

國立交通大學

資訊科學與工程研究所

碩士論文

在車用隨意網路中
以交易代幣為基礎之叢集化方法

Token-Based Clustering Method
in Vehicular Ad-Hoc Network

研究生：毛泰源

指導教授：孫春在教授

中華民國九十九年六月

在車用隨意網路中以交易代幣為基礎之叢集化方法

Token-Based Clustering Method in Vehicular Ad-Hoc Network

研究生：毛泰源

Student：Tai-Yuan Mao

指導教授：孫春在

Advisor：Chuen-Tsai Sun

國立交通大學

資訊科學與工程研究所

碩士論文

A Thesis

Submitted to Institute of Computer Science and Engineering

College of Computer Science

National Chiao Tung University

in partial Fulfillment of the Requirements

for the Degree of

Master

in

Computer Science

June 2010

Hsinchu, Taiwan, Republic of China

中華民國九十九年六月

在車用隨意網路中以交易代幣為基礎之叢集化方法

學生：毛泰源

指導教授：孫春在博士

國立交通大學資訊科學與工程研究所

中文摘要

由於車用隨意網路的發展，駕駛人已經可以透過車上裝置與其他車輛或是路側裝置分享路況資訊，作為行車時選擇道路的參考依據。在市區或是重要幹道中，由於車輛密度較高，若是單純使用廣播方式傳遞路況資訊，容易造成廣播風暴問題，而建立叢集(Cluster)成為了一種常見的解決方式。

一般的叢集化方法都希望最小化管理叢集的負擔，但此類方法在叢集化時只參考車輛 ID 或連接數，在 VANET (Vehicular Ad-Hoc Network) 中仍有改進空間。本研究以最少叢集改變(Least Cluster Change, LCC)演算法為基礎，提出了交易代幣式叢集化(Token-Based Clustering, TBC)演算法，將車輛地理位置與資訊更新時間數值化成代幣(Token)，並讓車輛藉由代幣交易機制購買路況資訊，讓擁有最多代幣的車輛將成為叢集管理者(Cluster Head)。車輛付出代幣向臨近的車輛購買路況資訊，並依買方擁有的代幣數量與賣方資訊的更新時間計算交易價格。由於處於重要地理位置或已擁有較新路況資訊的車輛容易累積較多代幣，所以適合成為叢集管理者。

本研究以細胞自動機(Cellular Automata)建立交通環境模型，並比較在車輛密度與比例不同時，使用 LCC 演算法與 TBC 演算法的差異。在實驗中，發現 TBC 演算法在車輛密度與比例較高時，較能降低環境中的叢集數量，讓叢集中的成員數提高，並減少訊息傳遞的長度。

關鍵字：車用隨意網路、動態導航、代幣、叢集、最少叢集改變演算法

Token-Based Clustering Method in Vehicular Ad-Hoc Network

Student: Tai-Yuan Mao

Advisor: Dr. Chuen-Tsai Sun

Institute of Computer Science and Engineering

National ChiaoTung University

Abstract

With the development of vehicular ad-hoc network (VANET), drivers can share traffic information through inter-vehicle communication and take traffic information into their driving consideration. Because of the higher vehicle density in urban area, broadcasting traffic information to other vehicles will cause the broadcast storm problem. For the problem, clustering is a common solution.

Reducing the overhead in cluster management is important for general clustering methods. Based on least cluster change (LCC) algorithm, the study proposes a token-based clustering (TBC) algorithm. Being different from existing clustering algorithms, in our model vehicles pay tokens to buy traffic information and the price of the information is calculated based on update time and vehicle location. The vehicles at important locations or with latest traffic information are easier to accumulate tokens, so the vehicles are more suitable to be cluster heads than the other vehicles.

The study uses cellular automata (CA) to construct a simulation traffic environment and analyzes different cases with varied traffic densities and proportions of vehicles. Experimental results show that the TBC algorithm can reduce the number of clusters and decrease the number of packet hops.

Keywords: VANET, Dynamic Navigation, Cluster, Token, LCC Algorithm

誌謝

兩年過去了，回想從當初決定要繼續讀研究所開始，從開始面對修課的繁忙與論文研究的訓練，直到最後總算完成了研究所學業，這一路上遇到了許多困難與挑戰，但是也很高興認識了許多老師、同學與朋友，總是在我需要幫忙時適時伸出援手，並且也提醒了我很多不足與需要改進之處。

首先要感謝我的指導老師，孫春在老師，在身為教授的事務忙碌之餘，還是盡力對每一個同學的論文詳加指導。孫老師不但歡迎學生們主動找老師討論任何想法，廣博的見識與專業的素養更使我在求學過程中受益良多。其次，我也相當感謝黃崇源老師的教導，從論文方向的訂定至研究結果的細節，黃老師都不厭其煩地與我們討論，並且將我們不成熟的想法引導成一個明確的研究方向。在論文口試過程中，也感謝曾憲雄老師、胡毓志老師與陳穎平老師的建議與指導，嚴謹與仔細地審視我的論文，也提出了許多研究方向上的建議與一些思考不足之處。

除了許多老師的指導以外，也感謝許多學長與同學們的幫助，感謝宇軒、聖文、王豪、宜叡、基成與鵬羽學長的指導，花了學長們許多時間讓我詢問與請教。感謝一起努力打拼互相幫助的同學全榮、壯為、嘉宏、柏志、璽文、謹譽以及幫忙分擔實驗室雜務的學弟妹們。

最後，謝謝我的家人，在我就讀研究所學業的同時，從背後支持我，讓我沒有後顧之憂，能專心在學業上。

泰源 謹致

2008.07

目 錄

中文摘要.....	i
Abstract.....	ii
誌謝.....	iii
目 錄.....	iv
表 目 錄.....	vi
圖 目 錄.....	vii
第一章 緒論.....	1
1.1 研究動機.....	1
1.2 研究背景.....	5
1.3 研究目標.....	6
1.4 研究重要性.....	8
1.5 論文架構.....	10
第二章 文獻探討.....	12
2.1 車間無線通訊技術.....	12
2.1.1 無線隨意網路相關研究探討.....	12
2.1.2 泛流機制相關研究探討.....	13
2.1.3 Vehicular Ad-Hoc Network (VANET)相關研究探討.....	17
2.2 VANET動態路徑導航之封包傳送演算法.....	18
2.3 效能評估機制.....	22
第三章 研究方法.....	23
3.1 實驗設計.....	24
3.1.1 封包內容.....	24
3.1.2 參數設定.....	26
3.1.3 模擬交通模型.....	28
3.1.4 模擬程式流程.....	34
3.2 交易代幣式叢集化(Token-Based Clustering, TBC)演算法.....	35
3.2.1 TBC演算法概念與設計.....	35
3.2.2 初始化.....	37
3.2.3 叢集調整方法.....	38
3.2.4 選擇封包傳遞對象.....	40
3.2.5 代幣交易機制.....	41
3.3 效能評估方法.....	42
3.3.1 效能評估指標種類.....	42
3.3.2 評估指標蒐集方式.....	43
第四章 研究發現與分析.....	45
4.1 模擬交通環境與參數.....	45

4.2 實驗結果.....	48
4.2.1 平均叢集數目.....	49
4.2.2 叢集調整次數.....	52
4.2.3 平均封包傳遞時間.....	56
第五章 結論與貢獻.....	59
5.1 研究結果綜合分析.....	59
5.2 研究結論.....	60
5.3 研究貢獻.....	61
5.4 未來展望.....	62
5.4.1 模擬交通模型.....	62
5.4.2 演算法調整.....	63
5.4.3 代幣檢驗機制.....	63
參考文獻.....	65



表 目 錄

表 1. 實驗參數表.....	46
表 2. 研究結果綜合整理.....	59



圖目錄

圖 1. ON-BOARD UNIT(OBU)與ROADSIDE UNIT(RSU)之網路傳遞關係圖.....	2
圖 2. 交通中的叢集架構概念圖.....	4
圖 3. 以概念相似性之泛流機制分類.....	13
圖 4. 鄰近節點機制與資訊散佈關係.....	16
圖 5. 叢集架構示意圖.....	17
圖 6. 連接導向之無線隨意網路ERPC演算法.....	19
圖 7. 連接導向之無線隨意網路ERPC演算法.....	19
圖 8. GPSR演算法之網路形成過程.....	20
圖 9. GPSR演算法之網路簡化過程.....	21
圖 10. 在叢集中車輛傳遞資訊之範例.....	23
圖 11. 封包內容.....	24
圖 12. 車流量計算方式.....	27
圖 13. 模擬交通模型架構圖.....	28
圖 14. 以細胞自動機建立之交通環境、道路與車輛圖.....	31
圖 15. 以細胞自動機為基礎之網格狀交通模型.....	31
圖 16. 配置車輛後的網格狀交通模型.....	32
圖 17. 應用在交通模擬之細胞自動機變化規則.....	32
圖 18. 一維細胞自動機可能遇到之所有情形.....	33
圖 19. 每回合進行之實驗模擬流程.....	34
圖 20. TBC演算法架構.....	36
圖 21. 平均叢集數目(低密度).....	49
圖 22. 平均叢集數目(中密度).....	50
圖 23. 平均叢集數目(高密度).....	50
圖 24. 叢集調整次數(低密度).....	53
圖 25. 叢集調整次數(中密度).....	53
圖 26. 叢集調整次數(高密度).....	54
圖 27. 平均封包傳遞時間(低密度).....	56
圖 28. 平均封包傳遞時間(中密度).....	57
圖 29. 平均封包傳遞時間(高密度).....	57

第一章 緒論

1.1 研究動機

當代交通與人們生活息息相關，隨著不斷增長的車輛數目、不停拓展的交通路網及與時俱進的交通法規，交通運作過程中所產生的資料量與複雜度已超越人力所能處理，而發展出的交通科學也成為了重要的研究領域。

隨著資通訊設備與技術的進步與發展，行動式隨意網路(Mobile Ad-Hoc Network, MANET)的技術已經日益成熟，而使用在車輛上的 MANET 技術便被稱為車用隨意網路(Vehicular Ad-Hoc Network, VANET)。人們已經可以藉由車輛上裝置的車上設備(On Board Unit, OBU)或是道路上的路側設備(Roadside Unit, RSU)收集行車間的數據與各地的交通狀況，直接進行處理或是傳送給遠端的電腦建立交通模型，並且用以評估與分析交通狀況 (Vahdat & Becker, 2000)。

VANET 架構中，無線網路的通訊方式依溝通雙方設備之類別，主要可分為 RVC (Roadside-to-Vehicle Communication)、IVC(Inter-Vehicle Communication)與混合 RVC 及 IVC 兩者等三種。VANET 使用者可以在具有 RSU 的道路上，使用 OBU 向 RSU 取得附近停車場的位置與剩餘車位數量，也可以對從對向車道來的車輛詢問前方道路的路況，成為了提供各種進階服務發展的基礎技術，圖 1 即為 OBU 與 RSU 之網路傳遞關係圖。

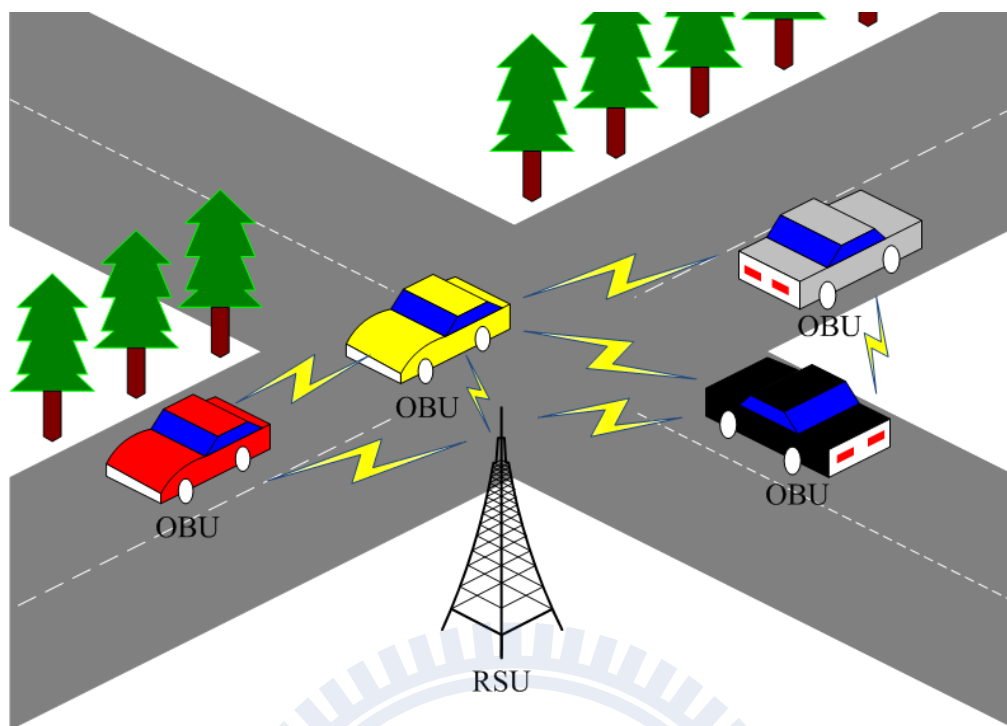


圖 1. On-Board Unit(OBU)與 Roadside Unit(RSU)之網路傳遞關係圖

在以 RVC 為主的 VANET 架構下，先前的研究主要都是著重於當車輛在交通路網中行駛時，判斷何時可以將目前正在通訊的 RSU 切換成另一個車輛正在靠近 RSU，而仍然保持通訊不中斷 (Bechler & Wolf, 2005; Ghassemian, Friderikos, & Aghvami, 2005; Little & Agarwal, 2005)。而在以 IVC 為主的 VANET 架構方面，由於車輛間的網路拓撲會快速的變動，如何選擇傳遞訊息的路由方式便成為 IVC 領域的主要議題 (Little & Agarwal, 2005; Nadeem, Shankar, & Iftode, 2006)。

VANET 技術的發展提供了許多應用的可能性，其中一種重要的應用服務就是車輛導航系統。駕駛人將車輛目的地輸入 OBU，透過地理資訊系統(Geographic Information Systems, GIS)取得地圖與道路資訊，再經由全球定位系統(Global Positioning Systems, GPS)取得車輛在地球上的位置，並判斷目前車輛正在行駛的道路，再作出車輛路徑規劃，導引駕駛人行駛最短的路徑。駕駛人不需要熟悉出發的或目的地附近的道路，也可以輕易到達目的地 (Skog & Handel, 2009)。

然而，有了良好的車輛導航系統，卻也產生了新的問題。首先，由於車輛導

航系統大部份採用以靜態的道路長度作為權重之最短路徑演算法，容易將起點與終點相似的車輛導引到同一條道路上，再加上所規劃出的路線往往會是大部分駕駛人經常選擇行駛的路線，車輛密度通常都會高於一般道路。其次，在規劃路徑時，車輛導航系統並沒有將車速、車流量以及車流密度納入考慮，可能會將車輛導引到行駛距離較短但卻壅塞不堪的路線上，造成行駛效率低落。當駕駛人發現以上情形時，往往已經陷入其中而難以脫離了。

為了避免發生此種情形，許多研究都提出了以即時路況資訊進行路線規劃的動態路徑規劃 (Karp & Kung, 2000; Skog & Handel, 2009)。由 OBU 取得自身車輛行駛速度與方向等行駛狀態，再透過車間無線網路與不同車輛或 RSU 交換行駛狀態訊息與路況訊息，取得車輛周圍道路以及往目的地之道路的行車路況資訊作為路徑規劃時的參考因素，以得到更順暢有效率的導航結果。

儘管動態路徑規劃已經改善了傳統靜態導航系統的許多缺點，但由於交通是具有高度動態的複雜系統，所以如果要在市區中使用，仍然有許多困難處需要克服 (Blum, Eskandarian, & Hof, 2004)。由於在 VANET 中最基本的傳遞資訊方式就是廣播，再加上在市區中，一段道路上可能會有數以千計的車輛，對一輛車來說，在 VANET 的可傳輸範圍中，可能也會有數百台車輛可以與自己傳遞訊息，如此高密度的節點之無線網路環境造成的密集資料傳輸，使得資料容易產生碰撞，造成資料遺漏或損壞，造成需要重傳更多次的封包以彌補先前發送的封包，產生廣播風暴問題(Broadcast Storm Problem)(Tseng, Ni, Chen, & Sheu, 2002)，使得網路的有效利用率大幅降低。

過去有關於無線網路的研究中，提出了許多關於廣播風暴問題的解決方法，其中一種主要的解決方式是產生叢集(Cluster) (Basu, Khan, & Little, 2001; Yu & Chong, 2005)。將一群車輛組成叢集，並由叢集管理者(Cluster Head)之車輛負責向叢集外要求資料或是回應叢集內的查詢路況資訊之要求，使得封包傳送量與傳

送路徑長度都可以大幅減少。使用叢集實作的車用無線網路概念如下圖 2，以叢集管理者為中心，每一個叢集領導者就代表了一個叢集，其他車輛則被稱為叢集成員(Cluster Member)，而連接到兩個叢集的叢集成員，也被稱為叢集閘道(Cluster Gateway)。

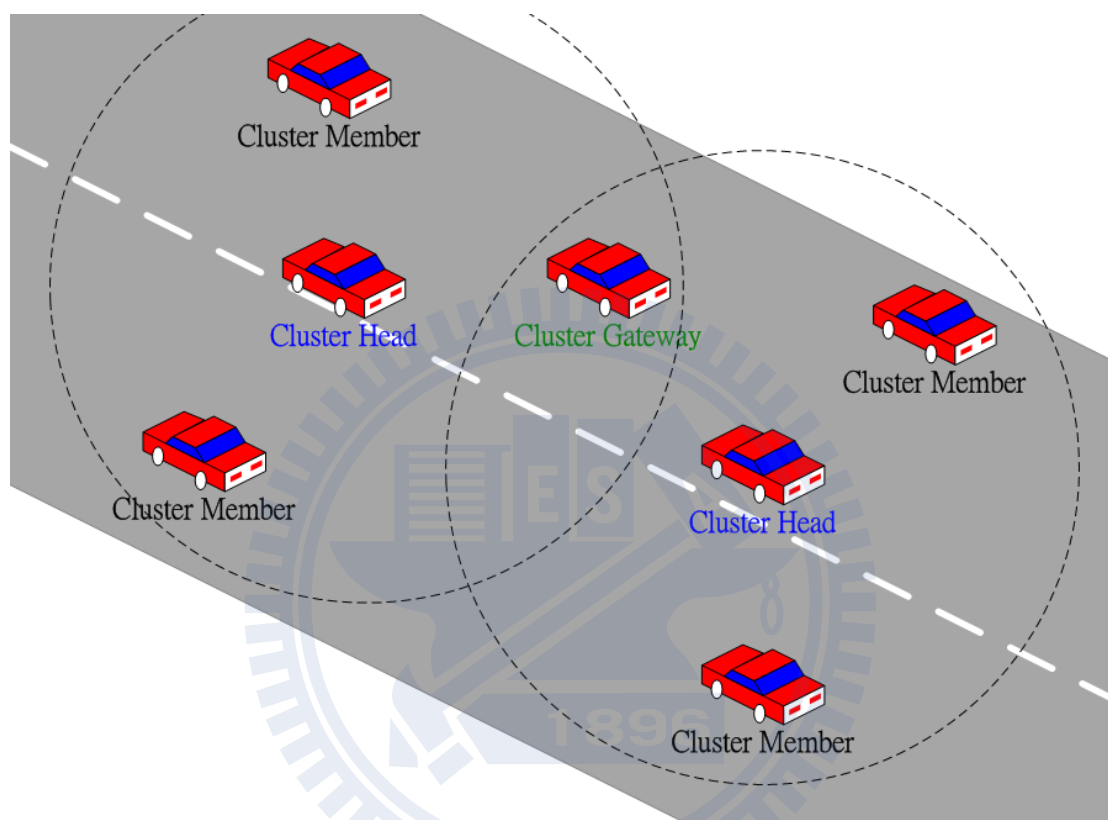


圖 2. 交通中的叢集架構概念圖

雖然使用叢集化演算法作為網路基礎架構將會具有以上所提到之優點，但由於目前對於叢集化演算法(Clustering Algorithm)的研究所提出之方法大多是經由靜態車輛資訊、基本網路拓樸特性或針對如高速公路等特殊道路情況產生叢集管理者，如選擇鄰近區域中 OBU 之 ID 最小車輛、節點分支度(Degree)最高車輛與位置最接近其他叢集之車輛等 (Basu, Khan, & Little, 2001)，而不是依實際資料傳輸需求而選擇代表車輛。經由此類方法所產生的代表車輛，由於選擇方式過於簡單，也缺少將地理位置與實際的資訊傳播情形加入考慮，不容易適應各種變化的交通狀況。

因為以上所提之各種原因，本研究將提出一個依實際傳輸資料的需求並加以分析，能依各種路況資訊傳遞情形動態形成與調整叢集之演算法，稱為交易代幣式叢集化(Token-Based Clustering, TBC)演算法。

1.2 研究背景

近年來，智慧型運輸系統(Intelligent Transportation System, ITS)在世界各地有與多的相關應用。以美國運輸部的 ITS (Intelligent Transportation System)計畫為例，便是將 IEEE 802.11p 分成兩個主要部分：智慧型基礎建設(Intelligent Infrastructure, IF)與智慧型車輛(Intelligent Vehicles, IV)，並以此兩種技術為中心，推出一系列的相關應用 (Intelligent Transportation, 2010)。而在世界上其他地區，也對於 IEEE 802.11p 的發展有著各自的計畫，如歐盟所推行的 SAFESPOT 計畫(Safespot, 2010)與日本所推行的 DSRC (Dedicated Short Range Communication)計畫，都是目前正在進行的 IEEE 802.11p 應用計畫。

IEEE 以具有行動隨意網路(Mobile Ad-Hoc Network, MANET)能力之 IEEE 802.11 為基礎，在 2004 年對車用無線網路制定了 IEEE 802.11p 通訊標準，主要目的為支援智慧型運輸系統的相關應用，提升運輸系統之效率、便利性與安全性所制定的無線區域網路通訊協定。

由於智慧型運輸系統的研究與發展，車輛已經可以藉由 RVC、IVC 或混合使用 RVC 與 IVC 等三種方式交換路況資訊，提供了動態路徑規劃的能力。而在動態路徑規劃中，MANET 技術的發展扮演了相當重要的角色。MANET 不需要有線基礎建設支持，所以被大量使用在車間無線區域網路上，如 VANET 便是專門的車用無線隨意網路之通訊協定。

隨著在 VANET 中動態路徑規劃的發展，許多動態路徑規劃的相關研究不斷被提出，而這些研究主要都著重於如何挑選傳遞目標與路徑，讓封包能順利且有效率地傳送往目的地再送回，如以隨機車輛配對的方式將封包傳遞的成功率最大化並將資源消耗與延遲時間最小化 (Vahdat & Becker, 2000)與以車輛的位置與目標車輛的方向的封包傳遞貪婪演算法(Greedy Algorithm) (Karp & Kung, 2000)。Zhao 與 Cao 基於 Karp 與 Kung 的研究，提出了在考慮車輛的相對位置與方向下比較封包傳遞的決策方式，並提出了混合型的決策方式 (Zhao & Cao, 2008)。

在過去的研究中，主要著重於傳播資訊時如何選擇一個目標或是一條路徑，讓資料能最快到達目的地，但是當車輛發出查詢路況的要求時，也只能將此要求盡快送往該條道路，隨著被查詢路況的道路距離越遠，傳輸距離也會使傳輸時間增加，造成使用上的不便，而且隨著大量車輛發出查詢長距離路況資訊之請求，更容易造成大量封包在網路上傳遞，造成使用率下降。

本研究提出之 TBC 演算法可以在區域中選擇一部叢集之代表車輛，稱為叢集管理者，負責替叢集內車輛對外查詢交通資訊並保存在自身系統中，當車輛需要查詢路況時，可先將要求傳送到鄰近的叢集代表車輛中，若此代表車輛已經擁有該條道路最近的路況資訊，則可直接就近回應給發出查詢之車輛，由統一的管
理者負責處理訊息，節省大量封包傳遞時間與數量，同時可以大量減低廣播風暴問題產生的機會及其影響。

1.3 研究目標

本論文之研究問題為針對將使用在 VANET 環境中，用以解決廣播風暴問題之叢集概念。在原有的叢集概念中，考量車輛實際資料傳輸情形後，讓此叢集選擇代表車輛的方式具有適應性，即可以針對不同的環境與資料傳輸情形做出調整，用以改進傳統缺乏適應環境能力之演算法 (Yu & Chong, 2005)。

本研究的目標為基於 VANET 所提供之網路通訊架構與功能，提出將一群行駛中的車輛，依其車輛所處地理位置與路況資訊更新時間等實際資料傳輸的情形，將一群車輛組合成叢集，並選出叢集代表車輛之演算法，稱為交易代幣式叢集化 (Token-Based Clustering, TBC) 演算法。

本演算法之核心概念為商業行為中常見的市場機制。車輛在行駛中使用代幣 (Token) 向鄰近車輛購買路況資訊，並依購買到之路況資訊準確度與時效性決定給與代幣之數量。當車輛開始行駛並加入整體車間無線網路時，系統會自動產生若干個代幣給自身車輛，當車輛向某目標車輛發出查詢路況的請求時，會同時給予部分自身的代幣給此車輛，因此經過多個回合後，最常被查詢的車輛將會累積最多代幣，而持續讓大部分車輛都選擇這輛車輛當作傳遞訊息的目標，這時便將這台代幣累積最多之車輛稱為叢集管理者，而其他叢集中的車輛稱為叢集成員，以一個叢集管理者為中心，與其鄰近的叢集成員形成一個叢集。

部分位於交通樞紐位置的車輛，或是在密集車輛群體中心的車輛，由於較容易收到附近車輛的查詢路況請求，再加上最多被查詢最多次且能提供最準確的資料之車輛必然會累積最多代幣，故可使部分此類車輛擁有較多的代幣，讓擁有最多代幣之車輛成為此叢集之管理者。

如此以實際資訊交換行為作為根據，所自然產生之叢集管理者，在網路傳輸情形發生變化變化時，如從上班時間至下班時間之車流移動方向轉移，或是因週末假日時往遊樂區之道路大量擁塞，皆可以依演算法，由改變的路況資訊傳遞情形決定新的叢集管理者以適應新的路況。

1.4 研究重要性

系統負擔

在 VANET 中，由於電腦與通訊系統等 OBU 必須裝置在車輛的有限空間中，無法輕易的修改或加裝硬體，效能也未必能應付日漸增加的 VANET 服務所需。在能源供應方面，OBU 也需要由車輛提供電力，無法隨意使用效能較高但耗電量亦高的裝置。由於 VANET 是採用無線方式傳遞封包，在快速移動的車輛之間傳遞，硬體需花費一定程度的額外資源在處理封包不完整、碰撞與遺失等問題。由以上數點可知，若要使用 VANET 中的各種服務，在硬體方面將會受到相當大的限制，若是可從軟體面加以輔助，將可以提升系統的能力與民眾的接受程度。

叢集化演算法常被使用於分散式系統中，目的是在處於各個節點之系統只知道區域資料的情形下，可以共同形成一個叢集，以增加管理效率與減少傳遞資訊負擔。若是將叢集化的概念應用於 VANET 中，以每一台車輛為一個節點，建立叢集與選出管理者，由管理者統一處理與分配資訊，可以減低許多額外封包的傳遞量與負擔，如處理封包不完整、碰撞與遺失等。

在各種叢集化演算法中，由於通常只以單一數值作為叢集化時的參考依據，如最小叢集改變(Least Cluster Change, LCC)演算法中，可分為以設備 ID 作為叢集化依據的最小 ID 叢集化(Lowest ID Clustering, LIC)，以及使用節點分支度作為叢集化依據的最高連接度叢集化(Highest Connectivity Clustering, HCC)等兩種方法(Chiang, Wu, Liu, & Mario, 1997; Ephremides, Wieselthier, & Baker, 1987; Gerla & Tsai, 1995)，也因此使得不同種類的叢集化方法容易互相整合，故相容性極高。

舉例而言，若是目前在環境中主要以 LIC 演算法進行叢集化，而想在此環境中加入使用 HCC 演算法進行叢集化的節點時，使用 HCC 演算法節點在交換

訊息時，只需將使用 LIC 演算法節點發送之資訊中所記錄之 ID 值加以轉換，如下列公式：

$$\text{Transformed_ID} = \frac{\text{Original_ID} \times \text{Average_Degree}}{\text{Average_ID}}$$

將此 ID 除以節點 ID 平均值後再乘以連接度平均值，讓此 ID 值經過轉換後會接近節點連接度大小範圍，即可進行較準確的比較。

駕駛人

對於駕駛人而言，一個有效率的動態路徑導航能大幅提升交通的效率，更快速與更順暢的到達目的地，也能讓駕駛人減少需要待在車陣中等待阻塞退去的情形，提升駕駛過程中的舒適度。若採用好的叢集化演算法所產生的動態路徑導航，不但可以減低 OBU 的負擔，也因為行駛效率的提高，駕駛人不但可以省下了許多時間，也減少了許多燃料的消耗，即能節省部分花在交通上的開銷，也可以對駕駛人增加安裝 OBU 的吸引力。同時，更多的駕駛人安裝 OBU，路徑導航的資料來源就更多，也能提供更準確與詳細的資料，產生互利。

汽車廠商

由於本演算法使用叢集概念，由原本所有車輛之間的點對點廣播，改變成僅有叢集成員對叢集管理者的溝通，可以大量降低封包傳送量，也同時減低廣播風暴問題的產生機會，讓 OBU 省下大量重送封包的必要，OBU 才能分配更多的資源處理收到的資訊或是其餘的系統工作，讓整體系統之效能提高，也不需要花額外的開銷，讓較低效能的硬體裝置也能發揮不錯的效能。除此以外，對汽車廠商而言，若是採用 TBC 演算法，也只需在原本已經安裝的 OBU 硬體上安裝新的軟體與演算法，不需要更改或升級硬體，實行的成本相對其他方法較低。

政府

一般而言，要減低廣播風暴問題，一種方法就是在程式的各個道路建置大量的無線網路基礎設施以擔任資料傳遞中心，也就是 RSU。當車輛行駛接近資料中心時，便與此資料中心溝通，取得或是回報附近的路況資訊，並取代原先與其他車輛溝通以取得資料的方式。由於此種資料中心屬於 RSU，可以直接使用運算較快的硬體設施，而不需要擔心耗電量、體積與重量的問題。

但 VANET 的通訊範圍廣大，覆蓋範圍往往可涵蓋整個城市，若要建立足夠數量的資料中心以涵蓋整個城市，花費將會相當的龐大，也會需要額外的經費與人力作管理與維修。如果改為使用 TBC 演算法所產生的叢集管理者代替資料中心的角色，不但可以節省下要花費在資料中心上的經費與人力，也因為網路中擁有大量的車輛可以擔任資料中心，當一台叢集管理者行駛離開網路或是 OBU 故障時，可以直接由附近的另一台車輛接替其工作，成為叢集管理者，增加整體的可用性與穩定性。

另一方面而言，若政府推動使用動態路徑規劃的車輛做為日常的交通工具，讓各個駕駛人都可以選擇最不阻塞的道路行駛，加快行駛速度與縮短行駛時間，可以節省大量的自然資源，也可以維護自然環境，減少自然資源消耗。

1.5 論文架構

本論文將會建立一個真實路網的簡化模型，探討加入 TBC 演算法後之叢集負擔與封包傳遞效能，並與最少叢集改變(Least Cluster Change, LCC)演算法進行比較。在第二章中將會對相關文獻進行探討與分析，包括無線網路的發展技術、在交通上的應用與過去處理動態導航系統的方式。第三章為研究方法，包括模型設計、參數設定、程式流程、TBC 演算法設計與效率評估方式。第四章為研究

的結果，如叢集負擔與封包傳遞等項目，進行分析與探討，而第五章為本研究之綜合分析、研究結論、研究貢獻與未來展望。



第二章 文獻探討

2.1 車間無線通訊技術

2.1.1 無線隨意網路相關研究探討

無線隨意網路(Ad-Hoc)是一種點對點的傳輸模式，提供在設備間可直接進行網路溝通的能力，而不須有線的基础建設支持。而採用 IEEE 802.11p 之 CDMA/CA 機制，更進一步標準化車間(Vehicle-to-Vehicle, V2V)的無線區域網路，傳輸距離可達數百公尺，所以在世界各國產生多種的應用。美國的 ITS (Intelligent Transportation System) (Intelligent Transportation, 2010)、歐盟的 SAFESPOT 計畫 (Safespot, 2010)與日本的 DSRC (Dedicated Short Range Communication)計畫，都是目前正在進行的車用無線隨意網路之應用計畫。

IEEE 802.11p 架構提供車輛間的無線區域網路溝通的能力，讓 OBU 在行駛過程中能蒐集各段道路的行車資訊，如車速、車流量與車流密度等，再經由車間無線區域網路在車輛間交換資訊，讓 OBU 能得到更多來自不同車輛所提供的即時道路資料，進而對當時路況做出最適合的路徑規劃。

由於無線隨意網路並非所有節點都是固定連接的，所以經由建立虛擬軌道的方式在非固定連接的網路中傳遞封包之演算法被提出 (Li & Rus, 2000)。在此演算法中，Li 與 Rus 假設了兩個前提：1.所有存在於網路中的節點，其任何動作皆為已知 2.若此節點為發送傳遞封包訊息的節點，則該節點動作則為未知，無法確定，此兩前提為一種理想情況下之假設，才能藉由已知節點的動作決定建立虛擬軌道的方式。

2.1.2 泛流機制相關研究探討

在MANET環境中，並沒有規定必須有一個特定的有線裝置存在以供資訊的交換，故節點將通常會以互相進行以點對點連線為基礎，在節點之間交換資訊，以達到資訊散佈的目的。

而對於一個節點來說，主要將自身所擁有的資訊的散佈出去之方式為廣播(Broadcast)，但是一個節點的可傳輸範圍有限，故若想傳遞資訊到可傳輸範圍以外的節點，必須透過在兩者之間的許多節點，以多重跳躍(multi-hop)之方式傳遞資訊，此種方式被稱為泛流(flooding)。在近年的研究中，基本的泛流已經被加入許多新的觀念，分成數種泛流方式。以下為泛流方式的分類整理(Tseng, Ni, Chen, & Sheu, 2002)：

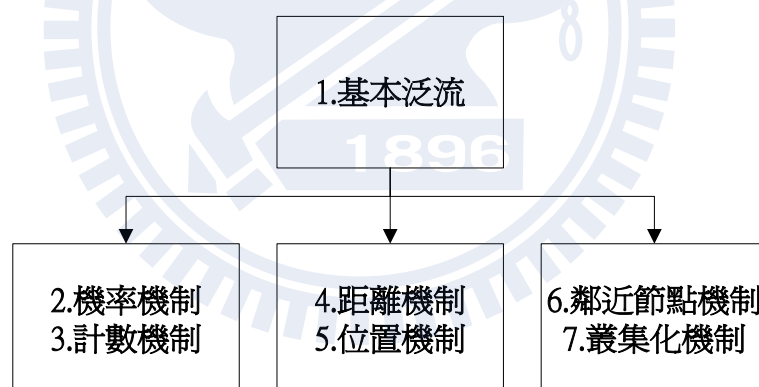


圖 3. 以概念相似性之泛流機制分類

1. 基本泛流(Simple Flooding)：

當節點收到可傳輸範圍內其他節點傳送來的封包時，會先查看封包中的TTL(Time to Live)值，TTL若是大於零即為有效封包，收到封包的節點就會查詢其歷史紀錄是否已接受過此封包，若是第一次接收的封包，才繼續將此封包之TTL值遞減再傳遞出去，直到TTL值歸零時，收到封包的節點才不再

繼續傳遞資料給其他節點 (Dube, Rais, Wang, & Tripathi, 1996)。

此種基本泛流方法雖然簡易，但若是處於一個節點之間都非常接近的無線網路環境中，由於節點密度過高，造成容易在短時間之內收到大量的廣播封包，使封包之間容易產生碰撞與競爭，進而導致大量的封包被重傳，所以持續造成封包不斷碰撞與重傳之惡性循環，也就是廣播風暴問題(Tseng, Ni, Chen, & Sheu, 2002)。

針對廣播風暴問題，許多研究中提出了利用不同的系統架構或環境參數來降低發送廣播封包的次數，以避免廣播風暴的發生。接下來的介紹的其他機制即是為了解決此問題而被提出。

2. 機率機制 (Probabilistic Scheme)：

基於基本泛流之傳播決策方式，但是在節點接收到封包時，首先產生一個介於0至1之間之機率亂數，再將其與系統規定之臨界值比較，若此機率亂數大於臨界值則再依基本泛流機制傳送此封包。

此種機制之實用性完全取決於臨界值的選擇，若臨界值太大，則容易造成封包傳遞距離過短；但若臨界值太小，則幾乎每一個封包都會被判斷可以繼續傳輸，造成機率機制完全沒有產生防止廣播風暴的效用。即使找到一個最適合系統狀況的臨界值，但由於是根據機率產生決策，仍然有機會發生廣播風暴或是封包傳遞距離過短。

3. 計數機制 (Counter-Based Scheme)：

由於純以機率決定是否傳遞封包有以上提到的缺點，所以可以仿照一般無線網路中之作法，在收到需要廣播的封包之後並不立即廣播，而是等待系

統中之倒數器產生一個亂數時間並開始倒數，在倒數完成前若是收到重複的封包，再產生一個臨界值，當倒數到達此亂數時間之後，收到重複封包的數量若小於臨界值，則視為沒有廣播風暴現象而傳送此封包；若是收到重複封包的數量大於臨界值則相反，不傳送封包。

對於此臨界值的設定，可以經由分析無線網路的環境提出了以依節點通訊範圍與覆蓋範圍作比較(Tseng, Ni, Chen, & Sheu, 2002)，並就時間內收到封包的數量與覆蓋範圍之關係，作為設定臨界值的標準。

4. 距離機制 (Distance-Based Scheme)：

在此方法中以距離取代時間作為主要的判斷依據 (Dube, Rais, Wang, & Tripathi, 1996)。距離機制主要依收到封包之訊號強度，用以換算封包來源節點的距離。當接收到重複的封包時，若是接收到的重複封包距離過於接近而小於系統規定的臨界值，則不會繼續廣播此封包；若是封包距離大於臨界值則繼續廣播之。

5. 位置機制 (Location-Based Scheme)：

與距離機制類似，但是將判斷節點間距離的來源由封包強度改為額外的附加工具，如透過全球衛星定位系統(Global Positioning System, GPS)取得節點之絕對位置，再依幾何方法算出兩點之間的距離，並使用前段所提之距離機制進行決策，判斷是否要廣播封包。

6. 鄰近節點機制 (Neighbor Knowledge Scheme)：

在每一個節點中皆建立一張鄰居表，記錄所有在可傳輸範圍內之節點ID。若是接收到封包並需要加以廣播時，比對自己的鄰居節點是否是封包來源的

鄰居節點，若為兩者之鄰居節點則不傳送封包 (Lim & Kim, 2000)。

以圖4為例，雖然節點B接收到了來自發送端節點S之封包，並要再廣播出去，但因節點A與節點C亦同時為節點B與節點S之鄰居，且只有節點B能傳送封包制節點D，故節點B只會負責廣播資訊給節點D，而不會廣播資訊給節點B與節點C。

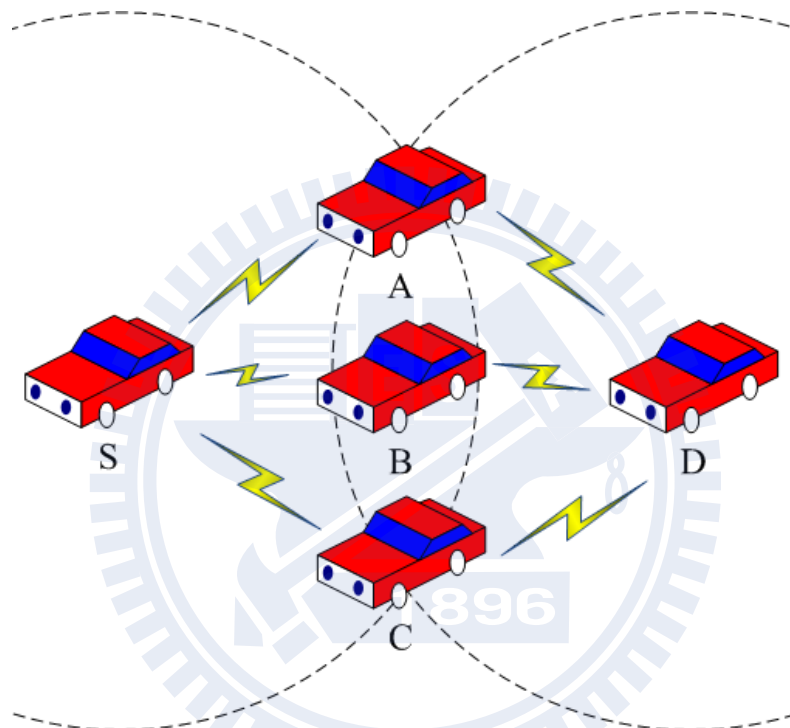


圖 4. 鄰近節點機制與資訊散佈關係

7. 叢集化機制 (Cluster-Based Scheme) :

此機制主要是藉由蒐集鄰近節點之封包，並利用節點所發出的封包之標頭檔案(Header)資料取得來源節點ID。在一定區域ID最小的車輛會擔任叢集管理者(Cluster Head)，管理叢集(Cluster)內部的訊息管理與對外之訊息傳遞，而其他非叢集管理者之節點則被稱為叢集成員(cluster member) (Basu, Khan, & Little, 2001)。若是一個叢集成員同時處在兩個叢集之間，則會成為兩個叢集之溝通管道，負責分配與調控叢集間的封包往返，稱為叢集閘道(Cluster

Gateway)。下圖5即為叢集架構示意圖。

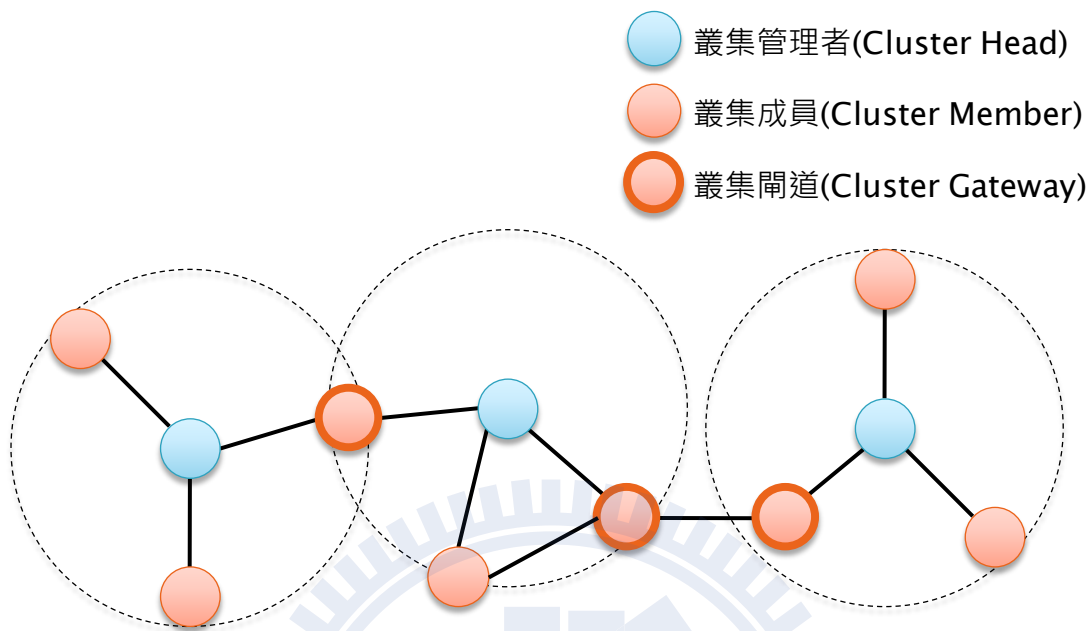


圖 5. 叢集架構示意圖

由於此種叢集化機制依貪婪演算法選出叢集管理者，可能會因為叢集管理者實際地理位置或與其他車輛之相對位置，造成仍有不必要的訊息傳遞產生，這也是本論文計畫要改進之處。

2.1.3 Vehicular Ad-Hoc Network (VANET)相關研究探討

由於VANET之使用環境為實際交通路網，而節點為一般車輛，節點之移動必然具有方向性，且通常是沿著固定的直線道路移動，所以在傳遞資訊時必須將此種交通路網特性列入考慮。

根據封包傳遞之方向性，可以將傳遞封包方式分為三類，分別為同向模式(Same-Direction Model)、反向模式(Opp-Direction Model)與混和模式(Bi-Direction Model) (Nadeem, Shankar, & Iftode, 2006)。

1. 同向模式(Same-Direction Model)：將封包傳遞給與來源車輛行進方向相

同之車輛，通常是在較不緊急的封包上使用。

2. 反向模式(Opp-Direction Model)：藉由對向車道上的車輛，將訊息傳遞給該車輛後再轉為同向模式，即可將此封包更快速的帶往目標車輛後方。

3. 混和模式(Bi-Direction Model)：即為兩種模式之綜合方式，同時使用同向與對向車輛進行資料傳輸。

在本研究中，主要使用混和模式之資料傳遞方式以傳送與接收VANET之訊息封包，在選擇傳遞封包的目標車輛較具有彈性與效率，以適應各種傳遞封包與路徑規劃之演算法。

2.2 VANET 動態路徑導航之封包傳送演算法

由於 VANET 屬於高度動態變化與非連接導向的網路，網路中車輛之間的網路連線並不固定，所以當附近不存在適合傳遞封包的車輛時，車輛可以選擇繼續攜帶著封包行駛，直到可傳遞範圍內出現適合的車輛為止 (Li & Rus, 2000) (Vahdat & Becker, 2000)。因此，目前對於動態車輛導航的研究，主要是著重於在車輛傳遞封包時，要如何挑選傳遞目標與路徑，讓封包能順利且有效率的傳送往目的地。

隨機車輛配對的方式，將封包傳遞的成功率最大化並將資源消耗與延遲時間最小化的漫延式封包傳遞法(Epidemic Routing)被提出 (Vahdat & Becker, 2000)。

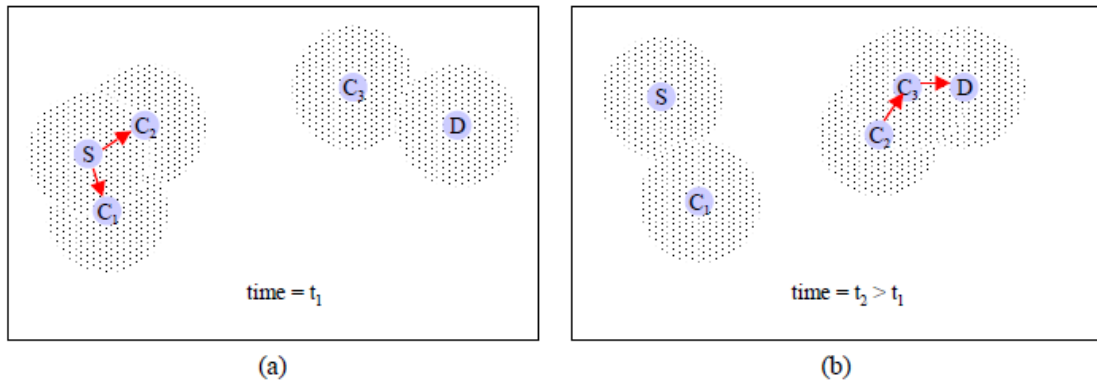


圖 6. 連接導向之無線隨意網路 ERPC 演算法

如圖 6 所示，當車輛 S 希望將封包傳遞往目的地 D，但 S 與 D 之間並無連線時，會將封包傳送給周圍所有的車輛，即車輛 C1 與車輛 C2。C1 與 C2 在行駛過程中亦會繼續嘗試尋找連接至 D 的連線，並透過此連線傳送資料。如無連線，仍會繼續將封包往附近所有車輛傳送，直到發現存在至 D 之連線為止，再交由此條連線傳遞封包。

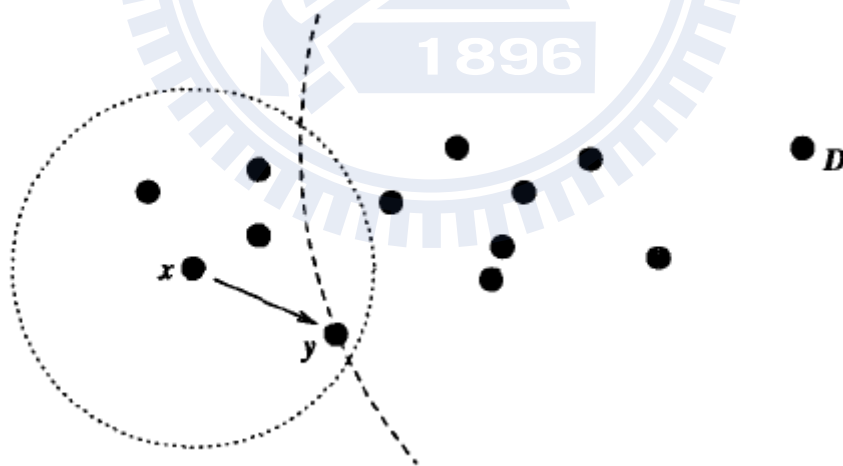


圖 7. 連接導向之無線隨意網路 ERPC 演算法

以車輛的位置與目標車輛的方向的封包傳遞貪婪演算法(Greedy perimeter stateless routing, GPSR)也被應用在傳遞封包中 (Karp & Kung, 2000)。在 GPSR 演算法中，如圖 7 所示，若節點 x 欲傳送資料給另一節點 D，但兩節點 x 與 D 互不再對方的可傳遞範圍內，則節點 x 在其可傳遞範圍中，以貪婪演算法(Greedy

Algorithm)方式尋找一個最接近節點 D 之節點，即為節點 y，用以建立節點 x 與節點 y 之間之傳輸路徑。

當整體網路中的傳輸路徑逐漸被挑選決定後，便會形成如下圖 8(a)之網路結構。此時由於節點間路徑過多，容易使傳遞封包時因選擇路徑而延遲，故將此網路路徑取其 Relative Neighborhood Graph (RNG)子集，進行平面化，去除在網路中多餘之而不必要之路徑，便得到圖 8(b)之網路結構。再對此網路取 Gabriel Graph (GG)子集，在保持各個節點連通的情況下，考慮節點之間之訊息可傳遞範圍，再次除去網路中多餘之而不必要之路徑，即得到簡化後的網路圖 8(c)，圖中任意兩節點皆有明確且簡單的路徑可進行訊息交換。

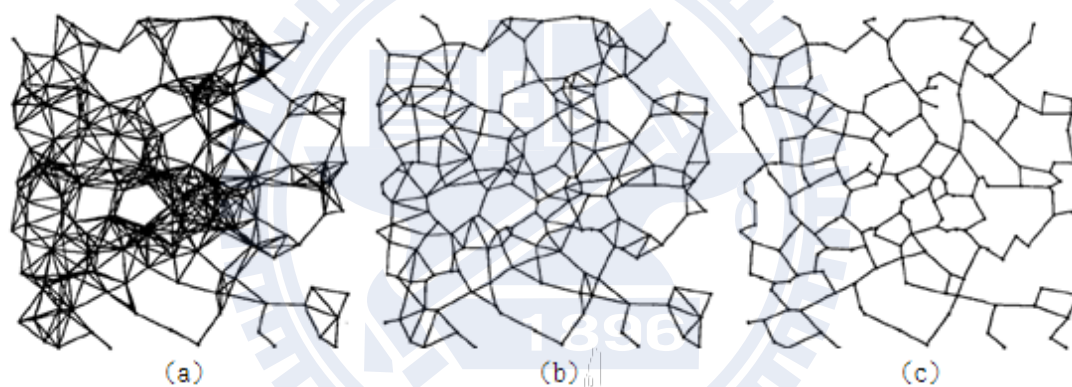


圖 8. GPRS 演算法之網路形成過程.

關於動態路徑規劃的研究中，將挑選封包傳遞對象的方式分成三類 (Zhao & Cao, 2008)，以下圖 9 為例：

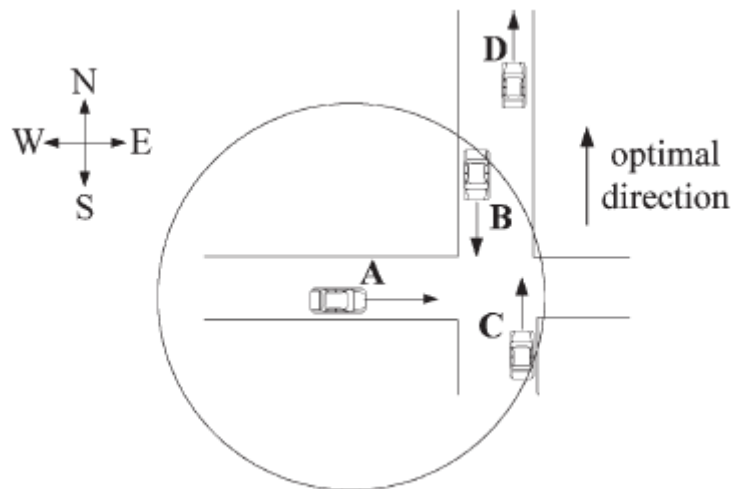


圖 9. GPSR 演算法之網路簡化過程

1. 位置優先(Location First Probe)：

在可傳遞範圍中，將封包傳遞給離目的地最近的車輛，傳送封包的距離短，但可能因為車輛的行進方向造成傳遞迴圈而減低效率，甚至無法傳到目的地。以圖 9 為例，若車輛 A 之目的地在北方，則車輛 A 會依位置優先規則，在可傳遞範圍之中挑選距離目的地最接近的車輛 B 為傳送封包之目標。

2. 方向優先(Direction First Probe)：

在可傳遞範圍中，將封包傳遞往目的地開去的車輛，傳遞距離可能較長，但可避免發生迴圈。以圖 9 為例，則車輛 A 傳送封包時，會依方向優先規則，在可傳遞範圍之中挑選行駛方向為北方的車輛 B 為傳送封包之目標。

3. 混和型(Hybrid Probe)：

根據此兩類方法的特性，進一步提出了混和型方式，作法為將位置優先加入迴圈偵測機制，當偵測到已經發生迴圈時，則改成方向優先傳送封包。

2.3 效能評估機制

在測量與評估一個動態路徑演算法之效能方面，考慮到動態路徑規劃法及其使用環境與一般無線網路的差異與特性，也就是車用網路為高度動態與非連接導向的網路，所以主要可以分成四個重要項目(Enge, Wailter, Pullen, kee, Chao, & Tsai, 1996; Hein, 2000; Vahdat & Becker, 2000; Ochieng & Sauer, 2003)：

1. 準確性(Accuracy)：

當資訊經過多次傳遞、接收與分析之步驟後，得到的結果資訊，如位置與速度等，與原始交通資訊的相似程度與關聯程度。

2. 完整性(Integrity)：

評估接收到之是否可以信任及是否有參考價值，因網路傳遞中可能因為未預料的錯誤或意外造成資料並不完整而不可使用。

3. 可用性(Availability)：

交通網路範圍之覆蓋率，代表系統蒐集資訊範圍是否含蓋了足夠範圍，以得到足夠詳細與完整的資料。

4. 服務連續性(Continuity of Service)：

系統在不中斷的情形下，能連續提供服務的能力。

在本研究中，由與研究目標為提出一個演算法概念，並模擬其在交通環境中的效能表現，所以並未將服務連續性此指標納入效能評估。

第三章 研究方法

本章中將基於車用隨意網路(Vehicular Ad-Hoc Network, VANET)之網路架構與功能，提出一個適用於市區交通環境的車輛叢集化(Clustering)演算法，稱為交易代幣式叢集化(Token-Based Clustering, TBC)演算法。TBC 演算法可以根據車輛交易的次數以及所持有代幣(Token)的數目，從將地理位置鄰近的車輛進行叢集化，並選出一台車輛作為叢集管理者(Cluster Head)，而將其他的車輛則是叢集成員(Cluster Member)。在 TBC 演算法中，當叢集(Cluster)內任何車輛需要傳遞或請求路況資訊時，依其在叢集中屬於管理者或是成員，選擇其目標車輛的演算法。如圖 10 所示，箭頭方向即為一種在車輛間傳遞路況資訊之方式。

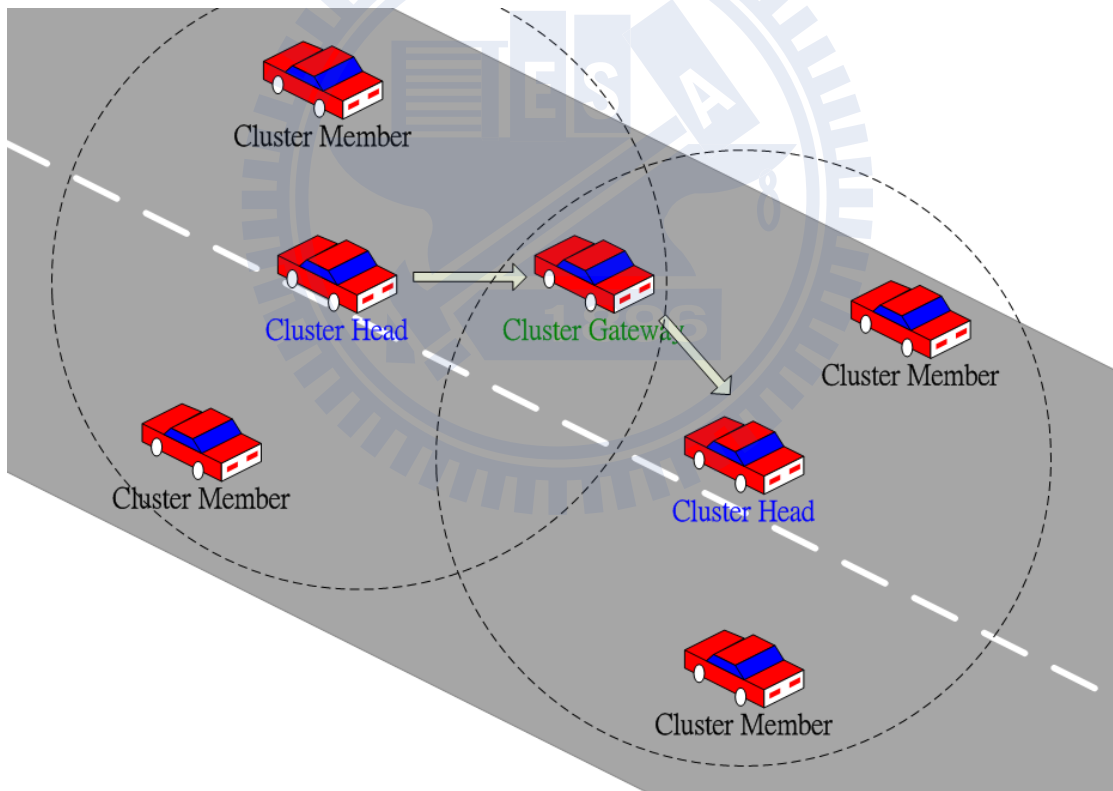


圖 10. 在叢集中車輛傳遞資訊之範例

在 TBC 演算法中，最主要的精神就是將車輛的地理位置與路況資訊的更新時間數值化成代幣數，而讓有路況資訊需求的車輛花代幣購買這些資訊，愈接近被查詢地點的車輛得到的路況資訊與愈晚得到的路況資訊價格便愈高。

在本章中，首先將會在 3.1 節介紹整個研究實驗的設計，包括如何建立模擬用的交通模型、參數的設定方式與來源、模擬程式流程與資料蒐集方式。在 3.2 節中，將會介紹 TBC 演算法之概念、設計、叢集調整方法、訊息傳遞方法與代幣計算機制。在 3.3 節中，首先會提出數種用以分析與評估實驗結果之指標，並且會將各種模擬環境與參數所產生的結果逐一探討。

3.1 實驗設計

在本節中，主要介紹整個研究的實驗架構、參數設定與模擬程式流程。在 3.1.1 小節中會介紹在此實驗中，為了達到動態路徑規劃的效果與實作 LCC 演算法與 TBC 演算法，封包所需要附帶的資料內容。在 3.1.2 小節中將會介紹本實驗的參數如何設定、參考用的實際資料出處與如何轉換在這些實際資料在模擬程式中使用。在 3.1.3 小節中，

3.1.1 封包內容

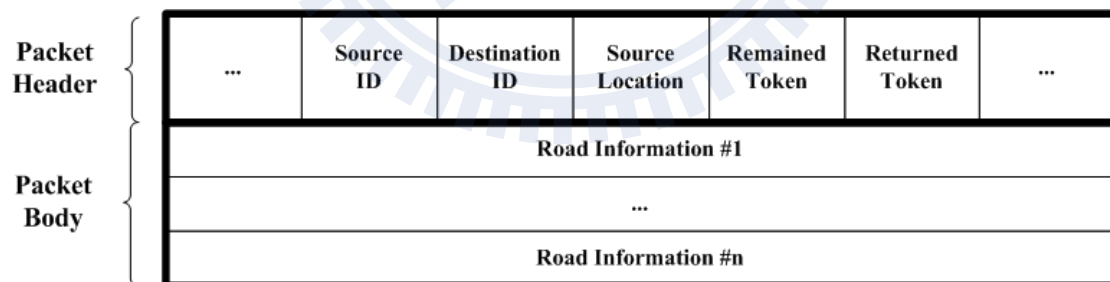


圖 11. 封包內容

在本研究中所使用的封包內容如上圖 11 所示，封包格式依循一般 VANET 中的規範與定義，主要分為封包標頭(packet header)與封包本體(packet body)兩部分。在本實驗中，封包標頭為之內容主要包含此封包的特性與敘述等資料，而封包本體為此封包實際要傳遞之訊息，在此只介紹與本實驗相關之欄位資料，敘述如下：

封包標頭(Packet Header)

1. Source ID：發出封包車輛所裝置的車上裝置(OBU)之 ID。各車輛所使用的 ID 皆為唯一，意即不同車輛之 ID 不會相同。
2. Destination ID：此封包傳遞目標車輛所裝置的車上裝置之 ID，若為此欄位為空白，則代表此訊息為並不具特定傳遞目標之廣播訊息，所有車輛皆可接收。
3. Source Location：發出封包車輛之地理位置座標，由車輛所裝載的 GPS 得到。此欄位將在 TBC 演算法中使用，作為挑選叢集管理者時的參考依據，在 TBC 演算法中的使用方式將會在 3.2 節中介紹。
4. Remained Token：發出封包車輛目前擁有的代幣。此欄位將在 TBC 演算法中使用，作為挑選叢集管理者時的參考依據，在 TBC 演算法中的使用方式將會在 3.2 節中介紹。
5. Returned Token：此封包查詢要求所退回的代幣，當有車輛回應此查詢訊息後，會依該回應車輛所擁有路況資訊之更新時間退回部分代幣。在 TBC 演算法中的使用方式將會在 3.2 節中介紹。

封包本體(Packet Body)

在實驗中，將以查詢路況作為範例服務進行模擬，意即假設駕駛人有透過 VANET 查詢本身所要行駛至之目的地與路程中的道路路況資訊之需求。在此範例服務之情形下，如圖 11 所示，封包本體中所含的內容即為各條道路之路況資訊，此封包之目標即為傳遞含有道路編號 1 至道路編號 n 之路況資料給其目標車輛。

3.1.2 參數設定

本實驗的目標主要為比較在不同車輛密度與不同車輛比例中，使用 LCC 演算法之車輛與使用 TBC 演算法之車輛在模擬交通環境中的效能比較。

在本實驗中，密度的定義是以臺北市作為模擬環境之範例，根據九十八年度臺北市交通流量及特性統計(一般路口、圓環路口及路段調查部分)計算車輛密度，並將其作為模擬模型之初始參數 (台北市交通管制工程處, 2009)。

在本實驗中，對於車輛密度的定義將會依下列公式設定，並使用在模擬之交通模型中：

$$\text{車輛密度} = \frac{\text{車輛總數}}{\text{道路總長度} / \text{每台車輛所佔長度}}$$

在選定了觀測的道路之後，該道路之車道數與平均車速也可以從統計資料中取得，故在本實驗中車輛密度將會使用由以下公式所推得：

$$\text{車流量} = \frac{\text{平均車速}}{\text{每台車輛所佔長度}} \times \text{車道數} \times \text{車輛密度}$$

因為車流量可直接由臺北市交通流量及特性統計(一般路口、圓環路口及路段調查部分)之統計資料中直接取得，再經由上列之轉換公式將其轉換為車輛密度，便可以直接在模擬中使用。以下圖 12 為例，其意義為在固定時間內行駛經過道路某截面之車輛總數。

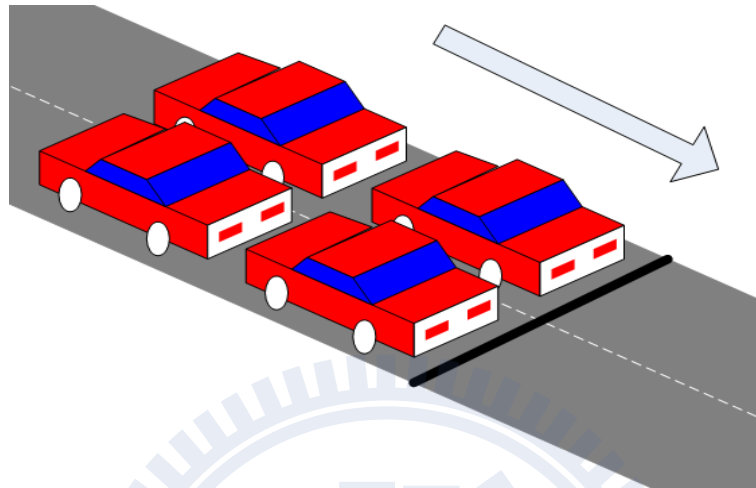


圖 12. 車流量計算方式

車輛之比例設定是依台灣 ITS 發展目標所定 (中華民國交通部運輸研究所, 2010), 並經由計算車輛密度得到交通環境中總車輛數目後, 再依此車輛數目與比例, 決定交通環境中需要配置使用 TBC 演算法或 LCC 演算法等使用動態路徑規劃之車輛。

3.1.3 模擬交通模型

在本實驗中，將以細胞自動機(Cellular Automata)為基礎概念 (Ohara, Nojima, & Ishibushi, 2007)，建立實際交通道路之模擬交通模型，並依細胞自動機之規則，在其上配置道路物件與車輛物件進行實驗。在模擬環境(Environments)物件中，將建立兩類物件，即車輛(Vehicles)物件與道路(Roads)物件。

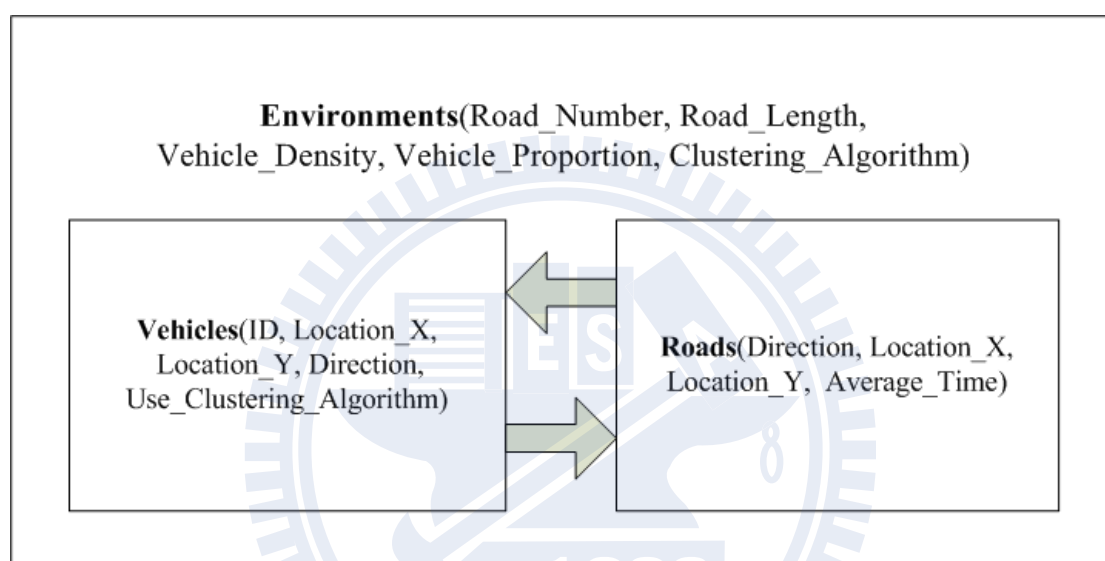


圖 13. 模擬交通模型架構圖

如上圖 13，Environment 物件為模擬之交通環境模型，並且包含了 Vehicles 物件與 Roads 物件，其參數分別為：

1. Road_Number :

道路總數量，以此參數建立道路模型，若此數為 n，則建立為 n 條垂直道路與 n 條水平道路的交通模型。

2. Road_length :

道路長度，在道路交叉口間的網格(cell)數目，意即兩個交叉路口間可以容納的車輛個數。

3. Vehicle_Density :

車輛密度，在給定 Road_Number 與 Road_length 兩參數後，可以計算出在此交通模型中的總網格個數，而再以車輛密度決定所要配置在交通模型中之車輛總數。以總網格數 1000 為例，若 Vehicle_Density 為 0.5，則平均每個網格中會擁有 0.5 個車輛物件，總共在交通環境中會有 500 個車輛物件。

4. Vehicle_Proportion :

車輛比例，再給定 Road_Number、Road_length 與 Vehicle_Density 三參數後，可以計算出交通模型中的總車輛物件數，再依車輛比例決定共有多少車輛使用叢集化演算法。若以總車輛物件數 500 為例，若 Vehicle_Proportion 為 10%，則總共會有 50 個車輛物件使用叢集化演算法進行動態路徑規劃，而其餘 450 個車輛物件則使用靜態的最短路徑規劃，依在各個道路中可以行駛的速度決定路徑。

5. Clustering_Algorithm :

叢集化演算法類型，即為從 LCC 演算法或 TBC 演算法中挑選出其中一種進行實驗，決定以此種叢集化演算法進行實驗。

Vehicles 物件為模擬之車輛物件，其參數分別為：

1. ID :

車輛物件之編號，為唯一值。

2. Location_X 與 Location_Y :

表示車輛在環境中之水平與垂直位置。

3. Direction :

表示車輛接下來將要行駛的方向。

4. Use_Clustering_Algorithm :

為布林值(Boolean)，表示是否適用本實驗參數 Clustering_Algorithm 所設定之叢集化演算法，若為否，則此車輛物件使用靜態的最短路徑規劃，依在各個道路中可以行駛的速度決定路徑。

Roads 物件為模擬之道路物件，其參數分別為：

1. Direction :

道路方向，表示此道路為水平道路或垂直道路。

2. Location_X 與 Location_Y :

道路兩端之端點位置，以 Direction、Location_X 與 Location_Y 三個參數，可決定一條唯一的道路。

3. Avg_Time :

車輛在此道路上之平均行駛時間，在最近幾回合內車輛在此道路上之平均行駛時間。

在本實驗中，將以細胞自動機為例建立交通環境與道路物件，並在其上配置車輛物件。原始的交通環境示意圖如下圖 14，以下將簡介如何由輸入之參數設定交通模型。

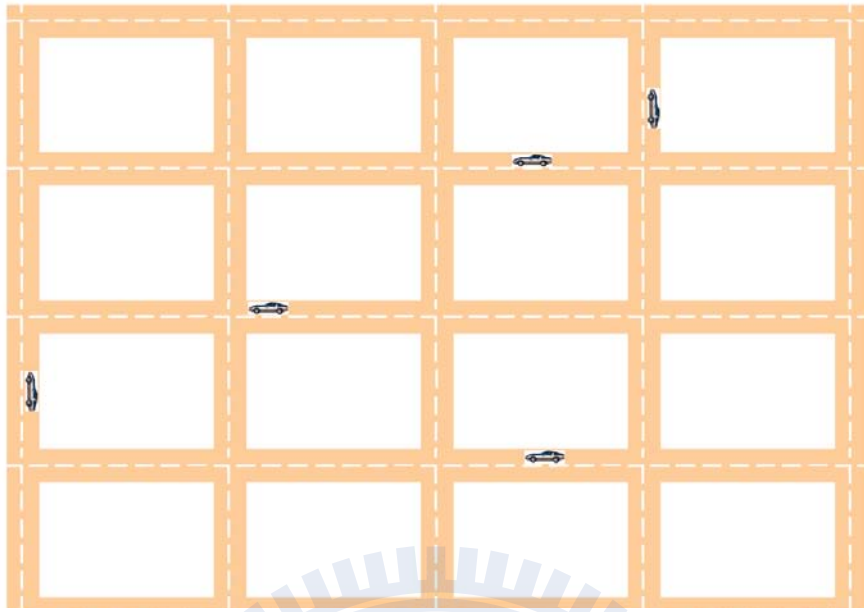


圖 14. 以細胞自動機建立之交通環境、道路與車輛圖

首先，以圖 15 為例，若需要模擬圖 14 中的交通道路，則將參數 `Road_Number` 設為 4，即代表垂直與水平各有 4 條道路，若 `Road_Length` 為 20，則表示兩個道路交叉口之間長度為 20 個網格，則可以使用這些參數建立一個以細胞自動機為基礎之網格狀交通模型，如下圖 15。

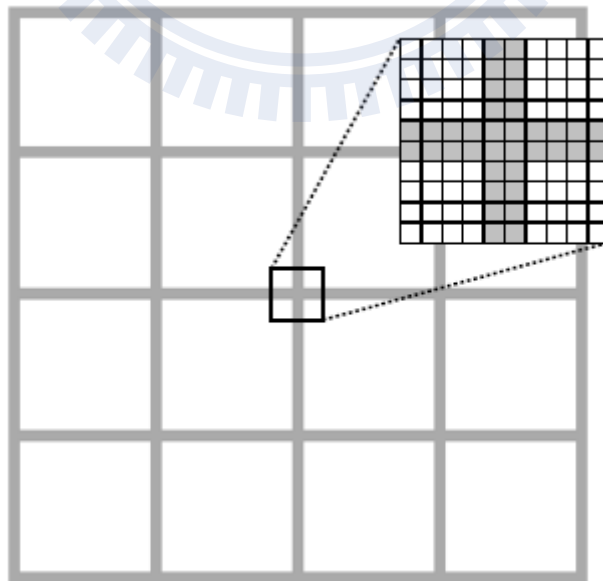


圖 15. 以細胞自動機為基礎之網格狀交通模型

當網格狀交通模型建立完成之後，將依 Vehicle_Density 參數初始化，隨機在道路中任何網格配置車輛物件，配置後的示意圖如下圖 16，黑色網格即為具有車輛的道路網格，而灰色部分則為無車輛占據之道路網格。



圖 16. 配置車輛後的網格狀交通模型

本實驗以細胞自動機為基礎建立實驗用交通模型，以下圖 17 為例，網格 s_i 為在時間為 t 時要計算之目標網格，網格 s_{i+1} 為在時間為 t 時網格 s_i 前方的網格，網格 s_{i-1} 為在時間為 t 時網格 s_i 後方的網格，而最下方的網格即代表網格 s_i 在時間為 $t+1$ 時的網格，而最下方網格之內容會由網格 s_{i-1} 、 s_i 與 s_{i+1} 等三個網格之內容共同決定。

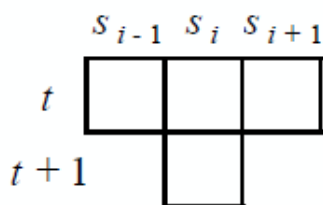


圖 17. 應用在交通模擬之細胞自動機變化規則

下圖 18 列出所有在一維道路上所有可能遭遇的車輛排列情形與車輛之移動結果，其中黑色網格代表此網格已具有車輛，將其值設為 1；而白色網格代表此

網格尚未具有車輛，將其值設為 0。以下圖 18 中最左上角之小圖為例，此圖表示網格 s_{i-1} 、 s_i 與 s_{i+1} 在時間為 t 時皆有車輛，則在時間為 $t + 1$ 時網格 s_i 中的車輛會因前方尚有車輛而繼續待在原來網格中，故網格 s_i 中仍會有車輛，將此網格之值設定為 1。

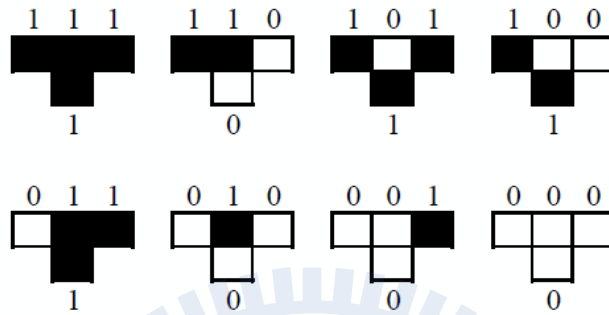


圖 18. 一維細胞自動機可能遇到之所有情形



3.1.4 模擬程式流程

下圖 19 為在實驗程式中，每一個車輛物件在每一回合模擬之中所進行之動作流程圖。在模擬過程中，車輛物件會依照以下過程與條件式逐步執行動作，直到整體模擬結束為止。

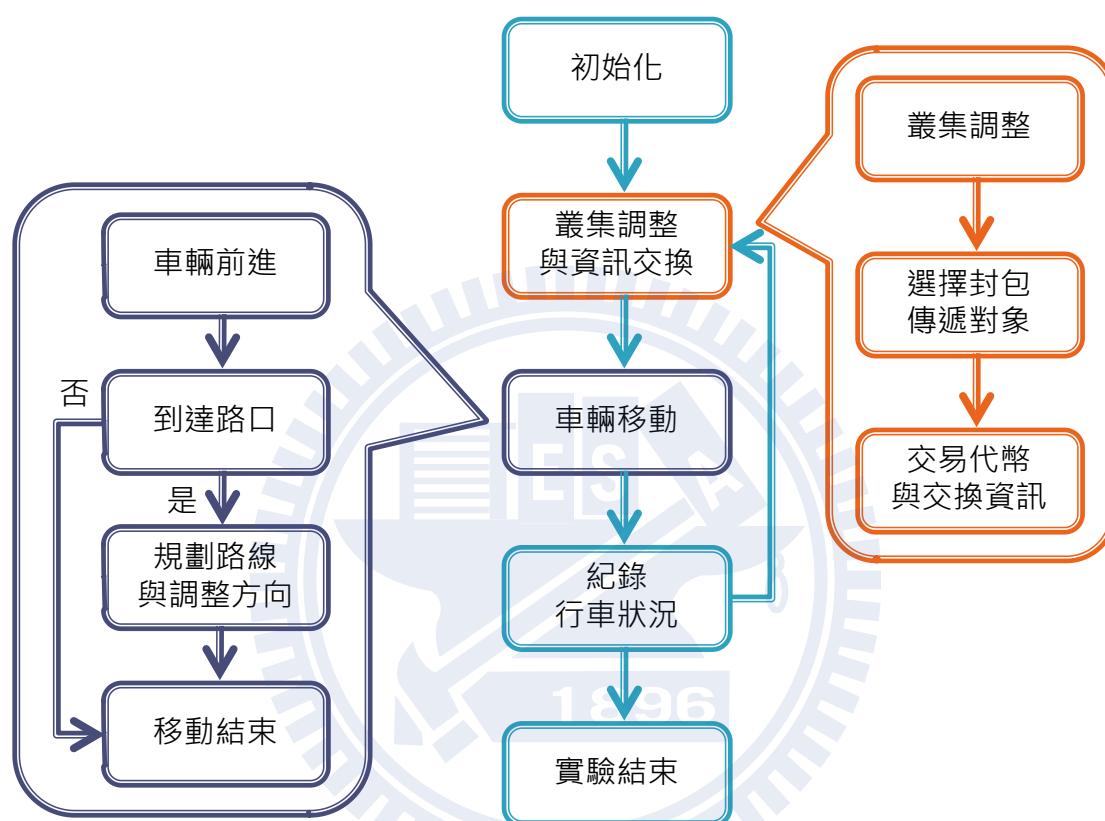


圖 19. 每回合進行之實驗模擬流程

在本實驗中調整叢集時，由於使用了與 LCC 演算法相同之基礎規則(Chiang, Wu, Liu, & Mario, 1997; Yu & Chong, 2005)，並且因為在 LCC 演算法架構中將一個 clock 定為 $125 \mu s$ ，而在少數 clock 中即可執行完叢集調整動作，故在本實驗中並不特別討論在調整叢集時所花費的額外時間。

3.2 交易代幣式叢集化(Token-Based Clustering, TBC)演算法

3.2.1 TBC 演算法概念與設計

在一般的叢集化演算法中，通常都是選定節點上某個數值作為叢集化時選擇叢集管理者之參考 (Basu, Khan, & Little, 2001)，如最小 ID 叢集化(Lowest ID Clustering, LIC)演算法是以節點 ID 作為叢集化時的參考，選擇 ID 最小的車輛作為叢集管理者 (Ephremides, Wieselthier, & Baker, 1987)，而最高連接數叢集化(Highest Connectivity Clustering, HCC)演算法則是考慮網路拓樸情形，以分支度(Degree)最高的節點最為叢集管理者 (Gerla & Tsai, 1995)。

在 LCC 演算法中 (Chiang, Wu, Liu, & Mario, 1997; Yu & Chong, 2005)，無論是使用 LIC 演算法或 HCC 演算法，應用在 VANET 環境中，仍有需要面對的問題。首先，在使用 LIC 演算法的情形下，若是環境中有一車輛 ID 較其他大部分車輛 ID 小時，因車輛 ID 並不會改變，則可發現此車輛容易持續擔任叢集管理者，造成負擔比其他大部分車輛重，無法平衡各個車輛的負擔，若是在車輛密度較高的環境中，可能會造成叢集管理者負擔過重，而無法有效的對叢集成員要求作出回應。其次，若是有一 ID 較小的車輛進入了一個叢集，則叢集管理者將會改變成此 ID 較小的車輛，在高密度的交通環境中容易造成不必要的叢集調整。

同上段所述，若是採用 HCC 演算法，也會有需要解決的問題。在 VANET 環境中，由於車輛移動速度很快，車輛節點間形成的網路拓樸容易快速且連續的變動，造成需要不斷的改變叢集管理者。其次，若是在一個節點密度較高的環境中，由於每一個節點分支度皆相當高，所以在叢集化時，分支度並不是相當好的參考指標。

因為上述問題，所以本研究以 LCC 演算法為基礎，並加入了代幣交易機制作為叢集化時的參考，提出了 TBC 演算法。車輛在加入環境中時，即配置一定數量的代幣給該車輛，而車輛可以使用代幣向其他車輛購買路況資訊，而在車輛購買路況資訊時，會依買方所持有的代幣數量與賣方的資訊品質計算交易價格，並且再依車輛所持有的代幣數量進行叢集化，以持有代幣數量最高的車輛擔任叢集管理者。

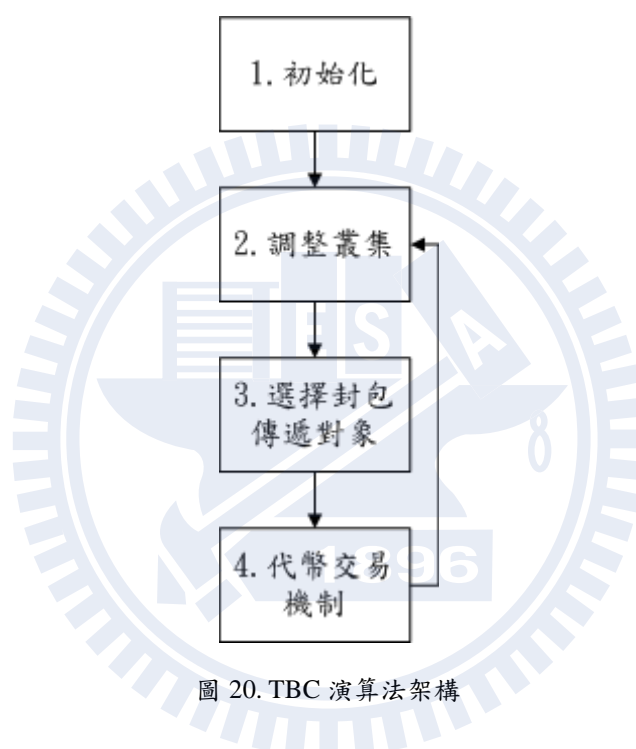


圖 20. TBC 演算法架構

如圖 20 所示，在 TBC 演算法中，主要可分為四個部分：

1. 初始化：

當車輛進入交通網路時所進行之初始化動作，如調整車輛狀態與配置預設之代幣。

2. 叢集調整方法：

負責在交通環境中產生或移除叢集，也就是判斷車輛狀態是否該從叢集成員改變為叢集管理者，而產生新的叢集，或是將車輛狀態由叢集管理

者改變為叢集成員，而移除舊的叢集。

3. 選擇封包傳遞對象：

車輛在自身所處叢集中，當需要發出查詢路況資訊等請求時，依其叢集關係與地理位置，選擇車輛作為傳遞封包之對象。

4. 代幣交易機制：

再選定封包傳遞對象後，依本身買方之資訊更新時間與傳遞對象賣方所擁有代幣數目計算交易價格。

關於以上所提到的 TBC 演算法之四個部分，每一個部份之詳細內容將在接下來的小節中敘述。

3.2.2 初始化

演算法

```
initial(){  
    self.state←"cluster_member"  
    self.remained_token←100  
}
```

說明

當車輛進入交通環境時，車輛之狀態為叢集成員，並隨即配置一定量之代幣給自己，而在此實驗中，將進入系統車輛之代幣值設為預設值 100。

關於代幣之預設值，在本實驗中其實並不重要，因為無論代幣預設值高於或

低於整個交通環境中所有車輛代幣之平均值，也會因 TBC 演算法的特性中，擁有較多代幣的車輛消耗代幣也會較快，而擁有較少代幣的車輛消耗代幣會較慢，故可因此特性調整代幣從預設值至適合值。

3.2.3 叢集調整方法

演算法

```

generate_cluster_head(){
    state result_value ← "cluster_head"
    for each vi ∈ in_Transmit_Range(self)
        if vi.state = "cluster_head" ∧ self.remained_token <
            vi.remained_token
            result_value ← "cluster_member"
            break
        end if
    end for each
    self.state ← result_value
}

```

```

remove_cluster_head(){
    state result_value ← "cluster_head"
    for each vi ∈ in_Transmit_Range(self)
        if vi.state = "cluster_head" ∧ self.remained_token <
            vi.remained_token
            result_value ← "cluster_member"

```



```
        break
    end if
end for each
self.state←result_value
}
```

說明

在 TBC 演算法的叢集調整方法中，主要依循 LCC 演算法中的調整規則，在以下兩個情形時才會產生或移除一個叢集管理者車輛 (Chiang, Wu, Liu, & Mario, 1997)：

1. 產生叢集管理者：

當一個狀態為叢集成員的車輛之可傳輸範圍內不存在狀態為叢集管理者的車輛時，將會檢查自身所擁有的代幣數是否高於所有臨近車輛之代幣數，若是自身所擁有的代幣數較高時，則將自身狀態修改為叢集管理者。

2. 移除叢集管理者：

當一個狀態為叢集管理者的車輛之可傳輸範圍內存在狀態為叢集管理者的車輛時，將會檢查自身所擁有的代幣數是否高於該叢集管理者車輛，若是自身所擁有的代幣數較低時，則將自身狀態修改為叢集成員。

在 TBC 演算法中，當遇到 LCC 演算法所提出之兩個情形時，才會產生或移除一個叢集管理者車輛，但是 TBC 演算法之叢集化參考值為代幣數目，而 LCC 演算法之叢集化參考值則為車輛 ID 或是分支度。

3.2.4 選擇封包傳遞對象

演算法

```
vehicle choose_transmit_target(){  
    vehicle v  
    if self.state = "cluster_head"  
        if in_Transmit_Range(self) != nil  
            v ← H-VADD(in_Transmit_Range(self) ∩  
                        is_cluster_member(Vehicles))  
        end if  
    end if  
    if self.state = "cluster_member"  
        if in_Transmit_Range(self) != nil  
            v ← H-VADD(in_Transmit_Range(self) ∩  
                        is_cluster_head(Vehicles))  
        end if  
    end if  
    return v  
}
```

說明

在 TBC 演算法中，當需要挑選傳遞封包的對象時，主要是依據 H-VADD 演算法選擇對象車輛 (Zhao & Cao, 2008)。在 H-VADD 演算法中，首先會以地理位置較接近封包傳遞目的地的車輛作為優先考量，但若是偵測到封包傳遞可能發生迴圈時，則改用車輛方向作為優先考量，將封包傳遞給行車方向較接近封包傳遞

目的地的車輛。

3.2.5 代幣交易機制

演算法

```
double calculate_paid_token(){  
    return target.remained_token / degree(target)  
}
```

```
double calculate_returned_token(){  
    double paid_token  
    paid_token ← target.remained_token / degree(target)  
    return paid_token  
}
```

說明

在 TBC 演算法中，最主要的精神就是將車輛的地理位置與路況資訊的更新時間數值化成代幣數目，而讓有路況資訊需求的車輛花代幣購買這些資訊，愈接近被查詢地點的車輛得到的路況資訊與愈新的路況資訊價格便愈高，而在本研究中，關於代幣計算時，所參考之地理位置選擇方式主要是依據 VADD 演算法中之混合型判斷方式(Zhao & Cao, 2008)。

當車輛所擁有之代幣數目愈多時，交換路況資訊時所花費的代幣數目也就愈多，因為當車輛擁有代幣過多時，容易造成代幣累積不均勻，當此車輛因移動至不佳的地點而不適合繼續擔任叢集管理者時，持有過多的代幣容易造成代幣機制調整速度過於緩慢，無法適應快速變化的交通情形。當車輛之分支度愈高時，叢

集調整所影響到的車輛也就愈多，故在分支度較高時，所交易的代幣也就愈少，讓代幣數目較為穩定，也可以提升叢集的穩定性。為了避免不必要的叢集調整，故在本實驗以各車輛代幣數目為根據以調整叢集時，額外以一倍標準差值作為緩衝值，使得不會因為極少數的代幣數目差異造成不必要的叢集調整。

如 3.1.4 小節所述，當車輛已經選定了一個目標車輛並對其傳遞查詢路況請求之封包時，便會依買方車輛已擁有代幣數目與賣方車輛回應路況資訊之品質決定此查詢路況資訊的交易價格。

3.3 效能評估方法

在本節中，將介紹本研究以何種方式選擇指標以及該指標如何對 LCC 演算法與 TBC 演算法進行比較。在 3.3.1 小節中將介紹評估指標種類、項目以及選擇原因，而在 3.3.2 小節中介紹如何在模擬實驗中藉由交通模型蒐集評估指標所需要的資料。

3.3.1 效能評估指標種類

由於在本實驗的目標為透過實際地理位置與資訊更新時間挑選出叢集管理者，以降低在調整或是管理叢集時需要的額外負擔，也讓形成的叢集能具有適應性，可以自行調整出較完整而不破碎的叢集，讓叢集能有效降低訊息傳遞時的複雜度，故在本實驗中會將指標分為以下三種類別，並對於每一個類別選擇所需要的指標：

1. 對於整體交通環境：

以環境中之平均叢集數量作為指標。對於環境中每一台車輛而言，若是能形成較完整不破碎的叢集，每一個叢集中的成員則會較多，讓整體交

通環境中的叢集數減少。若是環境中的叢集總數目越少，則只需要較少的車輛擔任系統負擔較大的叢集管理者即可，其餘車輛只需要擔任負擔較小的叢集成員。

2. 對於個別叢集：

以叢集調整次數作為指標。對於一個叢集而言，若是愈少調整叢集，則叢集成員不需花額外負擔更改叢集相關資訊，也只需對固定的叢集管理者發送資訊。在本實驗中，每產生一次叢集管理者或是移除一次叢集管理者，皆為叢集調整。

3. 對於個別車輛：

以平均封包傳遞時間作為指標。對於每一台車輛，若是能形成較完整不破碎的叢集，每一個叢集中的成員則會較多，讓整體交通環境中的叢集數減少。在整體車輛數與交通環境不變的情形下，若是能以較少叢集管理整個車用網路環境，對於一個固定出發地點與目的地的封包來說，傳遞過程中所經過的叢集數目即與傳遞所需時間成正比，故愈少叢集愈能減低封包傳遞的時間。

3.3.2 評估指標蒐集方式

當每回合車輛移動結束後，便會將其歷史資訊，如總移動距離與總移動時間等記錄下來，並在本研究第四章中提出結果分析。在 3.3.1 小節中提到了本實驗主要觀察的三項指標，以下將介紹在實驗中如何取得這些指標。

1. 平均叢集數量：

在實驗的每一回合結束前，計算目前交通環境中擁有幾台叢集領導者車輛，並且將其累加至 Cluster_Number 變數，因一個叢集領導者車輛即

代表了一個叢集。在實驗結束後，將 Cluster_Number 變數除以總實驗回合數，即可得到平均叢集數量。

2. 叢集調整次數

在實驗過程中，若是遇到車輛將其狀態由叢集領導者改變為叢集成員或由叢集成員改變為叢集領導者，即將 Cluster_Adjustment 變數加 1，在實驗結束後，Cluster_Adjustment 變數即為叢集調整次數。

3. 平均封包傳遞時間

因為在 VANET 中，封包傳遞時間可以視為在傳遞過程中之傳輸時間與處理時間之總和，而此時間與跳躍(Hop)次數成正比，故在此實驗中，將以累計封包之跳躍次數至 Hop_Number 變數，實驗結束後再將 Hop_Number 變數除以總路況資訊請求發送之次數，即可得到每次路況資訊請求時之平均跳躍數。

第四章 研究發現與分析

在本章中，主要的實驗目標為檢驗在具有基礎車用隨意網路(Vehicular Ad-Hoc Network, VANET)功能的模擬交通環境中，比較使用交易代幣式(Token-Based Clustering, TBC)演算法與使用最少叢集改變(Least Cluster Change, LCC)演算法之叢集(Cluster)效能與負擔等差異 (Yu & Chong, 2005)。

在本章的實驗中，首先將會選擇中等車輛密度之交通環境，並在其中實驗當使用 TBC 演算法之車輛或使用 LCC 演算法之車輛其比例由 5% 至 30% 時，在交通環境中的效能與負擔。接著將會在高車輛密度與低車輛密度的交通環境中，同樣調整車輛比例由進行 5% 至 30% 並進行實驗，再將實驗結果與中等密度實驗中所產生的結果對照。

本研究實驗指標將依第 3.3 節中所定義，依照各項指標之特性偏向全域性質或是偏向區域性質，分為對於整體交通環境、個別叢集(Cluster)與個別車輛等三類，並分別提出平均叢集個數、叢集調整次數與平均封包傳遞時間等三項指標。

在本章的內容中，首先將會在 4.1 節中介紹本論文使用的模擬交通環境與參數設定，接著在 4.2 節中再分別對 3.3 節中所提到之平均叢集個數、叢集調整次數與平均封包傳遞時間等三項指標之實驗結果進行分析與探討。

4.1 模擬交通環境與參數

本實驗使用細胞自動機(Cellular Automata)為基礎交通模型，建立交通環境模型並配置車輛與道路等物件 (Ohara, Nojima, & Ishibushi, 2007)，並依 3.1 節中所介紹之規則，每一個車輛物件只與臨近的其他車輛節點互動，並且只擁有區域性資訊。

表 1. 實驗參數表

模擬參數項目	參數值
系統參數	
1. 每次實驗回合數	1000
2. 封包可傳遞距離	180 公尺 (60 網格長)
3. 每回合車輛最大請求資訊量	1
4. 車輛移動速度	10.8 公里/小時 (1 網格長/秒)
5. 每單位網格大小	3 公尺*3 公尺
實驗參數	
6. Road_Number	5
7. Road_length	50
8. Vehicle_Density	0.2、0.45、0.7
9. Vehicle_Proportion	5%、10%、15%、20%、25%、30%
10. Clustering_Algorithm	SP、LCC、TBC

在上表 1 中將本實驗所使用到的所有初始參數分為兩大類，分別為系統參數與實驗參數。系統參數為每次實驗皆不會改變之參數，通常與系統能力有關，而實驗參數為在不同次實驗中接受調整並進行比較之參數。而兩類參數知詳細向牧與個別設定如下：

1. 每次實驗回合數：

每次實驗進行之回合數，實驗中之每一回合皆可以對應至真實環境之一秒。每次實驗會執行 1000 個回合模擬，也就是每一個車輛物件經過進行 1000 次移動與路況資訊請求後，即為一次實驗，可產生一組實驗結果。

2. 封包可傳遞距離：

在本模擬實驗中，具有 VANET 功能之車輛間所能進行封包傳遞之最大距離，超過此距離便無法在車輛間使用 VANET 功能傳遞封包，在實驗中將其依現有裝置之傳輸能力約在 150 至 250 公尺間，將此參數定為 200 公尺，並依網格大小進行轉換後使用在交通模型中。

3. 每回合車輛最大請求資訊量：
每一回合中，每個車輛物件最多可發出一次查詢路況資訊之請求，而收到此請求之車輛會依本身是否有該路段最近的路況資訊，決定直接傳回該路況資訊或繼續向其他車輛詢問。
4. 車輛移動速度：
在本實驗中，參考台北市主要幹道之交通流量（台北市交通管制工程處，2009），將車輛移動速度定為 20 公里/小時，再依網格大小進行轉換後使用在交通模型中。
5. 每單位網格大小：
在細胞自動機中每一個網格所對應之真實交通環境大小，並將其定為 3 公尺*3 公尺。
6. Road_Number：
道路總數量，在本實驗中將此參數設定為 5，意即垂直道路與水平道路各有 5 條。
7. Road_length：
道路長度，在本實驗中將此參數設定為 50，意即在兩個道路交叉口之間共有 50 個網格可供車輛物件行駛，而道路交叉口為垂直道路與水平道路交會處之網格。
8. Vehicle_Density：
在本實驗中，參考台北市主要幹道之交通流量（台北市交通管制工程處，2009），取尖峰與離峰時段車流量後，依 3.1.2 小節中之轉換公式將其轉換為車輛密度並得到尖峰時段車輛密度近似值 0.7 與離峰時段車輛密度

近似值 0.2，並取其中間值 0.45 並將其定義為中等車輛密度，在將此三數值做為實驗參數。

9. Vehicle_Proportion :

本參數依台灣 ITS 發展目標 (中華民國交通部運輸研究所, 2010)，將此參數其範圍最小值設定為 5%，而最大值設定為 30%，並在其中以每 5% 作為一個間隔。

10. Clustering_Algorithm :

在本實驗中，由於需要比較在不同使用動態路徑規劃之車輛比例中 TBC 演算法與 LCC 演算法與之差異。以 Vehicle_Proportion 為 10% 為例，將參數設定為 TBC 時，代表有 10% 之車輛進行使用 TBC 演算法叢集化之實驗，而將參數設定為 LCC 時，即代表有 10% 進行使用 LCC 演算法叢集化之實驗，而在實驗中其餘 90% 的車輛則是使用最短路徑(Shortest Path, SP)演算法，依路徑之權重進行道路規劃。

4.2 實驗結果

在本節中，將會就 3.3 節中所提出之平均叢集數目、叢集調整次數與平均封包傳遞時間等三項指標依序進行實驗，而在每一項指標之實驗中，實驗參數都會依照 4.1 節中所定義，對於 TBC 演算法與 LCC 演算法在各種車輛密度情形下，以各種車輛比例進行實驗。

4.2.1 平均叢集數目

在本小節中，將會在本實驗的每一回合結束前，計算目前交通環境中擁有幾台叢集領導者(Cluster Head)車輛，並且將其累加至 Cluster_Number 變數，因一個叢集領導者車輛即代表了一個叢集。在實驗結束後，將 Cluster_Number 變數除以總實驗回合數，即可得到平均叢集數量。

下圖 21 至 23 分別為在交通環境中車輛密度由低至高時，在不同車輛比例之情形下，使用 LCC 演算法與 TBC 演算法產生叢集數目之平均值。

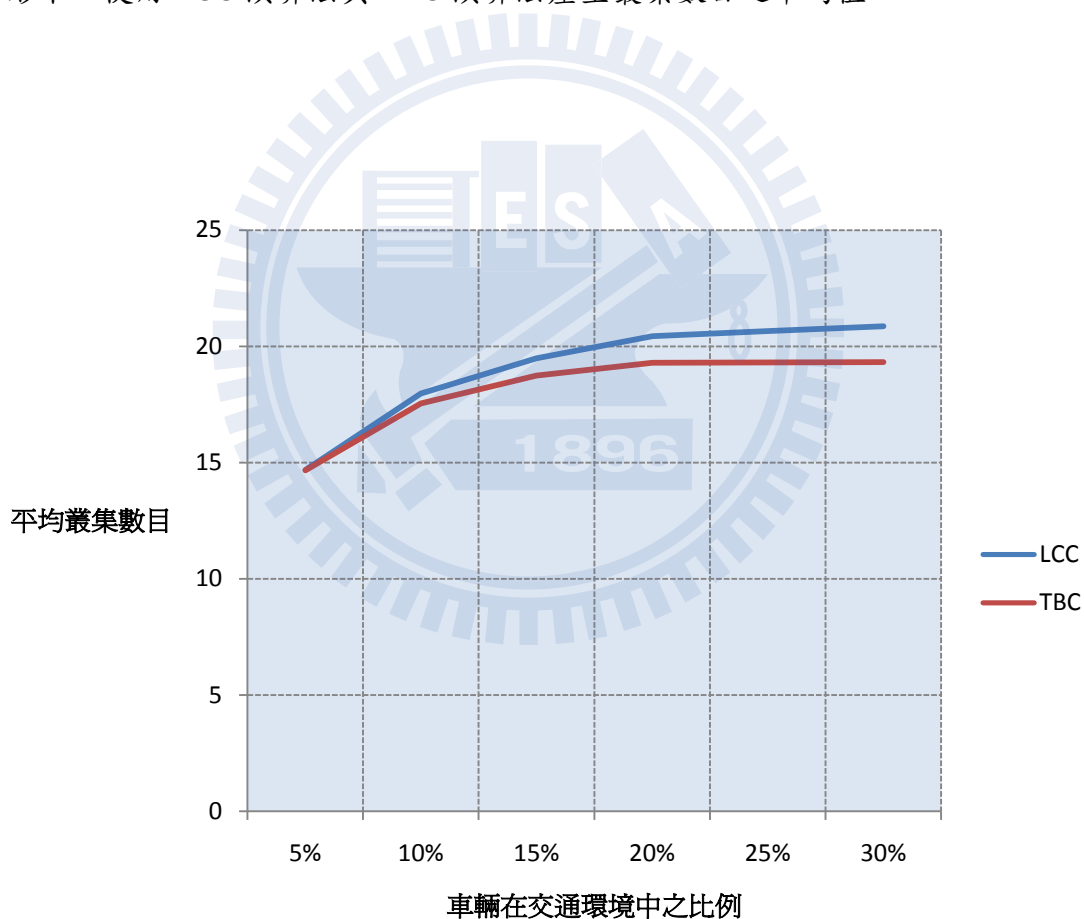


圖 21. 平均叢集數目(低密度)

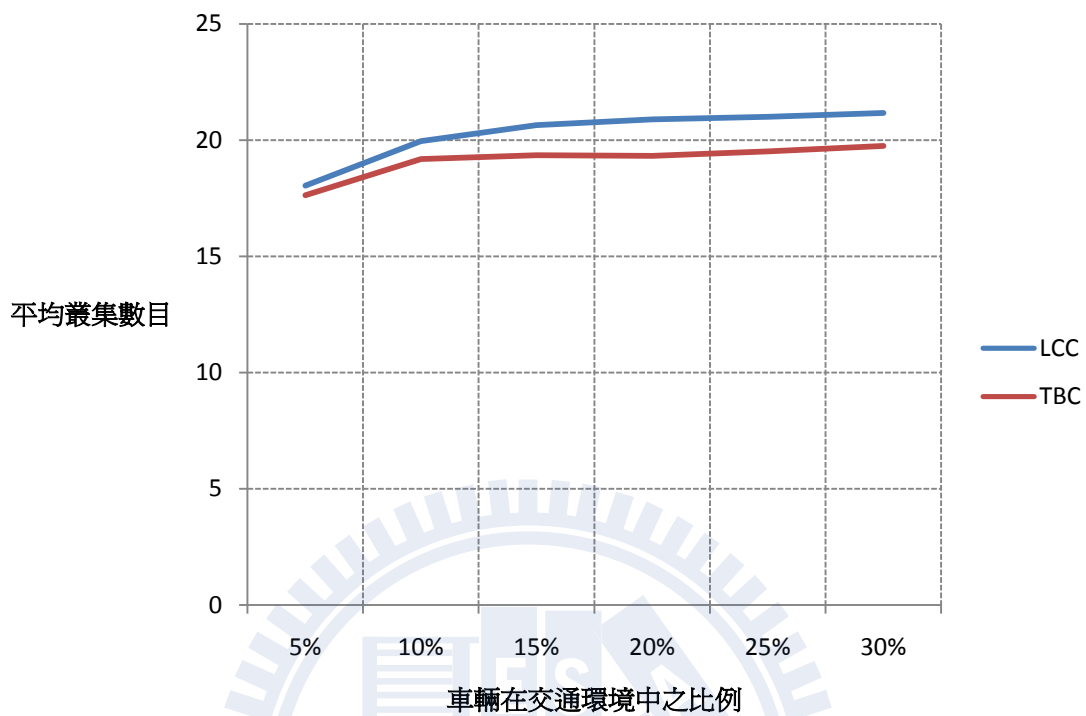


圖 22. 平均叢集數目(中密度)

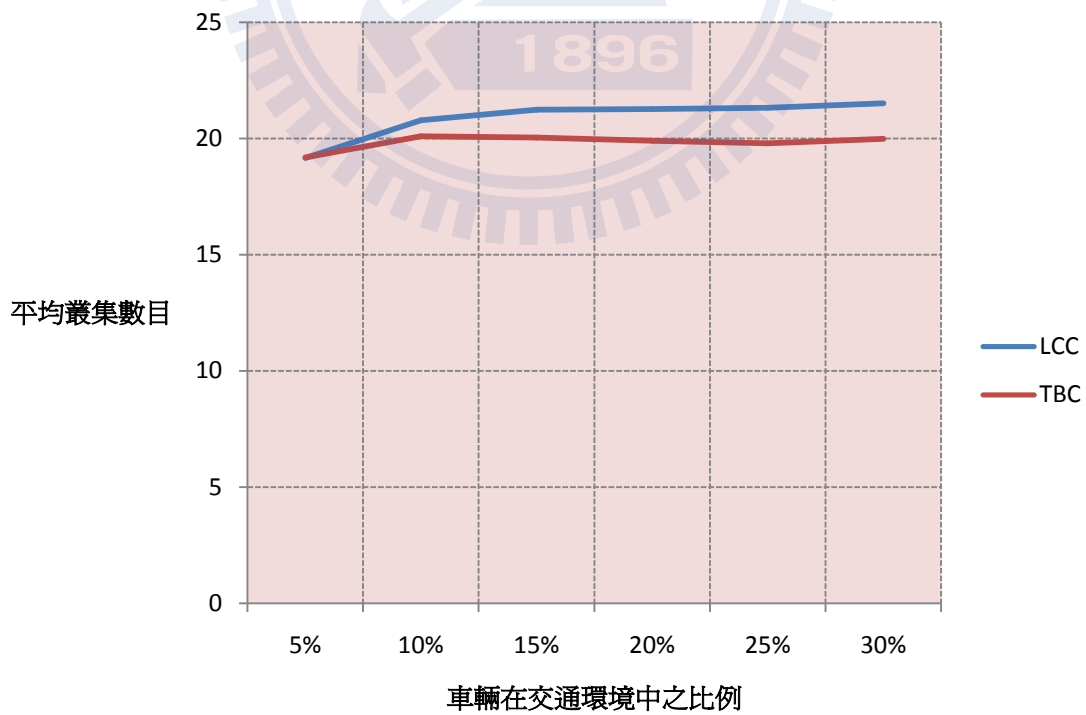


圖 23. 平均叢集數目(高密度)

在圖 21 至圖 23 中可以發現，無論在何種車輛密度與比例之實驗中，使用 TBC 演算法所產生之平均叢集數目皆較使用 LCC 演算法少，這是因為 TBC 演算法因為以代幣(Token)為基礎，選擇實際地理位置較佳或是已經擁有較新路況資訊之車輛作為叢集領導者，可以使叢集中的成員數增加，讓所形成的叢集較完整而不破裂，故可以降低環境中的叢集總數。

首先在圖 22 之中密度實驗結果中，可以發現當車輛比例在 10% 至 20% 之間時，平均叢集數量增加速度漸趨緩慢，但是當車輛比例超過 20% 之後，平均叢集數量會快速增加，然後再次減緩增加速度，這是因為當車輛比例較低時，由於使用動態路徑規劃方法之車輛將會避開一般使用 SP 演算法之車輛行駛路徑，故叢集數目只需要能覆蓋使用動態路徑規劃方法之車輛行駛之範圍即可，但是當車輛比例增加時，使用動態路徑規劃方法之車輛總數增加，而這些過多的車輛將無法避免行駛進入一般使用 SP 演算法之車輛行駛路徑，造成需要花費額外之叢集覆蓋此路徑，此情形為寡勝賽局(Minority Game)的一種實例 (Ohara, Nojima, & Ishibushi, 2007)。

在圖 23 中可以發現在高車輛密度的情形下，此寡勝賽局將會在車輛比例為 10% 至 15% 時出現，較中密度時更快出現由圖 22 與圖 23 可以推斷，因車輛比例乘以車輛密度所得之值大約與使用動態路徑規劃方法之車輛總數成正比，當此值超過一定之門檻後，使用動態路徑規劃方法之車輛將會進入一般使用 SP 演算法之車輛行駛路徑，造成叢集再次快速增加。

4.2.2 叢集調整次數

在本小節中，將會在本實驗的每一回合中，當叢集調整發生時，會將 Cluster_Re-formation 變數加 1。在實驗結束後，Cluster_Re-formation 變數即為總共發生之叢集調整次數。如同在 3.2.3 小節中所提到的，車輛在兩種情形下會進行叢集調整：

1. 產生叢集管理者：

當一個狀態為叢集成員的車輛之可傳輸範圍內不存在狀態為叢集管理者的車輛時，將會檢查自身所擁有的代幣數是否高於所有臨近車輛之代幣數，若是自身所擁有的代幣數較高時，則將自身狀態修改為叢集管理者。

2. 移除叢集管理者：

當一個狀態為叢集管理者的車輛之可傳輸範圍內存在狀態為叢集管理者的車輛時，將會檢查自身所擁有的代幣數是否高於該叢集管理者車輛，若是自身所擁有的代幣數較低時，則將自身狀態修改為叢集成員。

當叢集管理者改變的次數越少，則花費在管理叢集的封包數目則越少，也是在實驗中需要比較 TBC 演算法與 LCC 演算法在不同的比例下，所需要執行叢集調整之次數之原因。

下圖 24 至 26 分別為在交通環境中車輛密度由低至高時，在不同車輛比例之情形下，使用 LCC 演算法與 TBC 演算法產生叢集調整次數。

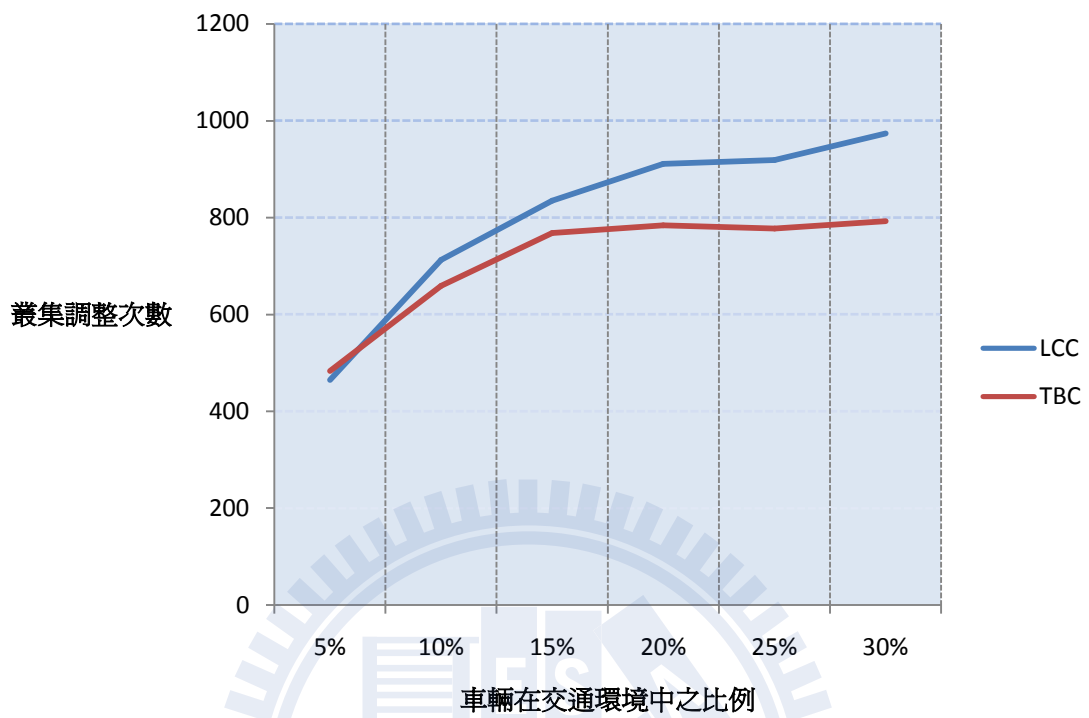


圖 24. 叢集調整次數(低密度)

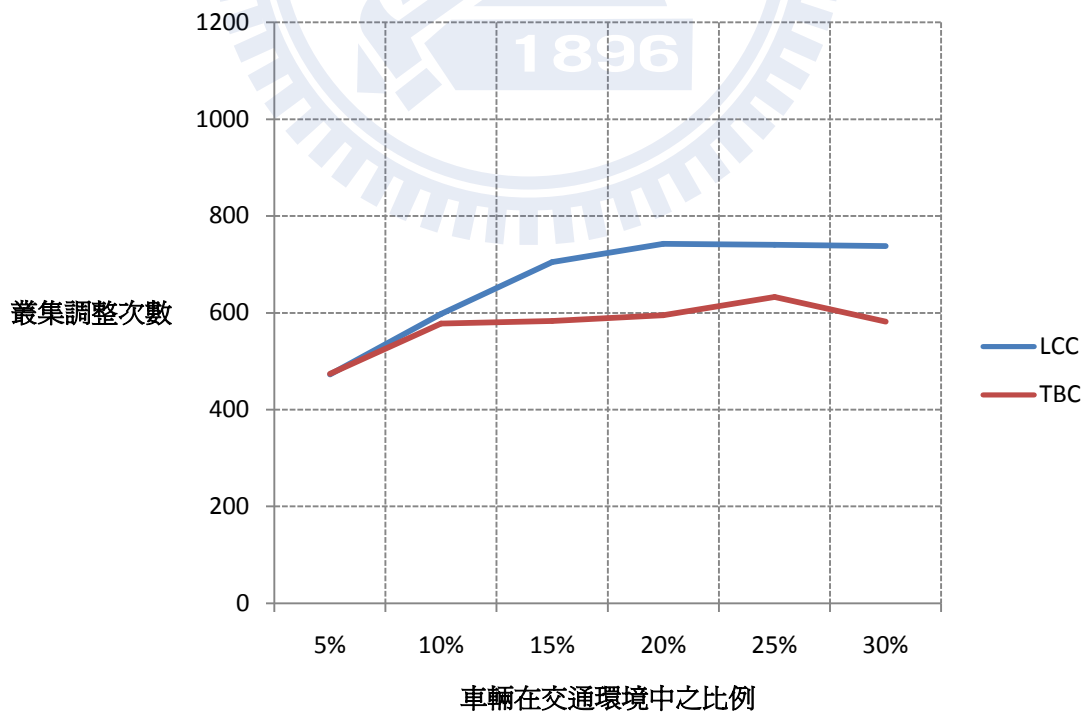


圖 25. 叢集調整次數(中密度)

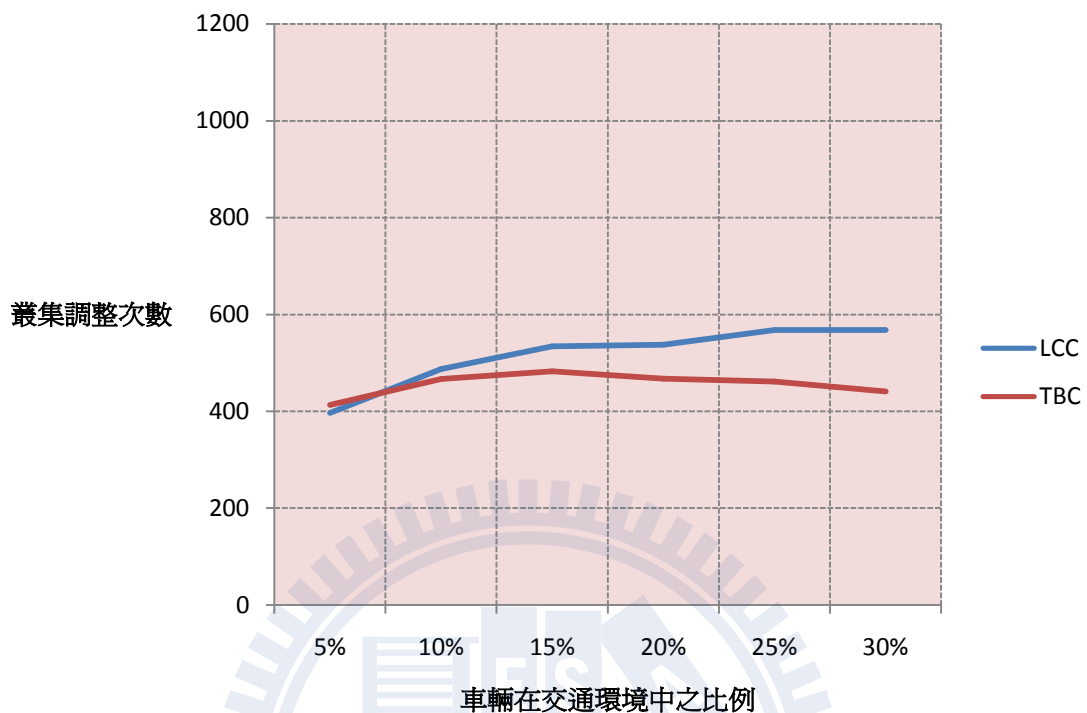


圖 26. 叢集調整次數(高密度)

從圖 24 至圖 26 中皆可以發現，無論是使用 TBC 演算法或是使用 LCC 演算法之車輛，隨著在模擬環境中的車輛比例提高，而叢集調整次數的成長都會趨於緩慢。此現象是因為當車輛密度與比例提高時，隨著叢集逐漸覆蓋所有道路後，叢集將趨於穩定不易變動。

由圖 24 至 26 中仍然可以發現，在與圖 21 至 23 中使用相同的參數設定時，同樣在一個短區間中叢集調整次數會漸漸減緩增加速度，但會突然大幅增加後，再次漸漸減緩叢集調整次數之增加速度，由此可知，本實驗在 4.2.1 小節中所發生之寡勝賽局現象再次出現，而發生時之參數設定也大致相同，驗證了對於車輛比例乘以車輛密度所得之使用動態路徑規劃方法之車輛總數，將會有一個門檻值存在。

由圖 24 可以得知，當車輛密度與比例都極低時，使用 LCC 演算法所產生的叢集調整次數少於使用 TBC 演算法，這是因為在較少車輛使用動態路徑規劃方法之情形下，車輛之間平均距離較大，不容易發生較小 ID 車輛進入其他叢集後產生之無意義叢集調整。而 TBC 演算法由於計算代幣，會造成計算上較複雜，且反而不如簡易的 LCC 演算法所產生叢集穩定。

雖然在車輛密度與比例較低時使用 LCC 演算法之效能較佳，但是當車輛密度或比例較高時，系統才會真正需要降低負擔。如圖 24 至圖 26 之實驗所示，當車輛密度或比例較高時，使用 TBC 演算法所產生的叢集調整次數少於使用 LCC 演算法，故在真正面對有降低系統負擔之需求時，TBC 演算法能較有效的降低叢集調整次數，使系統之額外負擔降低。

由於 TBC 演算法是使用 H-VADD 演算法，參考實際地理位置與資訊更新時間後，作為交換代幣之根據，且會依車輛之代幣數目決定有何車輛擔任叢集管理者，所以會選出最常收到與回應查詢路況請求之車輛作為叢集管理者，故相比於 LCC 演算法，TBC 演算法較不需要改變叢集管理者即可有效的減低通訊的額外負擔。

4.2.3 平均封包傳遞時間

在本小節中，將會在本實驗的每一回合中，當車輛物件發出查詢路況之請求封包時，每經過一個車輛物件的傳遞，並且讓封包進行一次跳躍(Hop)動作，就將 Packet_Hop_Number 變數之值加 1。在實驗結束後，將 Packet_Hop_Number 變數除以總共發出之路況資訊請求數目，即可得到平均封包傳遞長度。而在 VANET 環境中，因為在系統負載尚未超出處理能力前，回應每一個訊息的時間花費幾乎都會相同，故可在假定每台車輛之 OBU 效能相同的情形下，封包傳遞長度將與封包傳遞時間成正比，故可以使用封包傳遞長度作為計算封包傳遞時間之參考值。

下圖 27 至 29 分別為在交通環境中車輛密度由低至高時，在不同車輛比例之情形下，使用 LCC 演算法與 TBC 演算法產生平均封包傳遞時間。

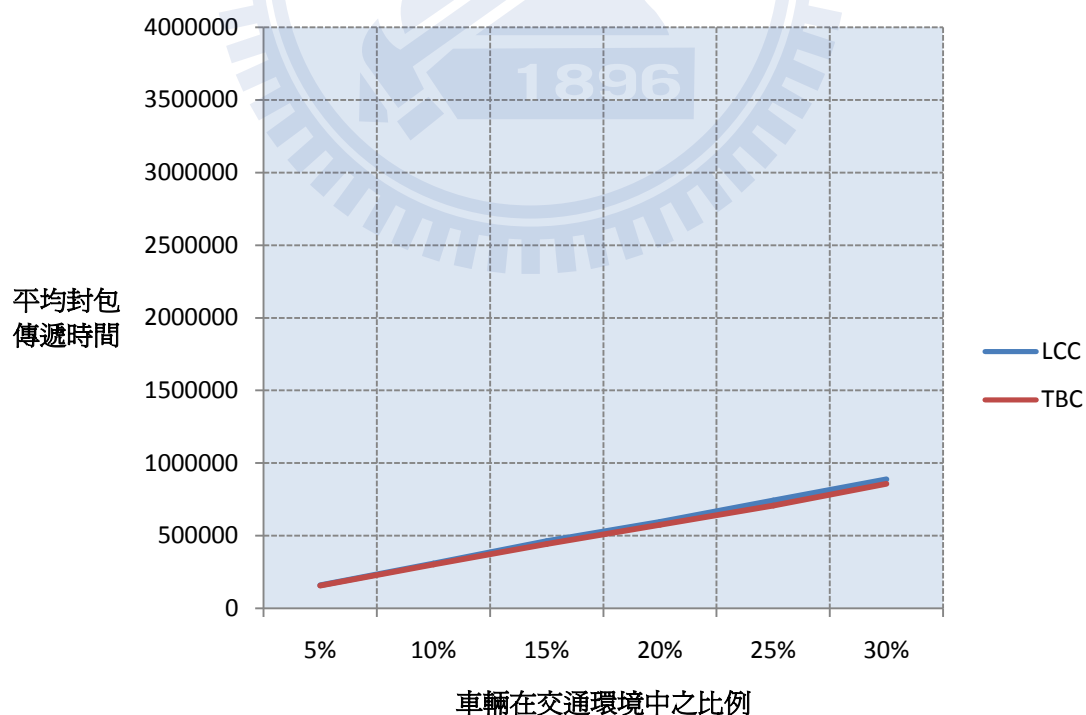


圖 27. 平均封包傳遞時間(低密度)

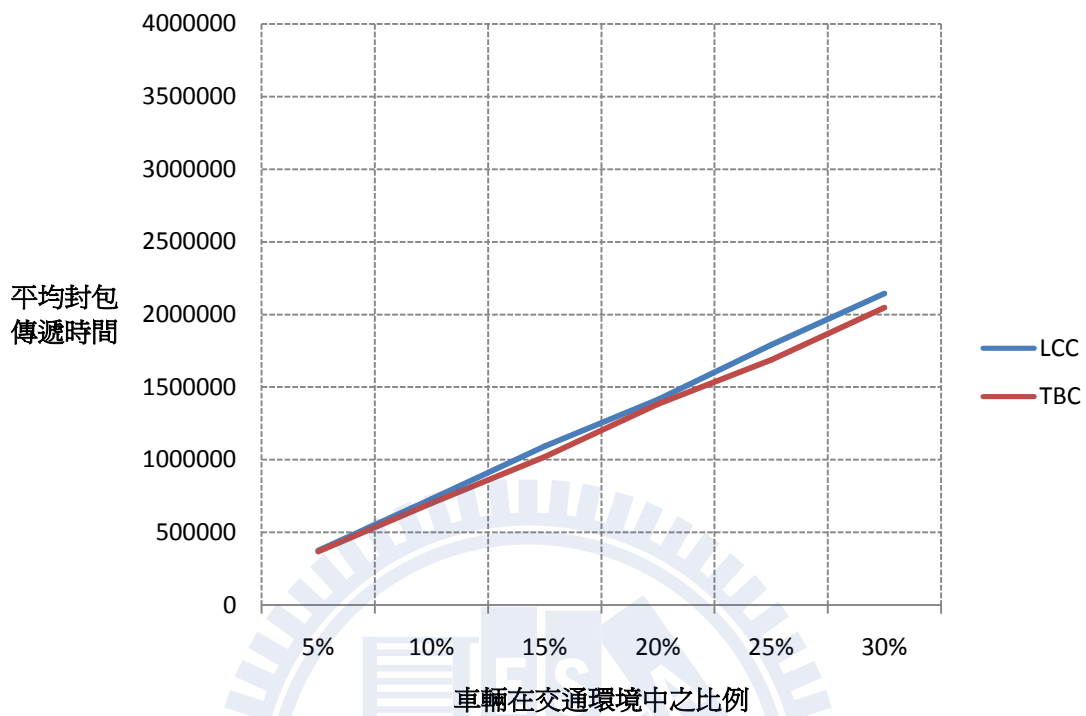


圖 28. 平均封包傳遞時間(中密度)

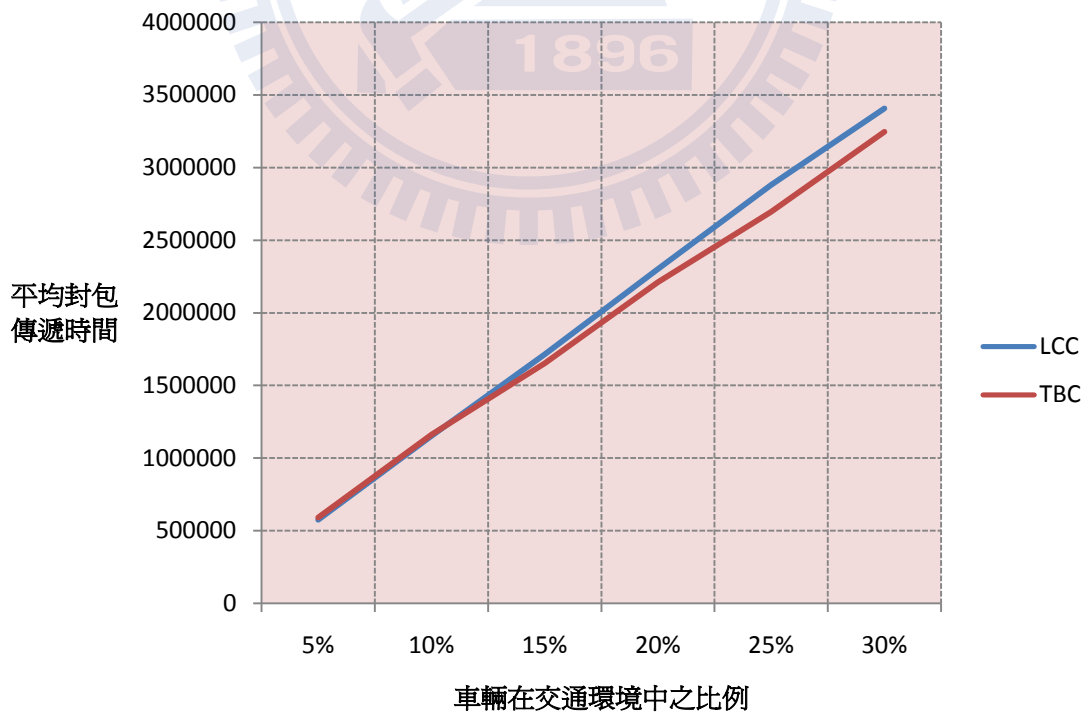


圖 29. 平均封包傳遞時間(高密度)

在封包傳遞過程中，通常只要每經過一個叢集，便會由接收到封包的叢集成員傳送給自身叢集之叢集管理者處理與判斷，之後再由叢集管理者挑選下一個叢集成員作為接下來的傳遞目標，產生至少兩次跳躍動作，若是叢集之間不存在一個共同的叢集開道，甚至會產生三次跳躍動作，故從固定的所在地至固定的目的地中，封包傳遞路徑長度取決於路徑上之叢集個數，即使封包傳送同樣的距離，若路徑上的叢集數量過多，仍然會造成較大傳遞上的時間延遲。

由圖 27 至圖 29 可以得知，平均封包傳遞時間與車輛密度及比例乘積成正比，當車輛密度與比例增加時，平均封包傳遞長度也會呈線性增加。在低密度情形下，使用 LCC 演算法與使用 TBC 演算法之效能相差不大，而在中密度情形下，在車輛比例低於 5% 時使用 LCC 演算法之效能亦較佳，同樣的在高密度環境中，當車輛比例低於 5% 時，也發現了使用 LCC 演算法之效能較佳，因在車輛總數較低時，使用 TBC 演算法無法形成完整而不破碎之叢集，且叢集調整規則較複雜，使得叢集容易產生不必要的變動，造成封包傳遞時找到一條固定而快速的跳躍路徑。

隨著車輛密度與比例增加，TBC 演算法仍能有效的降低封包傳遞時間，在高密度情形下相比 LCC 演算法可節省約 10% 之時間。

第五章 結論與貢獻

在本章中，首先將在 5.1 節中綜合整理在第四章中所實驗出的實驗結果，接下來將在 5.2 節中總結本實驗所有研究結果並提出結論，在 5.3 節中將就 5.2 節中所提出之研究結論提出本研究之貢獻，最後在 5.4 節中將會介紹本研究未來仍可以修正改進之處以及後續研究之方向。

5.1 研究結果綜合分析

在 3.3 節中已經介紹過各個評估指標以及選擇評估指標的原因與方式，並在第四章中對於各個指標實驗結果與原因進行分析，以下表 3 為綜合整理所有的實驗結果之表格。

表 2. 研究結果綜合整理

	Least Cluster Change	Token-Based Clustering
1-1. 平均叢集個數	多	少
1-2. 平均叢集成員個數	少	多
2. 叢集調整次數	低比例且低密度：少 高比例或高密度：多	低比例且低密度：多 高比例或高密度：少
3. 平均封包傳遞時間	低比例且低密度：短 高比例或高密度：長	低比例且低密度：長 高比例或高密度：短

由以上的綜合整理與本研究第三、四章，可以得知：

1. 在相同環境中，交易代幣式(Token-Based Clustering, TBC)演算法所產生

之叢集個數少於使用最少叢集改變(Lease Cluster Change, LCC)演算法所產生之叢集個數。

2. 在車輛密度與比例皆低之情形下，使用 TBC 演算法所產生之叢集調整次數較多，且平均封包傳遞時間亦較長。
3. 在車輛密度或比例較高時，使用 TBC 演算法所產生之叢集調整次數較少，且平均封包傳遞時間亦較短。因為 TBC 演算法之目標為減低管理叢集之負擔與減少封包傳遞時間，故在車輛密度與比例較低時，雖然 LCC 演算法之效能較佳，但此交通環境之負擔已經較低，能否再降低管理叢集之負擔與減少封包傳遞時間則不是最重要的問題。

5.2 研究結論

由第四章中的實驗結果，將其分為 TBC 演算法與 LCC 演算法兩者之共同現象與差異點，可以得知：

共同現象

1. 在交通環境中之平均叢集個數、叢集調整次數與平均封包傳遞時間大致上與車輛數目成正向關係，其中平均叢集個數與叢集調整次數會隨著車輛密度與比例的增加而減緩增加速度，而平均封包傳遞時間與車輛密度與比例成正比。
2. 無論是在交通環境中使用 TBC 演算法或是使用 LCC 演算法，當車輛密度與比例愈高時，平均叢集個數與叢集調整次數雖然亦會增加，但增加速度漸趨緩慢，故車輛密度與比例愈高時，對平均叢集個數與叢集調整次數效能提升愈多。

3. 在平均叢集個數與叢集調整次數的實驗中可以發現，在中等密度情形下，大約在車輛比例介於 10% 至 20% 時，圖型成長趨勢漸緩，但是在超過 20% 後圖形會再次快速遞增，然後再次隨著車輛比例與密度的增加而減緩圖型成長趨勢，這是因為寡勝賽局現象在通環境中出現。

在交通環境中有部分道路容易被未使用動態路徑規劃之車輛依其固定權重值而選擇，造成此道路固定較其他道路阻塞，當車輛密度與使用動態路徑規劃之車輛比例漸漸增加時，使用動態路徑規劃的車輛總數增加，而使這些車輛駛入平常不會進入的阻塞道路，而脫離原本叢集之覆蓋範圍，產生新的叢集。在這些阻塞道路都被叢集覆蓋後，平均叢集個數與叢集調整次數之增加速度便會再次遞減。

差異點比較

1. 在交通環境中，當車輛密度與比例愈高時，TBC 演算法與 LCC 演算法在平均叢集個數、叢集調整次數與平均封包傳遞時間等指標比較中之差異程度變愈明顯。
2. TBC 演算法較適合使用在比例或密度較高的交通環境，而雖然 LCC 演算法在車輛比例與密度較低時效能較佳，但是在較不繁忙的交通環境中，也較不需要考慮如何減低系統的負擔。

5.3 研究貢獻

本實驗對一般經由 VANET 進行叢集化的車輛加入 TBC 演算法，將車輛之實際地理位置與資訊更新時間數值化，轉換為代幣(Token)機制，讓車輛可以花費代幣購買路況資訊，並可藉由代幣的轉換流動，讓車輛間可以互相評估挑選叢

集管理者，而不再只是經由靜態的車輛 ID 或是容易變動的節點分支度(Degree) 進行評估。

在實驗中，可以發現在車輛密度與比例較高的環境中，使用 TBC 演算法之車輛可以降低調整叢集負擔與封包傳遞時間，以此叢集化技術應用在 VANET 環境中，即使在都市中車輛日漸增加，也可以使用較低負擔且有效率的動態路徑導航。

因本研究以交易為基礎進行叢集化，而與交易相關的經濟理論也已經是成熟且蓬勃發展的學科，具有相當大量的理論與機制可以使用，若是加入更多經濟理論進入系統中，可以為未來的後續研究提供一個嶄新的觀點。

5.4 未來展望

5.4.1 模擬交通模型

在本實驗中所採用的模擬交通模型，使用了細胞自動機(Cellular Automata)作為基礎模型的概念。雖然細胞自動機適合用來模擬簡化的交通模型，也有相當多的實驗採用細胞自動機作為基礎模型，並相當具有準確性，但在實際交通網路中，由於道路的配置也會直接的影響到車流量等交通情形，所以可以再改用實際的道路配置建立模擬交通模型。

由於細胞自動機明顯具有離散特性，故無法精確模擬在任何時間點中交通的狀態。在本研究的模擬中，由於只是在演算法層次中對於 TBC 演算法以及 LCC 演算法進行比較，所以使用細胞自動機作為基礎模型也能提供有參考價值的模擬結果。若是需要進行更加接近實際情形的模擬，可以再使用

VanetMobiSim(VanetMobiSim, 2010)或是 SUMO (DLR, 2010)等交通模擬軟體配

合網路協定模擬軟體 NS2 (The Network Simulator - ns-2, 2010)，輸入實際的路況資料進行實驗。

5.4.2 演算法調整

由第四章中的實驗結果得知，TBC 演算法在車輛密度或比例較低的時候，無論是在叢集個數、叢集調整次數或是封包跳躍次數等指標，效能皆無法與 LCC 演算法有明顯差距，在車輛密度與比例都極低時，TBC 演算法所產生之叢集調整次數甚至會明顯高於 LCC 演算法。

面對此問題，首先可以承接上一小節，蒐集各地的實際的交通路況資訊，並且建立可以完整對應至真實路況情形的模擬交通模型。當建立完模擬交通模型後，可以在模型中測量在不同地區、不同車輛比例或不同車輛密度的情形下，何種時候較適合使用 LCC 演算法，而何種時候又較適合使用 TBC 演算法。若是擁有以上的測試資料，車輛就可以自行決定在不同情形下選擇使用何種演算法。

TBC 演算法將車輛所處之地理位置與資訊更新時間數值化，並引進日常生活常見的交易行為與經濟概念，作為叢集化時的參考依據，能為將來的研究開啟一種新的觀點。以賽局理論為例，若是在車輛間加入賽局理論，讓車輛可以選擇封包傳輸策略，並且在其中尋找納許平衡(Nash Equilibrium)與以此納許平衡持續修正與調整封包傳輸策略。

5.4.3 代幣檢驗機制

若是在交通環境中使用 TBC 演算法作為傳遞應用服務封包的基礎方式，由於各個成員發出的大部分資料都會經過叢集管理者(Cluster Head)分配後，再決定傳遞封包對象，故可能會有惡意的駕駛人修改本身代幣數目，讓本身車輛更容易

成為叢集管理者，以得到分配鄰近區域封包之權力。

若是產生以上所提到之惡意駕駛人與車輛，經由修改本身代幣方式成為叢集管理者，由於在鄰近區域中，大部分封包都會經過此惡意叢集管理者，使得惡意叢集管理者較容易影響區域的資訊傳遞。以動態導航應用為例，此惡意叢集管理者可能會依本身計劃行駛的道路，發送此道路為阻塞的路況資訊給其他車輛，讓大部分車輛都避開此道路行駛，讓惡意叢集管理者自己擁有比其他車輛更順暢的行駛環境。

針對這種情形，建議可以使用週期性的檢驗代幣數目機制作為防範方式。每隔一段時間，車輛可以使用本身的交易歷程，統計出最近一段時間內對各個不同車輛的交易額，意即對於各個不同車輛的代幣流入或流出數目，然後車輛間彼此交換各自的交易額，便可以依此交易額數目檢驗附近車輛是否產生不正常的代幣變化，並將這些車輛視為惡意使用者，將其車上設備(OBU)之 ID 加入黑名單中。

參考文獻

- Basu, P., Khan, N., & Little, T. (2001). A Mobility Based Metric for Clustering in Mobile Ad Hoc Networks. *Distributed Computing Systems Workshop*, (pp. 413-418).
- Bechler, M., & Wolf, L. (2005). Mobility Management for Vehicular Ad Hoc Networks. *IEEE 61st Vehicular Technology Conference*, (pp. 2294-2298).
- Blum, J. J., Eskandarian, A., & Hof, L. J. (2004, 12). Challenges of Intervehicle Ad Hoc Networks. *IEEE Transactions on Intelligent transportation Systems*, 5 (4), pp. 347-351.
- Chiang, C.-C., Wu, H.-K., Liu, W., & Mario, G. (1997). Routing in Clustered Multihop, Mobile Wireless Networks with Fading Channel. *The IEEE Singapore International Conference on Networks*. Singapore.
- DLR. (2010, 5 26). Retrieved 5 26, 2010, from <http://www.dlr.de/ts>
- Dube, R., Rais, C. D., Wang, K.-Y., & Tripathi, S. K. (1996, 8). Signal stability based adaptive routing (SSA) for ad-hoc mobile networks. *Technical report CS-TR-3646*. University of Maryland, College Park.
- Enge, P., Wailter, T., Pullen, S., kee, C., Chao, Y.-C., & Tsai, Y.-J. (1996, 8). Wide area augmentation of the global positioning system. *Proceedings of the IEEE*, 84 (8), pp. 1063-1088.
- Ephremides, A., Wieselthier, J. E., & Baker, D. J. (1987). A Design Concept for Reliable Mobile Radio Networks with Frequency Hopping Signaling. *Proceeding of the IEEE*, pp. 56-73.
- Gerla, M., & Tsai, J. T. (1995). Multiuser, Mobile, Multimedia Radio Network. *Wireless Networks*, 1, pp. 255-265.
- Ghassemian, M., Friderikos, V., & Aghvami, H. (2005). Hybrid handover in multihop radio access networks. *IEEE Vehicular Technology Conference*, (pp. 2207-2211).

Hein, G. (2000). From GPS and GLONASS via EGNOS to Galileo—Positioning and navigation in the third millennium. *GPS Solutions*, 3 (4), pp. 39-47.

Intelligent Transportation. (2010, 5 26). Retrieved 5 26, 2010, from Intelligent Transportation Systems of America: <http://www.itsa.org/>

Johnson, D. B., & Maltz, D. A. (1996). Dynamic Source Routing in Ad Hoc Wireless Networks. *Mobile Computing*, (pp. 153–181). Norwell, MA.

Karp, B., & Kung, H. T. (2000). GPSR: Greedy perimeter stateless routing for wireless networks. *ACM MOBICOM*, (pp. 243–254).

Li, Q., & Rus, D. (2000). Sending Messages to Mobile Users in Disconnected Ad-hoc Wireless Networks. *ACM Mobicom*, (pp. 44–55).

Lim, H., & Kim, C. (2000). Multicast Tree Construction and Flooding in Wireless Ad Hoc Networks. *ACM International Workshop on Modeling, Analysis and Simulation of Wireless and Mobile Systems (MSWIM)*, (pp. 61-68). Boston, Massachusetts, USA.

Little, T., & Agarwal, A. (2005). An Information Propagation Scheme for VANETs. *IEEE Intelligent Transportation Systems* , pp. 155-160.

Nadeem, T., Shankar, P., & Iftode, L. (2006). A Comparative Study of Data Dissemination Models for VANETs. *Mobile and Ubiquitous Systems Workshops*.

Ochieng, W. Y., & Sauer, K. (2003). GPS integrity and potential impact on aviation safety. *The Journal of Navigation*, 56 (1), pp. 51-65.

Ohara, K., Nojima, Y., & Ishibushi, H. (2007). A Study on Traffic Information Sharing Through Inter-Vehicle Communication. *IEE International Symposium on Intelligent Control Part of IEEE Multi-conference on Systems and Control*, (pp. 670-675). Singapore.

Safespot. (2010, 5 26). Retrieved 2010, from Safespot: <http://www.safespot-eu.org/>

Skog, I., & Handel, P. (2009, 3). In-Car Positioning and Navigation Technologies—A Survey. *IEEE Transactions on Intelligent Transportation Systems* , pp. 4-21.

The Network Simulator - ns-2. (2010, 5 26). Retrieved 5 26, 2010, from <http://www.isi.edu/nsnam/ns/>

Tseng, Y.-C., Ni, S.-Y., Chen, Y.-S., & Sheu, J.-P. (2002). The Broadcast Storm Problem in a Mobile Ad Hoc Network. *Wireless Networks*, 2, pp. 153-167.

Vahdat, A., & Becker, D. (2000). *Epidemic Routing for Partially-Connected Ad Hoc Networks*. Durham, NC.: Technical report, Department of Computer Science, Duke University, .

VanetMobiSim. (2010, 5 26). Retrieved 5 26, 2010, from <http://vanet.eurecom.fr/>

Yu, J. Y., & Chong, P. H. (2005). A Survey of Clustering Schemes for Mobile Ad Hoc Networks. *IEEE Communications*, 7 (7), pp. 32-48.

Zhao, J., & Cao, G. (2008, 5). VADD Vehicle-Assisted Data Delivery in Vehicular Ad Hoc Networks. *IEEE Transactions on Vehicular Technology*, 57 (3), pp. 1910-1922.

中華民國交通部運輸研究所. (2010). 交通部運輸研究所中文版首頁. 2010 年 5 月 29 日擷取自交通部運輸研究所: <http://www.iot.gov.tw/mp.asp>

台北市交通管制工程處. (2009). 九十八年度台北市交通流量及特性調查(一般路口、圓環路口及路段調查部分). 台北市: 台北市交通管制工程處.