基於車輛網路動態路徑規劃之節能共乘配對演算法與 Android 平台上的實作

張英超　洪似妮　厲秉忠　李建德　邵盈智　王傳啟
國立彰化師範大學資訊工程學系
icchang@cc.ncue.edu.tw

摘要

現有共乘系統中，大多由司機在共乘平台上公開自己行程，乘客再根據司機公開的資訊選擇適合自己行程。此做法不易選到適合自己的司機或乘客，而雙方的個人偏好也不同，無法獲得良好的配對與路徑規劃。本研究希望設計 Android 應用程式，首先利用車輛網路和平台本身搭載的感測器收集路段資訊，利用車輛網路間的無線傳輸將蒐集的資料分享到周圍車輛節點，使路段和車輛狀態皆能夠即時更新，避免交通阻塞，接著由司機與乘客提出行程需求與選擇行駛模式為最少油耗或最短時間，透過「共乘配對演算法」，找出符合雙方需求的最佳配對結果並即時規劃出共乘路線，設計出支援「最少油耗」與「最短時間」的車載網路 A*(VBA*)路徑規劃演算法降低費用與節能省碳之目的。

關鍵詞：車輛無線網路、VBA*、共乘

Abstract

For most existing carpool system, drivers usually open their own itinerary on carpool platform, and then passengers choose the drivers according to the public information of the drivers. This method is not easy to meet the needs of users, and both individual preferences are different. It cannot get a good pairing and route. This project purpose is to design an Android application, which is according to driver's and passenger's travel demands as well as travel mode with "Carpool-Paring Algorithm" and modified A-Star Routing Algorithm to find the better paring result that is match both demands and plans carpool route in time. We also use VANET and sensors to collect road information so that road and car's information can update in time to avoid traffic jam to reach saving time and energy.

Keywords：VANET、VBA*、Carpool

1. 系統簡介

新聞報導曾有 5 位英國人過去 20 年都共乘一台車上下班，估計省下約新台幣 460 萬元，估計總省油 9.46 萬公升，節能減碳高達 280 噸[1]。由此得知，共乘確實在環保與金錢花費中達到良好效果。目前汽車導航結合 Geographic Information System(GIS)與 GPS 衛星定位，實現路徑規劃及道路導航等功能，並利用最短路徑演算法(Dijkstra's、A*等)計算兩點間最短路徑，但使用者無法針對實際的道路狀況快速調整規劃路線。

因此本研究期望並非只有共乘配對，更根據即時路況進行路徑規劃，利用 Android 平台實作開發出車輛上的感測器或平台所搭載的裝置，隨時與鄰近車輛交換道路訊息，更新原行駛路徑會遇上塞車路段，接著利用本研究提出的共乘配對演算法來決定該司機接送乘客的最佳順序，以 VBA*動態規劃出兩點乘客間的行駛最短時間及耗油最少的路徑，達到省時省油之共乘目的。本研究計畫貢獻如下：

1. 使用共乘配對演算法，依照乘客及司機的個人化需求找出最適合共乘配對的方式。
2. 以 VBA*規劃最佳行駛路徑，達到節能目的。
3. 能夠快速即時重新規劃共乘路徑。
4. 建立司機與乘客間評分機制及良善共乘環境。
5. 車輛透過 Android 平板搭載感應器，實作系統。

接下來的第二章節會有文獻探討的部分，將針對現有發展進行探討；第三章節為系統設計，將提出本研究所設計的方法；第四章節為環境建置與程式實作，將有本研究所實作的畫面。

2. 文獻探討

2.1 VANET 網路

VANET(vehicular ad-hoc networks)是以 Ad-Hoc 網路為基礎下車輛間通訊設計的新形態網路模式，以移動中的車輛及交通設施為節點，利用無線通訊技術，來形成行動網路。車載隨意網路包含了車輛與車輛間的通訊(InterVehicle Communications 簡稱為 IVC)以及路邊基地台與車輛間的通訊(Roadside-Vehicle Communications 簡稱為 RVC)。

2.2 A*(A-Star)路徑演算法

A*路徑演算法使用最佳解優先(Best-First

search, BFS)概念，其目的為給定一個初始節點及目的節點，找出最低成本的路徑。A*路徑演算法透過使用啟發函式(heuristic function)，對於道路進行成本預估，表示為公式(1)：

$$F(n) = G(n) + H(n) \quad (1)$$

其中 G(n)表示從起點到任意頂點 n 的實際距離，H(n)表示任意節點 n 到目標節點的估算距離。

2.3 導航模式與評估函式設計[2]

事實上，發動機負荷率、車速、有效油耗這三者是相輔相成的[3]。基於上述的觀點，使用[2]提出VBA*演算法所需的路徑規劃模式：

(1) 最短路徑模式

每條道路皆有固定的長度 Li，將起點及終點間所有道路長度相加，表示為：

$$\mathbf{F(n)} = \sum_{i=0}^{n} L_i + \mathbf{H(n)} \quad (2)$$

$\sum_0^n L_i$代表從起點開始到目前路段道路的總長度，H(n)代表目前道路至終點的直線距離。

(2) 最省時間改良公式

$$\mathbf{F(n)} = \sum_{i=0}^{n} \frac{L_i}{S_i} + \frac{\mathbf{H(n)}}{S_n \times (1+\cos\theta_n)} \quad (3)$$

公式(3)為將原始 A*演算法公式和本研究設計車輛行駛間交換的即時路段速度資訊結合，使成本計算方式由長度轉換為時間，迭代計算最小時間成本，得到花費時間最短路徑。

$$\mathbf{F(n)} = \sum_{i=0}^{n} (\mathbf{L_i} \times \mathbf{O}(S_i)) + \frac{\mathbf{H(n)} \times \mathbf{O}(S_n)}{(1+\cos\theta_n)} \quad (4)$$

公式(4)為將原始 A*演算法公式結合本研究設計車輛行駛間交換的即時路段速度資訊結合到公式中，使成本計算方式由長度轉換為油耗量，迭代計算最小油耗成本，最終得到花費油耗最短路徑。

2.4 目前共乘論文的相關研究

共乘系統目前最欠缺的方面在於動態與配對，[4]提出了優化的動態共乘服務，可以解決在分散式並行的請求，使用前提必須在分散式的情況下才可進行，且只在固定點才可接送乘客。但此篇論文在配對部分缺乏說明及研究。[5]中提出了詳細的配對演算法過程，並用屬性（距離和時間）及參數（駕駛者的行為等）來選擇適合的配對。通常這些參數會隨機改變的，而這篇論文針對這一點，提出一個同時也適用於動態環境的方法。但這篇論文僅僅只有配對演算法的過程，缺乏路徑規劃的設計及研究。[6]則是利用過去的歷史資訊去評估乘客的路線，算出每位載客點會接到客人的機率進行評估。

2.5 汽車共乘之實例

(1)桃北北宜基共乘網[7]：首頁有最新共乘資訊和消息，但缺乏路徑規劃，無法看到行車路徑。

(2)共乘王[8]：有分免費會員跟環保會員兩種，後者為付費會員，刊登資訊較易被其他會員看見。其油錢的分攤方法是由司機、乘客自行商議。

(3)共乘 APP：BlueNet[9]：該 APP 使用 Android 平台，缺點是沒有金錢分攤方法。下表格整理上述目前汽車共乘實例的功能比較表(如表 1)。

表 1 目前共乘網站的功能比較表

	共乘王	綠色共乘	BlueNet	本系統
路徑規劃功能	無	無	無	有
油錢分攤方式	司機議價	司機議價	司機議價	系統公平分配
個人喜好設定	無	有	無	有
司機評分	無	無	無	有
預約行事曆	無	無	無	有
如何進行配對	透過其他平台	透過其他平台	透過系統	透過系統
即時路況	無	無	無	有

3. 系統設計

圖 1 為汽車共乘示意圖，C1、C2 為司機 1 和司機 2；p1 和 p2 為乘客 1 和乘客 2，而下列的演算法將會規劃出共乘路線及配對。

圖 1 共乘節點示意圖

3.1 主程式流程

步驟 1：乘客與司機先設定共乘需求選項(如表 2)並透過 WiFi 網路傳至 Server 儲存配對。

步驟 2：Server 依照表 2 進行「共乘比對過濾」，並將通過過濾的司機列表給乘客選擇。

步驟 3：乘客選擇想要的司機，並將選擇的結果回傳給 Server。

步驟 4：Server 進行「共乘乘載順序演算法」，如果成功會將司機更多詳細資料傳給乘客；如果不成功，也會通知該司機不符合該名乘客需求，並回到步驟 3 選擇其他司機。

步驟 5：乘客看過司機詳細資訊後，按下 OK，Server 會將該名乘客傳給選擇的司機確認。

步驟 6：Server 會將驗證碼、路徑及油錢傳給乘客；司機若有空位則繼續等待其他乘客選擇，直到可乘載人數已滿或達出發時間。如果司機不確認，則乘客回步驟 3。

步驟 7：在指定地點時間會合，開始行程。

步驟 8：乘客下車後，司機與乘客互相評分，並將雙方的評分資訊傳回 Server 紀錄。

乘客、司機以及 Sever 流程圖，分別如圖 2、3、4。

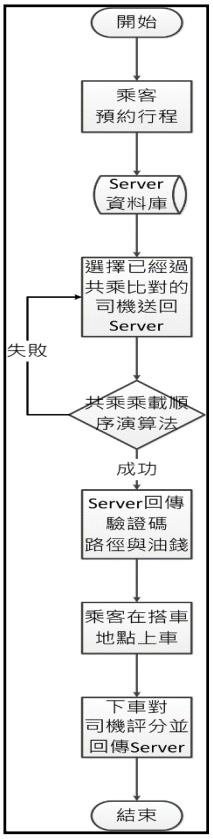


圖 2　乘客流程圖

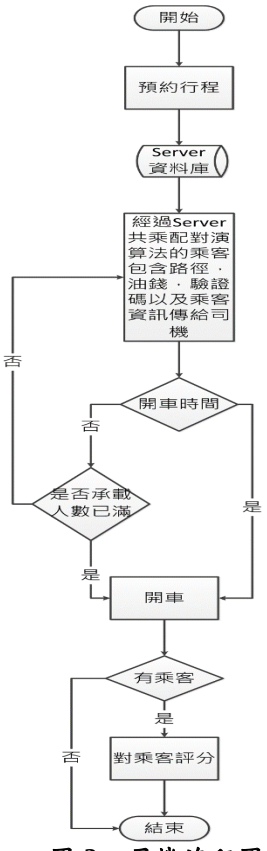


圖 3　司機流程圖

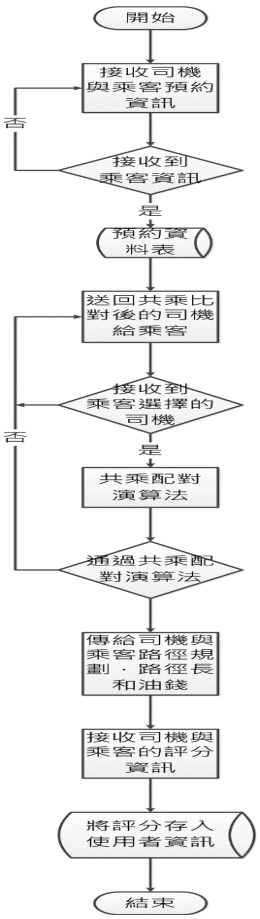


圖 4　Server 流程圖

表 2　駕駛與乘客參數

司機參數	乘客參數
出發時間(預定)	出發時間(預定)
到達時間(預定)	到達時間(預定)
車牌號碼	
車種	願意支付金額
最大乘載人數	共乘人數
可否包車	可否包車
司機終點	乘客終點
寵物.孩童.吸菸.乘客性別	寵物.孩童.吸菸.乘客性別

3.2 共乘配對演算法

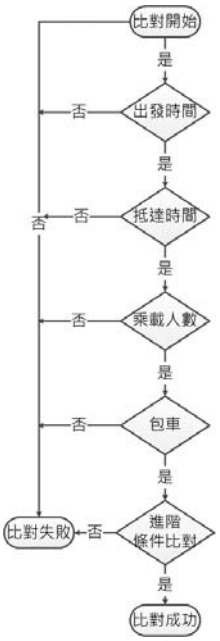


圖 5　配對演算法流程圖　圖 6　共乘比對流程圖

如圖 5，共乘配對演算法除了如圖 6 的乘客與司機本身條件過濾比對外，必須含共乘配對組合、乘載順序、共乘費用分攤。共乘配對組合是將乘客分配給司機的演算法。乘載順序依自由喜好選擇司機的演算法，提供最短路徑、最低油耗、最短時間的高低順序給乘客選擇。共乘油費分攤提供乘客以及司機行程油錢消耗，讓使用者評估共乘效益。

3.2.1 共乘過濾條件

為了執行主程式流程的步驟 2：找出乘客及司機最適合的共乘配對，其條件比對採取只要有一項不合需求便會淘汰的機制，以大幅過濾不適合的共乘對象(如圖 6)。

第一步：比對司機起點出發時間是否小於等於乘客起點出發時間。是的話前進第二步，若不符條件則棄掉這組配對不用。

第二步：比對是否司機是否能在乘客需的最晚時限內抵達目的地，是的話繼續第三步，若不符條件則棄掉這組配對不用。

第三步：比對司機車上乘客數目是否小於最大

承載人數(司機主動提供的條件)，意即是否有空位給乘客乘坐，是的話繼續第四步，不是的話棄掉這組配對不用。

第四步：比對有乘客包車條件是否與司機條件相符，意即乘客如果是要求包車的話，則比對司機車上是否有其他乘客，且司機是否允許包車，是的話繼續第五步，不符條件則棄掉這組組合。

第五步：過濾進階條件(是否有寵物、帶小孩上車、吸菸、司機性別是否符合要求以及乘客選擇的車種是否符合要求)，依序進行比對，如果有其中一項條件不符合，則棄掉這組配對另尋新的配對。若全部符合要求則配對成功，找出最適合的共乘。

3.2.2 共乘配對組合演算法設計

此部分為乘客與司機配對演算法，經由共乘過濾條件篩選出一些乘客之後，要更精細的將乘客直接配給某台司機的演算法。

每台乘客所規畫出來的圓應該都會大於司機所規畫出來的圓，這個時間部分在上述的共乘過濾條件已經篩選過了；所以在此定義每台乘客所規畫出來的圓所一定要完全涵蓋在裡面。

Definition：

P_j：第 j 位乘客、Di^+：第 i 個的司機起點，Di^-：第 i 個的司機終點

第一步：篩選出的乘客的起終點要和司機的起終點應該為同方向，這樣進行接送才會順路。

第二步：判斷乘客P_j是否在Di^+與Di^-所形成之直徑，且圓心座標為$(\frac{D^+之x軸座標+D^-之x軸座標}{2}, \frac{D^+之y軸座標+D^-之y軸座標}{2})$所構成的圓內，若乘客$P_j$的起終點落在圓內，則進入步驟 3；反之，本演算法則不考慮乘客P_j。

第三步：接著取司機圓內的乘客，在此限制乘客人數為四人，當司機圓內的乘客數大於四人時，便開始用距離的遠近進行挑選乘客；反之，若司機圓內的乘客數等於四人時，本演算法則直接跳出結束，不接續以下步驟。

第四步：先將第二步驟的Di^+與Di^-所形成之直徑的司機圓起終點取直線連線。

第五步: 將每個乘客的起點與上步驟所得到的司機起終點直線取垂直距離，同理，也將每個乘客的終點與第一步驟所得到的司機起終點直線取垂直距離。接著將上述的兩個值相加並列出來。

第七步：取所有乘客裡面的前四位的最小值作為該圓司機的乘客，其物理意義代表離司機的起終點直線距離為最近，為不需要繞遠路的接送路線。

3.2.3 共乘乘載順序演算法設計

司機與乘客之間的乘載順序是個重要的議題。每個乘客都有不同的需求及起終點，如何在每個乘客的時限內，不同道路皆不相同的油耗條件下以及不同的模式(最短路徑、最省油耗)下取得每個使用者都能滿意的結果，便是這個演算法討論的議題。

【共乘乘載順序演算法 Pseudo Code】

Definition：

$S_{所}$：所有節點形成的集合、$S_{穩}$：所有進入穩定狀態的節點集合、$S_{未}$：未進入穩定狀態的節點集合($S_{穩} = S_{所} - S_{未}$)。P_j：第 j 位乘客、D^+：司機起點，D^-：司機終點、n：$S_{所}$集合 node 中的所有個數。P_{ath}：確定行程所構成的陣列。$T_{emp}P_{ath}$：所有暫時構成之行程陣列。T_{x^+}：節點 x 出發時間、T_{x^-}：節點 x 最晚到達時間、$T_{x^+y^+}$：節點 x 出發時間到節點 y出發時間的時間差。Success：檢查是否符合乘客需求。矩陣$D_{[i][j]}$：根據 A*演算法與該段平均速率算出各節點間距離的時間差$T_{x^{\pm}y^{\pm}}$所形成之結果。矩陣$E_{[i][j]}$：根據 A*演算法算出各節點間距離所形成之結果。矩陣$F_{[i][j]}$：根據$E_{[i][j]}$與油耗運算式所形成之結果。

Input：

司機端(出發、終點、出發時間、最晚到達時間)，乘客(出發、終點、出發時間、最晚到達時間)以及司機行駛模式。

Output:

此司機是否符合乘客需求，若符合則加以規劃行程及計算油錢。

步驟 1：

將出發與到達的時間依排序結果設為 M，其中 M= 0~n-1，並加以定義S_x為節點 x 排序後結果所得到的 M 值，產生紀錄矩陣$C_{[i][j]}$，其中將S_x的排序結果設為列 j，同理排序司機與乘客的出發時間，將排序結果設為 N，其中 N = $0 \sim \frac{n}{2} - 1$，定義Q_x為節點 x 排序後結果所得到的 M 值再將其設為欄 i。

步驟 2:

若$S_{穩} = \{\emptyset\}$，則$S_{未} = S_{未} - \{D^+\}$，

$S_{穩} = S_{穩} \cup \{D^+\}$，查看矩陣$D_{[i][j]}$

if($T_{D^+} + D_{[D^+][P_j^+]} > T_{P_j^+}$)then for count 0 to n-1

$C_{[Q_{D^+}][\text{count}]}$=0；進入步驟 5

else

for count←0 to n-1

$C_{[Q_{D^+}][\text{count}]}$=1；

P_j^+加入P_{ath}陣列中，進入步驟 4。

步驟 3 :

$S_{穩} = \{P_j^+\} \cup S_{穩}$ 且 $S_{未} = S_{未} - \{P_j^+\}$， 若 $S_{穩} \subset \{\emptyset \cup \{D^+, P_j^+\}\}$且$S_{穩} \cap \{P_j^-\} = \emptyset$，則 for count← $S_{P_j^+} + 1$ to $S_{P_j^-}$

$C_{\left[Q_{P_j^+}\right][\text{count}]} = 2$；

for count← $S_{P_j^-} + 1$ to S_{D^-}

$C_{\left[Q_{P_j^+}\right][\text{count}]} = 1$；

重複步驟 4 直至$S_{穩}$包含所有P_j^+，之後進入步驟 5。

步驟 4 :

檢查紀錄矩陣$C_{[i][j]}$，

Success = true

for i←0 to $\frac{n}{2}$ -1

for j←0 to n-1

if($C_{[i][j]} == 0$)

then success = false；進入步驟 7

else if($C_{[i][j]} == 1$);

else

success = false；

tempi=i;

tempj=j;進入至步驟 6。

步驟 5 :

for j← tempj to n-1

if($C_{[tempi][j]} == 2$)

then Last2 = j；

if($C_{[tempi][j]} == 1$)

break；

查看記錄矩陣$E_{[i][j]}$、$D_{[i][j]}$、$F_{[i][j]}$

if(模式為最短路徑模式)then 依 dynamic programing 依序將節點存入$T_{emp}P_{ath}$，若$T_{P_j^+} + E_{\left[S_{P_j^+}\right][S_{P_i^-}]} < T_{P_j^+P_i^-}$，令$C_{[tempi][j]}$=0；反之將$C_{[tempi][j]}$令為 1，其中 j= tempj~Last2。

if(模式為最省油耗模式)then 在此使用最省油耗演算法，再依 dynamic programing 依序將節點存入$T_{emp}P_{ath}$，若$T_{P_j^+} + E_{\left[S_{P_j^+}\right][S_{P_i^-}]} < T_{P_j^+P_i^-}$，令$C_{[tempi][j]}$=0；反之將$C_{[tempi][j]}$令為 1，其中 j= tempj~Last2。

Success = true，回到步驟 6。

步驟 6 :

if(Success = false) then 司機不符乘客需求

if(Success = true)then 司機符合乘客需求，

$S_{穩} = \{P_j^-\} \cup S_{穩}$；$S_{未} = S_{未} - \{P_j^-\}$，

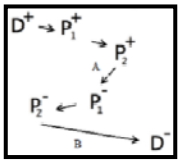
依時間先後合併每個 T_emp P_ath 形成 P_ath，並依此算出分攤的油錢，詳見油錢分攤演算法。

共乘乘載順序演算法首先是接續共乘配對組合演算法，將此集合內所有時間點進行排序，將結果產生紀錄矩陣$C_{[i][j]}$ (如表 3)，此紀錄矩陣橫軸表所有時間點的排序，縱軸為司機乘客的出發時間點的排序。反覆檢查紀錄矩陣$C_{[i][j]}$，若有任一值為 0，表示司機不符合乘客需求，則不考慮此乘客。若紀錄矩陣$C_{[i][j]}$中的值皆為 1 的情況，表示配對成功。但紀錄矩陣中$C_{[i][j]}$有值為 2，表示不穩定狀態，須進行修正直至所有$C_{[i][j]}$的值不為 2 的狀態。

開始修正時，會依照使用者選擇之模式(最短時間或最省油耗模式)有不同的路徑規劃（如圖 7、8 與表 4、5）。以圖 7 與圖 8 為例，若有一條共乘路徑，且初始時使用者選擇最短路徑模式，A 加 B 路段距離小於 C 加 E 路段距離，在此演算法會紀錄此路段排序方式，後將$C_{[i][j]}$的值修正為 1，若找不到任一解，則將$C_{[i][j]}$的值修正為 0；同理，若初始時使用者選擇最省油耗模式，C 加 E 路段的油耗少於 A 加 B 路段則演算法會紀錄此路段排序方式，並對$C_{[i][j]}$進行修正。

表 3　$C_{[i][j]}$紀錄矩陣(+表示起點，-表示終點)

	D^+	P_2^+	P_1^+	P_2^-	P_1^-	D^-
D^+	1	1	1	1	1	1
P_2^+	1	1	2	2	1	1
P_1^+	1	1	1	2	2	1



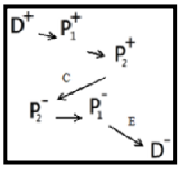


圖 7　最短路徑例子　圖 8　最省油耗例子圖

表 4　各點之間距離的距離表(m)

	D^+	P_2^+	P_1^+	P_2^-	P_1^-	D^-
D^+	0	100	300	400	500	600
P_2^+		0	100	150	300	400
P_1^+			0	20	50	200
P_2^-				0	20	150
P_1^-					0	200
D^-						0

表 5　各點之間單趟所需時間表(min)

	D^+	P_2^+	P_1^+	P_2^-	P_1^-	D^-
D^+	0	12	22	30	37	47
P_2^+		0	15	26	39	50
P_1^+			0	14	26	40
P_2^-				0	10	20
P_1^-					0	8
D^-						0

3.2.4 共乘油費的分攤演算法設計

針對共乘油費的分攤，我們設計出較為公平的機制：依據共乘路徑規劃演算法將會得出最適合的拜訪順序，加以計算共乘的路段分攤油費。Money{D,p1,p2….}為付油費的集合，裡面的元素為乘客 ID 和需付金額，本集合需動態加入共乘人。

步驟 1：司機出發時，將司機加入集合 Money，並記錄載到下個點之前花費的金錢(如表 6)。

步驟 2：載到第一個乘客P_1時，將 Money 集合中的司機部分更新為步驟 1 最後金錢的紀錄並將 p1(passenger 1)加入集合 Money(如表 6)。

步驟 3：重新記錄載到下一個目的點之前需花費的金錢(如表 7)。

步驟 4：當司機到下一個目的點時，將步驟 3 所計算的金錢/ Money 的元素個數後，對存在集合 Money 中所有元素需付的金額進行更新(如表 7)。

步驟 5：檢查目的點是否有人下車，若有則該下車的人需付金額即為集合 Money 中對應乘客 ID

的金額，付清後將從集合 Money 中刪除(如表 8)。

步驟 6：檢查目地點是否有人上車，若有則將集合 Money 加入資訊(該乘客 ID，0)。

步驟 7：檢查司機下個目的點是否存在，若否則抵達終點。若是則回步驟 3。最終油錢(如表 9) 。

表 6　油費分攤演算法步驟 1 及步驟 2

$D^+ P1^+$

	30					total
D	30					
P1						
-						
-						

目前路程:30 元　單位:新台幣

表 7　油費分攤演算法步驟 3 及步驟 4

$D^+ P1^+ P3^+$

	30	20				total
D	30	20/2=10				
P1		20/2=10				
-						
P3						

目前路程:50 元　單位:新台幣

表 8　油費分攤演算法步驟 5

$D^+ P1^+ P3^+ P3^-$

	30	20	30			total
D	30	10	30/3=10			
P1		10	30/3=10			
-						
P3			30/3=10			10

目前路程:80 元　單位:新台幣

表 9　司機及各乘客所需支付油錢表

	30	20	30	40	30	20	40	total
D	30	10	10	20	10	10	40	130
P1		10	10	20	10	10		50
P2					10			20
P3			10					10

全部路程：210 元　單位：新台幣

4. 環境建置與程式實作

兩台以上搭載 Google Android 4.0 版之 Acer A500 平板電腦作為用戶端安裝設計的共乘系統。桌上型電腦作 Server 端執行共乘配對演算法並使平板電腦透過無線網路傳送資料，透過平板上 GPS 接收器紀錄行駛路徑及收集道路交通資訊。

開發工具：eclipse + ADT、Android 4.1 + JAVA 1.7。

所需設備：桌上型電腦一台、平板電腦兩台以上、802.11 無線寬頻路由器一台。

本計畫在 Android 系統運用 APP 呈現畫面：行程資訊(如圖 9)，可設定出發與到達的時間和地點(如圖 10)。顯示油錢及路徑規劃(如圖 11、12)。

結論與未來工作

未來將考慮「動態調整共乘配對功能」與「訊息傳遞的安全性」，特別是結合車輛網路傳送的訊息。如何運用車輛網路資訊「動態調整共乘配對」將是未來的主要研究方向及目標。

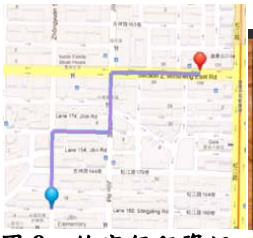


圖 9　給定行程資訊

圖 10　設定行程資訊

圖 11 配對後油錢資訊

圖 12　配對後的路線圖

參考文獻

[1] ET 新聞雲(共乘新聞)：http://www.ettoday.net/news/20130127/157698.htm

[2] Ing-Chau Chang, Hung-Ta Tai, Feng-Han Yeh, Dung-Lin Hsieh, and Siao-Hui Chang, "A VANET-Based A* Route Planning Algorithm for Travelling Time-and Energy-Efficient GPS Navigation App," *Hindawi Publishing Corporation International Journal of Distributed Sensor Networks Volume* , 24 June , 2013.

[3] 速度與油耗：車速多少油耗最低最經濟：http://yxk.cn.yahoo.com/articles/20100613/3ch5.html, 2010.

[4] Manel Sghaier, Hayfa Zgaya, Slim Hammadi, Christian Tahon, "A Novel Approach Based on A Distributed Dynamic Graph Modeling Set Up OverASubdivision Process to Deal With Distributed Optimized Real Time Carpooling Requests," *The 14th International IEEE Conference on Intelligent Transportation Systems,Washington, DC, USA, pp.1311-1316,* October 5-7, 2011.

[5] George Dimitra kopoulos, Panagiotis Demestichas,Vera Koutra, "Intelligent Management Functionality for Improving Transportation Efficiency by Means of the Car Pooling Concept," *IEEE Transactions on Intelligent Transportation Systems,Vol. 13, No. 2, pp. 424-436,* JUNE 2012.

[6] Yong Ge, HuiXiong, Alexander Tuzhilin, KeliXiao, Marco Gruteser, Michael J. Pazzani, "An Energy-Efficient Mobile Recommender System," *international conference of ACM SIGKDD, pp.899-908*, 2010.

[7] 桃北北基共乘網：http://carpool.ntpc.gov.tw/carpool/.

[8] 共乘王：http://www.carpoolking.com/tw/zh-hk/.

[9] BlueNet：https://play.google.com/store/apps/details?id=com.BlueNetPlay.CarpoolActivity