國　立　交　通　大　學
運　輸　科　技　與　管　理　學　系

碩 士 論 文

運用最短路徑演算法與動態資訊進行大眾運輸行前旅次規劃

The Transit Pre-Trip Planning Using Shortest Path Algorithm and Real-Time Information

研 究 生：任芬傑

指導教授：王晉元 副教授

中 華 民 國 九 十 八 年 七 月

運用最短路徑演算法與動態資訊進行大眾運輸行前旅次規劃
The Transit Pre-Trip Planning Using Shortest Path Algorithm and Real-Time Information

研 究 生：任芬傑　　Student : Fen-Chieh Jen
指導教授：王晉元　　Advisor : Dr. Jin-Yuan Wang

國立交通大學
運輸科技與管理學系
碩士論文

A Thesis

Submitted to Institute of Transportation Technology and Management

College of Management

National Chiao Tung University

In Partial Fulfillment of the Requirements

for the Degree of Master

In

Transportation Technology and Management

July 2006

Hsinchu, Taiwan, Republic of China

中華民國九十八年七月

運用最短路徑演算法與動態資訊進行大眾運輸行前旅次規劃

學生：任芬傑　　指導教授：王晉元 副教授

國立交通大學運輸科技與管理學系碩士論文

摘　要

在國內積極推動大眾運輸系統之環境下，各大眾運輸主管機關及營運機構均積極發展大眾運輸旅次規劃系統(Transit Trip Planning Systems) 。然目前國內外文獻所提及之大眾運輸旅次規劃演算法，大部分皆為特定邏輯設計，如最少轉乘以及最快到達。如此一來，所規劃出的方案將無法考慮到其餘邏輯。而本研究回顧目前線上系統，發現大眾運輸行前旅次規劃追求的，無論是轉乘次數、旅行時間、票價或者步行距離都是最少成本。故本研究認為，可將最短路徑演算法的概念套入行前旅次規劃演算法，進行追求最少成本，但相較於公路路網，大眾運輸路網具有一些特殊性，導致最短路徑演算法無法直接套入。故本研究首先將大眾運輸網路重新定義，並命名為加強式的網路。且回顧目前線上系統，發現由於未將動態資訊因素納入規劃分析中，當以時刻表規劃的方案中發生誤點現象時，使用者可能因為班次誤點，必須在轉乘站等待過久的時間。若能考慮動態資訊，供使用者在出發前，以目前時間點所得知的到站資訊進行規劃。而本研究利用即時讀取到站資訊的步驟考量動態資訊。最後以修改過的最短路徑演算法進行規劃。而本研究認為，規劃某特定邏輯時，應該以該邏輯作為目標式，其餘邏輯可由使用者自行設定作為限制式，例如：當進行最少轉乘邏輯規劃時，目標式即為轉乘次數最少，而限制式則可以是在多少旅行時間內、或是在多少票價內等限制。而本研究將利用目前較為多人討論的，最少旅行時間和最少轉乘次數作為主要邏輯。而本研究利用選擇變數較少的轉乘次數，作為使用者自行設定的限制變數。故本篇論文，將求解在使用者所限定的轉乘次數上限內，最少旅行時間之方案。經本研究以虛擬網路，以及新竹市區大眾運輸網路為例，發現本研究所構建的路網以及演算法，將可在短時間內規劃出合理的方案，並正確規劃出，考慮動態資訊之行前旅次規劃方案。

關鍵詞：大眾運輸旅次規劃、最短路徑演算法、動態即時資訊

The Transit Pre-Trip Planning Using Shortest Path Algorithm and Real-Time Information

Student : Fen-Chieh Jen Advisor : Dr. Jin-Yuan Wang

Department of Transportation Technology and Management
National Chiao Tung University

Abstract

Many transport authorities and operative organizations develop positively transit trip planning systems for elevating level of services, but those algorithms just for one specifically cost, that cloud be number of transfer, travel time, free or walking distance, and not take into real-time information. Given the origin/destination pair, all transit pre-trip systems will find the lowest cost way, and any costs are independent and important. However, if we just take only one cost into plan, the solution may be irrationality, so this paper modified shortest path algorithm to solve this problem. Choosing a specifically cost be an objection and the others could be subjection. In this paper, firstly, we redefined the transit network due to some study indicated shortest path algorithm can't use on it. Secondly, when passengers use the transit trip planning systems, it will load real-time information. Finally, the transit trip planning systems use the modified shortest path algorithm for planning. The purpose of this research is develops an algorithm, that can plan a trip with minimum main cost and under the other minor cost subject. This paper use travel time as main cost and the minor cost is number of transfer. Therefore, we test the accuracy by using a network which considering the real-time information of buses. The results have shown that the algorithm can generate the reasonable alternatives immediately.

keywords: transit trip planning, shortest path algorithm, real-time information

誌 謝

兩年前，做夢也沒想過自己會考上交通大學。兩年後的今天，也沒想過可以在兩年內畢業。讀商科的我，從沒想過自己居然會寫程式，這兩年從 Hello Word 到九九乘法表，最後我完成了我的論文，這一切都像是一場夢。在學的每一天都問自己，為甚麼要念研究所？讀的這麼辛苦，這到底為了什麼，休學的念頭一直都在我心中。在此時回憶發酵成酸甜苦辣，充滿著我的生活，久久不能釋懷。

本篇論文的完成，首先要感謝我的指導教授-王晉元老師，總是提醒著我，包容著我，謝謝您原諒學生曾經對您的忤逆。而蘇昭銘老師以及卓裕仁老師在口試的時的寶貴建議，更讓本篇論文增色不少。迷惘的時候，彥佑、建旻總是在我身旁，給我安慰以及鼓舞。壘球隊的戰友們，大均、金樺、老人、仲豪、小小，也給予我非常多的支持，我很愛你們。咬豬、可真、gay 彭，我對你們的愛，以無法用文字說明，認識八年了，很謝謝這兩年在新竹的照顧。尤其是咬豬，你總是接受我一些無理的要求，卻沒有怨言的陪著我。至於要繼續深造的士勛，學長必須在此勉勵您要堅持，研究的路上，是孤獨且寂寞的，可是，似乎用不在您身上。

一個成功的男人，背後一定有一個偉大的女人，而我比較幸運的，我擁有兩個偉大的女人，一個是我的母親，另一位則是佩萱。你們總是在我喪志的時候給我當頭棒喝，告訴我不能軟弱，告訴我一個男人必須具有肩膀。還有爸爸以及在英國的哥哥，不時的關心著我的進度，告知我不能害怕、不能退縮。

兩年的種種，我銘記在心。我很慶幸我生長在這個家庭，也很感謝出現在我生命中的每個人。在這感恩的季節，我無法忘記你們對我做的點點滴滴。謝謝你們的包容，感恩永於心中。

任芬傑　謹識於交大運管所

中華民國 98 年 7 月 23 日

目錄

表目錄

圖目錄

第一章 緒論

1.1 研究背景與動機

大眾運輸系統的發展，已成為各國改善交通問題之常用手段，然乘車相關資訊的缺乏，使乘客不知在何處搭乘大眾運輸系統，並於何時到達目的地。基於這些因素導致使用者降低搭乘大眾運輸的意願，並且減低大眾運輸系統之使用率。為了解決上述問題，行前旅次規劃系統因應而生，解決使用者搭乘大眾運輸工具的困擾，並進一步改善交通擁擠與空氣污染等課題。

本研究回顧的行前旅次規劃文獻，主要分為兩大研究方法：一、Huang 和 Peng [3]依照時刻表為基礎，提出一系列之最少成本演算法;二、以 Koncz et al.[6]發展之大眾運輸系統路徑規劃演算法（Transit Route Planning Algorithm ,TRPA）為首的最少轉乘演算法。在過去研究中，所提之演算法，無法同時間考慮兩種邏輯，且大部分僅考慮路線以及固定的時刻表兩種靜態資訊。然而較被後人所沿用的 TRPA，因演算法假設乘客在旅次中僅追求轉乘次數最少的方案，故將會低估起點至迄點之方案個數，而在某些情況下無法提供方案。例如：由甲地到乙地，同時擁有直達方案或一次轉乘方案，但因演算法的設計，導致直達方案無法搭乘時，也無法告知使用者，具有一次轉乘方案的替代方案。如此一來，若使用者希望在早上九點由甲地出發至乙地，但直達車十點才發出第一班車，而使用者將無法由 TRPA 演算法得知方案，但或許可在早上九點由甲地出發先至丙地，進行一次轉乘到達乙地。而因 TRPA 演算法的設計，將無法找到一次轉乘的方案。本研究者在張存保等人[11]一文中發現：『解決最佳路徑問題之演算法包括 Dijkstra 演算法、Floyd 演算法、矩陣算法，目前公認最好的方法為 Dijkstra 演算法』。但因大眾運輸網路存在特殊性，導致直接將 Dijkstra 演算法套入大眾運輸網路，將無法考慮轉乘次數帶給使用者的成本與不便。所以倘若要使用 Dijkstra 演算法套入至大眾運輸路網時，網路的架構必須能夠解決其特殊性。

本研究發現大眾運輸行前旅次規劃，無不是求解最少轉乘、最少旅行時間、最少票價等最少成本方案，且每個方案都是獨立且重要的。演算法不該特別重視某特定成本而忽略其餘成本，而當使用者設定了某特定成本時，其餘成本可作為限制進行旅次規劃。例如：使用者希望能夠得到，花費在一千塊內由高雄至台北的最少轉乘方案。故本研究的研究重心為，建立一個能夠考慮到大眾運輸路網的網路架構，接著將 Dijkstra 演算法進行修改，使其可套用至所有規劃邏輯。

本研究發現，過去的文獻，無法兼顧最少旅行時間與轉乘次數的多寡，而現實生活亦是如此。例如，市區公車中的直達車有時會產生繞路的現象，比如說，高雄市 218 號公車，由楠梓加昌站出發至高雄火車站，而 218 的路線卻因要服務九如路之居民，而行駛較遠的路線，導致由加昌站出發的乘客，可能必須花費較多的旅行時間方能到達目的地。實際上，218 路線可於左營北站轉乘 5 號公車行

駛中華路，此時，行駛時間將少於直達旅次。而在長途旅次中，苗栗至高雄可搭乘臺鐵直達火車，但此直達方案，遠比搭乘火車到達台中烏日高鐵站，轉高鐵至高雄來得久。轉乘對於使用者來說，是一種負成本，因使用者必須多花時間等車，以及必須再多支付費用。由於每個人對於轉乘次數與旅行時間之替換(trade-off)程度並不相同，所以，本研究選擇最少旅行時間作為主要邏輯，轉乘次數作為限制式進行旅次規劃。

本研究第二個研究重心為：如何在行前旅次規劃中增加動態資訊。本研究所指的動態資訊，不只包含在使用者查詢時，該時間點之動態資訊，尚考量未發出之班次時刻表。若行前旅次規劃系統能考慮動態資訊，將可進一步規劃更為理想的方案，若使用者原預定搭乘 1 號路線至火車站搭乘火車，但該火車班次發生誤點情事，使得使用者，若依照時刻表規劃出的方案至火車站時，必須等待較多時間進行轉乘。此時，行前旅次規劃系統若考慮到動態資訊，使用者可在出發前查詢該時間點之動態資訊，如此一來，可把在火車站等待誤點班次的時間，轉移為在家中的等待時間。提供使用者晚一些時間出發搭乘 1 號路線，仍可搭乘原預定班次，或可更改原訂計畫改搭乘別班次或別種運具。回顧目前國內線上之大眾運輸行前旅次規劃系統，大部分皆尚未考慮動態資訊，故本研究希望能夠將動態資訊考慮至行前旅次規劃。回顧國內系統，大部分之系統皆有考慮時間窗之限制，例如：預定何時出發、預定何時到達等時間窗限制，故本研究在此也將時間窗的限制考慮至行前旅次規劃邏輯。

基於上述原因，本研究將利用修改過之最短路徑演算法，以符合大眾運輸追求最少成本之概念，並以加強式網路架構解決大眾運輸之特殊性，且利用最少旅行時間邏輯為例並考慮動態資訊，進一步規劃出在轉乘次數與時間窗限制下，結合動態即時資訊與時刻表之最少旅行時間方案。

1.2 研究目的

提出一套適用於各種成本的演算法與大眾運輸網路架構，使得各種成本都可單獨考慮，不再受到演算法的設計而忽略其餘邏輯，並以最少旅行時間邏輯且結合動態資訊作為範例。然而對於旅行時間與轉乘次數，每個人的替換程度都不相同，故本研究中所發展出之演算法，將在使用者所設定的轉乘次數上限內，規劃出旅行時間最少的方案。

1.3 研究內容與範圍

本研究探討之問題為，如何利用已知的大眾運輸之路線、班次、即時資訊或時刻表，在最多容忍轉乘次數規劃出旅行時間最少的方案，當在考慮旅行時間最少時，本研究將忽略費用、步行距離等其他邏輯僅考慮轉乘次數之限制。因此，本研究所開發之演算法，應適用於各種有場站或者站牌的大眾運輸工具，但鑑於台灣地區長途客運，以及市區公車之中途站的動態資訊尚未健全，故本研究假設

動態資訊可得，並為了測試演算法之強固性，本研究先使用虛擬網路，作為範例進行強固性的驗證，最後利用新竹市區公車，作為實例測試本演算法之正確性。

1.4 研究步驟與流程

研究背景與動機：

主要針對過去文獻之旅次規劃邏輯的不足加以闡述，且描述並探討，與大眾運輸旅次規劃有關的背景資料，最後確認研究目的。

文獻回顧：

針對國內外相關的研究進行蒐集，蒐集內容包括：大眾運輸旅次規劃問題的定義、大眾運輸路網特性、旅次規劃分析方法、國內外相關旅次規劃資訊系統，充分瞭解國內外對於此問題之相關研究。

研究內容與範圍：

基於動機與目的，對於大眾運輸旅次規劃問題更進一步的定義，並界定研究範圍與對象，設定必要的前提與假設，以避免無法掌控或處理的狀況。

資料庫建構：

用資料庫描述實際大眾運輸路網以及匯入所需資料，提供旅次規劃邏輯設計之用。

旅次規劃演算法設計：

設計演算邏輯並用程式撰寫。

旅次規劃邏輯驗證：

在虛擬網路部份，利用演算法所規劃出的解與窮舉法進行比對。而在新竹市區公車實例測試部份，則利用目前新竹市區公車行前旅次規劃系統：竹塹行前旅次規劃系統進行比對，驗證本演算法的準確性。

問題檢討：

修正測試時所遭遇到的問題，並說明產生此問題的原因與解決辦法。

結論與建議：

根據上述之分析，提出本研究之結論與建議。

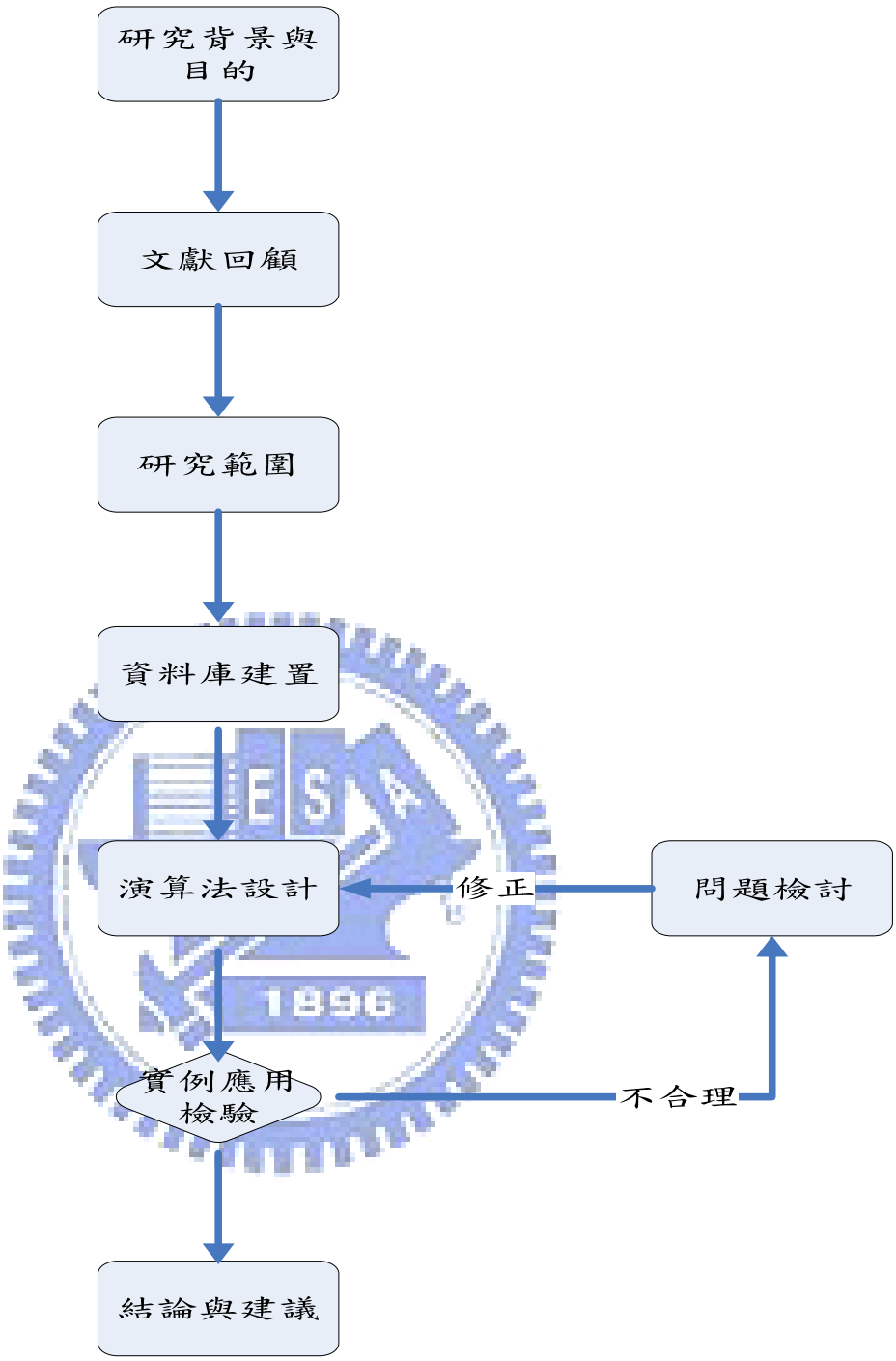

圖 1.1 研究流程圖

第二章 文獻回顧

旅次規劃決策需考量不同運具或路線間轉乘及時刻表因素，而這些因素使得在多運具運輸網路下進行決策時更顯複雜。Qiang. Li [8]將大眾運輸旅次規劃更明確定義為，多時窗限制下的多目標整數規劃問題，其模式如圖 2.1 所示。這意味著，大眾運輸旅次規劃模式，乃屬 NP-hard 問題，如今並沒有求解方法，可以直接確定並有效處理多目標整數規劃模式。

目標式

- 最小總旅行時間
- 最小總等待時間
- 最小轉乘成本

限制式

- 時間的限制上，則是大眾運輸工具根據時刻表到達一個預先排定的停靠站上，(時間點)並且必須在下一個離開班次離開。

圖 2.1 多時窗限制的大眾運輸旅次規劃模式

而張存保等人[11]一文提出，解決最佳路徑問題之演算法，包括 Dijkstra 演算法、Floyd 演算法、矩陣算法，目前公認最好的方法為 Dijkstra 演算法。但直接將 Dijkstra 演算法套入大眾運輸網路，存在著因大眾運輸網路之特殊性，而無法考慮到轉乘次數的問題。基於大眾運輸網路的特殊性，過去研究發展出適合大眾運輸旅次規劃演算法，大致上可分為搜尋法及最少成本演算法。而本研究的文獻回顧，分為四個部份。一、分析大眾運輸網路的特殊性，說明何謂大眾運輸特殊性，且該特殊性如何使最短路徑演算法無法直接套入;二、以追求最少成本演算法，應用於大眾運輸行程規劃;三、以 Koncz et al.[6]為首之 TPRA，應用於大眾運運輸行前規劃;四、國內目前系統回顧，用來分析實用的行前旅次規劃系統，需要哪些功能。

2.1 大眾運輸路網特性分析

Huang 和 Peng [1]提及在行前旅次規劃的分析中，最佳路徑的計算，可說是最重要的一環，尤其可提供旅行者以最快時間或最短距離抵達目的地，因此，此一最佳路徑邏輯處理的正確性與效率性，將是旅次規劃的關鍵要素。然目前大多數的最佳路徑演算法及程式皆設計給公路網路使用，Huang 和 Peng [1]更明確指出，應用於公路網路之路徑演算法，並不適合用來解決大眾運輸的路徑找尋問題。

縱然，現有的網路分析和最佳路徑演算法，應用於公路路徑和交通指派上十

分良好，不過，當這些方法套用在大眾運輸路網中卻滋生許多問題，例如：轉乘次數必須要計算。主要原因，來自大眾運輸路網，與公路路網彼此間具有不同的特性。一般針對公路網路所設計的演算法，不容易處理大眾運輸等車、轉車、及其他服務水準因素。因此，不適用於大眾運輸路網連線的搜尋。本研究參考何文基[10]的研究，將大眾運輸與公路路網特性差異，歸納為以下五點：

一、轉乘到達時間相依

不論是跨路線、或跨運具進行轉乘時，必定受到另一條路線班次影響，例如：有位旅客預計從高雄至台北，早上八點搭公車至火車站，其到達時間八點二十分，而欲轉乘火車出發至台北的班次，則僅能等待八點二十分以後之九點的班次。這說明大眾運輸規劃的演算法，不僅求取空間上的可行路徑即可，尚需找尋時間相配合的連線路線。

二、不同路線包含相同停靠站

同路段可能包含多條路線，而多條路線可能包含相同停靠站。例如，台北火車站就包含眾多公車路線、鐵路與捷運路線，顯示出，大眾運輸路網節線，與一般公路網節線的意義並不相同。台北車站的站牌包含了，14、2、218、220、260、274、299…等公車路線，顯示同一公車站牌包含多條路線的情形。

三、起迄點來回路線非對稱

除了一般軌道系統外，大眾運輸工具往返程路線常常不一致。部分停靠站在整條路線中，只單獨在往程路線設站，抑或在返程路線唯一設站。例如：公車路線是有方向的，同一條公車路線在兩個行駛方向上，其經過的停靠站不一定完全相同。簡單來說，大眾運輸網路存在著單邊設站的情況。

四、存在轉乘消耗

私人運具路網與大眾運輸路網的連通性含義不同。在私人運具路網方面，其道路交叉點連接多條路段，車輛在交叉點可任意的從一條路段到另一條。大眾運輸路網中，若多條路線交叉於空間上的同一停靠站，則旅客在此轉車存在轉乘消耗，包含時間消耗、費用消耗等。

五、大眾運輸路網包含不同運具的服務路線

大眾運輸路網，由許多不同速率的運具及其服務路線所組成。例如公車、捷運系統及鐵路系統等不同運具的服務路線。因此，在大眾運輸路網中除了可轉換不同路線外，尚可轉換不同系統的運具。

從演算法的角度來看，Huang 和 Peng [1]認為，一般公路網路在路段上和交叉路口中，都會給予唯一旅行時間和轉乘懲罰值，然而，大眾運輸系統在某一路段上可能包含很多運輸路線，且每條路線還擁有它的班距時間，使得估計及計算轉乘懲罰值的過程甚為艱鉅。

上述的特性差異，導致以公路網路為基礎所發展之最佳化路徑演算法，並無法直接適用於大眾運輸路網分析。因此，在發展大眾運輸旅次規劃分析方法過程中，即必須有效解決大眾運輸路網特殊性的問題。

2.2 大眾運輸旅次規劃方法回顧

過去文獻有關大眾運輸路徑找尋之演算法可歸納為：最少成本法以及搜尋法。以下，將所搜集之大眾運輸旅次規劃方法、重要文獻分為此兩部份回顧。

2.2.1 最少成本分析法

McCormack [2]利用物件導向程式設計，並提供合適的網路架構以及啟發式解法，解決多種運具旅次規劃問題。該文將多種運具整合成一個網路，包括公車網路、道路網路、鐵路網路。其中節點由公車站牌、建築物地標、火車站、道路以及多種運具之轉乘站。兩節點間若具有道路經過或具大眾運輸行駛路線，則以路線節線（Route link）相連。兩種運具之站牌相同或步行可到達時，兩節點以相連節線（Connection link）相連。然成本部份，則會依照使用者所選擇的決策而改變。使用者希望是找到最短旅行距離，成本為兩節點間的距離，若使用者追求旅行時間最短或者花費最少，節線成本更新為旅行時間或費用。且利用 A*演算法來計算最短路徑。

Huang 和 Peng [1]等人提出一套，結合以班距為基礎和考慮時刻表之混合式演算法，其特色是在每個交叉路口中，利用平均班距計算出懲罰值，再以藤構式(vine-building-based)路徑找尋演算法找出最佳路徑，之後，再加入時刻表資訊得出規劃結果，其中應用到的技巧是，加入虛擬節點以利藤構式演算法進行路徑分析，並且每半小時處理一次點對點最短路徑，最後，將結果存於伺服器中，藉以掌握不同時間內所提供的路線服務狀況。

Huang 和 Peng [3]所發展之演算法，屬於以時刻表為基礎之分析方法，包含向前、向後以及轉乘演算法。其主要邏輯概念如同向前搜尋法，該文利用向前搜尋法說明邏輯如下：(1)首先以起點作為根點；(2)當根點不為空集合則找尋每個根點的分支點；(3)判斷最早可能到達分支點的時間為存在到達根點的時間和所有從根點出發到達分支點的時間最小，假設 $t_j^{r_{ij}} > t_i + wt_i$ （$t_j^{r_{ij}}$ 為每個路線段 i 到 j 的到達時間，t_i 為到達時間，wt_i 為等待時間），意即沒有到達 j 點的時間比到達 i 點的到達時間加上等待時間更早；(4)重複步驟(1)至(3)直至根點為空集合。Huang 和 Peng [4]最後提及將此演算法加以利用物件導向之資料庫設計，能夠有效的減少運算時間。

Hickman[5]利用 AVL（Automatic vehicle location）資料加上時間相依網路，套用於大眾運輸網路。文中的網路架構如下：節點由站牌所形成，若兩站牌有路線行駛，則給予節線。節線成本為旅行時間，以及該旅行時間所發生的機率。例

如，此路段在某一時段，由 AVL 資料得知三種可能的旅行時間：40、42、38 分鐘，而發生的機率分別為 0.7、0.2、0.1。而資料庫中，此路段總旅行時間為 40 分鐘，也就是有 0.7 的機率會準時到達。文獻希望能夠提供使用者完整的路線資訊，其中包括了幾點：第一，是否能夠依照路線的時間到達目的地；第二，最晚什麼時候能夠到；第三，有多高的機率不會錯過轉車。文獻中將使用者的效用函數設為總旅行時間的函數，當旅行時間越少則效用越高。使用最短路徑演算法作規劃，在使用者輸入的時間範圍內，起點至迄點找到一條最短可能旅行時間路線。當有兩條相同最短可能旅行時間時，將路線分為路段依照該時間範圍的機率去相乘，取最大者作為最佳路徑。

以上的最少成本演算法，雖然能夠考慮到由起點至迄點之最少成本方案，但卻沒有考慮到轉乘次數的多寡，將會影響使用者搭乘之意願。故本研究認為，即使追求最少成本，也必須考慮轉乘次數的增加帶給使用者的不便。

2.2.2 搜尋法之分析方法

Koncz et al.[6]所發展之大眾運輸系統路徑規劃演算法（Transit Route Planning Algorithm ,TRPA）、張存保等人[10]城市公交問路系統、劉偉賢[8]結合長短途客運之轉乘邏輯、何文基[9] 整合時刻表之大眾運輸行前旅次規劃分析方法，皆屬於此類方法，此方法之基本假設為：將轉乘次數最少設為主要目標式。以下就邏輯部分敘述如下：

Koncz et al.[6]所發展之大眾運輸路徑規劃演算法(Transit Route Planning Algorithm, TRPA)步驟包括：(1)窮舉出起迄點步行可及的搭乘路線；(2)分別比對每一個起點路線，對於迄點路線是否具有 0 哩或者 0.4 哩距離範圍內的交叉點，並建立連結矩陣；(3)從連結矩陣中搜尋起迄點可行之路線；(4)若起點步行範圍內之路線(路線 A)與迄點步行範圍內之路線(路線 C)相符合，則儲存直達轉乘方案路徑，若不存在直達路線，則繼續下一步驟；(5)比對路線 A 與路線 C 所有上下車站點，是否具有步行範圍內 0.4 哩之交叉點，定義此路線為路線 B；(6)若存在為一次轉乘方案，結束搜尋。

張存保等人[11]城市公交問路系統之最佳路徑演算法則包括：(1)輸入起始站點 A 和目的地站點 B；(2)在公車站點資料庫中查出經過站點 A 的公車路線 L(i)（i=1,2,3,…,m,m 為正整數），以及經過站點 B 的公車路線 S(j)（j=1,2,3,…,n,n 為正整數）；(3)判斷是否有 L(i)=S(j)，若有一條路線滿足要求，則該公車路線即為最佳路線，輸出結果並結束運算；若有幾條路線滿足要求，則從公車路線資料庫中查出路線經過的站點，然後從鄰接站點距離資料庫中查出各站點間的距離，計算各條公車路線的距離，選擇一條距離最短的路線即為最佳路線，輸出結果並結束運算；(4)從公車路線資料庫中查出經過站點 A 的公車路線 L(i)的站點 E(i,g)（i=1,2,3,…,m；g=1,2,3,…,n；m,n 為正整數），以及經過站點 B 的公車路線 F(j,h)（j=1,2,3,…,p；h=1,2,3,…,q,p,q 為正整數）；(5)判斷是否 E(i,g)是否等於 F(j,h)，

若有一條站點滿足要求，該站點即為一次轉乘站點；(6)從公車站點資料庫中查得經過 E(i,g)的公車路線 T(k)（k=1,2,3,…,m,m 為正整數），從公車路線資料庫中查得路線 T(k)的站點 G(k,w)（k=1,2,3,…,m；w=1,2,3,…,n；m,n 為正整數）；(7)判斷是否有 G(k,w)=F（j,h）。若有某個站點 E 滿足要求，則該站點 E 為第二個轉乘站點。按照步驟 4 至 6 方法，求出從起始站點 A 到站點 E 的一次換乘路線，再按照 2 及 3 的方法，求出從站點 E 到目的站點的最佳線路。

劉偉賢[9]之長短途客運轉乘邏輯之主要網路分析模組步驟，包括：(1)分析起迄點是否在步行限制內，當使用者輸入之起迄點於步行限制內，代表無需使用汽車客運來規劃旅次，故直接告知使用者圖形的步行導引規劃方案結果；(2)搜尋資料庫中，起迄點重要地標在步行限制內所有的站牌及路線，比對起迄點的路線是否相同，若是相同代表具備直達方案，故系統即可輸出直達方案；(3)若起迄點路線皆不相同，比對起點與迄點的路線中是否相同，若相同則代表具備一次轉乘；(4)當一次轉乘亦無法組合出可行方案時，系統開始搜尋起迄點重要地標內，所有站牌路線中其他的站牌，比對其他站牌於步行限制內，可以步行到達的其他路線站牌，而此站牌所對應的路線，恰巧可成為連結起點路線與迄點路線的第三條路線時，便可透過結合此第三條路線與起點路線、迄點路線完成兩次轉乘的規劃方案。

何文基[10]利用了劉偉賢[9]之架構，加上了固定之時刻表，提供了考慮時刻表之大眾運輸行前旅次規劃。然實際上，並不是每一種運具都具有明確的時刻表，故何文基[10]利用，出發站的時刻表加上歷史旅行時間資料，進一步的得知在尖、離峰各路公車到達該站的歷史資料，進而規劃方案。轉乘發生時，則安排使用者在最近的轉乘站進行轉乘與搭乘。轉乘的班次連結，則是安排使用者在轉乘站，過濾掉銜接不上的班次。該研究法先得到最少轉乘的方案，進一步依照，最少轉乘的路徑中進行最快到達、最少步行等邏輯。

Lui[7]將一個站，同時擁有 15 條路線經過時視為一個中心(hub)，若一路線所通過的站牌中並沒有同時經過 15 條路線，則選擇最多路線通過的站牌視為中心，規定每一條路線必須具備兩個中心。中心與站牌為分別為，上層與下層的網路節點並結合階層式網路。

演算法一開始判斷起迄點是否相同，相同則不需要繼續演算。若起迄點不相同，並且有一條路線能夠同時經過起迄點，在此條路線上起點較迄點早被服務，此時輸出直達方案。當直達無方案時進入一次轉乘，路線一先經過起點再經過轉乘站 A，路線二先經過轉乘站 A 再經過迄點，則輸出一次轉乘方案。當直達與一次轉乘並未能找出方案時，演算法進入上層網路，找尋直達或者一次轉乘的方案。最後一步為結合上下層網路。

蘇夢豪等人[12]在交通部運輸研究所，發展出一套城際旅運規劃邏輯，其步驟說明如下：(1)找出所有從起點到終點的路徑，若有解則進入(2)，無解則結束。(2)找出每個路徑的路線班次，若無解則結束，有解則進入(3)。(3)查詢各段票價後結束。而在步驟(1)首先判斷起訖點是否皆為大都市，若是則僅查詢直達方案，否

則判斷直達方案是否存在。若不存在，而使用者也只查詢直達方案時，結束演算法。若使用者不只查詢直達方案時，將判斷路徑的第二站是否為終點站，若是則結束，否的話則進行下一站是否為終點站，直到下一站為終點站或以超過轉乘次數限制則停止。接下來，上述得知的可能路徑中，仍須在步驟(1)判斷該路徑是否在合理之時間範圍內，若是則保留路徑，否則捨棄路徑。

小結：

本研究所回顧之搜尋法，能夠找到最少轉乘的方案。Lui[7]將網路以及演算法分為兩部份，突破了轉乘兩次的限制，但仍有地方稍不恰當。例如：當使用者輸入的起迄點，在下層網路無法提供方案時，該演算法將會進入上層網路進行搜尋。上層搜尋完畢之結果，尚須與下層網路進行結合。此時，轉乘次數分為三個部份：(1) 由起點至起點中心 (2) 起點中心至迄點中心 (3) 迄點中心至迄點。此時，有可能在下層網路兩次轉乘就能夠到達，但因演算法的設計，而進入上層網路，導致規畫方案的轉乘次數多於實際的轉乘次數。

由於 TPRA 著重於轉乘次數最少，因此，其演算法得到的最快到達，最少步行等方案，也是在最少轉乘的條件下求得的。而有時可能多一次的轉乘可以減少旅行時間，而 TPRA 並不能夠得到此解。本研究將所回顧的行前旅次規劃文獻，根據其目標式作為分類，並說明其研究重點、優缺點。如表 2.1 所示。

表 2.1 文獻回顧整理

目標式	作者	研究重點	優缺點	是否結合動態資訊或時刻表
最少成本	McCormack [2]	※ 將多運具同時考慮在路網上。利用啟發式解法限制路網大小，以及利用 A* 演算法計算出最少成本的路徑。 ※ 系統將具有多種類的成本，以方便當使用者選擇規劃邏輯的不同而計算出最少成本的路徑。	※ 最短路徑演算法無法考慮到轉乘次數的多寡，有可能會提供轉乘次數過多的現象。 ※ 求解速度較快。	無
	Peng[1]	※ 以互動性與功能性	※ 混合式的方法可	有

		將當時運輸系統分類。 ※ 提出以班距和時刻表為基礎的混合式網路分析方法。	以掌握到，以班距為基礎和以時刻表為基礎分析方法的優點。 ※ 此種方法未考慮轉乘次數的影響。	
	Peng[3]	※ 提出向前、向後、最小轉乘時刻表演算法	※ 不僅可求得使用者預計出發時間的連線，尚可求得預計到達時間的連線。 ※ 大路網處理費時，需以物件導向資料庫輔助。	有
	Hickman[4]	※ 考慮了大眾運輸路網具有時間相依特性，並依照各路段的機率，旅行時間提供使用者最大機率到達迄點的方案。	※ 本文獻的機率旅行時間屬於歷史資料，不能有效反應目前的路況。 ※ 無法考慮到使用者轉乘時的心理感受。	有
最少轉乘次數	Kconz[5]	※ 提出大眾運輸路徑規劃演算法 ※ 旅行時間的計算方式	※ 使用可接受步行距離的概念，來過濾路線找尋可行方案。 ※ 提供多個最佳化路徑實為優點，但可行的方案數太多，對於使用者而言需額外花費時間篩選。	無
	劉偉賢[8]	※ 針對現有旅次規劃系統的介面設計與功能進行探討 ※ 提出長途客運允許	※ 前處理的過程可以減少線上處理負擔，增加處.理速度。	無

最少轉乘次數		二次轉乘分析的演算方法	※ 前處理過程耗時，對於經常性更動的路線或增加的點位資料，難以確保資料庫資料為最新。	
	何文基[9]	※ 延續劉偉賢的碩士論文，將其納入時刻表，提供具有時刻表方案的轉乘資訊。 ※ 利用事前運算好的路徑，再搭配時刻表與邏輯方案運算方案。	※ 納入時刻表提供使用者方案。不論是最快到達、最少步行皆是在起迄點的最少轉乘的路徑中，找尋使用者所選取的邏輯。並不能夠獨立算出最快到達的方案。	有
	Chao[7]	※ 將 Hub 的觀念帶入大眾運輸路網，利用階層式的網路增加搜尋法的廣度，不再局限於高於兩次就會帶來太複雜的演算次數。	※ 因為演算法設計的關係，有可能高估轉乘次數，導致較為不合理的方案產生。	無
	張存保[10]	※ 提出找尋公車最佳路徑的演算法。 ※ 說明大眾運輸路網特性。	※ 以資料庫設計解決大眾運輸路網來回路線非對稱問題。	無

2.3 目前國內行前旅次規劃系統回顧

Huang 和 Peng [1]認為，由於資訊在網際網路上取得方便，線上的大眾運輸旅次規劃系統(online transit trip planning systems)愈來愈受歡迎，不僅業者可減少對客服人員的支出，使用者可在提供使用網路的處所，任何時間、任何地點得到資訊。雖然，國內外已具備許多旅次規劃功能相關的線上交通資訊系統，這些系統根據大眾運輸環境的不同，發展出不同的需求特性與運具型態。但因時間的限制，本研究僅蒐集國內相關線上旅次規劃系統，彙整出限制條件與功能特色作為設計演算法之參考。

在回顧國內行前旅次規劃系統時，本研究將其分為長途旅次規劃系統，以及市區旅次規劃系統，且彙整出長途旅次以及市區旅次的不同處。

一、交通服務 e 網通[14]

該網站為交通部運輸研究所構建，網站內容包含即時資訊查詢、客運訂票、轉乘資訊提供和旅運規劃四大項。旅運規劃的區域，涵蓋台澎金馬等 26 個縣市，起迄點的輸入包括車站、交叉路口、重要地標、觀光景點，當選擇了其中一項後，系統將依據不同的選項，選擇縣市與地標，且在旁邊的地圖中，將顯示所選取的地標。使用者可在下方選擇所希望的規劃邏輯，其中包括最快到達、最少轉乘、最少步行及最少費用。當使用者設定完所有的參數後，系統將提供使用者方案。

二、台灣鐵路局旅次規劃系統[18]

臺鐵之鐵路行程規劃系統，整合現有之火車時刻查詢、網路訂票及線上付款系統，提供旅客安排行程的選擇方案與訂購票服務。台灣鐵路為全台各縣市的交通運具，系統將路線分為西部幹線、東部幹線、北部幹線等。當使用者選定了任何一部分的幹線，接著再選取起迄站名、設定出發日期，系統將會提供多種可行方案。

三、高鐵旅次規劃查詢系統[19]

高鐵網站所提供的行程規劃，採用步驟式的選擇方式，讓民眾一個步驟、一個步驟地選擇自己所需要的條件。如此，可避免使用者在複雜的條件選項下，面臨不知從何處下手的困擾。主要分為三個步驟：指定起終點、選擇交通工具、指定搭乘日期。在起終點的指定上，又可以地名、景點地標、運輸轉運站與關鍵字等四種方式來查詢。其地名之決定，主要採用鄉鎮市區之劃分，若為直轄市或省轄市，則再往下分至區，並加上一重點市區與其它重要區域，例如：新竹市分為東區、南寮、香山、新竹市區、新竹科學園區等。

四、台北大眾運輸旅次規劃系統[15]

本網站由台北市政府交通局建製。網站提供包含旅遊資訊、公車動態資訊系統、住宿資訊、路況報導、大眾運輸、無障礙乘車等資訊。其中，大眾運輸轉乘查詢功能服務區域包括，台北市和台北縣。起迄點輸入設計為公車站位、重要地標、交叉路口等分類。接著依照使用者所選取的查詢分類，進一步要求使用者自行輸入詳細名稱，例如，重要地標-台灣大學。運具種類，有公車、捷運兩種大眾運輸系統的轉乘規劃服務，而轉運方式，則是鄰近服務路線，站牌 500 公尺以內允許轉乘。

台北大眾運輸旅次規劃查詢，也具有公車動態資訊系統，告知使用者所查詢的路線、去程及回程公車的正確位置，然此動態資訊系統尚未與旅次規劃系統整合，大約 30 秒更新一次，目前公車所在的位置。

五、新竹市旅運規劃系統[20]

新竹市區旅運規劃系統，操作畫面有四個步驟：1.設定出發地，2.設定目的地，3.設定出發時間，4.使用者自行輸入條件。第一步驟及第二步驟，都是先利用類別將地標分類，種類大致上有：學校、公園、休憩場所等重要地標，第三步驟是輸入欲出發之時間，第四步驟則是請使用者自行輸入單次等車，最大容忍時間以及最大容忍步行距離。

六、台中公車動態與網路轉乘系統[16]

此系統為台中市交通局所建構，其操作方式與介面，與台北縣市的大眾運輸轉乘系統相類似。起迄點輸入設計為地標、公車站牌名稱、交叉路口等分類，再由以上的分類進行詳細名稱，例如：學校地標：東海大學，運具種類則只有公車。和台北縣市一樣具有公車動態資訊查詢、以及轉乘規劃系統。公車動態資訊與轉乘規劃系統也尚未作整合。

七、台南市公車行前路線查詢[21]

台南市區公車行前路線查詢系統，僅需輸入上車地點以及下車地點，而地點的選擇也是利用建築物的種類分類，再進一步進行詳細的地標選擇，例如，選擇了警察局的類別，系統將把台南市所有警察局列出，供使用者選擇，欲出發或到達之警察局。

八、高雄市公車轉乘系統[17]

此網站為高雄市公共汽車管理處，委託建置之交通專業網站。其提供之資訊服務，包括公車動態查詢、公車到站時間預約報知服務、及公車轉乘查詢服務等多項服務功能。可提供不熟悉路線的乘客，利用這些功能，達到前往目的地。使用者進入系統後，選擇上車地標、下車地標及預計出發時間，系統除列出相關轉乘路線資訊供參考外，尚提供預估到站時間及旅程時間。另一方面，除有直達目的地的建議方案外，另有提供可行的轉乘點服務，乘客可預知所花費的時間。

回顧國內系統，可彙整得表 2.2。表中可觀察出，目前國內的行前旅次規劃系統，皆未考慮到動態資訊。而在起訖點輸入的選項，也可得知長途起迄點設定，大部分皆以站名作為選項，而市區公車則大部分皆以重要地標、交叉路口或站名作為出發的設定值。

表 2.2 國內行前旅次規劃系統整理表

系統名稱	起訖點輸入資料	規劃準則	運具類型	時刻表	動態資訊
交通服務 e 網通	各縣市重要地標、交叉路口、站名	最快到達、最少票價、最少步行、最少轉乘	飛機、海運、城際客運、高鐵、臺鐵。	有	無
台灣鐵路局旅次規劃系統	站名	可行方案	臺鐵	有	無
高鐵旅次規劃查詢系統	站名	可行方案	高鐵	有	無
台北大眾運輸旅次規劃系統	重要地標、交叉路口、站名	可行方案	捷運、市區公車	無	無
新竹市旅運規劃系統	重要地標、交叉路口、站名	可行方案	市區公車	無	無
台中公車動態與網路轉乘系統	重要地標、交叉路口、站名	可行方案	市區公車	無	無
台南市公車行前路線查詢	重要地標、交叉路口、站名	可行方案	市區公車	無	無
高雄市公車轉乘系統	重要地標、交叉路口、站名	可行方案	市區公車	無	無
可行方案：代表該系統僅要求使用者輸入起迄點以及出發時間，而提供出的方案，並無任何邏輯(最少票價、最少轉乘等)之方案。					

除表 2.2 彙整的資料外，本研究尚發現，城際及市區旅運有兩點不同之處。一、城際運具因停靠站之間的距離，相較於市區運具過長，故在同一班車上具有階梯式的票價，例如：高鐵由台北出發至左營，途中停靠板橋、桃園、…、左營，票價皆不相同。而市區旅運則只有一種費用。因此，本研究認為在票價的部份，城

際旅運與市區旅運不同。二、城際旅運有只可上車或下車的站牌，例如，新竹客運由新竹火車站，開往台北火車站之城際客運，僅可在新竹市區上車且台北市區下車，此現象有別於市區公車之單邊設站現象。即使市區公車發生單邊設站，也不會限制乘客上下車。故本研究認為，若要同時間考量城際及市區公車，必須將此部份考量進去。本研究所發展的網路架構，主要考慮市區公車的單邊設站，以及往返路線不相同的現象，並未考量城際旅運之上下車的限制。如以最少旅行時間作為邏輯規劃範例，也尚未考量到票價的問題，而此部份可供後續研究者繼續進行研究。

2.4 結論

由以上回顧，本研究得知，若將整個大眾運輸行前旅次規劃目標式設計如同 Qiang. Li [8]的架構時，大眾運輸行前旅次規劃，將是一個 NP-hard 問題，如今，並沒有求解方法可以直接確定，並有效處理多目標整數規劃模式。回顧目前國內的系統，發現大眾運輸行前旅次規劃，可同時供使用者自行選取規劃邏輯，如最少轉乘、最少旅行時間、最少票價以及最少步行。本研究認為，大眾運輸行前旅次規劃，在追求最少成本的概念，且當追求某特定邏輯時，僅能有一個目標式。如當追求最少旅行時間，僅能將最少旅行時間作為目標式，無須再多考慮其餘邏輯，本研究認為，其餘邏輯可作為限制式，例如，在兩次轉乘內追求最少的旅行時間方案。

本研究將 Qiang. Li [8]多個目標式，修改為追求最少成本，其成本可為轉乘次數、旅行時間、票價、步行距離，且本研究利用最少旅行時間，作為範例進行說明。而在限制式的部份，可以讓使用者自行設定，例如：使用者想得到花費在 1000 元內最快到達迄點的方案。本研究則是選擇了轉乘次數作為限制式，進行行前旅次規劃。進一步將 McCormack [2]定義節線的方式，套入大眾運輸網路，使其分為轉乘節線以及旅行節線，在進行求解時判斷是否符合限制式。將每條路線的站牌視為唯一的節點，節線上的成本則為到站時間相減。將大眾運輸網路，建構為第三章所述說之加強式網路。如此，目標式僅有一個，類似於最短路徑演算法，進行求解最少成本，詳細內容如第三章所述。

第三章 研究方法

為了解決大眾運輸具有單邊設站、往返路線不一致的特殊性，本研究利用加強式的網路架構，做為網路架構，接著利用本研究所發展出的演算法進行求解。

加強式的網路架構，將大眾運輸路網中的每條路線，所經過的站牌都視為唯一節點，並事先以每個站牌為中心，定義方圓 500 公尺作為可轉乘之範圍，且在此範圍內之其餘站牌，定義為該站牌可轉乘的站牌，進行前置資料處理作業。回顧國內目前所有線上的行前旅次規劃，起迄點的設置，大多為重要地標至重要地標，故在前置處理方面，尚必須利用以重要地標為中心，定義方圓 500 公尺作為可及範圍，且定義在範圍內之站牌皆為該建築物之可及站牌。

演算法設計的重點為考慮動態即時資訊，以及在使用者所設定之轉乘次數、時間窗限制中找到旅行時間最少的方案。故本研究必須同時考慮，公車動態到站時間以及既有的時刻表，用來考慮尚未發出之班次銜接。當使用者每一次查詢，本研究所開發的演算法，將會根據使用者所查詢之時間，以及時間窗限制與轉乘次數上限，進行讀取動態即時資訊，最後利用演算法進行旅次規劃。

本章節將說明下列三個部份：

1. 說明設計資料庫的理念，以及如何利用資料庫，建構出加強式網路之架構。
2. 說明如何依照使用者所設定之時間窗，即時讀取到站資訊以便求解。
3. 說明如何在使用者限定的轉乘次數限制內，規劃出最少旅行成本之演算法。

3.1 資料庫建構以及建構加強式網路描述

3.1.1 大眾運輸路網描述

在進行旅次規劃分析之前，必須先將實際大眾運輸路網，抽象成具有拓樸(Topology)性質的網路圖概念，以利於進行網路分析。本研究利用 SQL sever 2005 資料庫的建置來描述大眾運輸路網，且利用資料庫結構的設計來符合大眾運輸路網特性。

本研究演算法設計建構於，本節所描述的加強式大眾運輸路網基礎之上，配合圖 3.1 之簡單路網，說明大眾運輸路網中起迄節點(Node)、停靠站(Stop)、節線(Link)及大眾運輸路線(Route)之內容。

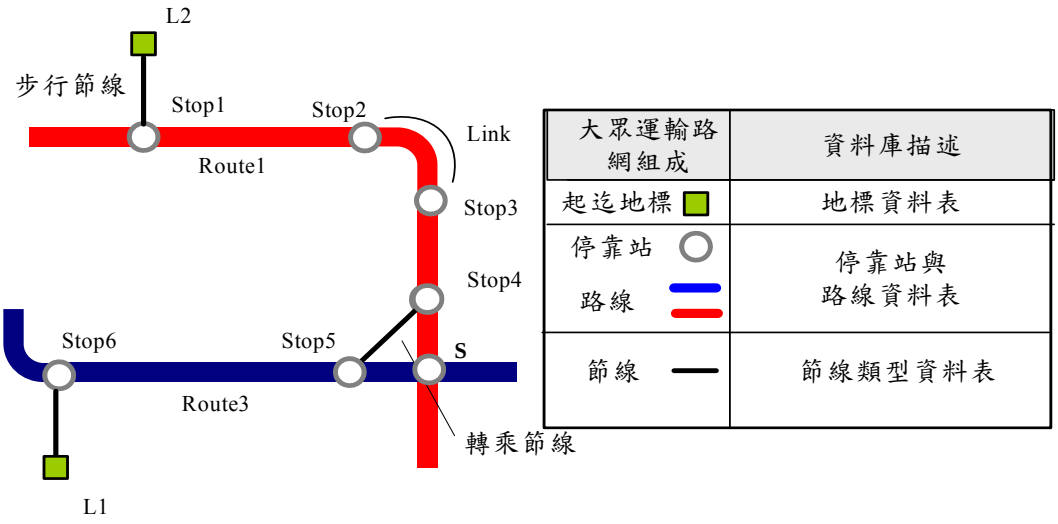

圖 3.1 大眾運輸網路組成示意圖

一、起迄節點

一般以重要地標代表旅次的出發點與目的地集合，可以是交叉路口、門牌號碼或某地方的公園、學校…等，如圖 3.1 中 Starting point 及 Landmark D，在資料庫中所屬重要地標資料表，欄位包含地標名稱和經緯度座標。

表 3.1 重要地標資料表

Name	Latitude	Longitude
L1	X100	Y100
L2	X110	Y110

二、停靠站與路線資料表

停靠站代表大眾運輸系統之場站或站牌，為使用者搭乘大眾運輸系統之上下車處，如圖 3.1 中之 Stop1、Stop2、Stop3、Stop4。路線代表大眾運輸路網中各大眾運輸系統之營運路線資料，如圖 3.1 中之路線 1 及路線 2。這些資料在資料庫中屬於停靠站與路線資料表，欄位包含停靠站代碼(StopID)、停靠站所屬路線代碼(ID)、停靠站名稱(name)、站序(sql)、路線名稱(Routename)、節點編號(NodeID)、所屬地標、班次，以及當班次出發時，該班次到達該站牌之動態即時資訊，或是當班次尚未出發時，紀錄之既有時刻表資訊、和經緯度座標。在所屬地標欄位部份，則是利用地標資料庫的座標資訊，定義方圓 500 公尺為可搭乘、可抵達範圍。利用直線距離公式計算出距離，若小於 500 公尺，則定義該站牌為該地標之可搭乘、抵達站牌，如圖 3.2 所示。圖中

藍色點代表的是建築物之中心，其餘黑色的點，代表方圓 500 公尺範圍內之節點。若範圍內一條路線同時有多個站牌可搭乘，此時，選擇與建築物直線距離最短的站牌為可搭乘、可抵達站牌，如圖 3.3 所示

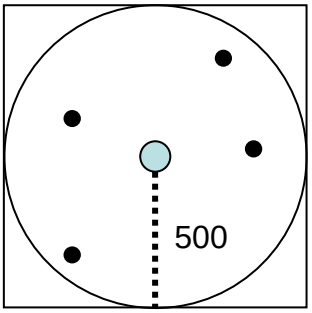

圖 3.2 判斷轉乘可及範圍示意圖

表 3.2 停靠站與路線資料表

ID	RouteName	Name	StopID	Sql	NodeID	班次	Latitude	Longitude	所屬地標	到站時間
1	1(往)	Stop1	1	1	1	1	X1	Y1	L1	Time1
1	1(往)	Stop2	2	2	2	1	X2	Y2	0	Time2
1	1(往)	Stop3	3	3	3	1	X3	Y3	0	Time3
1	1(往)	Stop4	4	4	4	1	X4	Y4	0	Time4
1	1(往)	S	5	5	5	1	X5	Y5	0	Time5
2	1(返)	S	5	1	6	1	X5	Y5	0	Time6
2	1(返)	Stop4	4	2	7	1	X4	Y4	0	Time7
2	1(返)	Stop3	3	3	8	1	X3	Y3	0	Time8
2	1(返)	Stop2	2	4	9	1	X2	Y2	0	Time9
2	1(返)	Stop1	1	5	10	1	X1	Y1	0	Time10
3	2(往)	S	5	1	11	1	X5	Y5	0	Time11
3	2(往)	Stop5	6	2	12	1	X6	Y6	0	Time12
3	2(往)	Stop6	7	3	13	1	X7	Y7	L2	Time13

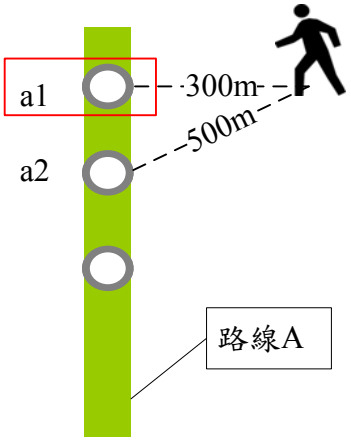

圖 3.3 起始站之選擇示意圖

三、節線類型

節線類型代表連接兩個節點(可能為起迄節點或停靠站)間之狀況，本研究除了根據大眾運輸路線之服務順序，給予旅行節線，尚需包括兩條不同路線場站間之轉乘節線，如圖 3.1 中所示 Stop2 與 Stop3 為旅行節線相連，而 Stop4 與 Stop5 為轉乘節線相連。將這些資料建置在資料庫中之節線類型資料庫，欄位包含代碼(ID)、起始路線代碼(FromRouteID)、起始節點代碼(FromID)、起始節點名稱(FromName)、到達路線代碼(ToRouteID)、到達節點代碼(ToID)、到達節點名稱(ToName)以及節線類型編號(Type)：1 為旅行節線、2 為轉乘節線。以下，將說明如何判斷兩節點之間的連結狀況

倘若兩節點屬於相同路線編號，如圖 3.1 中 Stop1 與 Stop2 與其所對應之節點編號分別 1 與 2。由於路線編號 1，先服務 Stop1 後服務 Stop2，因此，本研究給予一條由節點編號 1，出發至節點編號號 2 之旅行節線。

轉乘節線的部份則必須利用每個站牌為中心，定義方圓 500 公尺作為可轉乘之範圍，若有其餘站牌在範圍內，且範圍內之站牌並不與目前站牌屬於相同路線編號，以及並不屬於目前站牌所屬路線之反程路線。例如圖 3.1 與表 3.3 中的 Stop4 為路線編號 1 的站牌，且以 Stop4 作為中心，方圓 500 公尺搜尋到：屬於路線編號 2 的 Stop4 與 S 站、屬於路線編號 1 的 S、與路線編號 3 的 Stop5 以及 S 站。此時，路線編號 1 之 Stop4，僅能夠與路線編號 3 的 Stop5 以及 S 站給予轉乘節線相連。因為，路線編號 1 與路線編號 2，同時為 1 路線之往返程，故並不給予轉乘節線，而對路線編號 1 之 S 與 Stop4 因為屬於相同路線編號，故亦不給予轉乘節線聯繫兩節點，如表 3.3 所示。

表 3.3 節點類型資料表

ID	From RouteID	From ID	From Name	To RouteID	To ID	To Name	Type
1	1	1	Stop1	1	2	Stop2	1
2	1	2	Stop2	1	3	Stop3	1
3	1	3	Stop3	1	4	Stop4	1
4	1	4	Stop4	1	5	S	1
5	1	4	Stop4	3	12	Stop5	2
6	1	4	Stop4	3	11	S	2

3.2 最短旅行時間演算邏輯建立

3.2.1. 演算法假設

一. 假設使用者追求旅行時間為最短。

二. 旅行時間假設為，乘車時間加上轉乘站之等待時間。

三. 每個人對於轉乘次數與旅行時間之替換(trade-off)程度不相同。故當使用者輸入最多轉乘次數為 n 次時，使用者希望得到在 n 次轉乘次數內最少旅行時間的方案。

四. 假設兩路線到站時間差為 β 分鐘時能夠轉乘。

五. 使用者欲在所設定之時間窗限制內，搭乘最先出發的兩個班次。

3.2.2 演算法概述

由於本研究將導入動態即時資訊至大眾運輸行前旅次規劃，故並不能直接利用靜態的時刻表，尚須考慮當使用者查詢時，當下時間點之動態即時資訊，而動態即時資訊乃會隨著時間的移動而不相同，故在每次使用者查詢時，演算法將會根據轉乘次數上限，以及時間窗的限制，進行讀取動態即時資訊，以便提供準確的到站即時資訊。

本研究應用 Dijkstra's Algorithm 求解行前旅次規劃路徑。Dijkstra's Algorithm 於每次迭代(iteration)時，尋找標籤尚未被固定且成本最小的節點，以此點更新其標籤尚未被固定的下游節點。若下游點之成本標籤值比原成本小時，則將其成本標籤與其前置點更新，並將該節點標籤固定。

由於本研究之問題為，有轉乘次數限制之行前旅次規劃，除了需考量成本外，尚有轉乘次數限制須考量，故 Dijkstra 無法直接適用於本研究。因此，本研究應用多重標籤觀念，修正 Dijkstra 求解問題。

本研究提出修正後之 Dijkstra 為，考慮在轉乘次數下的最少旅行時間，故除了考量傳統 Dijkstra 之「累積成本 *Di*」與「前置點 *Pi*」兩標籤外，尚包含用來記

錄轉乘次數的標籤，其包含：「轉乘次數 Ti」的標籤、「前置班次 Pno_i」的標籤，成為多重標籤修正演算法，並運用於加強式網路。由於每個人對於轉乘次數與旅行時間的替換程度不相同，故本研究要求使用者自行設定的轉乘次數上限，且視上限內的轉乘次數為同質。但旅客發生轉乘時，心中仍存在著不安感，故本研究將在轉乘發生時給予懲罰值，懲罰值為兩倍的等待轉乘時間。如此一來，判斷是否能夠轉乘的門檻值 β 扮演著重要的角色，若 β 越高則發生轉乘的懲罰值也隨之增高，隨著 β 值的不同，規劃出來的方案也將會不相同。本研究之行前旅次規劃邏輯的流程如下圖：

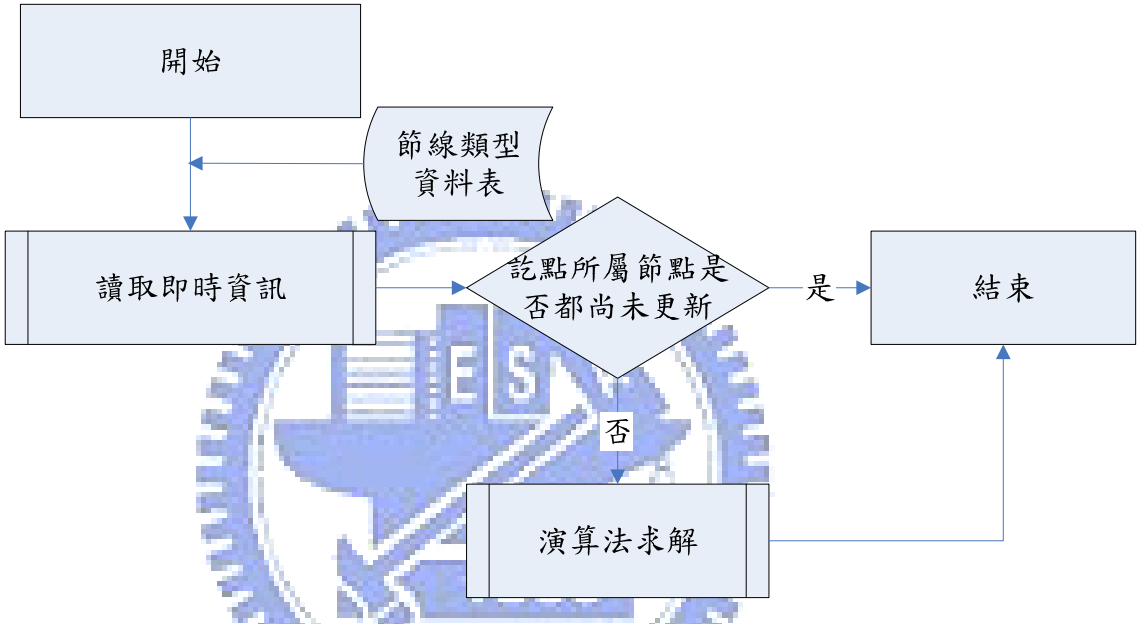

圖 3.3 總演算流程圖

3.2.3 開始

當使用者開始查詢行前旅次規劃時，首先讀入節線類型資料表，而後要求使用者自行提供以下四點限制

1.轉乘次數上限：在本次旅次中，使用者自行決定能夠容忍之轉乘次數上限。若使用者設定最多轉乘次數為一次，則系統將認定，該使用者認為轉乘一次與直達為同質性的旅次。

2.時間窗限制：此部份本研究讓使用者，自行輸入預計出發時間、預計到站時間或兩者皆輸入。當使用者僅輸入預計出發時間時，系統將時間窗限制視為預計出發時間至隔日的預計出發時間;倘若使用者設定預計到達時間，系統將時間窗限制視為目前系統時間至預計到達時間;若使用者輸入預計出發以及預計到達時間，系統將時間窗限制視為此時間範圍。

3.起點地標：使用者自行認定由該地標出發為最方便的地標。

4.迄點地標：使用者欲到達之目的地。

3.2.4 讀取即時資訊之步驟

當使用者限定轉乘次數以及時間窗限制時，演算法利用這兩個限制，進行讀取即時資訊，流程圖如圖 3.4 所示。

步驟 1.搜尋停靠站與路線資料表，可得屬於起點地標之節點集合，以及該路線編號。

步驟 2.更新由步驟 1 或步驟 4，所搜尋到之路線編號中，服務順序大於、等於所搜尋到的節點之節點到站資訊。da_1 與 da_2 分別為，符合時間窗限制內最快到達的第一個與第二個班次的到站時間。並記載計算次數為零。

步驟 3.(停止條件)判斷計算次數，是否等於使用者所限制的最多轉乘次數上限。若是則停止讀取即時資訊;若不是則進入步驟 4。

步驟 4.將每個已更新過到站資訊之節點重新定時，使得時間範圍為 da_1 至使用者所限定的預計最晚抵達時間，搜尋與該節點為轉乘節線相連的節點，集合中，最後將計算次數加一，回到步驟 2。

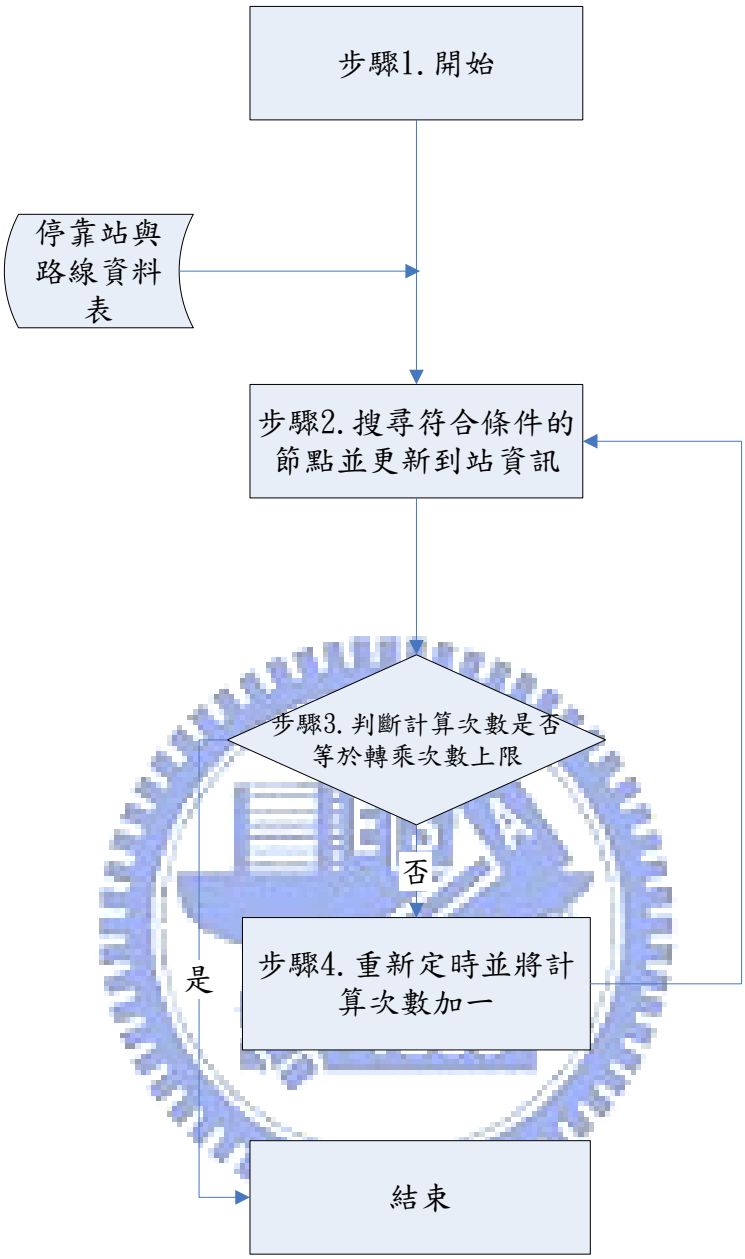

圖 3.4 即時讀取資料流程圖

3.2.5 判斷迄點地標所屬節點是否都尚未被更新

若都尚未被更新，則代表用者所設定之轉乘次數上限或時間窗限制，不能夠由起點地標到達迄點地標，故停止演算法。反之，則代表使用者所設定之轉乘次數上限或時間窗限制，可由起點地標到達迄點地標。

3.2.6 行前旅次規劃邏輯流程步驟

步驟 1. (多重標籤初始化)設定所有點的「第 i 個班次的前置點 P_i」為零、「第 i

個班次的前置班次點 Pno_i」為零、「第 i 個班次之累積成本 D_i」為無窮大、「第 i 個班次之累積轉乘次數 T_i」為無窮大、起點地標所屬節點的「第 i 個班次之累積成本 D_i」為零，起點地標所屬節點的「第 i 個班次之累積轉乘次數 T_i」為零。

步驟 2.從標籤尚未被固定的點中，找成本最小的點，設為 *minic*。並將 *minic* 的標籤固定。

步驟 3. (停止條件)若目前被標記的點*(minic)*屬於迄點地標所屬節點，則停止。反之進行至步驟 4。

步驟 4. 判斷節點 *minic* 是否有與其相鄰(*minic* 之下游點)，且尚未嘗試經過的點。若有，則設該點為 *x*，到步驟 5；若無，到步驟 3。

步驟 5. 判斷與 *minic* 相連的下游點 *x* 的連結狀況，若兩點屬於旅行節線相連則進入步驟 6;若為轉乘節線相連，到步驟 7。

步驟 6. 若 *minic* 與其下游點為旅行節線相連時，進行 $D_i(x)= D_i(minic)$，$T_i(x)= T_i(minic)$，$P_i=minic$，$Pno_i=i$。回到步驟 4。

步驟 7. 若 *minic* 與其下游點為轉乘節線相連時，判斷 minic 是否屬於起點地標之節點。若是，則回到步驟 4 若不是，到步驟 8。

步驟 8. 判斷兩班次時間差是否大於轉乘門檻 β $(da_j(x)-da_i(minic)>=\beta)$。若是，則回到步驟 4 若不是，到步驟 9。

步驟 9.判斷出發班次的轉乘次數加一是否超過使用者所限定之轉乘次數上限 U，$(T_i(minic)+1>=U)$。是，則回到步驟 4 若不是，到步驟 10。

步驟 10.判斷目標節點班次 *j* 的累積成本是否小於最小成本點之出發班次累積成本加上兩倍的到站時間差，$(D_j(x)<=D_i(minc)+2*\ da_j(x)-da_i(minic))$。若是，則回到步驟 11。若不是，到步驟 4。

步驟 11.進行標籤更新：$D_j(x)= D_i(minic) +2*\ da_j(x)-da_i(minic)$，$T_j(x)= T_i(minic)+1$，$P_j=minic$，$Pno_j=i$。

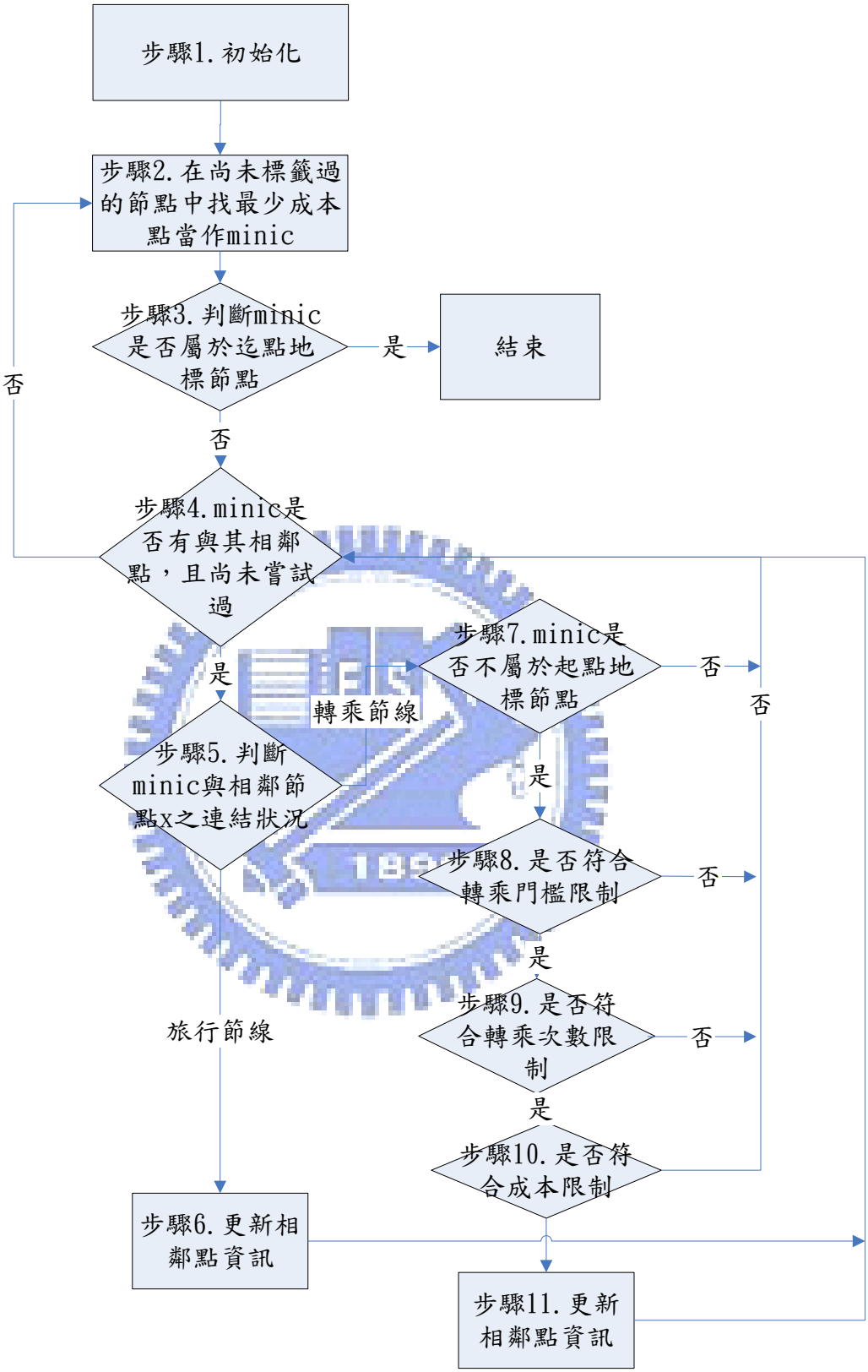

圖 3.5 行前旅次規劃流程圖

3.3 範例說明

本研究利用三條路線的去程進行本演算法流程的說明，本範例的網路如圖，且假設，此三條路線若發生停靠相同站牌名稱時，共用同一站牌，且每個站牌相距超過 500 公尺。而以 Location1 為中心，方圓 500 公尺內有 Stop1 之站牌。而以 Location2 為中心，方圓 500 公尺範圍內有 Stop5 的站牌。

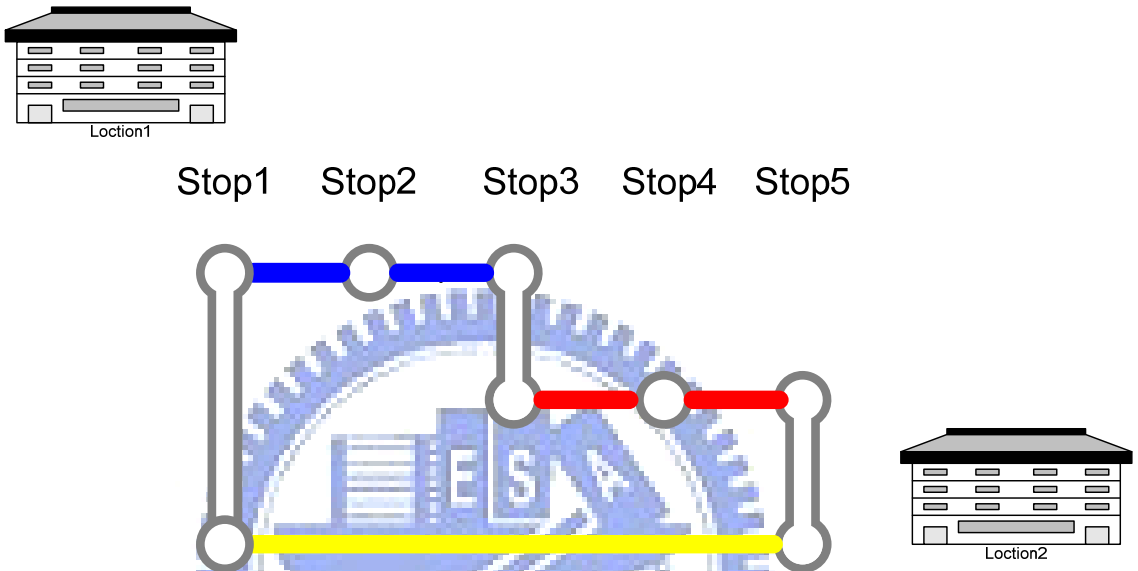

圖 3.6 範例示意圖

加強式路網示意圖如圖 3.7 到站資訊如表 3.3。

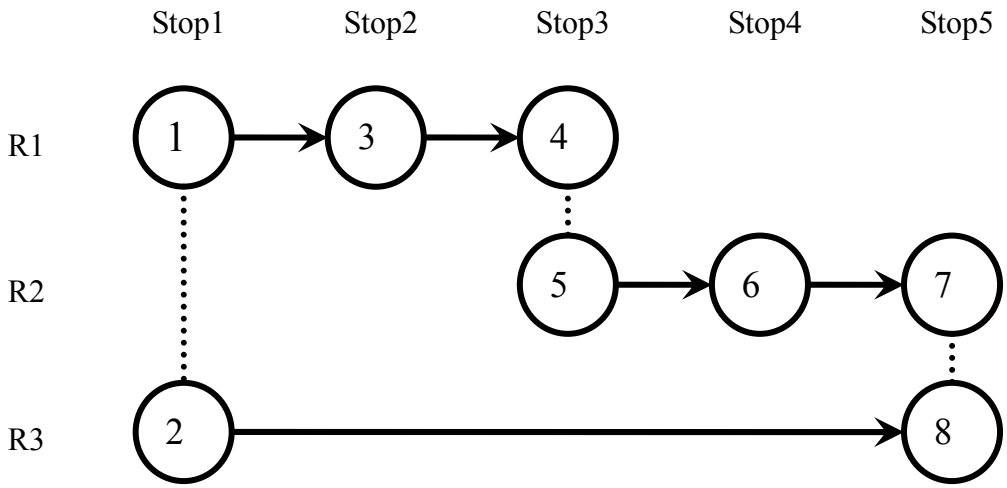

圖 3.7 範例加強式網路示意圖

表 3.3 範例資料表

RouteID	RouteName	Name	StopID	Sql	NodeID	Latitude	Longitude	班次	到站時間
1	1(往)	Stop1	1	1	1	X1	Y1	1	9:00
1	1(往)	Stop2	2	2	3	X2	Y2	1	9:08
1	1(往)	Stop3	3	3	4	X3	Y3	1	9:17
2	2(往)	Stop3	3	1	5	X3	Y3	1	9:02
2	2(往)	Stop4	4	2	6	X4	Y4	1	9:09
2	2(往)	Stop5	5	3	7	X5	Y5	1	9:16
3	3(往)	Stop1	1	1	2	X1	Y1	1	9:00
3	3(往)	Stop5	5	2	8	X5	Y5	1	9:50
1	1(往)	Stop1	1	1	1	X1	Y1	2	9:10
1	1(往)	Stop2	2	2	3	X2	Y2	2	9:17
1	1(往)	Stop3	3	3	4	X3	Y3	2	9:27
2	2(往)	Stop3	3	1	5	X2	Y3	2	9:22
2	2(往)	Stop4	4	2	6	X4	Y4	2	9:30
2	2(往)	Stop5	5	3	7	X5	Y5	2	9:40
3	3(往)	Stop1	1	1	2	X1	Y1	2	9:20
3	3(往)	Stop5	5	2	8	X5	Y5	2	10:05
1	1(往)	Stop1	1	1	1	X1	Y1	3	9:20
1	1(往)	Stop2	2	2	3	X2	Y2	3	9:27
1	1(往)	Stop3	3	3	4	X3	Y3	3	9:37
2	2(往)	Stop3	3	1	5	X2	Y3	3	9:30
2	2(往)	Stop4	4	2	6	X4	Y4	3	9:37
2	2(往)	Stop5	5	3	7	X5	Y5	3	9:49
3	3(往)	Stop1	1	1	2	X1	Y1	3	9:40
3	3(往)	Stop5	5	2	8	X5	Y5	3	10:25

3.3.1 開始

使用者所設定的條件分別如下：

1. 使用者設定 L1 為起點地標。
2. 使用者設定 L2 為迄點地標。
3. 假設目前時間為早上九點整，而使用者僅輸入預計到達時間為早上十點半。
4. 使用者設定最多轉乘次數為零次。

3.3.2 讀取即時資料

由使用者的設定可以得知，通過起點地標的路線分別為：路線編號 1(R1)與路線編號 2(R2)，而兩路線在範圍內的節點分別為：節點編號 1 與節點編號 2。而使用者所設定的時間窗範圍，是早上九點至早上十點半，故在讀取即時資料的第二步驟，將更新路線編號 1(R1)中的節點編號 1 至節點編號 4 的到站時間(da_1與da_2)，以及路線編號 3(R3)中的節點編號 2 與節點編號 8 之節點，到站資訊為，九點至十點半中最快到達的兩個班次之到站時間。查詢停靠站與路線資料庫得知，路線編號 1，九點與九點十分將有兩個班次經過 Stop1，故節點編號 1、節點編號 3 與節點編號 4 的到站資訊將更新如表所示，路線編號 3 的站牌亦為如此。故圖中的中括號為符合時間窗的第一班次與第二班次，$[da_1, da_2]$

表 3.4 範例說明一

節點編號	第一個班次	第二個班次
1	9:00	9:10
3	9:08	9:17
4	9:17	9:27
2	9:00	9:20
8	9:50	10:05

此時計算次數為零次等於使用者所限定之轉乘次數上限，故停止讀取即時資料將進入下一步驟。

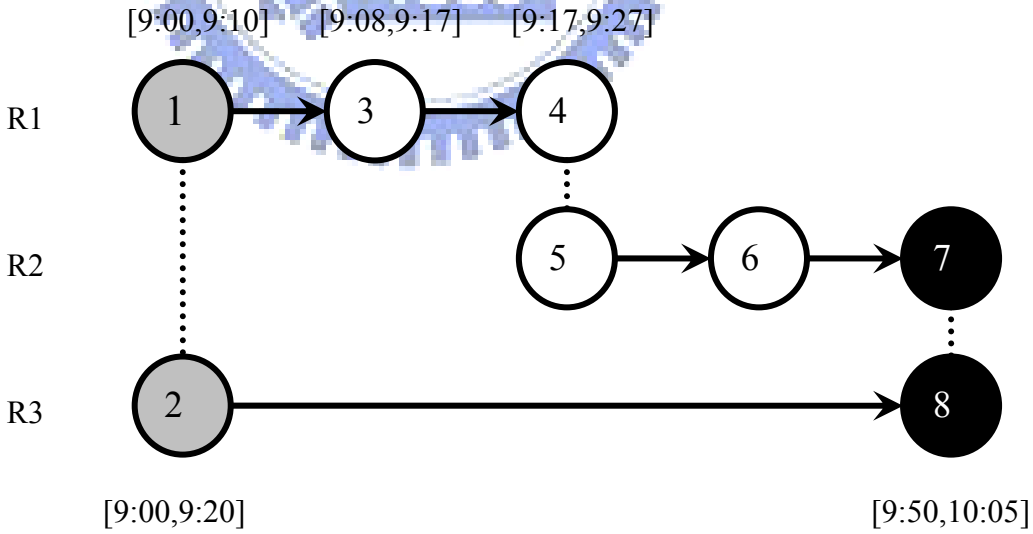

圖 3.8 範例一加強式網路說明一

3.3.3 演算法求解

演算法求解之第一步驟為初始化，而初始化資訊如圖所示。圖中小括弧中的數字代表到達該節點第一班次與第二班次的累積成本(D_1, D_2)，以斜體字加底線的數字則為到達該節點第一班次與第二班次之累積轉乘次數(T_1, T_2)，大括號中的數字代表第一與第二個班次的前置點(Pno_1, Pno_2)，第二步驟則選取具有最少成本之節點作為目前最小成本點($minic$)，並在圖中給予*的記號。而在圖中顯示灰色的節點代表為屬於起點地標的節點，顯示黑色的節點代表屬於迄地標的節點。

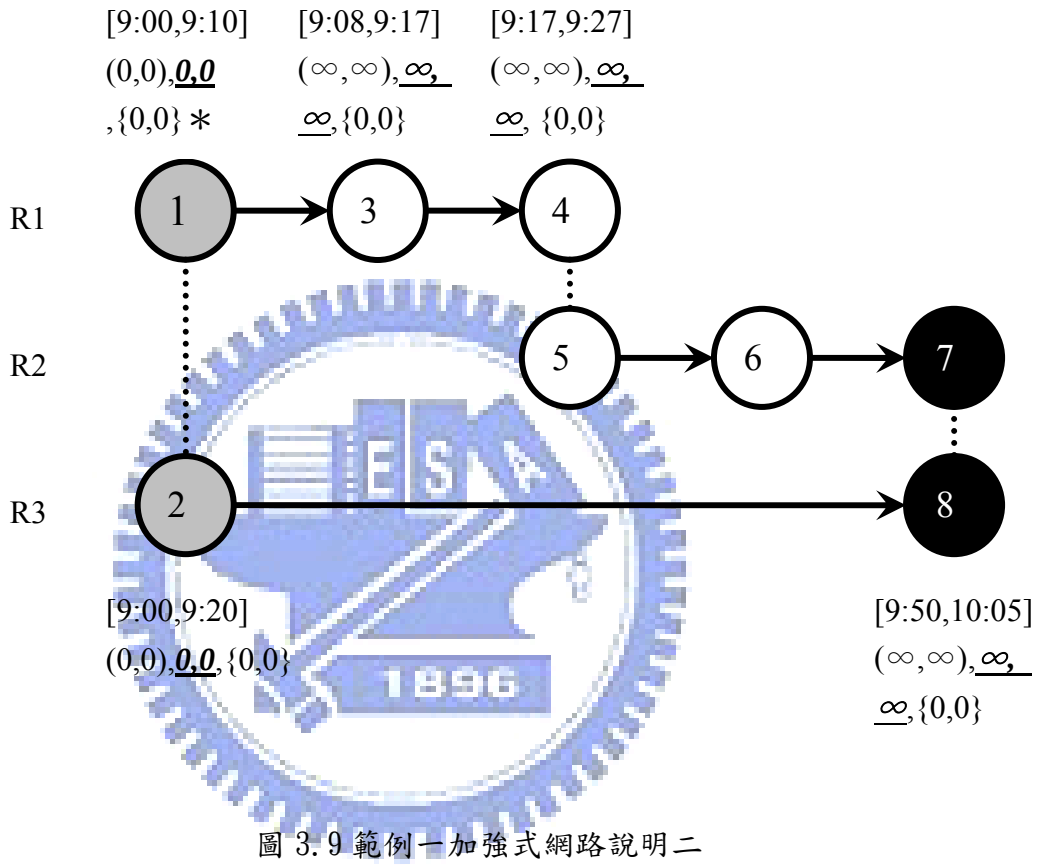

圖 3.9 範例一加強式網路說明二

在第一個 iteration 中，節點 1 分別和節點 2 與節點 3 相連。而對於節點 3 與節點 1 之連結狀況為旅行節線，故直接更新成本轉乘次數，如圖所示。但節點 1 和節點 2 為連結狀況為轉乘節線，因此，必須經過四個判斷條件，但由於節點 2 屬於起點地標之節點，故不更新成本與轉乘次數，如圖所示。此時節點 1 已無任何相連節點，故進行選取目前最少成本點，而目前最小成本點，為節點編號 2 之節點，故在此給予節點編號 2 一個*符號。

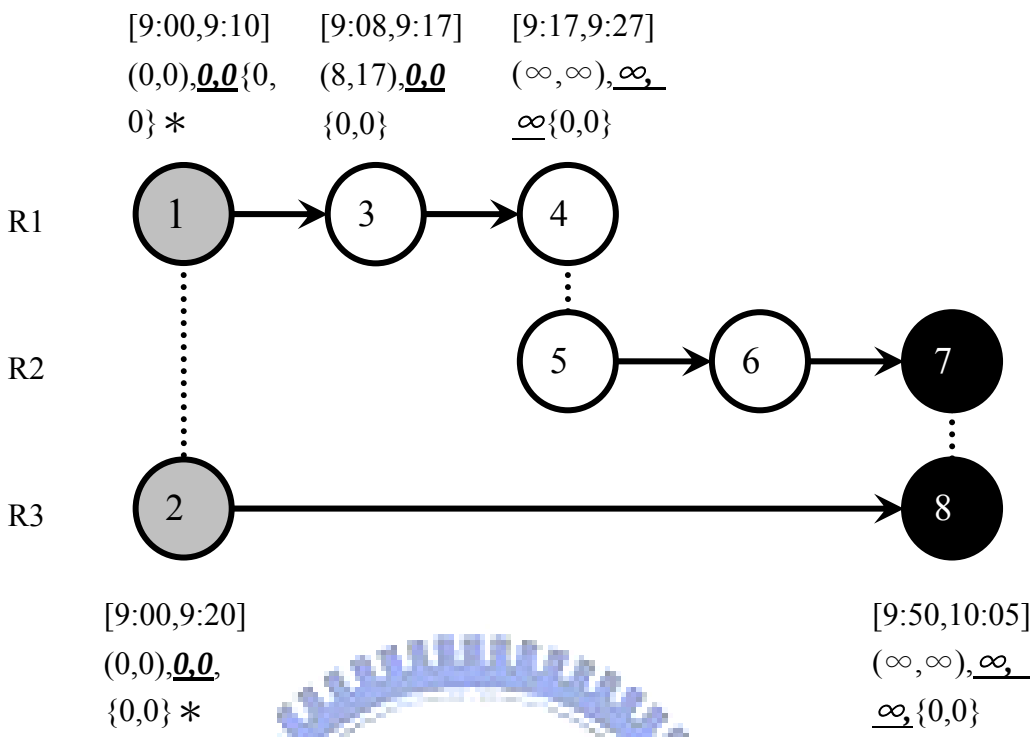

圖 3.10 範例一加強式網路說明三

當本範例進行至第四個 iteration 時，節點資訊如圖，且目前最少成本點為節點編號 8 之節點，且此節點屬於迄點地標之節點，故演算法停止。

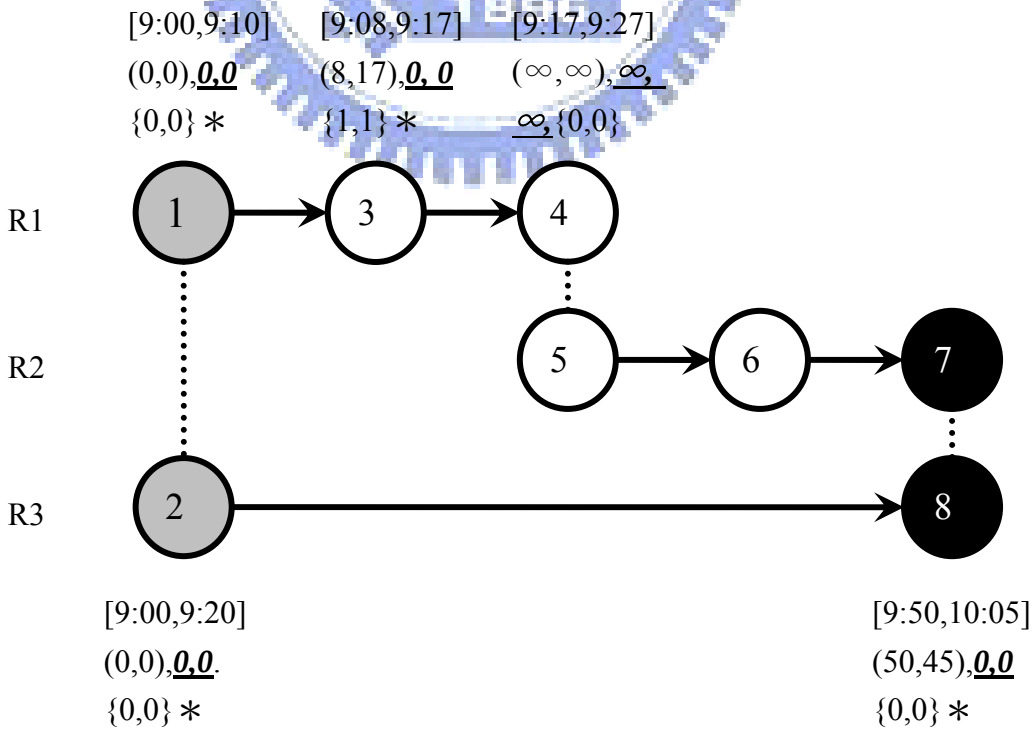

圖 3.11 範例一加強式網路說明四

3.3.3 範例說明二

如今使用者若設定轉乘次數上限最多為一次時，本範例將會在讀取即時資料的步驟時，進行每個節點重新定時，例如，將節點 4 重新定時，故時間窗限制將被更新為早上九點十七分至早上十點三十分，進行搜尋符合時間窗限制之班次，並更新與節點 4 為轉乘節線相連之節點的到站時間資訊，用此方式解決大眾運輸轉乘時間相依的現象。資訊如表，而加強式網路資訊為圖所示：

表 3.5 範例說明表二

節點編號	第一個班次	第二個班次
1	9:00	9:10
3	9:08	9:17
4	9:17	9:27
2	9:00	9:20
8	9:50	10:05
5	9:22	9:30
6	9:30	9:37
7	9:40	9:49

當計算到第七個 iteration 時，節點 7 將被標記住，然而節點 7 屬於迄點地標之節點，故演算法停止。此時，所規劃的方案為：在 Stop1 搭乘 R1 列車至 Stop3 轉 R2 列車至迄點地標。

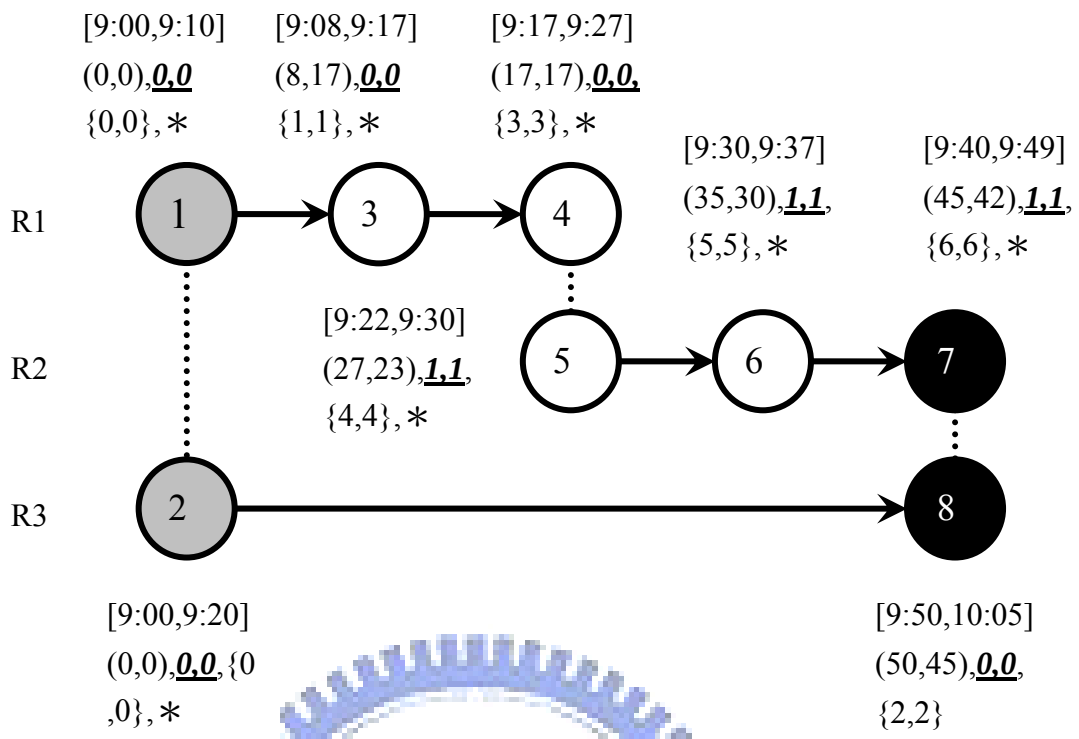

圖 3.12 範例一加強式網路說明五

當使用者設定最多轉乘次數為一次時，演算法將會規劃出 Stop1 搭乘 R1 列車至 Stop3 轉 R2 列車至迄點地標之方案，花費時間為 39 分鐘。此時觀察當使用者設定為零次時，演算法會提供在 Stop1 搭乘 R3 直達至 Stop5 且花費時間為 45 分鐘，故可以在得知此演算法可以提供在轉乘次數內最少旅行時間之方案。

3.3.3 轉乘站之選擇

利用本演算法進行行前旅次規劃時，會規劃出使用者旅行成本最少的方案，但若遇到在任何一個站，進行轉乘成本都相同的現象，本演算法將指派使用者，在兩路線發生轉乘時所花費時間最少的轉乘站進行。在此利用另一個範例說明，此範例之路線示意圖如圖所示。若使用者欲從起點 O 至迄點 D。而路線 A 服務順序為，O、A、B、C、E，路線 B 之服務順序為，E、C、B、A、D。此時將有四個轉乘站可進行轉乘。

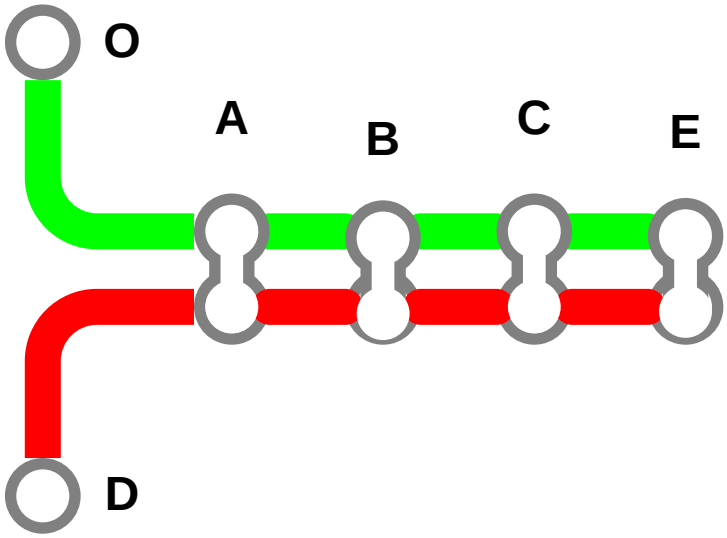

圖 3.13 範例二網路示意圖

若將此兩路線顯示為本研究之加強式網路，並假設路線 A 的前兩個班次，分別為九點以及九點半，而路線 B 的前兩個班次，分別為九點五十分以及十點二十分，且兩條路線的中任兩個站牌，皆相距十分鐘的車程。此範例之加強式網路以及到站資訊如圖 3.14。

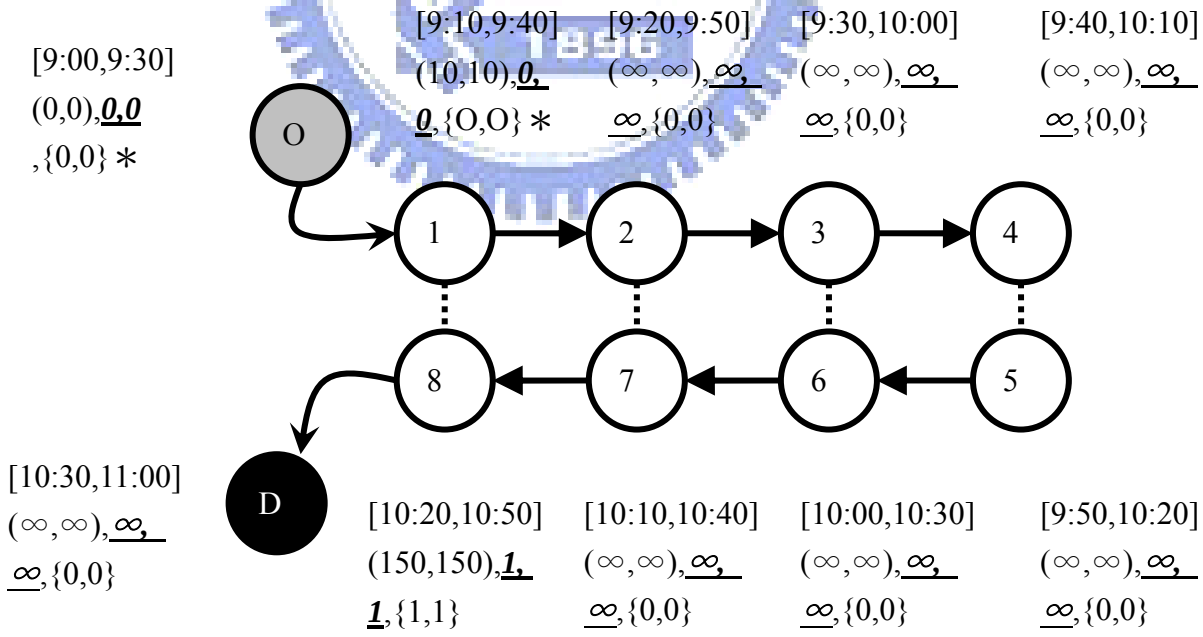

圖 3.14 範例二網路示意圖

當本範例進行至第二個 iteration 時，節點 1 被標記為最小成本點，此時將更

新節點 8 之成本以及轉乘次數為圖 3.14 中所顯示之數字。當進行至第五個 iteration 時，節點 5 會依據節點 4 的資訊，進行更新成本以及轉乘次數如下圖 3.15。

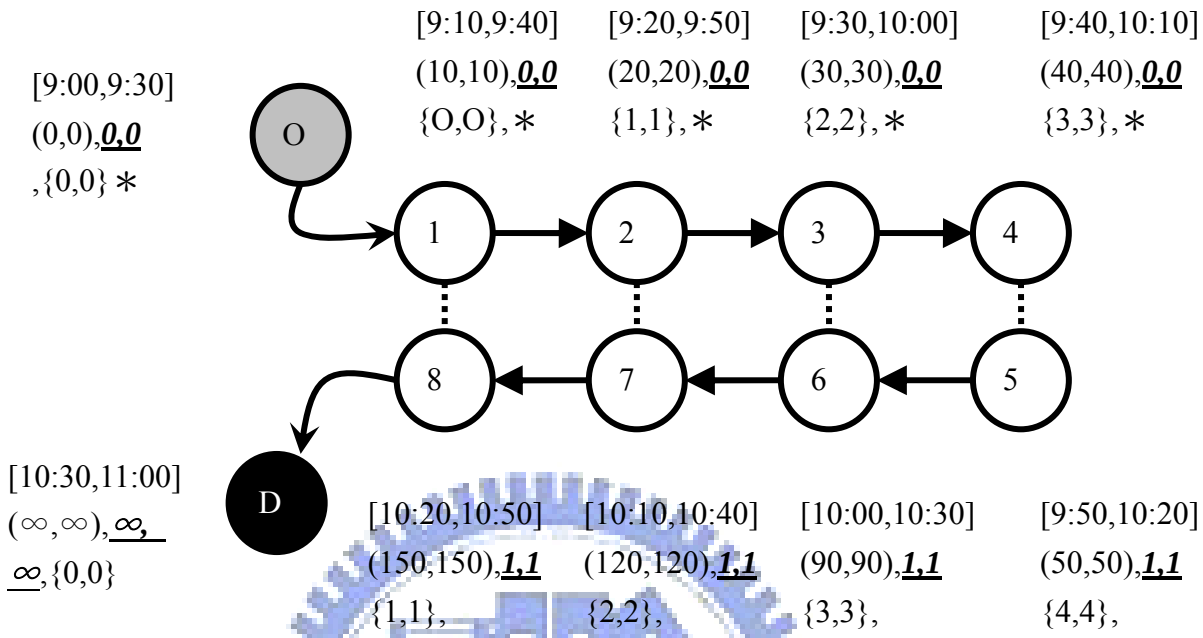

圖 3. 15 範例二說明一

且當本範例進入第六個 iteration，節點 5 將被標記為最小成本點，進行節點 6 是否更新的判斷，但因為由節點 5 出發至節點 6 之累積成本(50+10=60)並小於節點 6 原本的成本(90)，故會進行更新節點 6 資訊的資訊。而節點 6 至節點 7、與節點 7 至節點 8，亦會因累積成本小於目前成本，而更新節點資訊，如下圖 3.16。

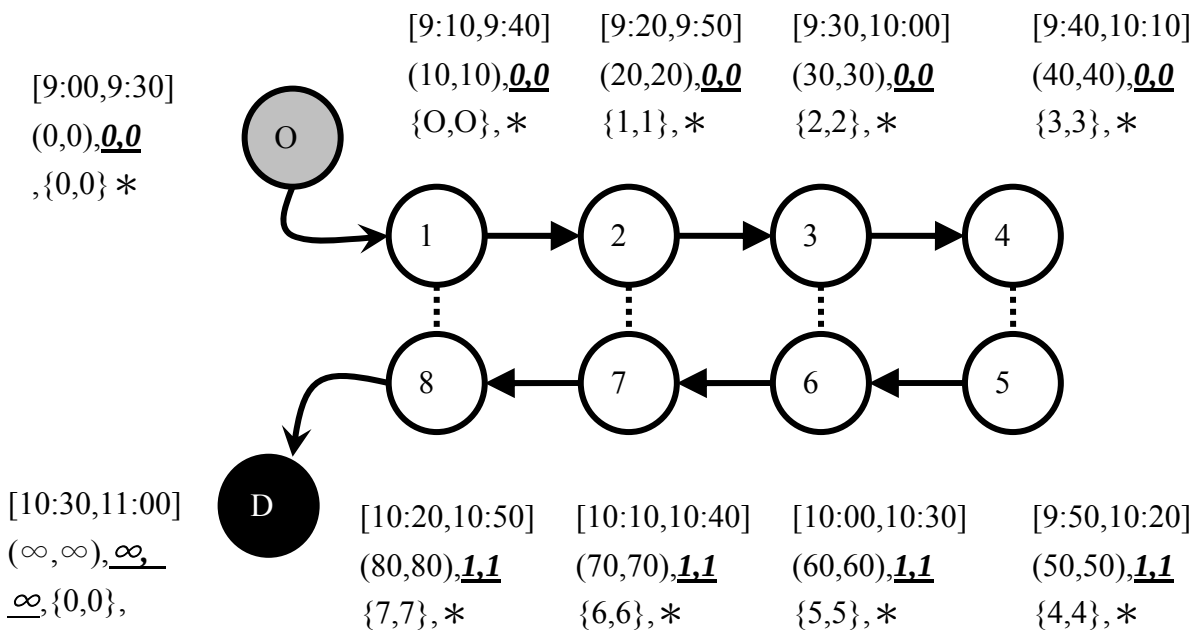

圖 3.16 範例二說明二

在第九個 iteration 時，節點 8 被選作為最小成本點，並進行節點 D 的累積成本(80+10=90)、轉乘次數以及前置點的更新如圖所示。當進入第十個 iteration，節點上的資訊如圖 3.17 所示。此時節點 D 將被標記為最少成本點，故演算法停止。此時節點 D 的前置點為節點 8。依照前置點可知整條路徑為節點 O、節點 1、節點 2、節點 3、節點 4、節點 5、節點 6、節點 7、節點 8、節點 D，而這條路徑在路線中將會顯示為由站牌 O 搭乘路線 A 至 E 站轉乘路線 B 至站牌 D。

由此得知，本研究所開發出的演算法，會找出成本最少的路徑。倘若遇到在所有站牌轉乘之旅行時間都相同時，將會因為基本假設：使用者追求旅行時間最少的方案，以及演算法在轉乘時，給予兩倍的等待時間的懲罰值，導致規劃出在轉乘站等待時間最少的轉乘站進行轉乘之方案。

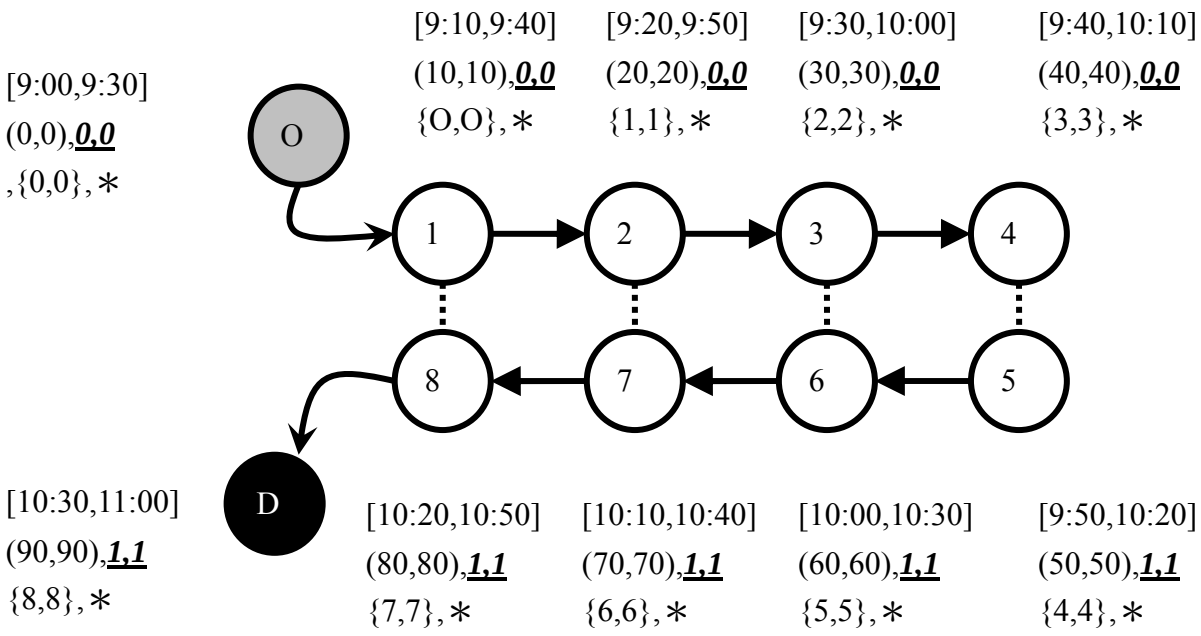
[9:10,9:40]
(10,10),0,0
{O,O},＊
[9:20,9:50]
(20,20),0,0
{1,1},＊
[9:30,10:00]
(30,30),0,0
{2,2},＊
[9:40,10:10]
(40,40),0,0
{3,3},＊
[9:00,9:30]
(0,0),0,0
,{0,0},＊
O
1
2
3
4
8
7
6
5
D
[10:30,11:00]
(90,90),1,1
{8,8},＊
[10:20,10:50]
(80,80),1,1
{7,7},＊
[10:10,10:40]
(70,70),1,1
{6,6},＊
[10:00,10:30]
(60,60),1,1
{5,5},＊
[9:50,10:20]
(50,50),1,1
{4,4},＊

圖 3.17 範例二說明三

第四章 實例驗證

鑑於現實生活中的大眾運輸網路，大部分皆可在兩次轉乘內即可到達。為了驗證準確度以及可行性，本研究首先利用虛擬網路，以及假設的即時資訊與時刻表，設計出一些範例驗證，是否能夠符合所想的理念。4.1 節說明虛擬網路的測試以及效能分析;4.2 節利用新竹市區公車進行測試以及對轉乘門檻 β 值進行敏感度的分析。

4.1 演算法邏輯測試

為驗證本研究之考慮動態資訊之轉乘邏輯正確性，研究者用虛擬網路圖 4.1 作為邏輯測試範例。驗證目的分為以下幾點：

1. 是否能夠正確的規劃出，使用者輸入最多轉乘次數內找到最少的旅行時間。
2. 是否能夠解決大眾運輸單邊設站之特殊性。
3. 是否能夠利用兩班次之間的銜接組合，提供使用者稍晚出發，使得旅行時間減少之方案。
4. 是否能夠在不同的時間限制下提供不同的方案。
5. 是否能夠提供多一次轉乘次數，但在總旅行時間中減少了轉乘時所花費的等待時間的方案。
6. 是否能夠突破現有搜尋法的兩次轉乘次數限制，提供多次轉乘之方案。

本研究將轉乘門檻 β 設定為 3 分鐘，而此虛擬網路一共有 20 條路線，平均一條路線停靠 8 個站牌，且每個站牌皆與其餘站牌相距超過 500 公尺，而其路線停靠站如表 4.1 與表 4.2 所示。圖 4.1 中方格的數字代表著，每個地標的編號，每個地標(Location)範圍內具有與其相同編號的站牌。本研究自行假設，各班次到達各站點之時刻表與動態資訊。且假設目前時間為早上 6 點，利用早上 6 點以後，每條路線各 4 個班次，進行上述之驗證。硬體環境為 AMD Athlon(tm)64 X2 Dual Core Processor 4800+ 2.41GHz，3.25GB 的 RAM。

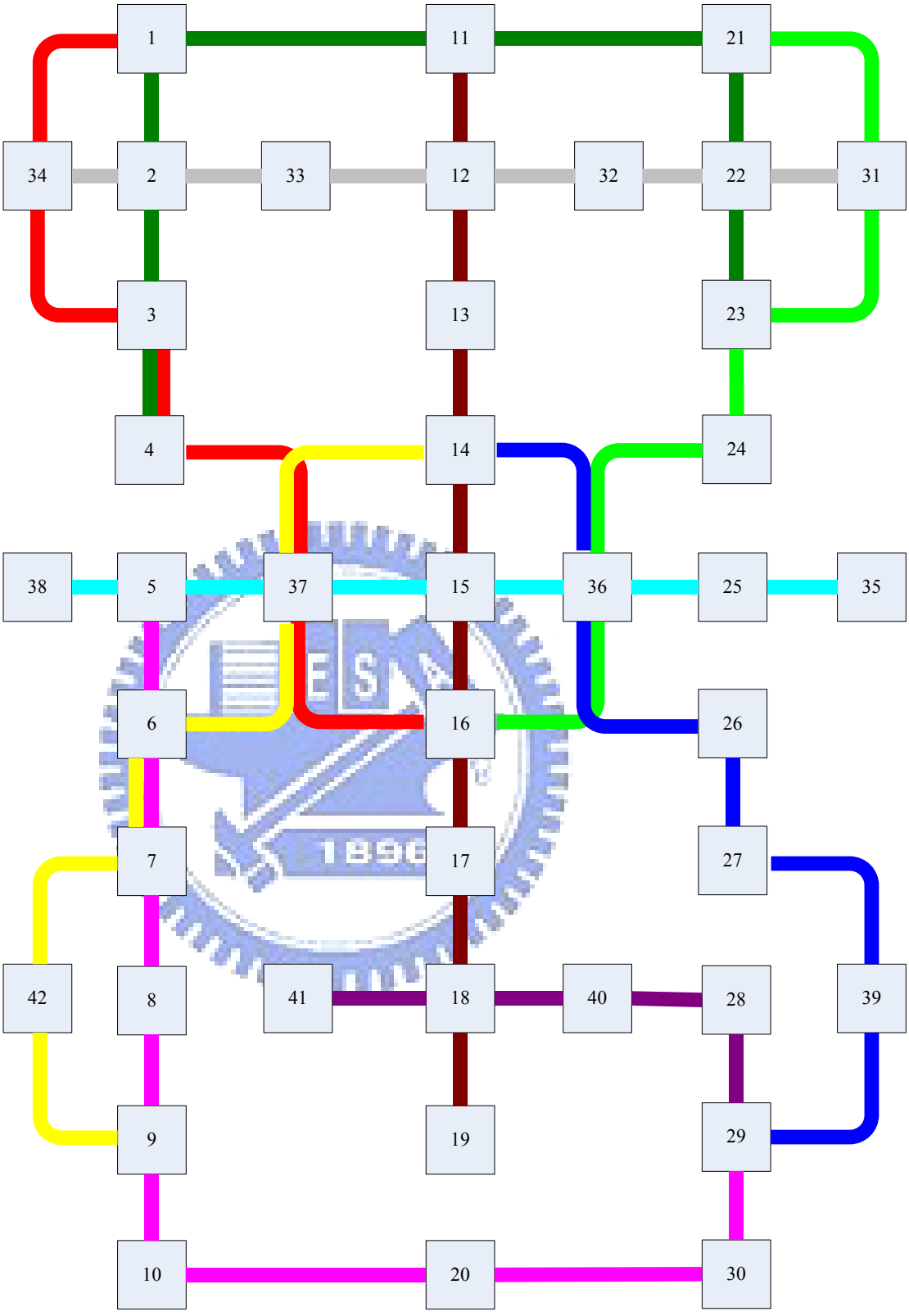
1
11
21
34
2
33
12
32
22
31
3
13
23
4
14
24
38
5
37
15
36
25
35
6
16
26
7
17
27
42
8
41
18
40
28
39
9
19
29
10
20
30

圖 4.1 虛擬網路示意圖

表 4.1 路線去程停靠站牌一覽表

路線編號	路線名稱	停靠站牌編號(節點編號)--依照服務順序排序
1	1（去）	1,34,3,4,37,16
2	2（去）	23,22,21,11,1,2,3,4
3	3（去）	34,2,33,12,32,22,31
4	4（去）	21,31,23,24,36,16
5	5（去）	11,12,13,14,15,16,17,18,19
6	6（去）	38,5,37,15,36,25,35
7	7（去）	5,6,7,8,9,10,20,30,29
8	8（去）	14,37,6,7,42,9
9	9（去）	14,36,26,27,39,29
10	10（去）	29,28,40,18,41

表 4.2 路線回程停靠站牌一覽表

路線編號	路線名稱	停靠站一覽表(節點編號)--依照服務順序排序
11	1（返）	16,37,4,3,34,1
12	2（返）	4,3,1,11,21,23
13	3（返）	31,22,32,12,33,2,34
14	4（返）	16,36,24,23,31,21
15	5（返）	19,18,17,16,15,14,13,12,11
16	6（返）	35,25,36,15,37,5,38
17	7（返）	29,30,20,10,9,8,7,6,5
18	8（返）	9,42,7,6,37,14
19	9（返）	29,39,27,26,36,14
20	10（返）	41,18,40,28,29

4.1.1 最少旅行時間邏輯確認

表 4.5、表 4.8 以轉乘次數上限作為分類，仔細測試上述之驗證。轉乘次數分別為零次到三次，主要測試使用者變更起迄站、轉乘次數與時間窗限制後，是否能夠提供旅行時間最少之旅次規劃。而本研究試驗轉乘次數上限時，將進行不同的時間窗限制，一來，測試時間窗之限制是否能提供不相同之方案，再則是否能在時間窗限制與轉乘次數限制內找到正確之方案。

表 4.4 轉乘次數上限為零次

編號	起點站	迄點站	時間窗	方案
1	Location23	Location1	06:00 出發	在 Stop23 搭乘 06:07:00 到站的 2(去) 路線到達 Stop1,約 06:27:00 到達。
2	Location23	Location1	07:00 出發	在 Stop23 搭乘 07:00:00 到站的 2(去) 路線到達 Stop1,約 07:20:00 到達
3	Location23	Location1	07:00 到達	在 Stop23 搭乘 06:07:00 到站的 2(去) 路線到達 Stop1,約 06:27:00 到達
4	Location35	Location38	06:00 出發	在 Stop35 搭乘 06:15:00 到站的 6(返) 路線到達 Stop38,約 06:35:00 到達
5	Location9	Location37	07:40 到達	在 Stop9 搭乘 06:54:00 到站的 8(返) 路線到達 Stop37,約 07:40:00 到達

表 4.5 為轉乘次數上限為一次，且轉乘次數為一次之隨機測試結果。在此，除了驗證上限為一次之正確性，接著測試本研究所提供之方案中，是否能夠因為使用者將轉乘次數多設定一次，而提供多一次轉乘，但減少等待轉乘時間之旅行時間方案。表 4.4 之測試編號 6 與表 4.3 之測試編號 7，可得知 Location9 到 Location37 搭乘 8 號車返程路線，無須轉乘即可到達，而旅行時間為 46 分鐘。但當使用者輸入轉乘次數上限為一次時，系統將提供使用者在 Location9 搭乘 7 號車返程路線到達 Location5，轉 6 號車去乘路線到達 Location37，旅行時間為 28 分鐘。

表 4.5 轉乘次數上限為一次

編號	起點站	迄點站	時間窗	方案
1	Location 1	Location 17	06:30 出發	在 Location 1 搭乘 06:40:00 到站的 2(返)路線到達 Stop11，轉 5(去)路線至 Location 17，約 07:50:00 抵達
2	Location 1	Location 17	07:00 出發	在 Location 1 搭乘 07:00:00 到站的 2(返)路線到達 Stop11，轉 5(去)路線至 Location 17，約 08:20:00 抵達
2	Location 3	Location 12	07:40 到達	在 Location 3 搭乘 06:35:00 到站的 2(返)路線到達 Stop11，轉 5(去)路線至 Location 12，約 07:00:00 抵達
3	Location 41	Location 16	08:00 到達	在 Location 41 搭乘 06:50:00 到站的 10(返)路線到達 Stop18，轉 5(返)路線至 Location 16，約 07:20:00 抵達
4	Location 21	Location 5	06:20 出發	在 Location 21 搭乘 06:40:00 到站的 4(去)路線到達 Stop36，轉 6(返)路線至 Location 5，約 07:21:00 抵達

5	Location 9	Location 37	07:40 到達	在 Location 9 搭乘 06:50:00 到站的 7(返)路線到達 Stop5，轉 6(去)路線至 Location 37，約 07:18:00 抵達
6	Location 10	Location 17	06:00 出發	在 Location 10 搭乘 06:45:00 到站的 7(返)路線到達 Stop9，轉 8(返)路線至 Location 17，約 07:50:00 抵達。
7	Location 10	Location 17	07:40 到達	在 Location 10 搭乘 06:17:00 到站的 7(去)路線到達 Stop29，轉 9(返)路線至 Location 17，約 07:20:00 抵達。

表 4.6 轉乘次數上限為二次

編號	起點站	迄點站	時間窗	方案
1	Location21	Location6	06:00 出發	在 Location 21 搭乘 06:40:00 到站的 4(去)路線到達 Stop36，轉 6(返)路線至 Stop5，再轉 7(去)路線至 Location 6 約 08:00:00 抵達。
2	Location1	Location29	08:00 到達	在 Location 1 搭乘 06:00:00 到站的 1(去)路線到達 Stop16，轉 4(返)路線至

				Stop36，再轉 9(去)路線至 Location29 約 07:30:00 抵達。
3	Location1	Location8	07:00 出發	07:20:00 在 Location 1 搭乘 1(去)路線到達 Stop37，轉 8(去)路線至 Stop9，再轉 7(返)路線，再轉 Location 8 約 09:10:00 抵達。

由於本範例之 2(返)路線具有單邊設站之現象，故使得由 Location22 至 Location8 必須要三次轉乘方能到達，而本研究使用之演算法，亦可求得大於兩次轉乘之方案。表 4.7 即為運算結果。

表 4.7 轉乘次數上限為三次

編號	起點站	迄點站	時間窗	方案
1	Location 22	Location 8	06:00 出發	06:12:00 在 Location 22 搭乘 2(去)路線到達 Stop3，轉 1(去)路線至 Stop37，再轉 6(返)路線抵達 Stop5，最後轉 7(去)路線到達 Location 8。約 07:55:00 抵達
2	Location 22	Location 10	06:00 出發	06:12:00 在 Location 22 搭乘 2(去)路線到達 Stop3，轉 1(去)路線至 Stop37，再轉 6(返)路線抵達 Stop5，最後轉

				7(去)路線到達 Location 10。約 08:05:00 抵達

4.1.2 測試是否能夠解決單邊設站之問題

本測試用意在於，檢驗本研究之網路以及演算法，能否解決大眾運輸網路之單邊設站之特殊性。表 4.1 與表 4.2 中，路線編號 2 與路線編號 12，雖然同為 2 號路線，但其往返程所停靠之站牌不一樣，故在表 4.8 以 Location1 與 Location2 作為測試範例，驗證本研究，是否能夠將區分因往返程停靠路線不相同，進而提供不相同之方案。而結果顯示出，當起點站為 Location1，迄點站為 Location2，設定轉乘次數為零次時，可搭乘 2 號去程路線到達 Location2。但當起迄點互相交換時，設定直達時，系統將無法提供方案。使用者必須將轉乘次數設定為一次，系統才能提供出正確之方案。故此驗證，本研究可解決大眾運輸單邊設站之特殊性。

表 4.8 單邊設站測試結果

編號	起點站	迄點站	時間窗	方案
1	Location1	Location2	06:20 出發	在 Stop1 搭乘 06:27:00 到站的 2(去) 路線到達 Stop2,約 06:32:00 到達
2	Location1	Location2	06:20 出發	在 Stop1 搭乘 06:27:00 到站的 2(去) 路線到達 Stop2,約 06:32:00 到達
3	Location2	Location1	06:20 出發	請放寬轉乘次數或時間窗限制
4	Location2	Location1	06:20 出發	在 Stop2 搭乘 06:32:00 到站的 2(去)路線到達 Stop3，轉 1(返)路線至 Stop1，約 07:00:00 抵達

4.1.3 檢驗是否能利用兩班次的銜接使得旅行時間減少

當系統規劃出轉乘方案時，本研究追求，在轉乘次數內旅行時間最少者，旅行時間包括，乘車時間加上轉乘等待時間。而本研究將利用班次銜接，減少相同路線組合中，使用者在轉乘站等待之時間。而此部份，利用表 4.5 中測試編號 1 之範例檢驗，檢驗本研究所提供出之方案，是否為在轉乘站等待時間最少之班次組合。表 4.9 顯示出，本研究所提供之方案在轉乘站所等待的時間為 20 分鐘，而其餘之班次組合的等待時間，皆大於本研究所提供之方案，因而驗證，本研究所提供之方案，乃為所考慮之班次中，轉乘花費時間為最少者。

表 4.9 是否能利用兩班次的銜接將在轉乘站等待時間減少

起點站	迄點站	時間窗	班次組合	轉乘站等待時間	方案
Location1	Location17	08:00 到達	2-2	20	在 Location 1 搭乘 06:30:00 到站的 1(去)路線到達 Stop16，轉 5(去)路線至 Location 17，約 07:50:00 抵達
Location1	Location17	08:00 到達	1-2	35	
Location1	Location17	08:00 到達	1-1	25	

4.1.4 測試是否可因為時間窗限制而提供不相同之方案

測試目的在於驗證本研究所提出之演算法，是否能夠可解決，以轉乘次數最少為首要目標式，其演算法所無法考量的問題：在時間窗的限制下，因為，以轉乘次數最少為原則，而忽略了其餘轉乘方案的現象。也就是，倘若直達方案因為時間窗的限制，而無法提供方案時，將提供使用者無方案可行，而無法進一步告知使用者，一次轉乘存在可行方案。本研究解決了此現象，測試結果顯示表 4.10

表 4.10 測試時間窗限制

起點站	迄點站	時間窗	方案
Location37	Location6	07:00 到達	06:31:00 在 Location 37 搭乘 6(返)路線到達 Stop5，轉 7(去)路線抵達 Location 6。約 06:45:00 抵達
Location37	Location6	07:30 到達	07:10:00 在 Location 37 搭乘 8(去)路線直達 Location 6,約 07:20:00 到達

4.2 新竹市區公車測試

本研究以實際之都市交通路網進行測試實驗，希望了解此演算法在大型路網下之應用與效率。此路網為新竹市區公車路線，其中包括 13 條路線、284 個停靠站，本研究僅考慮早上 9 點至下午兩點中的 50 個班次。如圖 4.2 所示。新竹市區公車交通路網，在本研究所利用之加強式網路中，共有 653 個節點。

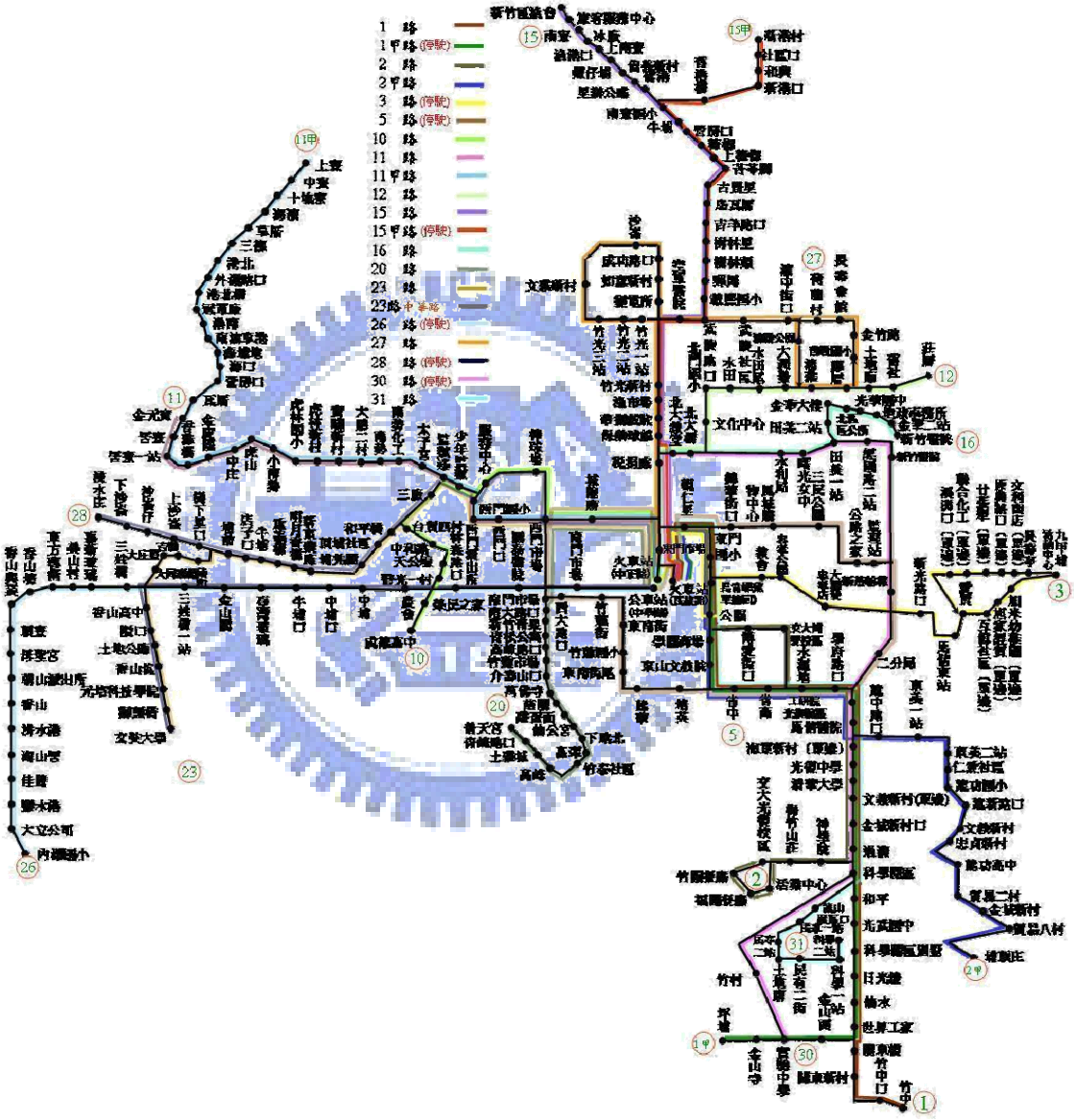

圖 4.2 新竹市公車營運路線圖

本研究測試步驟主要分為以下三點，說明如下：

一、演算邏輯測試：本步驟測試，主要希望能觀察演算結果是否會因路網結構、起迄點設定、出發時間的不同、或者時刻表的差異而產生謬誤。根據本文，大眾運輸路網具有不同一般公路網的特殊性，若未能解決路網特性所產生的

問題，則發展的演算法，就無法協助旅行者作出最有效的旅運決策。因此，實驗設計利用人為控制行前旅次規劃需求資料(包括旅次起迄點、出發時間、轉乘次數限制)，並將測試狀況依照大眾運輸路網特性，分為轉乘次數限制不同之狀況、往返路線停靠站並不對稱、出發時間不同。之後，透過實體時刻表和營運圖來對照方案的輸出結果，藉以判斷正確性，最後，彙整分析所獲得的結果。實驗結果的評估準則以正確率做為指標。

二、演算效能分析：由於本研究發展之演算法，會依照使用者的轉乘次數上限不同，而有可能會有不同的方案，故利用轉乘次數上限為零次、一次、兩次、三次進行各三十筆的測試，分別紀錄計算時間，實驗結果以平均執行時間作為評估的準則。

三、敏感度分析：由於本研究在測試演算邏輯以及效能分析時，將控制變數：轉乘門檻 β 皆設為三分鐘，也就是兩班次到站時間差必須大於三分鐘方能轉乘。故在此部份，將對轉乘門檻 β 值做敏感度分析，試著觀察隨著轉乘門檻的改變，演算法所提供的方案旅行時間的變化，並隨機選取固定的 30 個起迄點配對，進行不同的轉乘門檻值分析，而分析的依據則以 30 條方案之平均旅行時間作為指標。

表 4.12 效能分析

轉乘次數限制	測試筆數	平均執行時間(秒)	準確度
零次	30	0.164	100%
一次	30	0.546	100%
兩次	30	1.672	100%
三次	30	2.394	100%

4.2.1 演算法合理性確認

一、不對稱的往返路線停靠站測試

在實際公車路網中，常會發現單邊設站的情形。因此，主要在測試，演算法是否可判別單邊設站的路線，提供正確的規劃方案。此測試將轉乘次數固定為一次、時間窗限制為早上九點至九點三十分，藉由下圖所示可知文教新村在 1 路返程路線設站，因此，當起點設定為清華大學、迄點設定為文教新村時，將無法求得 1 路公車，直接在文教新村下車，還必須在文教新村，下一站－金城新村口往回走一些距離，方能到達文教新村。但當使用者設定起點為文教新村、迄點為清華大學，演算法才會提供 1 路公車路線的可行方案。

表 4.13 不對稱的往返路線停靠站測試

起點站	迄點站	轉乘次數	時間窗	方案
清華大學	文教新村	最多一次	9:30 到達	09:15:00 在清華大學搭乘 1(去)路線到達金城新村口，約 09:29:00 抵達
文教新村	清華大學	最多一次	9:30 到達	09:10:00 在文教新村搭乘 1(返)路線直達清華大學,約 09:20:00 到達

圖 4.3 新竹市 1 號路線公車停靠站

二、不同的轉乘次數限制測試

本演算法提供的最少旅行時間，是在使用者所限定之轉乘次數下成立的，也就是當使用者設定最多轉乘次數為 n 次，則演算法將提供零次到 n 次轉乘方案中最少旅行時間之方案。若今日使用者將起點站設定為清華大學、迄點站設定為南寮漁港、時間窗限制設定為早上九點出發、轉乘次數為零次時，演算法將無法提供方案。倘若今日將轉乘上限設定為一次，即可提供一次轉乘方案。而將轉乘次數更改為兩次，演算法仍規劃出一次轉乘方案，會發生此狀況的條件有兩個，第一，起迄點只有一次轉乘方案;第二，起迄點在時間窗限制內無兩次轉乘方案，倘若更改時間窗限制後，可能會提供多一次轉乘的方案。

表 4.14 不同的轉乘次數限制測試

起點站	迄點站	轉乘次數	時間窗	方案
清華大學	南寮漁港	最多零次	12:30到達	請放寬時間窗限制或轉乘次數限制。
清華大學	南寮漁港	最多一次	12:30到達	10:10:00 清華大學至搭乘 1(返)路線到達東門市場站,轉 15(去) 至南寮漁港，約 11:30:00 到達。
清華大學	南寮漁港	最多兩次	12:30到達	10:10:00 清華大學至搭乘 1(返)路線到達東門市場站,轉 15(去) 至南寮漁港，約 11:30:00 到達。

三、不同的時間窗限制測試

本演算法提供的方案建議為，時空組合的關聯路線，而非單純未考慮時間之路線組合，意味著符合空間限制之路線組合，並不代表在時間的限制下，此路線組合是可行的，倘若目前時間為早上九點四十分，使用者欲由新竹火車站出發，且希望在十點三十分之前，到達交通大學光復校區，但早上九點十分的班次，已經由火車站發車，而早上十點十分的班次，無法在十點半之前到達。且觀察新竹市區公車，已無其餘方案，可由新竹火車站至交通大學光復校區，故此時，演算法將無法提供方案。倘若，今日使用者設定之時間窗範圍，為早上十點五分出發，此時，演算法將可以提供使用者，搭乘十點十分由火車站出發的班次，直達交通大學光復校區。

表 4.14 不同的時間窗限制測試

起點站	迄點站	轉乘次數	時間窗	方案
新竹火車站	交通大學光復校區	最多一次	10:30到達	請放寬時間窗限制或轉乘次數限制。
新竹火車站	交通大學光復校區	最多一次	10:05出發	10:10:00 在新竹火車至搭乘 2(去)路線直達交通大學光復校區,約 10:40:00 到達。

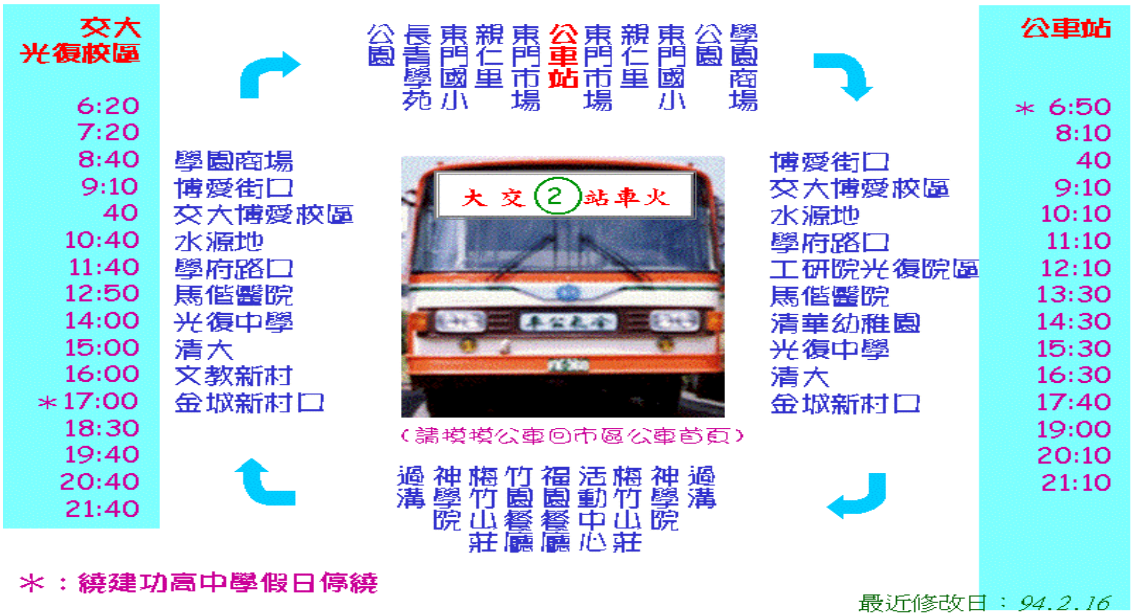

圖 4.4 新竹市 2 號路線公車停靠站

4.2.2 演算法運算效率分析

本演算法利用轉乘次數上限，作為讀取即時資訊的次數，故當使用者將轉乘次數上限，設定的越高則執行時間將會越高。在表 4.12 中也顯示出，當轉乘上限設定的越高時，執行時間越高。但實際上，在新竹市區公車交通路網中，大部分的配對起迄點，皆可在一次轉乘或直達即可到達，故導致本研究，在新竹市區公車範例，可能，因為建構網路的停止條件，而花費了一些不必要的時間，可是，均可在三秒鐘內做出回應，應該還在使用者能夠接受的範圍內。在此，希望後續研究者，可以改善此現象，讓執行時間變得更短。

4.2.3 敏感度分析

本演算法利用轉乘門檻 β 判斷兩路線是否能夠進行轉乘，而本研究利用 3 分鐘作為轉乘門檻，在此部份，利用固定的 30 個起迄點配對進行 β 的敏感度分析，利用規劃出的 30 個方案之平均旅行時間作為指標進行分析。首先，利用每 5 分鐘作為間隔進行敏感度分析，判斷轉乘門檻值落在哪個時間範圍內，平均旅行時間最短。由下表得知，當轉乘門檻設定在 5 分鐘、10 分鐘、15 分鐘、20 分鐘、25 分鐘時，平均旅行時間並未有明顯的改變，而當轉乘門檻值設定為 25 分鐘與 30 分鐘時，可能會因為在轉乘站沒有符合轉乘門檻的班次，導致無法提供方案。當演算法無法提供方案時，本研究令其旅行時間為 120 分鐘，故當轉乘門檻值過高時，平均旅行時間將會比較高。

表 4.15 敏感度分析表一

轉乘門檻值	平均旅行時間(分鐘)
5 分鐘	42.8465
10 分鐘	43.1953
15 分鐘	43.5472
20 分鐘	45.5477
25 分鐘	50.1184
30 分鐘	62.7482

而在零分鐘到五分鐘這區間內，再進行一次敏感度的分析，從零分鐘開始，以每一分鐘作為間隔，來觀察當轉乘門檻設定為多少值，平均旅行時間為最短。由表可得知，在此時間範圍內，平均旅行時間並沒有明顯的減少，主要原因推測為，轉乘站的等待時間，若以五分鐘為門檻，導致每個班次皆可進行轉乘，故在此區間內無任何明顯的差距。然而當轉乘門檻值設定的越高，所提供的方案之旅行時間有機會變得比較高。但也不能夠選擇最少平均旅行時間的轉乘門檻：0 分鐘作為轉乘門檻，因為乘客還必須上車、下車、或者步行一段距離至轉乘站。由下表得知，將轉乘門檻設定為 3 分鐘到 5 分鐘，規劃 30 次的行前旅次規劃，其平均旅行時間並未有明顯的不同，故將轉乘門檻設定為 3 分鐘到 5 分鐘，應該為合理的轉乘門檻。

表 4.16 敏感度分析表二

轉乘門檻值	平均旅行時間(分鐘)
0 分鐘	42.8494
1 分鐘	42.8494
2 分鐘	42.8494
3 分鐘	42.8455
4 分鐘	42.8455
5 分鐘	42.8465

第五章 結論與建議

5.1 結論

1. 本研究將大眾運輸行前旅次規劃，定義為單一邏輯的規劃，並認為其餘邏輯可供使用者，自行作為限制條件進行行前旅次規劃， 將原本的 NP-hard 問題，修改為可使用最短路徑演算法進行求解的問題。以最少旅行時間作為主要邏輯、轉乘次數作為限制式進行求解，並加入動態資訊反應，當使用者開始查詢時，該時間點的運具到站情況，實際反應出較為準確的行前旅次規劃。有效地解決，僅考量時刻表之大眾運輸旅次規劃問題。
2. 利用本研究可以解決過去文獻中，以最少轉乘次數為第一目標式，將會低估起迄點配對之方案數的問題，且當最少轉乘演算法，因時間窗的限制而無法提供時，亦無法提供使用者，任何替代方案的問題，而提供了多一次轉乘，或許可以符合時間窗限制且到達起迄點。如此一來，將可以更加完善的建議使用者，如何在時間窗限制內到達目的地。

5.2 建議

1. 在本研究的網路架構中，可以解決市區公車的單邊設站，以及起迄路線不對稱的現象，但實際上，市區旅運以及城際旅運仍有些不相同，如同在 2.4 節所述說的一些情況，因此，本研究認為，若要將此網路架構以及演算法套入至城際旅運，尚必須將網路架構稍做修改，以便考量到城際旅運的一些特殊的現象。
2. 本研究在本篇論文中，有個強烈的假設：在最少旅行時間邏輯中，使用者僅追求旅行時間最短，並未考慮到票價、步行距離等因素，在大部分的情況都可提供出合理的方案，倘若遇到文中提及的：在每個轉乘站進行轉乘，旅行時間都相同的情況下，將會指派使用者，在轉乘站等待時間最短的站牌進行轉乘。然而，若此現象發生在城際旅次，則乘客有可能必須多付一些費用，故在後續的研究，是否能夠增加票價的限制，或利用本研究之網路架構提出最少票價的規劃邏輯。
3. 本研究在進行新竹市區公車時，利用最傳統的最短路徑演算法，進行選取目前最少成本點的選擇，倘若能夠將站牌座標考量至演算法中，則可進一步使用啟發式解法 A*演算法加快執行的速度。
4. 本研究在定義轉乘可及與建築物的可及站牌時，是利用直線距離進行判斷，本研究認為，有時兩點直線距離，可能在方圓 500 公尺的限制內。然實際上，步行距離有可能超過 500 公尺，本研究建議，後續研究者，可以將方圓 500 公尺，改成實際步行距離 500 公尺作為可轉乘、可抵達之範圍定義。

參考文獻

1. Huang R. and Peng Z.R（2002）,“Design and development of interactive trip planning for wed-transit information systems”,Transportation Research C,Vol.8 ,pp.409-425
2. J.E.McCormack,S.A.Roberts(1996),”Exploiting object oriented methods for multi-modal trip planning systems”,Information and software technology ,38 pp.409-417
3. Ruihong Huang and Zhong-Ren Peng (2001a), “An Object-Oriented GIS Data Model for Transit Trip Planning Systems," Journal of the Transportation Research Board: Transportation Research Records, No.1804, pp. 205-211
4. Ruihong Huang and Zhong-Ren Peng (2001b) ,“Schedule-Based Path Finding Algorithms for Transit Trip Planning Systems,” Journal of the Transportation Research Board: Transportation Research Records, No.1783, pp.142-148.
5. Hickman（2002）,” Robust passenger itinerary planning using transit AVL Data”,IEEE 5th International Conference on Transportation Systems 3-6 September,pp.840-845
6. Koncz,N.,Greenfeld,J.,and Mouskos,K.（1996）,”A strategy for solving multilple-optimal-path transit networks problems”,Journal of Transportation Engineering,Vol.122,Issue 3,pp 218-225.
7. Chao-Lin Lui(2001),”Path-planning algorithms for public transportation systems” Intelligent Transportation Systems. Proceedings.IEEE,pp.1061-1066
8. Qiang Li and Carl E. Kurt (2000), ”GIS-based itinerary planning system for multimodal and fixed-route transit network”, Mid-Continent Transportation Symposium 2000, pp 47-50
9. 劉偉賢，”汽車旅次行前規劃決策支援系統之規劃與設計”，私立中華大學科技管理研究所碩士論文，民國九十四年。
10. 何文基，” 整合時刻表之大眾運輸行前旅次規劃分析方法”，私立中華大學科技管理研究所碩士論文，民國九十五年。
11. 張存保、李華、嚴新平、趙新澤、高虹亮，「基於 WebGis 的程式公交問路系統」，2003 海峽兩岸智慧運輸系統學術研討會，A2-9~A2-15，民國九十三年
12. 蘇夢豪等人，”海陸空客運資訊中心城際與都市客運之整合建構（一）”

交通部運輸研究所，民國九十六年九月。
13. 蘇夢豪等人，”海陸空客運資訊中心城際與都市客運之整合建構（二）”交通部運輸研究所，民國九十七年六月。
14. 交通服務 e 網通(http://210.69.172.215/mtaiwan/default.aspx)
15. 台北大眾運輸旅次規劃系統(http://5284.taipei.gov.tw/)
16. 台中公車動態與網路轉乘系統(http://citybus.tccg.gov.tw/web/default.aspx)
17. 高雄市公車轉乘系統(http://khbus.gov.tw/transit/transit.jsp)
18. 台灣鐵路局旅次規劃系統 (http://service.tra.gov.tw/tw/ticketprice/index.aspx)
19. 高速鐵路旅次規劃查詢系統 (http://www.thsrc.com.tw/tw/travel/route/transfer.aspx)
20. 新竹市區公車旅次規劃系統 (http://service.tra.gov.tw/tw/ticketprice/index.aspx)
21. 台南市區公車規劃系統(http://tourguide.tncg.gov.tw/tnbus/)