國立臺灣大學工學院工業工程學研究所

碩士論文

Graduate Institute of Industrial Engineering
College of Engineering
National Taiwan University

Master Thesis

仿頻寬限制資料傳輸之離散優化演算法

Bandwidth Restricted Transmission-Simulated Discrete Optimization Algorithm

李仁富

Ren-Fu Li

指導教授：楊烽正 博士

Advisor: Feng-Cheng Yang, Ph.D.

中華民國 100 年 8 月

August, 2011

國立臺灣大學碩士學位論文
口試委員會審定書

論文中文題目：仿頻寬限制資料傳輸之離散優化演算法

論文英文題目：Bandwidth Restricted Transmission-Simulated Discrete Optimization Algorithm

本論文係李仁富（R98546023）在國立臺灣大學工業工程學研究所完成之碩士學位論文，於民國 100 年 7 月 29 日承下列考試委員審查通過及口試及格，特此證明

口試委員：

楊烽正 _____
（指導教授）

徐旭昇 _____

黃奎隆 _____

蔡瑞煌 _____

系主任、所長：陳正剛 _____

誌謝

研究所兩年一眨眼就過去了，承蒙楊烽正老師的嚴謹教誨，讓我體會到求學應有的態度及精神。這不僅止於做研究，也是做人處事應該具有的態度。感謝老師在寫作上面不厭其煩的教導，並不斷給予我們琢磨與建議得以順利完成碩士論文。在此致上最由衷的感謝與敬意。也很感謝蔡瑞煌教授、徐旭昇教授、黃奎隆教授在口試上的糾正與建議，特此並致謝忱。

感謝家人對我在求學過程的支持，使我能專心致力於學業當中，得以完成大學和碩士的學業。感謝同 group 的小魚和簡子堯兩位戰友，不只在學業上的互相幫助，在休閒娛樂上也能一起奮勇作戰。感謝 group 的學弟妹昱年、桓彬、敏樺、瑋婷和思婷，幫我分擔了網管的工作，也幫助我們順利的進行口試。祝福明年你們在論文上能順利得以完成碩士學位。再來感謝仁愛之家的室友們：終結、噴噴、有沒、小羅，雖然同住一個屋簷下只有短短的一年，以後還是會懷念在客廳聊天打屁的日子。接著感謝酒友呂明倫、NONO、紀柏宇，在研究之餘能一起小酌一杯、聊天講八卦，祝你們軍旅生活順利。最後，感謝女友兔兔陪伴度過忙碌的碩二生活。很感謝有工工所的各位陪伴，雖然只有短短兩年時間，但每天朝夕相處就像一家人一樣，祝福各位從工工所離開之後都能展翅高飛，往自己的理想努力邁進。

李仁富 謹誌於
國立台灣大學工業工程研究所
中華民國一百年八月

摘要

本研究提出一個創新的啟發式演算法名為「仿頻寬限制資料傳輸優化演算法」(Bandwidth Restricted Transmission-Simulated Optimization Algorithm, BRT-S)。BRT-S 的演算機制是仿網路資料傳輸的過程。BRT-S 針對解代理人求解所使用的資源加以限制，代理人須彼此競爭搶占資源才能進行解建構。當先行代理人佔用了某建構步驟上的資源，後續代理人將避開資源不足的建構步驟，改選其他尚有資源的建構步驟，保持迭代中求得解的多樣性。演算系統只針對搶得資源且成功建構解的代理人進行解評估，節省評估解所花費的運算資源。並記錄成功建構解的代理人所選擇的建構步驟，根據其解品質的優劣，添加或扣減建構步驟上的資源，加速解的演化收斂。

本研究提出的 BRT-S 是以離散優化問題為求解對象設計演算流程。研究內容並針對物件排序優化問題中的旅行銷售員問題(Traveling Salesman Problem, TSP)及物件分群優化問題中的裝箱問題(Bin Packing Problem, BPP)建立 BRT-S 求解模式再藉以開發出 BRTSDOS 求解系統。再以開發的求解系統求解 TSPLIB 和 BPPLIB 中的標竿問題，並和其他啟發式演算法比較求解結果。在求解 TSP 範例中，BRT-S 能在公平的停止條件下求得品質比其他演算法好的解，且在目標函數呼叫次數花費上較為精簡。在求解 BPP 範例中，能針對一系列不同容量屬性的標竿問題求得不違反箱子容量限制的解。本研究創新的仿頻寬限制資料傳輸優化演算法能有效地求解 TSP 和 BPP 兩種問題，且能花費較少的運算資源求得品質良好的解。

關鍵詞：啟發式演算法、旅行銷售員問題、裝箱問題、仿頻寬限制資料傳輸優化演算法

Abstract

This research presents an innovative heuristic algorithm called “Bandwidth Restricted Transmission-Simulated Optimization Algorithm” (BRT-S) for solving discrete optimization problems. BRT-S simulates the process of transferring data over the network. BRT-S restricts the resources that the solution agents use for searching solutions, so agents must compete with others to obtain the resources. The preceding agent can obtain more resources than the succeeding agent, so the succeeding agent needs to avoid the construction steps lacking of resources. Only the constructed solutions are subject to objective value evaluations for saving computing resources. Concluding the construction steps selected by successful constructed agents, resources enhancement and deduction are performed based on solution quality.

BRT-S is designed for solving discrete optimization problems. We develop BRTSDOS solving system for Traveling Salesman Problem and Bin Packing Problem through programming language. Using the benchmark of TSPLIB and BPPLIB, we compare results with other heuristic algorithm’s results to verify the feasibility of BRT-S. In the example for TSP, BRT-S can obtain better solutions and call less evolution functions than the others. In the example for BPP, BRT-S can obtain optimum number of bins in the benchmarks which have different capacity attributes items and effectively balance the load between the bins.

Keywords: heuristic algorithm, traveling salesman problems, bin packing problems, bandwidth restricted transmission-simulated optimization algorithm

目錄

圖目錄

表目錄

第一章　緒論

啟發式演算法已廣泛運用在求解各種優化問題。不同啟發式演算法在求解過程中有不同的演算機制，且各有求解品質及效率的差異。本研究研習並分析各演算法優缺點，師法網路資料傳輸日益快速精進的特性開創一新的啟發式優化演算法。同時透過求解標竿問題來驗證，所開發的啟發式演算法的成效。本章為緒論，第一節說明開創新演算法的動機；第二節敘述研究目的；第三節說明開發演算法的研究流程；最後說明本論文架構及各章概述。

1.1 研究動機

求解優化問題一直是決策科學關注的課題之一。其困難點是當問題的維度及規模越龐大時，解空間的驟增是相當可觀的。使用傳統演算法求解優化問題，不是求解效率不佳就是只能針對特定的優化問題進行求解。因此，在這數十年間興起各種智慧型的求解方式。其中廣為應用的方法就是啟發式演算法。啟發式演算法屬於有向引導式的隨機搜尋法。使用有限的運算資源，在龐大的解空間中有效地搜索最佳解。啟發式演算法的開發，往往是觀察自然界的生物習性或物理特性，進行演算機制的設計。在標準的演算機制下再依求解問題的特性微調，以適用於求解不同的優化問題。目標是希望透過啟發式演算法的演算機制，在可接受的運算時間內，求得到一個品質幾已近全域最佳解的解。

過往的啟發式演算法在求解優化問題時，可分為非建構式和建構式的求解方法。非建構式的啟發式演算法，如：遺傳演算法(Genetic Algorithm, GA)、粒子群演算法(Particle Swarm Optimization, PSO)等，通常是先隨機產生求解問題的解，再透過演算機制將解循優演化。由於在演化過程中，並不會根據問題的特性來引導搜索方向，演化出來的解其品質差異很大。並在限制條件較多問題中，須花費較多的運算資源來對違反限制的解進行修補，以確保求得的是合理解。而建

構式的啟發式演算法，如：蟻拓優化法(Ant Colony Optimization, ACO)模擬螞蟻逐步前進的方式建構解，建構過程會依問題特性引導解代理人往較佳的方向進行搜索。其特性是在逐步建構時可以避開會違反限制條件的情況，搜索出品質較好且較穩定的解。

建構式啟發式演算法在建構解時，會在較好的建構步驟上添加有利訊息，引導解代理人朝較佳的方向搜索。譬如：在蟻拓優化法中，會添加費洛蒙在螞蟻走過得路徑上。但這種演算機制常會陷入區域最佳解中，使搜索停滯不前無法覓得全域最佳解。原因在於，較佳的建構步驟存放過多好的訊息，讓解代理人容易選擇相似的建構步驟。因此本研究在新開發的啟發式演算法中，會設法限制演化迭代中建構步驟被選用的機會，以提高迭代中代理人求得的解的多樣性。同時再根據建構的解其品質高低，調整挑選次數的限制。

1.2 研究目的

本研究的主要內容是開發出一新的萬用啟發式演算法(Meta Heuristic)，針對離散優化問題進行求解。透過求解不同類型的標竿問題，和廣為人知的幾種啟發式演算法進行比較，以驗證本研究提出的啟發式演算法在求解品質及效率上是否有良好的表現。研究內容有主要下列幾點：

1. 以網路資料傳輸特性為主要架構，並參考過往啟發式演算法求解上的優缺點，開發出仿頻寬限制資料傳輸之萬用啟發式演算法(Bandwidth Restricted Transmission-Simulated Optimization Algorithm, BRT-S)。建立 BRT-S 的主要演算機制及流程，並透過程式實作 BRT-S 的求解系統。
2. 使用 BRT-S 求解物件排序及物件分群優化問題，分別以旅行銷售員問題(Traveling Salesman Problem, TSP)及裝箱問題(Bin Packing Problem, BPP)

作為求解範例。根據問題特性調整 BRT-S 的演算機制及流程，有效地進行求解。

3. 求解 TSPLIB 和 BPPLIB 內的標竿問題，並和其他啟發式演算法比較求解結果。驗證 BRT-S 在求解不同問題時，能求得品質較高的解並所花費的運算資源較精簡。

1.3 研究流程

本研究流程先蒐集並研讀啟發式演算法的相關文獻，探討各啟發式演算法的優缺點，再結合網路資料傳輸特性開發 BRT-S 演算機制。收集物件排序及物件分群優化問題的相關文獻和文獻中提及的標竿問題。接著開發求解系統並測試系統效能，最後和其他啟發式演算法進行求解比較。本研究流程圖如圖 1.1 所示。

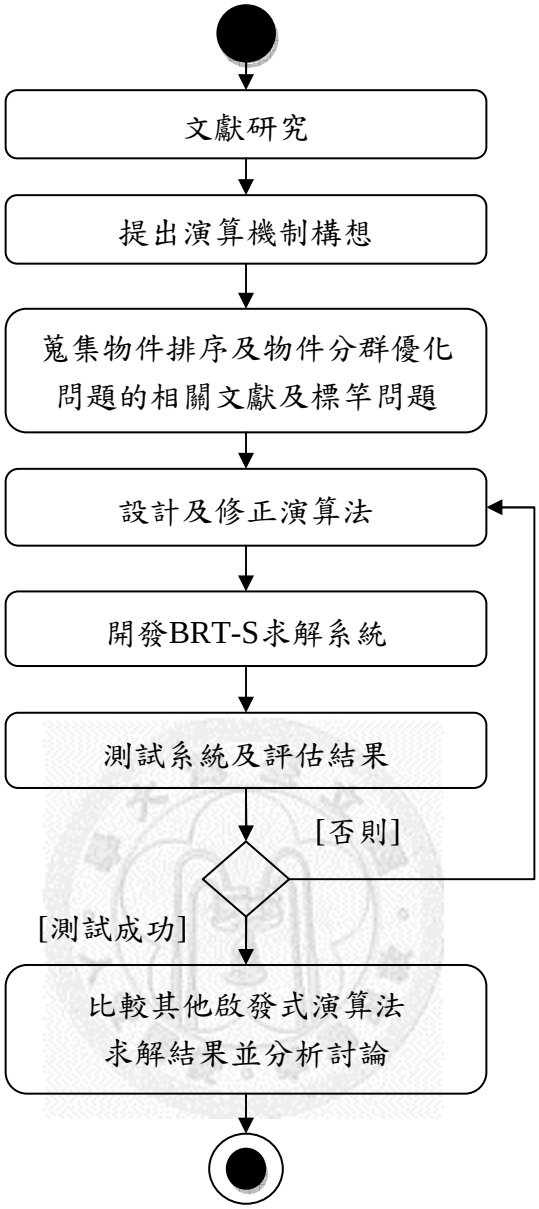
文獻研究
提出演算機制構想
蒐集物件排序及物件分群優化
問題的相關文獻及標竿問題
設計及修正演算法
開發BRT-S求解系統
測試系統及評估結果
[否則]
[測試成功]
比較其他啟發式演算法
求解結果並分析討論

圖 1.1 研究流程圖

1.4 章節概要

本論文共分五章，編排方式如下：第一章緒論，說明本研究動機、目的、及流程概述。第二章文獻回顧，介紹常見的幾種啟發式演算法演算機制及其優缺點，並研究物件排序優化問題及物件分群優化問題的問題定義及常見求解方法。第三章介紹本研究提出的仿頻寬限制資料傳輸優化演算法的相關演算機制和流程，並說明如何求解旅行銷售員問題和裝箱問題。第四章是數值範例測試及分析，求解 TSPLIB 和 BPPLIB 內的標竿問題，並和其他啟發式演算法比較。第五章歸納本研究所得結果及探討，並對未來可行的研究方向提出建議。

第二章　文獻回顧

本章將先介紹常用的啟發式演算法，敘述其演算精神和演算流程。再來，介紹本研究探討的離散優化問題及其相關研究；包含物件排序優化問題和物件分群優化問題，分別以旅行銷售員問題與裝箱問題為例。

2.1 啟發式演算法

求解優化問題有許多不同的球解法。傳統演算法在求解過程中無法保證求解品質，且大多是針對特定問題進行求解。而啟發式演算法是模擬自然界生物特性或物理現象，設計相關的演算機制來求取問題的最佳解。能在龐大的解空間中，有向且有效地進行搜索解的工作。目前啟發式演算法已廣泛應用在各種優化問題上，本節將介紹常用的啟法式演算法流程及應用。

2.1.1　遺傳演算法

遺傳演算法(Genetic Algorithm, GA)是由學者 Holland 在 1975 年所提出(Holland, 1975)，目前已廣泛應用到不同的優化問題當中。GA 的概念源自於達爾文所提出的「物競天擇，適者生存」的進化論。生物為了適應環境，需要不斷的調整自己以提升生存能力，適應力強的物種較容易存活下來，而 GA 模仿生物繁衍後代的機制建立其演算程序。生物在繁衍的過程中，兩個親代會透過染色體交配的機制分別將基因傳送給子代，獲得較好基因的子代具有較強的競爭力存活下來，在演化過程中會因為基因突變去開創新的特性去適應環境。經過汰弱存強的競爭後，最後具有高適應力的生物存活下來。

GA 的主要作業分別為編碼(Encoding)、交配(Crossover)、突變(Mutation)、篩選(Selection)反覆演化，逐步改善求解品質。

(1) 編碼(Encoding)

根據不同問題的特性，會設計出不同的編碼方式。常見的基因編碼有實數(Real)、整數(Integer)、二元(Binary)、字元(Char)、符號(Symbol)編碼。同一種問題採用不同的編碼法會影響求解品質與效率，不同編碼法會對應到不同的交配與突變方法。

(2) 交配(Crossover)

交配作業會根據使用者所設定的交配率從染色體群中挑選出數條染色體進入交配池(mating pool)當中，再從交配池當中任意兩兩配對進行交配。交配作業就如同自然界一樣是為了產生新的子代出來，適合度高的子代比適合度低的子代更有能力生存下來。常見的交配法有部分配對交配(Partial-Mapped Crossover, PMX))、順序為基交配(Order-Based Crossover, OBX)、順序交配(Order Crossover, OX)、位置為基交配(Position-Based Crossover, PBX)、單點交配(One-Point Crossover)、雙點交配(Two-Point Crossover)、多點交配等(Multiple-Point Crossover)。

(3) 突變(Mutation)

生物在演化過程中因為基因發生突變現象而產生新的物種，新的物種可能擁有比原物種更強的適應力而取代原物種生存下來。在遺傳演算法中，突變作業在交配過程中每一條新的子代都有機會發生突變，透過突變作業可使陷入區域最佳解情形跳脫出來。常見的突變方式有基因插入(Insertion)、基因交換(Exchange)、基因反轉(Inversion)、部分基因取代(Displacement)等。

(4) 篩選(Selection)

根據達爾文「物競天擇，適者生存」的觀點，適應力強的物種比適應力弱的物種容易存活下來。在遺傳演算法中，透過篩選作業適應度較高的染色體有較高的機率被選取，讓他們能夠繼續演化下去。因為是透過機率的方式篩選，所以適應度較低的染色體仍有機會被選取到。常見的篩選方式有菁英篩選、輪盤式篩選等。

只要找到適合的編碼方式產生問題的解，即可透過 GA 的演化機制進行求解，使 GA 能廣泛地被用於求解不同的優化問題。演化過程中 GA 的解代理人須經過競爭，保留求解品質較好的代理人持續參與演化，使整體演化能保持往較好的方向前進。然後 GA 的代理人是以跳動較大的移動方式在解空間搜索，無法保證代理人的解是循優演化。

2.1.2 蟻拓優化法

蟻拓優化法(Ant Colony Optimization, ACO)是在萬用啟發式演算法之中，求解品質相當不錯的演算法之一。演算精神是源自於螞蟻覓食的行為，找尋螞蟻往返蟻穴和食物間的最短路徑。螞蟻在移動的過程中會分泌一種獨特的賀爾蒙稱做費洛蒙(Pheromone)，螞蟻透過費洛蒙作為彼此間的溝通方式，線段上殘留的費洛蒙濃度越高越能吸引螞蟻選擇此路徑通過。隨時間的遞移，各路線段上殘留的費洛蒙濃度和螞蟻經過的頻率，在較短的線段上將殘留濃度較高的費洛蒙。久而久之，最短路徑上將殘留濃度較高的費洛蒙，而螞蟻也傾向於選擇此路徑通過，如圖 2.1 所示。

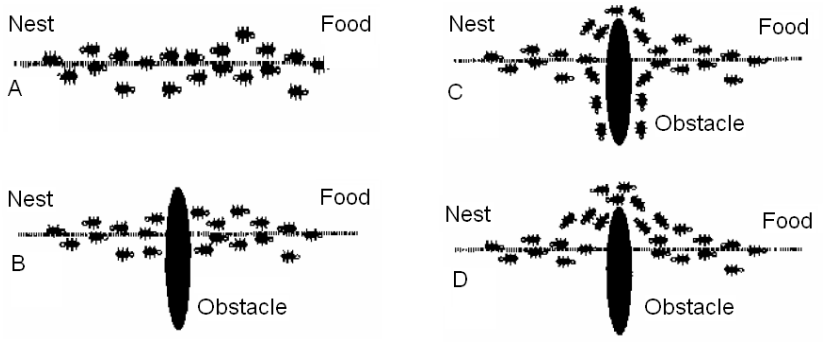

圖 2.1 螞蟻覓食行為圖

圖中越多螞蟻走過的路徑則遺留下的費洛蒙越多，而遺留越多費洛蒙又會吸引越多螞蟻行走該條路徑。因此，當螞蟻面臨兩條路徑以上之抉擇時，其行走某一路線之機率與其遺留費洛蒙量有關，而越短之路線其螞蟻通過的時間越短，使得最短路線上遺留之費洛蒙量越多，進而誘使更多螞蟻依循最短路徑前進，漸漸所有螞蟻將依循最短路徑行走。較短之路徑所需行走的時間較短，費洛蒙被蒸發的時間也較短，容易殘留較多之費洛蒙，因而吸引較多隻螞蟻，最後螞蟻將依循最短路徑，而此路徑及為最佳解。ACO 透過模仿螞蟻行為，以進行最佳化求解工作。

蟻拓優化法最早是由 Dorigo 於 1991 年他的博士論文所提出的螞蟻系統(Ant System, AS)而來，文獻中提出了螞蟻系統的三種模式，Ant-density、Ant-quantity 和 Ant-cycle，其差別在於採取不同的費洛蒙殘留更新方式，並成功地運用在解旅行銷售員問題(Traveling Salesman Problem, TSP)。Dorigo 和 Gambardella (1997) 文獻中提出蟻群系統(Ant Colony System, ACS)，改良了建構步驟時的挑選法則，新增虛擬隨機比例法則(pseudo-random-proportional rule)，以固定的比例使螞蟻挑選線段上費洛蒙濃度和能見度乘數最大者，縮減螞蟻搜索時間以減少整體運算所花費的時間。並將費洛蒙更新區分為全域費洛蒙更新與區域費洛蒙更新。ACS 改良 AS 不足的地方，提升了求解品質及效率。Dorigo、Maniezzo、Colorni (1996)

提出根據 AS 所改良的菁英螞蟻系統(Elitist Ant System, AS_elite)，採用菁英策略進行費洛蒙更新。對迭代中表現較佳螞蟻所選擇的線段添加費洛蒙濃度，加強演化求解的收斂效果。而 Bullnheimer、Hartl 和 Strauss (1997)的文獻中提出分等螞蟻系統(Rank-based Ant System, AS_rank)，不同於 AS_elite 在費洛蒙更新上，是對表現較佳的螞蟻其解線段上添加相同的費洛蒙濃度。採用分等的方式將螞蟻依照求解品質進行排序，取前面表現較好的螞蟻添加其解線段上的費洛蒙，排序越前面添加的費洛蒙量越多。極大-極小螞蟻系統(Max-Min Ant System, MMAS)是由 Stutzle 和 Hoos (1997)文獻中所提出，針對線段上的費洛蒙濃度加以限制，訂定了費洛蒙濃度的上下界。當線段添加費洛蒙超出上界時，則不再進行費洛蒙的添加；反之，線段上費洛蒙蒸發到下界時則不再扣減其費洛蒙，讓該線段保留少許被挑選到的機會。透過設定費洛蒙濃度上下界使搜索解過程不易停滯下來，進而增加跳出區域最佳解的機會。而 Yang 和 Chou (2009)提出的優加劣減蟻拓系統(Superior/Inferior Segment-Discriminated Ant System, SDAS)，根據傳統的蟻拓法容易陷入區域最佳解而造成搜索停滯的現象進行改善。設計出較佳較差界限法將迭代中螞蟻依照求解表現區分優劣，對較佳螞蟻建構解的線段額外添加費洛蒙，反之，對較差螞蟻建構解的線段扣減費洛蒙。透過優加劣減廢費洛蒙更新策略，使螞蟻更有效率的建構出最佳解。

求解離散優化問題中，蟻拓最佳化法相較於其他啟發式演算法有較突出的表現。主要原因在於，蟻拓最佳化法需根據求解問題的特性設計費洛蒙網圖，透過建構式的求解方式逐步將解建構出來。過程中透過求解資訊記憶的方式，讓解代理人能有效地往較佳的方向進行建構。但缺點在於持續在較佳線段上添加費洛蒙，使迭代中螞蟻皆挑選較佳線段進行建構，建構出過度相似的解浪費運算資源。

2.1.3 粒子群演算法

Kennedy 和 Eberhart (1995)提出粒子群最佳化演算法(Particle Swarm Optimization, PSO)，藉由觀察鳥群和魚群的群體行為發展的啟發式演算法。個體間具有特別的訊息傳輸方式，將個體所得的資訊提供群體進行經驗交流，使群體朝相同目標前進。PSO 模仿群體的經驗交流和相互修正前進方向的方式來求解優化問題。PSO 的基本演算流程如下：以隨機產生初始粒子群，每個粒子即代表求解問題的一個解。粒子移動過程中，會參考自己的最佳經驗，以及群體的最佳經驗，修正移動的方向。經由粒子間不斷相互修正移動方向，最後粒子群體會逐漸收斂在最佳解的位置上。

以鳥群的覓食行為而言，鳥即是 PSO 中定義的粒子。剛開始所有鳥皆不知食物的位置，只能以隨機的方式選擇方向及移動。單隻鳥雖無法得知食物的確切位置，但仍可透過感官來判斷附近是否有食物存在以及距離現在位置多遠。單隻鳥會根據過往經驗和食物最近的位置，逐漸修正飛行方向往食物的位置前進。鳥群間有資訊溝通的能力，知道整體鳥群的搜索過程中離食物最近的位置時，各鳥除了根據自身經驗外，也會參考群體的經驗，增加找到食物的可能性。這樣的覓食模式可分為兩個部分：

1. 認知模式(Cognition-Only Model)：生物個體會記憶自我的最佳經驗。PSO 的粒子會記憶自身的迄今最佳解資訊。

2. 社會模式(Social-Only Model)生物群體具有社會性，會彼此交流所得到的經驗並傳承下去。PSO 的群體會記錄群體的迄今最佳解資訊。

問題的最佳解即是鳥群尋覓的食物所在。透過自身經驗與群體分享，驅策群體朝最佳解移動。PSO 如同 GA 是先根據問題特性產生初始解，透過自身及群體的經驗，逐步修正各代理人的移動方向，進而演化出求解問題的最佳解。當解代

理人皆朝同一方向(目前最佳解方向)前進，過程中也未找到更好的解來改變群體移動方向時，代理人會逐漸收斂在同一區域內，也無法像 GA 有突變機制能跳脫區域最佳解。在一開始群體如果無法分布廣布在整個解空間時，有極大的機會陷入區域最佳解當中。

2.2 離散優化問題

常見的離散優化問題有物件排序和物件分群優化問題，前者以旅行銷售員問題(Traveling Salesman Problem, TSP)為代表，後者則以裝箱問題(Bin Packing Problem, BPP)為代表。本節將介紹此兩種問題以及其相關研究。

2.2.1 旅行銷售員問題

旅行銷售員問題(Traveling Salesman Problem, TSP)最早是由 Hassler (1934)提出。旅行銷售員問題是探討如何尋找一條最短的封閉迴路，銷售員需要拜訪若干個城市，每個城市都只能經過一次，最後回到所出發的城市。旅行銷售員問題已經被證明屬於 NP-Complete 的問題，雖然沒有複雜的限制條件，但問題的求解空間隨城市數量增加成指數成長，當有問題中有 n 個城市時，將會有 $\frac{(n-1)!}{2}$ 個可行解。要在龐大的解空間中找尋出最佳解，是相當困難且需花費極高的時間成本。至目前為止，大型的旅行銷售員問題尚未有一套有效的搜索方式，能確保能求得問題的最佳解。旅行銷售員問題由 Hassler 提出以來，至今已有許多演算法針對它進行求解，嘗試有效地找出問題的最佳解。下表為整理針對旅行銷售員問題進行求解的演算法；

表 2.1 常見求解 TSP 問題的啟發式演算法

演算法名稱	首次求解作者	年份
模擬退火法(SA)	Kirkpatrick	1983 年
遺傳演算法(GA)	Goldberg and Lingle	1985 年
禁忌演算法(TS)	Glover	1989 年
門檻接受法(TA)	Dueck and Scheuer	1990 年
螞蟻系統(AS)	Dorigo	1997 年
粒子群演算法(PSO)	Wang and Hung and Zhou	2003 年
仿水流優化演算法(WFA)	Yang and Wang	2007 年

2.2.2 裝箱問題

根據 Garey 等人 (1976)定義裝箱問題(Bin Packing Problem, BPP)如下：給定一個有限制的物品容量屬性集合W，每一物件都有其對應的容量屬性值w_i，在已知箱子數量$\bar{x}$和箱子容量上限L，要如何將所有物件放入$\bar{x}$個箱子當中，或用更少箱子來裝箱。裝箱問題為典型的物件分群優化問題，當每增加一個物件時，求解空間也隨其成指數成長，被歸類於 NP-Hard 問題。必須透過啟發式演算法來降低求解所花的時間成本，有效地找尋全域最佳解。下表將整理針對 BPP 進行求解的啟發式演算法；

表 2.2 常見求解 BPP 問題的啟發式演算法

演算法名稱	首次求解作者	年份
模擬退火法(SA)	Dowsland	1993 年
遺傳演算法(GA)	Falkenauer	1994 年
禁忌演算法(TS)	Bennell and Dosland	1999 年
鄰近搜尋法(VNS)	Hansen	1999 年
螞蟻系統(AS)	Frederick	2003 年
仿水流優化演算法(WFA)	Yang and Wang	2007 年

過往啟發式演算法求解離散問題時，以蟻拓最佳化法這類的建構式方法，能較快速地求得問題的最佳解或近似最佳解。而採取直接產生離散問題的解，再透過演算機制循優演化的基因演算法、粒子群演算法等。由於不像蟻拓法這類的建構式演算法，亦步亦趨地避開問題限制條件以及利用問題中各物件的特性進行解建構，使求解就像在解空間中漫無目的的搜索一般，需花費較多的運算時間，也有可能演化不出問題的最佳解。綜觀現有啟發式演算法的優缺點，及離散問題的問題特性。要有效求解離散優化問題，須依照問題特性來做為引導解代理人建構方向的指標。並避免同迭代中建構出的解過於相似，應限制代理人選用相同建構步驟的機會，以增加迭代中解的多樣性。最後，須適時調整代理人選用建構步驟的機會，以達到演化收斂。根據上述觀點，本研究師法網路資料傳輸特性開創一新的啟發式優化演算法，其演算法細節將在下一章進行說明。

第三章　仿頻寬限制資料傳輸優化演算法

綜觀現有廣為使用的啟發式優化演算法，每一個解代理人，如染色體、螞蟻、粒子，在每一迭代的演化中都需進行解的評估。即使兩個代理人求得的解完全相同仍會分別進行評估，造成運算資源的浪費。本研究創新開發的仿頻寬限制資料傳輸優化演算法(Bandwidth Restricted Transmission-Simulated Optimization Algorithm, BRT-S)。規劃代理人彼此間存在「競合」關係。透過競爭有限資源再合作演化的機制進行求解演化，避免演化過程中所有代理人都進行解評估。僅讓搶得資源完成解建構的代理人進行評估，可減少解評估的資源花費。同時透過各解代理人的資訊記憶，保留有利的資訊以提升解搜索的深度和品質。

3.1 網路資料傳輸特性

當今社會人們廣泛地透過網路收集資料、傳輸訊息。傳輸訊息時使用者間透過各個不同的傳輸節點間的連結纜線進行傳輸。網路上各段線路有一定數量的頻寬供使用者使用。網路頻寬資源是有額度的撥給每個使用者，因此傳輸者會佔用一定數量的「頻寬」。當線路中已經有使用者佔用部分頻寬在傳輸訊息時，後續的使用者只能使用剩餘的頻寬進行傳輸。如果剩餘頻寬無法供後續使用者傳輸時，訊息將會透過傳輸節點轉由其他具足夠剩餘頻寬的線路傳輸出去。因此，先傳輸的使用者有較充裕的頻寬資源，而後續的使用者可使用的頻寬資源就相對較少。此外，由於網路纜線普遍鋪設在地底下，會因為環境中的氣溫、濕度或外來的震動、挖掘擠壓造成纜線耗損。纜線的頻寬總額、傳輸品質也會因此減少和變差。本研究提示的仿網路資料傳輸演化與優化問題的解搜索的對應概念說明如下：

1. 傳送及接收點間的網路纜線越長，傳輸所需花費的時間也越長；反之，則越短。在傳輸資料的過程中，資料會透過網路傳輸節點的選擇，經由花費時間較短的線路進行傳輸。在演算法中解的建構步驟移轉是透過規劃的啟發法則來輔助解代理人進行解的建構。在求解優化問題中，會傾向挑選較有利的線段，如：旅行銷售員問題解建構步驟會傾向挑選較短的線段。此概念和網路傳輸會通過花費時間較短的線段相似，可作為解代理人在解建構步驟中挑選「物件」的依據。

2. 頻寬配額是有限的，先行使用的人可以順利佔用較多的配額，後續者將只能使用剩餘的頻寬。當預計通過的線路頻寬不足使用時轉由其他線路傳輸，若無任何線路可用則傳輸失敗。過往其他啟發式演算法沒有限制代理人的求解資源。透過資源限制的概念，可避免代理人往同方向建構解。可行的作法是在演化過程中，降低和先行代理人選擇相同建構步驟的機率，如同部分頻寬已被先行代理人佔用。如此可以增加其他建構步驟的挑選機率，建構不同的解。

3. 網路公司進行纜線汰舊布新或重新鋪設時，依據過往纜線上的流量統計資料，對使用流量較大的線段更換能提供頻寬較多的纜線。因此若根據解代理人的表現調整其建構步驟上的求解資源，使代理人能經由眾人求解過程的資訊，應能有效地往較佳的方向進行建構解。

4. 網路纜線鋪設在實體環境下，受氣溫的冷熱、溼度的變化以及纜線材質的劣化、老化會產生纜線的耗損。模擬纜線耗損的現象，演化過程中求解資源會隨時間逐漸減少，降低解代理人選取相同建構步驟，提升迭代中解的多樣性。

3.2 仿頻寬限制資料傳輸優化演算法演化機制

啟發式演算法求解優化問題時，是透過代理人以規劃妥當的演算程序在求解空間中進行解的搜尋或建構。經由經驗法則為基的啟發式導引，在求解過程中逐步選擇解建構中的各個物件，建構具有一定品質的解。再透過群體的合作演化出類全域最佳解。本研究提示的求解法師法網路資料傳輸的特性，讓代理人在建構解過程透過競合機制，盡量避免重複解的產生，提高解的多樣性。再加上啟發式的運算讓代理人有向地逐步選擇建構解過程中的物件。歸納網路資料傳輸特性，說明本研究提示的求解法使用的演化機制：

1. 解代理人登錄起始點：常見的建構式啟發式演算法中，如：拓樸法，多是隨機挑選離散問題的物件當作解建構的起始點以進行廣域搜索。譬如：隨意設定旅行銷售員問題的起始城市、裝箱問題的起始配置物品。解建構時挑選不同的物件，會影響後續其他物件被挑選的機率。以圖 3-1TSP 著重城市間距離長短選擇為例：由 A 城市出發挑選 B,C,D,E 的機率差異較小；而由 E 點出發挑選 A 的機率與挑選 B,C,D 的機率差異較大。在演算法中加入代理人登錄的起始點屬性，供解建構時使用。若迭代某代理人一再沿用同一起始點時，類似於搜索鄰近較佳解。

圖 3-1 不同起始點在挑選候選元件的差異

2. 資源競爭及佔用： 典型啟發式演算法的代理人在建構解時，對於解建構過程中物件的選取並無限制。因此會有多個解代理人選取相同的物件序列導致重複解產生，浪費重複解的評估運算。本研究根據資源有限的觀點，讓解代理人須競爭資源以建構解。當先行代理人已佔用部分資源時，讓後續代理人能往其他資源較充裕的方向建構，降低選取相同的建構步驟。

3. 擴充縮減求解資源：在建構式啟發式演算法中，通常記錄解代理人的建構步驟，依照設計的法則在這些建構步驟上擴充或縮減正向資訊，如：優加劣減蟻拓法以較佳較差界限法將螞蟻分出優劣，分別對其建構步驟添加獲扣減費洛蒙。本研究在開發的演算法中加入擴充縮減求解資源的演算機制，依照解代理人每一迭代建構解的品質提升或下降，對其建構步驟上的求解資源進行擴充或縮減。能增加代理人挑選品質較佳建構步驟的機會，而減少挑選求解品質下降者的機會，讓求解演化能達到收斂的效果。

4. 隨機耗損求解資源：大多數的啟發式演算法是在好的解建構步驟上添加正向資訊，增加被依循的機率。例如：ACO 在演化過程中會添加費洛蒙在好的解路段上，讓後續的螞蟻代理人依循費洛蒙的強弱建構類似解。但持續在好的解步驟上添加正向資訊，容易過早收斂在區域最佳解中。本研究開發的演算法中，讓求解資源在演化中隨機的耗損。並和擴充或縮減求解資源的演化機制相輔相成，使較少被挑選的建構步驟上的求解資源逐漸減少，降低解代理人建構時挑選的機會。

本研究提出的演算法當中以網路資料傳輸特性為主軸，加入登錄起始點、求解資源競爭、擴充縮減求解資源和隨機耗損等演化機制，開發出仿頻寬限制資料傳輸優化啟發式演算法。在下一節將對整體演算細節加以說明。

3.3 仿頻寬限制資料傳輸優化演算法應用及演化流程

本研究開創的仿頻寬限制資料傳輸優化演算法(Bandwidth Restricted Transmission-Simulated Optimization Algorithm, BRT-S)是一種建構式啟發式演算法，由解代理人逐步建構一個優化問題的解，以下簡稱本法為 BRT-S 法。建構解過程是一次處理一個目標問題中所定義的物件，如：旅行銷售員問題中的城市。旅行銷售員問題的建構步驟是一次選擇一個城市直到完成迴圈的建構。實際上離散優化問題依各自的特性可容易地定義出物件群組，如：裝箱問題中有物品群組與箱子群組。

透過 BRT-S 求解優化問題時，需要先定義問題的物件與其物件連結。在 BRT-S 中物件被視為傳輸節點，這些物件可能來自不同的物件群組；而物件連結被視為具有頻寬限制級傳輸能力的纜線。傳輸節點與纜線線段構成訊息傳輸的網路系統。

令傳輸節點集合 $N=\{1,2,\cdots,\bar{n}\}$；$\bar{n}$ 是傳輸節點總數；而線段集合 $E=\{(i,j)|i,j\in N\}$，其中線段 (i,j) 是節點 i 與 j 間的傳輸纜線。當目標問題有多個節點群組時，節點 i 與節點 j 可能並不屬於同一個節點群組中。因此 N 集合的通式是 $N=\bigcup_{i=1}^{g}N_i$，其中 g 是節點群組數量，N_i 是第 i 個節點集合。令 n_i 是節點集合 N_i 中的節點總數，則 $n_i=|N_i|$ 且 $\bar{n}=\sum_{i=1}^{g}n_i$。BRT-S 定義這些節點為傳輸節點且線段為連結傳輸節點的實體纜線。最先，對每一個線段 (i,j) 配置固定數量的頻寬配額 $b_{i,j}=b_0$， b_0 是使用者設定的頻寬初始配額量。線段 (i,j) 上的頻寬配額 $b_{i,j}$ 設為整數，方便 BRT-S 在演化過程中進行評估。

線段上的頻寬配額會在進入下一迭代時，根據訊息傳輸的結果執行添加或扣減。另外，纜線以及傳輸品質在演化過程中，會受到環境造成的損耗與纜線本身的老化而減損。因此，當模擬完所有解代理人訊息傳輸作業後，會一併更新各線段(i,j)的頻寬配額$b_{i,j}$。除了頻寬配額，每一線段(i,j)會依問題特性定義一個啟發式評估值$h_{i,j}$。在解建構過程中，線段的選擇會參考頻寬配額的值與啟發式評估值。使用 BRT-S 求解優化問題，須定義和計算所有線段的啟發式評估值。但如果啟發式評估值設成動態，須在演化過程中適時進行調整。BRT-S 的優勢之一是透過啟發式評估值的取得和求算，有向地搜索問題的最佳解。一般而言，線段(i,j)上的啟發式評估值$h_{i,j}$越大，代表分配被挑選的機率值也越高。在旅行銷售員問題中，啟發式評估值$h_{i,j}$可設成城市i與j間距離的倒數，因為選擇距離較短的線段較能建構出一條最短路徑。

BRT-S 是由群體解代理人進行解的建構，並讓代理人間相互競爭與合作朝最佳解演化。BRT-S 中的解代理人即為訊息傳輸者，透過選擇傳輸節點間的纜線段傳輸訊息。透過迭代演化反覆模擬所有訊息傳輸者傳輸，直到滿足停止條件。令訊息傳輸者集合為$M=\{1,2,...,m\}$，其中m是訊息傳輸者總數，以下簡稱傳輸者。在每一迭代的傳輸作業中，傳輸者會依循設定的傳輸順序傳輸訊息。在此以一序列表示傳輸順序，$O=[o_1 o_2 ... o_m]$, $o_{k'} \in M$, $k'=1,2,...,m$。此序列的傳輸順序會在每迭代中評估傳輸者的表現優劣調整變更。初始時傳輸順序是依索引順序設定。傳輸纜線上的頻寬配額有資料傳輸量的限制，先行傳輸者比後續傳輸者有較充裕的頻寬配額。傳輸者使用的頻寬是由頻寬初始配額量b_0、傳輸者總數m和他的傳輸順序k'設定。第k'個進行傳輸的傳輸者$o_{k'}$使用的頻寬量是$c_{k'}=\left\lfloor \frac{(2-m)k'+m^2-2}{2m^2-2m}\cdot b_0 \right\rfloor$。頻寬使用量是根據傳輸順序的順位大小依比例設

定。依此公式設定第一順位傳輸者o_1使用的頻寬是$c_1=\frac{b_0}{2}$，而最後傳輸者o_m使用的頻寬是$c_m=\frac{b_0}{m}$。

一個訊息能夠成功地被訊息傳輸者$o_{k'}$傳輸出去，其選擇通過的所有線段上的頻寬量必須滿足其頻寬使用量$c_{k'}$。一旦傳輸者成功地將其訊息傳輸出去，代表一個優化問題的解由選擇的線段建構成功。一個解可被表示成一組線段序列，$S=\left[\left(s_1,s_1'\right),\left(s_2,s_2'\right),\cdots,\left(s_n,s_n'\right)\right];s_i,s_i'\in N$，或者依照問題特性簡化成一組節點序列，$S=\left[s_1s_2...s_n\right],s_i\in N$。式中$n$是完整構成一個解所需的線段(建構步驟)總數，而$n$的大小決定於問題特性。在單一物件群組的優化問題中，n等於物件總數$\bar{n}$，如旅行銷售員問題的城市總數。而在多物件群組問題中，n可能取決於其中一個物件群組的物件數量，即$n=\left|N_{i^*}\right|,i^*\in\{1,2,...,g\}$。譬如裝箱問題中有物品群組與箱子群組，線段是連接物品與箱子的置入關係。因為裝箱問題的解序列是物品與箱子的置入線段，則n即是物品總數並非箱子總數，也非箱子和物品數的和。

在模擬訊息傳輸的過程中，傳輸者可能因無具足夠頻寬的線段可選用導致傳輸失敗無法成功建構一個解。令β_{ij}是傳輸過程線段(i,j)上當下的剩餘頻寬量。每一迭代開始前會初始化成線段(i,j)上現有的頻寬配額，即$\beta_{ij}\leftarrow b_{ij}$。在解的建構過程中，會因某傳輸節點沒有任何連接線段的頻寬足夠所需且不違反問題的限制條件，因而造成傳輸失敗。因此，傳輸者$o_{k'}$每步驟可供選擇的候選線段不得違反問題的限制條件，且剩餘頻寬量β_{ij}必須大於使用量，即$\beta_{ij}\geq c_{k'}$。當沒有任何候選線段可被選取時，解建構過程無法繼續而傳輸失敗。演化過程無法保證所有傳輸者都能成功地傳輸。他們必須透過競爭有限的頻寬資源來進行傳輸(建構解)。如同真實世界裡有些傳輸者能成功傳出，有些則失敗。令$\tilde{M}$是當前迭代中成功傳輸者的集合，而失敗者集合則是$M-\tilde{M}$。

傳輸者建構解是在建構步驟中，從候選節點或線段集合中挑選一個節點或線段進行解建構。在求解演化過程中，為了能保證訊傳輸者能有效地搜索較佳的解，每個傳輸者登錄有一個起始節點或起始線段，作為解建構的第一個步驟。傳輸者 k 的起始線段和起始節點分別以 $\left(s_1^k, s_1'^k\right)$ 和 s_1^k 表示。假設傳輸者 k 的傳輸順序為 k'，則 $k = o_{k'}$。一開始每一個傳輸者的起始節點或起始線段都是隨機設定，只有在無法成功地建構出解時才會變更。令序列 S_k 是當次迭代傳輸者 k 的建構解，在建構解開始之前令 $S_k \leftarrow \left[\left(s_1^k, s_1'^k\right)\right]$ 或 $S_k \leftarrow \left[s_1^k\right]$。在解序列中先加入所登錄的起始節點或起始線段，再開始解建構的運算步驟。之後，傳輸者 k 一次執行一個解建構步驟，直到完成一個解或傳輸失敗。在每一個建構步驟中，傳輸者根據當下部分完成的解去設定線段的候選集合 $J = \{(j_1, j_1'), (j_2, j_2'), \cdots\}, (j_i, j_i') \in E; j_i, j_i' \in N$。建立候選線段 (或對應的節點) 集合時不得違反任何問題的限制式，同時候選線段上的剩餘頻寬不得小於傳輸者 k 的傳輸使用量，即 $\beta_{j_i, j_i'} \geq c_{k'}$。當沒有任何候選者供選擇時，$J = \{\ \}$，傳輸者 k 在當次迭代中訊息傳輸失敗。解建構過程改由下一個傳輸者執行，即 $o_{k'+1}$。當 J 中的候選者多於一個時須進行挑選作業，先計算每個候選線段被挑選的機率值，屬於 J 中線段 (j_i, j_i') 的機率值

$$p_i = \beta_{j_i, j_i'} \cdot h_{j_i, j_i'} \tag{3.1}$$

是剩餘頻寬與啟發式評估值的乘積。較多的剩餘頻寬量伴隨著較高的啟發式評估值將獲得較高的被挑選機率。此外使用者須設定確定型模式使用率 λ，$0.5 \leq \lambda < 1$ 用以確定挑選的模式是確定型或隨機型。因此訊息傳輸者有 λ 和 $1-\lambda$ 的機率分別使用確定型及隨機型挑選模式，由 J 中挑選一個線段加入解中完成一個解建構步驟。若確定型模式挑選的線段是 (j_{i^*}, j_{i^*}')，則 $i^* \leftarrow \arg \max_{\forall i=1,2,\cdots,|J|} \{p_i\}$。隨

機型模式則是採用機率比例方式挑選。首先，將所有挑選機率標準化後組成一個機率池，配置方式如同輪盤法。令 $p'_i = \frac{p_i}{\sum_{\forall i \in J} p_i}, i = 1,2,...,|J|$ 是標準化後候選者 i 的被挑選機率。挑選時產生一個介於 0.0 到 1.0 間的隨機亂數以挑選出線段 i^*，符合 $\begin{cases} \square\, U(0,1) \le p'_i, \text{if } i^* = 1 \\ \sum_{i=1}^{i^*-1} p'_i < \square\, U(0,1) \le \sum_{i=1}^{i^*} p'_i, \text{otherwise} \end{cases}$。將所挑選的線段 (j_{i^*}, j'_{i^*}) 加入解序列當中，並轉移建構步驟到下一步驟。

當傳輸者 k 成功地建構出解，則解是 $S_k = \left[\left(s_1^k, s_1'^k\right), \left(s_2^k, s_2'^k\right), \cdots, \left(s_n^k, s_n'^k\right)\right]$；$s_1^k, s_1'^k \in N$。解中包含的線段上的剩餘頻寬將扣除使用量 $c_{k'}$，$\beta_{s_i^k, s_i'^k} \leftarrow \beta_{s_i^k, s_i'^k} - c_{k'}$；$i = 1,2,\cdots,n$。同時將 k 加入成功傳輸者的集合當中，即 $\tilde{M} \leftarrow \tilde{M} \cup \{k\}$。

假設建構解 S 的目標函數值定義為 $f(S)$ 且為望小優化問題。在每一迭代訊息傳輸作業結束後，只有完成建構的解需進行目標函數值評估。為了計算目標函數值的改善量，每一個傳輸者需要記錄前次成功地建構出解的目標函數值。令 f_k 和 f'_k 分別為傳輸者 k 當次迭代與前次(不一定是上一迭代)所獲得的目標函數值，則 $f_k = f(S_k)$，其中 S_k 是當次迭代中被傳輸者 k 建構出的解。傳輸者 k 的目標函數值改善量是 $\Delta f_k = \begin{cases} f'_k - f_k, \text{if } k \in \tilde{M} \\ 0, otherwise \end{cases}, k = 1,2,\cdots,m$。未完成解建構者，改善量逕設為 0。

目標函數值評估完後，將 $\tilde{M}$ 中的傳輸者前次目標函式值更新為當次目標函數值，即 $f'_k \leftarrow f_k, \forall k \in \tilde{M}$。因為 BRT-S 是建構式演算法，會一直記錄迄今最佳解 S^* 與其目標函數值 f^*。因此，如果在 $\tilde{M}$ 當中的傳輸者所獲得的解品質優於 S^*

時，更新迄今最佳解及其目標函數值；即如果 $f(S_k) < f^*$ 且 $k = \arg\max_{k \in \tilde{M}} \{\Delta f_k\}$，則更新 $f^* \leftarrow f(S_k)$ 及 $S^* \leftarrow S_k$。

當所有傳輸者都執行過訊息傳輸作業以及對迄今最佳解進行更新後，檢查所設定的停止條件。BRT-S 允許使用者設定迭代數上限 $T^{(iter)}$、目標函式值評估次數上限 $T^{(obj)}$、迄今最佳解連續無改善累計次數上限 $T^{(imp)}$ 和 CPU 運算時間上限 $T^{(cpu)}$。為了評估停止條件，分別設定下列對應的變數：目前迭代數 $t^{(iter)}$、目標函數值評估累積次數 $t^{(obj)}$、迄今最佳解連續無改善累計次數 $t^{(imp)}$ 和 CPU 運算時間 $t^{(cpu)}$。在求解演化過程中，這些變數相對應地更新，如：一旦執行了一個目標函數值評估計算，即增加 $t^{(obj)}$ 的計次一次。停止條件檢查作業檢視計次用的變數是否達到使用者設定的上限，如果符合則停止 BRT-S 演算並回傳迄今最佳解給使用者，如：$t^{(obj)} \geq T^{(obj)}$。否則，繼續執行進入下一迭代的演化作業。

進行下一迭代前，須模擬線段的自然劣化並進行人工擴充或縮減頻寬作業。自然劣化作業是根據使用者設定的「劣化率」δ，隨機扣減某些線段上的頻寬配額量，其中 $0 \leq \delta \leq 0.1$。因此，這項作業會隨機挑選 $\delta \cdot |E|$ 個線段扣除一固定量的頻寬配額 ρb_0。扣減量中的 ρ 是使用者設定的「頻寬配額增減比例」，建議值域範圍設定成 $0 < \rho \leq 0.05$。另外，為了維持線段上可供傳輸的最小頻寬配額，令 $\underline{u}$ 是使用者設定的「頻寬配額下限比例」，值域為 $0 < \underline{u} \leq 0.5$，建議值是 $\underline{u} = \frac{1}{m}$。假設隨機挑選的頻寬劣化線段集合是 D，則頻寬劣化作業是 $b_{i,j} \leftarrow \max(b_{i,j} - \rho b_0, \underline{u} b_0), \forall (i,j) \in D$。當 $b_{i,j} - \rho b_0 < \underline{u} b_0$ 時，頻寬配額設為下限量 $\underline{u} b_0$；反之頻寬配額扣減為 $b_{i,j} - \rho b_0$。

網路管理人員會統計及分析哪些纜線在訊息傳輸中頻繁地被使用，哪些較少被使用。並根據各線段的使用率調動頻寬配額，以改善傳輸效能。人們可能會用

光纖纜線取代某段電纜增加頻寬配額量，或在較少被使用的線段上騰出部分頻寬給其他需要增加頻寬的線段。BRT-S 檢視較佳或較差解內線段分別執行人工擴充或縮減頻寬配額。「較佳解」挑選的線段接受頻寬配額量的添加，而「較差解」的線段則受頻寬配額的扣減。較佳解是解的目標函式值有正向的改善量，即解 $S_k, k \in \tilde{M} \wedge \Delta f_k > 0$；相反地，較差解是 $S_k, k \in \tilde{M} \wedge \Delta f_k < 0$。未完成解建構者因改善量為 0，不參與擴充或縮減作業。因為解是一個線段序列，如 $S = \left[(s_1, s_1'), (s_2, s_2'), \cdots, (s_n, s_n') \right]; s_i, s_i' \in N$，可透過拆解運算取得線段集合。本研究定義解的拆解運算子是 $H(\cdot)$，運算結果是包含於解中的線段集合，即 $H(S) = \{ (s_1, s_1'), (s_2, s_2'), \cdots, (s_n, s_n') \}$。

頻寬擴充作業是在較佳解內的線段上添加固定的頻寬配額 ρb_0，多次的頻寬添加可能發生在被多個較佳解包含的線段上。為了避免過度添加頻寬配額，使用者需設定「頻寬配額上限比例」$\bar{u}$，值域為 $1 < \bar{u} < 2$，建議設定 $\bar{u} = 1 + \frac{1}{m}$。因此，頻寬添加作業是 $b_{i,j} \leftarrow \min\left(b_{i,j} + \rho b_0, \bar{u} b_0 \right), \forall (i,j) \in H(S_k) \wedge \Delta f_k > 0$。相對地，頻寬扣減作業則是 $b_{i,j} \leftarrow \max\left(b_{i,j} - \rho b_0, \underline{u} b_0 \right), \forall (i,j) \in H(S_k) \wedge \Delta f_k < 0$。因為 BRT-S 是演化式的演算法，為加強收斂的效果，在迄今最佳解的線路上的頻寬一律強制設成上限 $\bar{u} b_0$。因此，迄今最佳解的頻寬強化作業是 $b_{i,j} \leftarrow \bar{u} b_0, \forall (i,j) \in H(S^*)$。

進入下一個迭代前，須根據當次迭代中傳輸者的表現更新傳輸順序。傳輸者的傳輸順序是依目標函數值改善量遞減方式排序。令更新後的傳輸順序是 $O \leftarrow [o_1 o_2 \cdots o_m], o_k \in M, k = 1, 2, \cdots, m$；其中 $\Delta f_{o_k} \geq \Delta f_{o_{k+1}}, k = 1, 2, \cdots, m-1$。如前面所述，傳輸失敗者其目標函數值改善量為零。相反地，如果傳輸成功者的目標函數值改善量為負值，則他將被放置於傳輸順序的後段。此外，強制改變解建構失敗者的起始節點(或起始線段)，嘗試能有更好的表現。因此，

$\left(s_1^k, s_1'^k\right) \leftarrow \left(\text{Rand}(N), \text{Rand}(N)\right)$ 或 $s_1^k \leftarrow \text{Rand}(N)$ ， $\forall k \in M - \tilde{M}$ 。式中運算子 $\text{Rand}(N)$ 是由其集合運算元 N 中隨機選取一個元素，如 $\text{Rand}(A)$ ： $A \rightarrow a_i$ ，而 $A = \{a_1, a_2, \cdots\}$ ， $a_i \in A$ 。

BRT-S 整體演算流程

彙整上述的演化作業規則，在此說明 BRT-S 的整體演算流程。首先初始化 BRT-S 的演化系統，再開始求解演化。迭代中每個訊息傳輸者將依序進行訊息傳輸作業，再來評估成功建構解的傳輸者其解的品質。執行訊息傳輸作業後依解的優劣，對解線段進行頻寬配額擴充或縮減作業。同時模仿隨時間遞移執行傳輸纜線劣化作業。在迭代最後根據傳輸者求解的優劣變更在下一迭代的傳輸順序。下面展示 BRT-S 的演算流程。

```
Bandwidth_Restricted_Transmission-Simulated_Optimization_Algorithm()
1    Initialize_BRT-S_Optimization_Algorithm()
2    repeat
3         Messengers_Compete_For_Bandwidth_To_Send_Message()
4         Transmission_Evaluation_For_Optimum_Evolution();
5         if Termination_Condition_Met()
6              exit repeat
7         end if
8         Bandwidth_Deterioration_And_Modulation()
9         Reset_Transmission_Orders()
10   end repeat
```

上述流程中行 1 Initialize_BRT-S_Optimization_Algorithm()初始化傳輸網路系統，包含初始化頻寬配額、傳輸者使用頻寬量和設定每一個訊息傳輸者的起始節點(或起始線段)。行 3 到 9 模擬執行一個迭代的訊息傳輸作業。在行 3 Messengers_Compete_For_Bandwidth_To_Send_Message()中訊息傳輸者彼此競爭頻寬進行訊息傳輸。行 4 的 Transmission_Evaluation_For_Optimum_Evolution()評估傳輸成功解的品質並且視結果更新迄今最佳解。行 8 的 Bandwidth_Deterioration_And_Modulation()模擬頻寬的劣化並根據解品質好壞在

線段上添加或扣減頻寬。行 9 Reset_Transmission_Orders()根據目標函式值的改善量重新排定下一迭代訊息傳輸者的傳輸順序。

初始化 BRT-S 模擬系統

BRT-S 初始化時(1)設定每個傳輸者的起始節點(或起始線段)、(2)初始化傳輸者的傳輸順序O、和(3)設定傳輸者使用的頻寬配額量。以使用者設定的頻寬初始配額量b_0作為每條線段可供使用的頻寬配額。最後，設定演化停止條件的相關變數。運算過程的演化法如下。

Initialize_BRT-S_Optimization_Algorithm ()

1 **foreach** messenger k **in** M

2 $\left(s_1^k, s_1'^k\right) \leftarrow \left(\text{Rand}(N), \text{Rand}(N)\right)$

3 $f_k' \leftarrow null$

4 **end foreach**

5 $O \leftarrow \left[o_1\ o_2 \cdots o_m\right] = \left[1\,2\,3 \cdots m\right]$

6 $c_{k'} \leftarrow \left\lfloor \frac{(2-m)k' + m^2 - 2}{2m^2 - 2m} \cdot b_0 \right\rfloor ; k' = 1, 2, \cdots, m$

7 $b_{i,j} \leftarrow b_0, \forall (i,j) \in E$

8 $S^* \leftarrow null;\ f^* \leftarrow null$

9 $t^{(iter)} \leftarrow 0; t^{(obj)} \leftarrow 0; t^{(imp)} \leftarrow 0$

上述的演化程序中，行 2 和 3 隨機地設定每個傳輸者的起始線段，同時設定前一次傳輸者的目標函式值為空值。行 5 初始化傳輸順序序列依索引標號排序。行 6 依傳輸順序設定各順位傳輸者使用的頻寬量。行 7 初始化所有線段上的頻寬配額為頻寬初始配額量b_0。

訊息傳輸作業

訊息傳輸作業開始前，將線段的剩餘頻寬量$\beta_{i,j}$設定成目前頻寬配額$b_{i,j}$，供訊息傳輸者競爭使用。依照傳輸順序傳輸者依序進行傳輸作業。傳輸者先將起始

節點或起始線段加入其解的序列中並建立候選線段集合。當解尚未建構完成時，若候選集合中沒有任何線段可供挑選，則該傳輸者訊息傳輸失敗，由下一位進行傳輸。當候選集合內有多於兩個候選線段時，計算每個候選者的被挑選機率值，再產生 $\sim U(0,1)$ 的隨機亂數與確定模式使用率 λ 比較，當 $\sim U(0,1) > \lambda$ 執行隨機型模式，反之則執行確定型模式。挑選到的線段或節點加入解序列當中，再進行下一個建構步驟。當傳輸者完成一條完整的解序列後，將傳輸者加入成功傳輸者集合，並將解內各線段上的剩餘頻寬量扣除該傳輸者的頻寬使用量。接著計算傳輸成功者的目標函數值及改善量，並以本迭代的目標函數值取代前次的目標函數值。強制變更傳輸失敗者的起始節點或起始線段，並設定其目標函數值改善量為零。當所有傳輸者皆進行過傳輸作業後，結束訊息傳輸作業。演化細節如下所列。

Messengers_Compete_For_Bandwidth_To_Send_Message()

1 $t^{(iter)} \leftarrow t^{(iter)} + 1$
2 $\tilde{M} \leftarrow \{\ \}$
3 $\beta_{ij} \leftarrow b_{ij}, \forall (i,j) \in E$
4 $t^{(start)} \leftarrow$ Current_Time()
5 **for** $k' \leftarrow 1$ **to** m
6 $k \leftarrow o_{k'}$
7 $S_k \leftarrow \left[\left(s_1^k, s_1'^k \right) \right]$
8 $failed \leftarrow$ false
9 **for** $r \leftarrow 2$ **to** n
10 $J \leftarrow$ Construct_Candidate_Set$\left(S_k, k'\right)$
11 **if** $J = \{\ \}$
12 $failed \leftarrow$ true
13 **exit for**
14 **else**
15 $\left(j_{i^*}, j'_{i^*}\right) \leftarrow \begin{cases} \text{Select\_Link}\left(J, \text{stochastic}\right), & \text{if } \sim U(0,1) > \lambda \\ \text{Select\_Link}\left(J, \text{deterministic}\right), & \text{otherwise} \end{cases}$
16 $S_k \leftarrow S_k \oplus \left(j_{i^*}, j'_{i^*}\right)$
17 **end if**
18 **end for**
19 **if** $failed =$ true
20 $\Delta f_k \leftarrow 0$

21 $\left(s_1^k, s_1'^k\right) \leftarrow \left(Rand(N), Rand(N)\right)$
22 **else**
23 $\tilde{M} \leftarrow \tilde{M} \cup \{k\}$
24 $\beta_{s_i^k, s_i'^k} \leftarrow \beta_{s_i^k, s_i'^k} - c_{k'}, \forall i = 1,2,\cdots,n$
25 **end if**
26 **end for**

行 3 將所有線段上的剩餘頻寬量設定為現有頻寬配額。行 6 到 25 是每個傳輸者執行的訊息傳輸作業。其中行 8 的 *failed* 旗標是用來登錄該傳輸者的傳輸是否失敗，初始時設定否，即 *failed* ← false 。行 10 到 17 是訊息傳輸者執行一次的解建構作業，首先必須建立候選線段集合。當候選集合為空集合時，傳輸失敗令 *failed* 為真並中斷解的建構。不然執行 Select_Link()作業，由候選集合中挑選一個線段加入解序列當中。行 19 到 25 判斷該訊息傳輸者的成敗。當 *failed* 為真時傳輸失敗，隨機重設該訊息傳輸者的起始線段，並設定當代目標函數值改善量為零。當 *failed* 為假時訊息傳輸成功，將該訊息傳輸者加入成功者集合內，並檢出解所挑選的線段逐一扣減其剩餘頻寬量。

建立候選集合

每步驟的解建構作業首先建立候選線段集合。根據目前已建構的解序列參酌問題限制條件及頻寬用量，列出可供挑選的線段。首先，須對問題的限制條件篩選目前不違反條件的線段組成集合 $\hat{J}$ 。再來，檢視 $\hat{J}$ 中各線段的剩餘頻寬量是否足夠傳輸者使用，若可以才將此線段加入候選集合 J 。完整作業程序如下所示。

Construct_Candidate_Set$\left(S_k, k'\right)$
1 $J \leftarrow \{\ \}$
2 $\hat{J} \equiv \{(j_i, j_i')\} \leftarrow$ Current_Constraint_Satisfied_Links$\left(S_k\right)$
3 **if** $\hat{J} = \{\ \}$
4 **return** J
5 **end if**
6 **foreach** (i, j) **in** $\hat{J}$
7 **if** $\beta_{i,j} \geq c_{k'}$

8 $J \leftarrow J \cup \{(i, j)\}$
9 **end if**
10 **end foreach**
11 **return** J

行 2 執行的 Current_Constraint_Satisfied_Links(S_k)運算式是根據目前已建構的解序列S_k，挑選出不會違反問題限制條件且尚未使用的線段。行 7 逐一檢視$\hat{J}$中的線段，其剩餘頻寬量是否足夠第k'順位傳輸者使用。如果剩餘頻寬量足夠才將此線段加入候選集合J中。

線段挑選作業

當候選集合J中有多餘一個候選者時須進行線段挑選作業。候選者被挑選的機率是剩餘頻寬及啟發式評估值的乘積。如果是採取確定型模式，則回傳機率值最大者。隨機型挑選模式則產生一介於 0 到 1 之間的隨機亂數，依照候選者被標準化後的機率值進行挑選並回傳。演化程序細節如下所列。

Select_Link$\left(J = \left\{(j_i, j_i') \middle| i = 1,2,\cdots,|J|; j_i, j'_i \in N\right\}, \text{selection_mode}\right)$

1 **if** $|J| = 1$
2 **return** (j_1, j_1')
3 **end if**
4 $p_i \leftarrow \beta_{j_i, j_i'} \cdot h_{j_i, j_i'}; \forall i = 1,2,\cdots,|J|$
5 **if** selection_mode = deterministic
6 $i^* \leftarrow \arg \max_{\forall i=1,2,\cdots,|J|} \{p_i\}$
7 **else**
8 $p_i' \leftarrow \dfrac{p_i}{\sum_{\forall i=1,2,\cdots|J|} p_i}; \forall i = 1,2,\cdots,|J|$
9 $temp \leftarrow \text{Rand}(0,1)$
10 $i^* \leftarrow |J|$
11 **for** $i \leftarrow 1$ **to** $|J|$
12 **if** $temp < p_i'$
13 $i^* \leftarrow i$
14 **break for**

15 **else**
16 $temp \leftarrow temp - p_i'$
17 **end if**
18 **end for**
19 **end if**
20 **return** $\left(j_{i^*}, j_{i^*}'\right)$

行 1 設定候選集合 J 中各線段被挑選的機率值為線段上的剩餘頻寬和啟發式評估值的乘積。行 2 判斷隨機採取的挑選模式，如果是確定型挑選模式則執行行 3 回傳機率最大者，否則為隨機型挑選模式則執行行 5 到 15 的作業。隨機型模式先將機率值標準化，並由 Rand(0,1) 產生一個介於零到一的亂數值 $temp \leftarrow U(0,1)$。判斷 $temp$ 落於哪一段候選機率區間中並回傳被挑中者。

解評估並演化迄今最佳解

當傳輸作業結束後，評估成功傳輸者的解目標函數值。根據問題所設定的目標函數計算解的品質與改善量。改善量最大者與迄今最佳解進行比較，優於迄今最佳解則更新迄今最佳解。下面是完整演化程序。

Transmission_Evaluation_For_Optimum_Evolution ()
1 **foreach** messenger k **in** $\tilde{M}$
2 $f_k \leftarrow f(S_k)$
3 $t^{(obj)} \leftarrow t^{(obj)} + 1$
4 **if** $f_k' = \text{null}$
5 $f_k' \leftarrow f_k$
6 $\Delta f_k \leftarrow 0$
7 **else**
8 $\Delta f_k \leftarrow f_k' - f_k$
9 $f_k' \leftarrow f_k$
10 **end if**
11 **end foreach**
12 $\tilde{k} \leftarrow \arg\max_{\forall k \in \tilde{M}} \{\Delta f_k\}$
13 **if** $f^* \neq \text{null} \wedge f_{\tilde{k}}' \geq f^*$
14 $t^{(imp)} \leftarrow t^{(imp)} + 1$
15 **else**

16 $f^* \leftarrow f_{\tilde{k}}$
17 $S^* \leftarrow f'_{\tilde{k}}$
18 $t^{(imp)} \leftarrow 0$
19 **end if**

行 2 到 10 計算傳輸成功者 k 的目標函數值，同時添加目標函數評估計次一次。接著，計算目標函數值改善量和更新前次成功求解的目標函數值。行 12 到 19 判斷是否找到比迄今最佳解更好的解，若有則更新迄今最佳解並將最佳解無改善次數歸零；沒有則添加最佳解連續無改善累計次數一次。

停止條件判斷

判斷演算過程中各停止條件是否滿足。滿足時停止 BRT-S 演化，並將迄今最佳解回傳。尚未滿足停止條件則繼續演化。演化程序如下所列。

Termination_Condition_Met()
1 $t^{(cpu)} \leftarrow \text{Current\_Time}() - t^{(start)}$
2 **if** $\left(t^{(iter)} \geq T^{(iter)}\right) \vee \left(t^{(obj)} \geq T^{(obj)}\right) \vee \left(t^{(imp)} \geq T^{(imp)}\right) \vee \left(t^{(cpu)} \geq T^{(cpu)}\right)$
3 **return** true
4 **else**
5 **return** false
6 **end if**

此運算式判斷是否達到停止條件，分別判斷迭代次數、目標函式值評估次數、最佳解連續無改善累計次數和已執行模擬時間是否達到使用者所設定的標準，達到則傳回 true；否則回傳 false。

頻寬劣化和人工調整作業

本作業先模擬纜線因環境的影響產生的自然劣化。依使用者設定的自然劣化率 δ，挑選出 $\delta \cdot |E|$ 個線段扣除一固定的頻寬配額。再依照訊息傳輸成功者建構的解品質優劣，擴充或縮減其解線段上的頻寬配額。此作業結果，須維持線段上

的頻寬可供使用且不能無限擴充。因此擴充或縮減會以使用者設定的上限值 $\bar{\mu}b_0$ 及下限值 $\underline{\mu}b_0$ 為界限。頻寬劣化和人工調整作業流程細節如下所示。

Bandwidth_Deterioration_And_Modulation ()

1 **for** $i \leftarrow 1$ **to** $\delta \cdot |E|$
2 $(i, j) \leftarrow \text{Rand}(E)$
3 $b_{i,j} \leftarrow \max\left(b_{i,j} - \rho b_0, \underline{u}b_0\right)$
4 **end for**
5 **foreach** messenger $k \leftarrow 1$ **to** m
6 **if** $\Delta f_k > 0$
7 $b_{i,j} \leftarrow \min\left(b_{i,j} + \rho b_0, \bar{u}b_0\right), \forall (i, j) \in H(S_k)$
8 **else if** $\Delta f_k < 0$
9 $b_{i,j} \leftarrow \max\left(b_{i,j} - \rho b_0, \underline{u}b_0\right), \forall (i, j) \in H(S_k)$
10 **end if**
11 **end foreach**
12 $b_{i,j} \leftarrow \bar{\mu}b_0, \forall (i, j) \in H(S^*)$

行 1 到 4 執行頻寬劣化作業，任意挑出 $\delta \cdot |E|$ 個線段進行劣化。執行劣化的線段其頻寬配額扣除 ρb_0 後不得小於設定的下限 $\underline{\mu}b_0$。行 5 到 11 執行頻寬擴充及縮減作業。若目標函數值改善量為正值，在所挑選的線段上添加頻寬量。相對地，若目標函數值改善量為負值，在所挑選的線段上扣減頻寬量。行 12 則是在迄今最佳解的線段上，將其頻寬配額量強制設成頻寬配額上限量 $\bar{\mu}b_0$。

調整傳輸順序

在迭代最後，根據本迭代訊息傳輸者建構的解品質改善量大小，調整傳輸者在下一迭代中的傳輸順序。讓表現較好的傳輸者能優先進行傳輸且使用較多的頻寬。注意此處的表現非指目標函數值優劣，而是與上次求得的目標函數值間的改善量。下列展示的是使用冒泡(bubble sort)排序法的傳輸順序調整演算程序。程式實作可選用較具效率的 Quick Sort 排序法。

```
Reset_Transmission_Orders()
1    O ← [o_1 o_2 ⋯ o_m] = [1 2 3 ⋯ m]
2    for i ← 1 to m − 1
3        for k ← 1 to m − 1 − i
4            if Δf_{o_k} < Δf_{o_{k+1}}
5                temp ← o_k
6                o_k ← o_{k+1}
7                o_{k+1} ← temp
8            end if
9        end for
10   end for
```

行 2 到 10 進行傳輸順序的排序，根據訊息傳輸者的目標函數值改善量進行順序的調整，改善量越大者放置在傳輸順序的前端，在下一迭代中能享有較豐富的頻寬資源；反之，改善量越小者放置在傳輸順位的後端。

上述為整體 BRT-S 求解的演化流程。BRT-S 在求解不同優化問題時，須依照問題特性調整部分演化程序細節。下面兩節說明 BRT-S 應用於求解物件排序優化問題和物件分群優化問題，分別以旅行銷售員問題和裝箱問題為例。根據問題特性定義 BRT-S 的資料結構，和說明建立候選集合的演算程序。

3.4 求解物件排序優化問題的 BRT-S 模式

旅行銷售員問題(Traveling Salesman Problem, TSP)是典型的物件排序優化問題，現實中有許多優化問題都由 TSP 衍生而成。TSP 要優化的是銷售員拜訪城市行經的路線最短化。由起始城市出發逐步拜訪所有城市，每一個城市僅能通過一次，最後回到起始城市形成一條迴圈。令 N 是旅行銷售員問題中所要拜訪的城市集合，$N=\{1,2,...,\bar{n}\}$，其中 $\bar{n}$ 是城市總數；E 是連結兩城市之間的線段集合，$E=\{(i,j)|i,j\in N\}$，其中線段 (i,j) 是城市 i 與城市 j 間的連結。TSP 的解是一個物件序列 $S=[s_1 s_2 \cdots s_{\bar{n}}], s_i \in N$，其中 s_i 是第 i 順位拜訪的城市。TSP 求解的目標函數是最小化行經路線的長度，

$$f^{TSP}(S)=\left(\sum_{i=1}^{\bar{n}-1} d_{s_i,s_{i+1}}\right)+d_{s_{\bar{n}},s_1}。 \tag{3.2}$$

式中，d_{ij} 是城市 i 和 j 間的距離。由於 TSP 的求解目標是找到一條通過所有城市最短的迴圈，所以在建構過程中傾向挑選較短的線段，以期望建構出最短的迴圈。因此，將線段 (i,j) 的啟發式評估值 h_{ij} 設計成距離的倒數，即 $h_{ij}=1/d_{ij}$。由於解建構過程是由其中一個城市出發，逐步挑選下一個城市逐一拜訪。針對 TSP，BRT-S 中的頻寬配額直接佈設在任兩城市間的連結線段；因此 b_{ij} 是城市 i 和 j 間線段上的頻寬配額量。假設傳輸者 k 當下建構的解釋 $S^k=[s_1^k s_2^k \cdots]$ 且候選城市集合為 $J=\{j_1', j_2', \cdots\}, j_i' \in N$。假設目前完成 j 個城市選取，令上次選取的城市是 $i^*=s_j^k$。在 TSP 中，J 內的候選者 i 的候選機率設為 $p_i=\beta_{i^*,j_i'}\cdot h_{i^*,j_i'}$。

演化過程中若線段上的剩餘頻寬量越多及線段的距離越短，則被挑選的機率越大。當訊息傳輸作業結束後，以式 3.2 評估成功傳輸者建構解的目標函數值。

在人工頻寬擴充或縮減作業中，依照訊息傳輸者 k 的目標函數值改善量 Δf_k 的正負值，進行擴充或縮減組成其解的所有線段頻寬。當 $\Delta f_k > 0$ 時進行擴充頻寬。經拆解運算取得組成 TSP 解 S_k 的線段集合，$H^{TSP}(S_k) = \{(s_i, s_{i+1}) | i = 1, 2, \cdots, \bar{n} - 1\} \cup \{(s_{\bar{n}}, s_1)\}$，在不超過頻寬上限的情況下，擴充線段上頻寬 ρb_0 的量。反之，當 $\Delta f_k < 0$ 時在不低於頻寬下限的情況下，縮減線段上頻寬 ρb_0 的量。例如：若訊息傳輸者 k 建構的解序列 $S_k = [1\ 3\ 2\ 5\ 4]$，透過拆解運算式可得到線段集合 $H^{TSP}(S_k) = \{(1,3),(3,2),(2,5),(5,4),(4,1)\}$。若該傳輸者解的目標函數值改善量 $\Delta f_k > 0$，則在線段 $(1,3)$、$(3,2)$、$(2,5)$、$(5,4)$ 和 $(4,1)$ 上擴充頻寬配額，即添加 ρb_0 的頻寬量給 $b_{1,3}$、$b_{3,2}$、$b_{2,5}$、$b_{5,4}$ 和 $b_{4,1}$。擴充後的頻寬不會超過上限 $\bar{\mu} b_0$。

BRT-S 求解 TSP 的主要演算流程如 3.3 節所述，以下說明求解 TSP 使用的 Current_Constraint_Satisfied_Links(S_k) 演算程序 。作業中將根據目前建構中的解序列 S_k，篩選出尚未規劃入解序列的城市組成集合 $\hat{J}$。演算程序細節如下所示。

Current_Constraint_Satisfied_Links (S_k)

1 $\hat{J} \leftarrow N$
2 **foreach** city i **in** S_k
3 $\hat{J} \leftarrow \hat{J} - \{i\}$
4 **end foreach**
5 **return** $\hat{J}$

行 1 複製城市集合 N 成不違反條件的城市集合 $\hat{J}$。行 2 至 4 移除 $\hat{J}$ 中已被解序列 S_k 中選取的城市。行 5 回傳尚未選取的城市集合。

3.5 求解物件分群優化問題的 BRT-S 模式

裝箱問題(Bin Packing Problem, BPP)屬於物件分群優化問題。一維裝箱問題內有一具容量屬性值的物品集合Y；設$Y=\{1,2,\cdots,\bar{y}\}$且$\bar{y}$是物品總數。令w_i是物品i的容量屬性值，$w_i>0,\ \forall i\in Y$。BPP 的工作是將集合中的物品放置於具有容量限制值L的箱子當中。一維裝箱問題求解的目標是最小化使用的箱子數量，或者是在給定箱數下，平衡箱子裝置的物品總容量。本研究探討的是已知箱數的裝箱問題，屬於標準的分群優化問題。假設箱子集合X，$X=\{1,2,\cdots,\bar{x}\}$，其中$\bar{x}$是箱子總數。

BPP 中的物件分為物品群組與箱子群組，且建構過程是將物品一個一個放置在箱子當中，所以訊息傳輸網路的連結關係設成物品與箱子間的「置入關係」。並且各傳輸者登錄的起始物件是起始進行配置的物品，並非物品與箱子間的置入關係。令$b_{i,j}$是物品i與箱子j間線段上的頻寬配額量。選擇不同物品進行配置，會改變箱子的剩餘容量，進而影響後續配置的物品在選擇置入箱子時挑選的可能性。所以物品配置的先後順序，會直接影響最後裝箱的結果。因此，每位傳輸者進行建構解前，須先設定物品裝箱作業執行順序。假設物品執行裝箱序列是$Q=\left[q_1q_2\cdots q_{\bar{y}}\right]$，$q_\alpha\in Y$，其中$q_1$設成傳輸者登錄的起始物品，其餘物品則以亂數方式隨機配置於序列Q中。假設當下是第k'個順位的傳輸者$o_{k'}$，執行第α順位物品的配置，且設$i'=q_\alpha$及$k=o_{k'}$則當下的候選線段集合是

$J=\left\{(i',j_z)\middle|\beta_{i',j_z}\geq c_{k'}\wedge L_{j_z}<L,\ j_z\in X, z=1,2,\cdots,|J|\right\}$。式中$c_{k'}$是目前第$k'$順位傳輸者的頻寬使用量。除了候選線段上的剩餘頻寬量不得小於訊息傳輸者的傳輸需求量外，箱子內物品的總容量也不得超出箱子容量限制。注意此處，本研究僅考慮箱子j的容量未超過上限，並未考量其剩餘容量$L-L_j$是否足以納入當下物品i'的容量$w_{i'}$。原因在於演化過程中設定的箱數是已知最小箱數，解建構過程很容易發生裝箱超出容量上限的情形，需要時間來演化出無違反箱子容量限制的解。因此在演化過程中，允許傳輸者建構出違反箱子容量限制的解，透過演化機制使最佳解達到無違反箱子容量限制。

不同於 TSP 中啟發式評估值在演化過程中是固定的，BRT-S 求解 BPP 使用的啟發式評估值$h_{i,j}$是一動態值且僅與箱子j及目前建構中的解有關。設h'_j是 BPP 中箱子j的啟發式評估值，且$h'_j=L-L_j$。此啟發式評估值會隨解的建構過程改變。因此，J中候選線段(i',j_z)被挑選的機率值是$p_z=\beta_{i',j_z}h'_{j_z}=\beta_{i',j_z}\cdot\left(L-L_{j_z}\right)$，當線段$(i',j_z)$上剩餘頻寬越多和箱子$j$的剩餘容量越大，則此線段被挑選的機率越大。

傳輸者配置完物品即完成一個解。令$\boldsymbol{S}=\left[(q_1,\xi_1),(q_2,\xi_2),\cdots,\left(q_{\bar{y}},\xi_{\bar{y}}\right)\right]$是建構出的解序列，線段$(q_i,\xi_i)$是物品$q_i$和箱子$\xi_i$的置入關係。且$H^{BPP}(\boldsymbol{S})=\left\{(q_1,\xi_1),(q_2,\xi_2),\cdots,\left(q_{\bar{y}},\xi_{\bar{y}}\right)\right\}$是解序列$\boldsymbol{S}$的線段集合。設$L_j$是箱子$j$在裝箱結果$\boldsymbol{S}$下的總容量值，則$L_j=\sum_{\forall(q_i,\xi_i)\in H^{BPP}(\boldsymbol{S})\wedge\xi_i=j}w_{q_i}$。令$\bar{L}$是所有$\bar{x}$個箱子裝箱

結果的總容量平均值，$\bar{L}=\frac{1}{\bar{x}}\sum_{j=1}^{\bar{x}}L_j$。箱數已知的 BPP 求解目標是最小化所有 $\bar{x}$ 個箱子的總容量標準差 σ，$\sigma=\sqrt{\frac{\sum_{j=1}^{\bar{x}}\left(L_j-\bar{L}\right)^2}{\bar{x}-1}}$。因此，令 BPP 第一種望小的目標函數是

$$f^1(\boldsymbol{S})=\sigma=\sqrt{\frac{\sum_{j=1}^{\bar{x}}\left(L_j-\bar{L}\right)^2}{\bar{x}-1}}\quad。\tag{3.3}$$

為能驅策解的演化效率，本研究另在標準差上加入違反箱子容量限制的箱數因子成為 BPP 的第二種目標函數，

$$f^2(\boldsymbol{S})=\left(\frac{e}{\bar{x}}+1\right)\sigma=\left(\frac{e}{\bar{x}}+1\right)\sqrt{\frac{\sum_{j=1}^{\bar{x}}\left(L_j-\bar{L}\right)^2}{\bar{x}-1}}\quad,\tag{3.4}$$

式中 e 是違反容量限制的箱數，即 $e=\underset{\forall j\in X}{\text{Count}}\left(L_j>L\right)$。當 e 等於零時，由式 3.3 及 3.4 可得知 $f^1(\boldsymbol{S})=f^2(\boldsymbol{S})$。

BRT-S 求解 BPP 的主要演算流程如 3.3 節所述，以下將說明 BRT-S 求解 BPP 所使用的 Current_Constraint_Satisfied_Links(S_k,α) 演算程序。作業中將根據目前建構中解序列 $\boldsymbol{S}_k$ 及執行第 α 順位的物品 q_α，篩選出無違反容量限制的箱子組成的置入關係的線段集合 $\hat{J}$。演算程序細節如下所示。

Current_Constraint_Satisfied_Links (S_k,α)

1 $\hat{J}\leftarrow\{\ \}$

2 $L_j=\sum_{\forall(q_i,\xi_i)\in H^{BPP}(\boldsymbol{S}_k)\wedge\xi_i=j} w_{q_i},\forall j\in X$

3 $h'_j \leftarrow L - L_j,\ j \in X$
4 **foreach** bin j **in** X
5 $\hat{J} \leftarrow \hat{J} \cup \{(q_\alpha, j)\}$, if $h'_j > 0$
6 **end foreach**
7 **return** $\hat{J}$

行 2 計算各箱子內已裝配物品後的總容量值。行 3 設定箱子 j 的啟發式評估值是箱子容量限制和箱子 j 內總容量的差值。行 4 至 6 篩選箱子 j 啟發式評估值大於 0 者加入無違反限制條件的線段組成集合 $\hat{J}$ 中並在行 7 時回傳。

3.6 小結

本章介紹了 BRT-S 的演化標準流程用以求解離散優化問題，並說明各演算作業細節。求解物件排序及物件分群優化問題時，須分別規劃建立無違反限制條件的線段候選集合作業的細節，以符合問題的限制條件。下一章會展示本研究開發出來的 BRT-S 求解系統。接著以 BRT-S 求解若干標竿問題，並和其他啟發式演算法比較求解結果，以驗證 BRT-S 求解離散優化問題的效能。

第四章　數值範例測試與結果分析

本章將展示仿頻寬限制資料傳輸優化演算法(BRT-S)求解物件排序及分群優化標竿問題的求解系統並驗證其效能。本研究測試求解 TSPLIB 和 BPPLIB 內的旅行銷售員問題(TSP)和一維裝箱問題(BPP)標竿問題。用不同類型的優化問題來驗證 BRT-S 的求解品質及效率。本章分節展示 BRT-S 兩種求解問題的範例測試內容，並討論及比較相關設定對求解結果影響。

4.1 仿頻寬限制資料傳輸優化演算法求解系統軟體

本研究依照上一章所提出的仿頻寬限制資料傳輸優化演算法，在.Net Framework 環境下使用 C#程式語言開發一套仿頻寬限制資料傳輸優化演算法的離散優化問題求解系統(Bandwidth Restricted Transmission-Simulated Optimization Algorithm based Discrete Optimization System, 簡稱 BRTSDOS)。整套軟體系統使用開發環境提供的各式物件與元件，可在跨平台.Net Framework 環境下執行。BRTSDOS 已納入物件排序優化問題及物件分群優化問題的求解模式，使用者可分別選定此兩類問題透過本系統進行求解。

圖 4-1 是本求解系統的人機介面。供使用者選擇及設定求解的問題。圖中左方是 BRT-S 演算法的初始設定區域，包含問題檔案讀入、相關參數設定、和範例問題資訊。右方是求解過程展示區域，上面是迄今最佳解的圖形展示，下面是求解過程中迄今最佳目標函數值的演化及收斂過程，視覺效果展示供使用者檢視目前求解的品質和收斂程度。圖 4-2 是 TSP 求解過程和結果展示的示意。圖中左上方是停止條件設定區域，下方是求解結果及求解最終的相關資訊。圖中右邊展示求解的 TSP 標竿問題的已知全域最佳解和求得的迄今最佳路徑。下面是最佳目標函數值、迭代平均目標函數值、和各迭代成功傳輸者數量的演化及收斂過

程。圖 4-3 是 BPP 求解結果展示。圖中是求解 BPP 的迄今最佳解裝箱結果各箱內容的展示。

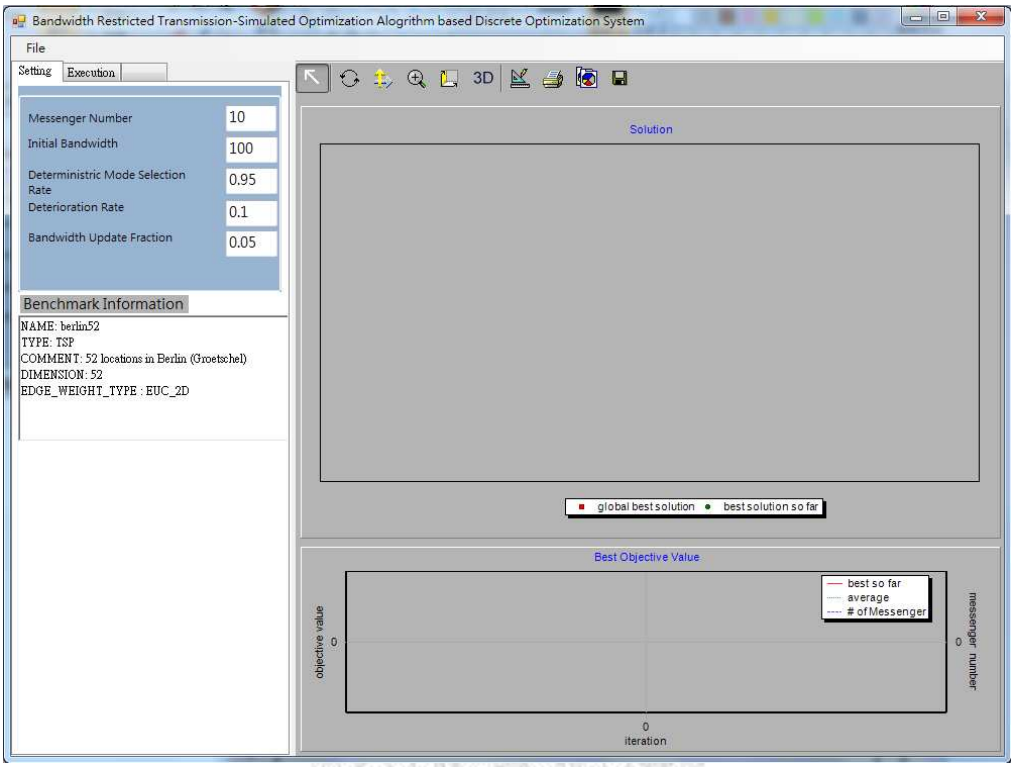

圖 4.1 BRTSDOS 初始設定介面

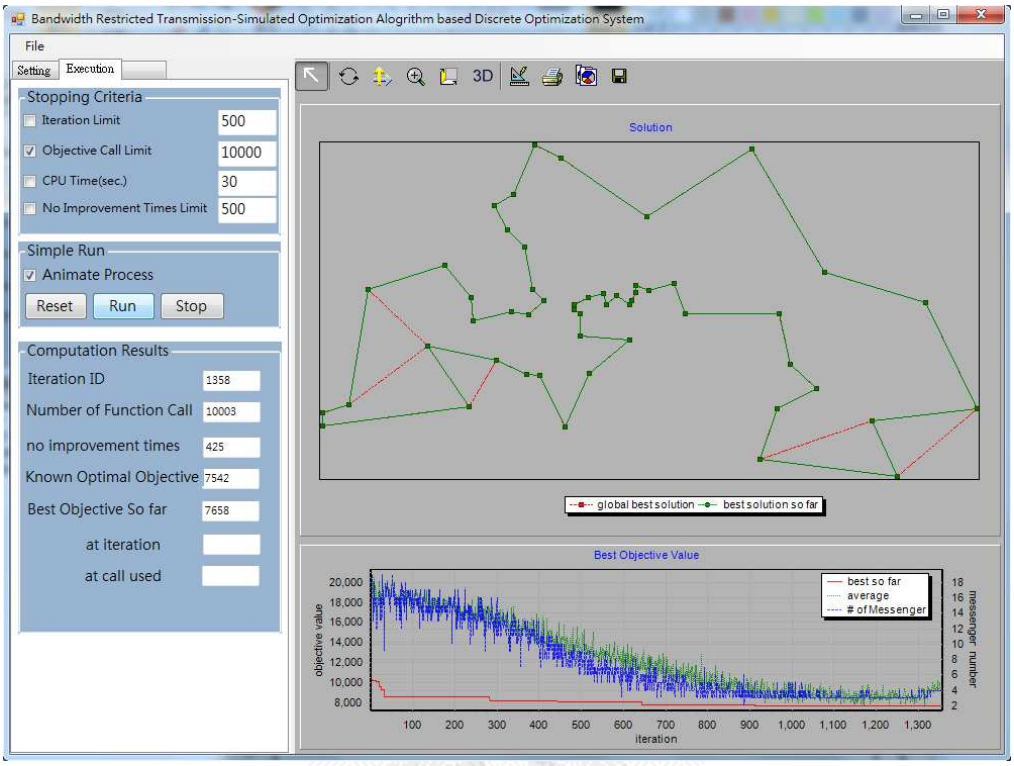

圖 4.2 TSP 求解結果介面展示

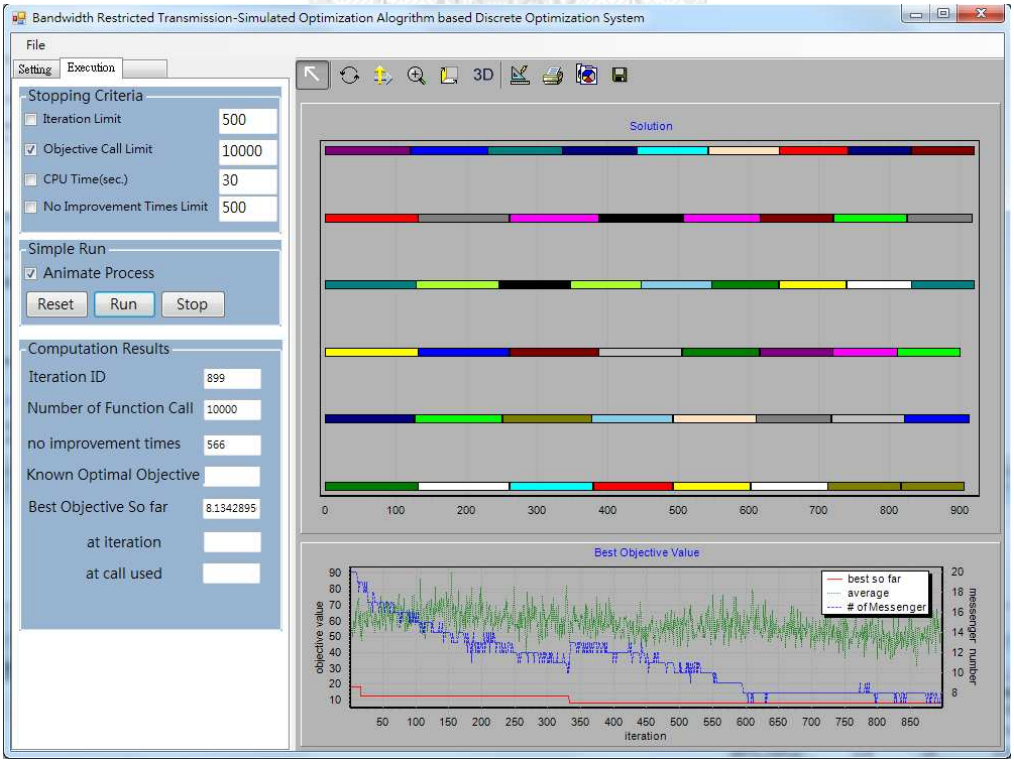

圖 4.3 BPP 求解結果介面展示

4.2 BRT-S 求解旅行銷售員問題測試

BRT-S 求解旅行銷售員問題時,針對不同規模的標竿問題須進行參數調校以獲得較佳的求解品質。以下將展示三組實驗測試的結果，同時說明參數組合的設定內容。

第一組實驗是以 BRT-S 和六種蟻拓最佳化技術在充足演化停止條件下比較求解品質。實驗中進行比較的六種蟻拓最佳化技術，其求解品質與呼叫目標函數次數是參考(Ugur and Aydim, 2009)內的數據。第二組實驗是比較 BRT-S 和其他啟發式演算法在演化停止條件較嚴苛的情況下，求解四個標竿問題的品質與效率。實驗中比較的啟發式演算法是由 Shuang、Chen、和 Li (2009)提出的 PS-ACO 法以及比較它們的對象用的 GA、ACO、和 MMAS 法。第三組實驗是測試 BRT-S 求解大型 TSP 標竿問題的品質，結果則和六種蟻拓最佳化技術比較。實驗中六種蟻拓最佳化技術的求解結果是參考自(Yang and Chou, 2009)內的數據。

第一組實驗比較的對象是由 Ugur 和 Aydim (2009)建構的 TSPAntSim 線上模擬系統。該系統提供使用者在網路上以蟻拓優化法求解旅行銷售員問題。系統中實作有六種蟻拓法,分別為 Ant System (AS)、Ant Colony System (ACS)、Elitist Ant System (ASE)、Max-Min Ant System (MMAS)、Rank-Based Ant System (RBAS)、和 Best-Worst Ant System (BWAS)。實驗由本研究所提的 BRT-S 法和上述六種蟻拓法求解 TSPLIB 內的 eil51、berlin52、st70、eil76、和 rat99 五個標竿問題，並比較求解品質。表 4-1 列出 BRT-S 法求解五個標竿問題時的參數設定。因演化停止條件較為充裕，傳輸者數量設成問題城市的數量，讓傳輸者能發揮團隊合作的能力。頻寬初始配額量 b_0 建議設定成傳輸者數量的四倍到五倍，此實驗中設為五倍。確定型模式使用率 λ 設定可依城市數大小設定，數量越大時建議採取較高的使用率，避免過度隨機搜索。而在城市數量較小的問題中，建議保留較多的隨機挑選率，讓解代理人有機會跳脫區域最佳解。因此，針對城市數量較少的第一

組實驗，我們將確定型挑選模式使用率 λ 設定為 0.9。在城市數量較小的問題中，建議自然劣化率設定在 0.2 到 0.01 間，在此問題中設為 0.1。頻寬配額增減比例設成 0.01，每次增減量維持在 0.01 倍的頻寬初始配額量，即 $\rho = 0.01$。求解結果展示於表 4.2。

表 4.1 BRT-S 在實驗範例一中五個標竿問題參數設定內容

參數設定	**標竿問題**				
	eil51	**berlin52**	**st70**	**eil76**	**rat99**
訊息傳輸者數量 m	51	52	70	76	99
頻寬初始配額量 b_0	255	260	350	380	495
確定型挑選模式使用率 λ	0.9	0.9	0.9	0.9	0.9
自然劣化率 δ	0.1	0.1	0.1	0.1	0.1
頻寬配額增減比例 ρ	0.01	0.01	0.01	0.01	0.01

BRT-S 求解這些問題時設定的停止條件是呼叫目標函數達 200000 次。每一個標竿問題實驗五次求算最佳解路線長平均值。表 4-2 中第一欄是標竿問題的名稱，名稱內含有城市數量；第二欄是標竿問題已知全域最佳解的路線長；第三到八欄分別是六種蟻拓法分別求解五次的最佳解路線長平均值。第九欄則是 BRT-S 求得的最佳解路線長平均，括弧中是求得最佳解的目標函數呼叫次數平均值。

表 4.2 BRT-S 和六種蟻拓法求解五次平均路線長比較

標竿問題	最佳解路線長	求解法						
		AS	**ACS**	**ASE**	**MMAS**	**RBAS**	**BWAS**	**BRT-S**
eil51	426	437.3	428.1	428.3	427.6	428.1	427.9	427.2 (50630.4)
berlin52	7542	7554.6	7542	7542	7542	7542	7542	7542 (17436.6)
st70	675	702.1	678.9	696.4	676.6	686.4	676.2	676.6 (126306)
eil76	538	548.6	542.3	545.9	538	545.5	538	538.6 (67415.4)
rat99	1211	1255.8	1215.4	1251.8	1211.2	1221.3	1213.8	1212.8 (138684)

表 4-2 顯示 BRT-S 求解 eil51 標竿問題較六種蟻拓最佳化法求得的解品質好。berlin52 標竿問題除了 AS 無法求得最佳解，本法及其他方法在五次實驗中皆可求得最佳解。而在 st70、eil76、和 rat99 這三個標竿問題求解表現略遜 MMAS 和 BWAS 兩種方法。將求解結果的目標函數值相對誤差繪製成圖 4-4，相對誤差計算方式是 $\frac{\text{求解路線長}-\text{已知全域最佳解路線長}}{\text{已知全域最佳解路線長}}\times 100\%$。本 BRT-S 法和 MMAS 和 BWAS 法在求解路線長的相對誤差皆小於 0.5%，彼此間的差距相當小。由於文獻中沒有提供這六種蟻拓最佳化技術平均求得最佳解時的目標函數呼叫次數，本研究在實驗中會記錄 BRT-S 求得最佳解時已呼叫目標函數的次數，由表 4-2 第九欄可看出 BRT-S 求解所花費的呼叫目標函數次數遠低於設定的停止條件。大致而言 BRT-S 的競合演化可在呼叫目標函數次數較少的情況下求得品質不錯的解。

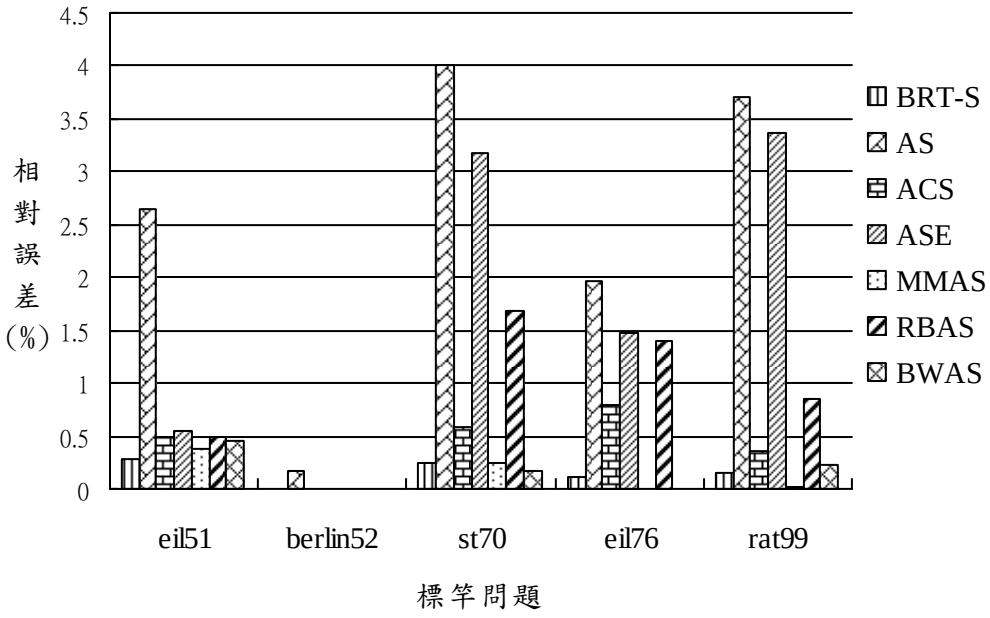

圖 4.4 BRT-S 和六種蟻拓最佳化技術求解路線長相對誤差比較

在第二組實驗中在呼叫目標函數次數較少的停止條件下，以 BRT-S 法與 Shuang 等人(2009)提的 PS-ACO 法及其比對的 GA、ACO、和 MMAS 法進行求解比較。此組實驗主要是測試 BRT-S 法與其他演算法在停止條件較嚴苛下，是否能求得品質較佳的解。本實驗以 TSPLIB 內的 att48、eil51、pr76、及 rd100 四個標竿問題為求解對象。文獻中設定解代理人數量等於標竿問題的城市數量且模擬停止條件設為演化代次上限達 2000 次。由於 BRT-S 在迭代演化中並非所有解代理人都進行解評估，為求比較公平性將模擬停止條件設定為呼叫目標函數次數達問題城市數量乘代次上限數來進行比較。BRT-S 求解這四個標竿問題的參數設定內容見表 4-3。對於小規模 TSP 範例，訊息傳輸者數量與頻寬初始配額量的設定方式同第一組實驗。由於演化停止條件較為嚴格，將確定型挑選模式使用率提高，讓解代理人更傾向逕行選取剩餘頻寬越多和距離越近的線段。自然劣化率與頻寬配額增減比例同第一組實驗設定。

表 4.3 BRT-S 在實驗範例二中四個標竿問題參數設定內容

參數設定	標竿問題			
	att48	**eil51**	**pr76**	**rd100**
訊息傳輸者數量 m	48	51	76	100
頻寬初始配額量 b_0	240	255	380	500
確定型挑選模式使用率 λ	0.99	0.95	0.9	0.92
自然劣化率 δ	0.1	0.1	0.1	0.1
頻寬配額增減比例 ρ	0.01	0.01	0.01	0.01

表 4-4 列出測試結果比較，其中第一欄是問題題目，括號內是已知全域最佳解的路線長，方括號內是呼叫目標函數次數上限條件。第二欄是演算法名稱，第三、四和六欄分別是實驗二十次平均最佳解路線長、相對誤差和求得最佳解時呼叫目標函數次數平均值，第五欄則為實驗二十次中最佳的路線長。表中結果顯示 BRT-S 法在求解 eil51、pr76 和 rd100 三個標竿問題的品質優於其他四個演算法。求解 att48 標竿時，BRT-S 與 PS-ACO 皆能找到問題的最佳解，但求解品質 BRT-S 略遜 PS-ACO。求解結果的相對誤差比較如圖 4-2，但在平均呼叫目標函數次數比較下 BRT-S 比 PSACO 較為精簡。第二組實驗顯示 BRT-S 能在呼叫目標函數次數較嚴苛的情況下，有不錯的求解品質與效率。

表 4.4 BRT-S 和四種演算法求解路線長及呼叫目標函數次數比較

標竿問題 (最佳解路線長) [呼叫目標函數次數]	演算法	路線長 平均值	平均相對誤差 (%)	最佳解路線長	平均呼叫目 標函數次數
att48	GA	35172.40	4.92	34099	-
(33522)	ACO	35066.25	4.61	34123	31051.20
[96000]	MMAS	33739.76	0.65	33524	66015.36
	PS-ACO	33560.70	0.12	33522	57007.20
	BRT-S	33571.70	0.15	33522	53934.70
eil51	GA	442.20	3.86	433	-
(426)	ACO	441.45	3.62	436	32226.90
[102000]	MMAS	433.15	1.67	430	27389.55
	PS-ACO	427.40	0.32	426	59022.30
	BRT-S	427.25	0.29	426	53706.55
pr76	GA	114678.13	6.03	112380	-
(108159)	ACO	117390.55	8.54	114083	75323.60
[152000]	MMAS	113055.50	4.53	112372	142366.24
	PS-ACO	110162.10	1.85	109744	118841.20
	BRT-S	109986.20	1.69	108274	116664.60
rd100	GA	8251.76	4.32	8138	-
(7910)	ACO	8410.40	6.33	8255	119940.00
[200000]	MMAS	8183.2	3.45	8043	190452.00
	PS-ACO	8017.40	1.36	7967	140840.00
	BRT-S	7978.45	0.87	7922	142949.80

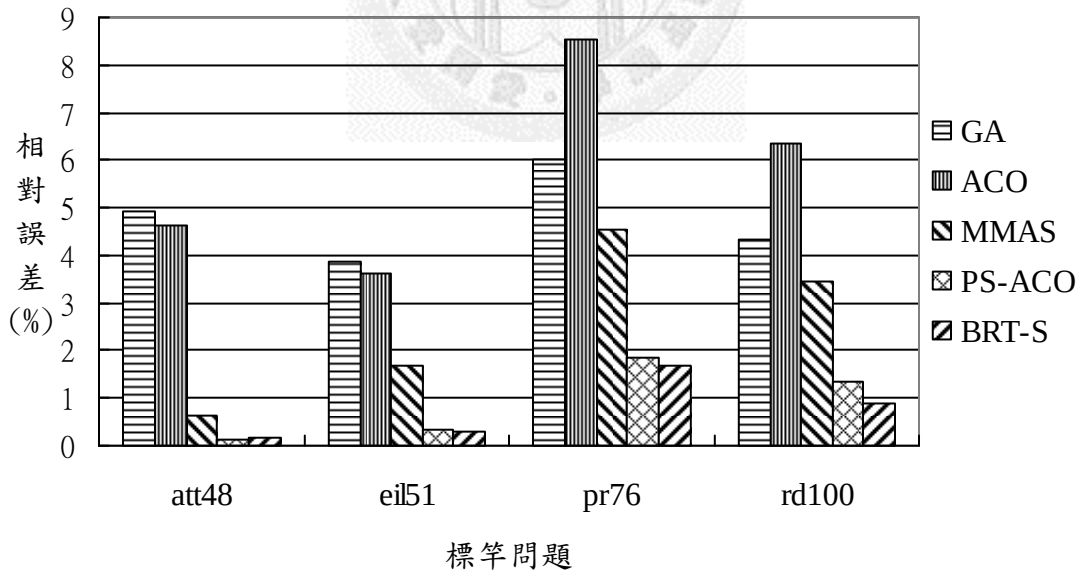

圖 4.5 BRT-S 和四種啟發式演算法求解路線長相對誤差比較

第三組實驗係比較求解大 TSP 型標竿問題的成效。參考並比較 Yang 和 Chou (2009)提出的 Superior/Inferior Segment-Discriminated Ant System (SDAS)法和五種蟻拓法的求解數據。五種蟻拓法分別是 Ant System (AS)、Ant Colony System (ACS)、Max-Min Ant System (MMAS)、rank-based Ant System (AS_rank)和 elitism-based Ant System (AS_elite)。實驗設定的停止條件依循文獻中使用各演算法求解使用的呼叫目標函數次數平均值設定上限。表 4-5 的 BRT-S 參數設定中，第一項是標竿問題的求解次數，第二項是執行呼叫目標函數次數上限。由於目標函數呼叫次數相當有限，而 BRT-S 演化中需要足夠的演化迭代才能發揮自然劣化與人工擴充縮減作業的效果。因此，使用較少的訊息傳輸者進行求解，以增加演化的迭代次數。頻寬初始配額設定為傳輸者的五倍，確定型模式使用率設定在 0.9 至 0.99 範圍內。自然劣化率採取較高的設定值，使用高耗損的環境，讓較少被解代理人挑選中線段上的頻寬因沒有執行擴充頻寬作業而減少。頻寬配額增減比例設定為 0.01。

表 4.5 BRT-S 在實驗範例三中四個標竿問題參數設定內容

參數設定	標竿問題			
	a280	**att532**	**d1291**	**fl1577**
實驗次數	30	30	10	10
執行呼叫目標函數次數	8227	5463	2923	2081
訊息傳輸者數量 m	20	20	20	20
頻寬初始配額量 b_0	100	100	100	100
確定型挑選模式使用率 λ	0.99	0.99	0.9	0.9
自然劣化率 δ	0.5	0.5	0.75	0.8
頻寬配額增減比例 ρ	0.01	0.01	0.01	0.01

表 4.6 BRT-S 和六種蟻拓最佳化技術在大型 TSP 範例中求解路線長及呼叫目標函數次數比較

標竿問題 (最佳路線長)	求解法	平均最佳解路線長	最佳解路線長標準差	相對誤差 (%)	找到最佳解平均呼叫目標函數次數	平均呼叫目標函數次數
a280	SDAS	2856.93	115.661	10.444	7397.3	8015.3
(2586.77)	AS	3067.82	40.085	18.597	4446.7	7966.0
	ACS	2981.63	58.350	15.264	4950.0	8364.0
	MMAS	2915.54	84.239	12.710	8337.3	8584.7
	AS_rank	3027.98	48.254	17.056	5769.3	8008.7
	AS_elite	2952.95	54.730	14.156	7296.7	8427.3
	BRT-S	2909.33	56.635	12.470	5557.7	8227.0
att532	SDAS	32231.50	559.337	16.418	4853.3	5536.7
(27686.00)	AS	33290.03	263.675	20.241	3900.7	5686.0
	ACS	32526.90	227.463	17.485	2196.7	5294.0
	MMAS	35286.83	785.648	27.454	4456.7	5239.3
	AS_rank	32992.10	278.888	19.165	4033.3	5740.0
	AS_elite	32611.83	505.634	17.792	4318.7	5284.0
	BRT-S	32212.80	334.142	16.351	3466.8	5463.0
d1291	SDAS	61363.03	837.163	n/a	2190.0	2870.0
(n/a)	AS	59680.17	603.799	n/a	2266.0	2760.0
	ACS	58232.76	537.835	n/a	2198.0	2984.0
	MMAS	64409.44	568.871	n/a	2706.0	2962.0
	AS_rank	59811.86	386.568	n/a	2412.0	2982.0
	AS_elite	59337.51	434.469	n/a	2498.0	2980.0
	BRT-S	59863.70	668.605	n/a	2350.4	2923.0
fl1577	SDAS	26222.18	302.145	17.858	1454.0	2014.0
(22249.00)	AS	27309.92	240.343	22.747.	1638.0	2100.0
	ACS	26378.26	273.728	18.559	1386.0	2100.0
	MMAS	31399.05	395.484	41.126	2014.0	2078.0
	AS_rank	27571.13	202.879	23.921	1732.0	2094.0
	AS_elite	27380.95	203.192	23.066	1654.0	2100.0
	BRT-S	26475.30	356.018	18.996	1339.5	2081.0

求解結果展示於表 4-6。第一欄是標竿問題與城市數量，括號內是已知全域最佳解路線長。第二欄是求解法名稱。第三到六欄分別列最佳解的路線長平均、標準差、相對誤差和求得最佳解時呼叫目標函數次數平均。第七欄是各演算法呼叫目標函數次數平均。BRT-S 演化的停止條件，是由這六種蟻拓法呼叫目標函數次數取平均值設成目標函數呼叫次數上限。

如表中結果顯示，在 a280 問題中 BRT-S 求解表現略遜於 SDAS，但比較兩者求得的目標函數值的標準差及平均呼叫目標函數次數，BRT-S 求解效果較為穩定且能較快速的求得最佳解。在 att532 問題中 BRT-S 求得解的品質明顯優於其他方法，且花費較少的呼叫目標函數次數。最後在 fl1577 問題中，BRT-S、SDAS

和 ACS 能求得比其他三種方法好的解，但 BRT-S 表現略遜於 SDAS 和 ACS。如果從相對誤差來看差距在 0.5 至 2%之間，算是相當小的差距。總結第三組實驗，在求解大型 TSP 時 BRT-S 雖無明顯優於其他蟻拓法，但其求解品質與效率亦有一定的水準。

小結三個 TSP 測試範例的結果，BRT-S 在求解小型和大型 TSP 時，求解的品質與效率有不錯的表現。和蟻拓優化法進行比較，雖然在少數問題中求解表現沒有明顯超越其中幾種蟻拓法，但整體而言，花費的求解時間較其他方法來的精簡，平均求解品質也多有優於其他方法的表現。因此 BRT-S 在求解 TSP 時，和其他廣為人知的演算法相較下仍有優良的表現。

4.3 BRT-S 求解裝箱問題範例測試

本節介紹兩組實驗測試驗證 BRT-S 求解 BPP 的品質和效率。第一組測試 BPPLIB 內由 Scholl、和 Klein (1997)設計的一系列標竿問題。比較若在使用相同容量限制、相同的箱子數、及相同的物品數下，不同的物品容量屬性值分布是否會影響 BRT-S 的求解效果。第二組實驗求解相同的標竿問題，測試使用不同的目標函數對演化結果的影響，即目標函數分別使用公式 3.3 及 3.4。

第一組測試的範例是 BPPLIB 中 set_2 系列的標竿問題；這些標竿問題是根據表 4-7 的參數產生各物品的容量屬性。此系列問題的箱子容量限制 L 設定是 1000，處理的物品數 N 是 50。物品的容量屬性值平均依箱子容量限制設有 $\frac{L}{3}$、$\frac{L}{5}$、$\frac{L}{7}$、及 $\frac{L}{9}$ 四個水準。物品容量平均值愈大箱子內可放置物品的數量越少。物品容量屬性的最大偏差量 B 設定為三個水準，分別是 20%、50%、和 90%。B 值越大者代表物品容量變異越高。組合兩組參數組成可得 12 組參數組合。每個參數組合分別產生 10 個標竿問題。問題的命名方式以 NaWbBcRz 表示，格式中的 a、b、c、及 x 是相關參數的水準索引。以標竿問題 N1W2B3R1 為例，其參數水準分別是物品數 $N=50$、物品容量平均 $W=\frac{L}{5}$ 及物品間容量最大偏差量 $B=90\%$；而 Rx 的 x 指該參數組合的標竿問題序號。

表 4.7 標竿問題參數設定

問題參數	**參數水準**			
	1	**2**	**3**	**4**
箱子容量上限 L	1000	n/a	n/a	n/a
物品數量 N	50	n/a	n/a	n/a
物品容量平均 W	$\frac{L}{3}$	$\frac{L}{5}$	$\frac{L}{7}$	$\frac{L}{9}$
物品容量最大偏差量 B	20%	50%	90%	n/a

第一組實驗分別求解 12 組問題的編號 R0 標竿問題。求解此 12 個標竿問題時，當迄今最佳解沒有裝箱結果違反箱子容量限制時即停止演化。各問題求解十次統計停止演化時的呼叫目標函數次數平均進行比較。求解時 BRT-S 的參數設定如下：訊息傳輸者數目等於問題物品數量，$m=50$；頻寬初始配額為傳輸者數量的五倍，$b_0=250$；確定型挑選模式使用率設定為 $\lambda=0.9$，自然劣化率為 $\delta=0.5$，屬中耗損環境；頻寬配額增減比例設為 $\rho=0.005$。實驗結果比較列於表 4-8 中。

表 4.8 求解不同物品屬性問題時求得符合容量限制的解呼叫目標函數次數平均比較

問題題目	呼叫目標函數次數平均	最小箱數
N1W1B1R0	690944.9	18
N1W1B2R0	867682.2	17
N1W1B3R0	398890.3	17
N1W2B1R0	19312.7	11
N1W2B2R0	146962.9	10
N1W2B3R0	112936.0	10
N1W3B1R0	22068.3	7
N1W3B2R0	28.0	8
N1W3B3R0	20554.3	7
N1W4B1R0	21.8	6
N1W4B2R0	22.1	6
N1W4B3R0	21.1	6

表中顯示當物品容量屬性值越大時(W 水準值低者)，求得無違反箱子容量限制解時呼叫的目標函數次數越多。原因是因箱子容量限制不變且物品的平均容量較大；導致箱內的物品數驟降必須使用較多的箱子。由於本研究提示的 BPP 求解模式是箱數已知的裝箱問題，當已知的最小箱數越多時，解的空間越大。以 N1W1B1R0 和 N1W4B1R0 兩個問題來比較，已知最小箱數分別是 18 箱和 6 箱。進行一次建構步驟需要衡量候選者的次數比達 $\frac{18}{6}=3$ 倍。而建構一個完整的解需

要 50 次的建構步驟運算的次數相差達3^{50}倍。所以物品平均容量較大的問題需要花費較多的呼叫標函數次數，以演化各箱裝箱結果成無違反容量限制的合理解。反之，物品平均容量較小的問題所需箱數較少，使得解搜索空間較小，BRT-S 能快速演化出無違反容量限制的解。

在第一組實驗中，已驗證 BRT-S 能求得各種參數組合的 BPP 問題的最小箱數解。第二組實驗則進一步測試 BRT-S 對於平衡箱子間總容量的效果。使用同第一組實驗中的 12 個不同容量屬性分佈的標竿問題，測試本研究針對 BPP 所設計的兩種目標函數在兼顧箱子間總容量平衡和演化出無違反容量限制解的效能；兩種 BPP 目標函數如式 3.3 及 3.4 所示。前者僅以最小化裝箱結果箱間總容量的標準差，而後者在目標函數中增加了違反容量限制箱數的因子e，以驅策解的演化往減少違反箱數的方向前進。

比較公式 3.3 及 3.4 可知在求得最小箱數時，$f^2(S)$的目標函數值等於$f^1(S)$。本實驗分別使用兩個目標函數求解標竿問題，以求得的最佳解違反容量限制的箱子數和箱子間總容量標準差作為衡量標準。演化停止條件設定為呼叫目標函數達 200000 次。BRT-S 的參數設定同第一組實驗，每個實驗執行十次，統計違反容量限制箱數的平均值和箱子間總容量標準差的平均值，結果列於表 4-9。

表 4.9 使用不同目標函數最佳解平均違反箱數及平均目標函數值比較表

問題題目	目標函數一 $f^1(S)$		目標函數二 $f^2(S)$	
	平均違反箱數	平均標準差	平均違反箱數	平均標準差
N1W1B1R0	4.6	93.991	2.8	95.154
N1W1B2R0	5.0	39.243	3.6	38.216
N1W1B3R0	1.3	34.903	0.6	28.715
N1W2B1R0	0.2	38.323	0.0	36.802
N1W2B2R0	0.5	13.686	0.2	11.247
N1W2B3R0	0.3	11.935	0.0	9.504
N1W3B1R0	0.0	2.602	0.0	1.916
N1W3B2R0	0.0	8.131	0.0	7.515
N1W3B3R0	0.0	4.915	0.0	4.539
N1W4B1R0	0.0	2.260	0.0	1.318
N1W4B2R0	0.0	2.739	0.0	2.218
N1W4B3R0	0.0	4.202	0.0	2.936

表 4-9 中第一欄是標竿問題題目，第二、三欄是使用目標函數一 $f^1(S)$ 求得的平均違反箱數和標準差。第四、五欄是使用目標函數二 $f^2(S)$ 求得的平均違反箱數和標準差。表 4-9 顯示除 N1W1B1R0 問題外，其餘問題使用目標函數二求得的平均違反箱數和平均標準差都優於使用目標函數依。結果証實目標函數二的設計較能兼顧違反箱數望小的目標。由第二組實驗結果可知，本研究針對 BPP 所設計的目標函數二 $f^2(S)$，在求解無違反箱子容量限制和平衡箱子間的總容量兩方面都有很好的表現。

綜合上述展示的兩個 BPP 實驗範例，本研究提出的 BRT-S 演算法求解 BPP 時，對於具有不同物品屬性的標竿問題，無論是求解品質或所使用的呼叫目標函數次數都有很好的表現。此外，本研究針對 BPP 導入違反箱子容量限制數量因子的目標函數，比單純使用箱子間總容量標準差更能兼顧箱子間的平衡和降低違反的箱數。

第五章　結論與建議

萬用啟發式優化演算法(meta heuristic)大多是師法自然現象或生物特性的求解程序而來。應用時須以電腦程式語言實作求解系統，來演化出求解問題的最佳結果。本研究提出的 BRT-S 法仿現實網路世界中頻寬資源有限的情況，各訊息傳輸者透過競爭取得頻寬來傳輸訊息。同時模擬頻寬隨環境和人為影響變動。本研究開發出以求解環境具資源限制的啟發式離散優化演算法。

5.1 結論

本研究參考現有啟發式演算法的優缺點，並以網路傳輸特性為主軸，創新開發出仿頻寬限制資料傳輸之萬用啟發式演算法。演算的基本架構是採取建構式的求解方式，在求解資源有限制的環境下搜索最佳解。針對離散優化問題，本研究設計出一套完整的演算流程，並設計出求解物件排序優化問題和物件分群優化問題的 BRT-S 求解法。透過範例測試和其他演算法比較求解結果，驗證 BRT-S 求解效能。歸納出以下結論：

1. BRT-S 是首創求解資源有限的代理人競合式啟發式演算法。求解過程中，解代理人間存在競爭資源的關係，須透過競爭來取得頻寬進行傳輸。且僅對搶得資源完成解建構的代理人進行評估，減少解評估的資源浪費。在求解離散優化問題時，可避免過度相似的解產生，節省運算資源。

2. BRT-S 演化過程中，透過解代理人的合作，依照問題中物件關連線段參與解建構頻繁程度多寡，透過擴充及縮減頻寬的方式，使各線段被挑選的機率有高下之分。使求解演化過程能較容易收斂。

3. 測試 TSPLIB 內的標竿範例，結果顯示 BRT-S 不論在小型或大型 TSP 問題中求解品質相當好，花費的運算資源也較其他啟發式演算法來的精簡。BRT-S 在求解 TSP 問題時普遍具有良好的品質與效能。

4. 測試 BPPLIB 內的標竿範例，顯示 BRT-S 面對不同物品屬性值分佈的標竿問題，皆能有效求得不違反容量限制的解。求解 BPP 時導入違反容量限制箱數因子的目標函數，會比只考慮箱子間的總容量平衡性的目標函數更能兼具平衡性與不違反容量限制。BRT-S 求解 BPP 問題能有效求得不違反容量限制並平衡箱子間總容量的解。

5.2 後續研究建議

本研究提出的仿頻寬限制資料傳輸優化演算法，經範例驗證 BRT-S 能有效求解物件排序及物件分群優化問題。然而此演算法尚有擴充的空間，有待後續研究繼續改良。本研究提出幾點建議，供後續研究參考：

1. BRT-S 法是以求解離散優化問題進行設計，目前僅針對物件排序及物件分群優化問題中的旅行銷售員問題及一維裝箱問題進行求解。後續可開發其他離散優化問題的 BRT-S 求解方法。擴大求解離散優化問題的涵蓋性。

2. 在求解 BPP 時啟發式評估值是以目前箱子的剩餘容量設定。僅以目前箱子的資訊作為衡量標準略嫌粗糙，可加入進行配置的物品相關屬性，使啟發式評估值能對求解演化有更好的效果。

參考文獻

Bennell, J. A. ,and Dowsland, K. A. (1999), ‘A tabu thresholding implementation for the irregular stock cutting problem.’ Int. J. Prod. Res. 37,4259-4275.

Bullnheimer, B., Hartl, R. F., and Strauss, C. (1997), “A new rank based version of the ant system: Acomputational study,” Working paper, University of Vienna, Austria.

Dorigo, M., and Gambardella, L. M. (1997), “Ant Colony System: A cooperative Learning Approach to the Traveling Salesman Problem,” IEEE Transactions on Evolutionary Computation, vol.1, no.1, pp. 53-66.

Dowsland, K. A., (1993), “Some experiments with simulated annealing techniques for packing problems.” European Journal of Operational Research 68, 389–399.

Dueck, G., and Scheuer, T. (1990), “Threshold Accepting: A General Purpose Optimization Algorithm Appearing Superior to Simulated Annealing.” Journal of Computational Physics, 90(1), 161-175.

Falkenauer, E. (1994), “A new representation and operators for GAs applied to grouping problems.” Evolutionary Computation ,1994(2), 123–144.

Garey, M., Graham, R., Johnson, D., and Yao, A. (1976), “Resource Constrained Scheduling as Generalized Bin Packing,” J. Comb. Theory, Ser. A 21.

Glover, F. (1989), “Tabu search-part I.” ORSA journal on Computing, 1(3), 190–206.

Goldberg, D. E., and Lingle, J. R. (1985), “Alleles, Loci and the TSP.” Proceedings of the First International Conference on Genetic Algorithms and Their Applications, 154–159.

Holland, J.H. (1975), “Adaptation in Natural and Artificial System,” The University of Michigan Press.

Kennedy, J., and Eberhart, R. (1995), “Particle Swarm Optimization,” In Proceedings of the IEEE International Conference on Neural Networks, vol. 4, 1942-1948.

Kirkpatrick, S., Gelatt, C. D., and Vecchi, M. P. (1983), “Optimization by Simulated Annealing.” Science, 220, 671-680.

Stutzle, T., and Hoos, H. (1997), “The MAX-MIN Ant System and Local Search for the Traveling Salesman Problem,” In Proceedings of the Fourth International Conference on Evolutionary Computation. , IEEE Press, 308-313.

Shuang, B., Chen, J. ,and Li, Z. (2009), “ Study on hybrid PS-ACO algorithm,” Applied Intelligence, Volume 34, Number 1, 64-73.

Stutzle, T., and Hoos, H. (2000), “MAX-MIN Ant System,” Journal of Future Generation Computer Systems, 889-914.

Scholl, A., Klein, R., and Jurgens, C. (1997), “ BISON: a fast hybrid procedure for exactly solving the one-dimensional bin packing problem,” Computers & Operations Research 24, 627-645.

Ugur, A., and Aydim, D. (2009), “An interactive simulation and analysis software for solving TSP using Ant Colony Optimization algorithms,” Advances in Engineering Software, Volume 40, Issue 5, May 2009, Pages 341-349

Yang, F., and Chou, Y. (2009), “Superior/Inferior Segment-Discriminated Ant System for combinatorial optimization problems,” Computers & Industrial Engineering, Volume 57, Issue 2, September 2009, Pages 475-495.