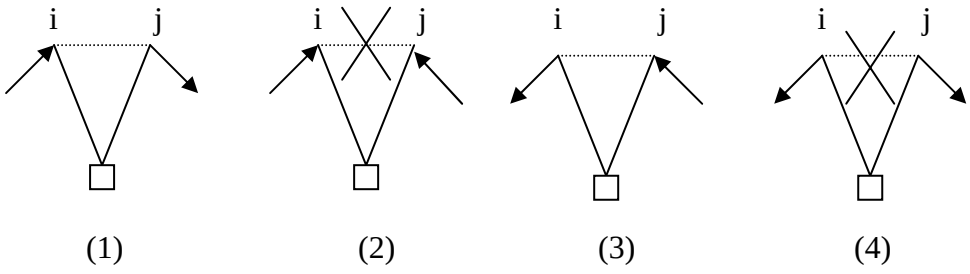


圖 2.2　節省法示意圖

i、j 分別位於兩條路線的端點，由於路線有方向，若欲將(i, j)加入路線中，必須兩條路線的方向不衝突始可加以合併，如上圖之(1)、(3)兩種情形可行，(2)、(4)兩種情形因為路線方向衝突，故不可行。除此之外兩條子路線必須符合下列條件才可以合併：

(1)i 和 j 在不同的子路線中。

j和j皆非路線的內部點(interior point)，即須為路線之端點。

(2)兩子路線合併後的需求量不可超出車輛容量。

(3)兩子路線之方向需可行（如上圖所示），且合併後各顧客點到達時間需符合時間窗限制。

(4)加入(i,j)後若造成在 j 點之等待時間過長則不採用(i,j)，即 $wj \le W$，W 為事先設定之最大等待時間。

2. 時間導向最近鄰點法

屬於循序構建啟發式解法。由 Solomon(1983)[29]提出，其概念如下：由場站(depot)開始，找最近鄰點加入路線末端，以擴大路線。即假設 i 點為路線末端，j 點為未排程點，找 c_{ij} 最小之 j 點(最近鄰點)，將 j 點接到 i 點之後。但此點加入後需滿足車輛容量限制與時間窗限制，若在未排程之所有點中皆無法找到可行點，則構建新路線，直到所有顧客點都排入路線為止。最近鄰點的指標 c_{ij} 考慮空間與時間因素，包括兩顧客點間的距離 d_{ij}、在 i 點完成服務後到 j 點開始服務前所經過的時間 T_{ij}（即旅行時間加上停等時間）、以及抵達 j 點後離時間窗上界之剩餘時間 u_{ij}。以數學式表示如下：

$$c_{ij} = \delta_1 d_{ij} + \delta_2 T_{ij} + \delta_3 u_{ij} \qquad \delta_1 + \delta_2 + \delta_3 = 1, \quad \delta_1 \ge 0, \delta_2 \ge 0, \delta_3 \ge 0$$

其中 $T_{ij} = b_j - (b_i + s_i)$

$$u_{ij} = l_j - (b_i + s_i + t_{ij})$$

由於 Solomon(1983)[29]提出時間導向最近鄰點法主要是以循序方式進行路線構建，近年來有若干的研究(王生德(2003)[38])將循序構建的鄰近點法，稍做變化改由平行方式進行路線構建。由於傳統鄰點法僅以距離為考慮因素，而 Solomon(1983)[29]提出的時間導向最近鄰點法又考慮過多權重因子，增加許多無謂的計算。為了因應 VRPTW 問題的特性，改良式鄰點法可以將時間窗類型問題所考慮到的因子，如：距離、時間窗上界以及停等時間等，直接納入考量，以減少許多無謂的計算。循序法的觀念在於：一條路線構建完畢之後再構建另一條新的路線，直到所有需求點皆已納入所有路線為止；而平行法的觀念在於：可以同時構建多條路線，直到所有需求點皆已納入所有路線為止。

3 插入法

屬於循序構建式啟發式解法。Solomon(1983)[29]將插入法應用於 VRPTW 問題，其使用兩階段構建路線，第一階段對於尚未排程之顧客點，選取各點的對應路段（插入位置）。第二階段選擇最佳之顧客點以插入其對應路段中。假設($i_0, i_1, i_2, \ldots, i_m$)為目前正在構建中的路線，u 為未排程的顧客點，第一階段以 $c_1(i, u, j)$ 來決定其在目前路線中之最佳插入位置，選擇最佳插入位置之準則如下式所示：

$$c_1(i(u), u, j(u)) = \min[c_1(i_{p-1}, u, i_p)] \qquad P = 1, \ldots, m$$

$$c_1(i, u, j) = \alpha_1 c_{11}(i, u, j) + \alpha_2 c_{12}(i, u, j) \quad \alpha_1 + \alpha_2 = 1, \quad \alpha_1 \ge 0, \alpha_2 \ge 0$$

其中 $c_{11}(i, u, j) = d_{iu} + d_{uj} - d_{ij}$

$c_{12}(i, u, j) = b_{ju} - b_j$　　b_{ju} 為插入 u 後，開始服務顧客 j 的時間

因為將 u 插入 i, j 中將影響 j 及 j 之後各點的抵達時間，因此需檢驗 j 之時間可行性。

第二階段以 $c_2(i, u, j)$選擇最佳之 u 點以插入路線中，選擇最佳點之準則如下式所示：

$c_2(i(u^*), u^*, j(u^*)) = \max[c_2(i(u), u, j(u))]$

u 為尚未排程顧客點中之可行點

當無法找到任何可行點時，則構建新路線，直到所有顧客點皆排程為止。

$$c_2 (i, u, j) = \lambda d_{0u} - c_1 (i, u, j) \quad \lambda \geq 0$$

第一階段對於一個尚未排程的顧客點來說，最佳的可行插入位置為使插入後所增加的距離和時間的權重組合最小的位置。第二階段則將 u 插入路線中的插入節省值最大化，例如當$\alpha_1 = \lambda = 1$ 且$\alpha_2 = 0$ 時，$c_2(i, j, u)$即為 u 在距離上的插入節省值。各參數設定建議 $(\alpha_1, \alpha_2) = (1, 0)$或$(0, 1)$，$\lambda = 1$ 或 2。

4. 時間導向掃描法

屬於循序構建式啟發式解法。Solomon(1983)[29]將掃描法應用於 VRPTW 問題，第一階段，將所有顧客分成數個區間，每個區間之顧客需求量不超過車輛容量。第二階段，應用插入法（如 3.4.3 節所述）分別對每個區間的顧客服務，並檢查時間可行性，若有顧客無法排程（無法滿足時間窗限制或路線時限限制），則先排除此顧客點。每個區間皆排程後，若所有顧客點皆已服務，則完成構建路線；若有某些顧客尚未被服務，則由原區間之一半開始，沿原掃描方向進行掃描，將剩餘尚未服務之顧客點重複掃描-排程的步驟，直到所有顧客皆被服務。

5.　修正插入法

Potvin 和 Rousseau(1993)[23]提出，以 Solomon(1983)[29]的循序節省插入法為基礎，發展平行路線構建的插入法，並修正第二階段選擇最佳點的準則。

開始必須先決定路線數，先以循序節省插入法來預測路線數 N，之後選擇 N 個距離場站最遠點為各路線之種子點。

第一階段，對於每個未排程的顧客點，首先計算其在每條路線中的最佳插入位置，選擇最佳插入位置之準則如下式：

$$c_{1r}^*(i_r(u), u, j_r(u)) = \min_{p=1,\ldots,m}[c_{1r}(i_{rp-1}, u, i_{rp})] \quad r = 1,\ldots,N$$

$$c_{1r}(i, u, j) = \alpha_1 c_{11r}(i_r, u, j_r) + \alpha_2 c_{12r}(i_r, u, j_r) \quad \alpha_1 + \alpha_2 = 1, \quad \alpha_1 \geq 0, \alpha_2 \geq 0$$

此處 $c_{11r}(i_r, u, j_r) = d_{ir,u} + d_{u,jr} - d_{ir,jr}$ ； $c_{12r}(i_r, u, j_r) = b_{jr,u} - b_{jr}$

第二階段，選擇最佳之 u*，選擇最佳點之準則如下所示：

$c_2(u^*) = \max_u[c_2(u)]$

$c_2(u) = \Sigma_{r\neq r'}[c_{1r}*(i_r(u), u, j_r(u)) - c_{1r'}*(i_{r'}(u), u, j_{r'}(u))]$

此處 $c_{1r'}*(i_{r'}(u), u, j_{r'}(u)) = \min_{r=1,\ldots,nr}[c_{1r}*(i_r(u), u, j_r(u))]$

將 u*插入 $i_{r'}(u^*)$與 $j_{r'}(u^*)$之間，重複此步驟直到所有顧客都排程為止。構建路線完成後可以將原路線數減少，再次執行以上步驟，以檢查是否能以較少的車輛數來服務所有顧客點。若還有些顧客未能排程，則將原路線數增加，再執行以上步驟，以獲得可行解。

此方法的 $c_{1r}(i_r(u), u, j_r(u))$與 Solomon(1983)[29]的定義相同，為路線排程與時間延遲的權重組合。而 $c_2(u)$則是一般化懊悔指標(generalized regret measure)，懊悔指標意指最佳插入位置與其他插入位置之成本差距總和。若不將最佳的顧客立即插入其最佳路線，在下一步驟可能造成損失。即對於 u 來說，其最佳的插入路線位置是$(i_{r'}(u), j_{r'}(u))$，若不立即將 u 插入此位置，在下個迭代後可能因路線改變，而必須選擇其它位置，因而造成的損失。此想法為對於某未排程顧客點而言，在所有路線中只有一條路線插入後的成本較小，插入其它路線的成本皆很大時，先處理此顧客點將會比較有利，因為其選擇性小；相對的，若對此顧客點而言，可以插入的路線很多且成本皆差異不大時，則此顧客點可以稍後考慮，因其選擇性多。

Potvin 和 Rousseau(1993)[23]測試後發現在均勻分佈顧客點（r 型）及半群聚顧客點（rc 型）的例題中，修正插入法比插入法表現要好；但在分群顧客點（c 型）的例題中，則插入法表現較修正插入法好。

2.2.3 路線改善法

1. VRPTW 之路線改善法

VRPTW 之路線改善法是以傳統 VRP 之路線改善法為基礎，傳統 VRP 之交換改善法進行交換時需符合車輛容量限制方為可行解，VRPTW 問題因為顧客點之時間窗限制與路線最大時限限制，進行節點或節線交換後的解需符合時間窗限制及路線時限限制方為可行解。Campbell and Savelsbergh(2004)[3]對符合時窗限制以及路線時限限制有詳盡的說明。

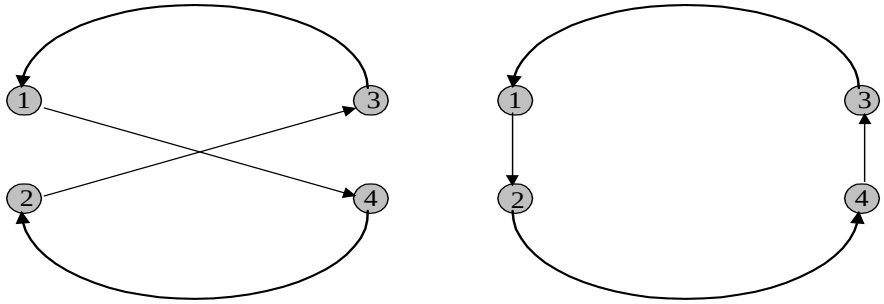
對於目標函數(C(x))的定義，VRP 的目標函數為總路線距離；而 VRPTW 則考慮車輛數、總路線距離與總排程時間，Chiang 和 Russell(1997)[4]對定義其目標函數 $P_1m+P_2(\beta_1D + \beta_2T)$，其中 m 為使用車輛數，D 為總路線距離，T 為總排程時間，$P_1 \gg P_2$，以強調使用車輛數優先考慮，$(\beta_1, \beta_2) = (1, 0)$。

由以上可知 VRPTW 之路線改善法，可以採用傳統 VRP 路線改善法，並修正其目標函數，以及加入時間可行性檢驗模組。以下介紹傳統 VRP 路線改善法。

2. 傳統 VRP 路線改善法[33]

(1) K-Opt 節線交換法

K-Opt 節線交換法係由 Lin(1965)[18]所提出，其中 K 表示每次交換的節線數，一般常被使用的 K 為 2 或 3。此交換法原先設計來用於 TSP 問題上，而對於 VRP 問題而言，可用於各路線內的節線交換。茲以圖 2.3 為例，說明 2-Opt 的解題觀念：



(a) (b)

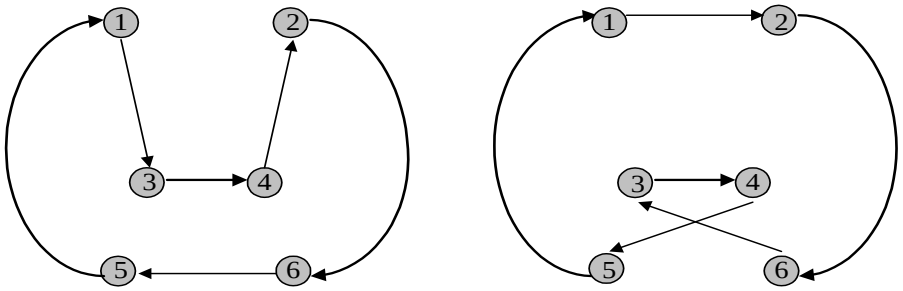
圖 2.3 2-Opt 節線交換法的解題觀念

令圖 2.3(a)為一路線解，很明顯地，若換掉其中(1，4)及(2，3)兩條節線，然後連接(1，2)及(4，3)兩條節線而成為圖 2.3(b)的路線，將有可能改善其解。2-Opt 節線交換法的解題架構即根據此觀念：對任一路線，可依圖 2.3 的方式交換路線上任兩條不相鄰的節線，然後檢查交換後的解是否優於交換前的解。若是，則更新解；否則維持原解，繼續交換其它兩條節線直到所有可能交換的節線對都檢查完畢為止。

從理論上來看，若能檢查所有可能的交換情形，則最後的解應該就是最佳解。但是可能的交換情形實在太多了，在實際求解時通常只檢查部份的交換情形（亦稱為 2-Exchange 法），並輔以其它方式來加快 2-Opt 的計算速度，因此仍算是一種求近似解的啟發式解法。

(2) or-opt

由於 2-Opt 或 3-Opt 的交換方式都會造成某些節線連接方向需要反轉的情形，徒增計算上的負擔，因此在 Or(1976)[21]提出了所謂的 Or-Opt 節線交換法。該方法可算是 3-Opt 的一種變型，由於不需要對任何一段之節線作反轉，因此提高了程式執行的效率。此交換法原先之設計與 K-Opt 類似，適用於 TSP 問題上，對於 VRP 問題而言，可用於各路線內的節線交換。其解題觀念如圖 2.4 所示：



(a) (b)

圖 2.4 Or-Opt 節線交換法的解題觀念

令圖 4(a)為一路線解，若換掉其中(1, 3)、(4, 2)及(6, 5)三條節線，再連接(1, 2)、(6, 3)及(4, 5)三條節線而成為圖 2.4(b)的路線，將可能改善其解且不必反轉節線。其中，點 1 到點 2 之間的點可以為一至三點。Or-Opt 的解題架構即根據此觀念：對任一路線，可依圖 2.4 的方式交換路線上任三條不相鄰的節線，然後檢查交換後的解是否優於交換前的解。若是，則更新解；否則維持原解，繼續交換其它三條節線直到所有可能交換的節線對都檢查完畢為止。

(3)保持方向性之路線內 3-exch(Orientation-Preserving 3-interchange)

Pesant & Gendreau (1999)[15]提出，其交換概念比 or-opt 交換概念更進一步，以下圖 2.5 所示。令圖中為 VRPTW 解之其中一條路線解，將路線內中任意三條

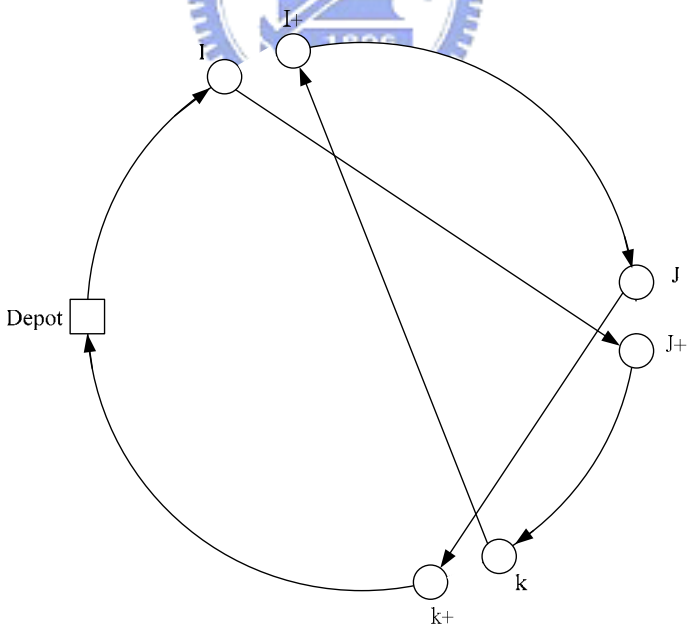


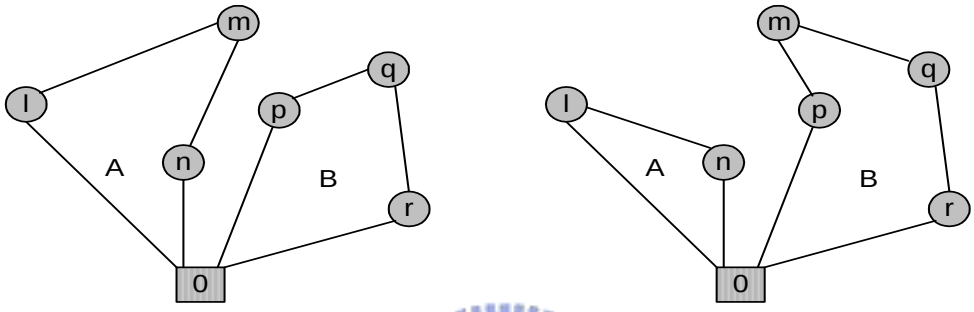
圖 2.5 保持方向性之路線內 3-exch 節線交換法的解題觀念

相鄰節點之節線打斷，即將 I 點與 I^+點、J 點與 J^+點以及 k 點與 k^+點間之節線打斷，其中 Depot 與 I 點、I^+點與 J 點、J^+點與 k 點、k^+點與 Depot 點可以重疊，再將 I 與 J+、K 與 I+以及 J 與 K+間之路段互相連結即可。該特色在於維持顧客服務的順序下，進行路線節線順序的交換。在進行節線交換時，無需將節線上之顧

客順序反轉，只需將節線順序對調即可。此方法的優點在於交換時避免路徑反轉，在搜尋時，可以搜尋所有符合時間窗之可行解。

(4) λ-interchange

Osman(1993)[22]將(1-0)節點交換法稱為一個轉移過程(shift process)，而將(1-1)及(1-2)節點交換法稱為λ-路線間節點交換法。茲以(1-0)節點交換法來說明λ-interchange 之概念。令圖 2.6(a)為 VRP 解之其中二條路線，若將路線 A 的節點 m 插入到路線 B 的節點 p 與 q 之間，且路線 B 仍滿足車輛容量限制，而成為圖 5(b)的二條新路線。



(a) (b)

圖 2.6 1-0 節點交換法的解題觀念

2.2.4 綜合型構建/改善法

Russell(1995)[27]提出綜合型構建/改善法，不同於傳統改善法先構建完路線後再進行路線改善，綜合型構建/改善法的精神為構建一部份路線即進行路線改善，再將路線擴大後，再進行路線改善，直到路線完成後再對完整的路線進行一次路線改善。以下說明綜合型構建/改善法之步驟：

首先要決定起始路線數 N，之後選擇 N 個種子點，暫時將各種子點指派給各路線，當有顧客點插入此路線後即將種子點由路線中刪除。未排程顧客點依一定的順序插入路線中，根據顧客插入順序，決定此顧客點之最佳插入位置：

$c_{1r}^*(i_r(u), u, j_r(u)) = \min_{r=1,\ldots,N}[\alpha_1 c_{11r}(i_r, u, j_r) + \alpha_2 c_{12r}(i_r, u, j_r)]$

此處 $c_{11r}(i_r, u, j_r) = d_{ir,u} + d_{u,jr} - d_{ir,jr}$ ； $c_{12r}(i_r, u, j_r) = b_{jr,u} - b_{jr}$

即尋找 u 在各路線中距離與時間權重組合之最佳插入位置。插入之位置需符合時間窗限制。當插入之顧客數達到總顧客數之 10%時，對目前之路線進行交換改善法。重複以上步驟直到所有顧客點皆檢查過一遍，若還有顧客點無法找到插入位置，則採用 Solomon(1983)[29]插入法（如 2.4.3 節所述）將剩餘顧客點排程。最後可試著將起始路線減少一條再重複上述步驟對所有顧客點進行排程。

2.2.5 巨集啟發式解法

1. 鄰域搜尋之概念

上述各種路線改善法雖有不同的設計理念，仍均屬於傳統的鄰域搜尋之解題架構。傳統的鄰域搜尋(neighborhood search)亦稱為「局部搜尋(local search)」，即是在現解 S 的「鄰域(neighborhood)」中搜尋下一個可行解 S'。所謂鄰域，$N_\lambda(S)$，乃是現解 S 之所有「鄰解(neighbors)」所成之集合；而鄰解則是指可經由變換現解 S 之λ個元素所形成之新的可行解 S'。以 2-Opt 節線交換法為例，鄰解 S'即是由更換現解 S 中的兩段節線而產生。從現解 S 轉變成鄰解 S'的過程，稱之為「移動(move)」；移動會引起目標值的改變。可行解與其鄰域聯集形成的解空間，稱之為「搜尋空間(search space)」；不同的鄰域搜尋法會產生不同的搜尋空間。此外，進行鄰域搜尋時，必須依據「選擇法則(selection rule)」來決定現解該移動到那個鄰解。傳統的選擇法則有：(1) 最佳改善(best improvement)，即所有優於現解之鄰解中，最佳的一個鄰解；與(2) 首先改善(first improvement)，即搜尋過程中找到的第一個優於現解之鄰解。

由於鄰域搜尋法在移動時採用嚴格的「接受法則(acceptance rule)」，亦即只會接受比現解好的鄰解，若沒有較佳的鄰解則停止搜尋，因此最後會陷入「局部最佳解(local optimum)」而無法自拔。因而近年來啟發式解法的趨勢為結合傳統鄰域搜尋方法與跳離局部最佳解之機制，發展巨集式啟發式解法(Meta-Heuristics)，以下所要介紹的兩類啟發式解法皆屬於此範疇。

2. 禁制搜尋法(Tabu Search, TS)

新近發展之啟發式解法最著名的首推禁制搜尋法，此方法最早由 Glover 及 Hansen(1986)[13]所提出，其觀念是想構建一個智慧型的問題求解程序：在目前解的鄰域(neighborhoods)進行搜尋，並應用人工智慧的記憶機制，將已經搜尋過的解記錄在「禁制名單(tabu list)」，以避免重複性或毫無目標的搜尋；等到整個鄰域都搜尋完畢後，再選擇一個最佳的方向進行移動(move)，以逐漸逼近最佳解。由此可知，TS 法的關鍵即在於記憶機制的設計，目前發展至今已形成相當複雜的執行架構，其所應用的高階策略主要包含了以下四個概念。

(一).記憶結構(Memory Structures)管理：

記憶結構乃是 TS 法之特色與核心，可分成短期記憶(Short Term/Recency-Based Memory)及長期記憶(Long Term/ Frequency-Based Memory)兩種結構。短期記憶已禁制列為基礎，將最近搜尋過的解或移動之屬性(Attributes)紀錄在禁制列，以避免後續搜尋的解重複先前的搜尋途徑。然而，經過一段禁制期間(Tabu Tenure)之後，禁制的屬性即可恢復自由。此

外，短期記憶亦可利用可望水準(Aspiration Levels)的機制來打破禁制列的限制；亦即當搜尋的新解優於目前最佳解時，雖然其屬性在禁制列之中，仍允許移動至該解。至於長期記憶結構則以記錄屬性出現的次數為主，再配合深度或廣度搜尋策略，以擴大TS法的搜尋範圍。

(二)深度搜尋與廣度搜尋(Intensification and Diversification Search)：

深度搜尋策略係在搜尋過程中將較佳的數個解記錄在精英列(Elite List)內，當短期記憶搜尋無法改善時，再從精英列中選擇一個解作為下階段搜尋的起點，重新開始。精英列不一定要紀錄一個完整的解，也可以只紀錄經常出現的部份解(Parts of Solution)。廣度搜尋策略則需要配合長期記憶結構紀錄搜尋過程中解或屬性出現的次數，當短期記憶搜尋無法改善時，選擇次數較少之屬性方向重新進行短期記憶搜尋。計算出現次數時，需乘以一懲罰值(Penalty)，以控制其搜尋方向。

(三)策略交替運用(Strategic Oscillation)：

是一個調和深度搜尋與廣度搜尋的機制，藉由臨界水準(Critical Level)來控制深度搜尋與廣度搜尋的切換時機。

(四)搜尋路徑連結(Path Relinking)：

先設定一個目標解(Guiding Solution)，然後藉由深度搜尋、廣度搜尋與渴望水準控制搜尋路徑朝向目標解前進。

3. 門檻型演算法(Threshold Algorithms)

此類方法Fisher(1995)[12]稱之為包容性搜尋法，其觀念乃是在鄰域搜尋陷入局部最佳解時，採取較鬆的接受法則，亦即接受劣於現解之鄰解，以便脫離局部最佳解的束縛而繼續搜尋下去。模擬鍛鍊法(Simulated annealing, SA)、門檻接受法(Threshold Accepting, TA)、大洪水法(Great Deluge Algorithm, GDA)與記錄更新法(Record-to-Record Travel, RRT)皆屬於這類方法。這類方法之基本觀念乃是在鄰域搜尋陷入局部最佳解時，採用較鬆的接受法則(通常為一門檻值)接受列於現解之鄰解，以便脫離局部最佳解的束縛而繼續搜尋下去。SA、TA、GDA與RRT等方法的執行架構與傳統鄰域搜尋法架構類似，差異之處僅在於使用的接受法則不同。傳統的鄰域搜尋法僅接受較佳的鄰解，門檻型演算法可接受暫劣解之鄰解。

模擬鍛鍊法的基本觀念最早是由Metropolis et al.(1953)所提出，然後由Kirkpatrick et al.(1983)[17]加以應用到組合最佳化問題之求解上，因而產生了目前所謂的模擬鍛鍊法。SA採用機率性的接受暫劣解法則，利用一個隨機產生的數值與門檻值做比較，此門檻值是鄰解與現有解之目標值差額與溫度的函數，亦即鄰解與現解的差值越大，則此鄰解被接受的機率越小；此處所問「溫度」對SA而言是一個抽象的觀念，僅作為控制門檻值高低的參數；降溫則是為了使SA能夠逐漸收斂。Li. H & Lim. A (2003)[20]使用模擬鍛鍊法對VRPTW問題進行求

解，在車輛數方面其結果有不錯的表現。而Li. H & Lim. A(2001)[19]使用模擬鍛鍊法對時間窗收送貨車輛路線問題進行求解，其結果亦有不錯的表現。

TA的觀念源自SA，但是其採用確定性的接受法則，只要鄰解與現解的差在門檻值以下就接受，其執行方式更為簡單，圖2.7說明SA與TA接受法則之差異。Dueck and Scheuer(1990)[9]發表TA法時，以一個442節點的TSP例題來驗證其可行性，結果顯示TA可以在很短的時間內找到相當不錯的解。

Dueck(1993)[10]又根據TA的觀念衍生出兩個新的方法：GDA與RRT。

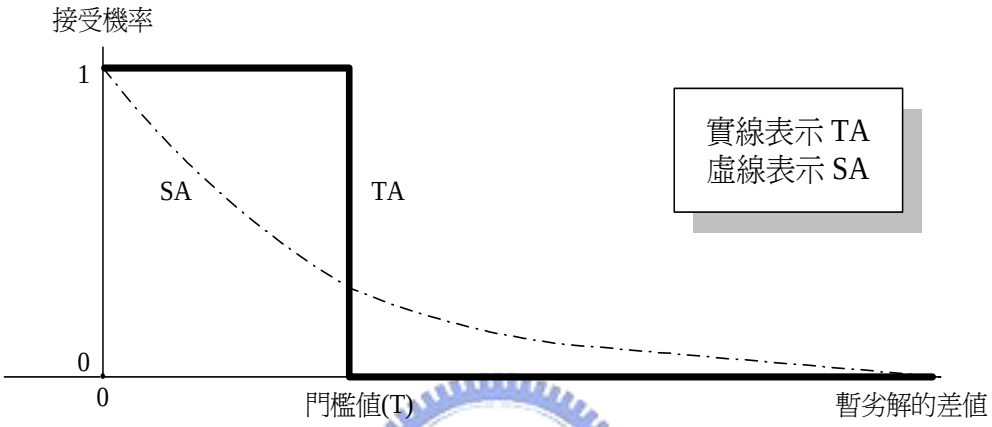
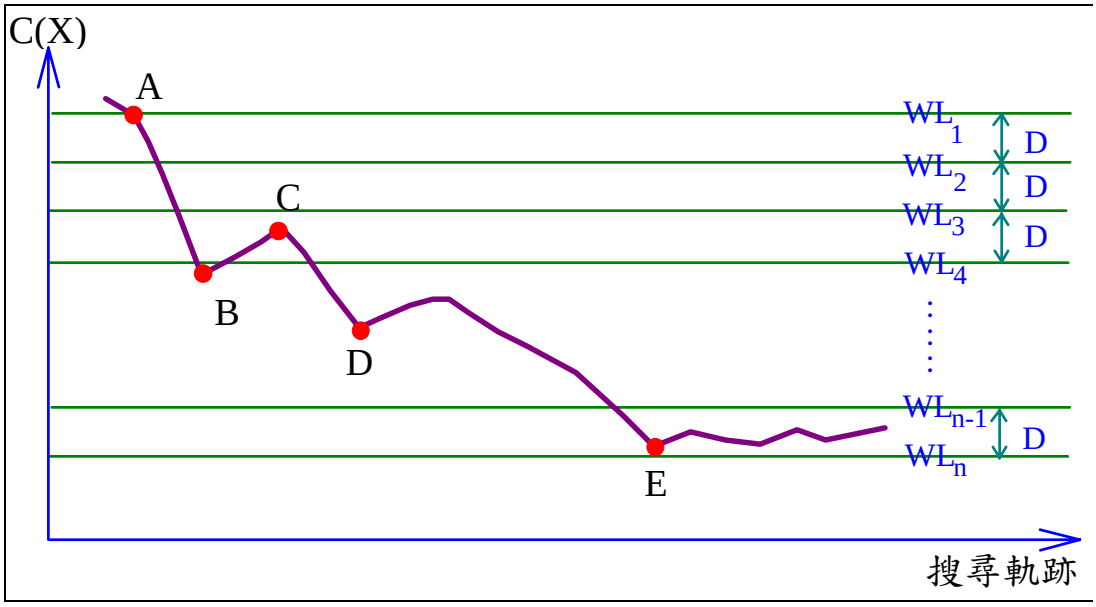


圖2.7　TA與SA接受暫劣解機率之比較

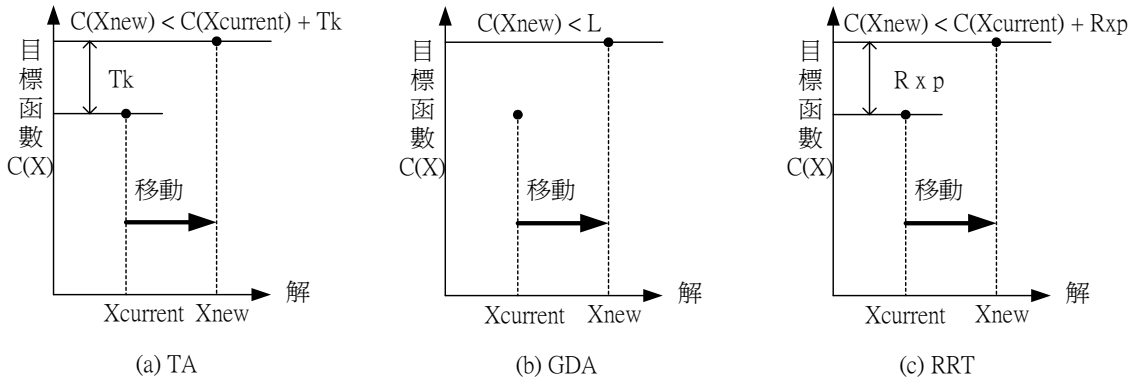
模擬鍛鍊法(SA)、門檻接受法(TA)、大洪水法(GDA)與紀錄更新法(RRT)等方法的執行架構與傳統鄰域搜尋架構相似，其差異處在於其使用的接受法則不同。傳統的鄰域搜尋法僅接受較佳的鄰解，包容性搜尋法則可接受暫劣之鄰解。現就以大洪水法來說明包容性搜尋之概念，大洪水法原本是設計用以求解最大化問題，因此以最大化問題做說明較容易了解：想像在一個有高低起伏的地面上，不停地下著大雨，假設下雨的速度吾人可以控制，地上的水面將隨著大雨往上升。我們希望在露出水面的陸地上找到高地，一旦找到了就讓水面上升一固定高度。這時原來在水面上的部份陸地，將被雨水所淹沒，而吾人在移動時，僅考慮露出水面的陸地。如此不停地找尋高地，直到找到地面上最高的山峰或無法再找到更高的高地為止。

對一個最小化的問題而言：可將原來的高地想像成窪地，而水面則如水庫洩洪一般由高處往低處下降。圖2.8為大洪水法之解題概念示意圖。假設我們現在的位置為圖中之A點，並將水位設於WL1。接著開始找尋「窪地」，設找到圖中B點，便將水位下降D的高度至WL2。此時水面WL2下的所有範圍皆為吾人找尋並可接受「窪地」的範圍，包括不是很低的C點。如此不斷地搜尋「窪地」及下降水面(至WLn)，直至找到水面下最低之山谷(E點)或無法再找到更低點為止。



資料來源：[37]　　圖2.8　大洪水法解題概念示意圖

而TA、GDA與RRT的觀念均來自SA，在接受法則上面也有所不同，SA的接受法則為不確定性接受法則，而TA、GDA以及RRT均採確定性接受法則。以圖2.9示意圖說明TA、GDA與RRT等方法之接受法則異同。簡單而言，TA法事先產生一組固定的門檻值數列(通常為遞減)，依次使用數列中的門檻值，其接受法則為C(Xnew) < C(Xcurrent) + Tk；GDA法設定一個起始水位，只要有改善就降低水位(固定的下降速度)，其接受法則為C(Xnew) < L；至於RRT法則是將目前的暫優解設為紀錄值，取紀錄值之固定百分比率作為門檻值，其接受法則為C(Xnew) < C(Xcurrent) + Rxp。



資料來源：[37]　　圖2.9　TA、GDA與RRT接受法則示意圖

表2.1以最小化問題之求解說明SA、TA、GDA與RRT等方法重要步驟之異同。其中「控制參數」指用以控制演算法執行與停止之參數；「接受法則」為判斷是否從現解S移動至某鄰解S'之準則，傳統之路線改善法採取嚴格的「接受法則」，即當交換後的可行解之目標值低於原有解時(C(S') < C(S))，才接受此解。包容性解法則可接受暫劣之鄰解；「收斂法則」是為確定搜尋過程會收斂，在現解移動後對其控制參數進行調整之方式；「停止法則」則規範演算法停止搜尋之

標準。其中，C(S)為現解S之目標值，C(S')為鄰解S'的目標值。有關TA、GDA與RRT的理論與執行細節，請參見文獻[33, 34, 35, 36]。

表 2.1　SA、TA、GDA 與 RRT 等方法之比較

方　　法	SA	TA	GDA	RRT
控制參數	◦溫度(T) ◦機率值($0 \le r \le 1$) ◦次數(K)	◦門檻(T) ◦次數(K)	◦水位(L) ◦速度(D)	◦誤差率($p < 1$) ◦記錄值(R) ◦次數(K)
接受法則	機率性接受： $r < \exp\left[\frac{C(S)-C(S')}{T}\right]$	確定性接受： $C(S') < C(S) + T$	確定性接受： $C(S') < L$	確定性接受： $C(S') < C(S) + R*p$
收斂法則	T遞減	T遞減	$L = L - D$	更新R值
停止法則	完成K次迴圈	完成K次迴圈	所有$C(S') \ge L$	完成K次迴圈

資料來源：[37]

4. 螞蟻演算法(Ant Colony Optimization, ACO)

螞蟻演算法的基本概念是利用螞蟻外出覓食時，會在行經巢穴與食物間的路徑上，留下一種稱為費洛蒙(pheromone)的荷爾蒙，因此螞蟻行經一路徑的機會與該路徑遺留的費洛蒙素濃度成正比，即當有更多的螞蟻走過該路徑時，遺留的費洛蒙含量就越多，而當費洛蒙素濃度高時，便會吸引更多的螞蟻行走該路徑。假設螞蟻面臨選擇路徑時，決定行走某依路徑的機率將與遺留在該路徑的費洛蒙含量關；另外，路徑越短時，螞蟻通過該路段的時間就越短，造成最短路線上會遺留較高的費洛蒙含量，吸引較多的螞蟻，最後螞蟻將沿最短路徑，求得最佳解。

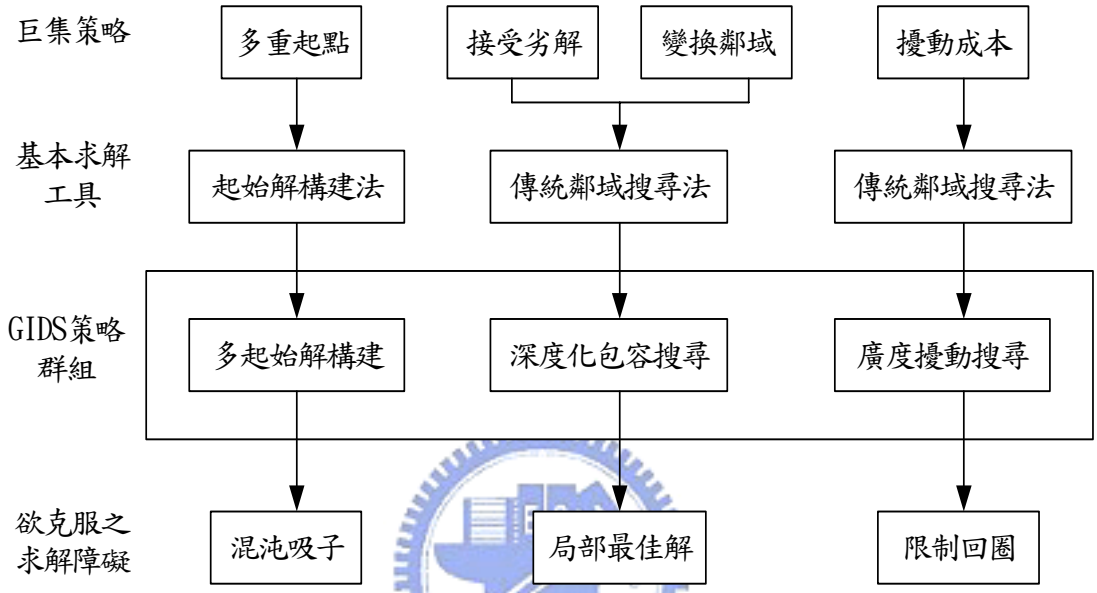
螞蟻演算法及在模仿螞蟻覓食的行為，並且利用螞蟻群體合作的原理來解決問題已獲得最佳決策之搜尋工作。此演算法又稱為蟻群演算法，其最初是由Macro Dorigo於1992年所提出，當時稱該演算法為螞蟻系統(Ant System, AS)，直到Dorigo et al.(1999)[8]將此演算法的精神延伸到求解非連續式最佳化問題上，並命名為ACO，為求解最佳化問題的巨集啟發式方法。在求解的過程是利用多點搜尋方式跳脫區域解。近年來在國外相關文獻中， Gambardella, Taillard and Agazzi(1999) [14]以蟻群系統求解時間窗車輛路線問題，結果有不錯的表現。國內亦有丁慶榮、陳家和(2005)[41]等人利用蟻群系統求解時間窗車輛路線問題。

5. GIDS

卓裕仁(2001)[41]結合多種巨集啟發式方法的特點與優點，將接受劣解、變換鄰域、擾動成本與多重起點等巨集策略融合在深度搜尋與廣度搜尋的概念中，發展出一套「包容性深廣度搜尋(Generic Intensification and Diversification Search, GIDS)」的巨集啟發式方法。

GIDS法共包含：(1)多起始解構建(Multiple Initialization Constructor, MIC)、(2)深度化包容搜尋(Generic Search for Intensification, GSI)與(3)廣度化擾動搜尋

(Perturbation Search for Diversification, PSD)三個策略群組。整套GIDS法係以傳統鄰域搜尋為實際執行求解之工具，以深度搜尋之GSI群組為核心，再搭配廣度搜尋之PSD與MIC群組。此外，並設計了此種模組來執行GIDS之策略模組：在MIC群組構建有加權起始(Weighted Initialization, WI)模組與鄰域搜尋(Neighborhood Search, NS)模組；在GSI群組設計有G1與G2兩種包容搜尋(Generic Search)模組；在PSD群組中則構建有成本擾動(Cost Perturbation,) CP模組。GIDS之解題概念如圖2.10所示。



資料來源：[40]　　圖2.10　GIDS之解題概念架構

6. 新近發表之 VRPTW 啟發式解法

表2.2為近年來文獻上，比較測試例題所發表求解VRPTW的啟發式方法，Potvin et al.(1996)[25]以禁制搜尋法求解VRPTW問題，使用的鄰域搜尋方法為經過修正之2-opt以及Or-opt，所得結果在R1類例題較以往文獻為佳，總車輛數為426輛；Chiang and Russell(1996)[4]以模擬鍛鍊法求解VRPTW問題，分別構建比較λ-interchange及k-node interchange兩種鄰域搜尋方法，在R1、R2、RC1及RC2四類問題較以往文獻佳；Chiang and Russell(1997)[5]以禁制搜尋法求解VRPTW問題，鄰域搜尋方法為λ-interchange，在R1、R2及RC2例題較以往文獻佳，其中有7題突破文獻已知最佳解，有16題與已知最佳解相同，總車輛數為411為目前文獻中最低之總車輛數；Taillard et al.(1997)[32]以禁制搜尋法求解VRPTW問題，鄰域搜尋使用CROSS-interchange、Or-opt以及GENIUS，在R1、R2及RC1較以往文獻佳，其中有17題突破已知最佳解，有20題與已知最佳解相同，總車輛數為416輛；林修竹[40]於1999年發表以TA和GDA，結合傳統交換法，設計多套解題模組以求解VRPTW之問題，其中有1題突破已知最佳解，有8題已知最佳解相同；Gambardella, Taillard and Agazzi(1999)[14]以螞蟻群聚法求解VRPTW問題，利用費洛蒙的更新，分別對車輛數以及路線成本進行求解，有13題突破已知文獻中的

最佳解； Li and Lim(2003)[20]以鄰點法產生起始解，並用Tabu-embedded禁制模擬鍛鍊法，搭配K-restart進行求解，有7題突破已知文獻的最佳解，有19題與已知文獻相同。

表2.2　近年VRPTW啟發式解法比較

文獻作者/年代	巨集搜尋策略	傳統鄰域搜尋	重要結果
Potvin et al.(1996)[25]	禁制搜尋	2-opt* Or-opt	R1 總車輛數 426
Chiang and Russell(1996)[4]	模擬鍛鍊	λ-interchange k-node interchange	R1、R2、RC1、RC2 總車輛數 422
Chiang and Russell(1997)[5]	禁制搜尋	λ-interchange	R1、R2、RC2 有 7 題突破 有 16 題相同 總車輛數 411
Taillard et al.(1997)[32]	禁制搜尋	CROSS-interchange Or-opt GENIUS	R1、R2、RC1 17 題突破 20 題相同 總車輛數 416
Luca Maria Gambardella, Eric Taillard and Giovanni Agazzi(1999)[14]	螞蟻群聚法	CROSS exchanges	R1、R2、RC2 13 題突破 總車輛數 407
Haibing Li and Andrew Lim(2003)[20]	禁制模擬鍛鍊法	Shift Operator Exchange Operator Rearrange Operator	C1、C2、RC2 7 題突破 19 題相同 總車輛數 411

2.3 時間窗車輛路線問題測試例題

Solomon(1983)[29]在博士論文中為測試 VRPTW 演算法，以 VRP 例題為基礎設計 56 題 VRPTW 例題，這些題目可在網際網路上找到。此 56 題測試例題皆為單一場站，100 個顧客點，資料結構為平面座標，顧客點間之旅行時間等於旅行距離。可區分成三種型態：R 型（以均勻分配隨機產生顧客地理資料）、C 型（以分群方式產生顧客地理資料）、及 RC 型（以半分群方式產生顧客地理資料），以及兩種問題集合：R1，C1，RC1（路線時限短，車輛容量小）、和 R2，C2，RC2（路線時限長，車輛容量大），因此共可分成六類問題集：R1、C1、RC1、R2、C2、RC2。