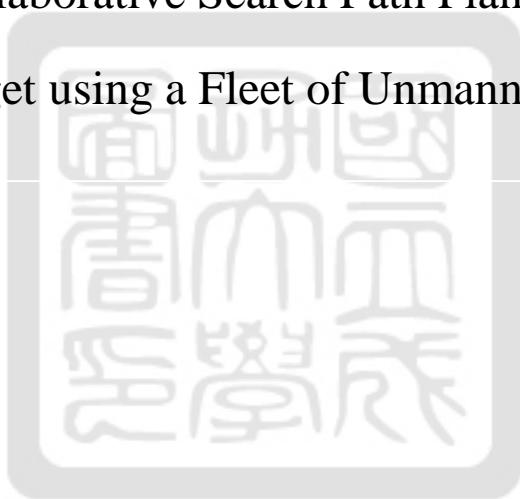


國立成功大學  
工業與資訊管理學系碩士班  
碩士論文

以無人機群搜尋單一靜態標靶之最佳協同搜尋路徑規劃

問題研究

An Optimal Collaborative Search Path Planning Problem for an  
Immobile Target using a Fleet of Unmanned Aerial Vehicles



研究 生：呂 昇 軒

指 導 教 授：王 逸 琳 教 授

中華民國一百零八年五月

國立成功大學

碩士論文

以無人機群搜尋單一靜態標靶之最佳協同搜尋路徑規  
劃問題研究

An Optimal Collaborative Search Path Planning  
Problem for an Immobile Target using a Fleet of  
Unmanned Aerial Vehicles



研究生：呂昀軒

本論文業經審查及口試合格特此證明

論文考試委員：



王逸琳

莊坤遠

陳柏

吳政鴻

指導教授：王逸琳

系(所)主管：

王惠嘉

中 華 民 國 108 年 5 月 26 日

## 摘要

長久以來，每當在深山或偏遠地區發生搜索與救援等服務需求任務時，搜救單位必須立即調派大量人力於事故地區沿路徒步搜索，期能儘快找到停留於某路段上的靜態標靶（譬如受傷的登山客）。由於欲搜尋之範圍廣大且路況不明，此種搜尋方式受制於平面路網的連結，即便事先已得知該靜態標靶較可能的位置，仍需沿路逐段徒步跋涉探索；相對而言，無人機具有較快的三維空間移動能力，可跳脫二維路網的路段連結限制；只要續航力足夠，同時指派多架無人機分工協同搜尋，應可更快完成搜尋任務，同時也能減少搜救團隊發生人身意外。本研究欲探討下述之搜救路線規劃問題：給定一網路（譬如某區域的諸多登山步道路線），假設單一靜態標靶僅會停留在某段節線上，且各節線上可尋獲該標靶的機率、可供無人機充換電的節點位址等資訊皆為已知，我們將規劃  $K$  架無人機的最佳移動路線，尋訪該標靶可能存在（即機率為正值）之所有節線，以使總期望搜尋時間能最短。

先前探討使用人力徒步沿路搜尋的相關文獻，大多僅能處理特殊的「單位節線長度網路」(unit graph，亦即所有節線的長度皆為單位長度)，本研究則首度提出可以處理一般節線長度的數學規劃模式。我們參考過去文獻針對目標式之期望搜尋時間的處理方式，將目標式由非線性改為線性，並利用層空網路(Level-Space Network)為基礎以建構數學規劃模式。而無人機必須尋訪所有之標靶存在機率為正值的節線（但不見得為所有節線），此與文獻中的鄉村多郵差問題(K Rural Postmen Problem，KRPP)類似；然而本研究之目標式為最小化總期望搜尋時間，與 KRPP 僅將最長的車輛移動時間最小化相比，本研究的目標式更為複雜。為能善用無人機的移動機動彈性，本研究假設無人機在節點間有兩種移動模式：(1)慢速的「沿路搜尋模式」，及(2)快速的「節點移動模式」，前者與人員徒步沿路移動方式相同，僅能在原節線路段連續進行；而後者則可在續航力允許下自由地在任兩節點間移動。為簡化問題，我們亦假設無人機除充電

(耗費固定時間，且充完必滿電)與結束任務外，必須一直保持移動。基於無人機續航力限制的現實考量，本研究亦假設充換電站位址皆在某些已知的節點上，且無人機在電力耗盡前可自行前往充換電站。上述諸項設定，除讓本研究問題更符合現實外，也大幅提高其困難度與挑戰性。

在實際進行數值測試後，我們發現數學規劃模型的作法將耗時過長，無法處理規模為中或大型之網路，因此本研究再發展數種啟發式演算法，將需要進行搜尋的節線編碼為一組解，設計結合動態規劃法與回溯法之解碼演算法，將該解再轉變為實際之飛行路徑，最後再利用基因演算法與區域搜尋法以求解最小之總期望搜尋時間。為加速求解，我們將無人機之電量限制視為軟性限制處理，數值測試結果顯示數學規劃模型雖可得到理論最佳解，但僅可求解小規模的網路；而區域搜尋法則皆可在短時間內求得近似最佳解。最後，我們推薦使用本論文發展之數學規劃法建模處理規模較小的搜救服務需求，而大規模的搜救服務作業則可使用本論文設計之區域搜尋法處理之。

關鍵字：搜救、路線規劃、無人機、整數規劃、啟發式演算法

# An Optimal Collaborative Search Path Planning Problem for an Immobile Target using a Fleet of Unmanned Aerial Vehicles

Yun-Hsuan Lu

I-Lin Wang

Department of Industrial and Information Management

## SUMMARY

Unmanned Aerial Vehicles (UAVs) are very useful for search and rescue missions, since they are more mobile, faster, remotely controllable, and can cover wider range in shorter time. In this thesis, we focus on the application of locating a single immobile target on a general network by a fleet of UAVs. Assuming the probability of finding the target on each arc is estimated beforehand, we seek optimal UAV routings so that the target can be located in a minimum expected time, whereas the UAV routings also consider the necessary battery swapping or recharging at specific nodes. Our settings are more realistic since we consider two UAV moving modes: a slower searching mode along arcs, and a faster moving mode between possibly nonadjacent nodes. A mixed integer programming formulation is proposed on a level-space network, which can only deal with networks of smaller sizes. Two heuristic algorithms, genetic algorithm (GA) and local search algorithm (LS), are designed to calculate solutions of good qualities in short time. The numerical results of our computational experiments indicate that LS can calculate high-quality solutions within limited time among all cases.

**Key words:** Search and Rescue, Path planning, Integer programming, Unmanned Aerial Vehicle, Heuristic algorithm

## INTRODUCTION

We focus on the search and rescue (SAR) mission for a single immobile target over some area that is difficult, dangerous, or time consuming to access on foot, assuming the probability of finding the target on each arc is estimated beforehand. Conventionally, it takes an SAR team much time, efforts, and even leads to casualty for conducting such a difficult and dangerous SAR mission. On the other hand, such an SAR mission is perfect to be conducted by UAVs, since UAVs are more mobile, faster, remotely controllable, and can cover wider range in shorter time. The challenges of effective deployment of UAVs rely on their correct routing and scheduling strategies, which should be carefully designed beforehand based on solid and rigorous mathematical analysis.

Conventional SAR routing literature focus on finding a minimum-length tour searching over all arcs in a network, similar to the Chinese Postman Problem (CPP). To match the more realistic settings of SAR mission, this thesis assumes the probability of locating an immobile target on each arc is estimated beforehand, then the focus is not only on the minimum travel time or distance as CPP but on the minimization of expected time searching through the arcs of positive probability. Related literature on searching for a single immobile target mostly focus on a specialized network where all arcs have the same unit length, and only has a searching mode in movement. Here we assume a UAV has two moving modes: a searching mode that makes a UAV to move slowly along each road segment (i.e., an arc) resembling the human searching, and a moving mode that makes a UAV to move quickly between nodes possibly nonadjacent by an arc but within its range. The moving mode allows a UAV move flexibly between nodes with faster speed, as long as its range is allowed. This makes UAVs more flexibility and shorten the searching time than human beings. Moreover, we consider the range constraint of the UAV movement, which forces a UAV flying over a battery swapping or recharging facility located in some specific nodes whenever necessary. All together makes routing fleet of UAVs for searching a single immobile target in the shortest expected time a very difficult problem.

## MATERIALS AND METHODS

To calculate an exact optimal solution, we formulate this problem as an integer program (IP) over a level-space network to deal with general networks and revises the objective from non-linear form to linear form. The UAVs need to traverse all the arcs of positive probability. Note that some zero-probability arcs are not required to be traversed but an optimal route may still contain such arcs because they may serve as good connection to required arcs. Thus the setting is quite similar to the K Rural Postmen Problem (KRPP) but with a different

objective function. Our IP formulation can avoid subtours and can incorporate with both searching and moving modes using parallel search arcs and move arcs, respectively, in the level-space graph. Theoretically, a search arc should be passed through at most once, whereas a move arc might be traversed for several times. To solve the IP formulation, we need to estimate the number of levels (i.e. arcs) first in a level-space network. In particular, we use the longest routing length calculated by the KRPP as the lower bound, and then using its three times as the upper bound. Unfortunately, the IP formulation can only deal with small scale networks. Therefore, we further design heuristic algorithms: the Genetic Algorithm (GA), and the Local Search Algorithm (LS) which would encode each routing solution as a sequence of arcs to be traversed. For GA, we refer to the popular GA framework for solving the Traveling Salesman Problem. For LS, we use the methods proposed in Hashimoto and Yagiura (2008), which used 2-opt\*, Or-opt and cross exchange to search neighborhood. To make our LS more effective, we prefer selecting the UAVs of the longest and shortest completion time to do the exchange. Since the trajectories of UAVs are difficult to calculate, we use the sequence of the search arcs to represent a solution. Our decoding algorithm is summarize as follows: Given the sequence of the search arcs and start from the first search arc. First, we use dynamic programming to find the best direction of the search arcs that minimize the expected search time and calculate the farest search arc reachable for that UAV at its current battery level. Second, check whether the remaining battery is sufficient for reaching out the nearest charging station. If no, we backtrack to the previous search arc until there is sufficient battery for reaching the nearest charging station. Then, we check whether all the search arcs are traversed. If yes, then stop; otherwise, we conduct the decoding algorithm for the rest of the search arcs.

## RESULTS AND DISCUSSION

Two methods are used to generate problem instances: First, we use python package, NetworkX, to generate random network and tree network. And we randomly generate hub and spoke network. Then, we construct small, medium and large instances for each type of networks. Based on our testing, we have found that the IP models, with or without initial solution, can only deal with small-scale problems. On the other hand, LS can calculate good solutions in a shorter time in all instances. We also compare the expected search time (i.e., objective value) between searching by UAVs and by human on foot, and learn that UAVs can save at least 20% of expected search time in the tree networks and at least 50% in the hub-and-spoke networks. These results indicate the significant time saving by using UAVs in conducting the search and rescue mission.

## CONCLUSIONS

To enhance the search efficiency and effectiveness, we not only use UAVs to search but also consider a search path planning problem with its objective being the expected search time minimization. In addition, we take into account the UAV range, battery level, and locations of battery swapping or recharging facilities. Comparing with other search path planning literatures that focus more on special graphs (e.g., unit graphs), ours are designed for general networks and thus more practical. We propose a novel integer program formulation and several heuristic algorithms. The results of our computational experiments show that LS can calculate good solutions in a shorter time. Moreover, searching by a fleet of UAVs does dramatically reduce the expected search time of searching by humans on foot. For future research, we suggest to enhance search efficiency by considering mobile charging stations, and new techniques in calculating IP optimal solutions.



## 誌謝

將這一頁獻給所有曾幫助過我的師長、家長、朋友們。時光匆匆，6 年前踏入成大如今也將要畢業離開，在研究所這 2 年的時間，我認為是自己成長最多的階段，謝謝逸琳老師的指導，讓我有機會參與許多比賽、專案，順利完成論文，並在過程中學習老師對於事情的處理態度與思考方式，也特別謝謝老師帶我們去美國參加比賽，那是一次難忘的回憶，也很高興學弟妹們有機會再去。我相信這些寶貴的經驗，對於未來都非常有幫助。

謝謝爸媽長久以來無願無悔付出，讓我能無後顧之憂的完成大學與研究所的學位，順利完成論文，我真的非常的幸運、幸福。此外，也謝謝你們當初讓我來就讀成大工工，我在這接觸到很多領域也學習到很多有趣的知識。

謝謝實驗室的夥伴們，首先，謝謝雪湄跟我一起討論作業，那時候寫作業與專案真的每天寫到凌晨，也謝謝妳時常帶我挖掘好吃的台南美食。謝謝嘉豪為實驗室帶來歡笑，你的熱心助人幫助大家很多事情，真的該多向你學習。謝謝筱昀、BK、思涵、冠偉，謝謝你們帶領我認識這個 ilin lab。也謝謝彥瑋、宗瀚、晏慈，你們的加入為這個實驗室帶來活力，也祝福你們比賽拿第一，找到好工作。

後謝謝蹭飯團的欲老師、妖怪、異果、雪湄、EJ、陞瑋、雅宣，有你們的陪伴讓我平常吃飯不無聊，出遊打電動都有伴，謝謝你們。

最後謝謝過去的自己堅持到現在，接下時間來還很長，期許未來的自己也要繼續努力，那研究所就先下台一鞠躬囉。

# 目錄

摘要.....	I
誌謝.....	VII
目錄.....	VIII
圖目錄.....	X
表目錄.....	XI
第一章 緒論 .....	1
1.1 研究背景.....	1
1.2 研究動機.....	2
1.3 研究目的.....	3
1.4 研究範圍.....	4
1.5 論文架構.....	5
第二章 文獻回顧.....	6
2.1 傳統人力徒步搜救路線規劃相關研究 .....	6
2.1.1 搜尋動態標靶.....	7
2.1.2 搜尋靜態標靶.....	7
2.2 無人機搜救路線規劃相關研究 .....	10
2.3 節線排程問題相關文獻 .....	12
2.4 小結 .....	14
第三章 無人機群搜尋路徑之數學規劃模型 .....	15
3.1 問題描述.....	15
3.2 問題假設.....	16
3.3 網路結構.....	17
3.4 固定充定站位置之數學模式 $M_L^{C_0}$ .....	17
3.4.1 符號定義.....	18
3.4.2 期望搜尋時間 .....	19
3.4.3 數學模式.....	21
3.4.4 考慮無人機群起訖組合模式 .....	24
3.5 最佳階層數 $L^*$ .....	25
3.6 小結 .....	25
第四章 無人機群搜尋路徑規劃之求解演算法設計 .....	26
4.1 編碼方式(Encode) .....	26

4.2 解碼演算法(Decoding Algorithm , DA) .....	27
4.2.1 演算法步驟.....	28
4.2.2 範例說明.....	32
4.3 基因演算法 (Genetic Algorithm , GA) .....	34
4.4 區域搜尋法 (Local Search , LS).....	38
4.5 小結 .....	44
第五章 數值分析.....	45
5.1 測試資料參數設定 .....	45
5.2 測式網路圖設定 .....	46
5.3 固定充換電站點之無人機群路徑規劃問題之測試 .....	46
5.3.1 整數規劃模式比較 .....	46
5.3.2 數學演算法測試.....	49
5.4 使用無人機之效益比較 .....	50
5.5 小結 .....	51
第六章 結論與未來研究議題建議.....	52
6.1 結論 .....	52
6.2 未來研究.....	53
附錄.....	55
參考文獻 .....	58

## 圖目錄

圖 1.1	待搜尋網路圖一 .....	2
圖 1.2	待搜尋網路圖二 .....	3
圖 1.3	待搜尋效益網路圖二 .....	3
圖 1.4	考慮效益之尋訪順序排程 .....	4
圖 1.5	最佳尋訪順序排程 .....	4
圖 2.1	待搜尋網路圖三 .....	13
圖 3.1	待搜尋網路圖 .....	16
圖 3.2	原問題之網路 .....	17
圖 3.3	以搜尋節線與移動節線建構之網路 .....	17
圖 3.4	本研究單一無人機 .....	18
圖 3.5	本研究多無人機之 .....	18
圖 4.1	路徑表示方法 .....	27
圖 4.2	無人機路徑圖 .....	27
圖 4.3	編碼示意圖(含節線方向) .....	27
圖 4.4	編碼示意圖(不含節線方向) .....	27
圖 4.5	動態規劃網路圖 .....	28
圖 4.6	DA 範例說明網路圖 .....	32
圖 4.7	第一次迭代 DP 網路圖 .....	33
圖 4.8	第二次迭代 DP 網路圖 .....	34
圖 4.9	選擇父母及交配的分割點 .....	36
圖 4.10	進行交換與複製 .....	36
圖 4.11	最後小孩結果 .....	37
圖 4.12	突變策略-交換法 .....	37
圖 4.13	突變策略-插入法 .....	37
圖 4.14	編碼示意圖 .....	38
圖 4.15	2-opt* 交換示意圖 .....	40
圖 4.16	cross exchange 交換示意圖 .....	41
圖 4.17	Or-opt 交換示意圖 .....	43
圖 4.18	區域搜尋法執行流程 .....	44
圖 5.1	測資 random _m 下 K = 2 時 模式 $M_L^{C_0}$ 之收斂情形 .....	48
圖 5.2	測資 random _m 下 K = 2 時 模式 $\overline{M_L^{C_0}}$ 之收斂情形 .....	48

## 表目錄

表 2.1	傳統人力徒步搜救路線規劃文獻比較 .....	9
表 2.2	無人機搜救路線規劃相關文獻比較 .....	12
表 4.1	動態規劃法流程 .....	29
表 4.2	回溯法流程 .....	30
表 4.3	第一次迭代之 DP 求出之線段服務完之期望搜尋時間與電量 .....	33
表 4.4	第二次迭代之 DP 求出 .....	34
表 4.5	2-opt*演算法流程 .....	40
表 4.6	cross exchange 演算法流程 .....	42
表 4.7	Or-opt 演算法流程 .....	43
表 5.1	測試資料參數設定 .....	45
表 5.2	網路圖資訊 .....	46
表 5.3	混整數規劃模式之問題大小估算 .....	47
表 5.4	數學模式測試(求解品質與效率) .....	48
表 5.5	數學演算法參數設定 .....	49
表 5.6	數學演算法測試(求解品質與效率) .....	50
表 5.7	使用無人機之效益分析 .....	51

# 第一章 緒論

## 1.1 研究背景

長久以來，搜索與救援事件不斷發生，依照事件的地點，搜救類型主要可分為 5 類：地面搜索與救援、城市搜索與救援、山嶺搜索與救援、海空搜索與救援、戰鬥搜索與救援，政府部門依照災害類型與規模派遣專業的搜救隊進行搜救。在搜救的過程中，搜救隊首要目標是在短時間內找到失蹤人員，否則隨著搜救時間的增加，失蹤人員存活的機率則下降，然而，在要求搜救加快的同時，搜救人員也同時承受極大的人身危險。台灣近年來經常發生搜救人員在搜救的過程中遭受人身意外，譬如 2017 年 3 月時，搜救人員在山區搜救的過程中，遭大水沖走溺斃；2018 年 4 月時，消防員在火場中搜尋受困人員時，反被困在火場中，最後導致 5 名消防員殉職。而這些援救任務若能以無人載具進行，除可避免人員傷亡，更能以較短時間完成更多區域搜尋任務，可謂一舉數得。

為了儘快找出靜態標靶，同時降低搜救之風險。無人機的使用有效增加搜救效益，當發生搜索與搜救事件時，無人機可快速投入搜救區域，無人機與搜救隊各司其職，搜救隊不再需要花費大量的時間在搜索靜態標靶，當無人機尋獲靜態標靶時，救難隊可直接前往進行救援，大幅降低搜救隊發生意外的機率。同時，由於有較快的三維空間移動能力，特別是針對雪地、峭壁等一些較難移動抵達之地形、較極端的氣溫以及能見度較差的情況，無人機移動速率與搜索效率較不受影響，且單價比直升機等載人飛行載具便宜許多。目前已有將無人機應用在搜索的領域，無人機可透過紅外線探測失蹤的登山客或是利用相機拍攝即時影像，來獲取第一手資訊。但即使無人機在搜索上極為有效率，仍需要訓練有素的操作者和飛行員，來協調多個搜救隊的搜索、無人機飛行的路徑與有策略的部署無人機

位址以使在最短時間內能找到失蹤人員。然而，本路線規劃問題的困難度及複雜度均十分高，即便是有經驗的路線規劃老手，應該無法全面考量。

## 1.2 研究動機

以往搜救路線規劃問題的處理方式，大多是由決策者自身的經驗來決定各組搜救隊的搜尋路線。因此，當搜救範圍擴大或是搜救人員分組太多時，此搜尋路線之規劃決策將難以人為主觀的判斷精算而得。本研究使用多台無人機分組搜救的方式，在路線規劃上更應將無人機的續航力與充換電節點列入考慮，以使得尋獲靜態標靶之期望搜尋時間愈小愈好。需要注意的地方是，假設標靶僅會出現在節線上。傳統人力徒步搜救路線規劃相關文獻，搜救隊必須沿路進行搜救，倘若兩個地方（即節點）並沒有路（即節線）直接相連，則搜救隊必須透過繞路的方式前往；若使用無人機進行搜救，則無人機可跳脫原網路的路段連結限制。以搜尋網路圖 1.1 為例，目標是將所有可能存在靜態標靶之節線至少經過一次。假設搜救隊自 A 點出發，若以傳統人力徒步方式，受制於原網路的連結，當搜救隊要前往 D 點時，搜救隊只能透過節線 (A,B) 與 (B,D)；反之若使用無人機，假設其續航力足夠，則可逕自從 A 點前往 D 點，此例顯示無人機的搜救路線會比傳統人力徒步方式更有彈性，因此也讓其路線規劃問題更加困難。

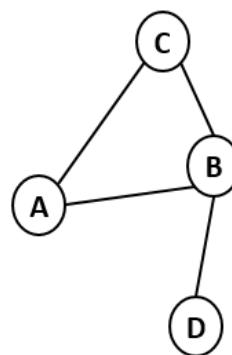


圖 1.1 待搜尋網路圖一

### 1.3 研究目的

本研究將探討無人機搜尋路徑規劃問題，將標靶可能出現的位置設定於於「節線」上，即無人機在節線上才能尋獲標靶，倘若無人機尋訪完此一節線後，該節線上將不可能再次尋獲標靶，可視為收集完節線上尋獲標靶的機率。由於並無法確定標靶出現在哪一條節線上，因此無人機群必須尋訪該標靶可能存在之節線（但不見得為所有節線），為使路徑規劃能達到最大搜尋效益，本研究之目標著重在極小化期望搜尋時間。文獻中，與本研究較有關聯的議題為「鄉村多郵差問題」(K Rural Postmen Problem, KRPP)。該議題亦探討如何派遣多組車輛尋訪所有必須搜尋的節線，目標為最小化所有車輛中通行時間最長的車輛之通行時間，且節線之間並沒有重要性的差別，僅有長短之別。在本問題中節線的尋訪雖沒有明確的規範先後關係，無人機可直接飛往任一節線進行尋訪，但是尋訪節線的順序對於期望搜尋時間影響極大，必須同時考慮節線的長度以及尋獲標靶的機率來決定尋訪順序。以待尋訪網路圖 1.2 為例來說明本研究問題決策困難性，假設有兩架搜救能力與移動速率相同之無人機，欲尋訪所有的搜尋節線，各節線上  $D_{ij}$  與  $\alpha_{ij}$  分別代表節線之長度與發現單一靜態標靶之機率。在求解方法的部分，定義每條節線  $(i, j)$  的搜索效益為  $\alpha_{ij} / D_{ij}$ ，如圖 1.3 所示，我們以效益愈高之節線優先

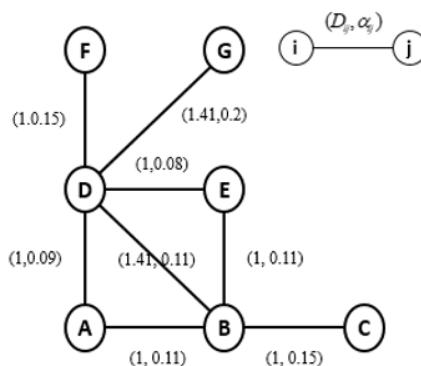


圖 1.2 待搜尋網路圖二

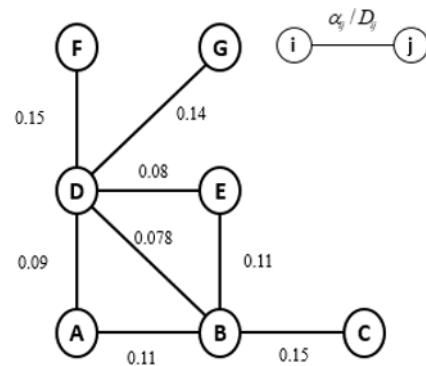


圖 1.3 待搜尋效益網路圖二

搜救的直覺方法進行求解，兩架無人機分別從效益最高

	0	1	2	3	4	5.1	時間
UAV 2	CB	BA	AD	DE	EC		
UAV 1	FD	DG	GE	EB	BD	DF	
	0	1	2.4	3.2	4.2	5.6	6.4
							時間

圖 1.4 考慮效益之尋訪順序排程

	0	1	2	3	4	6.25	時間
UAV 2	CB	BE	ED	DF	FC		
UAV 1	GD	DA	AB	BD	DG		
	0	1.4	2.4	3.4	4.8	5.9	時間
							時間

圖 1.5 最佳尋訪順序排程

之節線 FD 與 BC 開始搜救，搜救之排程結果如圖 1.4 所示，期望搜尋時間的計算方式為發現標靶之時刻乘上發現標靶之機率之累加(計算方法參考 3.4.2 節)，期望搜尋時刻為 2.11。若依照圖 1.5 之排程結果，期望搜尋時刻為 2.06，可知直覺式的方法仍有改善空間，若網路圖規模更大與複雜則直覺式的方法所得解的品質將會更差，且此簡例並未考慮無人機之充換電，若考慮無人機需要充換電，則路線之排程不僅需要考慮節線之搜救效益且需考慮與充換電節點之距離。

上述範例顯示，直覺式的方法仍有許多的改善空間，因此如何有系統地分析本問題、建立嚴謹的數學模式，並建構兼具效率與效能之演算法為本研究之目標。

## 1.4 研究範圍

本研究的研究範圍如下：

1. 建立  $K$  架無人機群之路徑規劃，已知無人機續航力與充換電之基地節點與網路上尋獲靜態標靶之機率，尋訪所有可能存在靜態標靶之節線，目標為極小化總期望搜尋時間。
2. 測試數學模型與演算法在不同網路規模與結構下之求解效能與效率。

## 1.5 論文架構

本論文之架構如下：第二章為文獻回顧，針對傳統人力徒步搜救路線規劃、無人機應用在搜救路線規劃與節線排程作業進行介紹。第三章討論已知充換電站位置下之無人機群路徑規劃問題，並提出數學模型，討論不同起訖點之組合與最佳階層數之設定。第四章會提出路徑的編碼與解碼，並將編碼方式運用在基因演算法與區域搜尋法之中。第五章為數值測試分析，比較數學模型及數學演算法在不同規模網路以及不同網路結構下之效率與效能上的差異。附錄將探討已知設立之充換電站數量下，決策最佳之設立位址與無人機群之路線，以使期望搜尋時間最小化。



## 第二章 文獻回顧

本章共分三節，首先將針對傳統人力徒步搜救路線規劃相關文獻進行探討，說明本研究探討之目標式與網路形式，接著說明無人機群搜救路線規劃的相關文獻與本研究相同與相異處，最後介紹節線排程相關文獻，本研究可被歸類為節線排程問題。

### 2.1 傳統人力徒步搜救路線規劃相關研究

搜索問題(search problem)是探討搜索人員如何找到被搜尋的標靶，研究問題大致可分為 2 個主要方向，依照搜尋標靶的特性可分為標靶的移動或位置受到搜救人員的行動而影響，譬如犯人躲避警察的追捕或是潛艇躲避反潛艇的船艦等雙側搜索問題(two-sided search problem)，與標靶的移動或位置不受到搜救人員的行動而影響，即是標靶的移動與搜救人員之前的行動相互獨立，譬如搜救行動等單側搜索問題(one-sided search problem)，本研究假設標靶無法得知搜救人員的位置，因此本問題屬於單側搜索問題。目前搜救文獻中，根據 Berger et al(2014) 的分類，問題特徵可區分為以下類型：

1. 靜態與動態標靶：欲搜尋之標靶是否隨著時間而移動可區分為靜態標靶與動態標靶。
2. 離散與連續時間：依照時間是否僅在特定離散間隔時刻有意義可區分為離散時間與連續時間。
3. 投入離散與連續之搜尋資源(search effort)：指搜尋期間可投入之資源，離散搜尋資源，例如：離散之時間、投入搜尋的搜救單位；連續搜尋資源，例如：連續之時間、在搜救地區之燃料使用量。

早期之搜救文獻大多源自 Stone (1976)的搜索理論，探討的主要は資源分配

問題，研究每個地區投入之搜尋資源量，而使尋獲標靶之機率能最大化。搜救文獻目標式大多は最大化尋獲標靶之機率，但近年來研究問題從早期的資源分配問題轉變為路徑規劃問題，搜救隊在眾多可選擇之路徑中選擇一條路徑使得尋獲標靶之機率最大或是期望時間最小，依照標靶移動狀態，可將問題分為兩類：

### 2.1.1 搜尋動態標靶

動態標靶會隨著時間而移動，然而標靶所在真實的位置並無法得知，Royset and Sato (2010)提出兩種方法描述標靶的移動，分別是透過蒙地卡羅方法從標靶運動模型進行抽樣，產生多條標靶可能移動之路徑與標靶選擇該條路徑之機率，或是利用馬可夫轉移矩陣，來求得每一期間標靶可能存在網格之機率，並依據上述之方法，建構數學模型以找出一或多條路徑滿足最小化最大未尋獲標靶之機率。

### 2.1.2 搜尋靜態標靶

靜態標靶並不會隨著時間而移動，假設有一網路  $G = (N, A)$ ，其中  $N$  為節點集合， $A$  為所有無向節線的集合，已知標靶出現在節線  $(i, j) \in A$  之機率密度函數 (p.d.f) 為  $\alpha_{ij}$ ，目標是找出一條路徑，而使整體尋獲標靶之機率最大或是期望搜尋時間最短。

Berger and Lo (2015) 探討多搜救隊於單位節線長度網路(unit graph)，亦即所有節線的長度皆為單位長度)搜尋靜態標靶，目標式為最大化尋獲標靶之機率，該論文考慮當標靶於網格  $c$  中，搜救隊存在機率無法尋獲標靶(即  $\text{Pr}(\text{搜救隊於網格 } c \text{ 尋獲標靶} | \text{標靶位於網格 } c) \neq 1$ )，且誤警率(false-alarm)為 0 (即  $\text{Pr}(\text{搜救隊於網格 } c \text{ 尋獲標靶} | \text{標靶並不位於網格 } c) = 0$ )。此外，由於數學模型無法處理較大之期間  $T$ ，僅可處理  $T \leq 10$ ，因此 Berger and Lo (2015) 將最大期間  $T$  切割為較小之期間  $\Delta T$ ，每個期間  $\Delta T$  只決定單期之移動，縮小問題之規模。

儘管傳統人力徒步搜救路線規劃的相關文獻目標式大多是最化尋獲標靶之機率，最小化期望搜尋時間的重要性也日益提升。舉例來說，搜救隊尋找受難人員時，需要把握黃金 72 小時，愈快找到受難人員，其生存機率愈高。Jotshi and Batta (2008)針對單一搜救隊搜尋靜態標靶，以最小化期望搜尋時間為目標，發展兩種啟發示演算法，試圖找出一條路徑經過所有節線。此篇提出期望搜尋時間之計算方式，式(2.1.1)代表尋獲標靶之機率，其中  $i(P)$  代表搜尋路徑  $P$  之第  $i$  條線段， $l_{i(P)}$  為該線段之長度， $L = \sum_{i=1}^{|E|} l_{i(P)}$  為搜尋路徑  $P$  之總長度， $L_{i(P)}$  為包含搜尋路徑  $P$  之前  $i-1$  條線段之集合，其中  $L_{1(P)} = \Phi$ ，當  $i(P) \in L_{i(P)}$  代表搜救隊並非第一次經過此線段，因此在該線段上發現標靶之機率為 0；反之，由於此論文假設標靶均勻分布於所有節線，因此發現標靶之機率為  $l_{i(p)} / L$ 。

$$\Pr(i(P)) = \begin{cases} 0 & \text{if edge } i(P) \in L_i(P) \\ \frac{l_{i(P)}}{L} & \text{otherwise} \end{cases} \quad (2.1.1)$$

$$E[T_s | P] = \sum_{i=1}^{|E'(P)|} \left( \sum_{j=1}^{i-1} l_{j(P)} + \frac{l_{i(P)}}{2} \right) * \Pr(i(P)) \quad (2.1.2)$$

期望搜尋時間之計算如式(2.1.2)，假設在節線上尋獲標靶之機率為均勻分佈， $l_{i(p)} / 2$  為期望尋標靶之位置，由起點出發至尋獲標靶之距離乘上發現目標機率之加總即為期望搜尋時間。

Yu and Batta (2010) 探討單一搜救隊於單位節線長度網路搜尋靜態標靶，以尤拉路徑(Eulerian path)之概念提出一混整數規劃模式與 SIEUG 演算法，並利用求解器 CPLEX 求解，當節線數小於 10 時 CPLEX 可在 30 分鐘內求得最佳解，若利用演算法求解，演算法可在 30 分鐘內處理節線數至多 85 並提供一可行解。Moskal (2013) 探討單一搜救隊針對不同型式之網路圖(一般節線長度網路與單位節線長度網路)與有向無向節線提出六種數學模型，儘管此論文概念是最小化

期望搜尋時間，數學模型則是以獎金收集節線排程問題(Prize-collecting Arc Routing Problem)為基礎，目標為愈快經過未曾經過之節線得到的獎勵愈多。

Li and Huang (2018) 探討多搜救隊在路徑上限  $l$  下，於位節線長度網路網路搜尋靜態標靶，作者針對多搜救隊提出新的期望搜尋時間的計算方式，如式(2.1.3)， $\alpha_l$  為第  $l$  期末尋獲標靶之機率，考量尚未被經過之節線同時被兩組搜救隊由不同方向進行搜救時，Li and Huang (2018) 證明當節線同時被兩組搜救隊搜救與節線僅被一組搜救隊搜救期望搜尋時間之差額為 0.25，由於差距極小可忽略不計。此外，單一搜救隊搜尋均勻分佈於網路之標靶，當網路結構為樹狀時，Li and Huang (2018)證明在給定任一個起點下，深度優先搜尋(Depth-First Search)演算法所得之路徑將會是最佳路徑，且最佳的起點位於葉節點。

上述三篇以期望搜尋時間最短為目標之論文皆是以層空(Level-Space)為架構建模，然而此種建模方式在求解模型前需先決定整段路徑經過之總階層數  $l$ ，由於最佳路徑經過之總階層數  $L^*$  未知，若  $L$  設定的太小，則模型可能無解；若設定的太大，則模型需要花費大量時間求解，因此設定好的  $L$  值極為重要。若以單搜救隊進行搜救，Yu and Batta (2010) 證明  $L$  之下界不小於最短尤拉路徑(shortest Eularian path)，而上界則需依據網路圖中奇點(odd vertex)之個數的不同，而使用不同之演算法計算；若以多搜救隊進行搜救時，Li and Huang (2018) 將  $L$  設為式(2.1.3)，其中  $L_{lb}$  是求解 Ahr and Reinelt (2002) 所提出的多郵差問題(Min-Max k-Chinese Postman Problem)之整數規劃的 LP 寬鬆式而得。

$$E[T_s(S(t))] = \sum_{t=1}^T (\alpha_{l-1} + \alpha_l) \frac{1}{2} \quad (2.1.3)$$

$$L = 2L_{lb} \quad (2.1.4)$$

表 2.1 傳統人力徒步搜救路線規劃文獻比較

作者	多搜救隊	期望搜尋 時間最小	靜態標靶分布		一般節線 長度網路
			均勻	隨機	
Berger and Lo (2015)	V			V	
Joshi and Batta (2008)		V	V		V
Yu and Batta (2010)		V	V		
Moskal (2013)			V		V
Li and Huang (2018)	V	V		V	
本研究	V	V		V	V

註：隨機代表的意思是每條節線皆是均勻分布，但是不同節線發現標靶之機率有高低之分，並未限制所有節線發現標靶之機率皆與節線之長度有正相關。

表 2.1 為相關文獻的整理，目前尚未有文獻探討靜態標靶隨機分布於一般節線長度網路，然而現實之情況，靜態標靶出現在任一條節線之機率與節線之長度可能都不盡相同，且以最小化期望時間為目標之傳統人力徒步搜救路線規劃的相關文獻中，少有多搜救隊之研究。本研究結合上述特點，將問題更加貼近實際之情形。

## 2.2 無人機搜救路線規劃相關研究

目前已有文獻探討無人機應用在搜索與救援的領域中，無人機可以在災害發生後短時間內快速佈署，立即進入災區救援並且蒐集資訊，然而，無人機的移動必需倚賴電力，同時移動之速率也會對電量的消耗有所影響，Kemper *et al.* (2010)針對無人機電力的補充提出充電與換電池兩種方式，由於搜救必須與時間賽跑，利用換電池此種方式可在短時間內讓無人機重新投入搜救行動中，較為合理。

Yao *et al.* (2017)認為無人機搜救策略可分成搜救隊對於搜救區域「不具備先備知識」與「具備先備知識」等兩種：

(1) 不具備先備知識：標靶可能出現在搜救區域內的任一處，搜救團隊必須規劃無人機之搜尋路徑，以使裝配在無人機上的偵測器涵蓋整個搜救區域，如：

地震或是土石流發生後，原有地形可能受到影響，原先對於該地區的認識可能變得不一樣。

(2) 具備先備知識：搜救隊已知標靶有可能出現在某些區域，搜救團隊必須規劃路徑尋訪所有可能存在標靶區域，如：發生山難時，當地人可能知道某些區域較容易發生意外，存在較高的機率發現受難人員。

本研究探討的問題係屬具備先備知識，已知標靶出現在節線上之機率，無人機必須尋訪所有可能出現標靶之節線，與大多數具先備知識搜救問題之求解方法較為類似。

Forsmo *et al.* (2013)考慮不具備先備知識無人機群進行搜救，無人機可自任一處起點出發，但每架無人機之起訖點須相同，無人機上架設偵測器，可偵測半徑 $r$ 範圍內之物體，任兩無人機不可太過接近，否則偵測範圍重疊過多導致效益太差。該論文首先利用演算法產生航點(Waypoint)，接著建立一混整數規劃，目標為無人機群必需經過部分航點，且最小化尋訪時間。

Raap *et al.* (2017)考慮具先備知識之固定翼無人機進行大範圍的搜救，以空難為例，當無人機在海上進行搜救時，若只是涵蓋整個搜救區域並不夠，因為標靶可能會隨著海流而移動。該論文以馬可夫轉移矩陣建立標靶可能之移動路徑，並考慮固定翼無人機不可盤旋或是進行大角度的轉彎等限制，建立二元整數規劃，規劃路徑以使尋獲標靶之機率最大。

Li *et al.* (2018)探討無人機在搜救的過程中電力可能耗盡，需進行充換電，已知建立充換電站之個數，考慮所有無人機的起點須相同，訖點可不同之條件下，決定無人機群的路徑規劃與充換電站的位置，需注意的是每個節點間存在重要性之差別，需優先尋訪重要節點(無人機可直接由起點移動至重要任務)，每個節點皆需尋訪，該論文以基因演算法(Genetic Algorithm)進行求解，染色體由節點組成，每架無人機服務均等數量的節點，目標為極小化完成時間、完成重要節點尋訪時間、違反續航力限制與前往充電距離之加權總合。

表 2.2 無人機搜救路線規劃相關文獻比較

作者	多無人機	期望時間	先備知識	電力	速率相同	標靶位址
Forsmo <i>et al.</i> (2013)	V				V	節點
Raap <i>et al.</i> (2017)	V		V		V	節點
Li <i>et al.</i> (2018)	V		V	V	V	節點
本研究	V	V	V	V		節線

回顧上述文獻後，可發現 Li *et al.* (2018)考慮無人機群需充電、每個節點皆須尋訪且無人機可於任兩個節點間移動與本研究之設定類似；該論文與本研究不同之處，在於其標靶出現在節點上，本研究則是設定在節線上，且本研究考慮不同的起訖組合、無人機有不同之移動模式，目標則是最小化期望搜尋時間。

### 2.3 節線排程問題相關文獻

節線排程問題(Arc Routing Problem, ARP)是探討給定網路圖  $G = (N, A)$ ，存在某些節線  $A^s \subseteq A$  必須經過或是被服務，而經過或服務每條節線都存在相對應之成本，目標為從起點出發找出一條路徑包含所有節線  $a \in A^s$  之路徑並回到起點之封閉路徑，而使總成本最低。當所有節線都須經過時，此問題為著名的中國郵差問題(Chinese Postman Problem, CPP)；若僅須經過部分節線，則此問題為鄉村郵差問題(Rural Postman Problem, RPP)。本研究欲指派  $K$  組無人機進行搜救，給定網路圖  $G = (N, A)$ ，考慮無人機可以在任兩點所構成之節線  $A' \subseteq A$  間移動，無人機移動之網路可視為完全圖，而網路圖中存在某些節線  $A^s \subseteq A$  必需經過，此與求解多組郵差必須經過某些節線的鄉村多郵差問題(KRPP)本質較為相同(但目標不同)。KRPP 可視為單一( $k=1$ )鄉村郵差問題的廣義版本。

鄉村郵差問題最早由Orloff (1974)提出，現實生活中較少案例需要服務網路圖中所有的節線，只需要服務特定節線的鄉村郵差問題更為貼近現實。Lenstra and Kan (1976)證明鄉村郵差問題在給定的無向圖中必須被服務的節線都相連的

情況下可能可以在多項式時間內求解，否則鄉村郵差問題至少為 NP-hard 的問題，而進一步延伸的鄉村多郵差問題也同樣為困難問題。

Aráoz *et al.* (2006) 提出私有化鄉村型郵差問題(Privatized rural postman problem, PRPP)，給定起點之無向網路圖，車輛經過任一條節線必須支付相對應的成本，反之，服務該節線也可得到利潤，然而此利潤僅會在第一次服務時獲得，之後再次經過便不存在，但通行之成本仍須再次支付。目標是最大化整趟迴路(由起點出發再回到起點)之效益(總利潤-總成本)，依此提出多種數學模型，該論文除了證明PRPP為NP-hard問題，也證明多種性質，有效縮減限制式之數量。

Benavent *et al.* (2009)考量多車輛之風向鄉村型郵差問題(MM K-WRPP)，目標為最小化所有車輛中通行時間最長的車輛通行時間，該論文提出一數學模型，並結合多起點演算法(Multi-start algorithm)、變動鄰域尋優法(Variable Neighborhood Descent)以及迭代區域搜尋法(Iterated Local Search)發展啟發式演算法以求解大規模之MM K-WRPP問題。

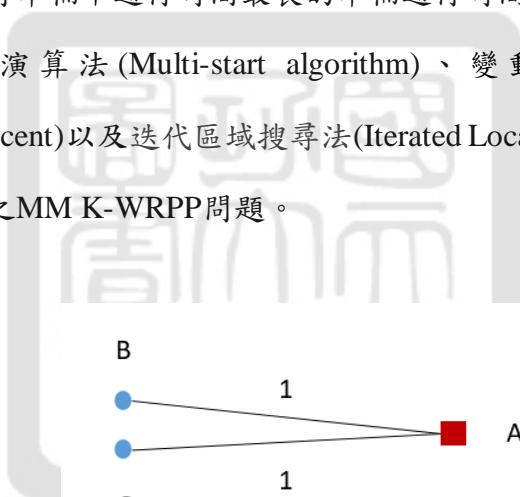


圖 2.1 待搜尋網路圖三

Campbell *et al.* (2018) 使用無人機進行節線排程問題，該論文提出三個假設：(1)當無人機服務完某線段後，無人機可直接前往任一節線進行服務、(2)無人機可以在節線之任一處開始或結束服務、(3)節線可為任一形狀並不一定為直線。該論文提出無人機的使用將大幅減少成本與時間之花費，如下圖2.1所示，給定網路圖，其中長度為1之節線AB、AC需被經過，節點B與節點C之距離為極小值 $\varepsilon$ ，但彼此並不相連，從A點出發，若為中國郵差問題，則此問題之最佳距離為4；若使用無人機，則距離將為 $2 + \varepsilon$ ，兩者距離差距近1倍。在演算法的部分，Campbell

*et al.* (2018) 首先將不一定要經過之節線進行縮減，若節線 $(i, j)$ 滿足 $c_{ij} \approx c_{ik} + c_{kj}$ 則進行縮減，接下來利用分支切割演算法(Branch-and-Cut Algorithm)進行求解。

## 2.4 小結

本章節針對以期望搜尋時間為目標式以及無人機相關之搜索與救援文獻進行討論，可以發現大多數以無人機進行搜救之文獻皆假設僅可在節點尋獲標靶，並未探討當標靶存在節線之情況，且較少文獻考慮無人機的充換電與移動速率等問題。以最小化期望搜尋時間為目標之文獻大多針對單位節線長度網路與標靶出現在任一條節線之機率皆相同等特殊情況進行討論，然而本研究考慮無人機充換電、節線長度與發現標靶機率可不相同等更為一般性之情況。在數學模式的文獻探討中多以層空網路來建構模式，並探討最佳階層數之設定。最後介紹網路形式與本研究相似的節線排程問題，儘管其目標式與本篇並不相同。下一章將針對本研究所探討之問題提出混整數規劃模式。

### 第三章 無人機群搜尋路徑之數學規劃模型

本章首先探討無人機群搜尋路線規劃問題及其假設，接著說明如何以期望搜尋時間為目標，應用在數學模型中。為使建立數學模型更為容易，我們改變原問題之網路結構，在已知充換電站基地節點下，我們提出以層空網路為概念之混整數規劃模式  $M_L^{C_0}$ ，並考量不同起訖組合之情況以及最佳階層數之設定。同時，我們也提出在已知設立充換電站數之條件下，求解最佳之充換電站設立位址與無機群移動路線之數學模型(詳細問題內容與數學模型置於附錄)。

#### 3.1 問題描述

假設有一無向網路  $G = (N, A)$ ， $N$  為所有節點的集合， $A$  為所有無向節線之集合， $A_r \subseteq A$  為必須經過至少一次之節線集合，已知  $D_{ij}$  為節線  $(i, j)$  之長度， $\alpha_{ij}$  為節線  $(i, j) \in A_r$  上尋獲標靶之機率與充換電站基地之節點  $V_c$ 。假設有  $K$  架相同的無人機負責進行搜尋單一靜態標靶，無人機存在續航力上限  $P_{\max}$ ，當續航力不足以服務下一條節線時，無人機必須到充換電站之節點進行充換電，並等待一段時間才可再次滿電出發。假設無人機的速率存在兩種移動模式，分別為慢速的「搜尋模式」，以及快速的「移動模式」；當無人機經過節線並進行搜尋時為搜尋模式，速率為 1；反之，若經過節線時並未進行搜尋則為移動模式，速率為常數  $v_r$  且  $v_r \geq 1$ 。電力消耗與移動速率成正比，移動模式之電力消耗速率為搜尋模式之  $v_r$  倍。本研究之目標為派遣  $K$  架無人機尋訪所有可能存在標靶之節線以使尋獲標靶之期望搜尋時間最小化。

以圖 3.1 為例說明兩架無人機移動之情形：實線為可能存在標靶之節線，由於無人機可於任兩節點間移動，因此網路為完全圖。由於節線數目過多，因此並未呈現在圖 3.1 中，僅呈現可能存在目標之節線。假設移動模式移動速率  $v_r$  為 1.25，

最大電量可飛行 5 單位距離，耗電量假設為移動速率乘以移動距離，若以兩架無人機尋訪此網路，起點皆由  $F$  點出發，無人機 A 首先以搜尋模式尋訪節線  $(F, D)$ ，經過 1 單位時間後，無人機抵達  $D$  點，電力降為  $5 - 1 = 4$ ；由於節線  $(F, D)$  已被無人機 A 以搜尋模式尋訪，因此無人機 B 必須以移動模式拜訪節線  $(F, D)$ ，經過  $1 / 1.25 = 0.8$  單位時間抵達  $D$  點，電力降為  $5 - 1 * 1.25 = 3.75$ 。

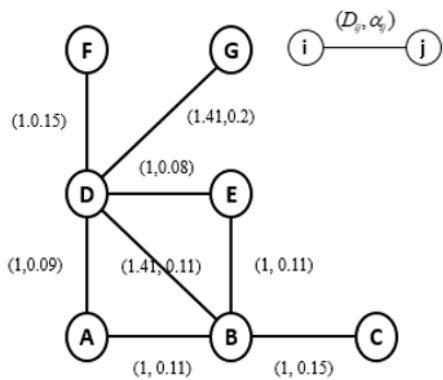


圖 3.1 待搜尋網路圖

由上述例子可以發現，無人機可能的路徑不勝枚舉，往往須考慮當下電力狀況、充換電站之節點所在、節線長度與尋獲標靶之機率等，十分複雜。而本研究即是決定節線尋訪順序，以及何時前往充換電站，此為本研究欲求解的排程決策。

### 3.2 問題假設

為了適當簡化數學模式與模型的使用限制，本研究有以下基本假設：

1. 無人機於節線上進行尋訪時，若偵測到靜態標靶，代表無人機尋獲標靶，不存在誤警(false-alarm)之情形(即偵測到標靶但標靶不存在於節線)。
2. 搜救具備先備知識，已知各節線上尋獲靜態標靶之機率，且所有節線上尋獲標靶之機率加總為 1。
3. 靜態標靶設定在節線上，而非節點上，且標靶出現在任一節線之位置服從均勻分佈。
4. 假設每架無人機最大電力與各移動模式之速率相同，且搜救能力與效率一致。

5. 假設每架無人機在節線上進行搜尋時，一定會將目前的工作完成後才會進行下一個工作，不會有工作到一半中止去尋訪其他節線之情況(即 non-preemptive)。
6. 無人機於充換電站補充電力需等待一段固定時間  $T_c$ ，才可再次滿電出發。

### 3.3 網路結構

本研究旨在了解每架無人機尋訪路徑之節線順序與移動模式，由於原網路結構僅能了解路徑上節線尋訪的順序，無法判斷經過節線時的移動模式，如圖 3.2，若給定一路徑 ABCBD，無法從圖 3.2 得知經過節線時的移動模式。若針對兩種移動模式，將節線分為搜尋節線與移動節線，則可判斷經過節線時的移動模式，如圖 3.3，無人機以搜尋模式經過搜尋節線，以移動模式經過移動節線。原問題將轉變為每條搜尋節線只能被經過一次，移動節線則可被多次經過。

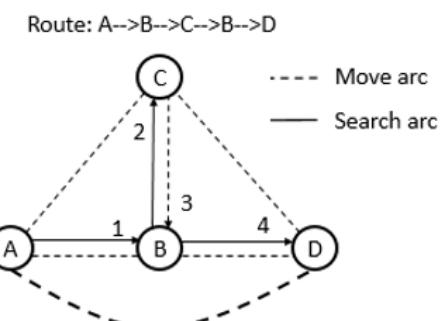
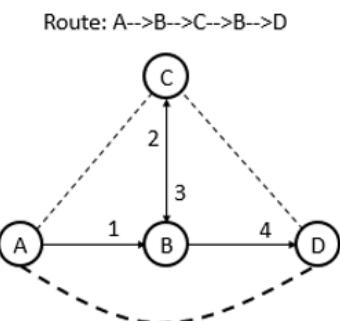


圖 3.2 原問題之網路

圖 3.3 以搜尋節線與移動節線建構之網路

### 3.4 固定充定站位置之數學模式 $M_L^{C_0}$

本研究以層空網路建構數學模式  $M_L^{C_0}$  (上標  $C_0$  代表固定充換電站位置，L 代表 Level-Space)，其單一無人機之層空網路圖如圖 3.4 所示， $o$  為虛擬節點， $d$  為虛擬訖點，虛線為移動節線，任兩節點間皆有一移動節線；實線為搜尋節線，在此圖中無向節線 (1,2) 為搜尋節線，因此節線 (1,2) 與 (2,1) 其中一條節線必須被經

過；若將無人機的數量增加，則多無人機之層空網路圖如圖 3.5 所示，每一個層空都代表一架無人機可能飛行之網路，共  $K$  層，無人機由  $o$  出發回到  $d$ 。介紹完網路圖，接下來 3.4.1 節介紹符號定義，3.4.2 節說明期望搜尋時間的計算方式，3.4.3 節說明本研究之網路結構圖，3.4.4 節說明數學模式，3.4.5 節考慮無人機群起訖組合模式。

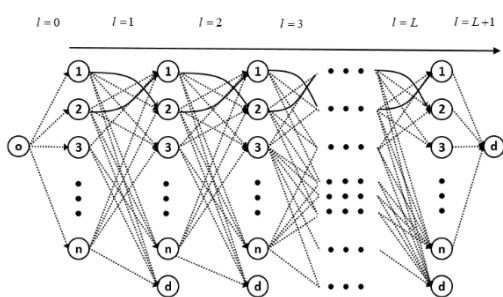


圖 3.4 本研究單一無人機  
之層空網路圖

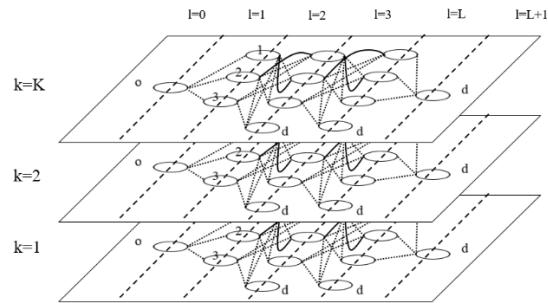


圖 3.5 本研究多無人機之  
層空網路圖

### 3.4.1 符號定義

#### 集合

$A$

所有節線之集合

$A^s$

搜尋節線集合， $A^s \subseteq A$

$A_h^s$

搜尋節線集合，節線  $(i, j)$  滿足  $i < j$ ， $A_h^s \subseteq A^s$

$A'$

移動節線集合， $i, j \in V$ ， $i \neq j$ ， $(i, j) = a \in A' \subseteq A$

$V$

所有節點之集合(包含虛擬起訖點  $o$ 、 $d$ )

$V_c$

所有充換電站點之集合， $V_c \subseteq V$

#### 參數

$L$

最大階層數

$K$

無人機的數量

$T_c$

充換電池所需花費的時間

$P_{\max}$	無人機最大電量
$D_a$	節線 $a \in A$ 的長度
$\alpha_a$	在搜尋節線 $a \in A^s$ 尋獲標靶之機率
$v_t$	移動模式之速率，假設搜尋模式之速率為 1
$M$	極大的數
$\varepsilon$	極小的數

註： $i, j \in N$ ， $i \neq j$ ， $(i, j) = a \in A$ ，定義符號  $\bar{a} = (j, i)$ ， $tail(a) = i$ ， $head(a) = j$ ，其中  $a$ 、 $\bar{a}$  代表同一節線不同方向。

### 變數

$x_a^{l,k} \in \{0,1\}$	無人機 $k$ 在第 $l$ 層經過搜尋節線 $a \in A^s$ 為 1；反之為 0
$y_a^{l,k} \in \{0,1\}$	無人機 $k$ 在第 $l$ 層經過移尋節線 $a \in A^t$ 為 1；反之為 0
$t_{l,k}$	無人機 $k$ 經過第 $l$ 層的時刻
$t_a^s$	無人機經過搜尋節線 $a \in A^s$ 的時刻
$p_{l,k}$	無人機 $k$ 經過第 $l$ 層後剩餘之電量
$rc_i^{l,k} \in \{0,1\}$	無人機 $k$ 經過第 $l$ 層抵達節點 $i$ 進行充電為 1；反之為 0

#### 3.4.2 期望搜尋時間

首先說明 Joshi and Batta (2008)提出期望搜尋時間之計算方式，接著提出本研究的期望搜尋時間的計算方式：

#### Joshi and Batta (2008)

式(3.4.1)代表尋獲標靶之機率，其中  $i(P)$  代表搜尋路徑  $P$  之第  $i$  條線段， $l_{i(P)}$

為該線段之長度， $L = \sum_{i=1}^{|E|} l_{i(P)}$  為搜尋路徑  $P$  之總長度， $L_{i(P)}$  為包含搜尋路徑  $P$  之

前  $i - 1$  條線段之集合，其中  $L_{1(P)} = \Phi$ ，當  $i(P) \in L_{i(P)}$  代表搜救隊並非第一次經過此

線段，因此在該線段上尋獲標靶之機率為 0，否則機率為  $\frac{l_{i(P)}}{L}$ 。

$$\Pr(i(P)) = \begin{cases} 0 & \text{if edge } i(P) \in L_i(P) \\ \frac{l_{i(P)}}{L} & \text{otherwise} \end{cases} \quad (3.4.1)$$

$$E[T_s | P] = \sum_{i=1}^{|E'(P)|} \left( \sum_{j=1}^{i-1} l_{j(P)} + \frac{l_{i(P)}}{2} \right) * \Pr(i(P)) \quad (3.4.2)$$

假設移動速率為 1，期望搜尋時間之計算如式(3.4.2)，假設在線段上尋獲標靶之機率為均勻分佈， $l_{i(P)} / 2$  為期望尋獲搜標靶之位置，由起點出發至尋獲標靶之距離乘上尋獲標靶機率之加總即為期望搜尋時間。

## 本研究

由於本研究假設不同搜救節線上發現標靶之機率可為相異，修改式(3.4.1)為式(3.4.3)， $\alpha_{i(P)}$  為節線  $i(P)$  上尋獲標靶之機率(介於 0 與 1 之間且所有節線上之機率加總為 1)。假設同一節線上尋獲標靶之機率仍為均勻分配，由於無人機之移動速率有不同模式，因此將式(3.4.2)的  $l_{i(P)}$  替換為  $t_{i(P)}$ ，如式(3.4.4)，其中  $t_{i(P)}$  為線段  $i(P)$  之移動時間， $\frac{t_{i(P)}}{2}$  為期望尋獲標靶的時間。

$$\Pr(i(P)) = \begin{cases} 0 & \text{if edge } i(P) \in L_i(P) \\ \alpha_{i(P)} & \text{otherwise} \end{cases} \quad (3.4.3)$$

$$E[T_s | P] = \sum_{i=1}^{|E'(P)|} \left( \sum_{j=1}^{i-1} t_{j(P)} + \frac{t_{i(P)}}{2} \right) * \Pr(i(P)) \quad (3.4.4)$$

由於式(3.4.4)中累積距離的變數乘上該節線是否為首次尋訪的變數為非線性，因此將式(3.4.4)進行改寫為式(3.4.5)，紀錄經過搜尋節線的時刻即可不必判

斷是否為首次尋訪， $t_a^s$ 為經過搜尋節線 $a$ 之時刻， $t_a$ 為無人機在搜尋節線 $a$ 之移動時間， $\alpha_a$ 為搜救節線尋獲標靶之機率。

$$E(T | P) = \sum_{a \in A_h^s} (t_a^s - \frac{t_a}{2})\alpha_a \quad (3.4.5)$$

### 3.4.3 數學模式

目標為希望期望搜尋時間愈小愈好，而 $\varepsilon$ 項則是將無人機總飛行距離納入考慮，使無人機在尋訪完節線後不會多繞遠路而直接回到虛擬訖點，故目標式如式(3.4.6)。

$$\text{Minimize} \quad \sum_{a \in A_h^s} (t_a^s - \frac{t_a}{2})\alpha_a + \varepsilon (\sum_{k=1}^K \sum_{l=1}^L \sum_{a \in A^s} x_a^{l,k} D_a + \sum_{k=1}^K \sum_{l=1}^L \sum_{\substack{a \in A^l \\ \text{tail}(a) \neq o \\ \text{head}(a) \neq d}} y_a^{l,k} D_a) \quad (3.4.6)$$

限制式部分，限制式(3.4.7)為流量守恆。

$$\sum_{\substack{a \in A^s \\ \text{tail}(a)=i}} x_a^{l,k} + \sum_{\substack{a \in A^l \\ \text{tail}(a)=i}} y_a^{l,k} = \sum_{\substack{a \in A^s \\ \text{head}(a)=i}} x_a^{l-1,k} + \sum_{\substack{a \in A^l \\ \text{head}(a)=i}} y_a^{l-1,k} \quad \forall l=1,..,L ; k=1,...,K \quad (3.4.7)$$

限制式(3.4.8)限制每架無人機從虛擬起點出發。

$$\sum_{\substack{a \in A^l \\ \text{tail}(a)=0}} y_a^{0,k} = 1 \quad \forall k=1,...,K \quad (3.4.8)$$

限制式(3.4.9)到限制式(3.4.14)為時間相關之限制式：

限制式(3.4.9)限制第0階層的時刻從0開始。

$$t_{0,k} = 0 \quad \forall k=1,...,K \quad (3.4.9)$$

限制式(3.4.10)表示各階層時刻的計算為前一層之時刻加上這一層經過之搜尋節線加上是否至充換電站換電池的時間，由於並無法得知無人機實際尋訪之節線，因此這一層之時刻應為 $t_{l-1,k} + x_a^{l,k} D_a + r c_i^{l,k} T_c$ 各種可能路徑的時刻中最大為此層之時刻。

$$t_{l-1,k} + x_a^{l,k} D_a + r c_j^{l,k} T_c \leq t_{l,k} \quad (3.4.10)$$

$$\forall j \in V \setminus \{o, d\}, (i, j) = a \in A^s; \quad l = 1, \dots, L-1; \quad k = 1, \dots, K$$

限制式(3.4.11)與(3.4.10)類似，考慮無人機尋訪之節線為移動節線之情況。

$$t_{l-1,k} + y_a^{l,k} D_a / v_t + r c_j^{l,k} T_c \leq t_{l,k} \quad (3.4.11)$$

$$\forall j \in V \setminus \{o, d\}, (i, j) = a \in A^t; \quad l = 1, \dots, L-1; \quad k = 1, \dots, K$$

限制式(3.4.12)記錄無人機經過搜尋節線的時刻，若搜尋節線  $a$  被尋訪，經過搜尋節線  $a$  的時刻為該階層之時刻。

$$T_a^s + M (1 - x_a^{l,k}) \geq t_{l-1,k} + x_a^{l,k} D_a \quad \forall a \in A^s; \quad l = 1, \dots, L; \quad k = 1, \dots, K$$

$$(3.4.12)$$

限制式(3.4.13)限制搜尋節線  $a$  與  $\bar{a}$  被尋訪之時刻應相同。

$$T_a^s = T_{\bar{a}}^s \quad \forall a, \bar{a} \in A^s \quad (3.4.13)$$

限制式(3.4.14)到(3.4.20)為電力相關限制式。

限制式(3.4.14)限制無人機在每一階層之電力都需要大於等於 0。

$$p_{l-1,k} - \sum_{a \in A^s} x_a^{l,k} D_a - \sum_{\substack{a \in A^t \\ tail(a) \neq o \\ head(a) \neq d}} y_a^{l,k} D_a v_a \geq 0 \quad \forall l = 1, \dots, L; \quad k = 1, \dots, K \quad (3.4.14)$$

限制式(3.4.15)限制第  $l$  階層的電量需小於等於第  $l-1$  層的電量減去移動所消耗之電力加上是否換新電池之電力，由於無人機可能充換電池，充換完電池後之電量不超過最大電量，因此限制式不可為等式。

$$p_{l-1,k} - \sum_{a \in A^s} x_a^{l,k} D_a - \sum_{\substack{a \in A^t \\ tail(a) \neq o \\ head(a) \neq d}} y_a^{l,k} D_a v_a + P_{\max} \sum_{i \in V \setminus \{o, d\}} r c_i^{l,k} \geq p_{l,k} \quad (3.4.15)$$

$$\forall l = 1, \dots, L; \quad k = 1, \dots, K$$

限制式(3.4.16)與(3.4.17)限制當無人機並未到充換電站更換電池時，電力應服從能量守恆。

$$p_{l-1,k} - \sum_{a \in A^s} x_a^{l,k} D_a - \sum_{\substack{a \in A' \\ tail(a) \neq o \\ head(a) \neq d}} y_a^{l,k} D_a v_a + M \sum_{i \in V / \{o, d\}} r c_i^{l,k} \geq p_{l,k} \quad \forall l=1,..,L ; k=1,...,K \quad (3.4.16)$$

$$p_{l-1,k} - \sum_{a \in A^s} x_a^{l,k} D_a - \sum_{\substack{a \in A' \\ tail(a) \neq o \\ head(a) \neq d}} y_a^{l,k} D_a v_a - M \sum_{i \in V / \{o, d\}} r c_i^{l,k} \leq p_{l,k} \quad \forall l=1,..,L ; k=1,...,K \quad (3.4.17)$$

限制式(3.4.18)說明每階層無人機之電量需小於電池的最大電量。限制式(3.4.19)說明若無人機有到充換電站更換電池，電量需大於等於最大電量。限制式(3.4.20)限制第0階層之電量為最大電量。

$$p_{l,k} \leq P_{\max} \quad \forall l=1,..,L ; k=1,...,K \quad (3.4.18)$$

$$p_{l,k} \geq P_{\max} r c_i^{l,k} \quad \forall l=1,..,L ; k=1,...,K ; i \in V / \{o, d\} \quad (3.4.19)$$

$$p_{0,k} = P_{\max} \quad \forall k=1,...,K \quad (3.4.20)$$

限制式(3.4.21)假如節點*i*並非充換電站，則 $r c_i^{l,k}$ 必為0。

$$r c_i^{l,k} = 0 \quad \forall i \in V / V_c; l=1,..,L; k=1,...,K \quad (3.4.21)$$

限制式(3.4.22)與(3.4.23)假如無人機*k*於階層*l*至充換電站節點*i*換電池，則必有無人機*k*於階層*l*尋訪搜尋節線或是移動節線至節點*i*。

$$\sum_{\substack{a \in A^s \\ head(a)=i}} x_a^{l,k} + \sum_{\substack{a \in A' \\ head(a)=i}} y_a^{l,k} + M (1 - r c_i^{l,k}) \geq 1 \quad \forall i \in V_c; l=1,..,L; k=1,...,K \quad (3.4.22)$$

$$\sum_{\substack{a \in A^s \\ head(a)=i}} x_a^{l,k} + \sum_{\substack{a \in A' \\ head(a)=i}} y_a^{l,k} - M (1 - r c_i^{l,k}) \leq 1 \quad \forall i \in V_c; l=1,..,L; k=1,...,K \quad (3.4.23)$$

限制式(3.4.24)限制每條搜尋節線只會被經過一次。

$$\sum_{k=1}^K \sum_{l=1}^L x_a^{l,k} + \sum_{k=1}^K \sum_{l=1}^L x_{\bar{a}}^{l,k} = 1 \quad a, \bar{a} \in A^s \quad (3.4.24)$$

限制式(3.4.25)限制每架無人機每階層只能至多經過一條節線。

$$\sum_{a \in A^l} x_a^{l,k} + \sum_{a \in A^t} y_a^{l,k} \leq 1 \quad \forall l=1, \dots, L; k=1, \dots, K \quad (3.4.25)$$

### 3.4.4 考慮無人機群起訖組合模式

傳統人力徒步搜救路線規劃相關文獻中可以發現當以人力的方式進行搜救時，只要所有搜尋節線皆被尋訪，搜救行動立即終止，然而在無人機搜救路線規劃相關文獻中，無人機在尋訪完所有搜尋節線後還需要回到原出發點。上述兩種起訖組合皆有其考量，本研究依照不同起訖組合提出相對應之限制式。

(3.4.3)節為單一無人機未固定起訖點之  $M_L^{C_0}$  (即起訖點相異)之情形，若考慮單一無人機起訖點相同，必須加入限制式(3.4.26)與(3.4.27)，若無人機  $k$  由虛擬起點  $o$  至節點  $i$ ，則無人機  $k$  必須從節點  $i$  回到虛擬訖點  $d$ 。

$$\sum_{l=1}^L y_{i,d}^{l,k} + M(1 - y_{o,i}^{0,k}) \geq 1 \quad \forall k=1, \dots, K \quad (3.4.26)$$

$$\sum_{l=1}^L y_{i,d}^{l,k} - M(1 - y_{o,i}^{0,k}) \leq 1 \quad \forall k=1, \dots, K \quad (3.4.27)$$

限制式(3.4.28)限制所有無人機皆從同一起點出發。

$$y_a^{0,k} = y_a^{0,k-1} \quad \forall (o, j)=a \in A^t; k=2, \dots, K \quad (3.4.28)$$

限制式(3.4.29)限制所有無人機皆從同一訖點回到虛擬節點。

$$\sum_{l=1}^L y_a^{l,k} = \sum_{l=1}^L y_a^{l,k-1} \quad \forall (i, d)=a \in A^t; k=2, \dots, K \quad (3.4.29)$$

本研究依照下列 3 種特性區分不同起訖組合，可依照不同的問題情境進行限制式的搭配：

- A. 無人機群相同起點，限制式(3.4.28)
- B. 無人機群相同訖點，限制式(3.4.29)
- C. 單一無人機起訖點相同，限制式(3.4.28)與(3.4.29)

### 3.5 最佳階層數 $L^*$

階層數  $L$  為無人機經過之節線數量，若  $L$  設定太小則模型可能無解或是無法取得全域最佳解，即可能有其他路徑經過之節線數量更多，且期望搜尋時間較小；若  $L$  設定太大，則模型需要耗費大量的時間進行求解。我們希望找到一個最佳階層數  $L^*$  (隨著  $L^*$  的增加，目標值皆不改變)，由於最佳階層數  $L^*$  難以在短時間內求得，因此可以給定一個上下界以使  $L^*$  介於  $[L^{lower}, L^{upper}]$ ，並在上下界搜尋。本研究訂定下界為求解  $K$  組郵差之鄉村郵差問題，即不考慮電力限制下，距離最短的路徑走法；在上界的部分，本研究目前設定為必需經過之節線數量之三倍。

### 3.6 小結

本章節首先討論問題定義與假設，接著說明本研究如何將 Joshi and Batta (2008)推導的期望搜尋時間之公式應用在本研究的數學模型中。由於問題極為複雜，因此以層空網路為架構提出固定充換電站位置之數學模式  $M_L^{C_0}$ ，並針對現實中每架無人機起訖點之各種組合進行討論。但由於混整數規劃變數與限制式過多，無法在中大型網路中，在有效的時間內求得最佳解。因此，下一章節將會針對無人機群路徑規劃問題提出較混整數規劃模式更有效率之數種求解演算法。

## 第四章 無人機群搜尋路徑規劃之求解演算法設計

在第三章以數學模型為基礎並利用求解器求解較大規模之問題，需耗時甚久，因此本章節提出數種啟發式演算法進行求解，啟發式演算法可以在短時間獲得好品質的解，在搜救等與時間賽跑的任務中，較符合現實的需求。4.1 節將針對路徑進行編碼，探討可能的編碼方式；並在 4.2 節解釋路徑解碼的演算法，最後將上述的編碼方式應用在基因演算法(4.3 節)以及區域搜尋法(4.4 節)。

### 4.1 編碼方式(Encode)

在單一無人機的鄉村郵差問題中，Kang and Han (1988)提出一組可行路徑由兩個部分組成，分別是必須尋訪之節線的編號與其方向，如圖 4.1 所示，假設有一節線集合  $E = \{1, 2, 3, 4, 5\}$ ，其中編號 1, 2, 3, 4, 5 分別代表節線  $(a, a')$ ,  $(b, b')$ ,  $(c, c')$ ,  $(d, d')$ ,  $(e, e')$ ，而每條節線有對應的方向，若方向為 0 則代表為正向，方向為 1 則為反向。若依照給定之節線與方向，可求得一條可行的路徑  $(a, a')$ ,  $(b', b)$ ,  $(c, c')$ ,  $(d', d)$ ,  $(e, e')$ ，如圖 4.2。若將問題擴展為多架無人機之問題，Lo *et al.* (2018)增加變數，記錄每架無人機需要負責的節線數量，其結果如圖 4.3 所示，第一架無人機服務 3 條必需經過的節線，第二架無人機服務 2 條必需經經過的節

線，其解空間為  $m!2^m \binom{m-1}{k-1}$ ，其中  $m$  為節線的總數， $k$  為無人機的數量，由於

此種編碼方式解空間較大，因此，本研究提出第二種編碼方式，僅考慮節線尋訪的順序與每架無人機負責的節線數量，如圖 4.4 所示，此種方法的解空間為

$m! \binom{m-1}{k-1}$ ，較前一種編碼方式小，然而編碼並不包含方向的資訊，需要在解碼

(4.2 節)時決定。

節線	1	2	3	4	5
方向	0	1	0	1	0

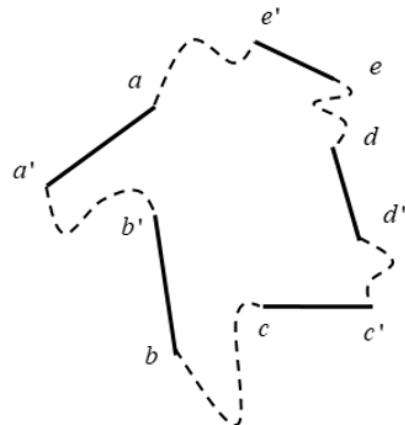


圖 4.1 路徑表示方法

圖 4.2 無人機路徑圖

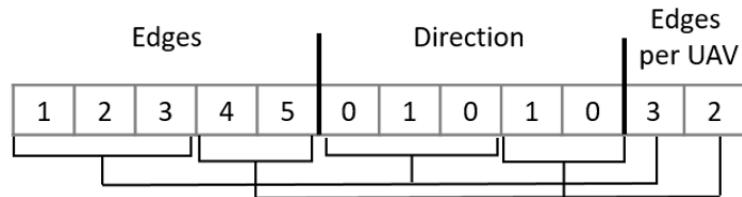


圖 4.3 編碼示意圖(含節線方向)

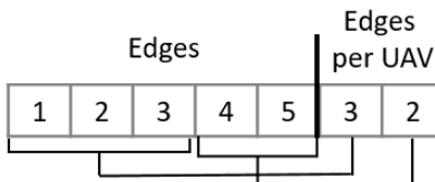


圖 4.4 編碼示意圖(不含節線方向)

## 4.2 解碼演算法(Decoding Algorithm , DA)

在一般途程規劃問題中，Muyldermans *et al.* (2005)針對一條路徑給定尋訪的節線的順序，欲決定每條節線的連接方向，該論文以動態規劃法進行求解，其網路圖如圖 4.5 所示，其中  $\sigma_i$  為路徑的第  $i$  條必需經過之節線， $reverse(\sigma_i)$  為路徑的第  $i$  條節線之反向， $c_{i,j}$  為節點  $i$  到節點  $j$  的距離， $h(\sigma_i)$  為節線的箭頭端， $t(\sigma_i)$  為節線的尾端，目標是找到一條路徑，使得由起點  $\sigma_i$  出發至終點  $\sigma_{i+r}$  距離和最小。上

述的問題並不考慮電量相關的限制，然而在本研究中，若給定一條路徑，除了需要考慮怎麼移動才能在最小期望搜尋時間內完成，也要考慮無人機之電量是否足夠，因此本研究提出結合動態規劃與回溯概念的演算法，求解考慮電力限制下，最佳的節線尋訪方向。

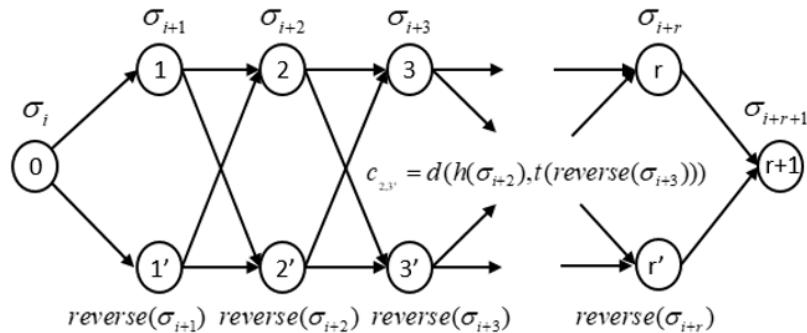


圖 4.5 動態規劃網路圖

#### 4.2.1 演算法步驟

假設給定第  $k$  架無人機的路徑  $S = (\sigma_1, \dots, \sigma_i, \sigma_{i+1}, \dots, \sigma_{i_k})$ ，無人機目前電量  $p_k$ 、位置  $n_k$ 、時間  $t_k$  與累積期望時間  $ET_k$ ，我們將演算法分成兩階段介紹，在「動態規劃階段(DP)」以動態規劃法計算經過每一條節線需要的電量，並產生一組無人機電量足夠服務的路徑  $S_{i,q} = (\sigma_i, \sigma_{i+1}, \dots, \sigma_q)$ ，其中  $i$  為起始路徑的編號、 $q$  為電量足夠服務的最後路徑的編號、 $S_{i,q} \subseteq S$ ，接下來在「回溯階段(BT)」根據前一階段生成的路徑，判斷無人機在服務完節線  $\sigma_q$  後，其剩餘電量是否足夠前往充換電站，若是，則將該充換電站作為下一個起點，將剩餘未被服務的路徑再做一次動態規劃，直到路徑  $S$  上之節線都被服務；若否，則回溯到前一條節線  $\sigma_{q-1}$ ，判斷無人機在服務節線  $\sigma_{q-1}$  後，剩餘電量是否足夠前往充換電站。當路徑  $S$  上之節線都被服務時，即可以得到無人機實際飛行的路徑。

#### 動態規劃階段(Dynamic Programming Phase)

表 4.1 列出動態規劃階段流程之步驟，已知  $G'$  為依照圖 4.5 所建構之網路， $s$  代表節線在路徑中的位置， $rev$  則是該節線的方向，每一段順向或反向節線的電量、時間與期望時間都是由前一條節線所決定。動態規劃階段的目的是求出當無人機經過每一條順向或反向節線時的電量、時間與期望時間，當上述都計算之後，我們即可求得最小期望時間下滿足電量限制的最佳路徑方向。

表 4.1 動態規劃法流程

---

### Dynamic Programming

---

**Data:**  $s, rev, p_k, n_k, t_k, ET_k, T_{status}, P_{status}, ET_{status}$

1: **function** DP( $s, rev, n_k, t_k, ET_k, T_{status}, P_{status}, ET_{status}$ )

2:   **local variables:** ET, ET'

3:   if  $s = 0$  then

4:        $ET_{status}[s, rev] = ET_k, T_{status}[s, rev] = T_k, P_{status}[s, rev] = 0$     (1)

5:       return  $ET_k$

6:   else

7:        $ET \leftarrow DP(s - 1, 0, n_k, t_k, ET_k) + calET(s - 1, 0, s, rev)$     (2)

8:        $ET' \leftarrow DP(s - 1, 1, n_k, t_k, ET_k) + calET(s - 1, 1, s, rev)$     (3)

9:       Update  $ET_{status}, T_{status}, P_{status}$     (4)

10:      return  $\min(ET, ET')$     (5)

**Result:**  $ET_{status}, T_{status}, P_{status}$

---

(1) 初始化:針對  $s = 0$  進行基本資訊的初始化。

(2) 計算服務完第  $s$  條節線方向為  $rev$  之期望搜尋時間：服務完第  $s$  條節線方向為  $rev$  之期望搜尋時間應為服務完第  $s - 1$  條順向(即  $rev = 0$ )節線之期望搜尋時間，加上服務第  $s$  條節線方向為  $rev$  所需之期望搜尋時間。

(3) 計算服務完第  $s$  條節線方向為  $rev$  之期望搜尋時間：服務完第  $s$  條節線方向為  $rev$  之期望搜尋時間應為服務完第  $s - 1$  條反向(即  $rev = 1$ )節線之期望搜尋時間，加上服務第  $s$  條節線方向為  $rev$  所需之期望搜尋時間。

(4) 更新資訊:紀錄第  $s$  條節線方向為  $rev$  之期望搜尋時間、當下時間與累積電量的資訊。

(5) 回傳最佳的期望搜尋時間:選擇(2)與(3)中,期望搜尋時間較小者作為期望搜尋時間。

### 回溯階段(Back Tracking Phase)

在上一階段,由於已知服務每一節線所需要的累積電量,因此可以找到無人機電量足夠服務的路徑  $S_{i,q} = (\sigma_i, \sigma_{i+1}, \dots, \sigma_q)$ , 依照路徑可求得服務完每條節線所需的電量  $P_{stage}$ 、時間  $T_{stage}$  與期望時間  $ET_{stage}$ , 表 4.2 列出回溯階段流程之步驟。

回溯階段目標是找到一條電量足夠回到充換電站的路徑,當回溯法執行完,可以得到無人機包含充換電站的路徑。

表 4.2 回溯法流程

---

#### Back Tracking

---

**Data:**  $G', S_{i,q}, ET_{stage}, T_{stage}, P_{stage}, V_c$

1: **function** BT( $S_{i,q}, ET_{stage}, T_{stage}, P_{stage}, p_k, V_c$ )  
 2:   **local variables:**  
 $G', currentTime, currentLoc, currentET, currentPower, curIdx, chg$   
 3:    $curIdx \leftarrow q - i$   
 4:    $S_{i,q}^{best} \leftarrow findPath(S_{i,q})$  (1)  
 5:    $currentPower \leftarrow p_k - P_{stage}[curIdx]$   
 6:    $currentTime \leftarrow T_{stage}[curIdx]$   
 7:    $currentLoc \leftarrow head(S_{i,q}^{best}[curIdx])$   
 8:    $nextLoc \leftarrow head(S[q+1])$   
 9:    $currentET \leftarrow ET_{stage}[curIdx]$   
 10:    $chg = chgbtwNodes(currentLoc, nextLoc, currentPower)$  (2)  
 11:   **if** cannot reach  $chg$  **then**  
 12:     **if**  $len(S_{i,q}) - 1 = 0$  **then**  
 13:        $G' = constructGraph(currentLoc, nextLoc, V_c)$  (3)  
 14:       Dijkstra( $G'$ )  
 15:       **if** cannot find a path from  $currentLoc$  to  $nextLoc$  **then** (5)  
 16:          $currentET += penalty$   
 17:          $pathOpt(nextLoc, S_{i+1,q}, P_{max}, currentTime, currentET)$  (6)  
 18:     **else**  
 19:       update  $currentPower, currentTime, currentLoc, currentET$   
 20:        $pathOpt(currentLoc, S_{i+1,q}, currentTime, currentET)$

---

---

```

21:     else
22:         BT( $S_{i,q-1}, ET_{stage}, T_{stage}, P_{stage}, p_k, V_c$ )           (7)
23:     else
24:         update  $currentPower, currentTime, currentLoc$ 
25:         pathOpt( $currentLoc, S_{q+1,n_k}, currentTime, currentET$ )
26: end function

```

---

(1) 依照 DP 的運算結果，計算最佳的路徑  $S_{i,q}$  連接方式。

由於當路徑含有的節線數不同時，會有不一樣最短路徑的走法，因此，每當路徑的數量有變動時，就必需重新計算最佳的走法。

(2) 尋找距離兩點最近且電量可到達之充換電站。

(3) 建立網路圖

建立網路圖  $G'$ ，包含  $currentLoc$ 、 $nextLoc$  與所有的充換電站點，儘管無人機可能無法一次性移動超過其電量的距離，但若是透過充換電站，則有可能透過不斷充電到達。

(4) 計算從  $currentLoc$  至  $nextLoc$  的距離與最短的路徑。

此部分可以使用不一樣的一對多的最短路徑演算法，網路圖中的節線都要滿足無人機自離節線最近的充換電站點出發足夠服務服務，本研究中使用 Dijkstra 演算法進行實作。

(5) 假若找不到一條由  $currentLoc$  至  $nextLoc$  的最短路徑，則代表違反電量限制，因此給予期望搜尋時間一個懲罰值。

(6) 將剩餘的節線再做一次 DP 與 BT 演算法。

由於已找到一條完整的路徑，因此將剩餘仍未被最佳化之路徑進行最佳化。

(7) 將原路徑  $S_{i,q}$  改為  $S_{i,q-1}$ ，再次呼叫 BT 演算法。

假如電量不足夠到最近的充換電站，則判斷前一條節線剩餘的電量是否足夠到達最近的充換電站。

#### 4.2.2 範例說明

假設有一網格網路圖如圖 4.6 左，實線節線長度均為 1，且節線上存在發現標靶之機率，為方便程式說明，我們以編節線號代替節線來說明，其對應如圖 4.6 右。在此若給定未給定方向之路徑  $S=(1,4,2,6,5,3)$ 、無人機當下電量  $p_k = 7$ 、無人機最大電量  $P_{\max} = 7$ 、無人機所在節點  $n_k = 1$ 、無人機當下時間  $t_k = 0$ 、無人機當下期望時間  $ET_k = 0$ 。

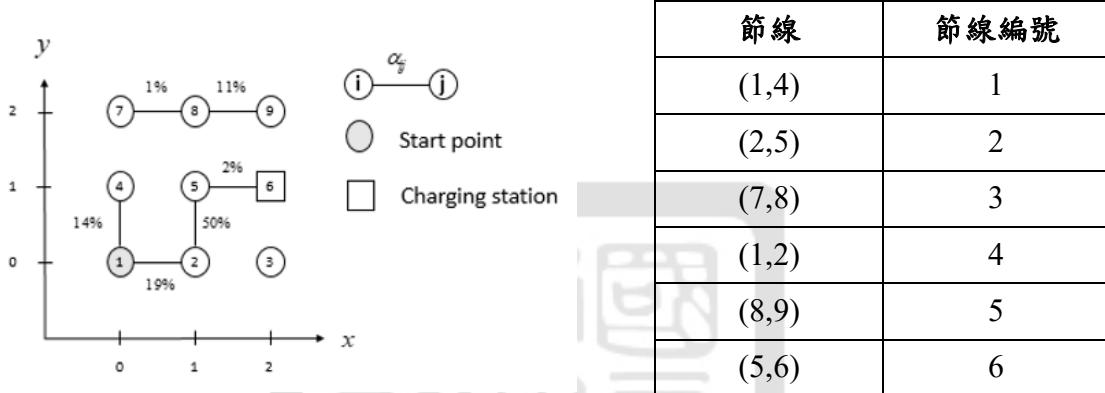


圖 4.6 DA 範例說明網路圖

#### 第一次迭代

##### (1) 動態規劃階段

首先依照未給定方向的路徑  $S=(1,4,2,6,5,3)$ ，可以建立網路圖  $G'$  如圖 4.7，其中  $O$  為已知的起點， $D$  為距離最後最近之充換電站，並以最小期望搜尋時間為目標計算動態規劃，在圖  $G'$  中的每一個節點都會對應其期望搜尋時間與累積電量，我們比較節點可以找到無人機電量最多能服務到的範圍，表 4.3 列出經過每個節點時的期望搜尋時間與電量，由於  $p_k = 7$  所以電量最多能服務 4 條節線，即  $S_{1,4} = (1, 4, 2, 6)$ 。

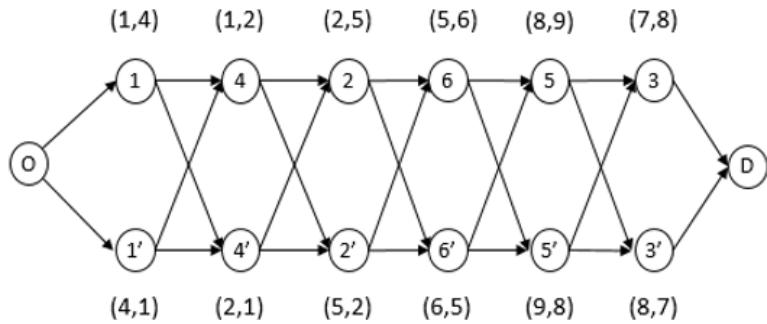


圖 4.7 第一次迭代 DP 網路圖

表 4.3 第一次迭代之 DP 求出之線段服務完之期望搜尋時間與電量

<b>Node</b>	<i>O</i>	<b>1</b>	<b>4</b>	<b>2</b>	<b>6</b>	<b>5</b>	<b>3</b>
<b>ET</b>	0	0.07	0.53	2.19	2.30	3.01	3.10
<b>P</b>	0	1	3.25	4.25	5.25	8.01	9.75
<b>Node</b>	<i>O</i>	<b>1'</b>	<b>4'</b>	<b>2'</b>	<b>6'</b>	<b>5'</b>	<b>3'</b>
<b>ET</b>	0	0.18	0.59	3.59	2.32	2.98	3.09
<b>P</b>	0	2.25	3.76	5.5	6.5	7.5	8.5

## (2) 回溯階段

由上一步驟，我們知道電量足夠服務完前 4 條節線，並計算最佳的路徑為  $S_{1,4}^{best}$

$= (1,2,4,6)$ ，接著計算無人機服務完路徑  $S_{1,4}^{best}$  所有節線後剩餘的電量

$currentPower = 7 - 5.25 = 1.75$ 、目前位置  $currentLoc = 6$ 、目前位置距離下一條

節線最近的節點  $nextLoc = 9$  與距離  $currentLoc$  與  $nextLoc$  最近的充換電站

$chg = 6$ ，由於目前電量足夠到達  $chg$ ，因此更新變數：

$currentLoc \leftarrow chg$

$currentPower \leftarrow 7$

$currentT \leftarrow 4.8 + \text{充電需要的時間}$

並呼叫函式  $pathOpt$  將剩餘的路徑  $S_{5,6} = (5,3)$  進行方向的最佳化。

## 第二次迭代

### (1) 動態規劃階段

將剩餘未被服務的路徑  $S_{5,6} = (5,3)$  建立網路圖，如圖 4.8，以動態規劃求解並可得到表 4.4，電量足以服務剩餘的 2 條節線。

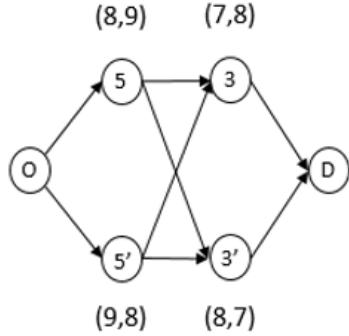


圖 4.8 第二次迭代 DP 網路圖

表 4.4 第二次迭代之 DP 求出之線段服務完之期望時間與電量

Node	O	5	3
ET	2.3	4.12	4.37
P	0	2.76	4.5
Node	O	5'	3'
ET	2.3	4.08	4.36
P	0	2.25	3.25

## (2) 回溯階段

由上一步驟，我們知道電量足夠服務完剩餘的 2 條節線，並計算最佳的路徑為  $S_{1,4}^{best} = (5', 3')$ ，接著計算無人機服務完路徑  $S_{1,4}^{best}$  所有節線後剩餘的電量  $currentPower = 7 - 3.25 = 3.75$ 、目前位置  $currentLoc = 7$ ，由於目前電量足夠回到  $chg$ ，更新變數，所有的節線都被服務，因此演算法停止，並可得到解碼的路徑，如下：

解碼之路徑：1 → 4 → 1 → 2 → 5 → 6 → 9 → 8 → 7 → 6

## 4.3 基因演算法 (Genetic Algorithm, GA)

基因演算法的概念源自於進化生物學，進化生物學中物種間存在個體上的差異是因為染色體的不同，由於染色體會因為交配、突變導致不一樣，當產生差異後，透過計算適應值，會選擇保留某些子代，隨著迭代的次數不斷增加，最後可以得到一組全域的最佳解。本研究參考 Karakatic and Podgorelec (2015) 與 Lo *et al.* (2018) 等論文以基因演算法求解多旅行銷售員問題或是車輛途程問題，將交配與變異的方法應用在本研究中。

## 編碼方式 (encode)

利用 4.1 節討論的 2 種編碼方式，我們可以將 GA 依照編碼方式分為  $GA_1$  與  $GA_2$ ， $GA_1$  是利用節線與方向編碼而成，如圖 4.3； $GA_2$  則是單純利用節線編碼而成，如圖 4.4，上述的編碼皆記錄在一個一維陣列中。

## 適應函式 (fitness function)

透過適應函式我們可以求出該條染色體的目標值，因本研究問題為最小化期望搜尋時間，故當值愈小表示愈接近最佳解，在計算目標值的方法上，可以透過 4.2 節介紹的解碼演算法分別計算每一架無人機的期望搜尋時間，並加總得到目標值。藉由接下來提到的交配以及突變策略，使得求出的目標值愈小。

## 挑選策略 (selection operator)

在每一次的迭代的一開始都會進行染色體的挑選，本研究選擇的挑選策略為輪盤法的方式，直覺想法是當解的目標值愈好時，其能夠被選進這次迭代的時候選染色體的機率就會愈高，以下介紹其方法。

假設有  $n_{pop}$  條染色體，其適應值分別為  $E(C_1), E(C_2), \dots, E(C_{n_{pop}})$ 。

1. 將所有的適應值取其倒數:  $E(C_i) = \frac{1}{E(C_i)} \quad \forall i = 1, \dots, n_{pop}$ 。
2. 計算所有適應值的加總:  $E_{total} = \sum_{i=1}^{n_{pop}} E(C_i)$ 。
3. 計算每條染色體被選到的機率:  $P_i = E(C_i)/E_{total}$ 。
4. 利用累積機率的方式，每條染色體的累積機率為  $Q_i$ 。
5. 最後產生隨機亂數  $x$ ，假若  $Q_i \leq x < Q_{i+1}$ ，則選擇染色體  $Q_i$

步驟 1 的計算方式之所以要取倒數是因為本研究的目標問題為最小化問題，目的是讓目標值愈小的染色體有愈高的機率被選中故以倒數表示。從步驟中可以發現到，有較佳適應值的染色體有較高機率被選中；而當染色體的適應值較差，該染色體仍有機率被選中，並不會完全被排除，其原因是儘管該染色體目前表現不好，在經過幾次迭代後，也可能會有不錯的適應值。

## 交配策略(crossover operator)

在交配策略中，由於每一條節線只能被一架無人機所服務，所以利用以下的方法來確保交配出來的為可行解。一開始先隨機挑選兩條染色體當作父母，如圖 4.9 所示，並在染色體中隨機選擇一個切割點，圖中即是以第三條節線為切割點。

接著將切割點後的節線進行交換，只要插入的節線並不存在於切割點之前，就插入；若存在則不插入，如圖 4.10，由於節線 6 並不存在於切割點前，因此插入相對應的位置。剩餘未被指派的節線則隨機插入空位中，如圖 4.11。



圖 4.9 選擇父母及交配的分割點

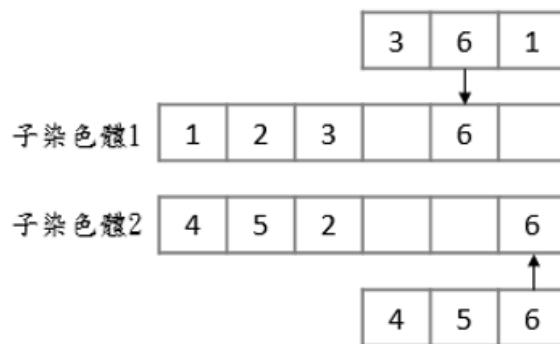


圖 4.10 進行交換與複製

子染色體1	1	2	3	4	6	5
子染色體2	4	5	2	3	1	6

圖 4.11 最後小孩結果

### 突變策略(mutation operator)

突變策略是為了避免基因演算法落入區域最佳解的方法，利用機率性的改變讓解有所不同，以下我們說明幾個不同的突變策略。

#### 突變策略 1:交換法

隨機選擇染色體的兩個位置，將位置上的基因進行互換，如圖 4.12，將節線 2 與節線 6 的位置互換。

#### 突變策略 2:插入法

隨機選擇染色體的兩個位置，將第一個位置上的節線插入第二個位置，如圖 4.13，將節線 2 插入節線 6 的位置之後，其他的依序往前遞移。



圖 4.12 突變策略-交換法

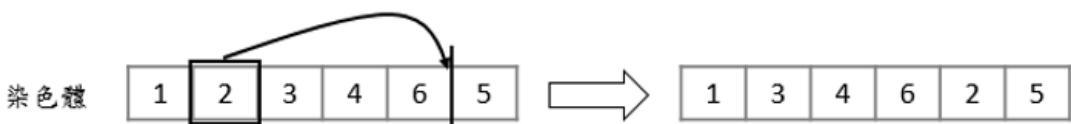


圖 4.13 突變策略-插入法

將上述進行方法進行整理，可將基因演算法表示如下表，其中  $n_{pop}$  為染色體個數、 $\alpha_c$  為迭代時進行交配之機率， $\alpha_m$  為突變之機率。

表 4.4 基因演算法流程

---

### Genetic Algorithm

---

1: **function** GA( $n_{pop}, \alpha_c, \alpha_m$ )

---

---

```

2: Generate initial population ( $n_{pop}$  random chromosome)
3: while stopping criteria not satisfied do
4:   evaluate each chromosome in population
5:   select  $n_{pop}$  chromosome and copy them to new population
6:   probability  $\alpha_c$  to do crossover and copy to population
7:   probability  $\alpha_m$  for each chromosome to mutate
8: end while
9: end function

```

---

#### 4.4 區域搜尋法 (Local Search , LS)

區域搜尋法是一種啟發式演算法，其概念是從一組初始解開始，向其周邊的鄰域解進行搜索，假如有更好的解則移動至該解，並從該解繼續搜尋其鄰域解；否則返回當前解繼續搜尋，區域搜尋法經常使用在車輛途程問題與旅行銷售員問題中。本研究參考 Hashimoto and Yagiura (2008)提出的區域搜尋法，該論文利用 2-opt\*、cross exchange 與 Or-opt 來搜尋鄰域解，同時為了讓搜尋更加有效率，針對每個節點只搜尋附近的節點。由於本研究的問題是以節線為討論對象，因此將此方法做修改並套用到本研究的問題中。

##### 編碼方式

參考 4.1 節中的編碼方式，僅以節線的編號來編碼，每組可行解我們比可以用二維陣列來儲存，若現在有 3 架無人機負責搜尋 10 條節線，可將其編碼以圖 4.14 表示之，以此例而言，無人機 1 會負責搜尋節線 2、3、5、9。以此類推。

無人機 1	<table border="1"><tr><td>2</td><td>3</td><td>5</td><td>9</td></tr></table>	2	3	5	9
2	3	5	9		
無人機 2	<table border="1"><tr><td>1</td><td>7</td><td>10</td></tr></table>	1	7	10	
1	7	10			
無人機 3	<table border="1"><tr><td>4</td><td>6</td><td>8</td></tr></table>	4	6	8	
4	6	8			

圖 4.14 編碼示意圖

## 鄰域列表 (Neighbor List)

為了更有效率的進行鄰域搜尋，我們針對每一條節線建立列表記錄附近的節線，假如兩條路徑中的部分節線進行交換時，判斷交換是否成立的條件是因為交換而連接的兩條節線是否在該節線的鄰域列表中，假如其中之一存在則交換成立；否則失敗。我們在程式的一開始就建立鄰域列表，並記錄距離每條節線最短的  $n_{nlist}$  條節線。

## 2-opt\* 演算法

2-opt\* 演算法是由 Potvin and Bengio (1996)所提出，為 Lin (1965) 針對旅行銷售員問題提出的 2-opt 之變形。其概念是從 2 架無人機的路徑各分成兩部分，並將 2 架無人機的後半部分進行交換，而演算法僅針對前半部的最後一個節點（下圖以節點表示節線），與後半部的第一個節點在鄰域列表中才可進行交換，如圖 4.15，針對無人機  $k$  與無人機  $k+1$  之路徑進行交換，首先可將無人機  $k$  之路徑拆成兩部分，第  $i$  個節點以前與第  $i+1$  節點以後，而無人機  $k+1$  同樣能將路徑拆成兩部分，並判斷第  $i$  個節點與第  $j+1$  個節點或是第  $j$  個節點與第  $i+1$  個節點是否在鄰域列表中，假如是就交換並執行搜索，若否則不進行交換。詳細的執行過程如表 4.5，此方法的鄰域解數量為  $O(mn_{nlist})$ ， $m$  為總節線數量。

為了使交換更為有效率，我們針對完成任務時間最晚的無人機與完成任務時間最早的無人機路線優先進行交換，其原因是當每架無人機使用時間愈平均，整個搜尋的時間會較短。

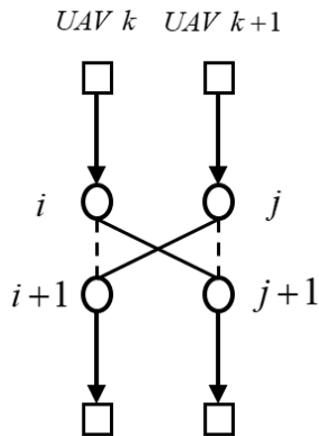


圖 4.15 2-opt\* 交換示意圖

表 4.5 2-opt\* 演算法流程

---

### 2-opt\*

---

Data:  $Gbest, S$

```

1: function 2-opt*( $Gbest, S$ )
2:   for  $UAV1$  sorted by completion time in increasing order do
3:     for  $UAV2$  sorted by completion time in decreasing order do
4:       for  $node1$  in  $UAV1$  path do
5:         for  $node2$  in  $UAV2$  path do
6:           separate  $UAV1$  path by  $node1$  into  $path1A$  and  $path1B$ 
7:           separate  $UAV2$  path by  $node2$  into  $path2A$  and  $path2B$ 
8:           if  $path1A$  last node and  $path2B$  first node in neighbor list or
9:              $path2A$  last node and  $path1B$  first node in neighbor list then
10:               $S \leftarrow$  exchange  $path1B$  and  $path2B$ 
11:              call cross_exchange( $Gbest, S$ ) function
12: end function

```

---

### cross exchange 演算法

cross exchange 演算法由 Taillard *et al.* (1997) 提出，其概念是將 2 架無人機之路徑各移除一段長度至多為  $L^{cross}$  的路徑，然後將路徑彼此進行交換，同樣的，為使交換更加有效率，只針對那些在鄰域的節點進行交換，如下圖 4.16，針對無人機  $k$  與無人機  $k+1$  之路徑進行交換，首先選擇節點  $i$  後長度至多為  $L^{cross}$  的路徑，

與節點  $j$  後長度至多為  $L^{cross}$  的路徑，並判斷節點  $i$  與節點  $j+1$  或是節點  $j$  與節點  $i+1$  是否在鄰域列表中，假如是則進行交換並搜索，並求解新的路徑  $S$  的目標值  $E(S)$ ，若目標值小於全域解的目標值  $E(Gbest)$  則取代全域解，並從新的路徑  $S$  做 cross exchange 演算法；若否則是不進行交換。詳細的執行過程如表 4.6，此方法的鄰域解數量為  $O((L^{cross})^2 mn_{nlist})$ 。

與之前相同，為了使交換更為有效率，我們針對完成任務時間最晚的無人機與完成任務時間最早的無人機路線優先進行交換。

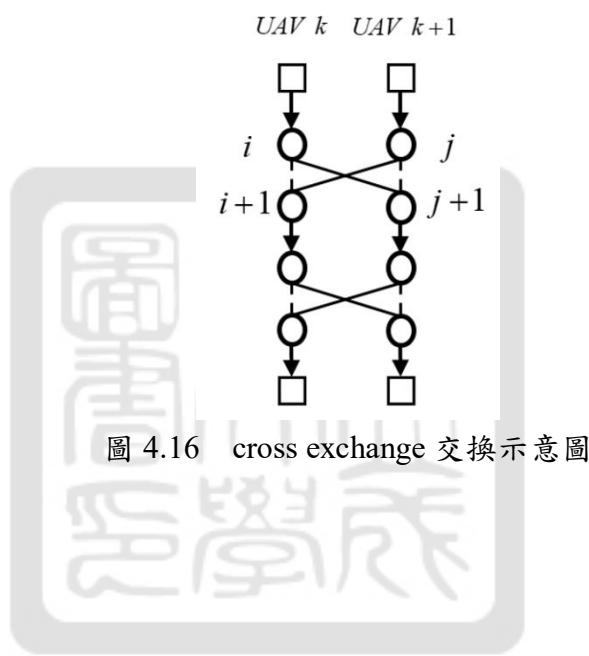


圖 4.16 cross exchange 交換示意圖

表 4.6 cross exchange 演算法流程

---

### **cross exchange**

---

Data:  $Gbest, S$

```

1: function cross_exchange( $Gbest, S$ )
2:   for  $UAV1$  sorted by completion time in increasing order do
3:     for  $UAV2$  sorted by completion time in decreasing order do
4:       for  $node1$  in  $UAV1$  path do
5:         for  $node2$  in  $UAV2$  path do
6:           for: $pathlen1$  in  $[1 \dots L^{cross}]$  do
7:             for  $pathlen2$  in  $[1 \dots L^{cross}]$  do
8:               remove  $UAV1$  path by  $node1$  with the path length equal to  $pathlen1$ 
9:               remove  $UAV2$  path by  $node2$  with the path length equal to  $pathlen2$ 
10:              if node  $i$  and node  $j$  or node  $j$  and node  $j+1$  both are in the
11:                neighbor list then
12:                   $S_{uav1}, S_{uav2} \leftarrow$  exchange two path
13:                  call Or-opt( $Gbest, S_{uav1}$ ) function
14:                  call Or-opt( $Gbest, S_{uav2}$ ) function
15:                  if  $E(Gbest) > E(S)$  then
16:                     $Gbest \leftarrow S$ 
17:                    2-opt( $Gbest, S$ )
18: end function

```

---

### Or-opt 演算法

Or-opt 演算法是由 Reiter *et al.* (1965)所提出，其概念是移除路徑中長度至多為  $L_{path}^{intra}$  的路徑，並插入到距離原移除位置至多  $L_{ins}^{intra}$  的位置，如圖 4.17，從節點  $i$  移除長度為  $L_{path}^{intra}$  的路徑，並將移除的路徑插入位置  $i + L_{ins}^{intra}$ ，並衡量交換完的新路徑的目標值是否較原本好，若是，則交換並從新的路徑開始尋找鄰域解；若否，則從原路徑繼續搜索，詳細的執行過程，如表 4.7。此方法的鄰域解數量為  $O(L_{ins}^{intra} L_{path}^{intra} m)$ 。

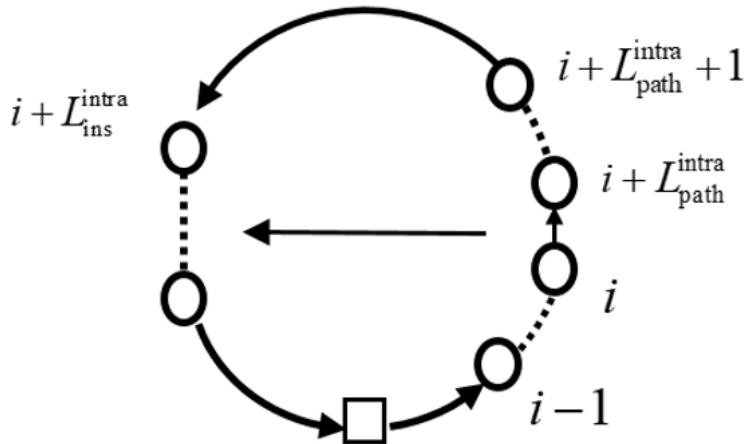


圖 4.17 Or-opt 交換示意圖

表 4.7 Or-opt 演算法流程

---

### Or-opt

---

Data:  $S_k$

```

1: function Or_opt( $S_k$ )
2:   for node in path do
3:     for len_intra_path in [1, ...,  $L_{\text{path}}^{\text{intra}}$ ] do
4:       for loc_ins in [node+1, ..., node+ $L_{\text{ins}}^{\text{intra}}$ ] do
5:         remove path from position node to node+ $L_{\text{path}}^{\text{intra}}$ 
6:          $S'_k \leftarrow$  insert the path into position node+ $L_{\text{ins}}^{\text{intra}}$ 
7:         if  $E(S_k) > E(S'_k)$  then
8:           call Or-opt( $S'_k$ ) function
9:   end function

```

---

介紹完上述演算法後，說明區域搜尋法的步驟如圖 4.18，演算法由上而下分別是 cross exchange、2-opt\* 與 Or-opt，由最下層的 Or-opt 函式開始，每當執行完 Or-opt 函式，就會執行一次 2-opt\*，然後繼續執行 Or-opt，直到 2-opt\* 執行完，同理，當 2-opt\* 執行完，就會再執行更上層的 cross exchange，假如有更好的解出現，立即接受，並重新從 Or-opt 函式開始，直到停止條件滿足則停止程式。

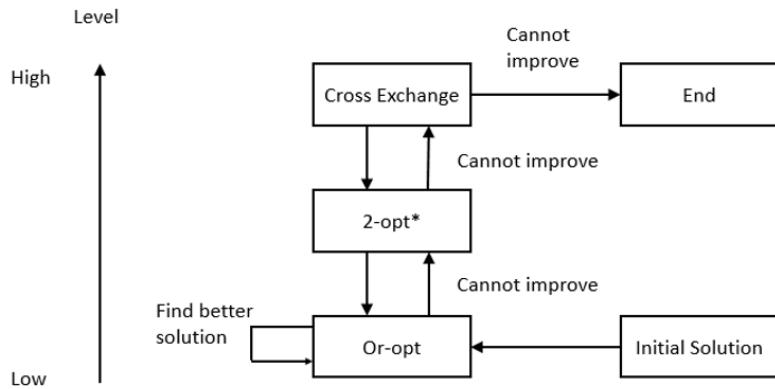


圖 4.18 區域搜尋法執行流程

## 4.5小結

儘管求解數學模型可以在小例子中求得最佳解，但往往問題隨著規模的擴大，求解時間會大幅增加，而求解品質則會逐漸下降。為求在短時間內得到品質高的解，演算法的實現將是必要的，本章首先提出將解利用編碼的方式來呈現，我們主要想要知道必須尋訪的節線順序，而詳細節線與節線間如何連接則是利用結合動態規劃與回溯法的解碼演算法而得。本章節提出兩種方式進行求解，第一種是基因演算法，透過染色體間彼此的交配變異，不斷迭代找到最好的組合。第二種是利用區域搜尋法，主要可分為兩個步驟，分別是無人機本身路線的優化與多架無人機間的路徑優化，首先，無人機本身路線的優化主要是透過內部路線的不斷重組來獲得，每當有更好的解時就將原來的解替換；多架人無人機間的路徑優化則是透過部分節線的交換來完成，每當完成一次無人機間的路徑優化，就立刻進行無人機本身路線的優化。為增加交換的效率，我們只針對鄰域的節線進行交換，當停止條件達成時即完成區域搜尋法。

## 第五章 數值分析

本章節針對本研究所建立的數學模型與數學演算法進行測試與分析，本研究的測試環境為 Windows 10 作業系統，搭配 Intel Core i7-8700，3.20GHZ\*6 處理器，與 8G 記憶體，以 Python 為程式語言撰寫數學模型與數學演算法，並利用 Gurobi 8.1.1 版求解數學模型，其中求解時間 *CPU TIME* 設定為 2 小時(7200 秒)，以下說明測試網路參數設定，接著針對不同的參數設定與網路大小等不同情況來分析測試結果。

### 5.1 測試資料參數設定

本研究探討問題之網路圖為完全圖網路，即是任兩點間皆任意可移動，表 5.1 為測試資料參數設定，其中必須進行搜尋之節線數為  $m$ ，節點數為  $n$ ，無人機的數量為  $K$ ，與充換電站數  $n_c$ ，其中  $m$  主要決定問題的規模。設定無人機移動模式速率為 1.25，搜尋模式速率為 1，無人機初始電量與最大電量皆為 5000，充電時間假設為 30 分鐘。之後都以表 5.1 參數設定，針對不同網路規模與不同無人機數量來產生測資。

表 5.1 測試資料參數設定

網路 結構	參數設定							
	搜尋 節線數	節點 數	無人 機數	充電 站數	移動模 式速率	搜尋模 式速率	無人機 電量	充電 時間
完全圖	$m$	$n$	$K$	$n_c$	1.25	1	5000	30

## 5.2 測式網路圖設定

針對隨機、樹狀以及軸輻式網路進行測試。本研究使用基於 python 開發的 NetworkX 套件(<https://networkx.github.io>)來產生隨機與樹狀網路圖。隨機網路利用 gnm\_random\_graph()函式給定節點與節線數並檢查網路之連通性後產生；樹狀網路則是利用 nrandom\_powerlaw\_tree()函式給定節點數來隨機產生。而軸輻式網路是首先產生多個集散點(hub)以及多個隨機節點，接著建立連接每個隨機節點至最近集散點之節線與集散點之間的節線。以下針對隨機、樹狀與軸輻式網路分別測試網路規模為小型與中型與大型的測試資料各 3 筆，其節點數與節線數如表 5.2 所示。

表 5.2 網路圖資訊

網路名稱	節點數	節線數
random _ s	8	10
random _ m	15	20
random _ l	35	50
Tree _ s	[11,13]	[10,12]
Tree _ m	[19,24]	[18,23]
Tree _ l	[49,51]	[48,50]
hub _ and _ spoke _ s	[10,12]	[10,12]
hub _ and _ spoke _ m	[20,22]	[20,22]
hub _ and _ spoke _ l	[50,52]	[49,51]

## 5.3 固定充換電站點之無人機群路徑規劃問題之測試

關於固定充換電站點之無人機群路徑規劃問題的測試，首先我們將針對整數規劃模式進行測試，接著再比較數學啟發式演算法之求解效率與求解品質。

### 5.3.1 整數規劃模式比較

本小節針對基本模式  $M_L^{C_0}$  加入初始解之模式  $\overline{M_L^{C_0}}$  進行測試，表 5.3 為模式大

小估算，以  $big-O$  估算模式之變數與限制式數量，可以發現變數或是限制式的數量主要取決於節點數  $n$ 、無人機群中最長路徑之節線數  $L$  與無人機數量  $K$ ， $L$  的大小主要受到  $n$  與  $K$  影響，當  $n$  愈大或是  $K$  愈小，則  $L$  愈大。

表 5.4 為已知充換電站位置下之數學模式  $M_L^{C_0}$ 、 $\overline{M_L^{C_0}}$  在不同網路規模與型態下的求解表現，其中  $Num$  為該組網路測試資料數； $Num\ opt.$  為該網路測試資料中求得最佳解之個數； $Num\ Arcs$  為搜尋節線的數量； $CPU\ Time$  為數學模式的求解時間； $Gap$  是利用最佳化軟體 Gurobi 之測試結果(求解時限為 7200 秒)，計算方式如下：

$$Gap = \frac{UB - LB}{UB} \times 100\% \quad UB \text{ 及 } LB \text{ 為 Gurobi 所求得之上下界。}$$

由表 5.4 可以發現當固定  $K$  下，當階層數  $L$  增加時，求解時間會大幅上升，當模型處理中等規模之網路時，所有測資都無法在有效的時限內求得，在求解品質的部分， $GAP$  值在階層數多時表現都極差，整體而言，數學模型只可求解小規模的網路。

加入區域搜尋法所產生的初始解之數學模式  $\overline{M_L^{C_0}}$ ，其求解效率與品質都並未明顯優於未加入初始解之數學模式  $M_L^{C_0}$ ，但由圖 5.1 與圖 5.2 可以發現模式  $\overline{M_L^{C_0}}$  相比模式  $M_L^{C_0}$ ，可由較佳的數值開始收斂，較節省時間，但由結果可以發現模式  $\overline{M_L^{C_0}}$  之結果未必優於  $M_L^{C_0}$ 。

表 5.3 混整數規劃模式之問題大小估算

建構網路	數學模式	二元變數個數	實數變數個數	限制式個數
層空	$M_L^{C_0}$	$O(n^2 LK)$	$O(n^2 + LK)$	$O(n^2 LK)$

表 5.4 數學模式測試(求解品質與效率)

network	K	L	Num	$M_L^{C_0}$			$\overline{M}_L^{C_0}$		
				Num opt	CPU Time(s)	Gap (%)	Num opt	CPU Time(s)	Gap (%)
random _ s	1	13	3	3	168	0.00	3	147	0.00
	2	7	3	3	145	0.00	3	151	0.00
	4	4	3	3	356	0.00	3	358	0.00
random _ m	1	30	3	0	-	94.9	0	-	95.6
	2	13	3	0	-	90.5	0	-	90.3
	4	8	3	0	-	79.7-	0	-	79.5
Tree _ s	1	16	3	2	4599	12.5	2	4812	22.6
	2	8	3	2	5214	24.6	1	5220	39
	4	5	3	1	4803	34.3	1	4803	34.3
Tree _ m	1	32	3	0	-	98.0	0	-	98.1
	2	15	3	0	-	96.4	0	-	95.7
	4	9	3	0	-	80.0	0	-	85.5
hub _ and _ spoke _ s	1	18	3	1	6993	55.9	1	7070	57.1
	2	10	3	1	4813	50.6	1	4843	50.4
	4	6	3	0	-	56.1	0	-	56.2
hub _ and _ spoke _ m	1	35	3	0	-	*	0	-	*
	2	17	3	0	-	96.3	0	-	93.4
	4	10	3	0	-	86.8	0	-	82.4

註: - 表示該組網路的測試資料，模式在求解時間上限(7200 秒)內無法求得最佳解。

\*\*表示模型在時限內無法求得可行解。

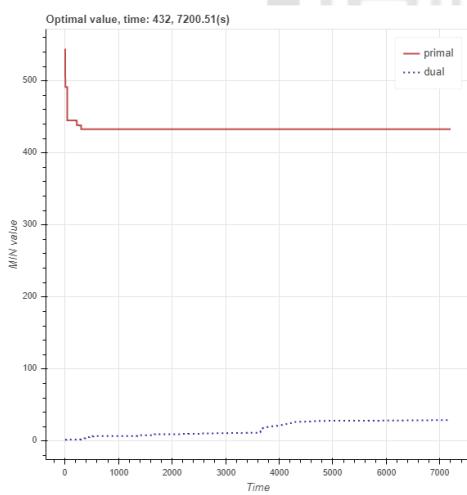


圖 5.1 測資  $random\_m$  下  $K = 2$  時

模式  $M_L^{C_0}$  之收斂情形

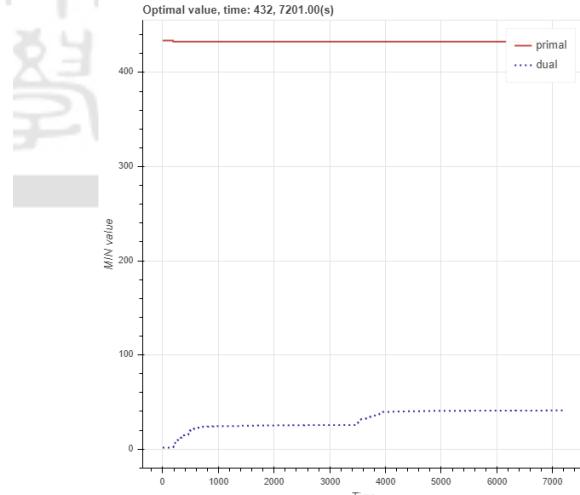


圖 5.2 測資  $random\_m$  下  $K = 2$  時

模式  $\overline{M}_L^{C_0}$  之收斂情形

### 5.3.2 數學演算法測試

在演算法的部分，我們針對 *GA-1*、*GA-2* 及 *LS* 進行測試，表 5.5 為演算法之參數設定，在 *GA* 的部分，初始的染色體總量  $n_{pop}$  為 100 條，交配的機率  $\alpha_c$  設定為 0.6 而突變的機率設定為 0.2，共迭代 400 次。而 *LS* 的部分，參數設定則是參考 Hashimoto and Yagiura (2008) 的設定，鄰域解的個數  $n_{nlist}$  為 10、cross exchange 中交換最大長度  $L^{cross}$  為 3、Or-opt 中移除最多  $L_{path}^{intra}$  條節線並插入鄰近的  $L_{ins}^{intra} = 3$  個位置，*LS* 則是進行 400 次迭代沒進步就停止。

表 5.5 數學演算法參數設定

GA				LS			
$\alpha_c$	$\alpha_m$	$n_{pop}$	<i>Iter</i>	$n_{nlist}$	$L^{cross}$	$L_{path}^{intra}$	$L_{ins}^{intra}$
0.6	0.2	100	400	10	3	3	3

我們分別對 3 個方法進行測試，測試的標準則是依據數學模型求得的最佳解 ( $M_L^{C_0}$  求解 2 小時的近似解或是演算法找到的最好的解)，演算法執行超過 20 分鐘即終止，由表 5.6 可看出，*LS* 在求解品質明顯優於基因演算法且 *Gap* 都能在 10% 內，除了無人機數量為 1 架時表現較差，其原因推測是單一條路徑交換有限，因此較易陷入區域解。此外，*LS* 的鄰域數量隨著節線數增加而大幅上升，因此求解時間也增加。在基因演算法的部分，在 *GAP* 值表現上 *GA-1* 與 *GA-2* 並無明顯差別，當問題規模增大時，*GAP* 值急遽上升，整體來說，基因演算法在解的品質上與區域搜索法有著相當大的落差，這部分還需要改善基因演算法的交配與變異方法。針對不同的網路型態，*LS* 的 *GAP* 值表現一致，且軸輻式網路表現又較樹狀網路以及隨機網路更佳。

表 5.6 數學演算法測試(求解品質與效率)

network	K	Num	GA-1		GA-2		LS	
			CPU Time(s)	Gap(%)	CPU Time(s)	Gap(%)	CPU Time(s)	Gap(%)
random _ s	1	3	12.6	5.4	11.7	10.2	2.7	0.8
	2	3	11.5	12.4	18.8	8.4	4.8	1.2
	4	3	12.7	18	13.3	19.5	0.7	1.8
random _ m	1	3	15.8	63	19	60.9	18.8	25
	2	3	17	65.8	30.2	66.9	56	9.2
	4	3	26.1	71.7	31.2	69.3	14.8	2.1
random _ l	1	3	31.3	25.5	43.6	25.7	104.6	0.0
	2	3	34.8	72.8	78.3	68.9	1177.8	0.0
	4	3	33.5	78.7	69.3	77.4	500.3	0.0
Tree _ s	1	3	22.4	19	12	14.9	3.4	11.1
	2	3	23.8	18.3	19.9	20.1	6.8	6.2
	4	3	22	24.2	13.9	32.1	1.2	4.6
Tree _ m	1	3	35.3	79.1	20.9	80.4	10.9	12.1
	2	3	37.5	81.5	31.2	85.4	94	4.2
	4	3	33.5	104	29	99.4	39	2.1
Tree _ l	1	3	37.0	41.1	57.7	45.4	129.5	0.0
	2	3	39.6	195	97.5	192	-	0.0
	4	3	41.6	245	89.9	246	694.6	0.0
hub _ and _ spoke _ s	1	3	25	6.1	13.9	2.8	4.5	3.7
	2	3	23	6.2	20.7	7.2	11.1	0.4
	4	3	25	7.2	14	9.8	1.3	4.2
hub _ and _ spoke _ m	1	3	33.7	30.3	24.6	25.8	35	8.7
	2	3	33.4	21.4	39.7	30.2	92.8	1.1
	4	3	31.5	36.5	33.7	36.4	20	2.0
hub _ and _ spoke _ l	1	3	34.4	21.0	51.5	18.8	117.5	0.0
	2	3	37.2	57.3	89.7	51.5	-	0.0
	4	3	36.6	69.6	85.5	69.7	671.0	0.0

註: - 表示該組網路的測試資料，模式在求解時間上限(1200 秒)內無法求解完畢。

## 5.4 使用無人機之效益比較

過往傳統人力徒步搜救路線規劃文獻中，搜救隊必須沿路搜救，任何的移動都必須建立在現有的節線上，而無人機的使用可以跳脫原有網路的路段連結限制，無人機可以直接移動至原網路未連結之鄰近節點。為比較兩者之差別，本研究假設所有必需搜尋節線均相連的情況下，傳統人力徒步搜救僅能沿著實體存在的道路進行移動，而無人機可於任兩節點間移動。表 5.7 說明相較於傳統人力徒步搜

救，使用無人機有助於減少期望搜尋時間，且不論是何種網路結構，期望搜尋時間都至少改善 20%以上，當網路規模增大時改善量亦增加。若從網路結構進行分析，可以發現無人機於軸輻式網路之期望搜尋時間改善量尤其明顯，推測原因是軸輻式網路中任兩節點的連接都需要經過集散點才可抵達，需在移動的過程耗費大量時間。

表 5.7 使用無人機之效益分析

<i>Network</i>	<i>Improve(%)</i>
<i>Tree<sub>s</sub></i>	22.58
<i>Tree<sub>m</sub></i>	38.17
<i>hub_and_spoke<sub>s</sub></i>	50.65
<i>hub_and_spoke<sub>m</sub></i>	73.47

## 5.5 小結

首先介紹測試網路的參數設定，以 *big-O* 計算數學模式變數與限制式的數量，並估算模式的大小，接著測試數學模型的求解效率與品質，並加入演算法求得之初始解進入模型中進行比較。接下來針對 3 種數學演算法進行數值測試，可以發現 *LS* 可以在不同規模的網路中都得到不錯的可行解，而基因演算法表現相較而言仍有許多的進步空間。針對特殊型態之網路，本研究挑選樹狀網路及軸輻式網路進行模型與演算法的測試，在演算法的部分，*LS* 在不同的網路結構中表現一致，均可在短時間內求得近似最佳解(nearly optimal)。與傳統徒步搜救路線規劃文獻比較，無人機的加入可以跳脫原有網路的路段連結限制，大幅降低期望搜尋時間，其改善尤以軸輻式網路最為明顯。

## 第六章 結論與未來研究議題建議

### 6.1 結論

當搜索與救援服務需求發生時，搜救隊必須把握黃金救援時間，在短時間內發現標靶，本研究旨在探討一個最佳化搜救服務系統設計以及搜救服務作業管理問題，當單一靜態標靶於一般節線長度網路上各節線出現之機率為已知時，如何決策每架無人機的移動路徑，搜尋所有可能尋獲標靶之節線，以最小化期望搜尋時間。為求更貼近實務，本研究假設：(1)無人機存在兩種移動模式，分別為慢速的「搜尋模式」，以及快速的「移動模式」；前者僅能在原網路上的節線路段進行，而後者則可在續航力允許下自由地在任兩節點間移動。(2)無人機存在續航力限制，在電量耗盡前無人機須前往某些充換電站的基地節點上充換電。上述問題與文獻中的 KRPP 較為類似，但設定更為符合實務，並具備困難性與挑戰性。以下臚列本論文的具體貢獻，並於 6.2 小節建議未來可能的研究方向：

#### 具體貢獻：

1. 本研究之目標為最小化期望搜尋時間，在過去無人機搜尋路徑規劃文獻中未曾被考慮，卻較符合現實需求。為求更貼近實務，本研究設定無人機存在兩種移動模式且移動需要考慮當下的電力狀態，若電力不足須前往某些充換電站之節點上充換電，才可再次滿電出發執行搜尋任務。
2. 與過去文獻相比，本研究可處理多組搜救人員搜尋單一靜態標靶，隨機分佈於一般節線長度網路之問題，並提出一數學規劃模式求解已知充換電站的基地節點下  $K$  架無人機之搜尋路徑規劃問題。(第二章；第三章)
3. 過去文獻中期望搜尋時間之計算方式利用網路圖之特性(如邊長皆為 1 之網格網路)簡化後加入目標式，本研究透過網路結構與建模之手法改寫期望搜

尋時間為線性方程式，並加入本研究數學模型之目標式。(第 3.3 小節；第 3.4 小節)

4. 提出「以階層空間網路建構」之模式，利用 Level-Space 的方式展開網路，每一個階層空間網路代表一架無人機，共  $K$  個階層空間網路，以階層節線表示無人機移動之路徑。此方法求解效率不佳，且受最大階層數  $L$  與無人機數  $K$  影響。(第 3.4 小節)
5. 提出「解碼演算法」將一組搜尋節線解碼為路徑：解碼演算法結合動態規劃法(Dynamic Programming)與回溯法(Backtracking)之概念，將一組必需搜尋節線所組成的解，考量電力限制與充換電站位置下，解碼為無人機之移動路徑。(第 4.2 小節)
6. 提出「最佳充換電站點設置與無人機群路徑規劃模式」：考量可建置的充換電站點個數，需決策最佳充換電站點設置的位置與無人機之搜尋路徑以使期望搜尋時間最小。(附錄)

## 6.2 未來研究

針對本研究之搜尋路徑規劃問題，尚有許多問題需要進一步探討，如下所示：

### 1. 可持續精進及發展本研究之數學規劃模式與演算法：

#### 數學模式

由於本研究之數學模式並無法求解中、大規模之網路(僅能處理最大階層數為 30 以下之網路)，小規模之網路也需耗費一些時間才可求解完畢，除了考慮是否有其他更有效率的模型建立方法外，以限縮層空網路中節線與節點的方式減少網路規模亦是未來可以嘗試的解決方案。

#### 演算法

本研究提出解碼演算法將一組搜尋節線，考量無人機續行限制與充換電站位

置下，解碼為一組路徑，需要透過回呼的方式將所有的搜尋節線解碼為路徑，因此方法較耗時也需要足夠的記憶體空間。在區域搜尋法的部分，每一次無人機間路徑的交換或是無人機自身路徑的交換，都會更動到原來的路徑，因此期望搜尋時間都必須重新計算，以判斷解的優劣，若可在短時間內判斷新路徑的優劣則可大幅減少運算時間。由於基因演算法表現太差，或許也可嘗試不同的染色體編碼方式等。

#### **2. 考慮充換電站點設置之無人機群搜尋路徑規劃研究：**

目前問題是針對已知充換電站位置下最佳搜尋路徑規劃問題，然而充換電站的位置對於期望搜尋時間與無人機的移動路徑都有極大的影響，設置最佳的充換電站點將有助於減少能源的消耗，並提升搜救之效益。目前尚未有演算法針對上述問題進行求解。

#### **3. 考慮移動式充換電站之無人機群搜尋路徑規劃研究：**

目前已有移動式充換電站之存在，隨著搜尋範圍逐漸擴大與深入，原本設立的充換電站點可能不敷使用，若是移動式充換電站則可隨著無人機群的搜尋，移動至無人機的附近，減少無人機充電所需的移動成本。

#### **4. 考量無人機共同搜尋之無人機搜尋路線規劃研究：**

當某一條節線有極大的機率發現靜態標靶或是其他無人機呈現閒置的狀態時，若仍限制一條節線僅可由一架無人機進行搜尋，將導致搜尋時間過長而未能最小化期望搜尋時間，考量上述情形，無人機應可共同搜尋節線，以加速完成所有節線的尋訪。

## 附錄

本研究假設充換電站位置為已知的情況下，規劃無人機群之搜尋路徑，然而我們發現充換電站的設址亦是影響期望搜尋時間之關鍵因素，為增進搜救之效益，應同時考量充換電站的設址問題與無人機群之搜尋路徑規劃問題。此部分最佳充換電站點設置與無人機群路徑規劃模式，原本應置於本研究 3.4 節之後，但礙於時間與問題規模，故將其移至附錄，做為本研究後續之討論。

本研究考慮充換電站位置是已知的情況下，無人機群路徑應如何規劃，而在研究中發現除了一開始無人機是由某一節點出發之外，無人機其餘的搜尋路徑皆是從充換電站出發，經過一些未尋訪之節線，再回到充換電站的模式。不同的充換電站位址對於期望搜尋時間有不同之影響，舉例來說，若充換電站設置在網路圖之最外層，當無人機電力耗盡前無人機必須耗費較多的移動時間回到最外層進行換電池。此外，假設無人機可尋訪的最遠的路程為  $R$ ，則搜救範圍最多在充換電站周圍半徑  $R$  的圓內，可能有某些區域無法被涵蓋。在相同資源下，充換電站位址會對搜救有影響，因此將充換電站部署在最佳位置，不僅可以減少期望搜尋時間，亦可增加無人機搜救範圍。

附錄將探討在充換電站在位置未知的情況下，已知搜救隊最多能建立的充換電站點個數  $C$ ，求解最佳的充換電站點部署以使期望搜尋時間最小。此模式藉由修改在(3.3)小節的固定充換電站點模式  $M_L^{C_0}$  而得，保留，並新增新的變數和限制式。以下新增新的符號定義：

### 參數

$Q$

充換電站點個數

## 決策變數

$u_i \in \{0,1\}$  若充換電站設置在節點  $i$  則為 1；反之為 0

$e_i^{l,k} \in \{0,1\}$  若無人機  $k$  在第  $l$  階層於節點  $i$  進行充電，且節點  $i$  為充換電站，則為 1；反之為 0

## 數學模型依照第 3.4 節之模式進行修改

刪除限制式(3.3.22)並新增限制式(6.2.1)與(6.2.2)限制當  $rc_i^{l,k} + u_i$  為 2 時  $e_i^{l,k}$  為 1；否則為 0。限制式(6.2.3)考慮充換電站之個數為  $C$  之條件。

$$e_i^{l,k} \leq (rc_i^{l,k} + u_i) / 2 \quad \forall i \in V \setminus \{o, d\}; l=1, \dots, L; k=1, \dots, K \quad (6.2.1)$$

$$e_i^{l,k} \geq rc_i^{l,k} + u_i - 1 \quad \forall i \in V \setminus \{o, d\}; l=1, \dots, L; k=1, \dots, K \quad (6.2.2)$$

$$\sum_{i \in V \setminus \{o, d\}} u_i = Q \quad (6.2.3)$$

將電力限制式中變數  $rc_i^{l,k}$  改為  $e_i^{l,k}$ 。

$$p_{l-1,k} - \sum_{a \in A^s} x_a^{l,k} D_a - \sum_{\substack{a \in A^t \\ tail(a) \neq o \\ head(a) \neq d}} y_a^{l,k} D_a v_a + P_{\max} \sum_{i \in V \setminus \{o, d\}} e_i^{l,k} \geq p_{l,k} \quad \forall l=1, \dots, L; k=1, \dots, K \quad (6.2.4)$$

$$p_{l-1,k} - \sum_{a \in A^s} x_a^{l,k} D_a - \sum_{\substack{a \in A^t \\ tail(a) \neq o \\ head(a) \neq d}} y_a^{l,k} D_a v_a + M \sum_{i \in V \setminus \{o, d\}} e_i^{l,k} \geq p_{l,k} \quad (6.2.5)$$

$$\forall l=1, \dots, L; k=1, \dots, K$$

$$p_{l-1,k} - \sum_{a \in A^s} x_a^{l,k} D_a - \sum_{\substack{a \in A^t \\ tail(a) \neq o \\ head(a) \neq d}} y_a^{l,k} D_a v_a - M \sum_{i \in V \setminus \{o, d\}} e_i^{l,k} \leq p_{l,k} \quad \forall l=1, \dots, L; k=1, \dots, K \quad (6.2.6)$$

$$\sum_{\substack{a \in A^s \\ head(a)=i}} x_a^{l,k} + \sum_{\substack{a \in A^t \\ head(a)=i}} y_a^{l,k} + M (1 - rc_i^{l,k}) \geq 1 \quad (6.2.7)$$

$$\forall i \in V_c; l=1, \dots, L; k=1, \dots, K$$

$$\sum_{\substack{a \in A^s \\ head(a)=i}} x_a^{l,k} + \sum_{\substack{a \in A^t \\ head(a)=i}} y_a^{l,k} - M(1 - rc_i^{l,k}) \leq 1 \quad (6.2.8)$$

$$\forall i \in V_c; \ l = 1, \dots, L; \ k = 1, \dots, K$$



## 參考文獻

- [1] Ahr, D., & Reinelt, G. (2002, September). New heuristics and lower bounds for the min-max k-Chinese postman problem. In *European symposium on algorithms* (pp. 64-74). Springer, Berlin, Heidelberg.
- [2] Aráoz, J., Fernández, E., & Zoltan, C. (2006). Privatized rural postman problems. *Computers & operations research*, 33(12), 3432-3449.
- [3] Benavent, E., Corberán, Á., & Sanchis, J. M. (2010). A metaheuristic for the min–max windy rural postman problem with K vehicles. *Computational Management Science*, 7(3), 269-287.
- [4] Berger, J., & Lo, N. (2015). An innovative multi-agent search-and-rescue path planning approach. *Computers & Operations Research*, 53, 24-31.
- [5] Berger, J., Lo, N., & Noel, M. (2014). A new multi-target, multi-agent search-and-rescue path planning approach. *International Journal of Computer, Electrical, Automation, Control and Information Engineering*, 8(6), 935-944.
- [6] Campbell, J. F., Corberán, Á., Plana, I., & Sanchis, J. M. (2018). Drone arc routing problems. *Networks*.
- [7] Forsmo, E. J., Grøtli, E. I., Fossen, T. I., & Johansen, T. A. (2013, May). Optimal search mission with unmanned aerial vehicles using mixed integer linear programming. In *Unmanned Aircraft Systems (ICUAS), 2013 International Conference on* (pp. 253-259). IEEE.
- [8] Hashimoto, H., & Yagiura, M. (2008, March). A path relinking approach with an adaptive mechanism to control parameters for the vehicle routing problem with time windows. In *European Conference on Evolutionary Computation in Combinatorial*

*Optimization* (pp. 254-265). Springer, Berlin, Heidelberg.

- [9] Jotshi, A., & Batta, R. (2008). Search for an immobile entity on a network. *European Journal of Operational Research*, 191(2), 347-359.
- [10] Kang, M. J., & Han, C. G. (1998, February). Solving the rural postman problem using a genetic algorithm with a graph transformation. In *SAC* (Vol. 98, pp. 356-360).
- [11] Karakatič, S., & Podgorelec, V. (2015). A survey of genetic algorithms for solving multi depot vehicle routing problem. *Applied Soft Computing*, 27, 519-532.
- [12] Kemper, F. P., Suzuki, K. A., & Morrison, J. R. (2011). UAV consumable replenishment: design concepts for automated service stations. *Journal of Intelligent & Robotic Systems*, 61(1-4), 369-397.
- [13] Lenstra, J. K., & Kan, A. R. (1976). On general routing problems. *Networks*, 6(3), 273-280.
- [14] Li, B., Patankar, S., Moridian, B., & Mahmoudian, N. (2018, August). Planning Large-Scale Search and Rescue using Team of UAVs and Charging Stations. In *2018 IEEE International Symposium on Safety, Security, and Rescue Robotics (SSRR)* (pp. 1-8). IEEE.
- [15] Li, S., & Huang, S. (2018). Multiple searchers searching for a randomly distributed immobile target on a unit network. *Networks*, 71(1), 60-80.
- [16] Lin, S. (1965). Computer solutions of the traveling salesman problem. *Bell System Technical Journal*, 44(10), 2245-2269.
- [17] Lo, K. M., Yi, W. Y., Wong, P. K., Leung, K. S., Leung, Y., & Mak, S. T. (2018). A genetic algorithm with new local operators for multiple traveling salesman problems. *International Journal of Computational Intelligence Systems*, 11(1), 692-705.
- [18] Moskal II, M. D. (2013). *A Mathematical Programming Approach to Immobile Entity Search*. State University of New York at Buffalo.
- [19] Muyldermans, L., Beullens, P., Cattrysse, D., & Van Oudheusden, D. (2005). Exploring

variants of 2-opt and 3-opt for the general routing problem. *Operations research*, 53(6), 982-995.

- [20] Orloff, C. S. (1974). A fundamental problem in vehicle routing. *Networks*, 4(1), 35-64.
- [21] Potvin, J. Y., Kervahut, T., Garcia, B. L., & Rousseau, J. M. (1996). The vehicle routing problem with time windows part I: tabu search. *INFORMS Journal on Computing*, 8(2), 158-164.
- [22] Raap, M., Meyer-Nieberg, S., Pickl, S., & Zsifkovits, M. (2017). Aerial vehicle search-path optimization: a novel method for emergency operations. *Journal of Optimization Theory and Applications*, 172(3), 965-983.
- [23] Reiter, S., & Sherman, G. (1965). Discrete optimizing. *Journal of the Society for Industrial and Applied Mathematics*, 13(3), 864-889.
- [24] Royset, J. O., & Sato, H. (2010). Route optimization for multiple searchers. *Naval Research Logistics (NRL)*, 57(8), 701-717.
- [25] Stone, L. D. (1976). *Theory of optimal search* (Vol. 118). Elsevier.
- [26] Taillard, É., Badeau, P., Gendreau, M., Guertin, F., & Potvin, J. Y. (1997). A tabu search heuristic for the vehicle routing problem with soft time windows. *Transportation science*, 31(2), 170-186.
- [27] Yao, P., Xie, Z., & Ren, P. (2017). Optimal UAV Route Planning for Coverage Search of Stationary Target in River. *IEEE Transactions on Control Systems Technology*.
- [28] Yu, W., & Batta, R. (2010). *Search for an Immobile Entity on a Unit Graph: A Mathematical Programming Approach*.