

國立成功大學

資訊管理研究所

碩士論文

以無人車執行最初與最後一哩路配送之

軸輻式城市物流網路設計問題研究

Strategic Design of Hub-and-Spoke City

Logistics Network

using Autonomous Vehicle for the First and

Last Mile Delivery

研究生：廖嘉豪

指導教授：王逸琳 博士

中華民國一百零八年五月

國立成功大學

碩士論文

以無人車執行最初與最後一哩路配送之軸輻式城市物流網路設計問題研究

Strategic Design of a Hub-and-Spoke City Logistics Network using Autonomous Vehicles for the First and Last Mile Delivery

研究生：廖嘉豪

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

莊坤達

陳龍

吳政勳

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

隨著日益高升的當日配送與即時外送需求，都市內的物流配送業務量成長迅速，不當的物流配送機制可能導致配送耗時過長、載具閒置、交通壅塞甚至環境污染等諸多問題。本研究針對起訖點在城市中，需一日內送達的龐大送貨需求（譬如：郵政快遞系統、食物幫買服務等），提出了軸輻式配送網路(Hub-and-Spoke)的設計架構，使用小型無人車來當最初及最後一哩路的配送載具，以大幅地降低物流的運輸成本。舉例來說，當某地出現一筆待送貨物，該處附近的集散點(Hub)將派出小型無人車直接去收取貨物，返回集散點後再與其它貨物由貨車將之轉運到其目的地附近的集散點，繼續由當地的小型無人車接力送達目的地。相較於傳統將每件貨物採點對點的直送方式，此種轉運方式雖較費時，但可發揮經濟規模的運送效益，大幅減少運輸成本。本論文在已選好集散點位置之後，主要探討此配送網路圖中集散點間如何決定轉運貨車路線與排程的「區位路線問題」(Location Routing Problem)，透過時空網路圖(Time-Space Network)，去設計數學規劃模式及啟發式演算法。

本研究提出的創新配送機制，綜合考慮了軸輻式網路、集散點間的貨車以及各區小型無人車隊的整體調度模式，可處理依時變化的配送需求，甚至可將各區小型無人車隊視為在各分區之間依需求而互補調度，以增加貨車跟小型無人車的使用率，也更能夠應付不同時空下較大的需求出現。

目前我們曾嘗試使用貪婪式演算法來快速求得可行的貨車路線，以其當基因演算法的初始解；亦發展出一個分段求解數學規劃模式的方式，在較短時間內得到高品質解。

**關鍵字：**城市物流、小型無人車、最初及最後一哩路、軸輻式網路、整數規劃

# Strategic Design of the First-and-Last Mile Hub-and-Spoke City Logistics Network using Autonomous Vehicles

Chia-Hao Liao

I-Lin Wang

Institute of Information Management

## SUMMARY

With the increasing popularity of on-line shopping and cloud kitchens, the same-day delivery demands have rapidly grown and caused much city logistics challenges. More vehicles for shipping huge amount of origin-destination (OD) deliveries inside a city lead to more traffic jams, accidents, energy consumption and air pollutions. We proposed a Hub-and-Spoke (HS) framework to design the logistics network for these OD shipments, where small autonomous vehicles are used to conduct the first-and-last mile delivery that connect customers to a hub, and then trucks tranship those shipments between hubs. Such an HS logistics network helps reduce the number of vehicles required for individual OD shipments. Given the estimated amount of OD shipments in each time period during one day, we seek optimal schedules and routes for transshipment trucks between hubs. Moreover, by treating the autonomous vehicles as shared vehicles, we can also reposition these autonomous vehicles between hubs in different time periods to increase the utilization of autonomous vehicles and trucks. We have proposed Integer Programs on a time-space network to calculate exact optimal solutions, but it is too time consuming. To solve more practical cases of larger sizes, we propose a greedy algorithm and a Genetic Algorithm to find good feasible truck routes in short time. Then we propose a framework that iteratively solves a smaller time-space network of few periods in a rolling horizon fashion, which produces the best results in our computational experiments.

**Key words:** City Logistics, Autonomous Vehicle, First-and-Last Mile, Hub-and-Spoke Network, Integer Program

## INTRODUCTION

In recent years, more e-commerce activities and on-line kitchens trigger huge amounts of same-day delivery requests in city logistics. In particular, many origin-destination (OD) shipments appear at different time periods and places. If we conduct all of these OD deliveries one by one using direct shipping by individual vehicles, the roads would be full of those vehicles and lead to traffic jams, accidents, and air pollution. Such a direct shipping mechanism requires many vehicles, and yet each vehicle only ships one package, meaning the utilization of vehicles is quite low. To reduce the number of required vehicles of low utilization, we propose a Hub-and-Spoke logistics network with small autonomous vehicles for serving the first-and-last mile delivery, and transshipment trucks help deliver packages between hubs with some sufficient economical scale. Although the OD delivery framework proposed in this thesis has not appeared in literature, we expect this to be a very good design for future city logistics.

## MATERIALS AND METHODS

Given the estimated amount of OD shipments in each time period during one day, we seek optimal schedules and routes for transshipment trucks between hubs. We consider two scenarios: (1) we assume the autonomous vehicles stay locally for each region of a hub, and (2) we treat the autonomous vehicles as shared vehicles, we can also reposition these autonomous vehicles between hubs in different time periods to increase the utilization of autonomous vehicles and trucks.

For both scenarios, we propose Integer Programs (IPs) on a time-space network to calculate exact optimal solutions, but it is too time-consuming. To solve more practical cases of larger sizes, we propose a greedy algorithm and a Genetic Algorithm to find good feasible truck routes in short time. Then we propose a framework that iteratively solves a smaller time-space network of few periods in a rolling horizon fashion. For example, we may solve a smaller case that covers 6 contiguous periods by IPs, and then only fix the decisions of the first 2 or 3 periods, then iteratively do the same thing starting from the first to-be-considered period for the next 6 contiguous periods. Such a rolling-horizon framework has the advantage of smaller-scale IPs that have looked sufficiently ahead of the near future. For the second scenario, we introduce an innovative idea that treats the small autonomous vehicles as shared vehicles. Indeed, it makes no sense to make some autonomous vehicles idle for a long time, if we already expect the future delivery demands in that region becomes less. Thus how to

“share” these autonomous vehicles in a systematic way is an interesting concept and here we already provide mathematical models for this new delivery mechanism. To solve more practical cases of larger sizes, we propose a greedy algorithm and a Genetic Algorithm (GA) to find good feasible truck routes in short time. In particular, the greedy algorithm can produce a good initial feasible solution, which can be used for GA and IPs. Then we propose a framework that iteratively solves a smaller time-space network of few periods in a rolling horizon fashion, which produces the best results in our computational experiments

## **RESULTS AND DISCUSSION**

Our testing was performed on a personal computer with Windows 10, Intel® Core i7-8700 3.20 GHz\*6 Processors, and 8GB RAM. All the solution methods are implemented in C++ language, compiled by Visual C++ 2015. Gurobi 8.1.1 version is used for solving integer programs. We have tested on cases of 9 regions and 5 trucks, where each region may have on average 10, 20 or 30 randomly generated OD delivery requests. For the cases of more frequent OD shipments, our proposed IP formulation cannot calculate a good solution in 2hrs. Nevertheless, the rolling-horizon mechanism can almost calculate locally optimal solutions in short time for each subset of smaller time-space network. Although the GA also performs better than the IP of the entire planning horizon, it still produces an optimality gap around 10-20%. We also compare the utilization of the two scenarios and confirm the scenario using shared autonomous vehicles does increase the utilization up to 5-10% on average.

## **CONCLUSIONS**

In this thesis, we have investigated expected-to-appear city logistics challenges caused by huge amount of same-day OD shipments. We proposed an HS logistics network in conjunction with small autonomous vehicles to conduct the first-and-last mile delivery between the hub of a region and customers, as well as transshipment trucks that help delivery collected shipments between hubs. In addition to the use of the autonomous vehicles as local vehicles inside a region, we further consider a systematical framework that treat these autonomous vehicles as shared vehicles and calculate optimal reposition strategies to move these autonomous vehicles between hubs in different time periods to increase the utilization of all vehicles, which in turn reduce the traffic jam, accidents, and air pollution. Our computational experiments indicate our proposed rolling-horizon solution framework to be the most effective and efficient one, compared with the IP for the entire planning horizon or the heuristic algorithm such as GA.

## 誌謝

感謝王逸林老師，這些日子以來學業上的指導，以及透過比賽、與公司合作計劃等等，磨練我們，在這無數的開會日子中，也讓我成長了不少，不僅僅是老師的專業知識上的傳授，更是從那堅忍不拔的做事態度中，學到了很多，令我相當的敬佩。

感謝實驗室的學長姊，筱芸、BK、富元、思涵、冠緯，在一開始進入這陌生的環境時，給予了我很多幫忙，尤其是坐在我旁邊的冠緯，感謝你平時的加油打氣，常常在關鍵的時候，把我從深淵拉出來。

感謝同屆的 gayry 跟雪湄，常常 carry 我大大小小的事情，幸好有你們，我才能度過各式各樣的難關。

感謝 lab 的學弟妹，晏慈、彥瑋、宗瀚，很抱歉感覺沒有教會你們太多東西，尤其是晏慈，在群創專案中，常常拖累到你，希望你能早日脫離群創苦海。

感謝碩班的吃飯團，陞瑋、雅宣、裕老師、EJ、葉韋呈，因為有你們，我的生活才可以多了些歡笑與回憶，希望大家以後別輕易退群組啊，往後還有許多地方要一起去玩的啊。

最後感謝我的家人，時常給予我鼓勵與建議，使我可以無後顧之憂完成學業，謝謝！

# 目錄

摘要.....	I
目錄.....	VI
圖目錄.....	IX
表目錄.....	XI
第一章 緒論.....	1
1.1 研究背景與動機.....	1
1.2 研究目的.....	3
1.3 研究架構.....	4
第二章 文獻回顧.....	5
2.1 無人車在城市物流相關文獻.....	5
2.2 軸幅式網路相關文獻.....	6
2.3 小結.....	10
第三章 區域性無人車收送網路設計問題.....	11
3.1 問題描述與假設.....	11
3.1.1 問題描述.....	11
3.2.2 問題假設.....	11
3.2 本研究之整數規劃模式.....	12
3.2.1 參數與變數定義.....	12
3.2.2 數學模式的建立.....	14
3.3 小結.....	19
第四章 共享性無人車收送網路設計問題.....	21
4.1 問題描述與假設.....	21
4.1.1 問題描述.....	21
4.1.2 問題假設.....	21
4.2 本研究之整數規劃模式.....	21
4.2.1 參數及變數意義.....	21



4.2.2 數學模式的建立 .....	22
4.3 小結 .....	23
<b>第五章 區域性無人車收送演算法設計 .....</b>	<b>24</b>
5.1 基因演算法 .....	24
5.1.1 基因演算法步驟 .....	24
5.1.1 染色體的編碼方式 .....	25
5.1.2 交配策略(crossover operation) .....	26
5.1.3 突變策略(mutation operation) .....	28
5.1.4 適應函式(fitness function).....	28
5.1.5 挑選策略(selction operator).....	30
5.2 計算初始解 .....	30
5.2.1 產生初始解步驟 .....	30
5.2.2 抵達下一個點的收貨權重 .....	31
5.2.2 抵達下一個點的送貨權重 .....	31
5.2.3 加入二次轉運的考量 .....	32
5.3 小結 .....	33
<b>第六章 數值分析 .....</b>	<b>34</b>
6.1 區域性無人車數學模式測試 .....	34
6.1.1 視覺化結果與分析 .....	34
6.1.2 分析模式求解情況 .....	37
6.1.3 滾動式模式求解 .....	39
6.2 共享性無人車數學模式測試 .....	40
6.2.1 視覺化結果與分析 .....	40
6.3 區域性無人車演算法測試 .....	42
6.3.1 視覺化結果與分析 .....	42
6.3 小結 .....	43
<b>第七章 結論與未來研究 .....</b>	<b>44</b>
7.1 結論 .....	44

7.2 未來研究 .....	45
參考文獻 .....	47

## 圖目錄

圖 1.1:軸輻式網路示意圖 .....	2
圖 1.2:上午下午集散點需求及貨車進出量示意圖 .....	3
圖 2.1:小型無人車與運送的貨車示意圖 .....	5
圖 2.2:第一類型的覆蓋問題 .....	8
圖 3.1:貨車在時空網路圖下之建構圖 .....	15
圖 3.2:貨物在時空網路下之建構圖 .....	16
圖 5.1:染色體編碼方式 .....	26
圖 5.2:染色體編碼示意圖 .....	26
圖 5.3:第一種交配方式 .....	27
圖 5.4:第二種交配方式 .....	27
圖 5.5:染色體突變前後對照圖 .....	28
圖 5.6:路線追蹤說明 .....	29
圖 5.7:計算收貨權重示意圖 .....	31
圖 5.8:計算送貨權重示意圖 .....	32
圖 5.9:第一台貨車尋找送貨路線示意圖 .....	32
圖 5.10:第二台車尋找送貨路線示意圖 .....	33
圖 6.1:顧客點以及集散點視覺化 .....	34
圖 6.2:貨物運送方式視覺化 .....	35
圖 6.3:貨車路徑視覺化 .....	36
圖 6.4:時空網路圖結果視覺化 .....	36
圖 6.5:測試貨車路線圖 .....	37
圖 6.6:共享性模式視覺化折線圖 .....	41
圖 6.7:資料集 10_1 的結果折線圖 .....	42

圖 6.8:資料集 20_1 的結果折線圖 .....	42
圖 6.9:資料集 30_1 的結果折線圖 .....	43

## 表目錄

表 2.1: 過往文獻比較 .....	10
表 3.1: 三種貨物運送方式 .....	17
表 5.1: 基因演算法步驟 .....	25
表 6.1: 小測資參數設定表 .....	35
表 6.2: 大測資參數設定表 .....	37
表 6.3: 數學模式測試結果 .....	38
表 6.4: 滾動式決定期數成果比較 .....	39
表 6.5: 滾動式與其它方法成果比較 .....	40
表 6.6: 共享性模式結果分析表 .....	41
表 6.7: 基因演算法參數 .....	42

# 第一章 緒論

## 1.1 研究背景與動機

近幾年，『最後一哩路』已經變成物流業者爭相想要解決的一大挑戰，它講述的是一個概念，貨物從零售業者的倉庫送至消費者手中的過程，稱為「最後一哩物流」，有以下幾點，讓最後一哩路的重要性大增：

1. 隨著電商的興起，使用者的消費習慣逐漸改變，從去商場買東西變成線上購物，因此物流業的需求漸漸增高，業者發現，在寄送運送成本中，如所示，最後一哩路所佔的成本居然高達 53 %，因此隨著訂單量漸漸增高，思考如何降低最後一哩路的成本顯得相當重要。
2. 在最後一哩路的成本分析當中，可以得知，人員的成本是所佔最高的，搭上近年無人載具的興起，像是 Amazon 或是其它新創，研發出像是無人機或是無人車的載具，來配送物品，這些方法都可以為最後一哩路帶來新的革命。
3. 隨著消費者習慣線上購物，開始逐漸會要求送貨速度、與取貨地點，像是希望當日送達，或是彈性的取貨時間，這些都大大的增加最後一哩路的困難
4. 隨著電子商務的發達，大量的零售業者冒出，甚至開始有了有別於以往的購物服務，像是即時性的幫買食物，Uber eat 或是 pandafood 等等的食物快遞公司，還是幫你去商城購買日常用品的服務，再送回顧客手中，這些都將會仰賴物流來配送。
5. 民眾對於取貨，比以往更為要求，希望可以當日送達之外，可能還想要限定取貨的時間區間，或是取貨地點，這都會大大增加配送的困難性。

以上重重的因素或是趨勢，都跟物流的最後一哩路環環相扣，也因為這樣的趨勢出現，在城市間將會面臨大量的貨物流進流出，降低運送成本，變成很重要的課題。

然而，本研究將會探討城市間，最初及最後一哩路之相關應用。現今已經有很多服務都用無人車來運送，以郵遞公司或是食物送貨公司為例，人們可以把想要寄的東西交給機器人，或是商家可以透過機器人，把食物送出，這是最初一哩路，之後一樣也是由機器人將貨物送到收件人手中。像是挪威、澳洲郵遞的機器人服務，已經完成一百多次遞送郵件的任務，民眾可以很彈性的選擇交貨時間，如果錯過了交貨時間，居民可以選擇夜間收件，讓機器人在晚間重新遞送包裹或郵件。美國的 doosh 公司也利用機器人，送件包裹、食物或是從零售商那邊送出貨物到客人手中。這些種種的應用已經蓬勃發展了。

本研究想要針對這些 P2P (point to point) 的送貨方式，提出一個軸幅式網路，來降低運送成本。

軸幅式網路 (Hub-And-Spoke) 常運用於貨運航空、快遞包裹、交通網路等的應用，即建立一個或數個轉運中心(Hub)，每個中心都有一個子系統，由中心集中處理某些事務(物流或資訊流)，再由中心的鄰近點(子系統)向外擴散，而中心間互相支援，如圖 1.1 所示。

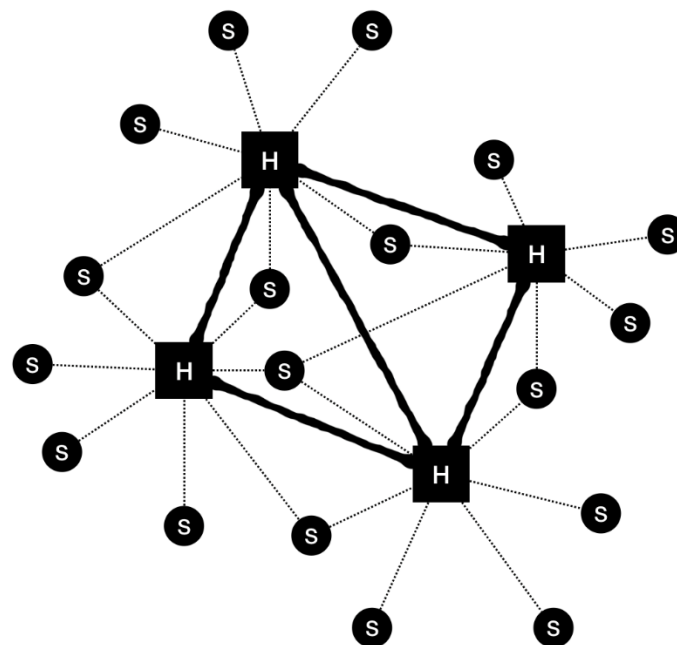


圖 1.1:軸幅式網路示意圖

套用在本研究中，最初一哩路的貨物會由小型無人車(Autonomous Vehicle)先送往集散點(Hub)，再透過之間貨車的運送，把貨物轉運到另一外一個集散點(Hub)，最後再由小型無人車從 hub 取貨送最後一哩路到顧客手中。Rodriguez (2007)曾提過，軸輻式網路(Hub-And-Spoke)的好處是，可以將有經濟規模的大量貨物透過單趟去運送，來減少多趟重複又少量的貨物運送成本，甚至因為是由集散點之間配送、轉運，所以可以更有效地利用機器人、貨車來轉運貨物，達到顧客所要求兼具時間、地點的彈性。

## 1.2 研究目的

本研究主要會搭配小型無人車的使用場景，設計軸輻式網路來滿足城市間的”當日配送“( Same Day Delivery) 需求，為了最小化整體配送的成本，將會透過數學模式以及演算法，求解出以下幾點：

1. 集散點(HUB) 的位置
2. 集散點(HUB)的服務範圍
3. 集散點間貨車的行車路徑與排程

為了要考慮需求隨著時間改變的因素，本研究加入了時間的排程在數學模式中，使得求解出的貨車行車路徑，可以隨著時間而調整，以下圖 1.2 為例，上午的時候，集散點 A、B、C 的需求量很大，所以三個集散點之間的貨車進出量很頻繁，到了下午時，改成集散點 C、D、E 的需求量很大，因此下午的貨車將這三個點進出量次數

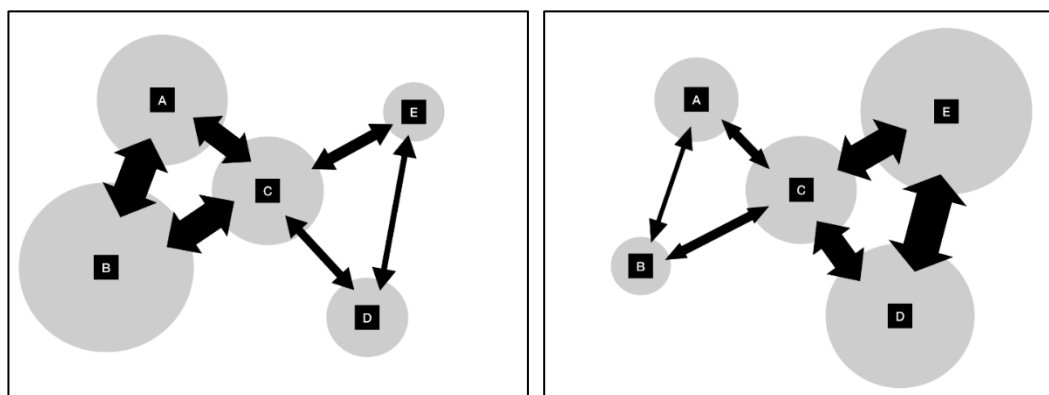


圖 1.2:上午下午發散點需求及貨車進出量示意圖



提高，如此的針對時間需求而進行貨車的排程，更能提升的貨車使用率。

另外，由於小型無人車有數量的上限，如何針對時間需求去調度無人車在集散點的量，讓進出貨量大的集散點可以有較多台的小型無人車來收送貨物，也會在本研究的數學模式中解決此問題，來提升小型無人車的使用率。

## 1.3 研究架構

本研究之架構如下：第二章的文獻回顧，會介紹無人車在城市物流的相關文獻，以及軸輻式網路的相關文獻。在第三章中，會詳細介紹區域性無人車收送網路的數學模式。在第四章中，會將第三章的問題延伸，探討當區域性的無人車能夠共享時，是否可以降低無人車的總體數量，以及提升使用率。在第五章中，由於第三章的數學模式利用時空網路所建的變數在大量測資時，求解效率不彰，因此在本章會構建基因演算法去求解，並且設計找尋初始解的方式，來快速求得品質好的解。在第六章中，會使用大量的測資來分析比較前幾章所建構的模式是否有合理的給出解答，並且跟演算法比較其求解的品質與效率。

## 第二章 文獻回顧

本章文獻回顧主要分成兩個部分，第一個部分會先講無人車在城市物流的相關文獻，第二部分會講述軸輻式網路(Hub-And-Spoke)的相關文獻：

### 2.1 無人車在城市物流相關文獻

Boysen, Schwerdfeger and Weidinger (2018)這篇，講述在最後一哩路的城市物流，目前有相當多種運送方式，像是電動車運送、用無人機為主的載具來運送與人力運送。其中，他提出了以貨車為主的無人車運送方式，如圖 2.1 所示，有別於上述幾項。概念是來自於一家德國的新創公司 Daimler (2017)，這家公司開發了一台能夠行走在接到上面的小型無人車，負重不超過 40 磅，他可以運送貨物、食物從甲地到乙地。當送達目的地時，使用者能夠透過智慧型手機去解鎖來取得貨物。基於安全原則，這種小型無人車速度只限於跟行人一樣的慢速行駛，因此當出發地跟目的地距離太遠，這種運送方式便顯得相當沒效率。為了避免這個缺點，本篇提出了用貨車來改善這種運送方式，讓貨車接收數台小型無人車，載往目的地的附近時，再放下無人車來送最後一哩路，進而改善距離太遠的問題。

這種以貨車為主的無人車運送原理如下，貨車會先在集貨中心把顧客待運送的貨物上車，另外在卡車內部固定空間內放置小型無人車。接下來貨車會開始遊走在城市



圖 2.1: 小型無人車與運送的貨車示意圖

中，到了預訂的派送區，貨車會放下一台至多台的無人車，讓它們去完成最後一哩路的運送。每一台無人車只會把貨物單次送給一個客人，完成任務後，便會回到分散式的無人車集中站，此站只會存放無人車，不會存放貨物，因此不需要太多的放置空間。貨車用這樣的模式訪問無人車的派送區，每次的派送區不會固定，會取決於車上貨物的目的地。假使車內的無人車以派送完畢，車上還有貨物待運送，便會先到其中一個無人車集中站來放上一批無人車，再繼續進行派送，直到車內的貨物送完畢後，再回到集貨中心，等待下一批要派送的貨物。

這種子母車去送貨的概念，在無人機送貨的領域，很常被廣泛討論，然而在面對較大需求時，這種設計無法應付太大量的起訖點流進流出，由於倉儲只有一個，會導致往返的車輛數需要很多，因此較軸輻式網路沒效率。

## 2.2 軸輻式網路相關文獻

軸輻式網路(Hub-and-spoke)最早的應用，是來自於航空網路方面的應用，隨著近年來發展，開始應用在其它的地方，像是郵遞問題、送貨問題、海運問題、通訊問題等等，在本章節會針對他在送貨方面的情境來加以討論：

集散點(Hub)是一個很重要的角色在這個網路圖，要同時收集貨，並且要分配到其它的集散點，O'Kelly (1986)提出說在集散點上面的考量主要有兩大點，分別是要決定有多少個集散點，以及要放在哪邊。大部分的研究，都會把這兩個要素同時考慮，部份為了降低難易度，會事前決定好集散點的數量，這種被統稱” p-Hub” 的問題。

當討論到集散點的涵蓋範圍，以及要如何指派集散點 (Hub) 給非集散點 (Non-Hub) 服務時，這些議題被稱作為集散點覆蓋問題(Hub Covering Problem)，在過往的研究中， Bryan and O'Kelly (1999)將網路圖分成了單一指派集散點與多重指派集散點這兩種，前者是指在模式中非集散點只能與一個集散點相連，多重則是允許有非集散點連線到多個集散點。在集散點與集散點中的考量中，他們設計了一個的折扣係

數 (InterHub discount factor) 在模式裡面，表示集散點之間互動的運輸規模經濟性，將集散點間因集中貨源而產生規模經濟性，其成本降低之比值定義為折扣係數，在相關軸輻路網模式中也都以折扣係數來表現出集散點之間的規模經濟性。Campbell (1994) 、Karimi and Bashiri (2011) 針對集散點覆蓋問題(Hub Covering Problem)有提出以下的分類：

1. 如果成本(時間或是距離)從  $i$  到  $j$  透過  $k$  還有  $m$  ( $C_{ij}^{km} = C_{ik} + \alpha C_{km} + C_{mj}$ ) 並沒有大於一個特別的值( $\beta$ )，因此起訖點( $i, j$ )分別被集散點  $k$  和  $m$  覆蓋(服務)
2. 如果成本(時間或是距離)對於各段在路徑  $i$  到  $j$  透過  $k$  和  $m$  ( $C_{ik}, \alpha C_{km}, C_{jm}$ ) 並沒有大於一個特別的數( $\theta$ )，因此起訖點( $i, j$ )分別被集散點  $k$  和  $m$  覆蓋(服務)
3. 如果成本(時間或是距離)對於各段在路徑  $i$  到  $j$  或是  $m$  到  $j$  ( $C_{ik}, C_{jm}$ ) 並沒有大於一個特別的數( $\gamma$ )，因此起訖點( $i, j$ )分別被集散點  $k$  和  $m$  覆蓋(服務)

其中， $\alpha$  為折扣係數。大部分的研究都是以第一種型別為主，然而第一種型別會遇到像是圖 2.2 所發生的問題，雖然規範了距離或是時間在一定的範圍下，還是會出現這三段中其中一段過長的問題。在這種集散點的位置設計問題，會隨著應用場景，有著不同需求的設計，像是快遞系統等的，就會需要第一種型別的限制，才可能盡量快速把包裹送到顧客手上。如果是第二種型別，比較會應用到的場景為旅途有特殊考量限制的，像是飛機航班來說，就會有燃料耗損的限制。第三種型別的應用場景為像是捷運這種大眾運輸工具，集散點與集散點之間的時間並不是最重要的考量，而是要考量到使用者到捷運站點的距離以及時間。若套用在本研究來看，會發現我們的使用場境融合了第一種型別跟第二種型別，為了達到當日配送的目標，需要限制出發與抵達的所需時間，另外在最初及最後一哩路上，無人車的配送因為有電力限制，有一定的旅程範圍，因此會同時考量這兩種限制。



圖 2.2:第一類型的覆蓋問題

在前述討論到的折扣係數，最一開始是 Skorin-Kapov, Skorin-Kapov and O'Kelly (1996)提出，對於集散點位置問題有很大的關係，由於這樣的設計才能讓數學模式目標式找出有流量大的位置設成集散點，使得最小化成本時，能選到較佳的位置點。然而在係數上的設定，通常是用一個不大於 1 的一個定值，作者利用航空公司的資料來進行測試發現，隨著係數越大，越容易出現多重指派，越小則是偏向單一指派，如果係數為 0 的話，最佳解將會變成單一指派，也就是每個非集散點只會有一個集散點。

在集散點跟集散點之間的連線，也分成三種，分別是全部線皆連，第二個是用樹狀圖的方式連，再來是部分連線，大部分的研究皆是全部連線皆可連，本研究也是各個集散點都可連，利用混整數規劃模式來規劃最終會有連線的集散點

在軸幅式網路設計的過程中，主要有兩大問題必須考慮，分別是“集散點的位置問題”跟“路線的排程問題”。這兩個問題在以往的文獻中，主要可以分成三種啟發式方法去解決這樣的問題：

1. 連續式方法 (successive methods)
2. 迭代方法 (iterative methods)
3. 整合式方法 (integrated or simultaneous approach)

第一種其實就是先決定好集散點的位置，再用集散點的位置去規劃出送貨路徑出來，在這個方法中，主要可以分成兩個面向，分別是 locate-route 或是 route-locate，也就是這兩個問題的決定順序。另外，這種的壞處是沒有回饋機制；在第二種迭代法時，會重複的在位置問題跟途程問題上面求解，相較於第一種，更可以趨向最佳解；第三種則是同時考慮。

Wasner and Zapfel (2004) 認為有必要同時考慮“集散點的位置問題”跟“路線的排程問題”，因此建立多項式規模下的混整數線性規劃模式。

Nagy and Salhi (1998) 就有針對多對多的區位途程問題建立混整數規劃模式，此外，也同時利用了啟發式的連續法去解決小型規模的問題。Wasner and Zapfel (2004) 這篇考慮了兩層的倉儲，有一個中央倉儲再加上其餘的倉儲，並且提前決定好集散點的數量，這種形式是相較於理論來說，更為實際的建置方式，在文中最後，這樣的設計利用啟發式的整合式方法來計算後，節省了 14.7 % 的成本。

Çetiner, Sepil, and Süral (2010)的應用場景為郵遞系統，該研究不使用數學規劃的模式，僅使用啟發式的迭代法去做。首先先決定倉儲設置點以及哪些郵局要被分配給倉儲，下一步則是找出分配給同樣倉儲的郵局中要走的路線。

Karaoglan, Altiparmak, Kara, and Dengiz (2012)同時考慮“集散點的位置問題”跟“路線的排程問題”，並且分析了兩種不同混整數規劃的方式去解決問題，分別是 node-based 跟 arc-based，作者在這篇建議使用啟發式方法的連續法，並且採取 locate-route 的策略，分別針對 node-based 跟 arc-based 求解，用了 360 個測資，結果顯示 arc-based 的解品質會高於 node-based，所花費時間也較短，但是 node-based 可處理較大規模測資。

Camargo et al. (2013)假設非集散點(Non-Hub)的點可以被分配給多於一個的集散點，意思說可以被兩個以上的集散點所服務，貨物在集散點之間的轉運，最多只會經過兩個集散點，這種假設會導致集散點跟集散點之間幾乎是全部連結，運送的成本會提高，這篇也利用了一個班德氏分解法(Benders Decomposition)的方式去解決這種多對多型式集散點路徑位置問題，這種的方式是相當有效率去解決這個問題。

Rodríguez-Martín, Salazar-González and Yaman (2014)在解決多對多的位置路徑問題中，把每個點都當成潛在的集散點，事先訂出一個集散點要設立的數量，並假設每個集散點沒有建置成本，並且都可以互相連結，每個集散點限制好最大可以服務的非集散點，並且限制有一台車可以訪問集散點服務範圍內的點。

在 Karimi (2018)中的情境，是最符合本研究的假設，這篇強調不但有同時處理收貨跟送貨問題之外，也給予了每個貨物限制他出發跟抵達的時間有一定的限制，不過

在路線規劃上面，並沒有排程。

比較上述的文獻後可以發現，本研究的不同之處是在於考量了時間需求的變化，  
 所多考慮的車輛路徑排程，如表 2.1 所示：

表 2.1:過往文獻比較

	<b>Cetiner (2010)</b>	<b>Karaoglan (2012)</b>	<b>De Camargo (2013)</b>	<b>Rodriguez- Martin (2014)</b>	<b>Hossein Karimi (2018)</b>	<b>本研究</b>
<b>非集散點 分配方式</b>	多重	單一	單一	單一	單一	多重
<b>事前定義 集散點個數</b>	-	-	-	V	-	-
<b>事前定義 最大車輛數</b>	-	-	V	V	-	V
<b>集散點 建置成本</b>	-	V	V	-	V	V
<b>集散點 存貨上限</b>	-	V	-	-	V	V
<b>起訖點 時間限制</b>	-	-	-	-	V	V
<b>考慮依時間 變化的需求</b>	-	-	-	-	-	V

## 2.3 小結

軸輻式網路目前尚未有考慮時間變化的需求，來決定集散點以及路線的規劃，隨著未來趨勢的發展，大量的數據可以收集，如果能夠針對需求變化來決策的話，更能節省不必要的成本，提高轉運站跟車輛的使用率。以上午下午的流量為例，如果 A 地上午是量的高峰，下午則是 B 地流量的高峰，如果有針對需求的路線排程，就可以針對流量高峰的地區排程較多車輛去滿足需求，使其更貼近現實的需求。

## 第三章 區域性無人車收送網路設計問題

本章節將會限制無人車在區域範圍內收送，並且建構軸輻式網路來轉運貨物，來完成城市內的物流。由於要考慮時間變化在數學模式裡，本研究利用時空網路的概念，來限制每個時刻下的動作，來完成同時規劃好集散點的位置，以及集散點間的路線設計與排程問題。

### 3.1 問題描述與假設

#### 3.1.1 問題描述

在一個城市內，有各式各樣的收送訂單  $p$ ，每筆訂單  $p$  會在某個時間區間  $t_p$  中發起需求，要從出發地  $u_p$  送  $Q_p$  的貨物數量到目的地  $v_p$ ，這批貨物會有三種運送方式，一種是會先由小型無人車去接收貨物，當貨物被送去所屬集散點後，會再集散點之間被貨車  $k$  轉運，當送到目的地  $v_p$  所屬的集散點後，再由當地的小型無人車將貨物送到收件人手上；一種則是在集散點管制範圍內小型無人車的直送；另一種則是會由送貨人員會從出發地到目的地的直送。在這一整個的收送過程中，本研究想要得出以下的最佳決策，來最小化總體的成本：

1. 城市內滿足需求的最少集散點數
2. 集散點的最佳建置位置
3. 隨著時間需求下的集散點間貨車路線最佳排程
4. 顧客指派給集散點的最佳分配

#### 3.2.2 問題假設

針對本問題的情境，提出下列假設：

1. 已知各個顧客點在各個時間區間下的需求(目的地、數量)



2. 皆不考慮貨物的上下車搬運時間
3. 不允許貨物整體的運送時間超過一個定值
4. 送貨人員的數量沒有上限
5. 小型無人車可以在集散點內換好全滿電力
6. 小型無人車送完貨物後，會直接回去集散點
7. 小型無人車在集散點中沒有數量上限
8. 小型無人車在集散點收送貨的時間皆會在一期內完成
9. 每筆訂單的量不會超過小型無人車負擔的上限
10. 在集散點中，待轉運的貨物有存貨上限
11. 集散點有一定的服務範圍(同等於小型無人車最大行駛範圍)

## 3.2 本研究之整數規劃模式

### 3.2.1 參數與變數定義

集合

$H$	集散點的集合
$C$	顧客點的集合
$P$	訂單的集合
$K$	貨車 $k$ 的集合
$A$	所有節線的集合， $A = \{(i, j): i, j \in H, C\}$

參數

$T$	時間總期數，其編號 $t = 0, 1, \dots, T, T+1$
$t_{ij}$	從節點 $i$ 到節點 $j$ 所花費的時間期數，其中 $(i, j) \in A$ ， $i = j$ 點則期數為 1
$Q_p$	每筆訂單的量

$Kc$	每台貨車 $k$ 的載貨上限
$Hc$	每個集散點存放貨物的上限
$u_p$	訂單 $p$ 的出發點顧客編號 $u$
$v_p$	訂單 $p$ 的目的點顧客編號 $v$
$t_p$	訂單 $p$ 的發起時刻在第 $t$ 期間
$Kt$	貨車的行駛成本
$Et$	送貨人員的行駛成本
$Rt$	小型無人車的行駛成本
$Hs$	集散點的建置成本
$Hr$	集散點的服務範圍限制(時間)
$L$	貨物整趟旅程耗費的時間期數上限

#### 變數

$x_{ij}^{tk} \in \{0,1\}$	第 $t$ 期末，有編號 $k$ 的貨車從集散點 $i$ 出發，往集散點 $j$ 為 1，貨車 會在第 $t+t_{ij}$ 期末抵達；反之為 0
$y_{ij}^{tp} \in \{0,1\}$	第 $t$ 期末，有訂單 $p$ 被運送從集散點 $i$ 出發，往集散點 $j$ 為 1，訂單 會在第 $t+t_{ij}$ 期末抵達；反之為 0
$a_i \in \{0,1\}$	集散點 $i$ 有被建置為 1；反之為 0
$z_{ui} \in \{0,1\}$	顧客點 $u$ 有被分配給集散點 $i$ 服務為 1；反之為 0
$e_p \in \{0,1\}$	訂單 $p$ 被送貨人員直送為 1；反之為 0
$m_p \in \{0,1\}$	訂單 $p$ 由小型無人車直送起訖點為 1；反之為 0

其中，只要是小型無人車或是貨車，都是期末出發與期末抵達，訂單的發起會歸類在期初發生。舉例來說，如果訂單的在第  $t_p=1$  期發起，會被歸類到第  $t=1$  期初，假

設訂單的出發地 $u$ 到所屬集散點 $i$ 要花費 1 期，小型無人車會在第 $t=1$ 期末從集散點出發，在第 $t=2$ 期末到達出發點，並且收取貨物將它載到集散點 $i$ ，因此 $y_{di}^{2p}=1$ ，接連著在第 $t=3$ 期期末抵達集散點 $i$ 。如有車子在那邊待命，可以趕在第三期末發車前將貨物上車，進行下一步的轉運。

### 3.2.2 數學模式的建立

此模式的目標式有三種成本要考量，分別是行駛成本、集散點建置成本以及超載成本，其中行駛成本有分貨車、小型無人車、送貨人員三種不同的成本計算，將這些成本寫成數學表達式如下。其中(3.2)前式考量到往返最初與最後一哩路，須乘以兩倍，後式考量如果訂單起訖點皆屬於同個集散點的管制範圍，小型無人車可以直接從集散點出發，直送起訖點，再回集散點。在(3.3)中，只考慮送貨人員往返的成本。

- 貨車行駛成本: 
$$Kt \cdot \sum_{t=1}^{T-1} \sum_{k \in K} \sum_{(i,j) \in A} t_{i,j} \cdot x_{i,j}^{t,k} \quad (3.1)$$

- 小型無人車行駛成本: 
$$Rt \cdot \sum_{p \in P} \left( \sum_{t=1}^T \sum_{j \in H} 2t_{u_p,j} \cdot y_{d,j}^{t,p} + \sum_{t=1}^T \sum_{i \in H} 2t_{i,v_p} \cdot y_{i,d}^{t,p} \right) +$$
  

$$Rt \cdot \sum_{p \in P} \sum_{i \in H} m_i^p \cdot (t_{i,u_p} + t_{u_p,v_p} + t_{v_p,i}) \quad (3.2)$$

- 送貨人員行駛成本: 
$$Et \cdot \sum_{p \in P} e_p \cdot t_{u_p,v_p} \quad (3.3)$$

- 集散點建置成本: 
$$Hs \cdot \sum_{i \in H} a_i \quad (3.4)$$

$$\begin{aligned} \min \quad & Kt \cdot \sum_{t=1}^{T-1} \sum_{k \in K} \sum_{(i,j) \in A} t_{i,j} x_{i,j}^{t,k} + Rt \cdot \sum_{p \in P} \left( \sum_{t=1}^T \sum_{j \in H} 2t_{u_p,j} \cdot y_{d,j}^{t,p} + \sum_{t=1}^T \sum_{i \in H} 2t_{i,v_p} \cdot y_{i,d}^{t,p} \right) + \\ & Rt \cdot \sum_{p \in P} \sum_{i \in H} m_i^p \cdot (t_{i,u_p} + t_{u_p,v_p} + t_{v_p,i}) + Et \cdot \sum_{p \in P} e_p \cdot t_{u_p,v_p} + Hs \cdot \sum_{i \in H} a_i \end{aligned} \quad (3.5)$$

The diagram illustrates a sequence of states over time steps  $t=0$  to  $t=T+1$ . The nodes are arranged in a grid with rows labeled 1, 2, 3, ..., H and columns labeled  $t=0$ ,  $t=1$ ,  $t=2$ ,  $t=3$ ,  $t=4$ , ...,  $t=T$ ,  $t=T+1$ . A source node 'd' (square) is present at  $t=0$  and  $t=T+1$ . Arrows indicate transitions between nodes across time steps. A horizontal line separates the top part (nodes 1, 2, 3, ..., H) from the bottom part (nodes d, 1, 2, 3, ..., H).

從上述的舉例可以得知，如果有第 $k$ 台車訪問某個集散點，必定會從那個點繼續

往下個集散點移動，也就是流量流進等於流出的概念，如(3.6)所示，因集散點  $i$  到  $j$  的時間為  $t_{i,j}$ ，左式所連到的會是從第  $t-t_{ij}$  期出發，且  $t-t_{ij} \geq 1$ 。

$$\sum_{i \in H} x_{ij}^{(t-t_{ij})k} = \sum_{r \in H} x_{j,r}^{tk} \quad \begin{matrix} \forall t = 2, \dots, T-1; k \in K; j \in H; \\ t-t_{ij} \geq 1 \end{matrix} \quad (3.6)$$

另外，針對第  $t=0, T+1$  期，必須限制如(3.7), (3.8)所示，限制貨車須從虛擬點  $d$  流進與流出，並且如(3.9), (3.10)所示，要特別對第  $t=0$  期與第  $t=T$  期做流量平衡，才能夠跟(3.6)的流量平衡結合。

$$\sum_{j \in H} x_{dj}^{0k} = 1 \quad \forall k \in K \quad (3.7)$$

$$\sum_{i \in H} x_{id}^{Tk} = 1 \quad \forall k \in K \quad (3.8)$$

$$x_{di}^{0k} = \sum_{j \in H} x_{ij}^{1k} \quad \forall k \in K; i \in H \quad (3.9)$$

$$\sum_{i \in H} x_{ij}^{(T-t_{ij})k} = x_{jd}^{Tk} \quad \forall k \in K; j \in H; T-t_{ij} \geq 1 \quad (3.10)$$

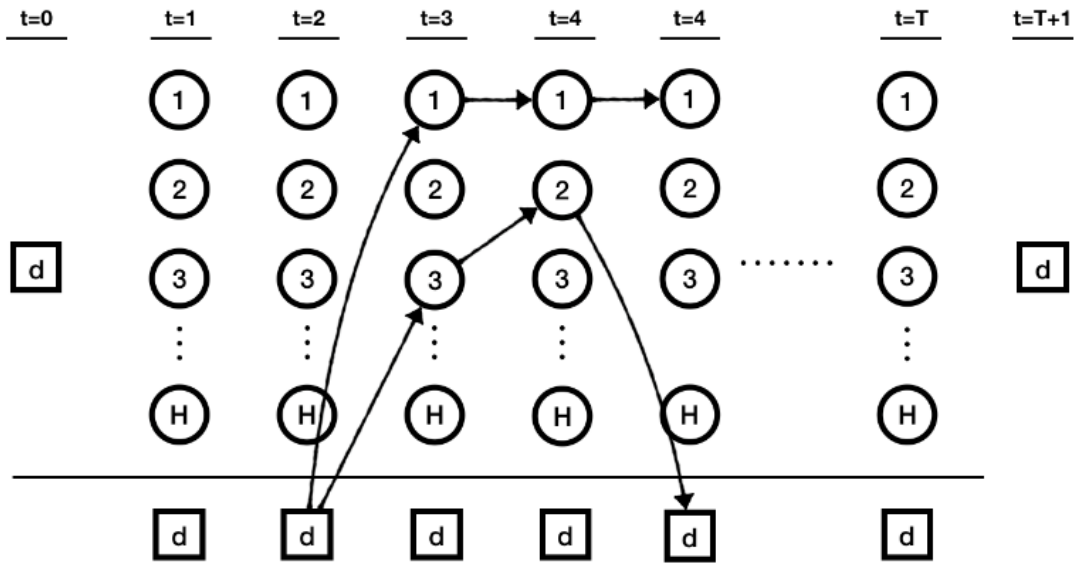


圖 3.2: 貨物在時空網路下之建構圖

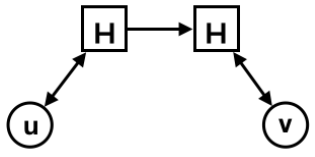
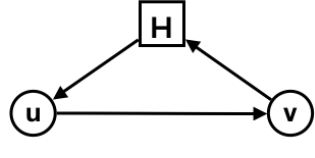
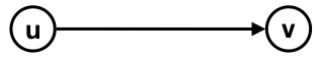
針對貨物相關的限制式，從貨物的時空網路圖中，如圖 3.2 所示，如果有訂單從第  $t=1$  期到第  $t=T-1$  期的虛擬點  $d$  流出，帶表有被小型無人車服務，送至集散點，

若從集散點流回虛擬點  $d$ ，則代表小型無人車把貨物載回目的地。舉裡來說，如果第  $t=1$  期，有一個訂單發起，假設距離最近的是集散點 1 且距離為  $t=1$ ，會在第  $t=2$  期的時候從虛擬點  $d$  流到集散點 3，代表有小型無人車送之過去。接下來有貨車送至集散點 2，如果剛好也是目的地的集散點，便送由小型無人車送至目的地。另外，如果有當貨物沒有車子被載到下一個集散點時，會繼續停留在當下的集散點，像是第  $t=3$  期有一個訂單，因為集散點 1 沒有車子，所以在第  $t=4$  期繼續停留在集散點 1。

流量守恒的部分與貨車的限制式雷同，限制式如(3.11)所示，也是需要  $t-t_{ij} \geq 1$  的限制。比較不同的是，這邊可以直接把虛擬點  $d$  的起訖點考慮進去，因為每一期貨物都有可能從虛擬點  $d$  流進集散點之間，或是從集散點之間流回。

$$\sum_{i \in H} y_{ij}^{(t-t_{ij})p} = \sum_{r \in (H \cap d)} y_{jr}^p \quad \forall t = 2, \dots, T-1; p \in P; j \in H \quad t - t_{ij} \geq 1 \quad (3.11)$$

表 3.1:三種貨物運送方式

運送方式	觸發條件	示意圖	限制式表示
貨車轉運	1. 當訂單起迄點隸屬不同集散點		$\sum_{i \in H} y_{d,i}^{(t_p + t_{u,i}),p} = 1$
無人車直送	1. 當訂單起迄點都是隸屬同個集散點		$\sum_{i \in H} m_i^p = 1$
送貨人員直送	1. 當起起點其中一個點沒隸屬於任一集散點 2. 貨物來不及送到 3. 使整體送貨成本降低時		$e_p = 1$

貨物的運送方式除了到集散點轉運之外，也有另外兩種可能，分別是小型無人車直接直送與送貨人員直送，觸發條件如表 3.1 所示，由於只有轉運才需要規劃貨物流動路徑，因此在時空網路圖的部分中只有呈現貨車轉運的方式，另外兩種會分別由

$e_p, m_i^p$  這兩個變數代表之。

針對這三種貨物運送方式，必須選其中一種，如(3.11)所示

$$\sum_{i \in H} y_{di}^p + \sum_{i \in H} m_i^p + e_p = 1 \quad \forall p \in P; t=t_p \quad (3.11)$$

針對貨車轉運類型的限制式，如(3.12)所示，當訂單起點  $z_{ui}=0$ ，代表著這個貨物

起點  $u$  並不屬於集散點  $i$ ，因此  $m_i^p, y_{di}^p$  應要為 0。

$$m_i^p + y_{di}^p \leq z_{ui} \quad \forall p \in P; i \in H; u = u_p; t = t_p \quad (3.12)$$

在(3.13)式子中，如果起點是要送去集散點轉運的話(左式等於 1)，那右式需要限制須有一條從集散點回來。在(3.14)限制如果目前訂單  $p$  待的集散點不是目的地所歸屬的集散點，不能回虛擬點  $d$ 。

$$\sum_{i \in H} y_{di}^p = \sum_{t=2}^{T-1} \sum_{i \in H} y_{id}^p \quad \forall p \in P; t = t_p \quad (3.13)$$

$$\sum_{t=2}^{T-1} y_{id}^p \leq z_{vi} \quad \forall p \in P; i \in H; v = v_p \quad (3.14)$$

針對小型無人車直送，由於這個方法行駛成本較低，所以如果三種方式可以選時，會直接選擇此方案，因此只須限制如果起訖點所歸屬的集散點不相同時，就不行用這個方法運送，如(3.15)所示。

$$2 \cdot m_i^p \leq z_{ui} + z_{vi} \quad \forall u = u_p; v = v_p; i \in H; p \in P \quad (3.15)$$

針對送貨人員直送，需由(3.16),(3.17)限制，當起迄點其中一個沒被歸屬於集散點，就一定會用此方法，若不是，也是有可能使用。

$$e_p \geq 1 - \sum_{i \in H} z_{ui} \quad \forall p \in P; u = u_p \quad (3.16)$$

$$e_p \geq 1 - \sum_{i \in H} z_{vi} \quad \forall p \in P; v = v_p \quad (3.17)$$

由於需要限制訂單  $p$  完成的時間，因此如(3.18)所示，最後抵達目的地時間減上發起時間需小於  $L$ 。

$$\left(\sum_{t=1}^T \sum_{i \in H} [y_{id}^{tp} \cdot (t+1)]\right) - t_p \leq L \quad \forall p \in P \quad (3.18)$$

在時空網路圖中，目前有貨車跟貨物在上面流動，需要限制如果有貨物在集散點中移動，其中一定是被某台貨車所載送，如(3.19)所示，如果有一台車子  $k$  開往集散點  $i$  到  $j$ ，可能部分或全部的訂單  $p$  會移動從集散點  $i$  到  $j$

$$\sum_{k \in K} Kc \cdot x_{ij}^{tk} \geq \sum_{p \in P} Q_p \cdot y_{ij}^{tp} \quad \begin{array}{l} \forall (i, j) \in A; t = 1, \dots, T-1; k \in K \\ i \neq j \end{array} \quad (3.19)$$

針對集散點的限制有兩個，分別是否要啟用如(3.20)所示、限制集散點對於顧客點的最大距離範圍(因小型無人車距離限制)如(3.21)

$$M \cdot a_i \geq \sum_{u \in C} z_{ui} \quad \forall i \in H \quad (3.20)$$

$$t_{ui} \cdot z_{ui} \leq 1 \quad \forall u \in C; i \in H \quad (3.21)$$

當不一定每個集散點都會被建置時，須限制貨車步行行駛那些沒建置的集散點，如(3.22)所示

$$M \cdot a_j \geq \sum_{k \in K} \sum_{t=1}^T \sum_{i \in H} x_{ij}^{tk} \quad \forall j \in H \quad (3.22)$$

### 3.3 小結

針對時空網路的建置，在上面的線段有分成兩種，一種是貨車的線段，另一種是貨物的線段，這兩種不但有自己的路線要跑，還會因為貨物需要被貨車所載，而互相牽制住，透過(3.19)的限制式，來加以合併者兩種線段的流動。

在設計此數學模式時，曾經考慮使用層空網路(Level-Space Network)，不過因為假設的情境需要時間排程，層空網路的表示較屬於考慮貨車跟貨物每一步的移動，無法看出時間前後的關係，需要加以用更多的變數來表達時間的移動，設計上會變得更困難。然而時空網路(Time-Space Network)雖然可以直接表示時間的移動在網路圖上，



但是由於要針對每個時間區隔都設計一批變數，因此當時間拉很長，或是時間區隔很短的話，都會造成模式上有更多的變數需要解決跟考慮，求解時間上面或許會被拉長。

目前本章的數學模式，是假設小型無人車在每個集散點的數量無上限，並且只能在自己的服務範圍內移動。在下個章節所介紹的數學模式，就會將假設改成有限的小型無人車之下，不但要如何滿足自己服務範圍的需求，還需要設計調度的方式，來讓此資源能夠有效的被分配，提高小型無人車的使用率。

## 第四章 共享性無人車收送網路設計問題

在本篇章，會先利用第三章的數學模式得出貨車的路線以及貨物收送時間，再加以規劃各個集散點空間的小型無人車，藉由貨車運送至其它的集散點服務其他需求，此做法可以達到兩個好處：

1. 提高小型無人車的使用率
2. 應付特定集散點相較高峰的送貨需求，來降低整體的無人車數量

### 4.1 問題描述與假設

#### 4.1.1 問題描述

在一個城市內，已知有集散點  $h$  會在某  $t$  期要派出小型無人車收貨或送貨，並知道有貨車  $k$  會在某  $t$  期出發或抵達集散點，透過小型無人車的在集散點之間的共享性，來找出最小成本的小型無人車總體數量，來滿足一整天的需求。

#### 4.1.2 問題假設

針對本問題，提出下列假設：

1. 已知一整天每台貨車的送貨路線
2. 已知每個集散點每一期的收送貨需求數
3. 不考慮小型無人車的搬上下貨車的時間
4. 小型無人車在集散點收送貨的時間皆會在一期內完成
5. 小型無人車可以在集散點內換好全滿電力

### 4.2 本研究之整數規劃模式模式

#### 4.2.1 參數及變數意義

集合

$H$	集散點的集合
$K$	貨車 $k$ 的集合
$A$	所有節線的集合， $A = \{(i, j): i, j \in H, C\}$

參數

$D_h^t$	某集散點 $h$ 在某第 $t$ 期的小型無人車需求數量
$KS_{ij}^t$	第 $t$ 期從集散點 $i$ 出發往集散點 $j$ 的貨車空間空間數
$WC$	小型無人車成本
$WS$	小型無人車空間大小單位量

變數

$w_{ij}^t$	第 $t$ 期從集散點 $i$ 出發往集散點 $j$ 的小型無人車數
------------	------------------------------------

#### 4.2.2 數學模式的建立

此模式會利用時空網路，來找出小型無人車每期在各個集散點的流動情況，與第三章的貨物或是貨車流動的不同在於，不需要針對任一台小型無人車做追蹤，只需知道每個集散點每期有多少台車即可，來滿足當下的收送貨需求量。

在目標式方面，要找出滿足需求的最少小型無人車總量，總量的表示可以找出期初從虛擬點  $d$  流入多少量進入各個集散點，如(4.1)所示：

$$\min \sum_{j \in H} w_{dj}^0 \times WC \quad (4.1)$$

在限制式方面，首先要讓小型無人車進入與進出集散點的流量守恒，如(4.2)所示：

$$\sum_{i \in H} w_{ij}^{(t-t_{ij})} = \sum_{r \in H} w_{jr}^t \quad \forall t = 2 \dots T-1; j \in H; t - t_{ij} \geq 1 \quad (4.2)$$

針對流量守恒，由於頭尾要從虛擬點  $d$  流進與流出，所以要另外設限制式來綁定流量守恒，如(4.3), (4.4)所示：

$$w_{di}^0 = \sum_{j \in H} w_{ij}^1 \quad \forall i \in H \quad (4.3)$$

$$\sum_{i \in H} w_{ij}^{(T-t_{ij})} = w_{jd}^T \quad \forall j \in H \quad (4.4)$$

小型無人車需滿足各集散點各期的需求量，如(4.5)所示：

$$w_{ii}^t \geq D_i^t \quad \forall i \in H; t = 1 \dots T-1 \quad (4.5)$$

小型無人車若可以從集散點移動到另一個集散點，需要有貨車經過且有一定大小的空間空間，空間空間如(4.6)所示來限制，並用(4.7)來表示沒貨車經過的路線，無人車也不行移動過去。

$$w_{ij}^t \leq KS_{ij}^t \quad \forall i, j \in H; t = 1 \dots T-1 \quad (4.6)$$

$$w_{ij}^t = 0 \quad \forall t = 1 \dots T-1; i, j \in H; t, i, j \notin KS \quad (4.7)$$

### 4.3 小結

原先在本章的研究範圍，是想要在小型無人車共享性的考量中，同時也考慮貨車路徑選擇跟集散點選點，但是在第三章中所建構的數學模式，已經無法實際地用大資料去解(因為時空網路所產生的大量節線)，折衷的用了第三章的結果，來縮小本章的研究範圍，讓共享性考量的數學模式可以在較小的解空間內解決此問題。

另外，在設計本章的數學模式時，只需要管控小型無人車在時空網路上所有節線的流量，而不需要清楚地知道每一台小型無人車的路線，原因在於小型無人車是沒有差異的，但是在第三章的時空網路圖上，假使貨車上現在有三個貨物，需要知道這三個貨物的訂單編號是哪一個，才能知道它的目的地在哪，因此貨物不能跟小型無人車一樣是視為一樣的個體。

## 第五章 區域性無人車收送演算法設計

在第三章建構的數學模式，會因為期數的增加與貨物數量的上升，讓限制式變得相當龐大，求解時間變得相當不實際。因此在本章，會在 5.1 節提出基因演算法去求解此問題，另外在 5.2 節，會利用貪婪式演算法，去求得初始解，再放入加入基因演算法去求解此問題

### 5.1 基因演算法

基因演算法最初是借鑑了進化生物學中的一些現象而發展起來的，這些現象包括交配、突變、自然選擇，透過染色體不斷的變異，來尋找組合最佳解。接下來會介紹本演算法如何找出貨車的行駛路徑。

#### 5.1.1 基因演算法步驟

基因演算法分成好幾大步驟，分別是產生染色體、評估染色體、選擇染色體、變異染色體(利用交配跟突變)，透過不斷迭代，讓好的染色體中好的基因可以留下來，使解的品質更高。在進行這些步驟時，有幾個參數可以做調整，分別是：

- $N_{pop}$ ：染色體個數
- $P_c$ ：發生交配的機率
- $P_m$ ：發生突變的機率
- $T_{stop}$ ：當演算法經過  $t_{stop}$  時間後，停止演算法
- $I_{stop}$ ：當連續經過  $i_{stop}$  次的迭代無法找到最佳解後，停止演算法

停止的條件其實可以有很多種方法，本研究用了兩種方式來當作停止的條件，來方便進行比較。下表 5.1 為基因演算法的流程

表 5.1: 基因演算法步驟

---

**Genetic Algorithm**

---

```

1: function GA(  $I_{stop}$ ,  $N_{pop}$ ,  $P_c$ ,  $P_m$  )

2:   clock = 0, iteration_num = 0, start_t = now_time()

3:   Generate initial population (randomly generate  $N_{pop}$  pairs of chromosome)

4:   while iteration_num <  $I_{stop}$  or clock <  $T_{stop}$  do

5:     calculate each pair of chromosome fitness in population

6:     select  $N_{pop}$  pairs of chromosome and copy them to new population

7:     crossover by probability  $P_c$  and copy them to new population

8:     mutation by probability  $P_m$  for each chromosome

9:     clock = now_time() – start_t

10:    if this iteration not find better chromosome then

11:      iteration += 1

12:    else

13:      iteration = 0

11:  end while

12:  return best chromosome

12: end function

```

---

### 5.1.1 染色體的編碼方式

首先要先將貨車的行駛路徑編碼成一串數字，以下圖 5.1 為例，如果有一條為期十期的路線，會做出一條 12 個位置的陣列(10 期+2 期虛擬點)，前後因為是虛擬點，設為-1，中間因為有抵達 3, 5, 4, 2, 7, 7 這些集散點，所以將當放入對應時期的格子，

剩下的格子填入-1，以表示行駛中的狀態，來完成貨車路徑編碼。

對於在中間補-1 的原因，一方面是可以表示車子在行駛中，另一方面這樣可以維持每條路線的長度一致，如果表示成 3, 5, 4, 2, 7, 7 這樣的話，每台車會因為經過的集散點數量不同，導致有不同的長度。

Period	0	1	2	3	4	5	6	7	8	9	10	11
Route	d → 3 → 5 → 4 → 2 → 7 → 7 → d											
Encoding	-1	3	5	-1	4	-1	2	-1	-1	7	7	-1

圖 5.1: 染色體編碼方式

將貨車的行駛路線編碼後，就可以將這樣的方式做出一組解(染色體)，如下圖 5.2 舉例，如果目前有三台貨車在城市內跑，一條染色體會用這樣表示其路線

<b>Chromosome</b>												
0	1	2	3	4	5	6	7	8	9	10	11	
-1	3	5	-1	4	-1	2	-1	-1	7	7	-1	
-1	2	-1	4	4	-1	5	3	3	-1	2	-1	
-1	7	-1	5	-1	4	3	4	-1	2	2	-1	

圖 5.2: 染色體編碼示意圖

### 5.1.2 交配策略(crossover operation)

交配這個階段，會讓染色體間互相交換基因，在本研究中，代表著將會互相交換貨車的路徑，在此本研究嘗試過兩種交換的方式，第一種如下圖，將其中一條貨車的路徑切段與另一個染色體的貨車交換，然而這樣的效果不好，原因是貨車的路徑實際

上是前後互相有關連的，如果這輛車的路徑表現的好，意義上為他在集散點之間轉運了很多貨物，但是如果用這樣的方式去交換，有可能把貨物的轉運路線切斷，使得交換讓整體的表現變差。以下圖 5.3 為例，切斷點為第二期之後，如果前面有貨物在第二期之前上車，會在第二期知後的某一期被轉運到目的地的集散點，會因為這樣的交換，很有可能無法將貨物送達，也就是交換後，無法保有彼此好的表現。

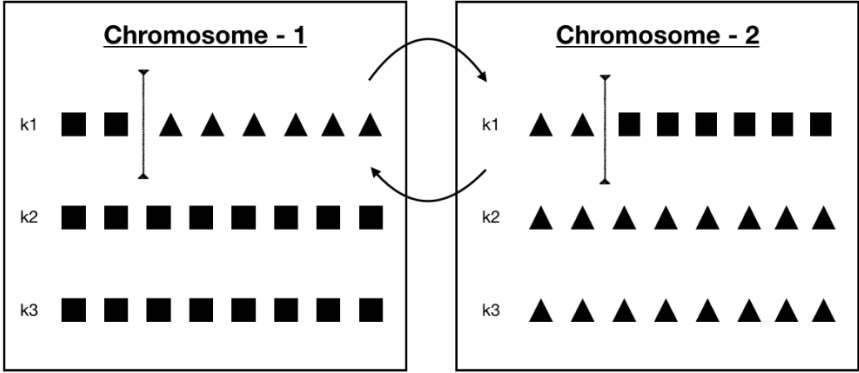


圖 5.3:第一種交配方式

於是為了保有貨物完整地的轉運路徑，將交配的策略改為下圖的方式，當某兩條染色體要交配時，會決定要從第幾台車之間切斷，然後互換車子的路徑，下圖 5.4 就是決定從第一台車切斷，彼此互換第一台車

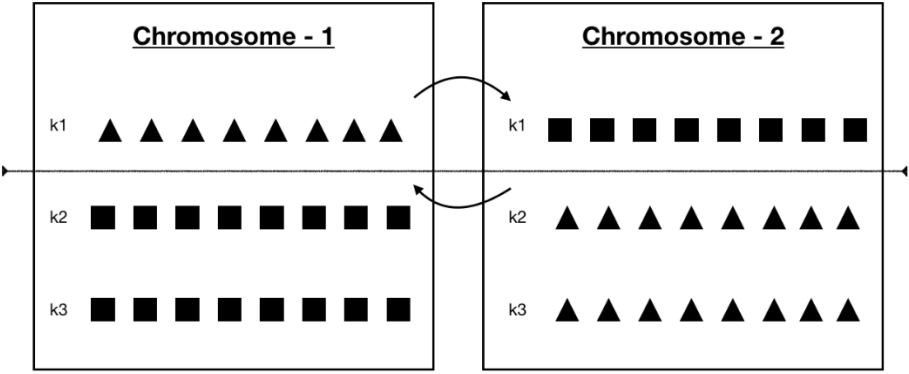


圖 5.4:第二種交配方式



### 5.1.3 突變策略(mutuation operation)

相同於交配策略，將不會單一的把某條路線的某段重排，而是將染色體其中一台車子的路線全部重排，來達到突變的效果，進而有機會跳出區域解，如下圖所示

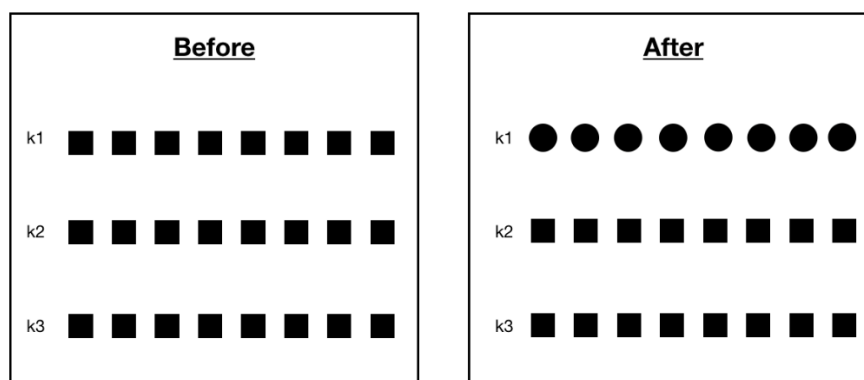


圖 5.5:染色體突變前後對照圖

### 5.1.4 適應函式(fitness function)

在基因演算法中，要透過適應函式來求得每條染色體的目標值，本研究的目標是最小化總體成本，然而要達到成本越低，需要盡可能地將貨物透過集散點去轉運，而避免因為無法轉運而雇用送貨人員去送。

目前每條染色體為所有貨車的路徑，需要設計一個方式可以快速知道，有多少貨物可以被這些路徑轉運，有多少貨物需要雇用送貨人員，如此一來就可以評估這些路徑的成效。

因此本研究染色體中的所有貨車路徑，由後往前作追蹤，進而得知每期每個集散點往後可以最早經過哪些集散點，如下圖 5.6 所示：

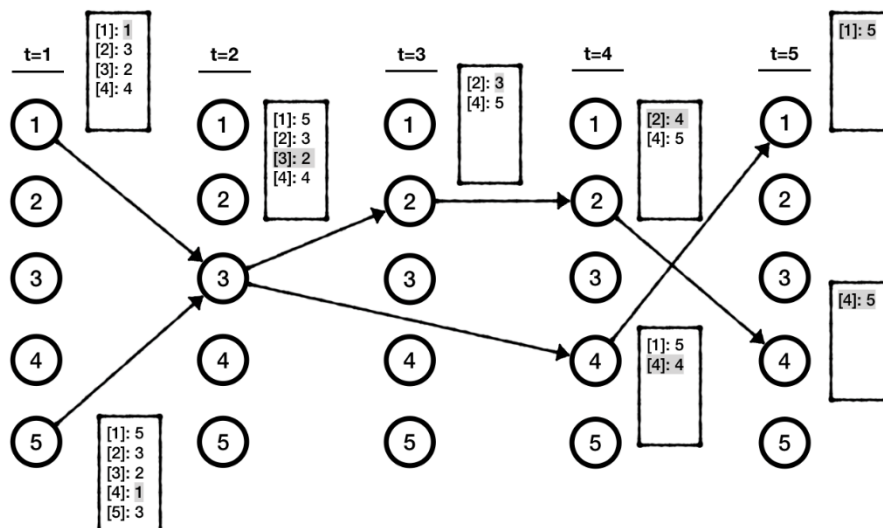


圖 5.6:路線追蹤說明

以圖為例，從第五期往前慢慢回推，在第五期的時候只有包含自己站點的資訊，在第四期時，集散點 2, 4 就有包含第五期的資訊和自己的資訊，依這類方式往前推進。如果有比較早的期數，變更新較早的，如果某期某集散點的下一步有負數個集散點的可能，便將那些資訊通通相加，如第 2 期集散點 3 把第三期的集散點 2 跟集散點 4 資訊相加。

當完成這路線追蹤後，每個貨物只需查詢其出發地點集散點是否資訊有包含其未來的目的集散點，如果有那就能成功轉運，其中還需要檢查是否有超過轉運的時間限制，舉例來說，如果有一個貨物在第 2 期時會到集散點 4，目的地為集散點 1，由於第 2 期並無經過集散點 4，於是往後期數檢查，直到發現第 4 期有經過集散點 4，並且在這點的資訊可以看出，在未來第五期未到達集散點 1，為此貨物的終點，於是可以將這貨物視為可以被轉運。

透過這樣的方式，即可將每條染色體算出其轉運的量，轉運量越多成本將會越低，這數量會當作一個指標來判斷這個染色體的好壞。

### 5.1.5 挑選策略(selection operator)

本研究選的挑選策略為輪盤法的方式，如果染色體經由適應函式(fitness function)算出的目標值越高(轉運的貨物量越高，成本越低)，被選的機率就會越高，以下介紹其方法：

假設目前有  $n$  組染色體組合，其適應值分別為  $E(C_1), E(C_2), \dots, E(C_n)$

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

在實際測試時，為了讓較好表現的染色體可以較容易被選到，有時會事先將染色體乘上一個權重，如  $E(C_i) = E(C_i) \times weight$  來讓被選種的機率增高。

## 5.2 計算初始解

基因演算法一開始產生的解(染色體)皆為亂數產生，在本小節要利用類似貪婪式的演算法，給予能夠送較多的貨的路線，或是能夠收較多貨的路線，較高的選擇機率，來取代亂數產生的機制，去生成更好的貨車路線圖。

### 5.2.1 產生初始解步驟

初始解的路線是一條一條找尋而成，如果一台貨車的路線找完，依序換第二台，直到所有車子的所有路線都找完。每次找下一條路線時，都會依照當下的狀態，給予所有可能的下一步一個權重，再把這個變成機率的方式來選擇下一步要走往哪。權重的考量有兩種：

1. 抵達下一個點的收貨的數量
2. 抵達下一個點可以送貨的數量(取決於目前車上有那些貨物)

因此每次決定完一個點後，都必須更新目前的送貨情況，接下來會介紹兩種權重。

### 5.2.2 抵達下一個點的收貨權重

此權重設計的方式很簡單，考慮往後走一步後，可以收到多少數量的貨，就給予那一條路線對應的權重，如下圖 5.7 所示，因此數量越多權重越高。此方法甚至可以針對往後多走幾步的考慮，而給予高低權重，不過在實作後發現，此權重不但沒有好的影響，甚至會打亂送貨的選擇，於是在設計上，只當車子目前是空的時候，才會有權重的效果，且只有走一步的考慮，走一步以上的權重，參考價值頗低。由於未來會以大需求作為問題假設，所以幾乎去任何一個集散點都會有為數不少的貨等待轉運。

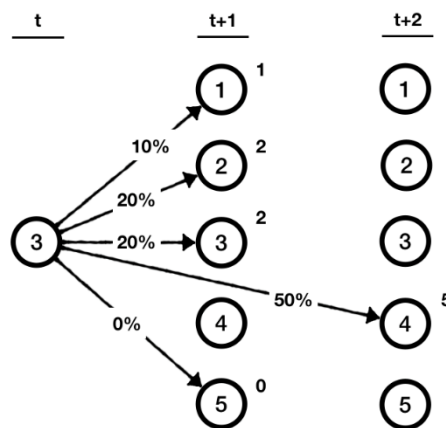


圖 5.7:計算收貨權重示意圖

### 5.2.2 抵達下一個點的送貨權重

送貨的目的集散點，是取決於目前貨車上有那些貨，如下圖 5.8 所示，假如目前有四個貨物，分別想去集散點 1, 1, 1, 2，則有很高的機率選種往集散點 1。

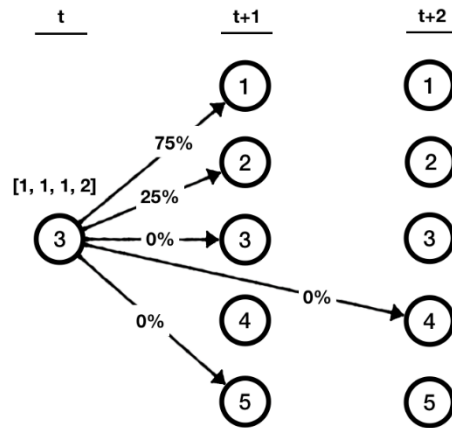


圖 5.8:計算送貨權重示意圖

如同收貨的情況，考慮一步的情況是最理想的，因貨物在問題設計上本來就不會被轉運太久；若考慮多步之後的權重，很容易會因為時間過長而送貨失敗。

### 5.5.3 加入二次轉運的考量

二次轉運，在這邊定義為，同一個貨物，被兩台以上的貨車運送，如果有加入轉運的考量，可以讓路線成效更好。因此在找路線時，如果貨物有成功載上車，但是卻沒有如期抵達終點集散點，便會記錄它走過的路徑，等待其它台車可以來帶它往終點。例如下圖 5.9 所示，貨物 A 的終點集散點為 4，但是在原本的路線並未送達目的地，且因為貨物的運送時間有限，所以在第四期就被踢出。

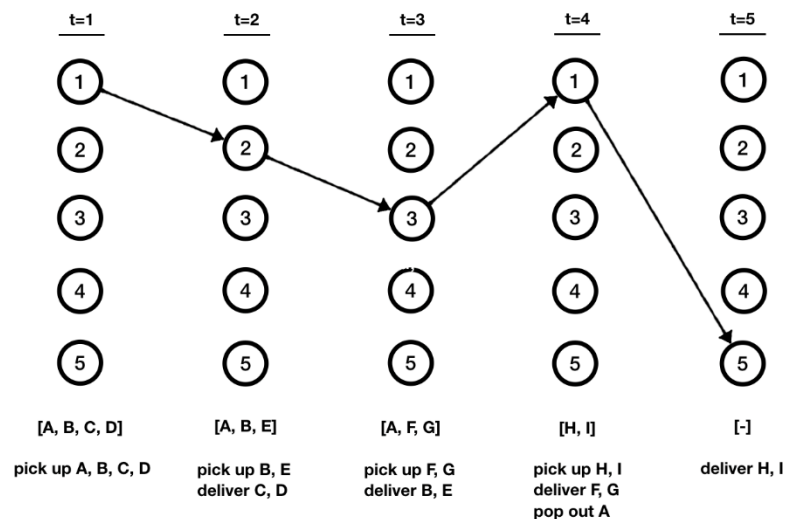


圖 5.9:第一台貨車尋找送貨路線示意圖

到了第二台車在第二期時，因為也走到集散點 2，所以可以戴上貨物 A，並轉運到集散點 4，如此一來，就能夠考慮車子與車子之間彼此的關聯性在路徑考量內，使解的品質提高。

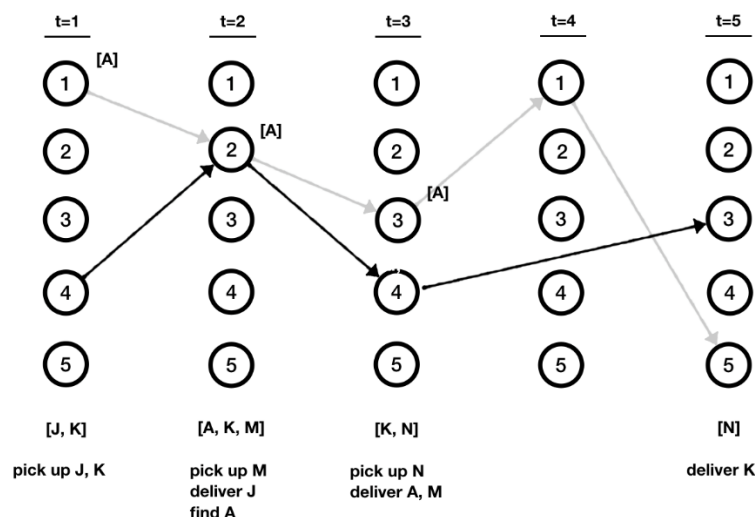


圖 5.10: 第二台車尋找送貨路線示意圖

### 5.3 小結

相較於以往找路徑的問題，本研究的問題限制較為特別一點，沒有辦法透過最短，或是最長來給得知此路徑好不好，要先利用路徑做出類似樹狀圖的路線，如果越多貨物可以在這樹狀圖中找到起點跟終點，才能代表此路徑越好，這種特殊的限制讓這個問題困難許多。如果貨物的量很大的話，光是在驗證這個路徑的好壞就要花費不少時間，在找尋初始解時，也會因為找尋下一條路時，計算當前的狀態給權重，而導致無法在極短時間內給出適合的路。

在設計找出初始解的演算法時，原本是想考慮跟不只是當前的情況，而是往後幾期的收送貨物狀態評估後，再給出下一步的，希望是可以跟數學模式一樣考慮全面的想法，但是這樣會導致解的品質很差，幾乎跟未加入初始解的情況相似，或許這點還須找出原因，或是調整出較為適合的權重參數去加乘。

## 第六章 數值分析

本章節針對本研究所建立的數學模式與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式與數學演算法，並利用 Gurobi 8.1.1 版求解數學模式。

### 6.1 區域性無人車數學模式測試

#### 6.1.1 視覺化結果與分析

利用 python 的 bokeh 視覺化套件，我們可以清楚的將結果顯現出來。首先先產生地圖的資訊，包含有 100 個客戶點以及 9 個集散點的選擇，如下圖 6.1 所示，圈圈的範圍為集散點的服務區域

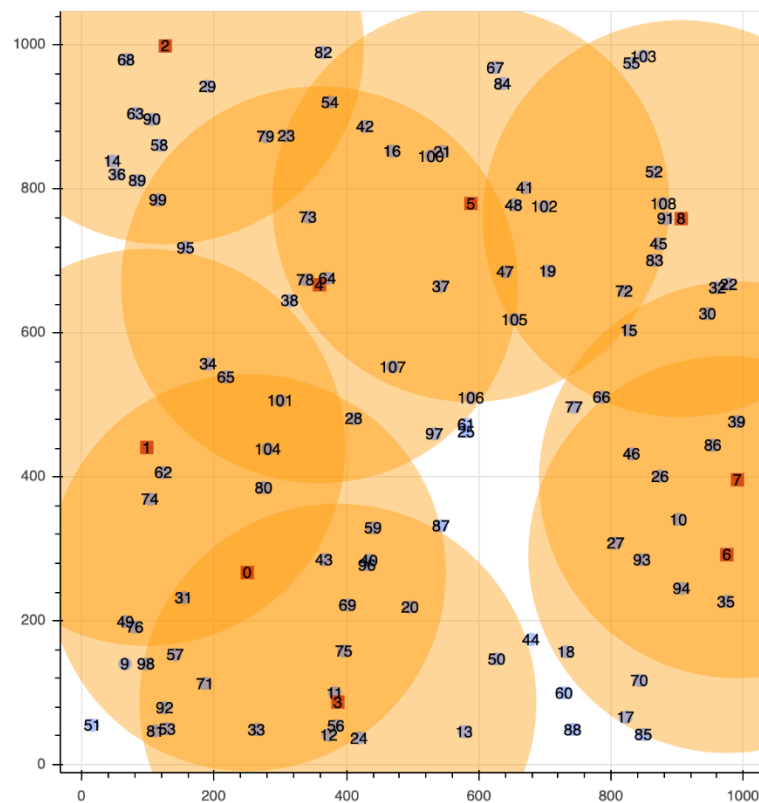


圖 6.1:顧客點以及集散點視覺化

接下來隨機產生送貨的資料後，開始跑模式，目前設定的參數如下表 6.1:

表 6.1:小測資參數設定表

候選點 數量	貨車 數量	期數	貨物 數量	貨物服務 時間上限	集散點內服務時間限制
9	5	10	33	5	1

跑完模式後，可以從視覺化的部分，看出貨物的運送情況，黑色為用集散點轉運的貨物，綠色為在集散點內被小型無人車直接運送的貨物，紅色為用送貨人員送的貨物，如下圖 6.2 所示，箭頭的起迄為貨物的起始客戶點跟終點客戶點。

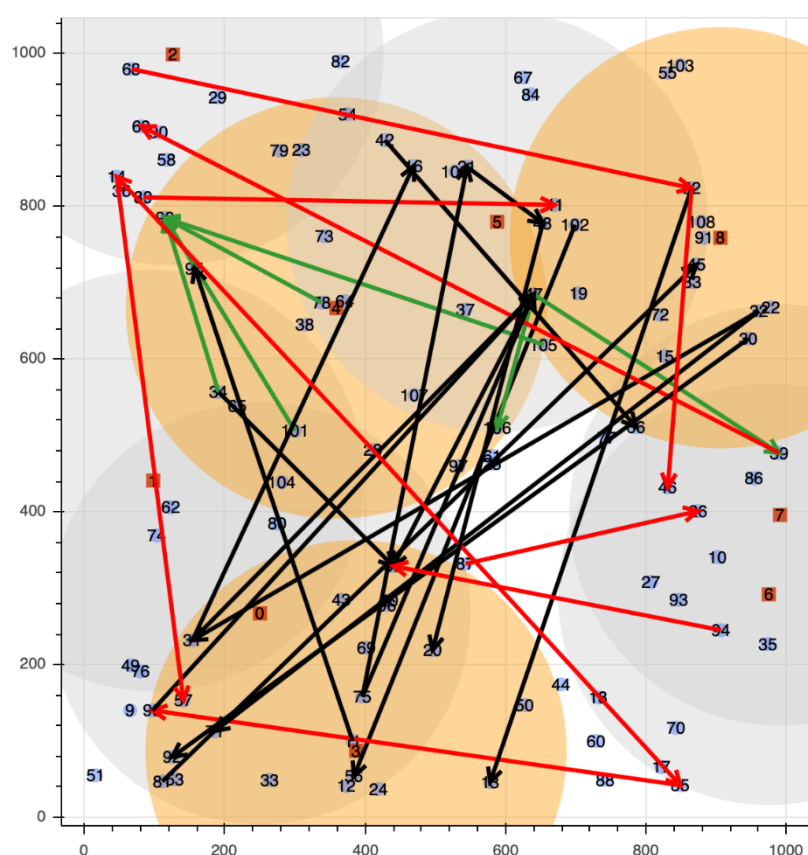


圖 6.2:貨物運送方式視覺化

從結果來看，因為選擇了 3,4,8 三個集散點，造成沒有被集散點的服務到的貨物幾乎都用快遞人員運送，如果貨物的起訖點是被同個集散點服務的話，也有被小型無人車運送成功。

除了看貨物的運送分配外，也可以透過圖來看出貨車的路線狀況，由於從圖 6.3



可以看出集散點 3,8 之間有較大量的貨物流動，所以 3,8 之間可以看出流動率較大。

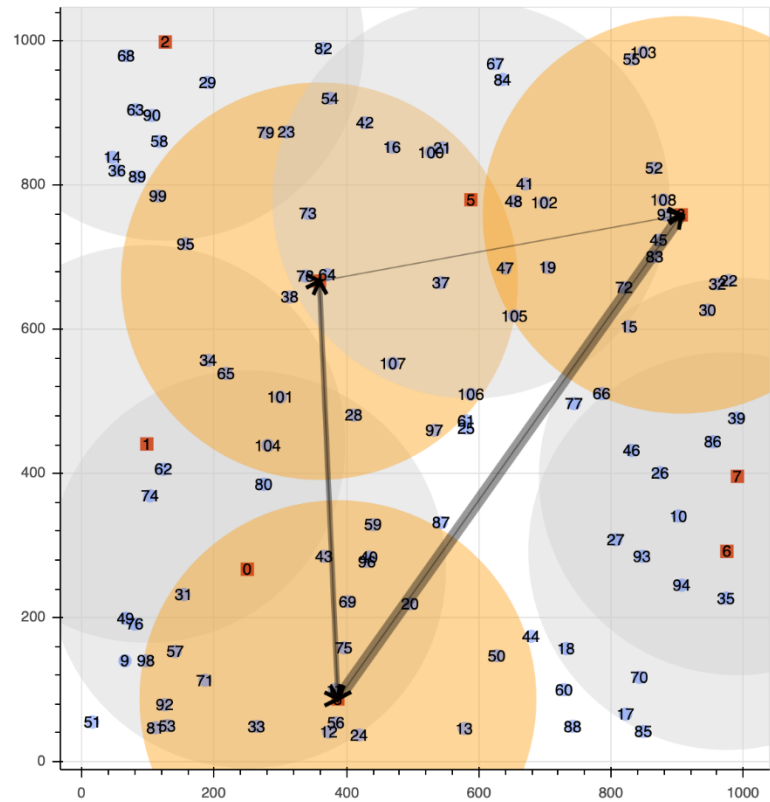


圖 6.3:貨車路徑視覺化

另外，也視覺化時空網路來查看，貨物是否有正確的被貨車在集散點之間運送，從下圖 6.4 可以看到，虛線的為貨車，實線的為貨物，綠色的線為目前被選出來查看的貨物，可以看出在集散點 8 等待了一期，然後被貨車從集散點 8 送往集散點 3，再回到顧客點，完成這趟旅程。

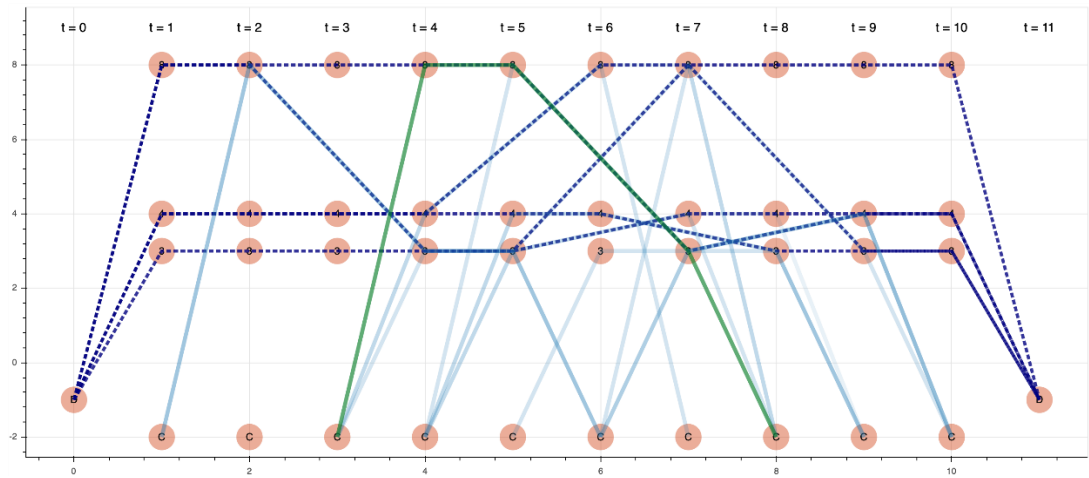


圖 6.4:時空網路圖結果視覺化

另外，我們也將我們的貨物資料特別設計，來檢查貨車的運送是否真的有達成目的，以圖 6.5 為例，左圖為設計其中某集散點到某集散點的貨物量特別多，也因此可以很明顯的看出有一條貨物路線特別明顯，右圖的則是某一個集散點為大部分貨物的終點，因此也可以發現呈現一種發散型的路線

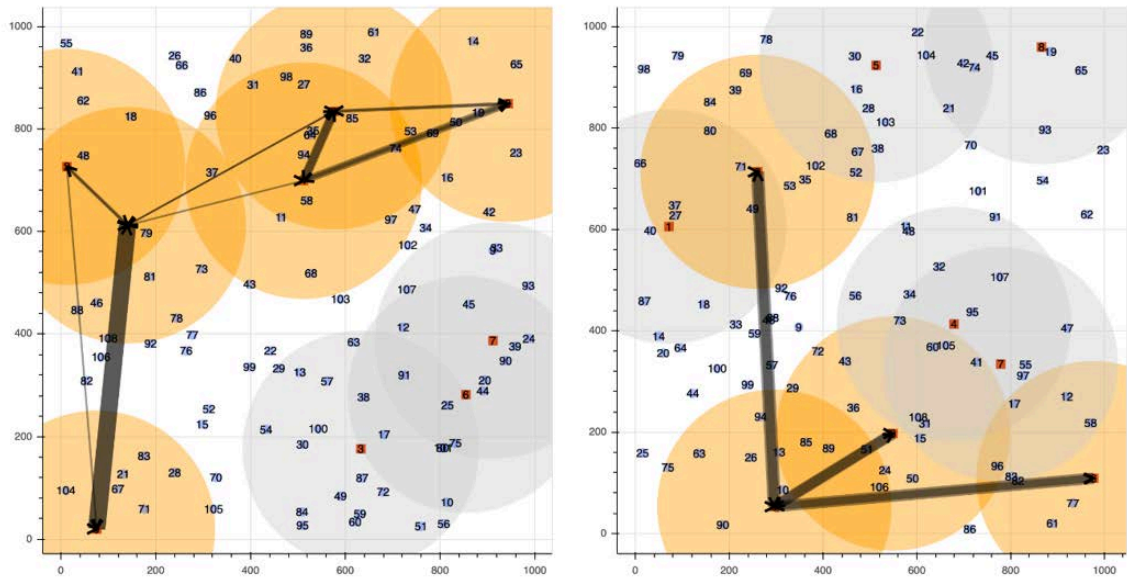


圖 6.5:測試貨車路線圖

### 6.1.2 分析模式求解情況

因為本模式的問題需求，是要模擬都市內遇到大量需求的情境，因此我們設計了大量的資料來看模式求解情況，以每期每個集散點會有的需求數量為一個變數，給出這個量的平均數跟變異數來算，舉例來說如果平均數為 10，那代表每期每個集散點的需求數量平均為 10 個，一直到末三期前結束。下表 6.2, 表 6.3 為測試模式的參數以及結果。

表 6.2:大測資參數設定表

候選點 數量	貨車 數量	期數	貨物服務 時間上限	集散點內服 務時間限制	貨車 空間上限	集散點 空間上限
9	5	24	5	1	1000	1000

表 6.3:數學模式測試結果

資料集	貨物每期 平均量	Preprocessing Time(s)	CPU Time(s)	Gap(%)	與 P2P 模式相比 減少的比率
10_1	10	43	7200	89%	39.99%
10_2		41	7200	90%	36.84%
10_3		44	7200	90%	39.02%
10_4		42	7200	92%	39.36%
10_5		42	7200	89%	40.48%
20_1	20	79	7200	93%	42.27%
20_2		82	7200	91%	44.41%
20_3		81	7200	91%	42.27%
20_4		79	7200	92%	43.02%
20_5		84	7200	92%	50.11%
30_1	30	141	10800	94%	40.66%
30_2		139	10800	95%	45.71%
30_3		143	10800	95%	41.91%
30_4		144	10800	95%	40.97%
30_5		140	10800	93%	41.05%

可以從表看出，求解的品質在大資料下是相當不理想的。由於模式是時空網路的方式所建置，每一期都有大量的變數，導致期數越多，負擔越大。在表中的停止時間在貨物平均數為 10, 20 的時候是設定 2 小時，30 設定為 3 小時，是因為 30 的貨物數量太大，導致兩小時也給不出一個解。

另外，最後一欄的比較，是以軸輻式網路概念算出的總運輸成本跟 P2P 模式算出的成本，如果直送情境為利用群眾外包的方式，來達到任意點任意時間的直送貨物，

就能模擬點對點大量直送的情境。因此從這兩個欄位比較可以得知，需求越大量，軸輻式的網路設計將會帶來更多經濟效益。

### 6.1.3 滾動式模式求解

為了使模式在短時間內求解出一個好品質，本研究嘗試利用滾動式的方式來進行求解，滾動式的方式為每次框出一個期間內去求解，然後決定前好其中幾期後，繼續滑動，舉例來說可以用五期求解，每次都會決定求解的第一期，如果有總共 24 期要做的話，就會用這樣的方式進行 24 次，第一次的範圍為第一期~第五期，決定第一期，第二次為第二期到第六期，決定第二期，以此類推，表 6.4 會嘗試固定五期求解，分別決定一期、兩期、三期來看期求解的效率，由於觀察出決定兩期的成本最低，以下的成本差異蘭位皆為和決定兩期的成本比較，來看出其差異。

表 6.4:滾動式決定期數成果比較

資料集	決定一期		決定兩期		決定三期	
	成本差異	時間	成本差異	時間	成本差異	時間
10_1	-0.65%	1509(s)	0.00%	1817(s)	22.26%	1149(s)
10_2	1.62%	1209(s)	0.00%	1908(s)	24.29%	615(s)
10_3	3.27%	1333(s)	0.00%	2183(s)	16.85%	998(s)
10_4	4.00%	1613(s)	0.00%	469(s)	24.19%	690(s)
10_5	6.52%	1372(s)	0.00%	1720(s)	21.95%	1071(s)
20_1	1.43%	2857(s)	0.00%	1286(s)	18.89%	790(s)
20_2	8.92%	3139(s)	0.00%	1396(s)	24.53%	876(s)
20_3	6.56%	3036(s)	0.00%	1398(s)	15.68%	797(s)
20_4	10.26%	2974(s)	0.00%	1290(s)	16.06%	765(s)
20_5	5.29%	3302(s)	0.00%	1469(s)	27.63%	5450(s)
30_1	12.08%	7523(s)	0.00%	3082(s)	21.93%	1866(s)
30_2	27.32%	8429(s)	0.00%	3405(s)	29.31%	2032(s)
30_3	3.00%	7667(s)	0.00%	3531(s)	13.72%	1911(s)
30_4	2.54%	8288(s)	0.00%	3630(s)	11.84%	2076(s)
30_5	7.24%	7458(s)	0.00%	3345(s)	19.87%	1973(s)

接下來用決定兩期的數值，跟原本數學模式與基因加入初始解的結果比較，如表 6.5 所示，成本差異皆與滾動式比較，其中可以看出，用滾動式的結果品質大部分都是裡面最好的，不過因為目前是用比較花時間的方式再做滾動式，所以相較於演算法，

還是略花了不少時間。

表 6.5:滾動式與其它方法成果比較

資料集	Model		GA-init		滾動式(兩期)	
	成本差異	時間	成本差異	時間	成本差異	時間
10_1	39.44%	7200(s)	17.01%	506(s)	0.00%	1817(s)
10_2	25.52%	7200(s)	15.23%	176(s)	0.00%	1908(s)
10_3	32.04%	7200(s)	21.20%	117(s)	0.00%	2183(s)
10_4	45.10%	7200(s)	16.14%	186(s)	0.00%	469(s)
10_5	18.37%	7200(s)	14.40%	126(s)	0.00%	1720(s)
20_1	17.41%	7200(s)	10.70%	208(s)	0.00%	1286(s)
20_2	12.17%	7200(s)	21.00%	245(s)	0.00%	1396(s)
20_3	-0.02%	7200(s)	9.92%	278(s)	0.00%	1398(s)
20_4	8.93%	7200(s)	12.75%	211(s)	0.00%	1290(s)
20_5	0.56%	7200(s)	17.16%	752(s)	0.00%	1469(s)
30_1	29.63%	10800(s)	17.93%	388(s)	0.00%	3082(s)
30_2	43.21%	10800(s)	16.94%	848(s)	0.00%	3405(s)
30_3	25.95%	10800(s)	11.14%	1184(s)	0.00%	3531(s)
30_4	12.62%	10800(s)	8.69%	692(s)	0.00%	3630(s)
30_5	21.46%	10800(s)	19.33%	598(s)	0.00%	3345(s)

## 6.2 共享性無人車數學模式測試

這邊會先將區域性的模式中小型無人車使用結果，跟共享性模式算出來的使用結果比較，來得知是否透過共享性的模式，可以有效減少小型無人車的使用，以及提升無人車的使用率。

### 6.2.1 視覺化結果與分析

下圖 6.6 的資料為 10\_1 的資料集，經過區域性無人車收送模式算出後，得知集散點 5 一整天的需求以及該放多少無人車在集散點服務，其數量分別代表 demand 跟 regional AV 的折線，另外再用共享性無人車模式算出，可共享的無人車方式，每期在集散點 5 停留的量，其數量為 shared AV 的折線。從結果來看，共享性的模式的確給出了較好的調度方式，來滿足變動的需求，除了在第 1,2 期高於需求很多之外，剩下的小型無人車數量幾乎貼近著需求線移動，如此一來就能夠更省小型無人車的成本。

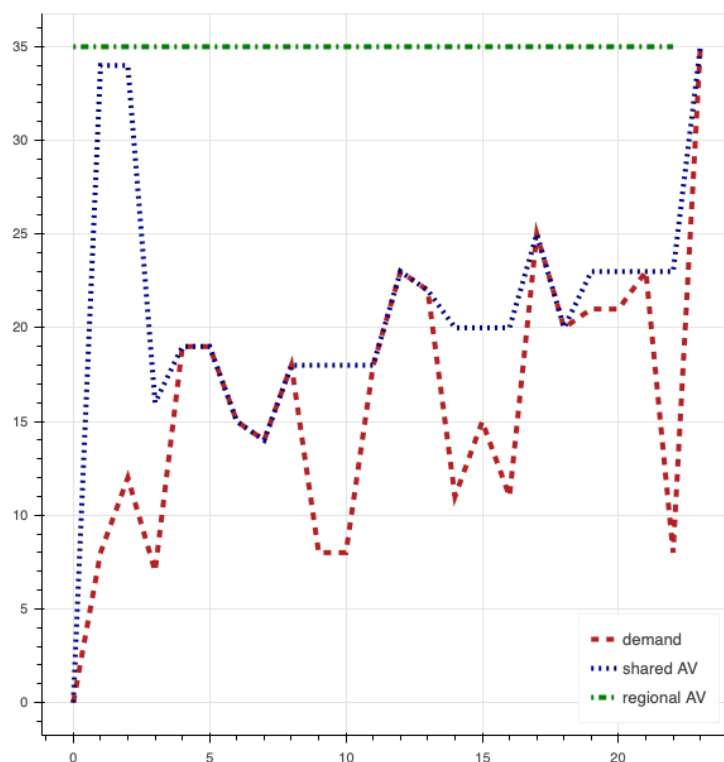


圖 6.6:共享性模式視覺化折線圖

表 6.6:共享性模式結果分析表

資料集	無人車總量			無人車整體使用率			共享性模式 求解時間
	區域性	共享性	減少量	區域性	共享性	提升使用率	
10_1	160	137	23	29%	34%	5%	0.07(s)
10_2	132	120	12	27%	30%	3%	0.06(s)
10_3	143	126	17	29%	33%	4%	0.04(s)
10_4	145	130	15	26%	29%	3%	0.05(s)
10_5	151	130	21	28%	32%	3%	0.08(s)
20_1	271	258	13	29%	31%	2%	0.07(s)
20_2	214	190	24	46%	52%	6%	0.06(s)
20_3	207	201	6	43%	44%	1%	0.05(s)
20_4	236	230	6	37%	38%	1%	0.07(s)
20_5	191	179	12	53%	56%	3%	0.05(s)
30_1	200	173	27	64%	74%	10%	0.08(s)
30_2	298	260	38	37%	43%	6%	0.12(s)
30_3	198	170	28	54%	63%	9%	0.08(s)
30_4	197	181	16	51%	55%	4%	0.11(s)
30_5	213	200	13	64%	68%	4%	0.09(s)

另外，我們也測試過所有的資料集，來檢查在大量需求下的資料，是否可以有效的減少無人車成本，並且提升使用率，結果如表 6.6。可以從表中得知，雖然模式可

以在很短的時間內給出答案，但是改善的幅度有限，小型無人車沒有少很多而且提升的使用率也不高，或許要把共享性的考量加入在第三章的模式，才有機會可以改善更多，不過這樣會使問題變得更複雜難解。

## 6.3 區域性無人車演算法測試

這邊將會比較未加入初始解的基因演算法，與加入後的比較，所使用的參數如下表 6.5 所示，其中 Stop iteration 為過了多少次的迭代如果還沒有找到更好的解，就結束演算法。

表 6.7: 基因演算法參數

Crossover rate	Mutation rate	Stop iteration	Stop time	Population number
80%	30%	100	7200	10

### 6.3.1 視覺化結果與分析

從區域性無人車數學模式測試的資料集 10\_1, 20\_1, 30\_1 來比較，在圖 6.7, 圖 6.8, 圖 6.9 以折線圖的方式呈現，x 軸為時間，y 軸為總成本，GA 為基因演算法，GA-init 為加入初始解後的基因演算法，Model 為數學模式解。

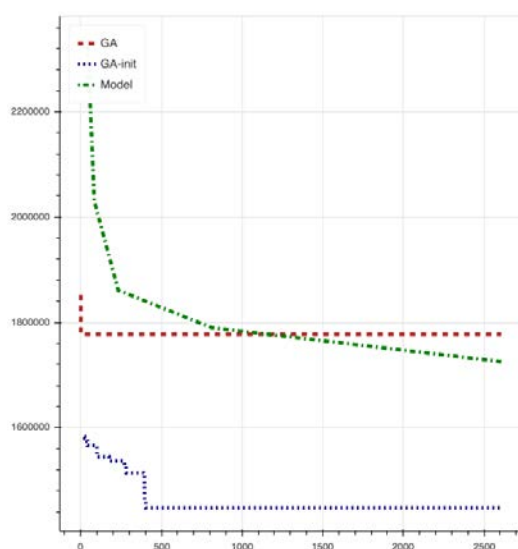


圖 6.7: 資料集 10\_1 的結果折線圖

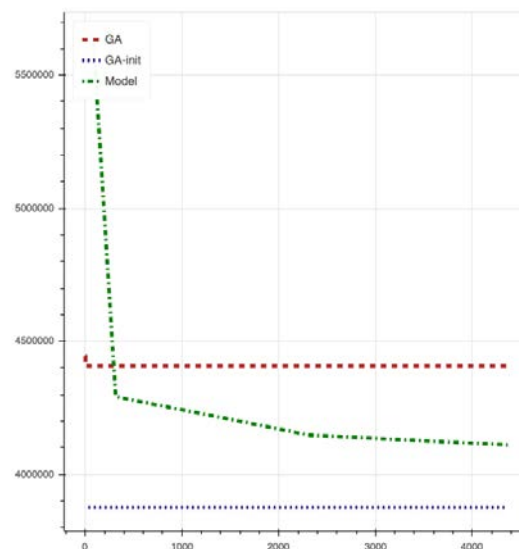


圖 6.8: 資料集 20\_1 的結果折線圖

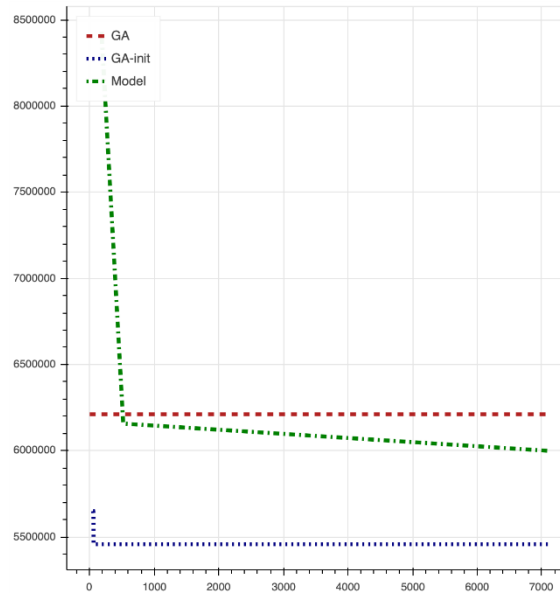


圖 6.9: 資料集 30\_1 的結果折線圖

可以從上面觀察得知，有加入初始解的 GA-init 都表現優於其它兩者，不過與未加入初始解的演算法一樣，幾乎很快就收斂，無法再找到更好的解。

## 6.3 小結

因為目前問題情境假設是大量的貨物運送需求，所以當資料量變大時，數學模式無法在短時間內給出好品質的解，這時在第五章所做的演算法就能過在短時間內給出優於模式的結果，但似乎很快就掉入區域解，無法再精進，或許未來可以再針對演算法去調整，進而越靠近全域最佳解。



## 第七章 結論與未來研究

### 7.1 結論

本研究提出了軸輻式網路(Hub-and-Spoke) 的概念，透過大批貨物集中轉運的方式，來大大降低運送成本，並且搭配著小型無人車(Autonomous Vehicle) 的概念，有效率地去收貨以及送貨，完成最初貨物的最初及最後一哩路，其目的為當城市內有大量起訖的收送貨需求時，相較於目前的點對點的直送機制，可以有更低成本，更有效率地去解決一日配送問題，以下幾點為本研究目前的貢獻：

1. **討論「依時間變化需求」的情況：**過去文獻中探討軸輻式網路的收送貨問題，由於未貼近現實中『依時間變化的需求』的考量，因此無法針對一日配送問題中高峰期的需求有更彈性的路線規劃。在本研究將這個因子考慮進去，藉由搜集過去數據，得知每一天的需求，進而安排出滿足高低峰期的需求車輛路徑，來更有效率，更省成本的解決都市內的大量收送貨需求。另外，在過往探討此問題時，通常是要決定一個「長期決策」，本研究的一日配送需求，乍看之下是為了滿足短期下的需求所做的決策，其實是假設能透過大數據來蒐集長時間下每一天的需求，進而預測未來的一天內需求量，透過這個預測，再規劃有如公車固定排班式的方式，讓貨車可以按班表去開往各個集散點轉運，來滿足考慮時間變化下的需求考量。
2. **由無人車收送最初及最後一哩路：**在過往文獻中，針對無人車的收送貨有提出過子母車的概念，貨車會由集中式的倉儲派出，然後經由演算法去找出放無人車下車的點以及回收無人車的點，然而這樣的概念在未來出現大量需求時，容易導致往返倉儲的貨車數過多，並不實用，因此本研究改搭配軸輻式網路的概念，來完成收送貨的最初及最後一哩路。
3. **利用時空網路圖來求解『區域性無人車收送網路設計問題』：**在第三章可以得知，有兩種路徑是要被數學模式求出，分別為貨車的路線，以及貨物的路

線，利用時空網路圖可以使我們方便的把這兩條路線的限制式寫出，並且限制貨物在集散點間的移動一定要跟著貨車的路線。然而時空網路的缺點如下：每一期都有所有集散點到下一個集散點的路線變數，將近每期都有一個完全圖的路線需要考慮，這使得總期數變大時該數學模式會越難在短時間內求得最佳解。譬如在第六章的數值分析中，利用 Gurobi 去求解數學模式時，如果當測試較大的測資時，通常無法在短時間內給出好品質的解。

4. **探討『共享性無人車收送網路設計問題』**：在第四章中，延續了第三章的情境，進一步探討將小型無人車共享至其它的集散點，目的要增加各個無人車的使用率之外，也可以用更少台的無人車完成所有收送貨的需求，然而在資料測試後發現，其成效改善有限，或許原因是因為當下的貨車路線是已知的，來自第三章的數學模式，導致無法彈性地共享各個集散點的無人車。我們亦曾想要同時考慮貨車路線，但是在第三章都已經很難求求解，這樣考慮下會使問題難度提升更高。
5. **設計啟發式演算法與基因演算法**：為了要快速求得品質較佳的解，在第五章中我們利用基因演算法的概念去求解，並且透過貪婪演算法，得到較好的初始解組合，來改善基因演算法的求解品質。貪婪演算法能讓貨車選擇路線時，逐步地考慮下一步的收貨、送貨數量，給予權重後再用機率選出，進而將所有貨車路線構建出來。再經過測資測試後可以發現，加入初始解將有助於更快搜索到好品質的解。
6. **滾動式求解『區域性無人車收送網路設計問題』**：由於原數學模式求解不易，在第六章的數值分析，有利用滾動式的方式，來逐步求解，從結果可以看出，此方式可以在短時間內算出一定品質的解，甚至遠好於演算法算出的解。

## 7.2 未來研究

由於目前情境的假設，有許多部份並未更有彈性地考量現實情況，可以在未來的時候延伸討論的有以下幾點：

1. **測試資料與現實有誤差：**測試資料目前並不是用一個地圖作為基底所設計的，因此在路線距離、路線旅程時間的考慮，皆會和現實有所出入。
2. **無法轉運的成本估算：**目前所有因附近無設置集散點而無法藉由軸輻式網路轉運的收送件需求，皆會被轉成由快遞人員直送來計算成本，然而這些收送需求其實亦可藉由快遞人員收送至較近的軸輻式網路集散點後再轉運，這樣不但較為彈性，成本也會減少，較為貼近現實中的考量。
3. **P2P 模式的現實面：**目前的每份收送需求皆假設送貨人員同一時間僅會處理一件；然而在現實中，其實送貨人員也有可能在送(收)貨途中收(送)貨，因此需要建構出針對會同時收送貨路線的車輛路徑問題( Vehicle Routing Problem )數學模式，才可以更符合現實地與軸輻式網路比較。
4. **並未考慮超過無人車上限的需求：**目前假設當有需求時，只要有一台無人車，即可收貨，如果考量收貨的量大於無人車時，派送兩台以上的車子，將會更貼近現實。
5. **限縮無人車的服務範圍：**當無人車來到某個集散點時，就只能針對此集散點所服務的顧客點做收送的工作，然而當有顧客點被重複被指派到多個集散點時，無人車是可以考慮轉而運送到另一個集散點，來讓貨物成功裝載上抵達的貨車來轉運，這樣的彈性收貨方式在本研究中並未考慮。
6. **加入公平性的考量：**目前的目標式是以最小化總成本來看，然而如果未來若有集散點需求量因為遠遠低於其它集散點，很有可能會被犧牲，貨車及有可能會去服務其它比較多需求的集散點。因此可以進一步的考量每一個集散點的一整天需求滿足率，讓數學模式可以盡量最大化最小的需求滿足率的集散點，來達到每個集散點公平的服務水準。

## 參考文獻

- Boysen, N., Schwerdfeger, S., & Weidinger, F. (2018). Scheduling last-mile deliveries with truck-based autonomous robots. *European Journal of Operational Research*.
- Bryan, D. L. and O'Kelly, M. E. Hub-and-spoke networks in air transportation: An analytical review. *Journal of Regional Science*, 39(2), 275-295, 1999.
- Campbell, J. F. (1994). Integer programming formulations of discrete hub location problems. *European Journal of Operational Research*, 72(2), 387-405.
- de Camargo, R. S., de Miranda, G., & Løkketangen, A. (2013). A new formulation and an exact approach for the many-to-many hub location-routing problem. *Applied Mathematical Modelling*, 37(12-13), 7465-7480.
- Çetiner, S., Sepil, C., & Süral, H. (2010). Hubbing and routing in postal delivery systems. *Annals of Operations research*, 181(1), 109-124.
- Daimler (2017). Vans & robots paketbote 2.0.  
<https://www.daimler.com/innovation/specials/future-transportation-vans/parcel-carrier-2-0.html>
- Karaoglan, I., Altıparmak, F., Kara, I., & Dengiz, B. (2012). The location-routing problem with simultaneous pickup and delivery: Formulations and a heuristic approach. *Omega*, 40(4), 465-477.
- Karimi, H. (2018). The capacitated hub covering location-routing problem for simultaneous pickup and delivery systems. *Computers & Industrial Engineering*, 116, 47-58.
- Karimi, H., & Bashiri, M. (2011). Hub covering location problems with different coverage types. *Scientia Iranica*, 18(6), 1571-1578.
- Nagy, G., & Salhi, S. (1998). The many-to-many location-routing problem. *Top*, 6(2), 261-275.

- O'Kelly, M. E. (1986). The location of interacting hub facilities. *Transportation science*, 20(2), 92-106.
- Rodríguez-Martín, I., Salazar-González, J. J., & Yaman, H. (2014). A branch-and-cut algorithm for the hub location and routing problem. *Computers & Operations Research*, 50, 161-174.
- Rodriguez, V., Alvarez, M. J., & Barcos, L. (2007). Hub location under capacity constraints. *Transportation Research Part E: Logistics and Transportation Review*, 43(5), 495-505.
- Skorin-Kapov, D., Skorin-Kapov, J., & O'Kelly, M. (1996). Tight linear programming relaxations of uncapacitated p-hub median problems. *European Journal of Operational Research*, 94(3), 582-593.
- Wasner, M., & Zäpfel, G. (2004). An integrated multi-depot hub-location vehicle routing model for network planning of parcel service. *International Journal of Production Economics*, 90(3), 403-419.