車載社群網路中一個擁有社會意識以軌跡為基礎的路由協定

章晴茹
國立彰化師範大學
資訊工程研究所
wendychang0104@gmail.com

張英超
國立彰化師範大學
資訊工程學系
icchang@cc.ncue.edu.tw

廖啟盛
國立彰化師範大學
資訊工程研究所
suboy@hotmail.com.tw

摘要

因近年來社群網路的崛起，加入社群概念的車輛網路路由協定陸續被提出，這些路由協定藉由計算車輛和封包目的地的社交關聯度，來選擇適合的轉傳車輛。但此類的路由協定大多沒有考慮到時間對於社交關聯度的影響，也沒有運用到車輛行駛的軌跡資訊，導致在封包傳送的效率上仍有提升空間。因此本文設計一個「車輛社群網路上具有社會意識，以車輛軌跡為基礎的路由協定(SATR)」將先前我們提出的運用車輛軌跡資訊的路由協定，結合社群以及時間區間的概念，設計一個適用於車輛社群網路的封包轉傳協定，改善傳統車輛社群網路沒有考慮不同時間社交關聯性的缺點，並分析車輛於不同時段的社交關聯性與車輛軌跡，設計新的車輛相遇序列圖架構，降低傳統車輛軌跡分析的計算複雜度。最後以 The ONE (The Oppor-tunistic Network Environment simulator)為模擬工具，執行網路模擬實驗，得到 SATR 比傳統車輛社群網路和車輛軌跡路由協定具有更高的封包到達率與更低的傳輸延遲。

關鍵詞：車輛社群網路、社交關聯度、車輛軌跡、車輛相遇序列、封包轉傳路徑圖

1. 簡介

車輛社群網路（Vehicular Social Network, VSN）[1] 是車輛隨意網路（Vehicular Ad Hoc Network, VANET）[2] 結合社群概念形成的一種新型式網路。傳統的車輛隨意網路，其路由協定利用多跳（multi-hop）的方式將封包從傳送端轉傳至目的端，轉傳的方式是有技巧的選擇合適的轉傳節點來提高封包成功傳送的比率；因為攜帶封包的車輛不一定會行駛到封包的目的地，必須透過行駛途中遇見的車輛，以車輛之間多跳躍轉傳(multi-hop forwarding)的方式將封包傳往下一個中繼車輛節點(relay node)，最終傳到目的地車輛節點。傳輸時中繼車輛的選擇上，會依據經過計算分析後，車輛和封包目的地的社交關聯度高低來做選擇。

因為車輛的未來移動軌跡與之前的軌跡有很大的相關性[3]，如果能夠得到車輛未來整趟行程的移動軌跡，預先選定一條行駛機率最高的路徑，便可以大幅改善封包的傳送率，進而求出與其它車輛可能的相遇機率，選擇最合適的車輛轉傳封包，直到封包傳到目的車輛。

近年來因社群網路的崛起，將車輛網路引入社群概念提出的協定[4-11]也日益增加。在車輛社群網路中，車輛的行駛模式就好比人在社群網路中的興趣，經過長期的統計可找出和自身關聯性較高的車輛。在車輛社群網路中，是由許多擁有相似興趣的車輛形成的許多小社群所組成的，相似興趣的辨別方式是藉由車輛行駛模式的相似性來做為依據判斷。因此，以社交為基礎的路由協定必須要能夠分辨出不同車輛間是否有相似興趣。

本文目標是使用歷史車輛行駛軌跡，分析車輛於不同時間的社交關聯性，以車輛相遇序列圖的架構，設計一個可適用於車輛社群網路的封包轉傳協定，稱為 Socially-Aware Trajectory-based Routing (SATR)轉傳協定。

本篇論文貢獻有

- 分析車輛於不同時段的車輛軌跡與社交關聯性。
- 提出適用於車輛社群網路的封包轉傳協定 SATR，分為二階段流程，第一階段是「車輛軌跡預測階段」，第二階段是「封包轉傳階段」，改善傳統車輛軌跡路由協定的高計算複雜度與車輛社群網路沒有考慮不同時間社交關聯性的缺點。
- 提出演算法，得出「車輛軌跡預測階段」

車輛相遇序列圖與車輛轉傳路徑圖，並事先計算出在不同時間區段內，車輛行駛經過的地點、中繼車輛與分配給中繼車輛的封包複本 Token 數目，計算複雜度上較傳統車輛軌跡路由協定的軌跡樹大幅度降低。

- 提出演算法，得出「車輛軌跡預測階段」車輛相遇序列圖與車輛轉傳路徑圖，並事先計算出在不同時間區段內，車輛行駛經過的地點、中繼車輛與分配給中繼車輛的封包複本 Token 數目，計算複雜度上較傳統車輛軌跡路由協定的軌跡樹大幅度降低。由於 SATR 利用歷史車輛行駛記錄，計算出每台車輛和其他車輛與行經地點間的社交關聯性，因此可以有很高的機率可以預測出未來行駛經過的地點與可以轉傳封包到目的地的中繼車輛，封包到達率可大幅增加，降低封包傳輸的延遲時間。同時避免傳統轉傳協定散布大量封包複本給許多未能真正協助封包轉傳到目的節點的相遇節點，造成傳播這些無效封包複本的網路額外負擔。
- 提出「封包轉傳階段」車輛轉傳路徑圖預測不準確時更為簡單有效的例外補救措施，增加封包到達率，降低複雜度。
  增加資料儲存裝置 Throwbox 的設置，提升封包傳輸效率。

本篇論文的章節編排如下：第二段將討論在車輛網路上使用軌跡資訊預測的作法，以及社群網路上利用節點之間互動的資訊選出適合之轉傳節點的方法。第三段則介紹本文提出的方法，包括建立車輛相遇序列圖，接著利用序列圖比較判斷來建立封包轉傳路徑，以及如何根據封包轉傳路徑設計出高效率的路由協定。第四段顯示我們的方法和其他方法在模擬的比較結果。最後是結論。

2. 相關研究

2.1 利用車輛軌跡資訊的封包轉傳協定

在車輛網路上，有許多的論文運用車輛的歷史軌跡來預測車輛的未來軌跡，來得到車輛的未來行動性的資料傳輸演算法。但是此類的論文在軌跡預測階段的計算複雜度普遍較高，當預測失準時所付出的代價也相當大。我們過去提過的 NTR(Novel Trajectory Routing)[12]首先建立新的軌跡樹架構，較傳統決策樹多記錄了時間戳記與平均車速等資訊，將車輛軌跡進行分析，剔除出現機率較小的軌跡資料建立軌跡樹（如圖 1 所示，圖中為部分軌跡樹），接著進行車輛的軌跡樹比對，從樹根的第一個子節點開始逐一比對，若兩軌跡樹上的路口相同以及經過路口的時間有重疊，則將該路口以及重疊的時間區間記錄到 BranchItem 內暫存；接著繼續比對兩軌跡樹該路口的子節點，直到不同為止。最後利用 BranchItem 內的路口及相遇時間資料建立車輛相遇樹，計算出和相遇樹中每輛車相遇的機率，接著把車輛相遇樹轉換成封包轉傳序列，根據轉傳序列的相遇機率分配每輛車可得到的 Token 值。其優點在於充分運用歷史車輛軌跡資訊，將其進行分析，選擇適合的轉傳車輛。缺點在於軌跡樹比對上的計算複雜度太高，一但預測失準就必須再重新進行比對，重新比對後的封包轉傳序列圖也有再一次失準的可能，計算時間空間複雜度較高，對於封包傳送的延遲上會有較大影響。

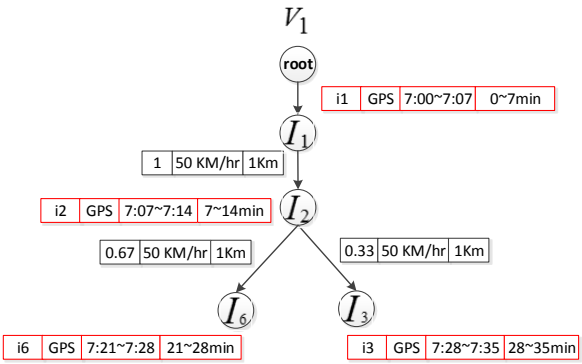


圖 1. NTR 軌跡樹範例

2.2 車輛社群網路的封包轉傳協定

此部分的論文大部分路由都是先計算每個節點的 Degree Centrality、Closeness Centrality 以及 Betweenness Centrality 值後，再透過這些 Centrality 值，配合自己提出的各種權重值算法，並以此來選擇資料傳輸時的中繼節點[1]。

Hoten (HOTspot ENtropy)[11]將網路環境切割成 k 個區域，每個區域稱為一個熱點(Hotspot)，依照每個行動節點內建的 GPS 所收集到的移動軌跡資料(Stay Points)，計算出每個熱點 h 的熱門程度w_h及受到特定節點p_i喜好的程度$w_{p_i}^h$，其算法如下：

$$w_h = \frac{n_h}{\sum_{h=1}^{k} n_h}$$
$$w_{p_i}^h = \frac{n_{p_i}^h}{\sum_{h=1}^{k} n_{p_i}^h}$$

其中w_h和$w_{p_i}^h$分別為公眾熱點(Public Hotspot)以及個人熱點(Personal Hotspot)的權重值，n_h為在熱點h內的停留次數，$n_{p_i}^h$為節點p_i在熱點h內的停留次數。接著透過w_h和$w_{p_i}^h$計算出每個節點i的Betweenness centrality、Personality值及與任一節點j之間的Similarity值，其算法如下：

Betweenness centrality：

$$C_b^i = (\sum_{h=1}^{k} w_{p_i}^h log \frac{w_{p_i}^h}{w_h})^{-1}$$

Personality：

$$Per_i = -\sum_{h=1}^{k} w_{p_i}^h log\, w_{p_i}^h$$

Similarity：

$$Sim(i,j) = \left(Sim\left(\frac{i}{j}\right) + Sim\left(\frac{j}{i}\right) \right)^{-1},$$

其中

$$Sim\left(\frac{i}{j}\right) = \sum_{h=1}^{k} w_{p_i}^h log \frac{w_{p_i}^h}{w_{p_j}^h}$$
$$Sim\left(\frac{j}{i}\right) = \sum_{h=1}^{k} w_{p_j}^h log \frac{w_{p_j}^h}{w_{p_i}^h}$$

透過Betweenness centrality、Personality和Similarity值，計算出每個節點和封包目的節點dst的*Hoten*值，其算法如下：

$$Hoten_i(dst) = \left[\frac{C_b^i}{C_b^i + C_b^j} + \frac{Per_i}{Per_i + Per_j} + \frac{Sim(i,dst)}{Sim(i,dst) + Sim(j,dst)} \right] \times \frac{1}{3}$$

在轉傳階段，當一個節點i攜帶著封包m遇到任意一個節點j時，若j為m之目的dst，i則將m傳給j；若否，則依照Hoten值大小決定i是否要將m傳給j，假如$Hoten_i(dst) < Hoten_j(dst)$，則$i$將$m$傳給$j$，反之則$i$繼續帶著$m$，直到遇到$Hoten(dst)$大於$Hoten_i(dst)$的節點為止。

Hoten的缺點在於沒有考慮到時間的問題，在不同的時間區間中，節點和同地點的社交關聯度會有所不同，如公司員工和辦公大樓在白天時和晚上時的社交關聯度會有所不同。

DFNMT(Data Forwarding algorithm based on Node Moving Trajectory)[13]將網路環境分成$M-1$個互不重疊的區域$C_1, C_2, \dots, C_{M-1}$，當節點u待在任一個區域C_i的累積時間超過β_1時，代表$u \in C_i$，因此一個節點可以同時屬於多個區域或者不屬於任何區域，一個區域也可以擁有許多節點。接著利用半馬可夫鍊[27]的概念，收集每個節點過去在所有地點停留的次數換算成停留機率，並將其記成矩陣$\bar{\pi}^u$，以及在下個時間區間，從目前停留的地點A到下個停留地點B的次數，將其化成轉移矩陣P，以便用來預測節點在未來可能出現的地方。算出轉移矩陣P後，我們將把目前節點出現在各區域的機率矩陣$\bar{\pi}^u = [\bar{\pi}_1^u, \dots, \bar{\pi}_{M-1}^u]$乘上轉移矩陣$P$，即可得到未來的節點移動到各個社群的機率，其中$\sum_{i=1}^{M-1} \bar{\pi}_i^u = 1$。而當節點$u$已在狀態$i$中停留$s$個時間區間，再經過$k$個時間區間後，停留在狀態$j$的機率$\hat{\phi}_{ij}^u(k,s) = \frac{\Pr(Z_{s+k}^u = j, t_{sojourn} = s | Z_0^u = i)}{\Pr(t_{sojourn} = s | Z_0^u = i)}$。接著定義了$Thd_u$及$MTsim$，用來選出適合的轉傳節點。$Thd_u$為節點$u$的移動軌跡之熱門度，計算方法如下：

$$Thd_u = \sum_{i=0}^{M-1} Chd_i \cdot \pi_i^u$$

其中Chd_i為社群i之中的節點數；π_i^u為節點u的穩態分布，而$\pi_i^u = \frac{d_i^u \bar{\pi}_i^u}{\sum_{j=0}^{M} d_j^u \bar{\pi}_j^u}$，$d_i^u$代表節點$u$在社群$i$的平均停留時間。

$MTsim$為兩節點u和v之移動軌跡相似度，算法如下：

$$MTsim(u, v, cur, s_1, s_2, T_{cur}) = \frac{1}{\delta} \sum_{i=1}^{\delta} sim(\hat{\phi}_{cur}^u \left(s_1, i \cdot \left\lfloor \frac{T_{cur}}{\delta} \right\rfloor\right), \hat{\phi}_{cur}^v \left(s_2, i \cdot \left\lfloor \frac{T_{cur}}{\delta} \right\rfloor\right))$$

上述公式中T_{cur}為封包存活時間，δ為時間區間總數，sim則為兩機率的cosine distance，算法同計算兩向量夾角餘弦值之公式。當$MTsim$值越小，代表兩節點移動軌跡相似度越低。

在封包傳送方面，假設目前節點u攜帶著目的地為$dest$的封包p，在移動的過程當中遇到了節點v，將根據以下不同情況執行動作：

(1) 若節點v等於$dest$，則u直接將p傳給v。

(2) 若$Thd_v \geq \beta_2 \cdot \frac{1}{N_{meet}} \sum_{i=1}^{N_{meet}} Thd_i$，也就

是說Thd_v大於或等於節點u過去所遇到過的節點其Thd值的平均，再乘上自訂的門檻值β_2，則u也直接將p傳給v。

若$Thd_v \geq \beta_3 \cdot \frac{1}{N_{meet}} \sum_{i=1}^{N_{meet}} Thd_i$，是指說$Thd_v$大於或等於節點$u$過去所遇到過的節點其$Thd$值的平均再乘上自訂的門檻值$\beta_3$，其中$\beta_2 > \beta_3$，則會再透過$MTsim(u, v, cur, s_1, s_2, T_{cur})$比較節點$u$和$v$的移動軌跡相似度，若$MTsim(u, v, cur, s_1, s_2, T_{cur}) \leq \lambda$，代表$u$和$v$的移動軌跡相似度小於自訂的門檻$\lambda$，在設定的容許範圍內，則$u$會直接將$p$傳給$v$。

BEEINFO 利用蜜蜂採花的概念，將節點比喻為蜜蜂，節點行經的地點比喻為花，提出一個在車輛社群網路環境下的路由協定。節點行經地點的社群密度計算公式如下：

$$DC(t) = \sum_{t=0}^{T} n$$

上述公式中，T代表的是時間區間長度，n代表在時間區間T內所遇到的節點數量。當節點在移動中經過各個地點時，即是透過第點的社群密度大小選擇轉傳點。

BEEINFO 的缺點在於沒有將時間確實分成特定的時間區間，也沒有長期收集車輛行駛資訊進行分析，找出適合的轉傳車輛，而是只根據每次行駛時所統計到的資料進行轉傳車輛選擇。

2.3 Throwbox 的設置

Throwbox 的設置，讓無法將攜帶封包的傳送至目的地的車輛將封包暫時存放，待有機會將封包傳至目的地車輛經過時，再將封包交給該車輛，更有效分散 Token，提高封包轉傳效率，再加上最多封包複本數N_{max}的限制，減少網路資源的消耗，降低負荷。設置地點的選擇策略有五種：第一種為選擇 k 個地點為所有節點經過次數較多的基地台；第二種為選擇前 k 個位處於整個環境中，所有任兩點之間的最短路徑次數最多的基地台；第三種則是從位於整個環境中任兩點之間的最短路徑次數最前十多的基地台中，挑選出前 k 個被環境中的車輛經過次數較多的基地台，若 k 的值大於 10，則再從位於最短路徑次數較多的基地台中；第四種將每個視該基地台為前十大走訪點的所有節點其 Degree 值(每個節點與其相連的鄰居數量)加總，最後選擇加總值較大的前 k 個基地台；第五種和第四種類似，唯一不同的是此方法選擇的不是 Degree 值而是將位於封包轉傳最短路徑的節點，其所屬路徑數量的加總。封包資料傳送的規則為封包可以在 Throwbox 和一般節點之間雙向傳輸，而在一般節點上，封包複本數量總和則不得大於N_{max}。在實驗中，此篇選擇了 Fresh[17]、Destination Frequency[18]、Centrality-Based[19]、Location-Based[20]四種路由做為實驗對象，結果中顯示 Throwbox 的設置確實可有效的改善封包，並發現在車輛節點密度高的網路環境中，第一種策略的封包傳輸率最高，封包傳輸延遲最低，且 Throwbox 的數量在 20 至 30 之間較適當。因此在本文中，使用的 Throwbox 設置策略為第一種。

本論文結合車輛行駛軌跡資料、時間、相遇車輛、路口編號…等，為每台車建立自己的車輛相遇序列圖，利用車輛相遇序列圖我們可以預測每台車子未來的可能經過的地點及可能會相遇的車子，希望能找到從發送端到接收端的最有效的傳送路徑。在 SATR 預測階段，改成直接蒐集每台車輛行駛途中接觸到的車輛和地點的資訊，直接建立出車輛相遇序列圖，和過去我們提出的 NTR 相比，大幅降低了計算的複雜度。並且引入社群的概念，加入了 Throwbox 增加封包傳輸效率，也運用在車輛相遇序列圖的建立和例外處理的部分。和目前車輛社群網路的路由相比，多了時間區段的概念，在不同時間區間，每台車行經相同地點的機會不同，因此車輛相遇序列圖也會有所不同，在轉傳中繼節點的選擇上也會有所改變。

	有無運用軌跡資訊	有無運用社群概念	轉傳車輛挑選	封包 Token 分配法	預測失敗補救	時間區間分類	複雜度
STDFS	有，計算預測接觸機率	無	根據 EDR/EDD	無(unicast)	無	無	高
NTR	有，比對軌	無	根據轉傳序	根據不同路	3 種	將一天	最高

	跡樹，算出轉傳序列		列得出的 EDR/EDD	徑 EDR/EDD 比例	CASES 較複雜	時間分成八個區間	
BEEINFO	無	有，統計一段時間內經過地點的車輛數，算出社群密度	根據車輛對地點的社群密度	無	無	無	最低
Hoten	無	有，運用熱點的受歡迎程度，算出節點和熱點的社交關聯度	根據社交關聯度算出的 Hoten 值	無	無	無	低
DFNMT	無	有，運用節點過去所停留的地點和時間，來預測未來該節點可能停留的地方	根據節點之間停留地點的熱門度與相似度	無	無	無	低
SATR	有，建立車輛相遇序列和封包轉傳路徑圖	有，運用地點熱門度設置 Throwbox	根據相遇次數與先後順序算出的支持值	根據不同路徑支持值比例	1 種 CASE 較簡單	一週七天，每天六個區間	中等

表 1. 路由協定比較

3. SATR 路由協定架構與設計

3.1 車輛相遇情況預測階段

圖 2 為一個車輛社群網路的環境圖。在車輛網路環境中，交通控制中心(TCC)[21] 維護區域內所有車輛的車輛相遇序列圖，車輛相遇序列圖包含車輛旅途中所經過的路口編號、經過時間，以及路程中有相遇的車子及相遇時的時間等資訊。將以一天的時間劃分成六個時間區段，分別為 00：00～06：00、06：00～07：30、07：30～09：30、09：30～17：00、17：00～19：00、19：00～24：00[22]。切割時段的好處不只在於分析車輛行駛模式，也讓車輛僅需在行駛前（家、辦公室等定點）從 TCC 下載更新車輛移動時間區段部分的相遇序列圖，可大幅減少從 TCC 的軌跡伺服器下載的資料大小。在每次車輛行駛結束到目的地後，上傳該次行駛的的資訊到 TCC，TCC 匯整先前蒐集到和這次行駛相同時段的行駛資訊來計算出和每輛車子的相遇及經過各個地點的情況，建立或更新出車輛相遇序列圖。

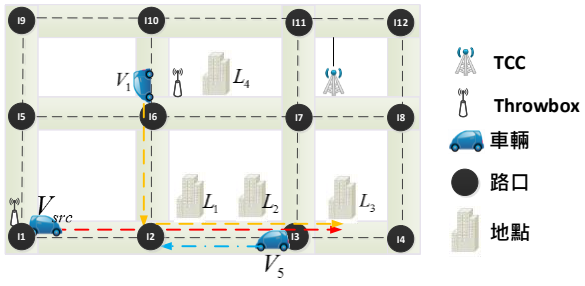


圖 2. 車輛社群網路環境示意圖

符號	意義
V_i	編號為 i 的車輛。
I_i	編號 i 的路口。
L_i	編號 i 的地點。
$S_V^{X^T}$	車輛V在第T天時對X的支持值。
V_{src}	封包來源車輛。

w_V^L	車輛V對地點L的私人熱門度。
w_i	路口i的公眾熱門度。
S_{V_i}	車輛V_i的車輛相遇序列。
H	封包轉傳次數上限。
$Sim_{S_{V_i},S_{V_j}}$	車輛V_i和V_j的軌跡相似度。
$\mathrm{G}^{\mathrm{D}}_{\mathrm{V_{src}}}$	封包來源車輛為V_{src}、封包目的地為D的車輛轉傳路徑圖。
TO_{V_i}	車輛V_i擁有的 Token 數量
$TO_{V_i}^X$	節點 X 能從車輛V_i得到的 Token 數。
β	TTL門檻值比例。

表 2. 符號表

SATR 車輛相遇情況預測階段，流程圖如圖 3 所示，首先將車輛每次行駛途中收集到的相遇車輛和經過地點的資訊做分析，根據社交關聯性計算出相遇車輛的權重值以及地點、路口的「熱門度」，接著以路口的熱門度挑選 Throwbox 擺放位置，確定好網路環境內參與封包傳輸的節點，建立出每台車輛的「車輛相遇序列圖」，並將其轉換成「車輛轉傳路徑圖」，再根據轉傳路徑圖分配傳送給每台相遇序列圖中車輛的 Token 值，以下分別說明每部份的詳細運作。

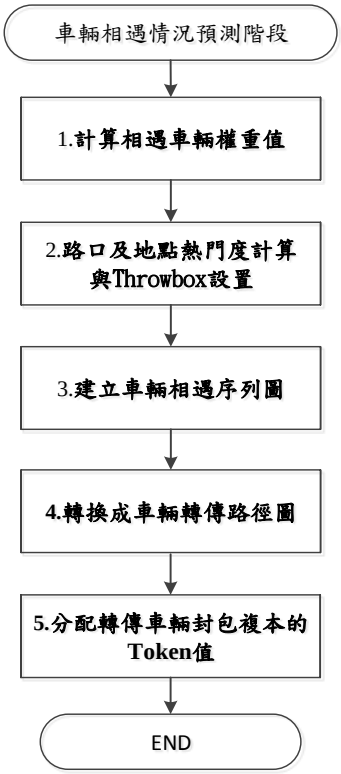


圖 3、車輛相遇情況預測階段流程圖

3.1.1 計算相遇權重值

首先，SATR 先統計(1)和各個不同車輛和地點遇到的次數，以及(2)相遇的先後順序。車輛相遇的先後順序對於封包轉傳路徑上的安排有著很大的影響，舉例來說，V_2會分別和V_1以及V_3行駛的途中相遇，若V_1所攜帶的封包正好需要傳給V_3，那麼在轉傳路徑的安排上，就會由V_1先傳給V_2，再由V_2傳給V_3，但在相遇的時間上，V_2會和V_3先相遇後，再和V_1相遇，此時封包傳送就會失敗，因此除了考慮車輛相遇次數（或機率）外，車輛之間相遇的先後順序必須列入考慮。每個先後順序皆有權重值，順序越前面，權重值越高。依照相遇的先後順序權重值，以及在該順序和各不同車輛及地點遇到的次數，算出每台相遇車輛和經過地點的支持（Support）值，其計算之公式如下：

$$S_V^{X^T} = \sum_{i=1}^{n} N_{V,i}^{X^T} \alpha_{V,i}^{X^T} \quad (1)$$

$$其中\ \alpha_{V,i}^{X^T} = 1 - \frac{1}{n}(i-1)$$

$$S_V^X = \frac{\sum_{T=1}^{M} S_V^{X^T}}{M} \quad (2)$$

其中$S_V^{X^T}$為車輛V在第T天時對X的支持值，S_V^X為長期統計下來（共M天）支持值的平均，X為車輛V在行駛過程中所遇到的車輛或路過的地點，M為此次計算的天數，n為該日中某一個特定時段內所遇見的車輛和行經的地點數量總合，$N_{v,i}^{X^T}$為X在某個軌跡i中和V相遇的次數，$\alpha_{v,i}^{X^T}$為車輛V所遇到的第i台車輛或地點的先後順序權重值。假設某台車V_{src}在近兩個月的每天上午 7:30~9:30 的車輛相遇以及路過地點的情況如圖 4，其中V_1、V_2、V_3、V_4為V_{src}在這段期間遇見的車輛；I_1為行經之路口；L_1、L_2、L_3為行經之建築物或地標。在 11/4（第一天，$T=1$）中，V_{src}在該時段所相遇車輛和經過的地點有7個，因此第一個順序車輛（或地點）的權重值α_1為$1-\frac{1}{7}(1-1)=1$，同理$\alpha_2 = 1-\frac{1}{7}(2-1) = 0.857$，$\alpha_3 = 0.714$、$\alpha_4 = 0.571$、$\alpha_5 = 0.429$、$\alpha_6 = 0.286$、$\alpha_7 = 0.143$，而$V_{src}$在第一天和$V_1$相遇的次數與順序為1，支持值$S_{V_{src}}^{V_1{}^1}$為$1 \times 1 = 1$，在其他七天的支持值分別為1、1、1、0.571、0.714、0.571、1，在這八天下來的平均支持值$S_{V_{src}}^{V_1} = \frac{1+1+1+1+0.571+0.714+0.571+1}{8} = 0.857$。同理$V_2$、$V_3$、$V_4$、$I_1$、$L_1$、$L_2$、$L_3$在這段時間的支持值分別為0.446、0.286、0.142、0.893、

0.750、0.429、0.143。

	1	2	3	4	5	6	7
11/4	V_1 7:10:11	I_1 7:10:14	L_1 7:11:35	V_2 7:12:19	L_2 7:16:54	V_3 7:17:23	L_3 7:25:11
11/11	V_1 7:11:12	I_1 7:11:20	L_1 7:12:32	V_2 7:12:59	L_2 7:17:20	V_3 7:17:46	L_3 7:25:00
11/18	V_1 7:10:05	I_1 7:10:30	L_1 7:11:56	V_2 7:12:34	L_2 7:16:13	V_3 7:17:58	L_3 7:25:53
11/25	V_1 7:09:40	I_1 7:10:01	L_1 7:11:22	V_2 7:12:46	L_2 7:17:37	V_3 7:17:40	L_3 7:25:18
12/2	I_1 7:10:31	L_1 7:11:14	V_2 7:12:35	V_1 7:12:38	L_2 7:16:36	V_4 7:17:43	L_3 7:25:32
12/9	V_2 7:13:00	L_1 7:13:41	V_1 7:13:58	V_4 7:14:09	L_2 7:17:24	V_3 7:17:49	L_3 7:25:47
12/16	I_1 7:11:21	V_2 7:11:37	L_1 7:11:45	V_1 7:12:17	L_2 7:16:34	V_3 7:17:13	L_3 7:25:48
12/23	V_1 7:10:32	I_1 7:10:56	L_1 7:11:36	V_3 7:12:39	L_2 7:16:41	V_4 7:17:30	L_3 7:25:40

圖 4、V_{src}在特定時間內的車輛相遇情況

3.1.2 地點熱門度計算與 Throwbox 設置

在固定地點路口和建築物部分，還需根據社交關聯性計算出其私人熱門度及公眾熱門度。私人熱門度的用途在於 Throwbox 中的封包轉傳時對於中繼車輛選擇的依據以及例外狀況處理，詳細用法於 3.2 封包轉傳階段會有完整說明。車輛V對地點L的私人熱門度w_V^L之計算方式如下：

$$w_V^L = \frac{N_V^L}{\sum_i N_i^L} \tag{3}$$

其中N_V^L為車輛V經過地點L的次數。假設車輛V_{src}、V_1、V_5在近七天內經過路口I_1的次數$N_{V_{src}}^{I_1}$、$N_{V_1}^{I_1}$、$N_{V_5}^{I_1}$分別為 7、10、8，則私人熱門度$w_{V_{src}}^{I_1} = \frac{7}{7+10+8} = 0.28$，$w_{V_1}^{I_1}$、$w_{V_5}^{I_1}$則分別為0.4、0.32。

公眾熱門度的用途在於挑選出設置 Throwbox 的地點。由於 Throwbox 的造價成本高，因此在設置地點的選擇上必須審慎挑選。我們透過地點公眾熱門度的計算，當作對於 Throwbox 設置地點挑選的依據。從車輛行駛過程中收集到每輛車經過各個路口的時間，來算出每個路口i的公眾熱門度w_i，依據w_i值的大小，挑選出最大的k個位置擺設 Throwbox。每個路口i的公眾熱門度w_i計算方式如下：

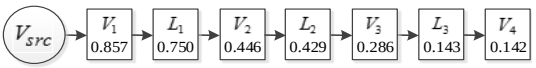
$$w_i = N_i \tag{4}$$

其中N_i代表在特定時間中，經過路口i的車輛總數。假設一網路環境中有十個路口，環境中的車輛經過每個路口的次數分別為23、11、19、16、24、7、6、31、14、18，則第一個路口的的公眾熱門度w_1值為23，其他路口依序為11、19、16、24、7、6、31、14、18。若欲建立 3 個 Throwbox，則依照w_i值大小選出前三高的路口1、5、8 擺設。最後在車輛相遇序列圖的建立中，必須將有經過擺設 Throwbox 之路口節點保留，其餘未設置 Throwbox 之路口節點則刪除，請參見 3.1.3。有 Throwbox 的設置，在封包轉傳路徑的安排上會有較多轉傳路線，封包傳送到達率以及延遲時間也會改善。

3.1.3 建立車輛相遇序列圖

依照上述中所算出的支持值，以及設有 Throwbox 的路口地點，建立出車輛相遇序列圖。此序列圖為一個有序之序列圖，排序依據為各車輛和地點之支持值，支持值值越大者順序在前，越小者在後。此相遇序列圖會隨著時間越長，蒐集的資料越多而有所變動，也會因時段不同，有不同的車輛轉傳路徑圖。以 3.1.1 中計算出的車輛支持值為例，假若I_1路口設有 Throwbox，則V_{src}之車輛相遇序列如圖 5(a)；若無則如圖 5(b)。

(a)I_1有 Throwbox 的情形



(b) I_1無 Throwbox 的情形

圖 5、V_{src}的相遇序列圖

3.1.4 轉換成車輛轉傳路徑圖

車輛相遇序列圖建立後，接下來就依據圖 6(a)及圖 6(b)的流程，使用圖中相遇車輛及路過地點與其支持值和相似度，來將其轉換成車輛轉傳路徑圖，並可以根據希望的轉傳次數上限值，來產生大小不同的轉傳路徑圖。以車輛V_{src}以及封包目的地D為例，首先必須設定封包轉傳次數上限H及車輛軌跡相似度上限SIM，封包轉傳次數上限H為車輛轉傳路徑圖$G_{V_{src}}^{D}$建立後，封包從V_{src}被傳送到D時所經過的轉傳車輛最大數量，用意是控制車輛轉傳路徑圖的大小，節省路徑圖建立的時間；車輛軌跡相似度Sim為任兩台車輛其車輛相遇序列中，路口或地點排列順序相同的數量。其計算方式如下：

$$Sim_{S_{V_i},S_{V_j}} = P_{ij} \tag{5}$$

其中P_{ij}代表S_{V_i}和S_{V_j}之中，地點及路口排序相同最長的部分。舉例來說，兩台車輛V_1和V_2，其車輛相遇序列圖$S_{V_1} = \{I_1, V_3, I_2, V_2, L_1, L_2, I_3, I_7\}$，$S_{V_2} =$

$\{I_5, I_1, I_2, V_1, I_6, I_7, L_2, I_3, L_3\}$，其中兩者依序經過的地點及路口分別為$\{I_1, I_2, L_1, L_2, I_3, I_7\}$及$\{I_5, I_1, I_2, I_6, I_7, L_2, I_3, L_3\}$，在兩者依序經過地點及路口的兩個集合中相同並連續的部分為$\{I_1, I_2\}$，因此V_1和V_2兩車的車輛軌跡相似度$Sim_{S_{V_1}, S_{V_2}} = 2$。設定$SIM$的目的是為了避免封包被傳送給行駛路徑過於類似的車輛，造成封包的散布狀況不佳，降低傳輸效率。

接著將V_{src}以及D加至即將建立的車輛轉傳路徑圖$G^D_{V_{src}}$，其中V_{src}為最頂層節點，D為最底層節點。根據V_{src}的車輛相遇序列圖$S_{V_{src}}$，若V_{src}會經過D（即$D \in S_{V_{src}}$），則將$G^D_{V_{src}}$中的V_{src}和D相連，並將V_{src}對D的支持值記錄在連接兩者的邊上。接下來再把到達D之前所經過的 Throwbox 加為V_{src}的下層節點，並記上V_{src}對各 Throwbox 的支持值。各 Throwbox 對D的部分，因 Throwbox 後續的封包傳送需仰賴實際環境中經過的車輛，無法預測接下來的轉傳路徑，因此以虛線連接。由於V_{src}會經過D的緣故，在$S_{V_{src}}$和其他車輛相遇序列圖的比對上，只需要比對在$S_{V_{src}}$中在節點D之前的車輛即可，因此設比對終點$END = D$；若V_{src}不會經過D（即$D \notin S_{V_{src}}$），則將比對終點$FINAL$設為$S_{V_{src}}$中最後一個節點，即需要比對整個$S_{V_{src}}$中的車輛。最後將記錄目前封包轉傳次數h的初始值設為0，再進入車輛序列比對演算法，繼續進行車輛轉傳路徑圖的建立。

```
1.  PacketDeliveryGraphConstruct(V_src, S_V_src, D, H, SIM)
2.  {
3.    // V_src : 封包轉傳路徑圖中的來源車輛
4.    // S_V_src : 來源車輛的車輛相遇序列圖
5.    // D : 封包目的地
6.    // H : 封包轉傳次數上限(自訂)
7.    // SIM : 車輛軌跡相似度上限(自訂)
8.
9.    G^D_V_src = null；
10.   將V_src加為G^D_V_src中的第一個節點； // G^D_V_src : 封包轉傳路徑圖
11.   將D加為G^D_V_src中的最後一個節點；
12.
13.   if (D ∈ S_V_src) // V_src是否會行經D
14.   {
15.     將V_src和D連接；
16.     把V_src對D的支持值記錄在兩者之間；
17.     FINAL = D；
18.     if (V_src在到達D之前有經過任何 throwbox T)
19.         將所有V_src在到達D之前經過的所有T加在G^D_V_src裡的V_src和D之間；
20.         把V_src和所有T連接並記下V_src對T的支持值於兩者之間；
21.         把所有T和D以虛線連接；
22.   }
23.   else
24.     將FINAL設為S_V_src中最後一個節點；
25.
26.   int h = 0；
27.
28.   VehicleEncounterSeqCompare(V_src, S_V_src, FINAL, SIM)
29.   //進入車輛相遇序列圖比對演算法
30. }
```

圖 6(a)、SATR 封包轉傳路徑圖建立演算法

```
1.  VehicleEncounterSeqCompare(V_src, S_V_src, FINAL, SIM)
2.  {
3.    for (S_V_src 中所有V_src在行經FINAL前遇見的車輛V)
4.    {
5.      將V加在G^D_V_src 中V_src下層並相連；
6.      把V_src對V的支持值記在兩者之間；
7.
8.      if (D ∈ S_V)   //判斷V是否會經過 D
9.      {
10.       if (Sim_S_V_src,S_V ≤ SIM)
11.       {
12.         將V和D相連並記下V對D的支持值於兩者之間；
13.         FINAL_V = D；
14.       }
15.       else
16.       {
17.         將V從G^D_V_src 中刪除；
18.       }
19.     }
20.     else
21.     {
22.       將FINAL_V設為S_V 中的最後一個節點；
23.     }
24.   }
25.
26.   h++；
27.
28.   if (h == H)
29.   {
30.     刪除在D的上層中未和D或其他任何節點相連的車輛節點；
31.   }
32.   else
33.   {
34.     for (G^D_V_src 中所有和V_src相連的V)
35.     {
36.       VehicleEncounterSeqCompare(V, S_V, FINAL, SIM)；
37.     }
38.   }
39. }
```

圖 6(b)、車輛相遇序列圖比對演算法

車輛序列比對演算法主要是檢查$S_{V_{src}}$中在$FINAL$之前的車輛V是否有機會成為轉傳車輛。對於每台車輛V，先加為$G^D_{V_{src}}$中V_{src}的下層節點連接並記上支持值，下一步判斷V是否會經過D（即$D \in S_V$），如是，則再判斷V_{src}和V的車輛軌跡相似度$Sim_{V_{src},V}$是否小於或等於設定的上限SIM（$Sim_{V_{src},V} \leq SIM$），若符合條件則將$G^D_{V_{src}}$中的V和D相連，記上V對D的支持值，並將V接下來檢查S_V時的比對終點$FINAL_V$設為D。如$Sim_{V_{src},V} > SIM$，此時將V和V_{src}對V的支持值從$G^D_{V_{src}}$中刪除；如V不會經過D（$D \notin S_V$），將V接下來檢查S_V時的比對終點END_V設為S_V中的最後一個節點。各項判斷後的動作結束後，再對其他尚未檢查的V執行上述車輛相遇序列比對演算法的判斷及之後的流程。

當$S_{V_{src}}$中在$FINAL$前的所有V皆檢查完後，

將目前封包轉傳次數的h值加1，如此時的h不等於封包轉傳次數上限H，表示$G^{D}_{V_{src}}$中V_{src}和D之間還可以加入適合的轉傳車輛，此時對剛加入$G^{D}_{V_{src}}$中的所有V，執行上段的車輛序列比對演算法；如h等於上限H，表示目前V_{src}和D之間的轉傳車輛數已經到達原先設定的值，則對$G^{D}_{V_{src}}$中的節點做最後檢查，找出D的上一層中，未與D相連的所有節點，將這些節點直到V_{src}下層節點中，相關的整個鏈結刪除，表示在達到上限H前，V_{src}沒有機會透過這些轉傳車輛將封包傳送到目的地D，完成封包轉傳路徑圖的建立。

以圖 7 中的車輛相遇序列圖為例，V_{src}為封包轉傳路徑圖的來源車輛，L_1為封包目的地，車輛軌跡相似度上限$SIM = 3$，封包轉傳次數上限在本例則分別示範$H = 1$和$H = 2$時的造成的封包轉傳路徑圖之差異。首先將V_{src}和L_1加入轉傳路徑圖，從圖 7 可得知V_{src}有經過L_1，因此將轉傳路徑圖中的V_{src}和L_1相連，並記上V_{src}對L_1的支持值，又V_{src}在行經L_1前經過了設有 Throwbox 的路口I_1，因此再將I_1加入轉傳路徑圖中V_{src}和L_1之間，分別以實線和虛線與V_{src}及L_1相連，且在V_{src}和I_1間記上V_{src}對I_1的支持值，下一步進入車輛相遇序列比對演算法，檢查V_{src}在經過L_1前所遇見的車輛其相遇序列圖。

在圖 7 中，V_{src}在經過L_1前遇見的車輛為V_1，因此將V_1先加入轉傳路徑圖中V_{src}和L_1之間與V_{src}相連，並記上V_{src}對V_1的支持值。V_1在遇見V_{src}後會經過L_1，而且兩車的車輛軌跡相似度$Sim_{V_{src},V_1} = 3 \leq SIM$，此時就可將轉傳路徑圖中的$V_1$和$L_1$相連，並記上$V_1$對$L_1$的支持值。$V_1$加入後，目前的封包轉傳次數為1，若$H = 1$，已達初始設定的轉傳次數上限，且$L_1$的上一層中沒有和$L_1$不相連的節點，封包轉傳路徑圖建立結束，最後內容如圖 8；若$H = 2$，尚未達到設定的轉傳次數，因此繼續轉傳路徑圖的建立，重複如前面所述V_1被加到轉傳路徑圖的步驟將V_5加入，此時封包轉傳次數為 2，已經達到轉傳次數上限，封包轉傳路徑圖建立結束，最後結果如圖 9 所示。圖 10 則分別代表$H = 2$時，I_1有無設立 Throwbox 所建立出的不同車輛轉傳路徑圖。

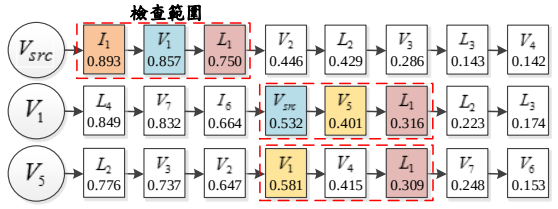


圖 7、車輛相遇序列圖範例

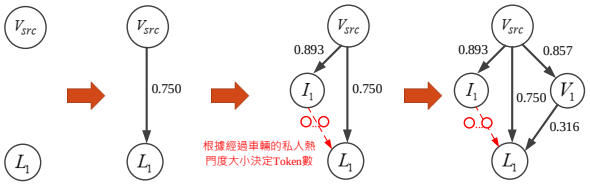


圖 8、車輛V_{src}對L_1轉傳路徑圖建立示意圖
(轉傳次數上限為 1)

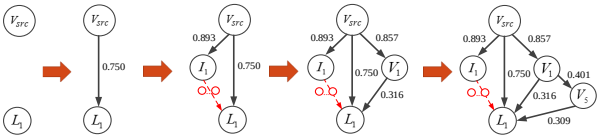
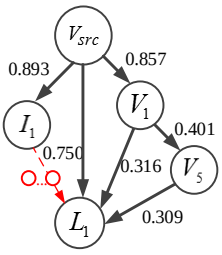
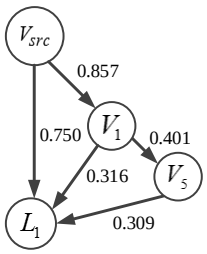


圖 9、車輛V_{src}對L_1轉傳路徑圖建立示意圖
(轉傳次數上限為 2)



(a)I_1有 Throwbox 的情況下



(b)I_1無 Throwbox 的情況下

圖 10、車輛V_{src}對L_1轉傳路徑圖

3.1.5 分配轉傳車輛封包複本的 Token 值

預測階段最後根據車輛轉傳路徑圖中的支持值，計算出 Multi-Copy 模式下，每台相遇車輛可分配到的 Token 比例，決定轉傳路徑圖中每個節點所能分配到的 Token 數量。若V_p本身和封包目的地直接相連的話，則會先將保留一個 Token，讓V_p在實際行駛中經過目的地時能直接將封包複本傳出；每台車輛、每個地點或 Throwbox X能從上層節點V_p得到某個封包的 Token 數量的計算公式如下：

$$TO^{X}_{V_p} = \begin{cases} 1，若X為封包目的地 \\ \left(TO_{V_p} - 1\right) \times \frac{S^{X}_{V_p}}{\sum_{i=1}^{n} S^{X_i}_{V_p}}，若來源車輛V_p有經過封包目的地 \\ TO_{V_p} \times \frac{S^{X}_{V_p}}{\sum_{i=1}^{n} S^{X_i}_{V_p}}，若來源車輛V_p沒有經過封包目的地 \end{cases} \quad (6)$$

其中X_i代表在車輛轉傳路徑圖中和V_p直接相連的車輛或 Throwbox，$S_{V_p}^{X_i}$為V_p對於直接相連的車輛或 Throwbox X_i的支持值，n為V_{src}在車輛轉傳路徑圖中相鄰的下層節點。以圖 10 為例，假如V_{src}所擁有的 Token 數量為 1001，則圖 10(a)中L_1(目的地)、V_1、I_1從V_{src}所直接得到之 Token 值分別為1、490、510。

$$TO_{V_{src}}^{L_1} = 1$$

$$TO_{V_{src}}^{V_1} = (1001-1) \times \frac{0.857}{0.893+0.857} = 490$$

$$TO_{V_{src}}^{I_1} = (1001-1) \times \frac{0.893}{0.893+0.857} = 510$$

若是沒有經過設有 Throwbox 路口的圖 10(b)，則L_1、V_1從V_{src}所得到之 Token 值分別為1、1000。

$$TO_{V_{src}}^{L_1} = 1$$

$$TO_{V_{src}}^{V_1} = (1001-1) \times \frac{0.857}{0.857} = 1000$$

接著V_1所分配到的 Token 會繼續往下層節點分配，至於設有 Throwbox 的路口I_1的 Token 傳送部分，因受到車輛實際行駛情況的影響，無法事先預估所能分配的 Token 數量，只能在車輛實際經過時才能進行判斷，詳見 3.2。

在 Token 分配上，SATR 和我們先前提出的 NTR 不同的地方在於 NTR 在封包轉傳路徑圖建立後，還需再將車輛相遇機率轉成 EDR 值，再根據 EDR 值分配 Token；而 SATR 提出的方法中，計算方式和 NTR 類似，但可直接透過轉傳路徑圖計算出每台車輛得到的 Token 數目，相較於 NTR 還需要再進行一次轉換才能算出 Token 值的計算方法節省計算時間。

3.2 封包轉傳階段

完成第一階段的相遇序列與轉傳路徑圖後，就可以利用該圖中所預測會遇到的車輛來進行封包傳送。轉傳階段演算法 SATRRelayStage 如圖 11 所示，並提出 SATR 預測失敗情況的修復方法。在轉傳開始前，必須先設定每個封包的存活時間(Time to Live, TTL)值以及預設的 TTL 比例值β。當封包開始進行轉傳後，便開始計時，若所剩時間低於設定的 TTL 乘上設定的比例值β，則會立即進行例外處理；若所剩時間為零，則封包會被自動捨棄，為的是提升封包傳送效率，避免過多的封包存在網路環境中，減少網路負荷（特別是 Multi-Copy 的封包轉傳模式）。

3.2.1 SATR 中的封包轉傳模式

當車輛V行駛在道路上時，先判斷目前封包的 TTL 值CurrentTTL是否低於一開始設定的 TTL 值乘上預設的比例值β，以及V目前擁有的 Token 數TO_V^D是否大於 0。若否，則執行例外處理狀況ExceptionHandling；若是，則繼續行駛，直到遇到任一個轉傳路徑圖中相鄰下層的節點為止。若遇到封包轉傳路徑圖中相鄰下層節點X，則有 Single-Copy 或 Multi-Copy 兩種處理模式：

1. Single-Copy：在 Single-Copy 的模式中，因為沒有封包複製的概念，因此只能將封包傳給X，然後進入TypeX演算法，判斷X為車輛、Throwbox 還是封包目的地後，分別進行不同的動作。
2. Multi-Copy：在 Multi-Copy 模式中，必須依照封包轉傳路徑圖，事先算出分配給X的 Token 值的大小，再將封包複本傳送出去給X，接著執行TypeX演算法，判斷X為車輛、Throwbox 還是封包目的地後，分別進行不同的動作。

圖 12 為判斷X種類演算法TypeX，如果該節點X為如果是(1)車輛，則繼續進行SATRRelayStage傳送封包 (2) Throwbox，則依照圖 13 的 Throwbox 的封包傳送機制演算法ThrowboxPacketDelivery傳送封包；如果是(3)封包的目的地，則封包傳送結束。

---

1. int CurrentTTL = TTL；
2. int StartTime = CurrentTime；
3. SATRRelayStage (G_V^D, CurrentTTL, TTL, β, V, TO_V^D)
4. {
5. while ((CurrentTTL > TTL × β) && ($TO_V^D > 0$))
6. {

```
7.          if (V_src遇到在G_V^D中相連的節點X)
8.          {
9.              switch (傳送模式)
10.             {
11.                case 'Single − Copy'：
12.                   將封包傳送給X；
13.                   TypeX(X)；      //判斷X為車輛、地點或 Throwbox
14.                case 'Multi − Copy'：
15.                   將封包依照G_{V_src}^D內分配的 Token 數量傳給X；
16.                   TO_V^D = TO_V^D − TO_X^D；
17.                   TypeX(X)；
18.             }
19.         }
20.
21.         CurrentTTL = TTL − (CurrentTime − StartTime)；
22.     }
23.
24.     if (CurrentTTL! = 0)
25.     {
26.         ExceptionHandling(G_V^D, W_V^D, V_p, TO_V^D)；
27.     }// if(23)
28. }
```

圖 11、封包轉傳階段演算法

```
1.  TypeX (X)
2.  {
3.     switch (Type)
4.     {
5.        case 'Vehicle'：
6.           SATRRelayStage (G_V^D, CurrentTTL, TTL, β, X, TO_V^D)；
7.        case 'Throwbox'：
8.           ThrowboxPacketDelivery()；
9.        case 'Destination'：
10.          End of Packet Transmission; // 封包傳送結束
11.    }// switch(3)
12. }
```

圖 12、判斷X種類演算法

3.2.2 Throwbox 封包傳送機制

如圖 13，當 Throwbox 從來源車輛V收到封包後，Throwbox 的 Token 值TO_{TB}為V之 Token 值TO_V，接著將V對於封包目的地D的私人熱門度W_V^D和經過 Throwbox 的車輛V_P對目的地的私人熱門度$W_{V_P}^D$做比較。若V_P對目的地的私人熱門度小於等於V，即$W_{V_P}^D \leq W_V^D$，則繼續等待和下一個經過的車輛比較對於目的地的私人熱門度大小；若V_P對目的地的私人熱門度大於V者，即$W_{V_P}^D > W_V^D$，在 Single-Copy 模式下，Throwbox 將封包傳送給V_P；在 Multi-Copy 模式時，則將V_P對目的地L的私人熱門度$W_{V_P}^D$除以V和V_P對目的地D的私人熱門度之和（$W_V^D + W_{V_P}^D$）所得的值，乘上 Throwbox 所持有的 Token 數所算出的 Token 值TO_{V_p}傳送給V_P，此時 Throwbox 所剩的 Token 值則為原來的數量TO_{TB}減去V_P得到的 Token 數TO_{V_p}，公

式如下：

$$TO_{V_p} = TO_{TB} \times \frac{w^L_{V_p}}{w^L_{V_c}+w^L_{V_p}} \quad (7)$$

$$TO_{TB} = TO_{TB} - TO_{V_p} \quad (8)$$

舉例來說，假設TO_{TB}擁有的 Token 數為200，V和V_p對目的地的私人熱門度為0.2和0.3，則V_p得到的 Token 數為$200 \times \frac{0.3}{0.2+0.3} = 120$，Throwbox 還有 80 個 Token。封包傳送給V_P後，如 Throwbox 中尚有 Token，則持續和行經的車輛比較對於目的地的私人熱門度大小，直到 Token 分送完畢。

而從 Throwbox 收到 Token 的V_P，將採取和 Throwbox 相同的封包傳送模式，和相遇的車輛比較對目的地的私人熱門度大小來傳送 Token，如V_P在途中行經目的地，將封包傳給目的地，丟棄剩下的 Token，結束封包傳送的流程。

```
1.  ThrowboxPacketDelivery (W_V^D, V_p, TO_TB)
2.  {
3.    if (W_{V_P}^D > W_V^D)
4.    {
5.      case 'Single − Copy'：
6.        將封包傳送給V_p；
7.      case 'Multi − Copy'：
8.        將封包依照公式(7)算出的 Token 數傳送給V_p；
9.        依照公式(8)更新 Throwbox 目前擁有的 Token 數TO_TB；
10.   }
11. }
```

圖 13、Throwbox 封包傳送機制演算法

3.2.3 SATR 封包轉傳的例外處理

若在行駛的過程中，當封包目前的TTL值低於初始設定的TTL乘上預設比例值β時，表示封包剩下的存活時間很少了，必須盡快將封包傳出給相遇的車輛。此時就開始進行例外處理 ExceptionHandling，如圖 14 所示，當來源車輛V尚未到達目的地前和任意車輛V_p相遇時，若V_p是V在封包轉傳路徑圖$\mathrm{G}^{\mathrm{D}}_V$中相鄰的下層節點，如果是在 Single-Copy 模式下，則將封包傳送給V_p；如在 Multi-Copy 模式下，則依照轉傳路徑圖分配 Token 值給V_p。若是遇到的車輛不是相鄰下層節點，則和相遇的車輛V_p比較私人熱門度大小，若V_p對目的地的私人熱門度大於V，在 Single-Copy 模式下，V將封包傳送給V_p；在 Multi-Copy 模式時，則使用公式(6)，將V_p對目的地L的私人熱門度除以V和V_p對目的地D的私人熱門度之和所得的值，乘上V目前所持有的 Token 數所算出的 Token 值傳送給V_p，回到圖 11 繼續進行封包傳送。在V_p收到封包後，繼續在路上行駛，和相遇的車輛比較對於目的地的私人熱門度大小。

```
1.  ExceptionHandling (G_V^D, W_V^D, V_p, TO_V)
2.  {
3.    if (V所遇見的車輛V_p為G_V^D中相連的節點)
4.    {
5.      case 'Single − Copy'：
6.        將封包傳送給V_p；
7.      case 'Multi − Copy'：
8.        將封包依照G_{V_src}^D內分配的 Token 數量傳給V_p；
9.    }
```

```
10.     else
11.     {
12.         if ($W_{V_P}^D > W_V^D$)
13.         {
14.             case 'Single − Copy'：
15.                 將封包傳送給$V_p$；
16.             case 'Multi − Copy'：
17.                 將封包依照公式(6)算出的 Token 數傳送給$V_p$；
18.                 依照公式(7)更新$V$目前擁有的 Token 數$TO_V$；
19.         }
20.     }
21. }
```

圖 14、例外處理演算法

圖 15 為 Multi-Copy 的封包傳送模式中，車輛V_{src}在實際行駛一段時間後時，TTL已低於一開始設定之TTL乘上預設的比例值β，此時進行例外處理。若V_{src}已經先到達I_1，分配510個 Token 給I_1，剩下490個 Token，但沒有如預期和V_1相遇，卻與不是V_{src}的相鄰下層節點V_2相遇，則依照相遇車輛V_2以及V_{src}對L_1的私人熱門度大小（0.3 , 0.2），依照公式(7)算出V_{src}欲傳送給相遇的車輛V_2的 Token 值$TO_{V_p} = TO_V \times \frac{W_V^D}{W_V^D + W_{V_p}^D} = 490 \times \frac{0.3}{0.2+0.3} = 294$。

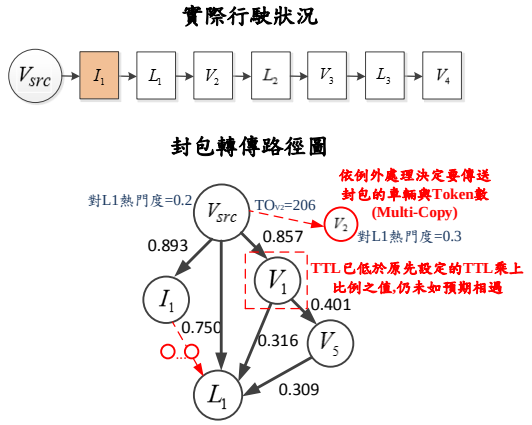


圖 15、車輛行駛情況和預測不同之例外處理

4. 實驗與效能

在效能評估的部分，我們使用 The ONE(The Opportunistic Network Environment simulator)[3]為主要工具，模擬參數如表 2。參考上海地圖資訊，如圖 16，模擬範圍大小為 12.9835 km x 12.4706 km。取 2007 年 2 月 26 日至 3 月 18 日每天上午 7:30 至 9:30 的車輛統計資料來建立軌跡預測，並使用 2007 年 3 月 19 日至 2007 年 3 月 25 日每天上午 7:30 至 9:30 的軌跡資料進行模擬實驗，結果為取 10 次平均。在我們的實驗模擬中，將會與 Hoten、BEEINFO、DFNMT 與 NTR 方法做比較。論文實驗中，會觀察封包存活時間、Token 值、收送對數和轉傳次數上限對各項方法的影響，以及各項方法的模擬時間。

用來比較的效能項目(Metric)如下：

1. Average Packet Delivery Ratio (APDR)：所有來源節點成功送達目的地的封包數量除以來源節點全部傳送的封包數目所得到的平均比值。
2. Average End-to-End Delay (AEED)：將所有從來源節點成功傳送到目的地節點的封包所需的延遲時間加總，除以所有成功傳送的封包數目得到的平均比值。
3. Average Overhead：主要考慮封包複本透過所有中繼車輛轉傳到目的車輛所消耗的網路頻寬。這個效能項目會受到給定的封包複本最大 Token 值、交換封包複本的排程機制、最大的 buffer 大小、網路負荷量與不同路由協定等的影響。在本實驗中測量值為當封包的 TTL 歸零時，同一個封包散佈到車輛網路節點上封包複本的總數在此定義為 Overhead。
4. 模擬時間：各個路由協定在模擬實驗中所花費的時間，因應各種方法的有不同階段的執行方式，本實驗將模擬時間分為軌跡資訊處理(Trajectory Info. Processing)、建立轉傳路徑 (Creating Forwarding Paths)和封包散布(Spraying Tokens)三個部分。

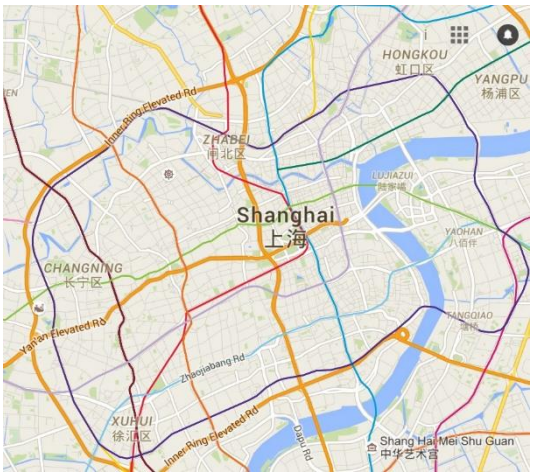


圖 16、實驗模擬地圖

表 3 模擬參數表

項目	設定值
無線傳輸協定	802.11p
模擬範圍	12.9835 km x 12.4706 km
公車數	2000
訊號範圍	300m
收集軌跡時間	2007 年 2 月 26 日至 3 月 18 日 每日 7:00 AM ~9:00AM
模擬時間	2007 年 3 月 19 日至 3 月 25 日 每日 7:00 AM ~9:00AM
轉傳次數上限	35
Throwbox 數量	25

4.1 封包存活時間對各種 metric 之影響

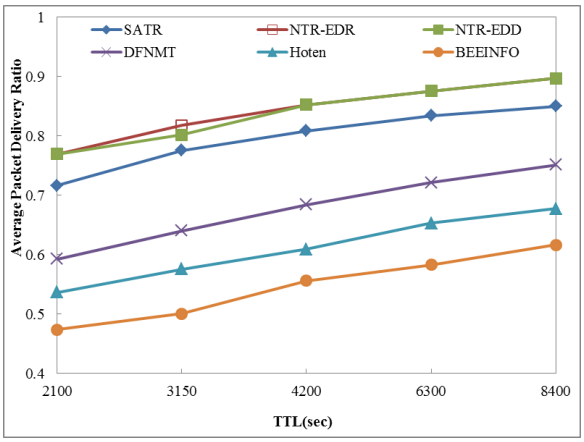


圖 17、封包存活時間對 APDR 之影響

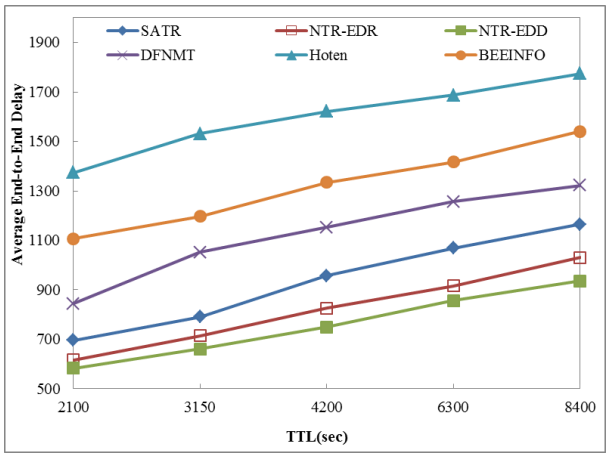


圖 18、封包存活時間對 AEED 之影響

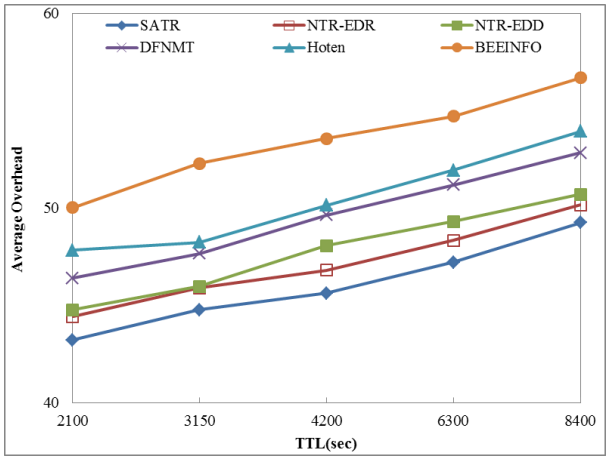


圖 19、封包存活時間對 Overhead 之影響

圖 17 顯示 TTL 對各項路由協定封包傳送率的影響，由於封包存活時間提升，使得封包能被攜帶的時間變長，讓節點有更多的時間可以將封包帶至目的地，傳送成功率因此提升。在上升幅度方面，各協定從 TTL 在 2100 提高到 8400 時皆上升約 15%，顯示封包存活時間對各種方法效能提升的影響沒有太多的差異。

在圖 18 中，各項路由協定的傳送延遲時間皆因為封包存活時間的提升而有明顯上升的現象，但也因為如此增加了封包到達目的地節點的機率。NTR-EDD 因使用軌跡資訊選擇延遲時間較短的節點協助傳送，所以封包傳輸的延遲時間也較 NTR-EDR 低。NTR 的 APDR 在 TTL 提升到 3150 後成長趨勢漸緩，AEED 則沒有明顯成長趨緩或增快的情況；SATR 的延遲時間在 3150 之後成長幅度較快，而在封包傳輸率上的成長則是趨於緩和；DFNMT 在 AEED 和 APDR 的成長幅度上則是沒有太大的起伏；Hoten、BEEINFO 的 AEED 和 APDR 成長趨勢則較無規律。

圖 19 顯示 TTL 的提升，增加了節點將封

包送到目的地的時間，除了使傳輸率及延遲時間提升外，也讓網路資源的使用率提升，所有路由協定的 Overhead 皆為上升狀態。SATR 在 TTL 來到 4200 之後的 Overhead 成長幅度增加，是由於時間拉長，使得節點在例外處理的機會增加，因此而提高了 SATR 的 Overhead。

4.2 Token 數目對各種 metric 之影響

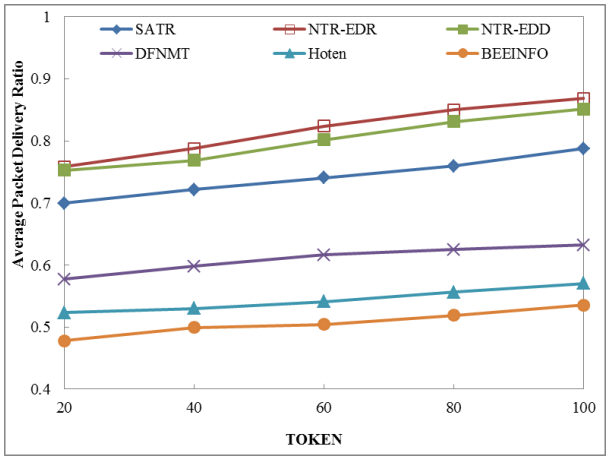


圖 20、Token 數目對 APDR 之影響

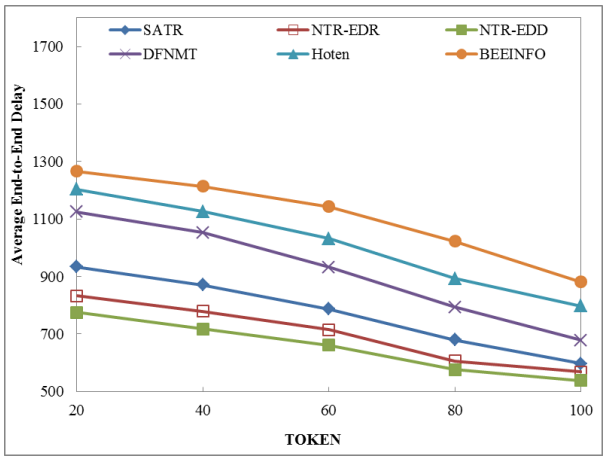


圖 21、Token 對 AEED 之影響

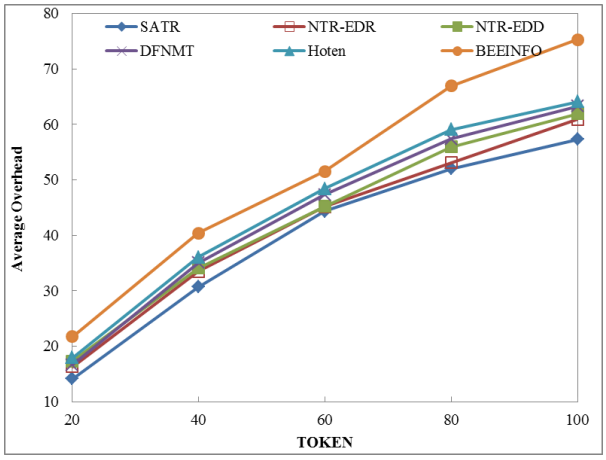


圖 22、Token 對 Overhead 之影響

圖 20 說明了 Token 對各項協定 APDR 的影響，當 Token 值越高時，對於有使用封包複製的方法來說，封包複本增加會使節點將封包傳送到目的地的成功率上升，本篇的所有路由協定皆受惠。其中使用軌跡資訊的 NTR 及 SATR 對於資訊的分析較仔細，上升幅度較明顯；DFNMT、Hoten、BEEINFO 在資料分析的過程上較簡略，即使複本數增加，在封包傳送的準確度上進步幅度仍然有限。

圖 21 說明了 Token 增加對於各種方法 AEED 的提升情況，當複本數增加，在環境中傳送的封包數量變多，傳送到目的地的機率也會提升。對於 DFNMT、Hoten、BEEINFO 來說，複本數增加可讓封包的散布情況提升，加快傳送速率降低延遲時間，因此變化幅度較明顯。

圖 22 可看出，本實驗中所有使用封包複本的協定其 Overhead 都隨著 Token 上限的增加而提高，BEEINFO 變化趨勢不穩定的原因在於複本數雖然增加，但方法本身參考的資訊較簡略，在轉傳節點的選擇上也比較不精確，造成實驗結果變化較大。

4.3 收送對數對各種 metric 之影響

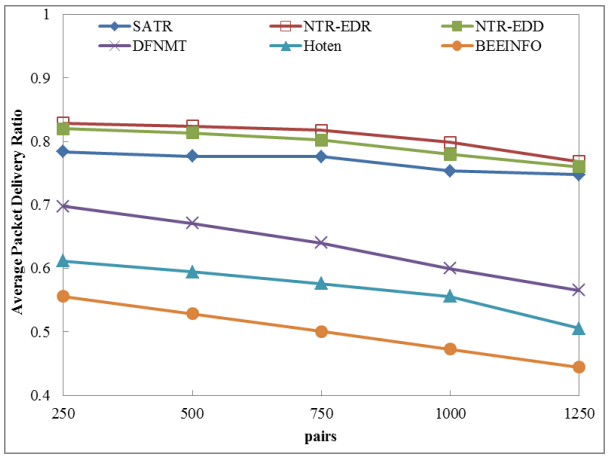


圖 23、收送對數對 APDR 之影響

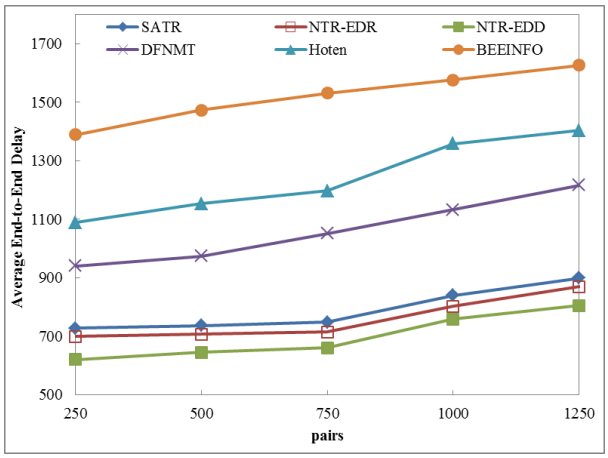


圖 24、收送對數對 AEED 之影響

從圖 23 以及圖 24 可以觀察出，在收送對數增加時也增加了封包碰撞的機率，造成了 APDR 降低以及 AEED 的上升。在圖 31 中，由於 NTR 收集了時間戳記以及車速等較多資訊建構出軌跡樹，因此在傳輸成功率的表現較其他路由協定佳，NTR-EDR 的封包傳送率比 NTR-EDD 的 APDR 高是因為 NTR-EDR 所設計分配 token 是依照車輛有較高機率傳送至目的車輛的路徑轉傳封包；而記錄車輛相遇順序並針對各個時段計算出不同車輛轉傳路徑的 SATR 也勝過其他車輛社群路由協定，僅次於 NTR；DFNMT 和 Hoten 皆利用節點在不同地點間移動的資訊計算出節點對於各地點的社交關聯度，再透過節點對目的地的社交關聯度來選擇轉傳節點，但 DFNMT 還多比較了節點間移動情況的相似度，剔除移動情況過於相似的節點，增加封包的傳送效率，因此在 APDR 的表現上勝過 Hoten；BEEINFO 僅透過節點在過去經過目的地次數的多寡，來決定封包轉傳節點，使得封包傳送率在各路由協定中表現較差。

在圖 24 中，NTR 的兩個路由協定因為在資料收集與分析上較仔細，在封包轉傳節點的選擇上比較精確，因此依舊在傳送延遲時間上的表現優於其他路由協定，而 NTR-EDD 在 AEED 方面勝過 NTR-EDR 是因為 NTR-EDD 挑選的是較低的封包延遲時間，故 NTR-EDD 的 AEED 比 NTR-EDR 低；SATR 除了使用車輛相遇序列外，也比較了節點之間的軌跡相似度，避免選擇到軌跡相似的點做為轉傳節點，降低傳送延遲時間；DFNMT 也因為比較了節點間的移動情況相似度，提升了在 AEED 方面的效能而勝過 Hoten；而 BEEINFO 僅根據節點經過各社群的次數選擇轉傳節點，未進一步分析節點在各社群間移動的先後次序，因此在傳送延遲時間上亦高於其他路由協定。

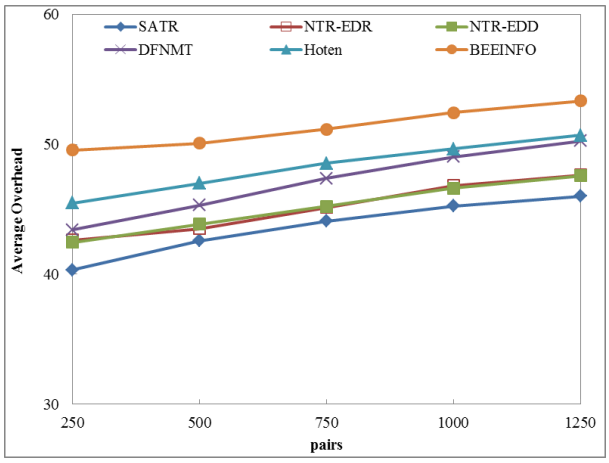


圖 25、收送對數對 Overhead 之影響

從圖 25 可以發現，有使用 Token 概念的 NTR 和 SATR 在 Overhead 上都低於未使用軌跡資訊其他的路由協定，顯示 Token 的分配有助於降低對網路資源消耗。SATR 在封包轉傳上限的控制下，轉傳路徑的數量較 NTR 少，因此參與轉傳的節點也較少，使得 Overhead 較 NTR 低；DFNMT 及 Hoten 分別透過半馬可夫鍊與 Entropy 來計算社交關聯度，在選擇轉傳節點上較 BEEINFO 謹慎，Overhead 也較低；DFNMT 又因為多了軌跡相似度的篩選，讓 Overhead 方面略勝過 Hoten。

4.4 轉傳次數上限對各種 metric 之影響

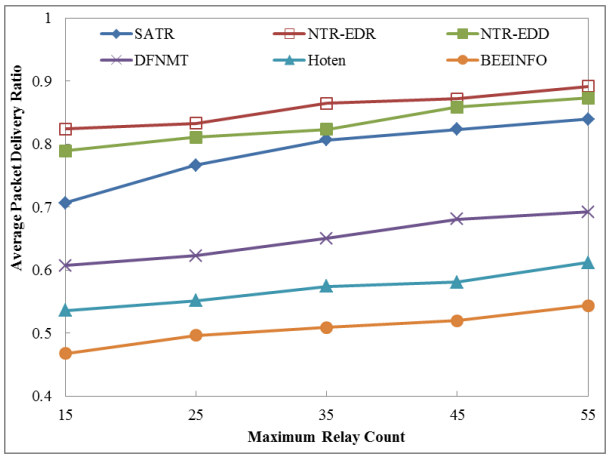


圖 26、轉傳次數上限對 APDR 之影響

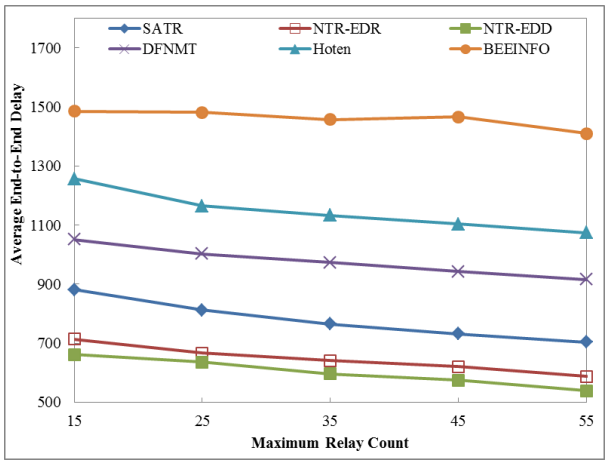


圖 27、轉傳次數上限對 AEED 之影響

圖 26 表示了轉傳次數上限對實驗中各項方法其 APDR 的影響，當轉傳次數上限提升後，從來源到目的地之間參與封包傳送的節點增加，於是封包傳送成功率也隨之提高。其中 SATR 的上升幅度較其他路由協定明顯的原因在於，當轉傳次數上限越高，SATR 封包轉傳路徑圖中的傳送路徑增加，在封包傳送上路徑選擇變多，封包更能順利散布到整個網路，攜帶該封包的節點變多，封包傳送成功率也就因而提高。

圖 27 中可觀察出，轉傳次數上限的影響依舊對於 SATR 較為顯著，尤其在轉傳次數上限從 15 增加到 35 時的幅度又較明顯，在 35 之後的下降幅度趨緩，顯示當上限增加時，雖然轉傳路徑圖中的路徑增加，但增加的數量並不如轉傳路徑圖一開始建立時的路徑增加量，才會造成上限提高但效能提升有限的情況。

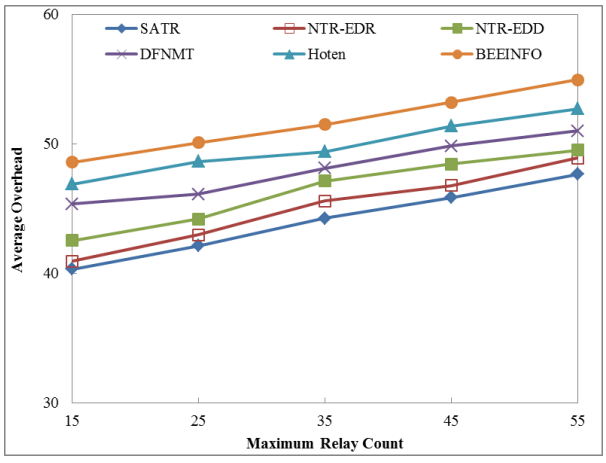


圖 28、轉傳次數上限對 Overhead 之影響

圖 28 可看出，轉傳次數上限增加，使得參與封包傳送的節點變多，除了提高封包傳送成功率之外，也會使得網路資源的利用增加，提高 Overhead。SATR 的 Overhead 上升趨勢在隨著轉傳次數上限的提高之中並沒有減緩或加快的趨勢，然而在 APDR 以及 AEED 之中皆可觀察到，當轉傳次數來到 35 後的進步幅度有限，因此本篇實驗中也以 35 做為轉傳次數上限的預設值。

4.5 各種方法之模擬時間

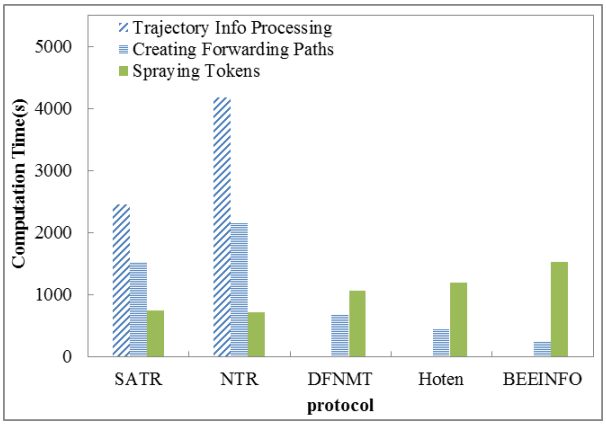


圖 29、各種方法之模擬時間

圖 29 為本文實驗中各項路由協定在各個階段所花費的模擬時間，和一般車輛社群網路 DFNMT、Hoten、BEEINFO 相比，雖然在軌跡資訊處理和封包轉傳路徑建立的資料前處理時間高出許多，但在前面所述的各項效能中 SATR 則是勝過 DFNMT、Hoten、BEEINFO 許多，因此可以視為一項值得的取設。而在和使用車輛軌跡資訊的 NTR 相比，雖然在上述的各項效能中除了 Overhead 以外皆略遜於 NTR，但在資料前處理的部分，軌跡資訊的處理時間上大約只有 NTR 的 60%，建立轉傳路徑所花費的時間也只有 NTR 的約 70%。因此本篇所提出的方法順利的降低了資料前處理的時間，也在封包成功到達率和延遲時間上和 NTR 保有一定程度的距離。

5. 結論

本篇論文提出了適用於車輛社群網路的封包轉傳協定 SATR，在「車輛軌跡預測階段」車輛相遇序列圖與車輛轉傳路徑圖，事先計算出在不同時間區段內，車輛行駛經過的地點、中繼車輛與分配給中繼車輛的封包複本 Token 數目，加上利用歷史車輛行駛記錄，計算出每台車輛和其他車輛與行經地點間的社交關聯

性，改善傳統車輛軌跡路由協定的高計算複雜度與車輛社群網路沒有考慮不同時間社交關聯性的缺點，還增加資料儲存裝置 Throwbox 的設置，提升封包傳輸效率。最後在網路模擬實驗結果中，比傳統車輛社群網路路由協定有更好的效能,和車輛軌跡路由協定 NTR 相比，SATR 因為利用車輛相遇序列圖，相較於 NTR 記錄了時間戳記和車速的軌跡樹簡潔，在建立和比對上相對快速；在封包轉傳路徑圖方面設置了轉傳次數上限和軌跡相似度上限，限制了路徑圖的大小，因此在路徑圖的建立也相較於 NTR 快速，使得 SATR 能有較快的處理時間以及接近 NTR 的效能。

參考文獻

[1] Anna Maria Vegni and Valeria Loscrí, ”A Survey on Vehicular Social Networks,” ***IEEE Communication Surveys & Tutorials***, Vol. 17, No. 4, pp. 2397–2419, Fourth Quarter, 2015.

[2] N. Santhiya Kumari and P. S. Nithya Darisini, “A Survey of Routing Protocols for VANET in UrbanScenarios,” ***IEEE Pattern Recognition, Informatics and Mobile Engineering (PRIME) International Conference***, 2013.

[3] B. Li, Y. Wu, and Y. Zhu, “Trajectory Improves Data Delivery in Vehicular Networks,” ***IEEE INFOCOM***, pp. 2183-2191, 2011.

[4] F. Xia et al., “BEEINFO: Interest-based Forwarding Using Artificial Bee Colony for Socially-Aware Networking,” ***IEEE Transaction Vehicle Technology***, Vol. 64, No. 3, pp. 1–11, Feb. 2014.

[5] R. Fei, K. Yang, and X. Cheng, “A Cooperative Social and Vehicular Network and its Dynamic Bandwidth Allocation Algorithms,” ***IEEE INFOCOM WKSHPS***, pp. 63–67, Apr. 2011.

[6] X. Gu, L. Tang, and J. Han, “A Social-aware Routing Protocol based on Fuzzy Logic in Vehicular Ad Hoc Networks,” Proc. ***International High Mobility Wireless Communications Workshop***, pp. 12–16, Nov. 2014.

[7] F. D. Cunha et al., “Socially Inspired Data Dissemination for Vehicular Ad Hoc Networks,” in Proc. 17th ACM Int. Conf. Model., ***Analysis and Simulation of Wireless Mobile System***, pp. 81–85, 2014.

[8] Quoc Nguyen Viet, Pham Van Phuoc, Trinh Quoc Son, and Lung Vu Duc, “Social Routing: A Novel Routing Protocol for Delay Tolerant Network based on Dynamic Connectivity,” ***IEEE Computer and Information Science (ICIS)***, pp. 35–40, 2015.

[9] Qi Wang, Qingshan Wang, “Data Forwarding Based on Node Moving Trajectory in Mobile Social Networks,” ***Wireless Personal Communications***, Vol. 87, pp.1285–1297, 2016.

[10] W. Gao, J. Chen, J. Fan, Y. Du, W. Gao, J. Chen, Y. Sun, “Geography-aware Active Data Dissemination in Mobile Social Networks,” ***IEEE MASS***, 2010.

[11] P. Yuan and H. Ma, “Opportunistic Forwarding with Hotspot Entropy,” ***Ad Hoc Networks***, Vol. 33, pp. 284–305, 2015.

[12] Ing-Chau Chang, Ming-Han Hung and Ching-Ju Chang, “Novel Trajectory Routing for Efficient Controlled Replication in Vehicular Delay Tolerant Network,” ***the International Scientific Conference on Engineering and Applied Sciences (ISCEAS)***, Okinawa, Japan, July 2015.

[13] Qi Wang, Qingshan Wang, “Data Forwarding Based on Node Moving Trajectory in Mobile Social Networks,” ***Wireless Personal Communications***, Vol. 87, pp.1285–1297, 2016.

[14] Q. Yuan, I. Cardei and J. Wu, “An Efficient Prediction-based Routing in Disruption-tolerant Networks,” ***IEEE Transactions on Parallel and Distributed System 23***, pp.19–31, 2012.

[15] J. K. Lee and J. C. Hou, ”Modeling Steady-state and Transient Behaviors of User Mobility: Formulation, Analysis, and Application,” ***Proceedings of the ACM international symposium on mobile ad hoc networking and computing***, pp. 85–96, 2006.

[16] T. Spyropoulos, K. Psounis, and C.S. Raghavendra, “Spray and Focus: Efficient Mobility-Assisted Routing for Heterogeneous and Correlated Mobility,” ***Fifth Annual IEEE International Conference on Pervasive Computing and Communications Workshops***, 2007. PerCom Workshops '07.

[17] H. Dubois-Ferriere and M. Grossglauser, M. Vetterli, “Age Matters: Efficient Route

Discovery in Mobile Ad Hoc Networks Using Encounter Ages," ***ACM MobiHoc***, 2003.

[18] V. Erranmilli, M. Crovella, A. Chaintreau, C. Diot, "Delegation forwarding," ***ACM MobiHoc***, 2008.

[19] P. Hui, J. Crowcroft, E. Yonek, "Bubble rap: Social-based Forwarding in Delay Tolerant Networks," ***ACM MobiHoc***, 2008.

[20] W. Gao, J. Chen, J. Fan, Y. Du, W. Gao, J. Chen, Y. Sun, "Geography-aware Active Data Dissemination in Mobile Social Networks," ***IEEE MASS***, 2010.

[21] ***Philadelphia Department of Transportation***, "Traffic Control Center," http://philadelphia.pahighways.com/philadelphiatcc.html

[22] Wei-Hsun Lee ,Kuo-Ping Hwang ,Wen-Bin Wu, "An intersection-to-intersection travel time estimation and route suggestion approach using vehicular ad-hoc network, "***Ad Hoc Networks 43***, 2016.