結合即時公車與計程車資訊的旅遊路徑規劃系統

藍信翔　陳健　陳俊穎　莊敬中　孫冠宇

國立交通大學資訊工程系

bluelewis.cs96g@g2.nctu.edu.tw　chienchen@cs.nctu.edu.tw　jyc@cs.nctu.edu.tw　gdly01.cs97g@nctu.edu.tw

joshuasun.cs97g@ nctu.edu.tw

***摘要*—由於 Google Maps 的熱門及智慧型手機的興起，人們普遍使用手機上網查詢旅遊資訊，為符合現代潮流，各大城市也相繼推出線上旅程規劃系統。常見的旅程規劃系統，能依使用者所輸入的起點及終點，透過電子地圖的介面來導引使用者搭乘大眾交通工具，並顯示各種可行路徑及各車站的預估抵達時間。有些系統更進一步顯示旅行路線與各站周圍的景點。然而旅行路線有許多不確定因素，如等待時間、步行距離等，一個完善的系統需考慮多種因素，讓使用者有更多選擇。大城市中計程車是一個普遍的交通工具，因此我們設計一套結合台北市公車與計程車的資訊規劃系統，讓使用者用手機來查詢。此系統以 Google Maps 為底圖，給定一個車資上限，能計算出旅程路線與預估旅程時間。由於台北市的公車密度相當高，簡單的路徑搜尋演算法會耗費較多系統資源。我們提出了改良的演算法，可以經過處理而省略一些公車站牌資訊，能有效降低資料處理數量。**

***關鍵詞*—旅程規劃、公車動態資訊系統、Google Maps、計程車**

一、導言

為了響應節能減碳，提高人們使用大眾傳輸工具的比例，及減少自用車的使用，進而降低交通壅塞量以及汽機車廢氣排放量，所以一個功能強大且符合使用者需求的大眾傳輸工具旅行規劃系統是必要的。同時，為了發展觀光，觀光地區通常都人滿為患，車流過於擁擠，找停車位不易，若能有完善的旅行規劃系統，就可提高民眾使用大眾傳輸工具旅行的意願。

在大都市中，旅行規劃系統是被普遍使用的，透過電子地圖的介面來導引使用者搭乘大眾交通工具，並顯示各車站的預估抵達時間，使用者可對路線週遭環境來做查詢。其中還有一些額外的功能，例如：最短旅行時間，最少轉乘次數，行走距離限制，交通工具的選擇等，來提供使用者做為附加條件，更能符合使用者的需求。

使用者搭乘公車有許多時間因素，如：到達公車站點之時間，公車尚未到達之等候時間、公車之行車時間。而公車之行車時間的計算非常複雜，它包含旅客上下車的等候時間、公車靠站時間，交通壅塞時間、號誌遲滯時間等因素。並且在上下班時段時，通常班次不是固定的，所以時間是不穩定且不好掌握的。當一個旅客初次拜訪一個陌生的城市時，常會搭錯車或下錯站往往浪費許多時間，甚至迷路，情況都有可能發生。若能有即時性的旅遊規劃系統，就能有效的掌握時間，避免突發狀況。

現行規劃系統皆透過即時公車動態資訊去規劃出最佳旅行路線，對於即時規劃路徑，可以說是相當準確的。但有時規劃出的路徑其公車站牌過於遙遠，行走距離相當長且耗時耗力，或等待公車時間太久，皆會降低使用者的搭乘意願。若公車班次較少或誤點，或是乘客過多，導致時間方面可能延遲，使得使用者無法有效的掌握時間，可能耽誤行程。要是公車故障，或坐過頭，使用者則需要有更即時的交通工具使用。當使用者有即時變更行程的需求，計程車是一種可供即時轉乘的交通工具，但花費也相對較高，當時間緊迫又不想花大錢時，使用者希望作最有效的花

費，達到即時到達目的地的需求。有鑑於此，本論文結合公車及計程車來規劃路徑，讓使用者做最佳的選擇。

本論文的目的為設計一套結合台北市公車資訊與計程車的規劃系統。有鑑於現代的旅行規劃系統，皆強調互動性與整合應用而使用 Web 2.0 的網路介面，所以系統將採用網路上人們接收度高的 Google Maps 當底圖，以及以 AJAX 技術發展的 Google Map API 與使用者互動，可讓使用者透過圖形介面查詢地圖上的景點或商店資訊以及衛星影像圖供使用者觀看，同時我們也設計以 Android 手機為平台的服務，提供行動上網讓使用者線上查詢或透過 Android 介面來規劃路徑。而 Google Maps API [7]是一套網路使用者使用的工具函式庫，其圖資是免費的，我們將 Google Maps 與地圖伺服器及資料庫進行整合應用，將來更可與計程車業者的系統做整合，形成一個無縫銜接的交通網。

台北市的公車密度相當高，城市中若公車站牌多倍數於路線時，且路線重疊率相當高時，如果用 R. Huang 所提出的交通網路上的路徑搜尋演算法[1]，由於每個公車站牌皆須存放一個包含到達時間的資料，所產生的記憶體資料量過於龐大，導致系統效能低落，所以我們提出了一個改良的演算法，動態產生資料，且演算法可以經過處理而省略一些公車站牌資訊，讓資料處理數量能有效降低。同時我們也加入計程車這個交通工具的新的演算法。

本篇文章接下來的部份包括，第二章：相關研究，我們查詢了一些城市的旅行規劃系統，歸納出一些特別功能，以及說明一個前人的路徑規劃的演算法。第三章：系統架構，規劃架構中各部份的相關性，手機與網站介面開發設計及各應用模組開發，資料庫與伺服器的整合應用。第四章：優化路徑搜尋演算法，由於使用前人的演算法將會相當耗系統資源，所以我們提出有效減少記憶體的改良演算法。第五章：路徑搜尋加入計程車的演算法，對於使用者的旅程路線提供更緊密且具多變性的路徑規劃。第六章：模擬結果與分析，影響時間因素分析。第七章：結論及後續發展的可能。

二、相關研究

我們瀏覽一些國外及國內旅行規劃系統，透過詳細的觀察來對各系統所提供的功能進行比較及歸納，國外城市：包含美國的紐約[8]、芝加哥[9]，法國巴黎[10]、英國倫敦[11]；國內則有台北[12]、新竹[13]、台中[14]、台南[15]。除此之外，我們也介紹全球性的系統 Google Maps[16]，地區性的系統 UrMap[17](台灣)。

其中較特別的是紐約市地圖顯示有 2D 或 3D 的選項；倫敦有 19 種語系支援及單車路徑支援；巴黎網頁設計風格時尚且景觀介紹詳細；Google Maps 對有名的建築物以維基百科做註解，能透過網路攝影機可以看到道路即時訊息。我們總結國內外的系統功能作整理，並列出功能列表為表 1，功能包括：

Fastest：選擇最短時間到達的路徑

Walk：表示使用者能設定容忍行走距離

Transfers：選擇最少轉乘次數的路徑

Waiting：能設定容忍等待時間

Cost：顯示使用者的花費

Map：有使用地圖來引導使用者

3D：除了空照衛星圖外，額外 3D 顯示功能

Taxi：額外交通工具 Taxi 支援

由表 1 可以看出，大部分的規劃系統，皆沒有 Cost 及 Waiting 的選項，國外系統 Walk 及 Transfers 的功能幾乎都有實作出，有些系統地圖會額外加工，除了 3D 功能以外，本論文將實作全部的功能，再加上其他城市皆沒有的交通工具：計程車。

表 1: 功能比較圖

	Fastest	Walk	Cost	Transfers	Waiting	Map	3D	Taxi
紐約	O	O	O	O	X	O	O	X
芝加哥	O	O	O	O	X	X	X	X
倫敦	O	O	X	O	X	O	X	X
巴黎	O	O	X	O	X	O	O	X
台北市	O	X	X	X	X	O	X	X
新竹市	O	X	X	X	O	X	X	X
台中市	O	X	X	X	X	O	X	X
台南市	O	X	X	X	X	O	X	X
UrMap	O	X	X	X	X	O	X	X
Google	O	X	X	X	X	O	X	X
本論文	O	O	O	O	O	O	X	O

Florian[2]提出以 Dijkstra's algorithm 為其基礎做修改的最短時間路徑方法。在大眾傳輸系統中，規劃路線在不同時間點有不一樣的路線及轉乘站[3]，[6]。另外提出動態處理大眾交通系統模組，並於路線上給定數值的資料結構[4]，[5]。本論文所參考的 Path finding Algorithm Using Pattern First Search[1]也是以 Dijkstra's algorithm 為基礎來改進，設定起點出發時間可推算出最快到達終點時間。大眾轉乘系統路線圖 NET(L,N,S)，由 L：路線(Line)、N：可轉乘站(Node)、S：不可轉乘站(Stop)所組成，若是不可轉乘站過多，資料處理將會過於龐大，做資料處理時只考慮路線與可轉乘站的話，可用以減少資料處理，路線圖 NET(L,N,S)可精簡成 NET(L,N)。

首先紀錄對於每個路線所包含的轉乘站，以及每個轉乘站所包含的路線。每個轉乘站包含一個 timestamp，資料參數如下

t　到達這轉乘站的時間
r　到達這轉乘站所搭的路線
x　搭乘公車路線 r 的上一個轉乘站
t^x　轉乘站 x 的出發時間

演算法主要需定義一個暫存集合，紀錄下一步所需更新的點，而集合內的點則依時間 t 大小來排序。設定起點出發時間，其他車站到達時間為無限大，開始從起點 s 去搜尋鄰居，鄰居為起點所在的路線 r 上，且為路線上的下行車站，計算新的到達時間 t'，比較原到達時間短則更新且加入集合內且更新轉乘站的 timestamp (t', r, s, t^s)。下一步，找集合內時間最小的點。重複步驟直到找到終點，或集合為空。

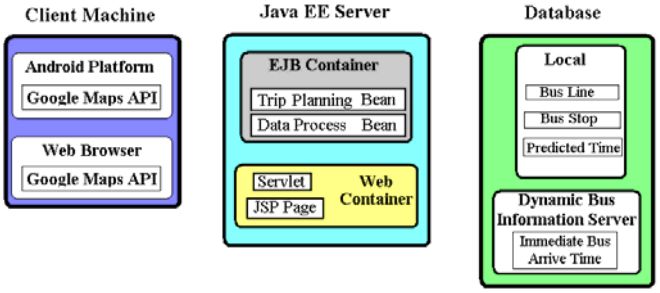

圖 1: 系統架構圖

三、系統架構

我們旅行規劃系統將讓使用者透過網路瀏覽器，及手機上 Android 平台來進行路徑規劃。由於 JAVA 的平台相當受歡迎，無論是手機平台或是伺服器端，都有很多程式開發者使用，所以我們統一以 JAVA 為程式開發語言，來設計手機軟體及伺服器端的程式，這樣就不會有多種程式語言混雜的情況，其相容性也較高，規劃設計一個 JAVA EE Server，讓使用者連上網路時以瀏覽器觀看，透過地圖介面自行設定起終點，同時交通工具的選項加入計程車這個交通工具提供使用者選擇，此外有各種影響時間的因素可供設定。本章將說明系統設計架構，並針對系統的功能模組做簡單介紹。

3.1 系統架構分析

分為三個層，分別為客戶端、伺服端、資料庫，客戶端提供使用者使用及觀看的介面，伺服端為伺服器所實作路徑規劃的功能，資料庫為本機所存放的資料庫與遠端抓取即時資訊的資料庫。

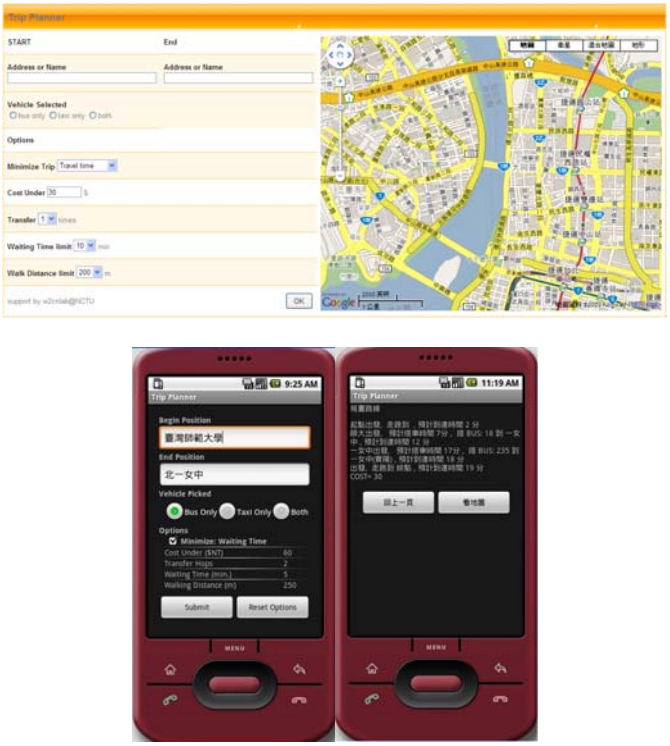

圖 2: 客戶端介面

3.1.1 客戶端

如圖 2 所示，使用 Google Maps 當底圖，另外使用額外 Ajax 框架套件 EXT JS 美化視窗。同時我們在免費智慧型手機軟體平台 Android 開發程式，其平台上有 Google Maps API 可供使用，由於手機上操作上較單調，設計簡潔的介面是必須的。功能選項為選擇最短時間到達的路徑、設定容忍行走距離等。交通工具有計程車可供選擇。所規劃的路線若有轉乘，則各路線用不同顏色來區分。另外在手機上的軟體平台 Android 上開發程式，可供使用者操作更多元，在使用者直立或橫立手機時，畫面能夠切換，不受限於網站介面。

3.1.2 伺服端

包含資料處理模組、旅行規劃模組。旅行規劃模組取得使用者所提供資訊，由資料處理模組分別查詢即時公車時間資訊資料庫及本系統規劃路徑資料庫，之後結合兩個資料庫資訊，交給旅行規劃模組利用結合公車與計程車路徑規劃演算法進行處理，並由旅行規劃模組產生路線、站牌資訊及預計時間。

3.1.3 資料庫

採用 PostgreSQL 資料庫，為一套免費軟體，其 PostGIS 模組可供擴充使用，且有相當多外掛函式庫可供使用。資料庫內資料的來源為台北市政府所提供之台北市公車站牌及路線資訊[18]。另外在規劃路徑時，連上伺服器，抓取台北市即時公車資訊[18]來分析及使用。

3.1.4 運作方式

瀏覽器介面透過 Google Maps API 讓使用者互動，送出使用者自訂資訊給伺服器用來產生路徑規劃程式，程式再去資料庫抓取所需資料，處理後的結果送回給使用者，使用者看到的畫面是透過 Google Maps API 所呈現。

四、優化方法

我們考慮到若公車站牌數目較多且過於密集，演算法更新時所須處理的資料會過於龐大。我們提出一個改進 Huang 演算法的方式，來省略一些計算並可減少記憶體使用空間。

一般而言，轉乘站數量多過路線數，且每轉乘站有多條路線經過。以台北市公車為例，公車路線數量為 429，站牌數為 6782。我們提出的這個優化演算法可減少記憶體使用空間。

在公車路線上，如有兩個車站其中下行車站所經過的公車路線皆為上行車站所經過的公車路線的子集合，而且所經公車路線皆走相同道路，表示無轉程的必要，便可省略。

如圖 3，A、B、C、D 代表轉乘站，L1、L2、L3 代表三個不同的公車路線，以 A、B 兩車站為例，所經公車路線皆走相同道路，其中 B 不為終點。B 節點路線集合{L1,L2}為 A 節點路線集合{L1,L2,L3}的子集合，表示無轉程的必要，便可省略 B 節點。

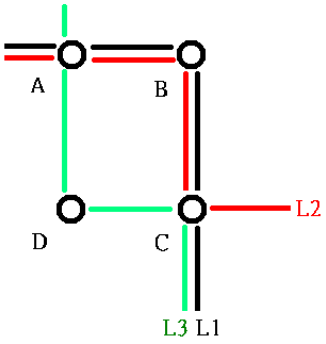

圖 3: Bus Line Example

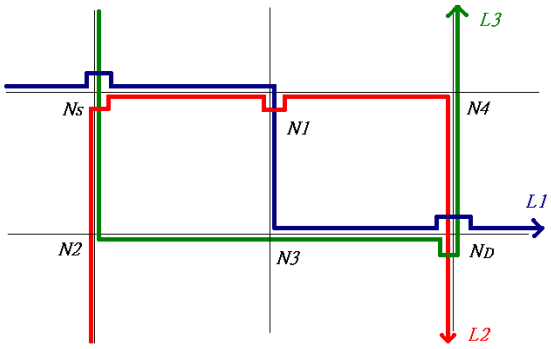

圖 4: Example

4.1 演算法

我們的演算法流程如下。大眾轉乘系統路線圖 NET(*L*,*N*) 由 *L*：路線及 *N*：轉乘站所組成。我們用與 Huang 演算法相同的方式去搜尋路徑，使用同樣參數(t, r, x, t^x)。我們的演算法步驟(6)以上述的方法去優化，在初始化時，即步驟(1)，我們演算法開始只初始起點，不像 Huang 演算法一開始就初始化所有轉乘站，使用龐大的記憶體空間。

(1) 初始化起點 N_S(t_0, null, null, ∞)、每個轉乘站的所屬路線集合，以及每個路線的所屬轉乘站集合，集合 T←{N_S}
(2) 假如 T 是空集合，停止。終點無法到達
(3) 從 T 集合中挑轉乘站 N_j 其 t 為最小
(4) 假如 N_j 所代表的轉乘站為 N_D，停止。終點到達
(5) 找 N_j 所經過的路線除了 timestamp 中的路線 r 之外，且符合等待時間限制
(6) 對找出的路線各別取出擁有的車站，車站 N'必須為 N_j 的順行轉乘站，取出 N'的所屬路線若為 N_j 的所屬路線的子集合，則不產生 N'的 timestamp，反之產生 N'，若 N'存在則比較 N'的 timestamp 的時間 t'，Ni 到 N' 的時間為 t*，若 t+t*<t'成立，則更新 N'的 timestamp，假如 N'不為 T 的子集合，T ← T ∪ {N'}
(7) T ← T − {Nj}
(8) Go to step (2).

4.2 範例

如圖 4 所示，有三條公車路線(L1,L2,L3)及六個公車站牌點(N_S, N_D, N1, N2, N3, N4)，起先初始化每個公車路線包含站牌集合，以及每個公車站牌點包含路徑集合，起點時間 t 設為 7:00。

步驟一：

在 N_S 的所屬路線集合為{ L1, L2, L3}：

L1 的所屬轉乘站集合為{N_S,N1,N3, N_D}，只考慮下行站{ N1, N3, N_D}，N1、N3 的所屬路線集合，為 N_S 從所屬路線集合的子集合，所以不更新。N_D 為終點，計算從 N_S 走路線 L1 到 N_D 的時間，產生 N_D。如圖 5 (a)。

L2 的所屬轉乘站集合為{ N2, N_S, N1, N4, N_D }，只考慮下行站{ N1, N4, N_D}，N1、N4 的所屬路線集合，為 N_S 從所屬路線集合的子集合，所以不更新。N_D 為終點，計算從 N_S 走路線 L2 到 N_D 的時間，比 N_D 的時間大，不更新。如圖 5 (b)。

L3 的所屬轉乘站集合為{ N_S, N2, N3, N_D, N4 }，只考慮下行站{ N2, N3, N_D}，N2、N3 的所屬路線集合，為 NS 從所屬路線集合的子集合，所以不更新。N_D 為終點，計算從 N_S 走路線 L3 到 N_D 的時間，比 N_D 的時間大，不更新。如圖 5 (c)。

步驟二：

取集合內的點為 N_D，完成。

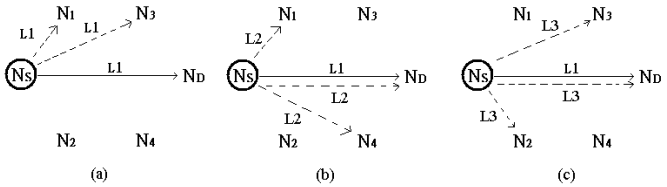

圖 5: State

五、路徑規劃加入計程車演算法

為了能夠讓旅程路線更具多變性，而不再只是使用大眾傳輸工具，我們考慮當兩條大眾傳輸工具的行車路線無交集時，可以搭計程車來銜接這兩條路線，以帶給使用者更多的路徑選擇。

交通路線圖除了原本轉乘路線、轉乘站外，每個轉乘站與轉乘站之間，皆有以計程車為交通工具的路線連接。

在做搜尋時，由於有條件限制，所以除了起點及終點只有一個時間標記之外，其他可轉乘站能有多個時間標記。假如以花費為限制條件，在未抵達終點前，累積花費可能已超過限制，所以花費及時間必須一起紀錄。但紀錄也必須篩選。可轉乘站只須紀錄兩種情況：時間少但花費高或是時間多而花費少。

演算法更新某轉乘站時，是以到達時間及花費來做比較。第一種情況，若新資料中到達時間及花費有比原先標記中有較佳的值，則刪除所有較差的標記，增加新的標記。第二種情況，若新資料中到達時間及花費有比原先標記中較差的值，則不更新。最後，若以上情況皆沒發生，則增加新的標記。

5.1 演算法：

考慮公車、走路、計程車路線，計算到達時間及花費。

(1)在設定的走路範圍限制下，從起點搜尋行走距離能找到的車站，以及坐計程車到車站及終點的路線。

(2)得到的車站下所經過的公車路線，若路線下的車站若存在車站集合內，則更新時間標記其內容為此車站。

(3)搜尋行走距離能找到的車站，檢查是否能更新車站的時間標記。

(4)搜尋計程車能找到的車站，檢查是否能更新車站的時間標記。

(5)之後將新增的時間標記存在記錄集合內。

(6)假如記錄集合內無東西，跳出。

(7)依序取出記錄集合內時間標記，時間標記點內的 Node 為起點，假如之前是坐計程車則忽略(9)(10) 。

(8)找起點所在的公車路線，若路線下的車站存在於車站集合內，檢查是否能更新車站的時間標記。

(9)搜尋行走距離能找到的點，檢查是否能更新車站的時間標記，檢查若有更新則存在記錄集合內。

(10)搜尋計程車能找到的點的紀錄，檢查是否能更新車站的時間標記，檢查若有更新則存在記錄集合內。

(11)移除時間標記，將(8)(9)(10)新增的時間標記存，回(6)。

5.2 範例：

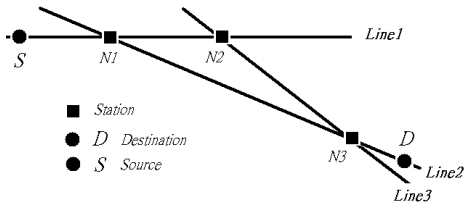

圖 6: Example

如圖 6，假設花費上限 90，起點為 S，終點為 D，N1、N2、N3 皆為車站。

步驟 1：

起點 S 能坐公車到 N1、N2，及坐計程車到 N1、N2、N3、D，經由更新後的結果為表 2。

表 2: 步驟 1

Record	Previous	Line	Node	Time	Cost
1	0	L1	N1	7:15	15
2	0	Taxi	N1	7:05	70
3	0	L1	N2	7:20	15
4	0	Taxi	N2	7:07	75

步驟 2：

取記錄中時間最小的記錄 Record 2，Node N1 能坐 L1 公車到 N3、D，經由更新後新增的 Record 為表 3。

表 3: 步驟 2

Record	Previous	Line	Node	Time	Cost
5	2	L2	N3	7:21	85
6	2	L2	D	7:25	85

步驟 3：

取記錄中時間最小的記錄 Record 4，Node N2 能坐 L2 公車到 N3，經由更新後新增的 Record 為表 4。

表 4: 步驟 3

Record	Previous	Line	Node	Time	Cost
7	4	L2	N3	7:23	85

步驟 4：

取記錄中時間最小的記錄 Record 1，Node N1 能坐 L2 公車到 N3、D，及坐計程車到 N2、N3、D，經由更新後新增的 Record 為表 5。

表 5: 步驟 4

Record	Previous	Line	Node	Time	Cost
8	1	Taxi	D	7:19	85
9	1	Taxi	N3	7:20	85

步驟 5：

取記錄中時間最小的記錄 Record 6，為 Node D，結束。

表 6: 步驟 5

Record	Previous	Line	Node	Time	Cost
1	0	L1	N1	7:15	15
8	1	Taxi	D	7:19	85

由表 6 得知最快路徑為從起點 S 坐 L1 公車到 N1 抵達時間 7:15，在由 N1 坐 Taxi 到終點 D 抵達時間 7:19。

六、模擬

為了了解我們系統的性能，我們做了一系列模擬。模擬時所取的樣本，是在台北市內為長 3.735 公里，寬 2.632 公里的長方形區域，隨機產生起點及終點並限制其路線距離各為 2 公里、3 公里及 4 公里。我們設定各種條件參數，針對每種距離各模擬 100 次，統計其結果，並討論下列因素所造成旅行時間與轉乘比例的差異：

(1)行走距離：使用者能接受的行走距離。

(2)等待時間：預計公車到站的時間。

(3)花費：使用者能提供的金錢上限。

6.1 步行距離

我們想了解若使用者願意走遠一點，是否能選到較快的公車路徑。模擬時所有公車的等待時間設為相同，步行速度設為 4 公里/小時，在起點到終點各為 2、3、4 公里時，量測使用者能夠行走之距離。圖 7 橫軸為可接受之步行距離，縱軸為找到公車路徑的機率。結果發現當使用者能夠接受的步行距離越長，成功搭乘的機會越高。圖 8 橫軸為可接受之步行距離，縱軸為旅行時間。結果顯示使用者能夠接受的步行距離越長，搭乘時間也能減少。若步行速度不是相當快的話，其時間差異是不大的。

6.2 等待時間

在規劃路徑時，規劃出不轉乘的路線可能比有轉乘的時間慢，但轉乘次數通常代表花費會增加，且轉乘之間的等待時間需考慮，我們設定兩種等待公車時間來量測，觀察轉乘次數比例的不同。

圖 9 橫軸為起點到終點的距離，縱軸為沒轉乘與轉乘一次的百分比。結果顯示，當等待公車時間較短時，也就是換車所需時間較短時，系統有較高的比例去規劃轉乘，來減少旅行時間。

6.3 花費

此模擬目的在於了解花費與旅行時間的關

係。在起點到終點為 2、3、4 公里時，圖 10 橫軸為花費上限，縱軸為旅行時間，結果為當花費上限越高時，能到達終點的時間越快。當起終點為 2 公里，花費超過 85 元上限時，規劃路徑皆為直接搭乘計程車到達目的地。

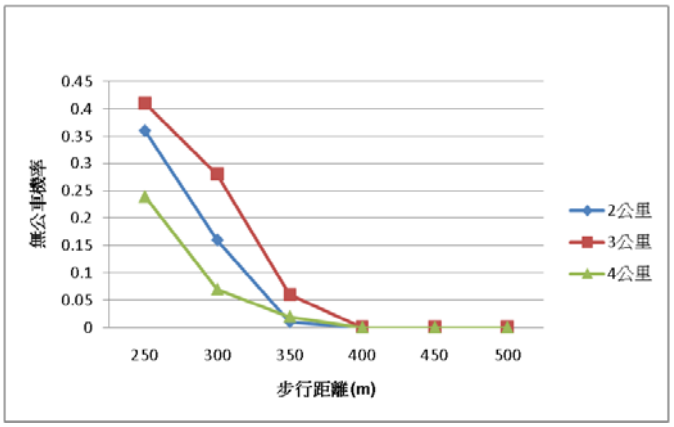

圖 7. 模擬步行距離(1)

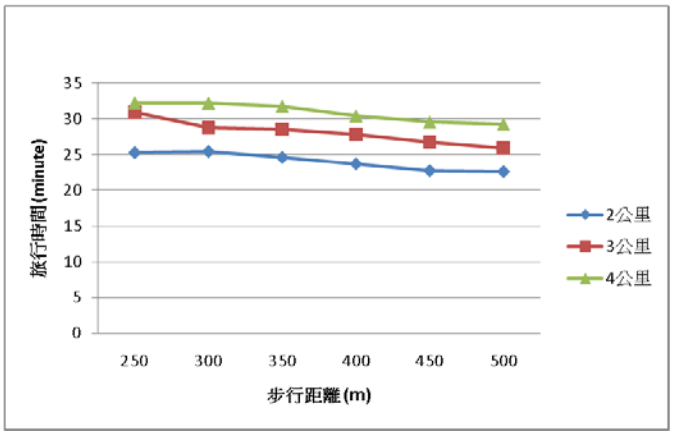

圖 8. 模擬步行距離(2)

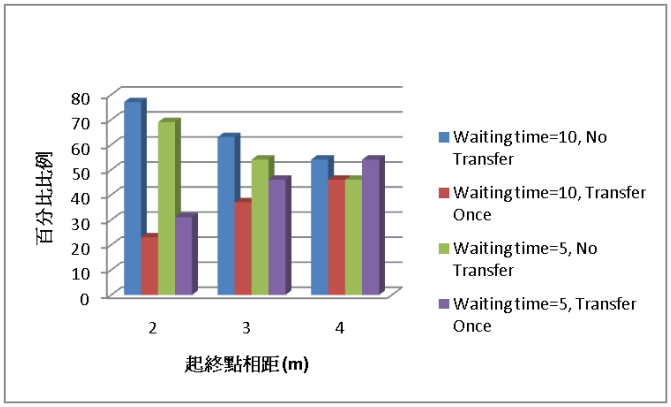

圖 9. 模擬等待時間

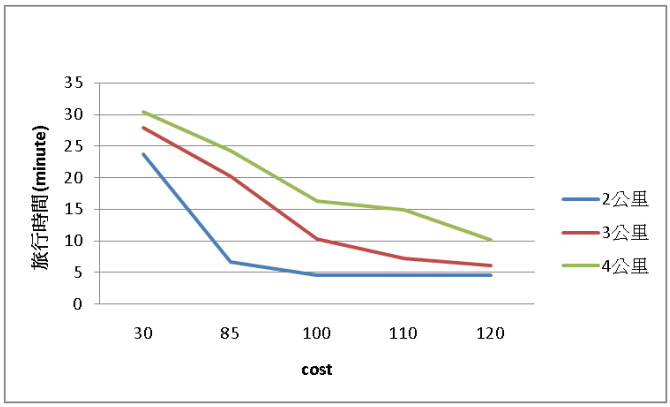

圖 10. 模擬花費

七、結論

本論文提出一套支援公車與計程車混和搭乘的旅程規劃系統，藉此讓使用者能規劃出更符合需求的路徑。另外在規劃路徑時，我們提出降低資料處理量的演算法。由於加入計程車為交通工具選項會導致運算量大增，未來將研究更周詳的方法來降低運算量。另外，從最近的單車熱可知購買自行車的人數不斷增加，規劃時也可考慮加入自行車，建立腳踏車專用道資料庫，使路徑規劃更具多元性。達成建立一個無縫銜接交通網的旅遊規劃系統。

八、誌謝

本論文所採用之台北市公車站牌及路線資訊係由台北市政府交通局所提供。

九、參考文獻

[1] R. Huang, “A Schedule-based Pathfinding Algorithm for Transit Networks Using Pattern First Search,” *Geoinformatica*, vol.1, no. 2, pp.269-285, June 2007

[2] M. Florian, “Finding shortest time-dependent paths in schedule-based transit networks: a label setting algorithm,” in Niguel H.M. Wilson and Agostino Nuzzolo (Eds.), *Schedule-based Dynamic Transit Modeling: Theory and Applications*, pp. 43–53, Dordrecht Kluwer, 2004.

[3] R. Huang and Z.-R. Peng. “An object-oriented GIS data model for transit trip planning system,” in TRB, National Research Council (Eds.), *Transportation Research Record*, no. 1804, pp. 205–211, TRB, National Research Council, Washington DC, 2002.

[4] F.Russo. “Schedule-based dynamic assignment models for public transport networks,” in Niguel H.M. Wilson and Agostino Nuzzolo (Eds.), *Schedule-based Dynamic Transit Modeling: Theory and Applications*, pp. 79–93, Dordrecht Kluwer, 2004.

[5] C.O. Tong and S.C. Wang. “Minimum path algorithms for a schedule-based transit network with a general fare structure,” in Niguel H.M. Wilson and Agostino Nuzzolo (Eds.), *Schedule-based Dynamic Transit Modeling: Theory and Applications*, pp. 241–261, Dordrecht Kluwer, 2004.

[6] R. Huang and Z. Peng. “A spatiotemporal data model for dynamic transit networks,” *International Journal of Geographic Information Science*, vol. 22, no. 5, pp. 527-545, 2008.

[7] Google Maps API
http://code.google.com/intl/zh-HK/apis/maps/

[8] MTA NYC Transit - Trip Planner
http://tripplanner.mta.info/

[9] Plan Your Trip With the RTA!
http://tripsweb.rtachicago.com/

[10] RATP Transports en île de France
http://www.ratp.info/touristes/

[11] Transport for London
http://www.tfl.gov.uk/

[12] 5284 首頁
http://5284.taipei.gov.tw/

[13] 新竹市旅運規劃系統
http://hisatisfy.hccg.gov.tw/eplan/

[14] 臺中都會區 公車動態 暨 路網轉乘系統
http://citybus.tccg.gov.tw/

[15] 台南市公車觀光導遊
http://tourguide.tncg.gov.tw/tnbus/

[16] Google 地圖
http://maps.google.com.tw/

[17] UrMap 你的地圖網
http://www.urmap.com/

[18] 臺北市即時交通資訊網
http://its.taipei.gov.tw/