

## 博碩士論文電子檔案上網授權書

(提供授權人裝訂於紙本論文書名頁之次頁用)

本授權書所授權之論文為授權人在 東海大學 資訊工程與科學系 \_\_\_\_\_ 組 96 學年度第二學期取得 碩士學位之論文。

論文題目：TCP流量控制演算法在行動隨意網路下之效能分析  
指導教授：朱延平

茲同意將授權人擁有著作權之上列論文全文(含摘要)，非專屬、無償授權國家圖書館及本人畢業學校圖書館，不限地域、時間與次數，以微縮、光碟或其他各種數位化方式將上列論文重製，並得將數位化之上列論文及論文電子檔以上載網路方式，提供讀者基於個人非營利性質之線上檢索、閱覽、下載或列印。

- 讀者基非營利性質之線上檢索、閱覽、下載或列印上列論文，應依著作權法相關規定辦理。

授權人：王祥明

簽名：王祥明

中華民國 97 年 07 月 01 日



東海大學碩士學位論文考試審定書

東海大學資訊工程與科學系 研究所

研究生 王 祥 明 所提之論文

TCP 流量控制演算法在行動隨意網路下之效能分析

經本委員會審查，符合碩士學位論文標準。

學位考試委員會  
召 集 人 王 曉 璿 簽章

委 員 吳 一 泓

林 冠 成

朱 廷 亨

指 導 教 授 朱 廷 亨 簽章

中華民國 97 年 6 月 24 日

# 中文摘要

行動隨意網路 (Ad Hoc Network) 是一種無須網路存取點 (Access Point) 而建立起的對等網路結構，只要安裝了無線網路介面卡的行動裝置彼此之間即可無線互連；其原理是網路中的任一台行動裝置與附近的行動裝置建立點對點的連線，相當於虛擬 AP，而其它行動裝置就可以透過這個已建立的點對點連線進行網路連線與共享。

而傳輸控制協定 (Transmission Control Protocol) 是 TCP/IP 中傳輸層的協定，它透過序列確認以及封包重送機制，提供了可靠性和連接導向的服務。而有資料送到網路上時，為了能夠公平分享網路頻寬，TCP 以壅塞控制機制來維持共享頻寬的公平性。也因此，對於先天上頻寬就有限制的無線網路來說，TCP 的效能必定是一個重要的關鍵。

本篇論文將現行常見的五種 TCP 版本，交叉搭配兩種先行式行動隨意網路路由協定以及四種不同的最大移動速率，透過 NS-2 網路模擬器來模擬分析各種 TCP 版本的 Throughput、封包丟棄數等，發現除了 Vegas 之外，其實 TCP 版本的影響並不如路由演算法選擇的影響來的大。

**關鍵詞：** 行動隨意網路，傳輸控制協定，行動隨意網路路由協定

# 英文摘要

An ad hoc network is a collection of wireless mobile nodes that self-configures to form a temporary network without the use of existing network infrastructure or centralized control. The principle of such network architecture is that when two or more mobile nodes are within the communication range of each other, they may establish a point-to-point connection and communicate with each other either directly or indirectly by having other intermediate nodes to relay their packets.

Transmission Control Protocol is a TCP/IP transport layer protocol. It provided reliability and connection-oriented services through the confirmation of sequence number and packet retransmission mechanisms. In order to fairly share the network bandwidth when there are data to be transmitted in the network, TCP uses congestion control mechanism to ensure the fairness between shared bandwidth. Therefore, TCP performance will be an important key for wireless network due to the bandwidth limitations of the wireless network.

In this thesis, we use NS-2 to evaluate the throughput, number of drop packets of five common TCP versions which are mixed with two proactive ad hoc network routing protocols and four different max moving speeds. Our simulation results have showed that the routing protocol has greater impact on wireless network rather than different TCP versions except Vegas.

**Keywords :** *Ad Hoc Network, Transmission Control Protocol, Ad Hoc Routing Protocol*

# 致謝

本論文能夠順利地完成，不是單單只倚靠個人的辛苦以及努力，而是要歸功於許多人的指導、協助。在此，希望以我最誠摯的心意，向你們說聲「謝謝」。

首先感謝我的指導教授朱延平教授，由於他不吝於分享其豐富的經驗與知識，讓我學會以更寬廣的視野去看事情、做研究。其次感謝黃一泓教授與林冠成教授對本論文的指導與建議，使本論文得以更為完整。第三，要感謝口試委員王曉璿教授對於本論文之指正與建議。

我亦非常感謝研究室的許多同學與學長：昭佑、振邦和琮堯學長的鼓勵與討論；群智、益仁、志銘在口試時所給予的協助，謝謝各位。

最後，最感謝的就是我的家人，爸爸、媽媽、妹妹，在我的求學生涯中，一直給予我支持，讓我可以無後顧之憂，專心於論文與完成學位，希望大家能一直身體健康快樂。

# 目錄

中文摘要.....	I
英文摘要.....	II
致謝.....	III
目錄.....	IV
圖目錄.....	VI
表目錄.....	VII
第一章 前言.....	1
1.1 簡介.....	1
1.2 動機.....	2
1.3 目的.....	3
1.4 論文結構.....	3
第二章 相關研究.....	4
2.1 無線行動隨意網路 (Ad Hoc Network).....	4
2.1.1 無線行動隨意網路發展背景.....	4
2.1.2 無線行動隨意網路的特徵.....	5
2.2 無線行動隨意網路路由協定.....	7
2.2.1 Table Driven 路由協定.....	11
2.2.2 On-Demand 路由協定.....	13
2.3 TCP 傳輸控制協定.....	16
2.3.1 壅塞控制.....	16
2.3.2 Tahoe.....	18
2.3.3 Reno.....	18
2.3.4 NewReno.....	19
2.3.5 Sack.....	20
2.3.6 Vegas.....	20
2.4 無線行動隨意網路中影響 TCP 效能的因素.....	22
第三章 系統實驗設計與模擬環境.....	24
3.1 系統實驗設計.....	24
3.2 模擬環境.....	25
第四章 實驗結果與討論.....	27
4.1 單一版本環境模擬結果分析.....	27
4.1.1 Tahoe.....	27
4.1.2 Reno.....	29
4.1.3 NewReno.....	31
4.1.4 Sack.....	33
4.1.5 Vegas.....	34
4.2 不同版本混合環境模擬結果分析.....	38

4.2.1 Reno vs. Tahoe.....	3 8
4.2.2 Reno vs. NewReno .....	4 0
4.2.3 Reno vs. Sack.....	4 1
4.2.4 Reno vs. Vegas .....	4 3
第五章 結論與建議.....	4 6
參考文獻.....	4 7

# 圖目錄

圖 2-1:平面式與階層式路由網路結構.....	8
圖 2-2: AODV 路徑搜尋運作圖.....	14
圖 2-3: DSR路徑搜尋運作圖.....	15
圖 4-1: Tahoe 在 DSDV 與 OLSR 下 Throughput 的比較.....	27
圖 4-2: Tahoe 在 DSDV 與 OLSR 下丟棄封包數量的比較.....	28
圖 4-3: Reno 在 DSDV 與 OLSR 下 Throughput 的比較.....	29
圖 4-4: Reno 在 DSDV 與 OLSR 下丟棄封包數量的比較.....	30
圖 4-5: NewReno 在 DSDV 與 OLSR 下 Throughput 的比較.....	31
圖 4-6: NewReno 在 DSDV 與 OLSR 下丟棄封包數量的比較.....	32
圖 4-7: Sack 在 DSDV 與 OLSR 下 Throughput 的比較.....	33
圖 4-8: Sack 在 DSDV 與 OLSR 下丟棄封包數量的比較.....	33
圖 4-9: Vegas 在 DSDV 與 OLSR 下 Throughput 的比較.....	34
圖 4-10 Vegas在 DSDV 與 OLSR 下丟棄封包數量的比較.....	35
圖 4-11: 各 TCP 版本在 DSDV 下 Throughput 的比較.....	36
圖 4-12: 各 TCP 版本在 DSDV 下丟棄封包數量的比較.....	36
圖 4-13: 各 TCP 版本在 OLSR 下 Throughput 的比較.....	37
圖 4-14: 各 TCP 版本在 OLSR 下丟棄封包數量的比較.....	37
圖 4-15: Reno vs. Tahoe 在 DSDV 下的 Throughput.....	38
圖 4-16: Reno vs. Tahoe 在 OLSR 下的 Throughput.....	39
圖 4-17: Reno vs. NewReno 在 DSDV 下的 Throughput.....	40
圖 4-18: Reno vs. NewReno 在 OLSR 下的 Throughput.....	40
圖 4-19: Reno vs. Sack 在 DSDV 下的 Throughput.....	41
圖 4-20: Reno vs. Sack 在 OLSR 下的 Throughput.....	42
圖 4-21: Reno vs. Vegas 在 DSDV 下的 Throughput.....	43
圖 4-22: Reno vs. Vegas 在 OLSR 下的 Throughput.....	43

# 表目錄

表 2-1: 三種類型的路由協定效能比較.....	10
表 3-1: 模擬實驗參數設定表.....	26
表 4-1: 各種 TCP 版本在不同路由協定下的效能比較表.....	45

# 第一章 前言

## 1.1 簡介

隨著使用者的成長、科技的普及、以及無線行動裝置的價格降低，手機、PDA以及筆記型電腦越來越多普遍，使得無線網路的使用率也相對的提昇許多。對於這些無線行動裝置，基本上傳輸的方式可分為兩大類，一種就是透過基地台或網路存取點 (Access Point)，由中央控制的傳輸方式，但必須事先將基地台等網路基礎建設建立好，而且一旦超出基地台可傳輸的範圍，就會無法繼續線；另外一種則是無線行動隨意網路 (Ad Hoc Network)。

Ad Hoc Network被提出當作解決無線網路建置時間以及網路基礎建設無法涵蓋的區域。Ad Hoc Network允許行動裝置之間透過無線頻道互相交換資訊，無須經過基地台或網路存取點。每個在Ad Hoc Network底下的行動裝置可視為一個終端系統、伺服器、路由器或是閘道等等，某些時候甚至會同時扮演這些角色。而TCP的效能預期會對Ad Hoc Network有相當程度的影響，不僅僅是因為無線網路的環境因素，也包含一些Ad Hoc Network特定的議題如機動性、路由以及電力限制等等。因為在Ad Hoc Network底下，只要想交換資訊的裝置在彼此間的傳輸範圍內，都可透過直接的連線來作交換資訊的動作，若是超出彼此傳輸的範圍，則可透過其他鄰近裝置來幫忙傳遞想交換的

資訊，因此，為了一直維持正確的路徑資訊，路由協定必須：(1)讓路由表格持續更新以保持正確的路徑，並且不能讓路由表格太大，(2)為想傳輸資料的傳送端以及接收端選擇最好的路徑。這對一個網路拓樸不斷在改變的環境來說，是項非常大的挑戰，尤其當裝置移動的速度加快，兩兩裝置間的鏈結是很容易改變的。

## 1.2 動機

既然路由對於Ad Hoc Network來說有如此重大的影響，不少研究針對不同的路由協定，探討各種TCP版本在這些路由協定中的效能。由[4] [15]的模擬結果可以看出，先行式的路由協定，不管在哪種TCP版本，其所呈現的效能，皆比反應性的路由協定來的好。因此，本篇論文將深入探討對於先行式的路由協定來說，各種TCP版本在不同的最大移動速率下的效能差異，以瞭解這些因素對於傳輸效能的影響，以便未來在Ad Hoc Network上設計傳輸軟體時，可以將這些因素依照影響的程度做先後的考量。再針對不同TCP版本在混合的環境下，觀察各種TCP版本是否還能維持傳輸的公平性。

### **1.3 目的**

由於目前有線網路在傳輸方面，還是以使用TCP為主，在網路如此普遍的情況下，Ad Hoc Network勢必也須與有線網路進行連線，因此本篇論文將探討路由協定 節點移動速率與TCP版本三種因素在Ad Hoc Network中影響TCP效能的重要程度。

### **1.4 論文結構**

本論文章節編排如下：首先先介紹與本篇論文相關的主題研究，介紹後面實驗使用到的兩種路由協定以及TCP版本。接下來是系統的模擬與結果分析，會說明所設置的模擬環境以及變數。最後是結論。

## 第二章 相關研究

在本章節中，我們將討論 TCP 壅塞控制的機制、無線行動隨意網路的發展過程，以及其上常見的先行式與反應式的路由協定，最後則說明現今學者所列出影響無線行動隨意網路的效能因素有哪些。

### 2.1 無線行動隨意網路 (Ad Hoc Network)

一般我們常用的無線網路都是由中央控管的，也就是要基於預先建設的網路設施才能運作。例如，手機通訊系統要有基地台的支援；無線區域網路一般也需要在有 AP 存取點和有線骨幹的模式下才能運作。但對於有些特殊場合來說，需要基地台的無線網路並不能勝任。如戰場上部隊快速展開和推進、地震或水災後的營救等。這些場合的通訊不能依賴於任何預設的網路設施，而需要一種能夠臨時且快速自動組成的行動網路。而 Ad Hoc Network 就是在這種需求下產生出來。

#### 2.1.1 無線行動隨意網路發展背景

Ad Hoc Network 是由封包無線網路 (Packet Radio Network) [16] 發展而來。封包無線網路原本是因為軍事通訊的需求而開始研究，至今已經研究了將近 20 年。美國 DARPA (Defense Advanced Research Project Agency) 早在 1972 年就開始封包無線網路的研究專案，開發封

包無線網路在戰場環境下資料交換的應用。完成專案之後，在1993年DAPRA又開始了一項高殘存性自適應網路 (SURAN, SURvivable Adaptive Network) 的研究專案，延續封包無線網路的成果並加以擴展，為了能夠支援更大的網路規模，並且還要開發能夠適應戰場上快速變化環境的自適應網路協定。DARPA又在1994年開始一項稱作全球移動資訊系統 (GloMo, Global Mobile Information Systems) 的研究專案。一樣是延續在封包無線網路中的成果，並對這些成果進行能夠滿足軍事應用需要的全面深入研究：可快速展開、高抗毀性的移動資訊系統，一直持續至今。而1991年成立了一個IEEE 802.11的標準委員會，其採用了「Ad Hoc」一詞來一併描述這些特殊的對等式無線行動網路。

### 2.1.2 無線行動隨意網路的特徵

Ad Hoc Network 是一種無線網路的特殊架構。網路底下所有節點皆具有平等的地位，不需要任何的中心控制節點。網路中的節點不單單只具有主機所需的功能，同時也有路由的能力，相較於傳統的無線網路，它具有以下特點[16]：

#### 1. 無基礎建設或中心控制節點

Ad Hoc Network 不需要基地台、存取點 (Access Point) 或是

任何的控制中心。所有節點的地位平等，是一種對等式網路。

節點隨時可以加入或離開網路。任一節點的損壞並不會影響整個網路，具有高抗毀性。

## 2. 自我組織

建置Ad Hoc Network 無需使用任何需要預先建設的網路設施。透過分層協定和分散式演算法，節點產生各自的連線方式，並透過無線電訊號就可以快速且自動的組成獨立的一個網路。

## 3. 多點式跳躍連接

當節點要與其傳輸範圍之外的節點進行資料交換時，需要中間節點來作多點式跳躍轉接。Ad Hoc Network與固定網路的多點式跳躍不同點在於Ad Hoc Network的多跳路由 (Routing) 是由普通的網路節點完成的，並不是由專門作路由的路由設備（如路由器）完成。

## 4. 動態拓撲

Ad Hoc Network是一種動態的網路環境。網路中各節點具有相當高的移動性，因此Ad Hoc Network的拓撲架構不會是一個固定的架構，而是隨時都在變化當中。

## 2.2 無線行動隨意網路路由協定

在Ad Hoc Network裡，移動節點透過多點式跳躍連接實現彼此間的通訊。整個網路中沒有固定的基礎設施，例如基地台。網路內任一個節點都可作為路由器，向其它節點轉送資料。而開發一種能有效地找到節點間路徑的動態路由協定就成為Ad Hoc Network設計的關鍵。Ad Hoc的路由協定需要能夠實現以下幾點功能[14]：

### 1. 能偵測網路拓撲結構的變化

Ad Hoc路由協定要能夠偵測到網路拓撲的動態變化。因為Ad Hoc網路需要進行多點式跳躍通訊，所以路由協定必須確保路徑中的鏈結具有高度的可靠性。Ad Hoc Network中的節點必須知道它的周圍環境和可以與它直接進行通訊的節點。Ad Hoc Network裡提供網路連接的方法主要有兩種：平面式路由網路結構和分層路由網路結構，圖2-1為兩種結構的示意圖。在平面路由網路結構中，所有的節點地位都是平等的，每條路徑都是基於對等的連接。但是在階層式路由結構中，較低層至少要有一個節點作為與較高層聯繫的閘道。

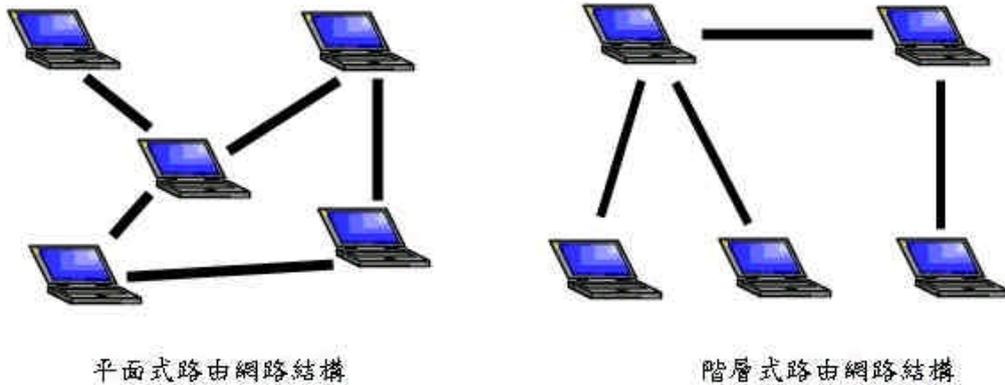


圖2-1 平面式與階層式路由網路結構

## 2. 維護網路拓撲的鏈結

因為每個移動節點會隨時改變位置，所以網路拓撲是動態且頻繁的在變化。如此一來，Ad Hoc Network的路由協定為了維持節點之間的鏈結具有較高的可靠性，它必須動態更新鏈結狀態和對重新配置。如果採用中心控制的路由演算法，為了把鏈結狀態改變的節點資訊傳送到其他所有的節點，就會消耗過多的時間和電力，顯然是不太適合的。

## 3. 高度自適應的路由

相對於有線網路裡的靜態節點，Ad Hoc Network要求一個高度自適應的路由機制，來處理快速的拓撲變化。而傳統的路由協定，如距離向量 (Distance Vector) 和鏈結狀態 (Link State) 演算法，必須在路由器之間作大量路由訊息的交換，因此在Ad Hoc Network裡都不能有效地工作。所以

針對Ad Hoc Network的特點，提出了新的路由演算法。大致上來說，這些路由演算法可以分為兩種類型：Table Driven、On-Demand。

Table Driven路由協定又可稱作先行式 (Proactive) 的路由協定，採用週期性的廣播封包，來交換路由資訊。每個節點皆維護到網路底下所有其他節點路徑的資訊。Table Driven路由的優點是當節點需要與其他節點作資料交換時，只要路由存在，發送訊息的延遲時間就很小；缺點則是Table Driven路由協定需要花費較高代價（如頻寬、電源、CPU資源等），使路由表能夠跟上當前網路拓撲結構的變化，但高度動態變化的拓撲結構又可能使這些花費高代價得來的路由表內容變成無效訊息，路由協定始終處於不收斂狀態。目前，這種類型的Ad Hoc Network路由協定已提出了幾種機制，用以改善這些方面的性能。

On-Demand路由協定又可稱作反應式(Reactive)的路由協定，是根據發送節點的需求，按需求進行路由以找到路徑，網路拓撲結構和路由表內容也是按需求建立的，所以其內容可能僅僅是整個網路拓撲結構資料的一部分。On-Demand路由的優點是不需要週期性的廣播路由訊息，節省了一定的網路資源；缺點是在作資料交換時，因沒有通往

目的節點的路徑，要臨時開始路由來尋找路徑，所以資料往往需要等待一定的延遲時間，並且由於路徑的尋找過程通常採用氾濫式廣播 (Flooding Broadcast) 進行搜尋，這在某些程度上也抵消了On-Demand節省一定網路資源的好處。

Table Driven與On-Demand的路由協定各有利弊。Table Driven的優點在於路徑搜尋快速，但平時需週期性的交換路徑資訊，造成系統負擔較大；而On-Demand的優點是可同時搜尋到多條路徑，平時不需交換路徑資訊，節省一定的資源，可是若遇到大規模的網路結構時，資料交換與尋找路徑的廣播封包同時傳輸，會降低網路效能。表2-1整理出兩種路由演算法類型的優缺點。

表2-1 三種類型的路由協定效能比較表

	Table Driven	On-Demand
維護路徑的負擔	高	低
路徑尋找速度	快	慢
尋找路徑時對網路的影響	低	高
節點移動時對網路的影響	高	低
網路資源的消耗	高	低

## 2.2.1 Table Driven 路由協定

以下介紹兩種常見的 Table Driven 路由協定。

### 1. DSDV (Destination-Sequenced Distance-Vector)

DSDV [12]是以傳統的 Bellman-Ford 路由機制為基礎的 Table Driven 演算法，普遍使用於 Ad Hoc Network 中。它需要每一個節點周期性的作路由廣播以更新路由表。DSDV 路由協定相對於傳統的距離向量協定的優點在於它透過序列機制保證了網路中節點間鏈結的可靠性。網路中的每一個節點在這種路由機制中都有一個路由表。路由表中含有所有可以到達的目的地節點、到達這些節點的距離以及這個目的地節點的序列號。這些路由表在網路中周期性的廣播以維持各節點的連接。DSDV 在路由協定中加上了序列號來區分路徑的新舊，解決了距離向量路由中的 Looping 問題。當節點收到其他節點傳來的路由訊息時，會去查看路由表中的目的地節點，是否有出現在自己的路由表中，若沒有，則加進自己的路由表中，若有，則去比較此目的地節點的序列號，如果比本身路由表中的序列號大，代表此路由資訊為較新的，便馬上更新，並廣播出去；如果序列號相同，則保留最短的路徑資訊。另一個優點是它採用了兩種 Route update 封包：Full dump 和

Incremental。當需要廣播全部的路由表時，使用 Full dump 封包；在下次使用 Full dump 封包前，更新路徑資訊都使用長度較小的 Incremental 封包。如此一來，減少了廣播路由表時所需的大量網路頻寬。

## 2. OLSR (Optimized Link State Routing)

OLSR [3]既然是Table Driven路由協定的一種，那麼它也需要週期性的廣播其路由資訊。但不同的是，每個節點的路由資訊並不是由節點本身來廣播的，而是從自己周遭的鄰近節點選擇一個稱為MPR (Multi-Point Relay)的節點來幫忙廣播，節點本身可稱作MPR Selector。每個MPR會定時發送一個Hello封包來告知整個網路自己的存在。也藉由交換Hello封包來紀錄每一條路徑，並將所紀錄的路由表回報給MPR Selector。在OLSR的網路中，每個節點都會選擇一個MPR，而MPR會重複，因此降低網路中廣播路由的封包數量，減少網路負載。

## 2.2.2 On-Demand 路由協定

接著介紹兩種常見的 On-Demand 路由協定。

### 1. AODV (Ad Hoc On-Demand Distance Vector)

AODV [13] 路由協定建立在 DSDV 的基礎之上。AODV 和 DSDV 不同點在於 AODV 只在需要的時候建立路徑，而不必像 DSDV 一樣要去維護包含所有節點的路由表。在 AODV 中，若是節點不在所選擇的路徑之中，就不會去維護路徑資訊，也不參與路由表的交換。當一個來源節點想傳送資料給某個目的地節點，但是自己本身的路由表未包含到目的地節點的路徑時，會啟動路徑搜尋程序去找出目的地節點的位置。它會廣播一個 Route Request (RREQ) 封包給它鄰近的節點，接到此封包的節點再轉送給它周遭的節點，依此類推。直到找到目的地節點，或是轉送的過程中某一個節點已含有到達此目的地節點的路徑資訊時，並且此路徑資訊還沒逾時，RREQ 的廣播動作才會停止，然後回覆一個 Route Reply (RREP) 封包給來源節點。AODV 假設節點與節點間的連線都是雙向的，圖2-2為AODV搜尋路徑的運作方式。

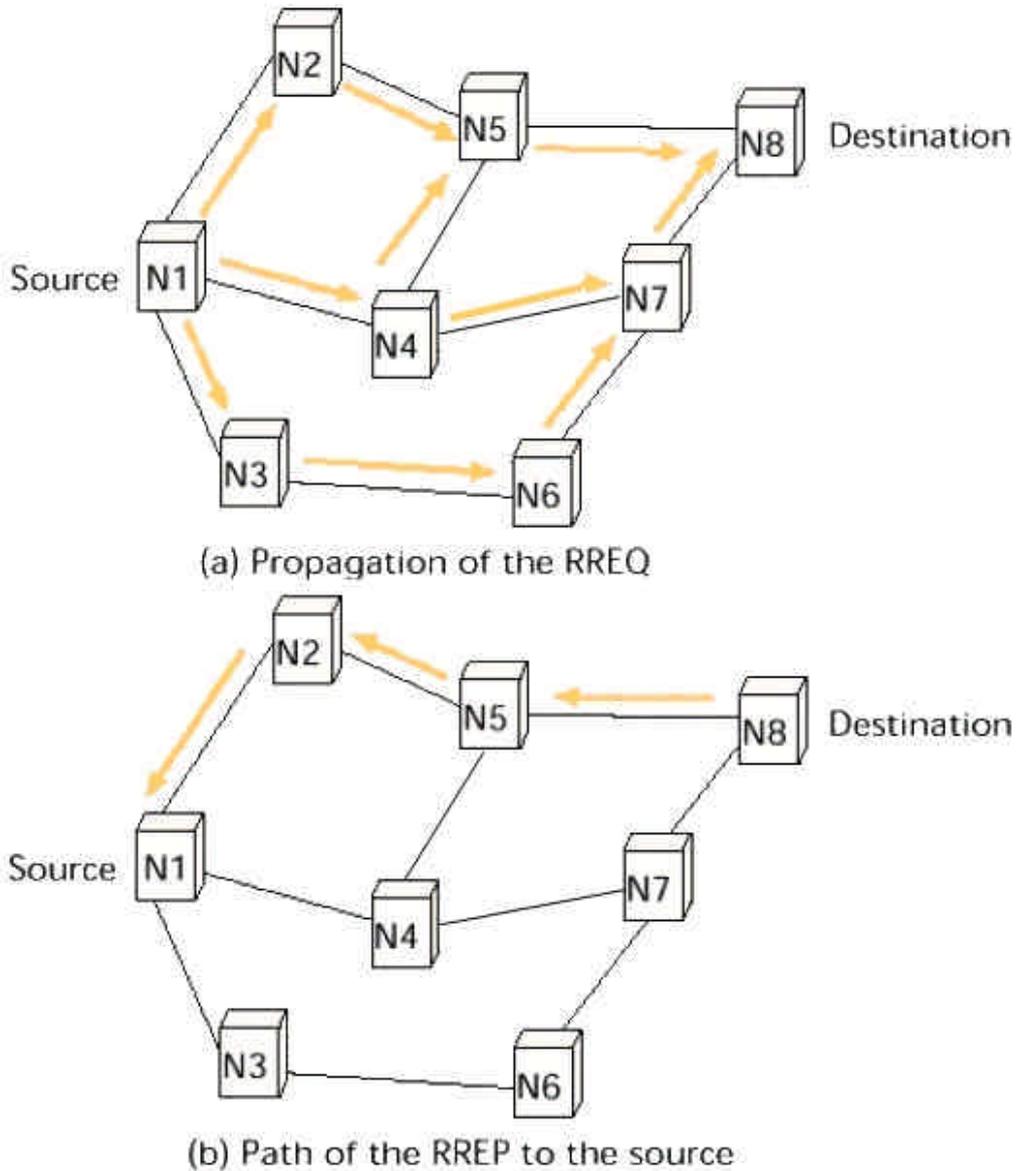


圖2-2 AODV路徑搜尋運作圖

## 2. DSR (Dynamic Source Routing)

DSR [7]與AODV類似，也是從來源節點廣播一個Route Request封包去搜尋目的地節點，但DSR不像AODV有路由表，所以當每次傳輸資料時，就必須作一次路徑搜尋程序。

且Route Request每經過一個節點，就會將此節點ID紀錄在

Route Record裡，當Route Request封包最後到達目的地節點時，就會有此路徑所有的節點資訊，目的地節點由眾多的Route Request選擇一條最佳的路徑，根據其Route Record回覆一個Route Reply給來源節點，之後所有要傳送給目的地節點的封包皆含有這個Route Record。圖2-3為DSR搜尋路徑的運作方式。

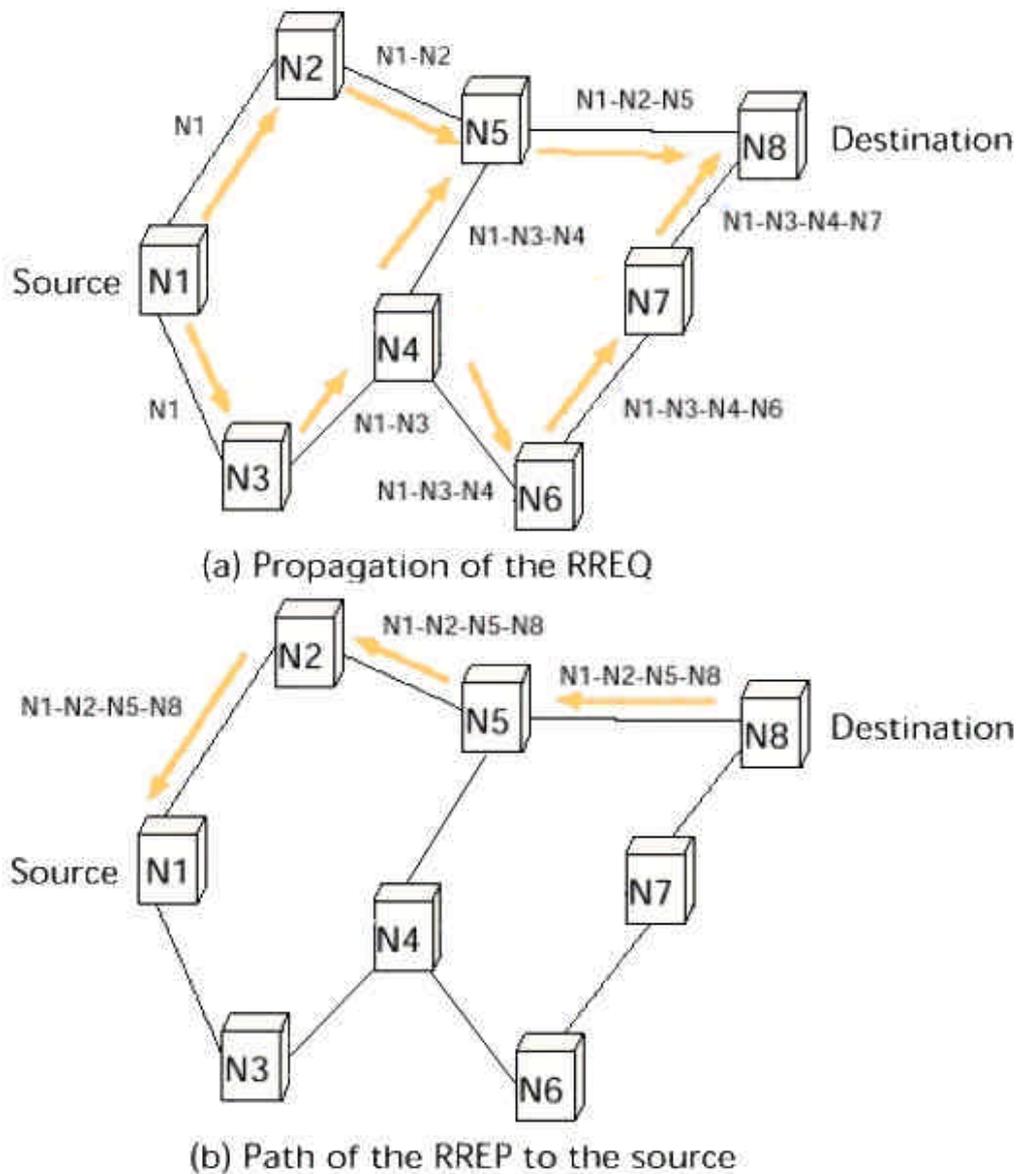


圖 2-3 DSR 路徑搜尋運作圖

## 2.3 TCP 傳輸控制協定

TCP/IP 為目前最常使用的通訊協定，而 TCP 提供的功能包括區段分割、連接導向的服務、可靠性的服務、應用程式間的流量控制以及壅塞控制服務。當有應用程式把資料傳到網路上時，為了要和其他應用程式公平分享頻寬並避免造成網路壅塞，TCP 藉著壅塞控制服務來控制傳送到網路上的資料量，也因此 TCP 的壅塞控制機制是影響 TCP 效能的最大主因，而依照壅塞控制機制的不同，產生了幾種不同的 TCP 版本，如 Tahoe、Reno、NewReno、Sack 以及 Vegas [1, 8] 等等。下面先介紹基本的壅塞控制機制，再分別對每個不同的 TCP 版本之間差異做介紹。

### 2.3.1 壅塞控制

壅塞控制主要分為五個階段：

1. Slow-start

早期開發的TCP版本在開始一個連線時會向網路發送大量的封包，這樣很容易導致路由器緩衝空間耗盡，讓網路發生壅塞，使得TCP的Throughput急遽下降。由於TCP傳送端無法知道網路資源目前的使用狀況，因此新建立的TCP連線不能一開始就發送大量資料，只能逐步增加每次發送的資料量，以

避免上述的情況發生。換句話說，當建立新的TCP連線時，壅塞窗口 (Congestion Window, cwnd) 初始值為一個封包大小。傳送端按cwnd大小發送資料，每收到一個ACK確認，cwnd就增加一個封包的發送量，這樣cwnd就將隨著Round Trip Time呈指數增長，傳送端向網路發送的資料量將急速增加。事實上，Slow-start一點也不慢，要達到每RTT發送W個封包的所需時間僅為 $RTT \times \log W$ 。由於在發生壅塞時，壅塞窗口會減半或降到1，因此Slow-start確保了傳送端的發送速率最多是網路頻寬的兩倍。

## 2. Congestion avoidance

如果cwnd的值大於Slow-start threshold (ssthresh) 的值，TCP就執行Congestion avoidance，此時，在每次收到一個ACK後，cwnd的值只增加 $1/cwnd$ ，也就是說，在一個RTT內，cwnd才增加1個segment，所以在Congestion avoidance階段，cwnd不是呈指數增加，而是線性增加。

## 3. Fast retransmission

Fast retransmission是當TCP傳送端收到到三個相同的ACK時，即認為有封包遺失，則傳送端重傳遺失的封包，而不必等待Timeout。同時將ssthresh設置為目前cwnd值的一半，並

且將cwnd值重設。

#### 4. Fast recovery

傳送端在重送遺失的封包後，在重送的封包ACK回來前，每收到一個重複的ACK，就將cwnd值加1。

#### 5. Timeout retransmit

當傳送端送出封包後，在一定的時間內，沒有收到相對應的ACK，則認為此封包遺失，便重送此封包。

### 2.3.2 Tahoe

Tahoe為早期的TCP版本，除了具備TCP的基本架構，Slow-start、Congestion avoidance以及回覆遺失的封包之外，還加入了Fast retransmit。在Fast retransmit階段中，會將sssthresh的值設為cwnd值的二分之一，並且將cwnd值重設為1，進入Slow-start階段。

### 2.3.3 Reno

Reno為目前使用最廣泛的TCP版本，其修改了Tahoe在重送遺失的封包之後，就進入Slow-start階段的方式，取而代之的是Fast recovery。Reno在Fast retransmit階段重送遺失的封包之後，會將sssthresh的值以及cwnd值都設為封包遺失時cwnd值的二分之一，並進入Fast recovery

階段。若在Timeout前收到重送遺失封包的ACK，就進入Congestion avoidance階段，否則就進入Timeout Retransmit階段。

### 2.3.4 NewReno

在Reno版本中，若同時有多個封包遺失，大部分都必須等到Timeout之後才能重送遺失的封包。這是因為在Reno中，同時有多個封包遺失時，Reno收到Partial ACK便結束Fast recovery。會收到Partial ACK即表示Fast retransmit只重新送達了部份遺失的封包而已。而在Reno結束Fast recovery進入Congestion avoidance後，對於其他未重送的封包來說，常常沒有足夠的重複ACK來觸發重送機制，只好等待Timeout。Timeout對於TCP效能有非常大的影響，在等待Timeout這段時間無法送出新的資料，而且Timeout之後cwnd值會被重設為1。於是NewReno就修改了Reno的Fast recovery機制。NewReno在收到Partial ACK之後並不會馬上結束Fast recovery，而會等待所有遺失的封包皆重送過後才結束Fast recovery，使得NewReno遇到多個封包遺失時，不需等到Timeout便可重送完所有遺失的封包，減少Timeout對效能的影響。

### 2.3.5 Sack

除了NewReno的方法外，要解決大量封包遺失的問題還有一個方式，就是讓傳送端知道哪些封包已經傳送到，哪些封包已經遺失。Sack修改接收端在送重複的ACK時，同時在ACK當中加入已收到的連續資料範圍，連續資料範圍與連續資料範圍之間間隔就是遺失的封包，藉此傳送端就可以知道哪些封包應該要重送，並可在一個RTT內重送一個以上的封包。

### 2.3.6 Vegas

Vegas是以RTT (Round-Trip Time) 為基礎的擁塞控制機制。它提出了一套新的壅塞控制演算法，利用RTT的值來調整cwnd的大小，而不用由封包的遺失狀況來偵測網路是否壅塞。因此Vegas較前幾個版本公平，並且其cwnd並非週期性的增減，因而提供一個較穩定的cwnd值。Vegas提出了三個方法來達到增加傳送的throughput以及降低封包的遺失。

1. 新的重送機制

當Vegas收到一個重複的ACK時，會去計算RTT是否有大於Timeout的值。RTT若是大於Timeout，則不需等到三個重複的ACK便立即重送此封包。若重送的封包回傳新的ACK時，

Vegas也會檢查在這之後的第一個或第二個回來的ACK，以期盡早發現Timeout，並重送這些遺失的封包。

## 2. 改進的Congestion avoidance

Vegas藉由計算RTT的值來調整cwnd的大小，其演算法如下：

$$Diff = Expected - Actual = \left( \frac{Window\ Size}{BaseRTT} - \frac{bytes\ transmitted}{measured\ Rtt} \right)$$

where BaseRTT = minimum of all measured RTT

$$Congestion\ window = \begin{cases} \text{linearly increases during nextRTT, } Diff < a \\ \text{linearly decreases during nextRTT, } Diff > b \\ \text{unchange, } a < Diff < b \end{cases}$$

當Diff的值小於a時，表示傳送速率太慢，便增加cwnd的值以加快傳送的速率；而當Diff的值大於b時，表示傳送速率太快，便減低cwnd的值以降低傳送的速率。

## 3. 較有效率的Slow-start

為了降低因為傳送速率太快導致封包遺失的情況發生，因此Vegas修改了Slow-start演算法，大約要經過兩個RTT的時間後，cwnd值才會增加一倍。同時將佇列長度維持在a與b之間，超過此區間便從Slow-start階段進入Congestion avoidance。

## 2.4 無線行動隨意網路中影響 TCP 效能的因素

在[6, 8, 9]有提到許多在 Ad Hoc Network 中會影響到 TCP 效能的因素，整理如下：

1. TCP 版本：不同 TCP 版本有不同的壅塞控制機制，當封包遺失時便會觸發這些機制啟動，在無線的環境下很容易遺失封包，因此這些機制是影響效能的一個因素。
2. 路由協定：不同的路由協定有不同的路徑搜尋方式，也有不同的路徑回復方式，當封包的計時器歸零時，路徑搜尋或路徑回復尚未完成，便會觸發 TCP 的壅塞控制機制，也對效能造成影響。
3. 節點移動速率：當節點移動速率越快，表示兩兩裝置間的鏈結越容易發生改變，當鏈結改變，路徑也跟著改變，傳送端必須重新建立一條傳輸路徑，會延遲發送時間，也影響到效能。
4. 拓樸大小（節點個數）：若是節點的個數越多，傳送的封包也越容易發生碰撞，觸發 TCP 的壅塞控制機制次數也會增加，

進而對效能造成影響

5. 電量： 行動裝置本身就有電量的先天限制，若是傳輸時因為建立路徑等等動作造成電量消耗過大，當裝置的電量用完時，便會消失在環境中，而裝置間的鏈結跟著改變，傳輸效能也會因此受到影響。

## 第三章 系統實驗設計與模擬環境

在本章節中，會介紹我們將選擇那些會影響效能的因素去做實驗、實驗設計以及實驗的環境參數設定。

### 3.1 系統實驗設計

由前面可以得知在 Ad Hoc Network 中，影響效能的因素有很多，本篇論文以路由協定、節點移動速率與 TCP 版本為主，並設計三種實驗來比較這三種因素：

1. 在固定移動速率與路由協定下，觀察不同TCP版本的效能差異。
2. 在固定移動速率與TCP版本下，觀察不同路由協定對於TCP的效能影響。
3. 在固定路由協定與TCP版本下，觀察不同移動速率對於TCP的效能影響。

因為在目前有線網路的環境中，使用 TCP 傳輸時，大多都是 Reno 的版本，因此實驗除了整個環境中，傳輸節點都設為同樣的 TCP 版本，各別實驗五個版本外，另外也讓環境中的傳輸節點分為兩群，一群使用 Reno 的版本做傳輸，另外一群則分別使用其他四種版本，看看四種版本與 Reno 混合下，是否還能正常傳輸。

## 3.2 模擬環境

本篇論文使用 NS-2 來作為實驗的模擬工具，藉由 NS-2 模擬器算出各 TCP 版本在不同的 Routing 以及最大移動速率下所呈現的 Throughput、丟棄的封包數等等。

在實驗中，我們設置 20 個節點，在 670 公尺見方的範圍內，總共有 14 條 TCP 的傳輸線路，每條 TCP 皆使用 FTP 作資料傳輸，封包大小設為 512 bytes。傳送端以及接收端是亂數下去配對的，整個模擬時間持續 100 秒，最大節點移動速率分別有每秒 5 公尺、每秒 10 公尺、每秒 15 公尺以及每秒 20 公尺，節點的移動也是由 Random Waypoint 模組來決定，當節點移動到目的地之後，將不停頓並繼續亂數選擇下一個目的地前進。網路頻寬為 2M bps，節點皆透過標準的 802.11 MAC 層互相連接，每個節點的傳輸範圍在 250 公尺之內。路由協定選擇兩種先行式的：DSDV 以及 OLSR。上列參數整理如表 3-1 所示。

表3-1 模擬實驗參數設定表

節點數	20
模擬環境範圍	670 m × 670 m
模擬時間	100 s
節點最大移動速率	5, 10, 15, 20 m/s
節點移動模組	Random Waypoint
到目的地後的停頓時間	0 s
傳輸頻寬	2M bps
傳輸範圍	250 m
路由協定	DSDV, OLSR

## 第四章 實驗結果與討論

這個章節中，會針對每個實驗結果去做分析與討論，並說明前面所選的三種因素是如何影響效能，以及影響的程度。

### 4.1 單一版本環境模擬結果分析

#### 4.1.1 Tahoe

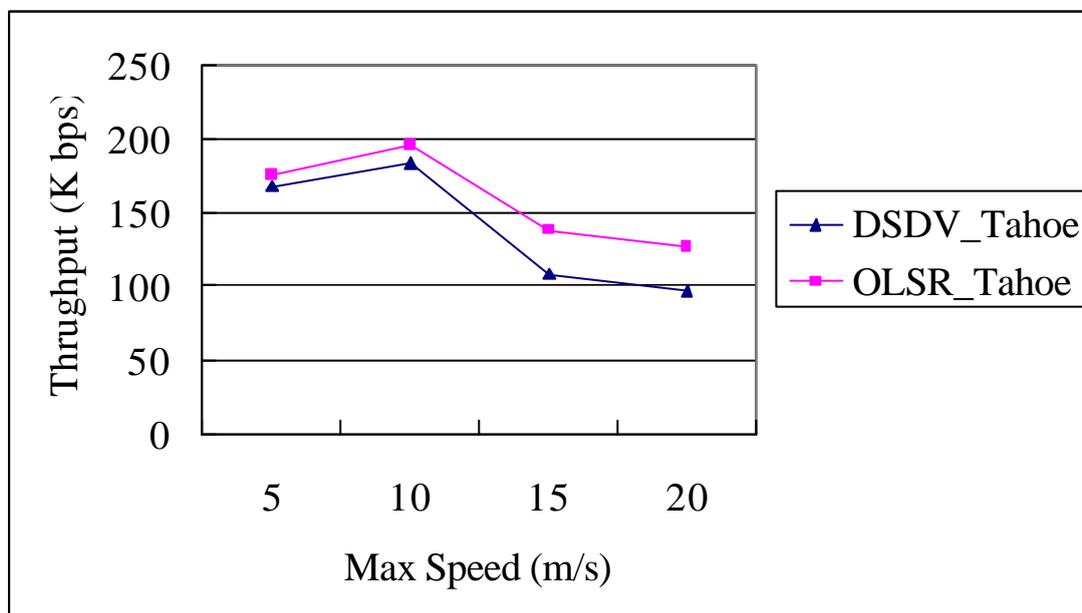


圖4-1 Tahoe在DSDV與OLSR下Throughput的比較

由圖4-1，我們可以發現最大移動速率越高時，效能會越來越低。這主要是因為當節點移動越快，路由表的有效時間越短，以至於會發生傳送端按照路由表所送出的封包，因為路由表的內容尚未更新，而非實際的節點所在位置與路徑，因此封包送到一半就沒有路徑繼續

傳輸，必須將封包丟棄後重新更新路由表，再重送一次封包，導致效能降低。在節點最大移動速率為每秒10公尺時，效能卻比節點最大移動速率為每秒5公尺來的好，這是因為在節點最大移動速率為每秒5公尺的環境中，某些隨機的傳輸配對因為距離太遠，而節點的移動速率又十分緩慢，就會有很長的時間甚至從頭到尾都沒有傳輸路徑，因此反而效能比節點最大移動速率為每秒10公尺來的低。因為OLSR使用MPR節點來做路由更新，所產生的控制訊息封包數量會比DSDV來的少，且因此降低控制訊息所消耗的頻寬，進而提高整體的效能。

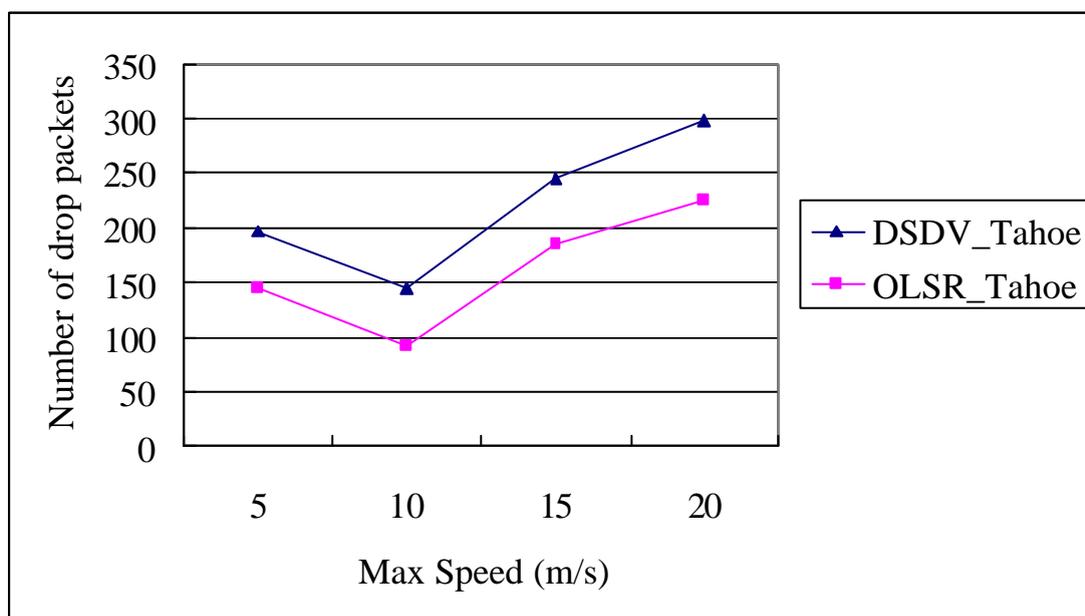


圖4-2 Tahoe在DSDV與OLSR下丟棄封包數量的比較

圖4-2則是Tahoe在兩種路由協定下所丟棄的封包數量。剛剛提

到，由於當節點移動越快，路由表的有效時間越短，以至於會發生傳送端按照路由表所送出的封包，因為路由表的內容尚未更新，而並非實際的節點所在位置與路徑，因此封包送到一半就沒有路徑繼續傳輸，必須將封包丟棄後重新更新路由表，再重送一次封包，所以節點最大移動速率越大，封包丟棄的數量也就越多。同樣的，在節點最大移動速率為每秒5公尺的環境中，因為某些隨機的傳輸配對因為距離太遠，而節點的移動速率又十分緩慢，就會有很長的時間甚至從頭到尾都沒有傳輸路徑，導致這些送出的封包因沒有傳輸路徑而一直不斷重複丟棄再更新路由表的動作，增加了封包丟棄的數量。

#### 4.1.2 Reno

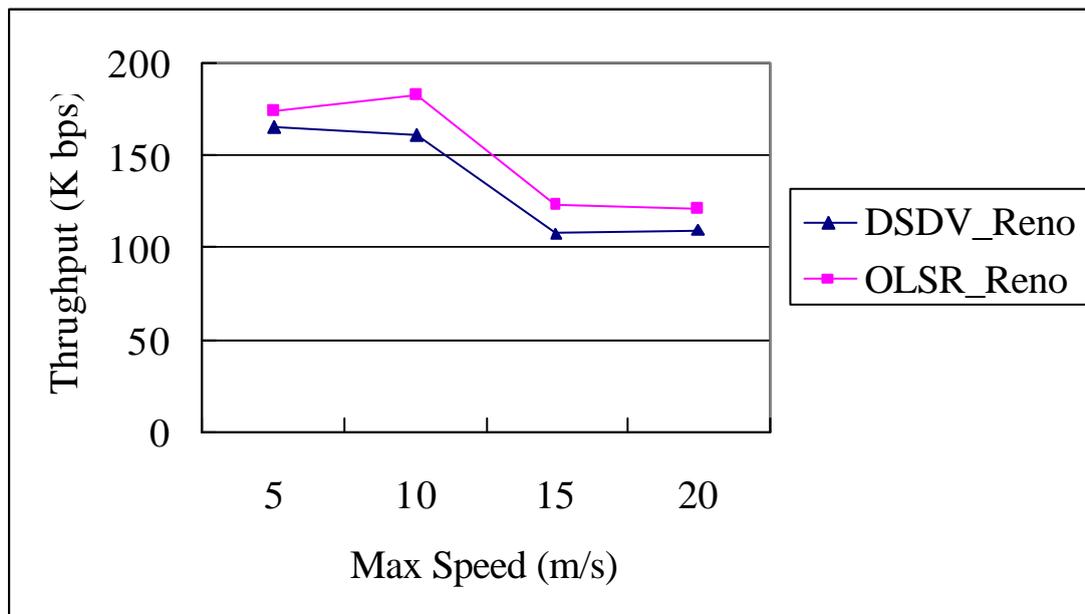


圖4-3 Reno在DSDV與OLSR下Throughput的比較

圖4-3顯示出其實Reno的狀況也類似於Tahoe，節點的移動速率越高，效能也會相對降低。OLSR仍舊保持較好的效能。

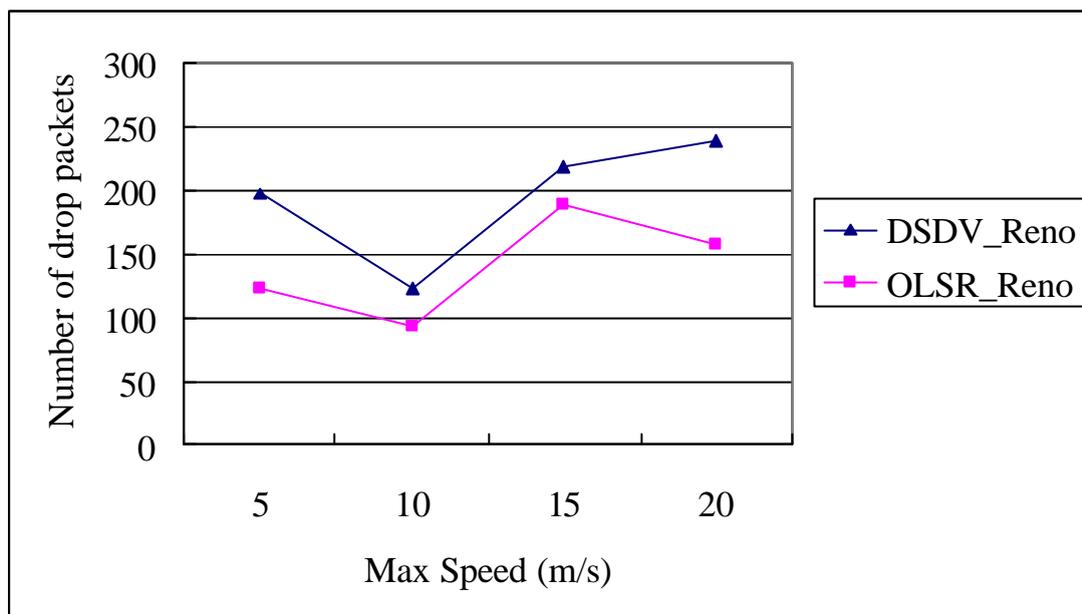


圖4-4 Reno在DSDV與OLSR下丟棄封包數量的比較

在圖4-4中，Reno的丟棄封包數量也是類似Tahoe，比較特別的地方在OLSR底下最大節點移動速率為每秒20公尺時。經過進一步的觀察後，發現在移動速率每秒20公尺時，有一些傳輸路徑的中間節點因為高速移動離開了路徑，在發生Router error之後，很短的時間內，有其他包含目的地節點位置的節點剛好移動到傳送端附近，在Timeout前就完成了路徑重新搜尋，因此封包丟棄的數量才比移動速率每秒15公尺時來的少。

### 4.1.3 NewReno

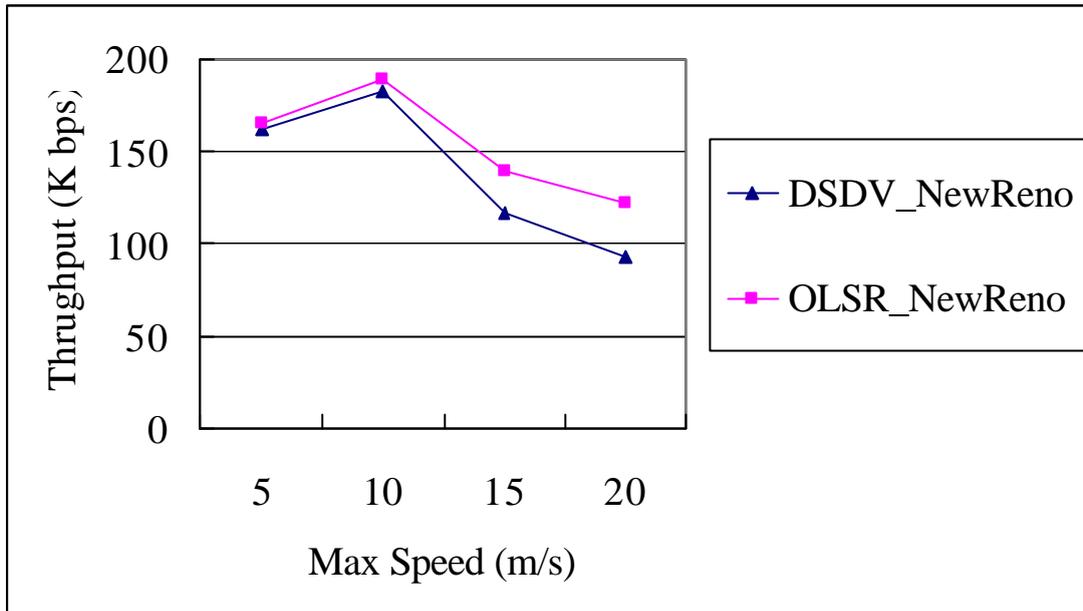


圖4-5 NewReno在DSDV與OLSR下Throughput的比較

圖4-5中，NewReno的情況也與之前相同，同樣因為在節點最大移動速率為每秒5公尺的環境中，某些隨機的傳輸配對因為距離太遠，而節點的移動速率又十分緩慢，就會有很長的時間甚至從頭到尾都沒有傳輸路徑，因此在節點最大移動速率為每秒10公尺時，效能比節點最大移動速率為每秒5公尺來的好。

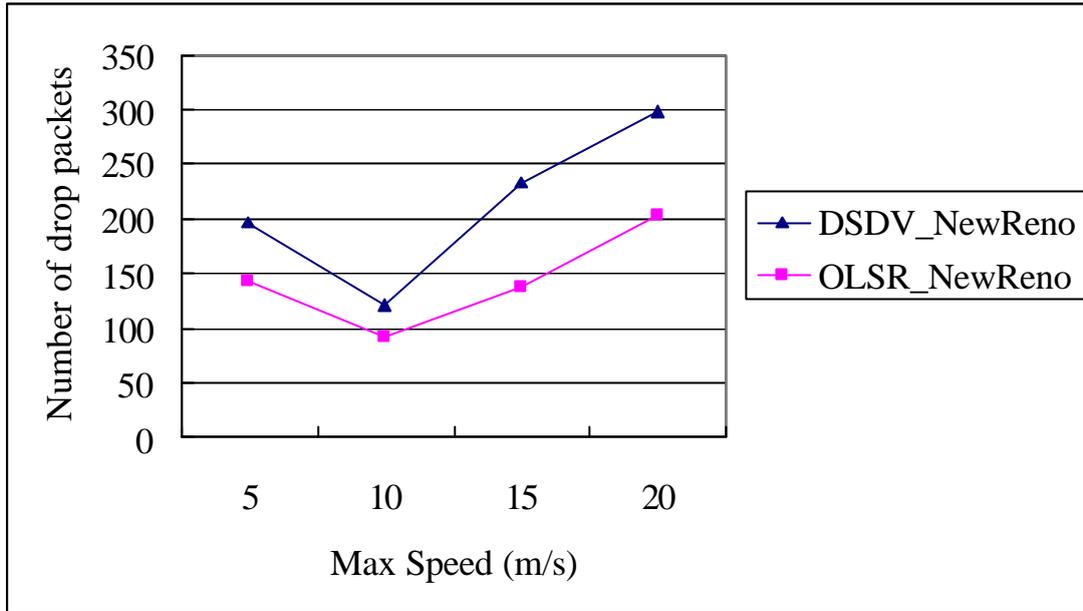


圖4-6 NewReno在DSDV與OLSR下丟棄封包數量的比較

圖4-6是NewReno在DSDV以及OLSR底下丟棄的封包數量。與Reno和Tahoe一樣，在節點最大移動速率為每秒5公尺的環境中，因為某些隨機的傳輸配對因為距離太遠，而節點的移動速率又十分緩慢，就會有很長的時間甚至從頭到尾都沒有傳輸路徑，導致這些送出的封包因沒有傳輸路徑而一直不斷重複丟棄再更新路由表的動作，增加了封包丟棄的數量，因此比節點最大移動速率為每秒10公尺的環境中，有較多的封包被丟棄。

#### 4.1.4 Sack

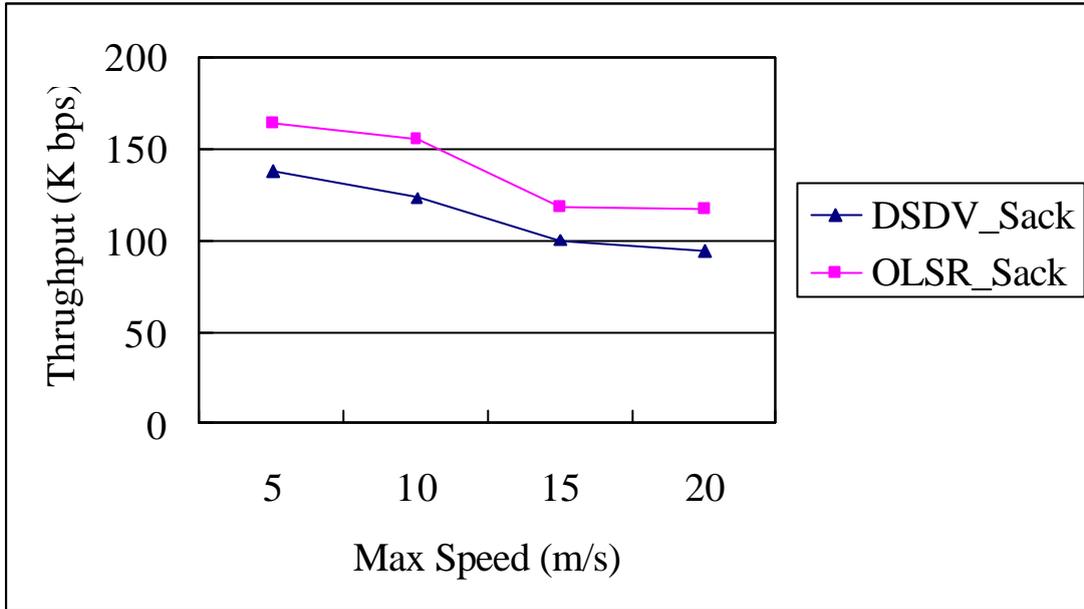


圖3-7 Sack在DSDV與OLSR下Throughput的比較

在圖4-7, Sack相對於前面的三個版本: Tahoe, Reno與NewReno, 所呈現出的效能就較為符合預期, 越高的移動速率, 效能就越低。

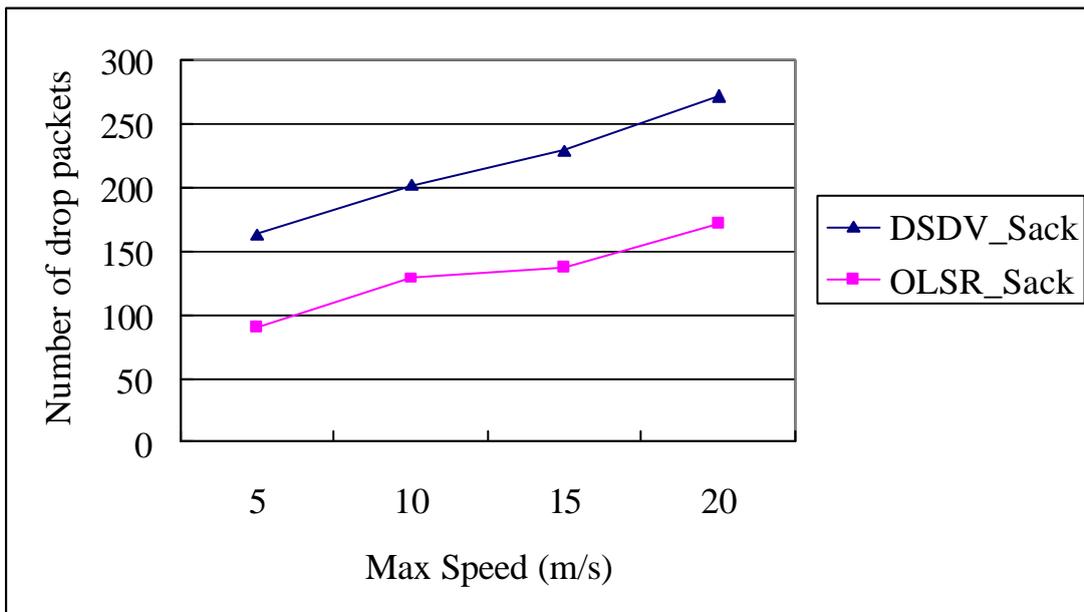


圖4-8 Sack在DSDV與OLSR下丟棄封包數量的比較

在圖4-8中，同樣的Sack相對於前面的三個版本：Tahoe、Reno與NewReno，所呈現出的效能就較為符合預期，越高的移動速率，封包的丟棄數量就越多。

#### 4.1.5 Vegas

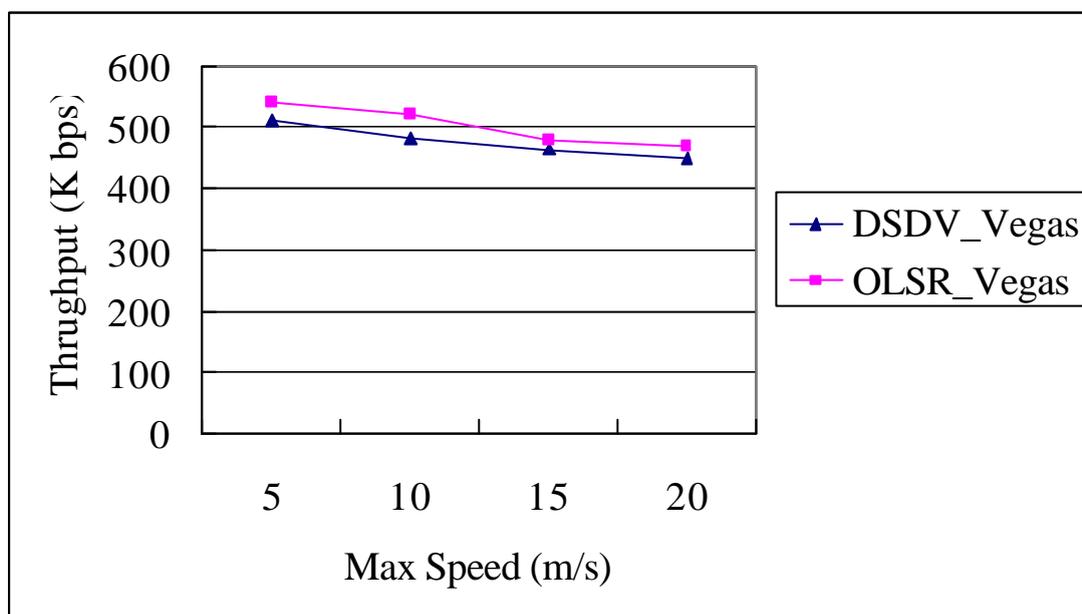


圖4-9 Vegas在DSDV與OLSR下Throughput的比較

由於Vegas可以藉由測量RTT來判斷網路是否壅塞，所以相對於前面幾個版本，在DSDV環境下與在OLSR的環境下，在圖4-9當中兩者呈現出的效能差異並不大。其實再進一步觀察後，發現Vegas在DSDV中，所傳送的封包數量，會比OLSR來的少，因為DSDV會花大量的網路頻寬用於路由資訊的更新，造成網路壅塞，而Vegas因此降低傳

送速率，以避免因網路壅塞發生封包碰撞丟棄的情形，也避免在DSDV更新路由資訊，消耗大量頻寬時做傳輸造成的效能不佳，所以在DSDV環境下與在OLSR的環境下，呈現出的效能差異才因此不大。

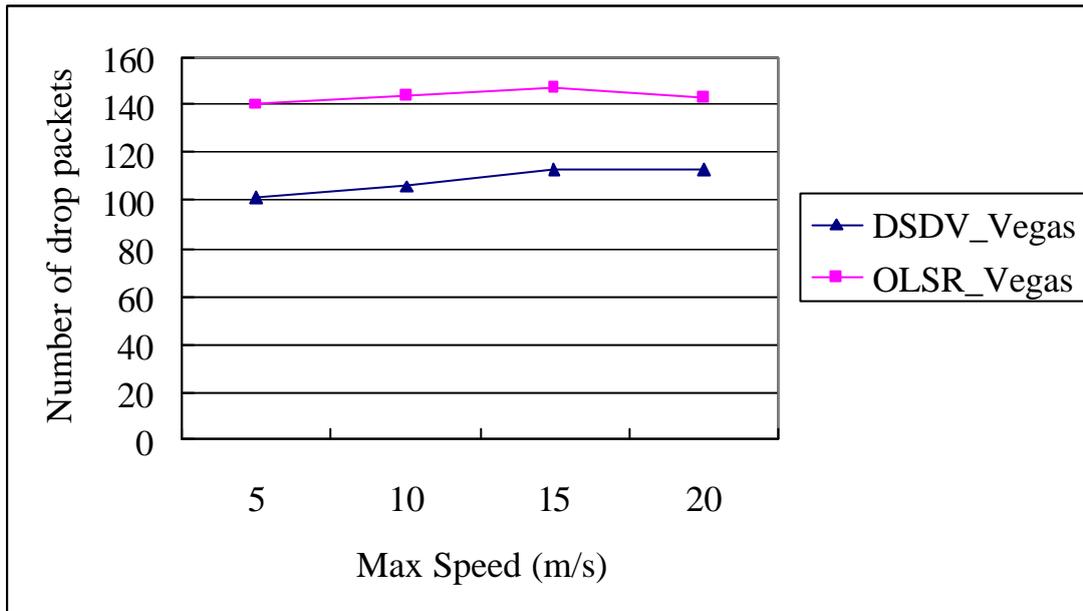


圖4-10 Vegas在DSDV與OLSR下丟棄封包數量的比較

剛剛提到Vegas會測量RTT來判斷網路是否壅塞，因此當DSDV花大量的網路頻寬用於路由資訊的更新，造成網路壅塞時，Vegas便降低傳送速率，以避免因網路壅塞發生封包碰撞丟棄的情形，也避免在DSDV更新路由資訊，消耗大量頻寬時做傳輸造成的效能不佳，所以Vegas在DSDV中，所傳送的封包數量，會比在OLSR中來的少，也因此在此圖4-10裡，丟棄的封包數量比在OLSR環境中來的少。

下面我們將五個TCP版本在相同的路由協定環境下，放在一起做比較。

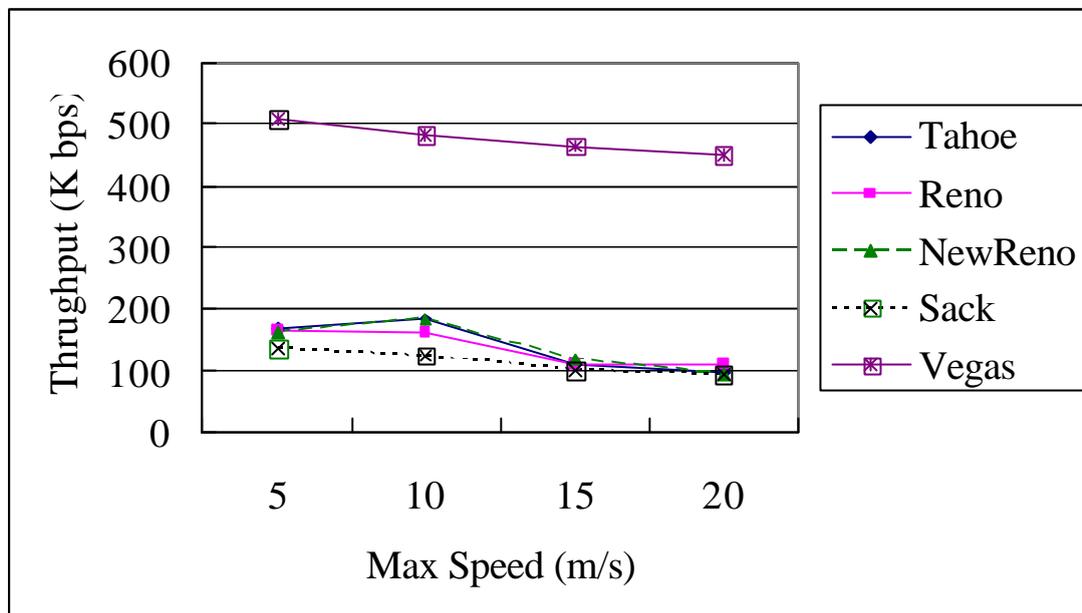


圖4-11 各TCP版本在DSDV下Throughput的比較

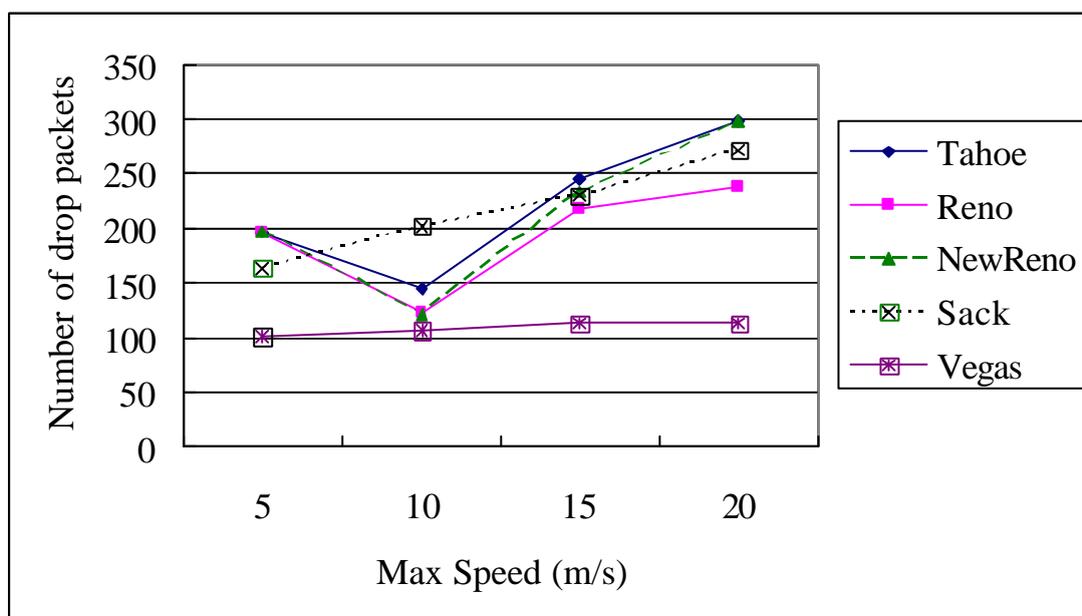


圖4-12 各TCP版本在DSDV下丟棄封包數量的比較

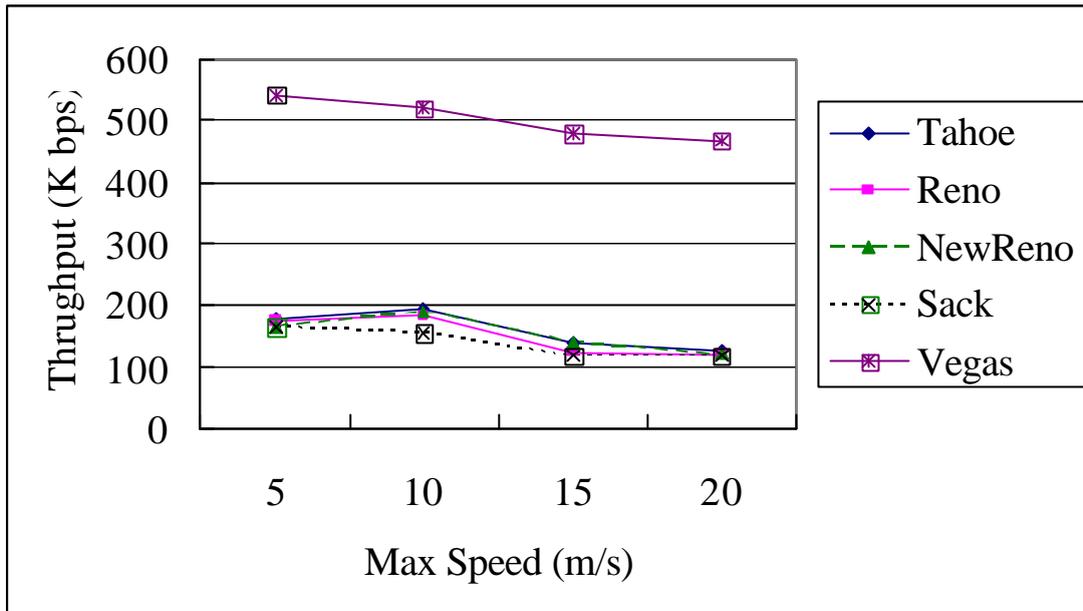


圖4-13 各TCP版本在OLSR下Throughput的比較

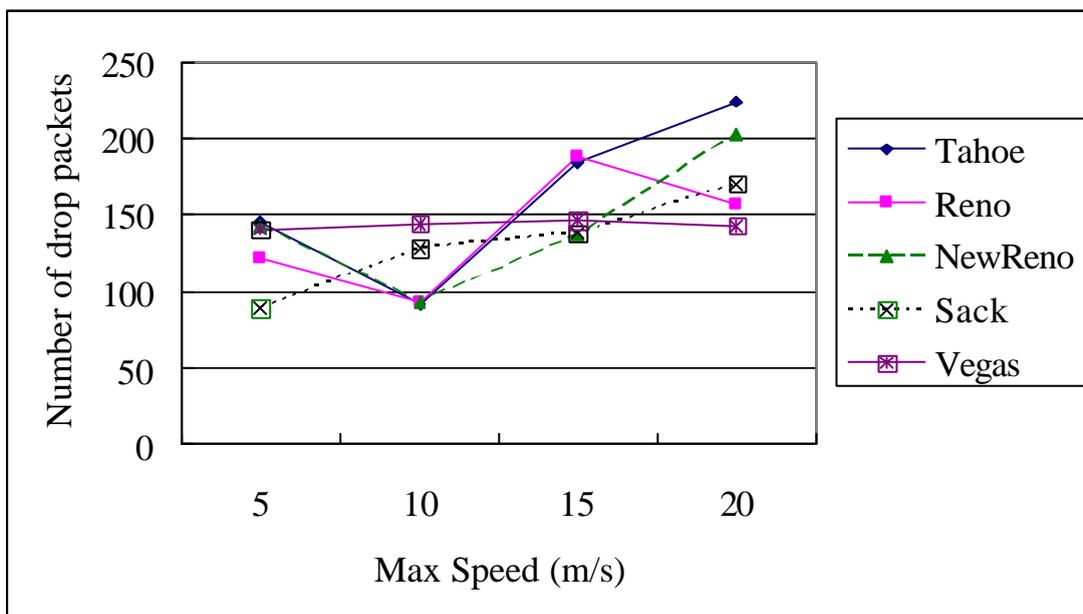


圖4-14 各TCP版本OLSR下丟棄封包數量的比較

由圖4-11與圖4-13可以發現，Vegas的效能不論在DSDV下或OLSR下，皆比其他四個版本要來的好。而在圖4-12與圖4-14，Vegas的丟棄

封包數量也不會因為移動速率的不同而有變化，皆維持在一個水平。

可以說Vegas利用本身的機制來調整傳輸速率，不與路由協定在更新路由資訊時搶頻寬造成效能降低，是十分適合在Ad Hoc Network上使用的TCP版本。

## 4.2 不同版本混合環境模擬結果分析

以上的實驗都假設所有節點的傳輸層使用一種TCP版本，但目前最廣為使用的版本還是Reno，因此接下來我們將其他四個TCP版本分別對Reno做混合實驗，將實驗中節點的傳輸層設定為不同的TCP版本，以觀察其他四個TCP版本是否能與Reno共存並正常運作。

### 4.2.1 Reno vs. Tahoe

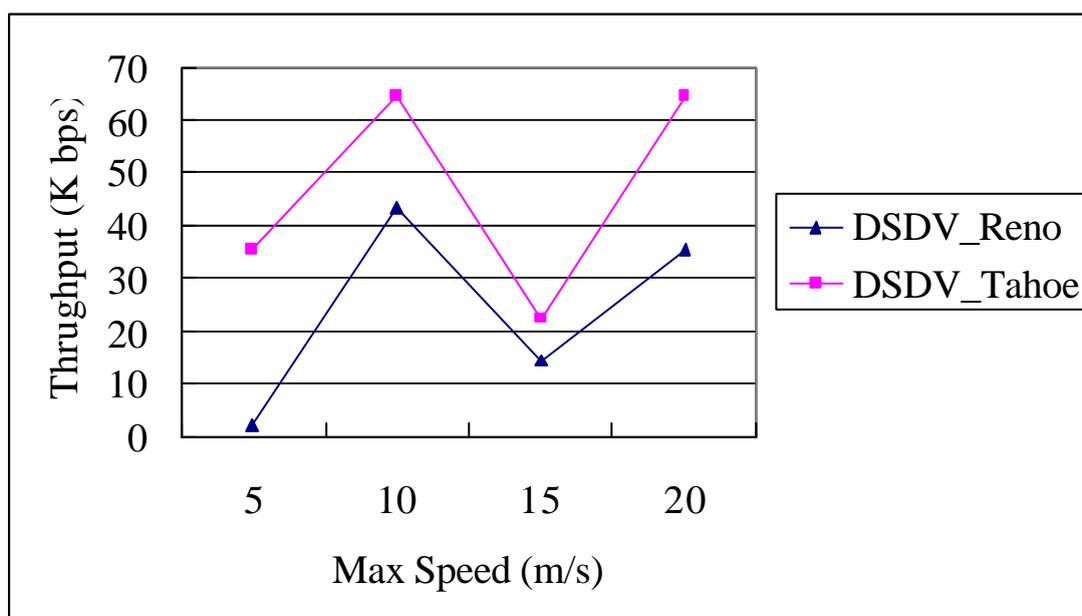


圖4-15 Reno vs. Tahoe在DSDV下的Throughput

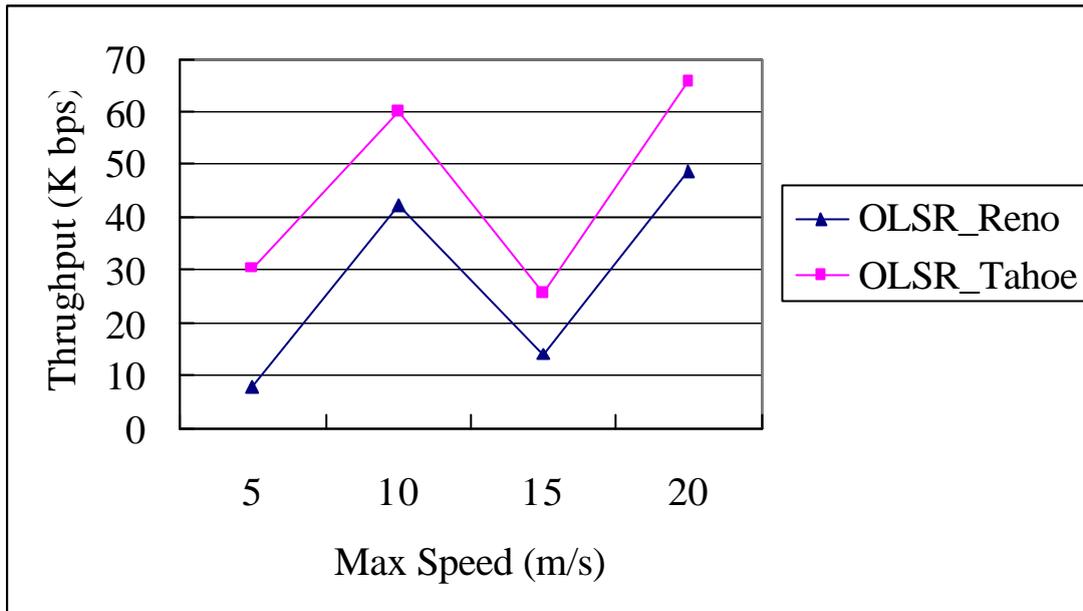


圖4-16 Reno vs. Tahoe在OLSR下的Throughput

一般來說，在有線的環境下，當有兩種以上的 TCP 版本混合運作時，會有搶頻寬的情況發生，造成環境當中某個 TCP 版本效能會比較好，而其他 TCP 版本效能降低。但在實驗中發現，由於在 Ad Hoc Network 底下做傳輸時，會先聆聽頻道是否閒置，若是閒置，才開始做傳輸，否則，則隨機等待一段時間後再重送。以圖 4-15 與 4-16 來說，Reno 使用比較侵略性的壅塞控制機制，會一直傳送封包直到網路發生壅塞。當頻道已被其他人佔用時，Reno 仍舊一直傳送大量封包出去，每送出一次就必須隨機等待一段時間後重送，次數越多等待時間越長，造成效能低落。

### 4.2.2 Reno vs. NewReno

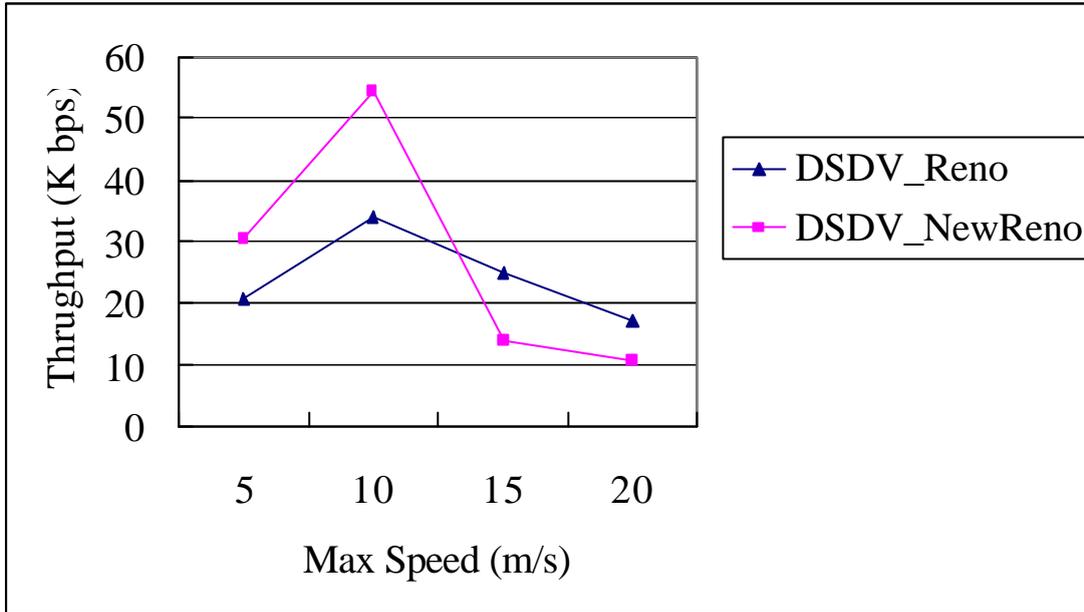


圖4-17 Reno vs. NewReno在DSDV下的Throughput

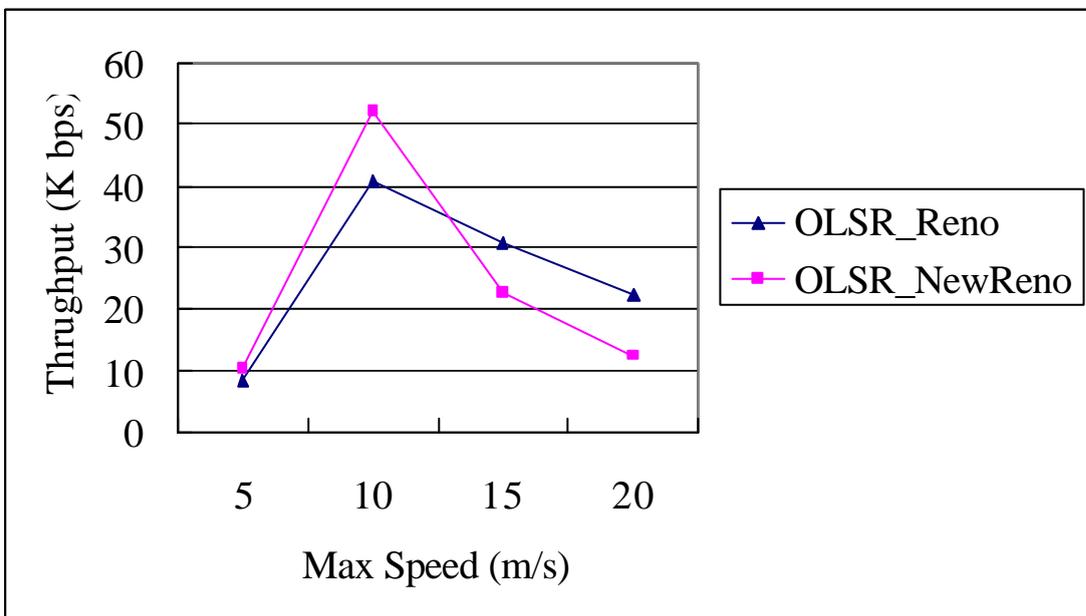


圖4-18 Reno vs. NewReno在OLSR下的Throughput

在圖4-17與4-18中，當節點最大移動速率超過每秒15公尺時，NewReno的效能會比Reno來的低。實驗發現到在高速移動的環境下，NewReno會常常停在等待所有遺失的封包皆重送後的階段，導致必須等到這些封包Timeout後，才能重新啟動，造成效能降低。

### 4.2.3 Reno vs. Sack

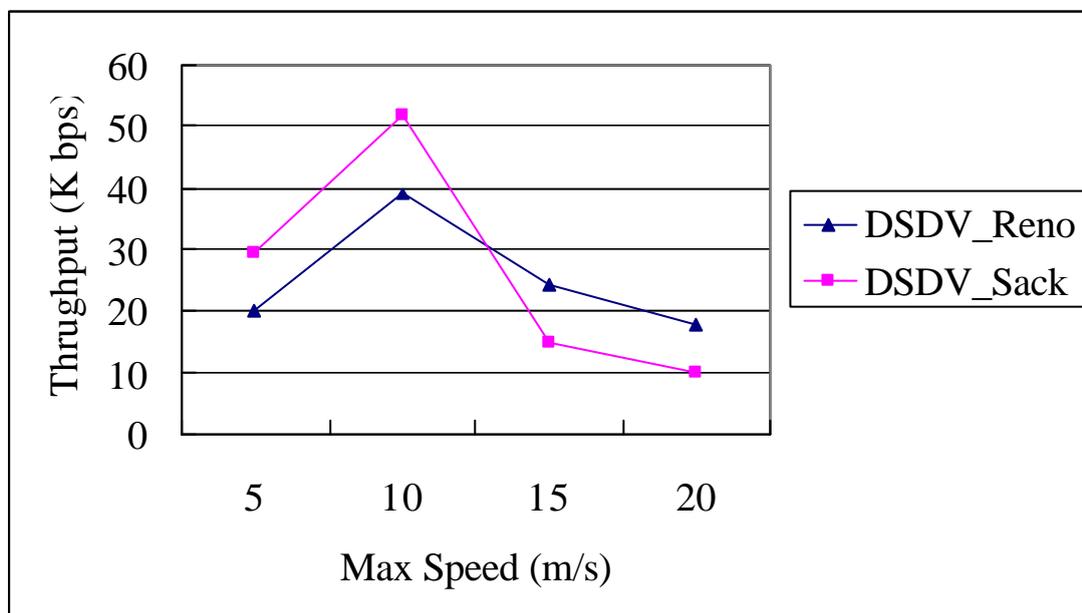


圖4-19 Reno vs. Sack在DSDV下的Throughput

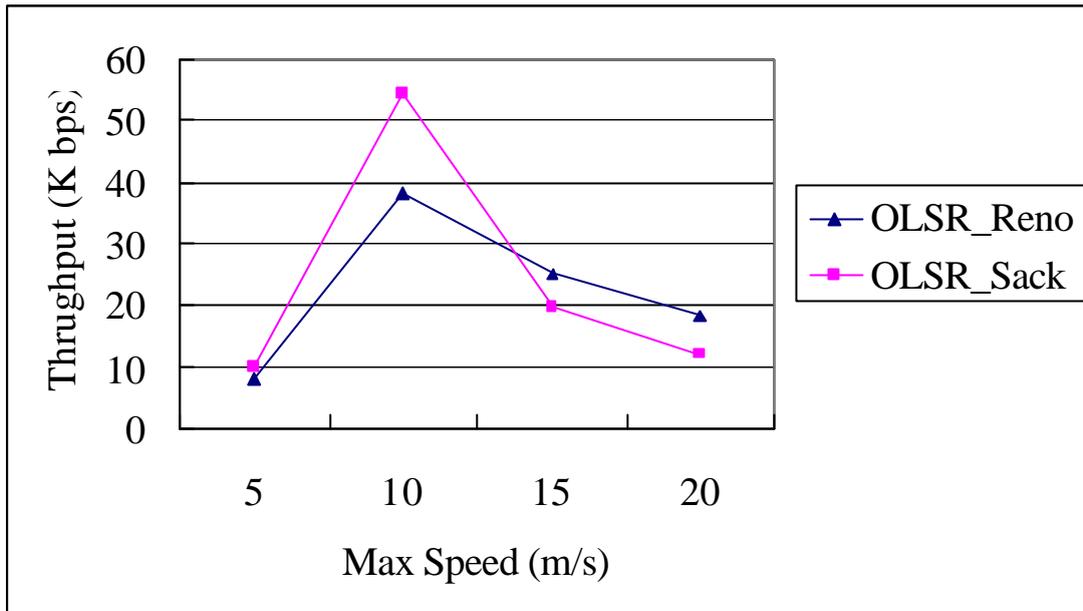


圖4-20 Reno vs. Sack在OLSR下的Throughput

在圖4-19與4-20，以Reno跟Sack混合的情況下，狀況與Reno和NewReno混合情況類似。同樣當節點最大移動速率超過每秒15公尺時，Sack的效能都比Reno來的低。也一樣發現到在高速移動的環境下，Sack會常常因為要重送大量遺失的封包，而這些封包又無法順利重送完畢，幾乎都得等到Timeout後才能重新啟動，造成效能降低。

#### 4.2.4 Reno vs. Vegas

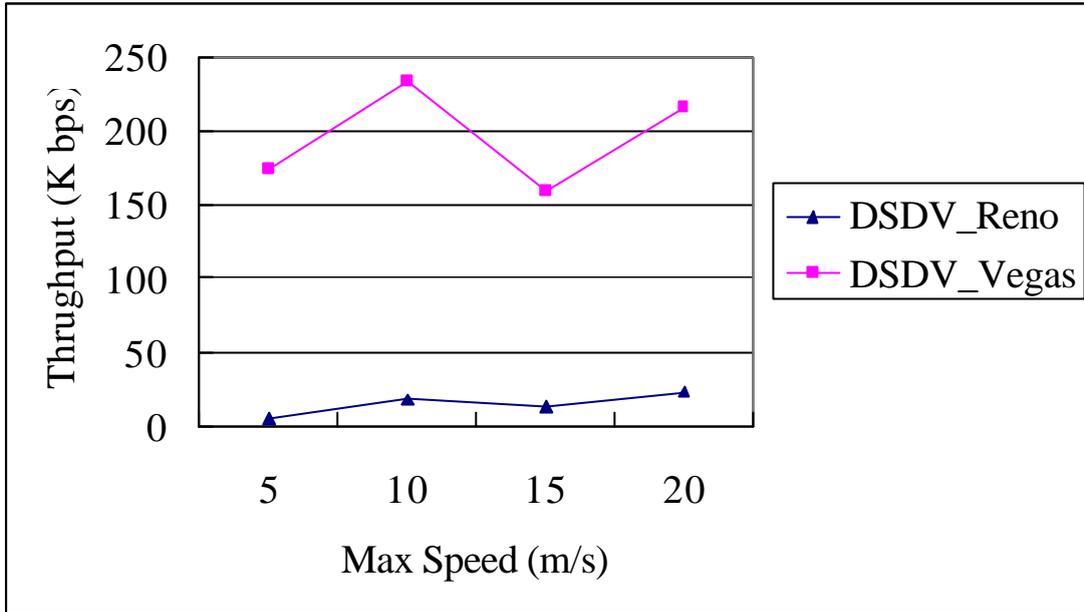


圖4-21 Reno vs. Vegas在DSDV下的Throughput

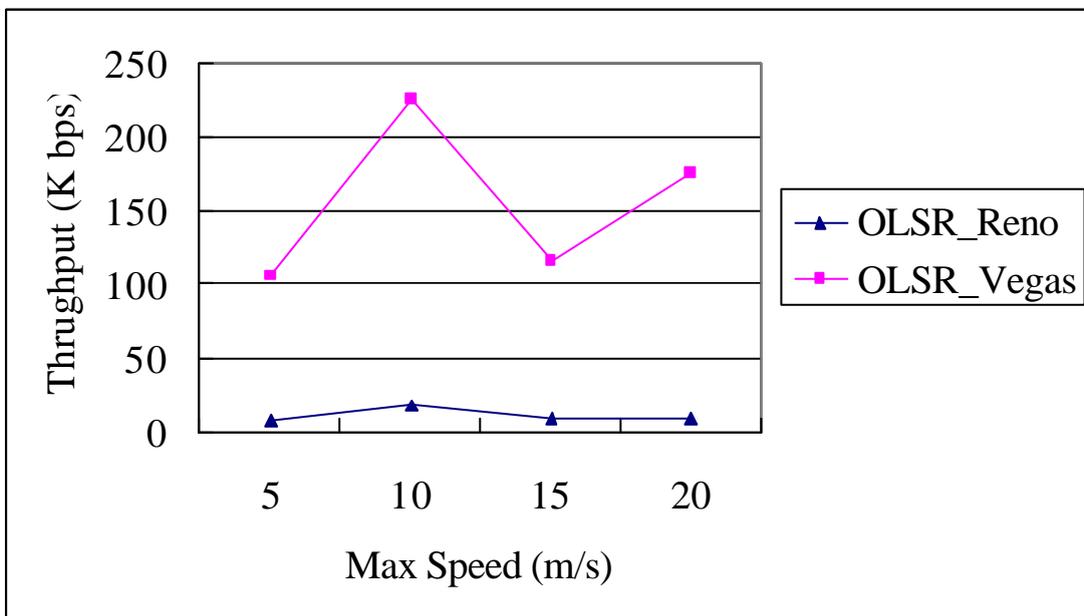


圖4-22 Reno vs. Vegas在OLSR下的Throughput

因為 Vegas 使用比較保守的壅塞控制機制，所以當 Vegas 發覺網路開始壅塞時，便會降低自己的傳送速率，不與 Reno 或路由協定的控制訊息搶頻寬而造成等待。當網路有比較多的頻寬後才開始加速封包傳送，因此在圖 4-21 與 4-22 中可以發現 Vegas Throughput 表現是四種混合環境中最好的。

綜觀以上幾個實驗，其實可以發覺在混合的環境底下，Reno 的 Throughput 相較於其他版本的 Throughput 是比較差的，這是由於 Reno 使用比較侵略性的壅塞控制機制，會一直傳送封包直到網路發生壅塞，而傳送越多的封包，發生碰撞的機率也就越高，尤其是在無線的環境中，封包碰撞的機率本來就比有線環境下高出許多，而碰撞越多，重新等待發送的時間也就越長，也造成 Reno 在實驗中 Throughput 都比其他版本來的低。

對於 TCP 效能的影響，最大的還是路由協定。路由協定維護路徑所佔用的網路頻寬，會影響到資料傳輸；回復路徑的速度若是夠快，在封包 Timeout 前就回復成功，則不會觸發 TCP 的壅塞控制，也不會嚴重影響效能。再來是移動速率，速率大體來說，的確是越快效能越差，但也並不是越慢效能就越好。若是移動速率太慢甚至趨近於不動，長距離的傳輸較不容易成功。最後則是 TCP 的版本，除了 Vegas 之外，其餘版本都是要等到封包 Timeout 或收到三個重複的 Ack，才知

道網路壅塞並開始降低傳送速率， Vegas則藉由測量RTT來判斷網路是否壅塞，不會在網路頻寬不足時還送出大量封包，讓這些封包發生碰撞遺失而降低效能。表4-1將各版本在不同的路由協定下，所呈現的效能及封包遺失多寡做個比較，以及混合環境中，與Reno的效能比較。

表4-1 各種TCP版本在不同路由協定下的效能比較表

		效能	封包遺失量	混合環境下與 Reno 的效能比較
Reno	DSDV	較差	較多	×
	OLSR	較好	較少	×
Tahoe	DSDV	較差	較多	較好
	OLSR	較好	較少	較好
NewReno	DSDV	較差	較多	較差
	OLSR	較好	較少	較差
Sack	DSDV	較差	較多	較差
	OLSR	較好	較少	較差
Vegas	DSDV	較差	較少	較好
	OLSR	較好	較多	較好

## 第五章 結論與建議

本篇論文探討了在 Ad Hoc Network 上會影響 TCP 效能的因素，並針對其中三種因素：路由協定、節點移動速率與 TCP 版本，設計模擬實驗以觀察這三種因素對於效能的影響程度。由實驗可發現，TCP 的效能在 Ad Hoc 環境之中，被 Ad Hoc 的動態拓樸影響非常大，同時也可證明，除了 Vegas 之外，其他版本差異並不會十分顯著，而是在於路由協定的選擇。由路由協定所產生的控制訊息，會造成的網路壅塞，因而觸發了 TCP 的壅塞控制機制。換句話說，不同的路由協定在建立路徑階段的方式會影響 TCP 的壅塞控制。所以說，如果建立路徑的時間可以小於 RTO (Recovery Time Objectives)，那麼 TCP 就不會產生封包遺失的情況。反之，若超過 RTO，則 TCP 便認定封包遺失，那麼就會觸發 TCP 的壅塞控制機制。

既然除了 Vegas，其他版本在 Ad Hoc Network 底下，效能差異並不會十分顯著，未來設計軟體時可以 Vegas 為主，並對於耗電量繼續研究，畢竟行動裝置的電量是設計資料傳輸時的一大限制。也由於在 Ad Hoc Network 中，有隱藏節點 (Hidden Terminal) 的問題，若是加大網路規模，讓傳輸的路徑必須透過更多中間節點來作傳遞時，是否會顯現出隱藏節點所帶來的影響，也是未來研究的一個方向。

## 參考文獻

- [1] M. Allman, V. Paxson, and W. Stevens, “TCP Congestion Control”, Request for Comments, RFC 2581, Network Working Group, Internet Engineering Task Force, Apr. 1999.
- [2] Josh Broch, David A. Maltz, David B. Johnson, and Yih-Chun Hu, “The dynamic source routing protocol for mobile Ad hoc networks”, <http://www.cs.cmu.edu/~dmaltz/internet-drafts/draft-ietf-manet-dsr-09.txt>, April 2003.
- [3] T. Clausen and P. Jacquet, “Optimized Link State Routing Protocol (OLSR)”, RFC3626, October 2003.
- [4] Kim Dongkyun, Cano Juan-Carlos, Manzoni P. Toh C-K., “A comparison of the performance of TCP-Reno and TCP-Vegas over MANETs”, Wireless Communication Systems, 2006. ISWCS '06. 3rd International Symposium on, pp. 495-499, September 2006.
- [5] T. D. Dyer and R. V. Boppana, “A comparison of TCP performance over three routing protocols for mobile ad hoc networks”, Proc. Of ACM Mobihoc, pp. 56-66, October 2001.

- [6] Ahmad Al Hanbali, Eitan Altman, and Philippe Nain, “A survey of TCP over ad hoc networks”, *Communications Surveys & Tutorials*, IEEE, pp. 22-36, Third Quarter 2005.
- [7] D. B. Johnson and D. A. Maltz, “Dynamic Source Routing in Ad-Hoc Wireless Networks”, *Mobile Computing*, T. Imielinski and H. Korth ed. Kluwer Academic Publishers, pp. 153-181, 1996.
- [8] James F. Kurose, Keith W. Ross, “Computer Networking: A Top-Down Approach Featuring the Internet”, Addison Wesley, 2004.
- [9] S. Kumar, V. S. Raghavan and J. Deng, “Medium Access Control Protocols for Ad-Hoc Wireless Networks: A Survey”, *Elsevier Ad-Hoc Networks Journal*, Vol. 4(3), pp. 326-358, May 2006.
- [10] Ka-Cheong Leung Li, V.O.K., “Transmission control protocol (TCP) in wireless networks: issues, approaches, and challenges”, *Communications Surveys & Tutorials*, IEEE, Vol. 8(4), pp. 64-79, 2006
- [11] Network Simulator [Online]. Available:  
<http://www.isi.edu/nsnam/ns/>.

- [12] Perkins, Charles E. and Bhagwat, Pravin, "Highly Dynamic Destination-Sequenced Distance-Vector Routing (DSDV) for Mobile Computers", ACM Computer Communication, pp.234-244, 1994.
- [13] C. E. Perkins and E. M. Royer, "Ad Hoc On Demand Distance Vector (AODV) Routing", IETF Internet Draft, draft-ietf-manet-aodv-02.txt, November 1998.
- [14] E. M. Royer and Chai-Keong Toh, "A Review of Current Routing Protocols for Ad-Hoc Mobile Wireless Networks", IEEE Personal Communications, Vol. 62, April 1999, pp. 46 -55.
- [15] Seddik-Ghaleb, A. Ghamri-Doudane, Y. Senouci, S.-M., "Effect of Ad Hoc Routing Protocols on TCP Performance within MABETs", Sensor and Ad Hoc Communications and Networks, 2006. SECON '06. 2006 3rd Annual IEEE Communications Society on, pp.866-873, September 2006.
- [16] Toh, C.-K., "Ad-Hoc Mobile Wireless Networks: protocols and Systems", Prentice Hall PTR, 2002.
- [17] S. Xu, T. Saadawi and M. Lee, "Comparison of TCP Reno and Vegas in wireless mobile ad hoc networks", IEEE LCN 2000.