國立交通大學

資訊科學與工程研究所

碩 士 論 文

SCTP 上的延遲取向無縫式轉換之研究

The Study of Seamless Delay-Based Handoff over SCTP

研 究 生：蔣承翰

指導教授：吳毅成　教授

中 華 民 國　九 十 六　年 六 月

SCTP 上的延遲取向無縫式轉換之研究

The Study of Seamless Delay-Based Handoff over SCTP

研 究 生：蔣承翰　　Student：Cheng-Han Chiang

指導教授：吳毅成　　Advisor：Yi-Cheng Wu

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 2007

Hsinchu, Taiwan, Republic of China

中華民國九十六年六月

SCTP 上的延遲取向無縫式轉換之研究

研究生：蔣承翰　　指導教授：吳毅成

國立交通大學 資訊科學與工程研究所

摘要

隨著無線網路通訊技術的進步，越來越多的行動裝置擁有兩個以上的網路介面，如 WLAN, 3G 和 GPRS。行動裝置可以透過這些網路來瀏覽網頁或者玩網路遊戲。這些網路介面各有其不同的特性，譬如頻寬(bandwidth)以及反應時間(response time)等；針對網路遊戲應用，玩家對於網路的反應時間的要求相當高，因此如何利用可用的網路介面以獲取快速的反應時間是相當重要的。SCTP multi-homing 的特性有利於在這些網路之間的轉換，然而其轉換機制並未針對網路的反應時間做最佳化。為了達到這個目的，本論文提出一套改良 SCTP 的方法，達成在不同網路間無縫轉換以獲取快速的反應時間。

基於 SCTP 的特性，本論文的方法可以使用多個網路介面建立連線，並且主要有兩個特點：一、利用 SCTP heartbeat 測試的機制，定期測量網路的 RTT(Round Trip Time)，以即時取得網路反應時間的資訊。二、藉由 RTT 資訊來選擇最快速的網路傳輸資料，並且在網路速度不穩定而無法決定何者最快的時候，利用並傳(simulcast)的機制加以協助。實驗數據顯示，本論文提出的方法能有效改善 SCTP 的缺點，善加利用 multi-homing 的特性，讓網路連線有更快速的反應時間，有助於需要快速反應時間的應用。

The Study of Seamless Delay-Based Handoff over SCTP

Student: Cheng-Han Chiang Advisor: Yi-Cheng Wu

Institute of Computer Science and Engineering

National Chiao Tung University

Abstract

More and more devices have more than one network interface, such as WLAN, 3G , and GPRS. These interfaces own different bandwidth and response time. For network game applications, response time is a highly required factor of service quality. So how to get fast response time by all available network interfaces is a very important research. The multi-homing feature of SCTP can benefit handoff between interfaces, but the handoff mechanism is not optimized for response time. To solve this question, this thesis proposes a new design for SCTP.

Based on feature of SCTP, our design has two features: (1) By the heartbeat testing mechanism of SCTP, periodically get RTT information of network paths. (2) Use RTT information to choose the path with fastest response time. And, use simulcast on unstable condition. By the experimental results, our design could efficiently reduce the problem of SCTP to get fast response time. The technology could support the application that requires faster response time.

誌謝

首先我要感謝我的指導教授吳毅成博士，在我碩士就學期間給了我嚴格的要求與指導，讓我獲益良多，也才能順利將本篇論文完成。

也要感謝實驗室的各位學長、同學跟學弟們，在與你們的相處之中，一同研究討論、互相鼓勵，不僅增長我的知識，完善了本篇論文，更拓展我的視野，豐富了我的生活。更要感謝我生活週遭的人們，敬洋、俊彬、育嘉、宗漢、函儒、李銘和禕涵，謝謝你們將生活中各種精采與我分享，正是有你們一路相伴，才成就了現在的我。

最後當然也要感謝我的家人，一直給予我無限的支持，你們的愛護永遠是我最強大的後盾。

道不盡的感謝，要謝的人太多了，僅以本篇論文，獻給諸位。

目錄

表目錄

圖目錄

第一章、緒論

隨著行動裝置以及無線通訊技術的進步與發展，其應用越來越普遍，且越來越多的裝置同時擁有超過一個的網路介面。這些無線通訊技術有著各種不同的規格，各有其優缺點，如頻寬(bandwidth)或反應時間(response time)等等；當在這些不同的無線網路之間漫遊時，有些應用將會需要有技術來確保連線不需中斷，而特別針對網路遊戲應用，對反應時間的需求更是相當的高。因此希望可以結合各種無線技術的優點，以提供使用者一個能達成無縫式轉換並擁有快速反應時間的環境。以下會先對現在較普遍使用的無線通訊技術做簡單的比較，然後說明相關研究、研究目標與論文大綱。

1.1 WLAN、3G 與 GPPS 的比較

目前已經發展出很多種無線通訊技術，我們就最廣泛使用且適合 IP 網路的三種技術，WLAN、3G 與 GPRS 來做簡單的比較[9][20][21]。如下表：

Network	GPRS	3G	WLAN 802.11g
Coverage	Ubiquitous	Ubiquitous	Local(35~110m)
Bandwidth	Low (up to 177.6K bps)	Medium (up to 14.4M bps)	High (up to 54M bps)
Round Trip Time	Slow (1000-2000ms)	Medium (300-500ms)	Fast (5-40ms)
Cost of data services	High (0.002NT/packet)	High (0.0006NT/packet)	Low

表 1-1 WLAN, 3G 與 GPRS 的比較

根據表 1-1 的描述，WLAN 具有高頻寬、低反應時間的優點，不過涵蓋範圍較窄。對 3G 與 GPRS 來說，其涵蓋範圍極大，不過頻寬較低，反應時間高，其中 GPRS 在這兩方面的效能是最差的，而且 3G 跟 GPRS 都需要比較高的費用。

1.2 相關研究

關於無縫式轉換方法的相關研究相當多，其中包括在網路層(Network Layer)[2][13][14][16][22][23][26][27]、應用層(Application Layer) [6][17][25]、及傳輸層(Transport Layer) [1][4][5][7][8][10][11][18][24]的實作方式。以下將針對與本論文所提方法之做法或目的上相近之實作方式加以介紹，並討論其優缺點。

1.2.1 PSockets

PSockets[17]屬於應用層的實作，架構如圖 1-1，藉由建立多條 TCP 連線，切割封包然後同時使用這些連線來做傳輸，如果其中有一條連線中斷，則會交由另一條連線來做封包補齊的工作，他的實作方法是抽換底層的函式庫，取代 socket API 裡面的 send 與 receive，改成同時用多條 socket 來做收發。

圖 1-1 PSockets 架構圖

此方法的優點為因為是應用層的實作，所以很容易部署；但也因為在應用層上實作，所以他必須設計一套 buffering 與 acknowledgment 的機制，當某條連線斷線或是壅塞，才能做資料緩衝或是重傳，因為這些事情同樣的在傳輸層也會做一次，所以會導致很沒效率，且要設計此套機制，也是相當複雜的。

1.2.2 PTCP

PTCP[5]的架構如圖 1-2，這是在傳輸層(Transport layer)的協定。會為每條連線建立一個 TCP-virtual 以負責各條連線的 congestion control，其上由 PTCP 負責實際的資料緩衝以及資料傳送線路的分配。當有資料要傳送的時候，PTCP 會選出一條尚未壅塞的線路來傳送；並且在封包逾時的時候，能重新指定可用的線路重傳。

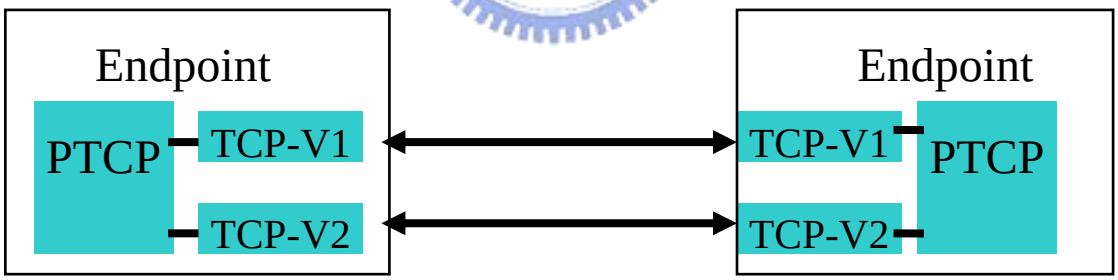

圖 1-2 PTCP 架構圖

此方法的優點是因為是傳輸層的實作，因此就避免了應用層重複實作 buffering 與 acknowledgement 的問題。不過在判斷線路斷線以及封包重傳的機制上，必須等待封包逾時，並不會很即時的作出反應。

1.2.3 SCTP

SCTP(Stream control transmission protocol)[18]是 IETF 的一個傳輸層協定,提供可靠的訊息導向資料傳輸[3][12]。本論文將會以 SCTP 為基礎技術,並在其上加以改良。因此下面會對這個協定作更詳細的介紹。

1.2.3.1 SCTP 簡介

SCTP 是與 TCP 屬於同一層級的通訊協定，位於 IP Network 之上，其最大的特性，就是支援傳輸資料的兩端點可擁有一個以上 IP 位址。其架構如圖 1-3

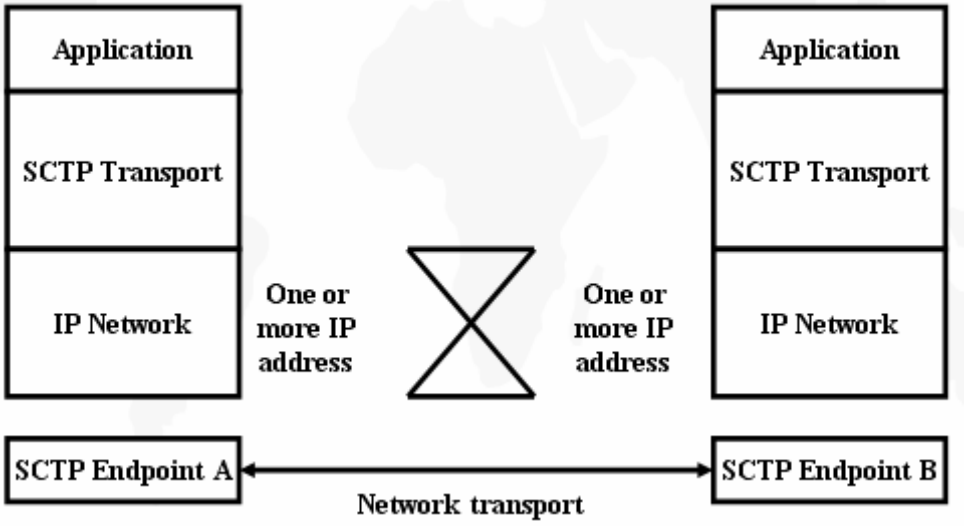

圖 1-3 SCTP 架構圖

SCTP 與 TCP 都必須建立連線才能進行傳輸，但 TCP 僅支援

reliable 與 ordered 的傳輸；SCTP 則支援 reliable、partial reliable、unreliable, ordered、partial ordered、及 unordered 等模式。SCTP 的端點可以擁有一個以上的 IP 的特色，稱之為 multi-homing。如圖 1-4，兩個端點都擁有一個以上的 IP 位址，當其中一個位置無法使用之後，不需要中斷連線仍然可以使用其他 IP 繼續傳送資料。

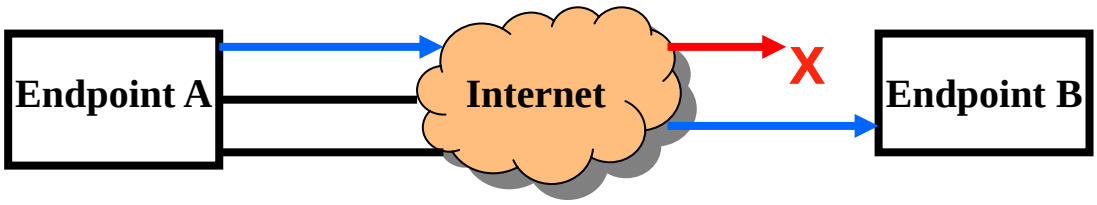

圖 1-4 SCTP multi-homing 示意圖

SCTP 的封包格式與資料封裝(bundle)格式如圖 1-5。每個封包擁有一個 SCTP Common Header 還有一個以上的 chunk，每個 chunk 擁有的各自的 chunk header 跟 data。Chunk 類型有：DATA、SACK、HB、HB Ack 等。SCTP 資料封裝的技術就是一個封包能夠同時包含多個 chunk，就能減低傳輸的次數及資料量。

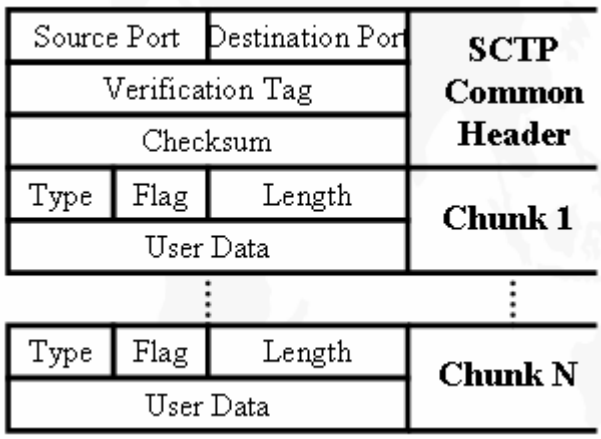

圖 1-5 SCTP 封包格式

SCTP 採用 Selective Acknowledgement (SACK)的方式。譬如圖 1-6 所示，當傳送封包 1 到 6 時而 3 號封包遺失時。對 TCP 基本的

acknowledgement 方式而言，當收到 2 號封包時，會回覆 Ack=3，表示他之後要接收 3 號封包，即使先接收到 4、5、6 號封包，也只能丟掉，就必須重傳 3、4、5、6 這 4 個封包；對 SACK 方式而言，當收到 2 號封包時，會回覆 Ack=3，表示他之後要接收 3 號封包，但如果先收到 4、5、6 號封包，則會回覆 Ack=3,(4,6)，表示 4 到 6 號也都有接收到。因此只需要重傳 3 號封包。這樣的模式便能改善重傳的效率。

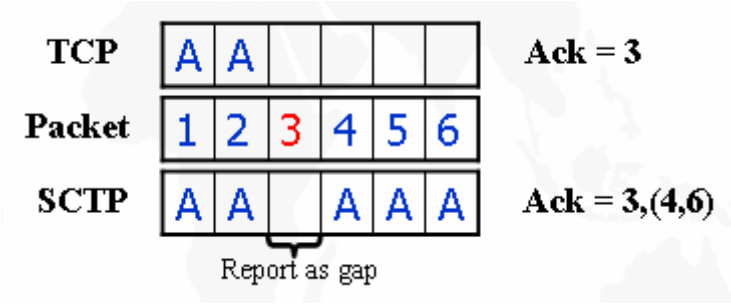

圖 1-6 SCTP SACK 與 TCP Ack 的比較

而 SCTP 也利用 multi-homing 的特性，當必須重傳時即代表原來使用的路徑有問題，便可以選擇不同的路徑來傳送

對於所有沒有在傳送資料的路徑，會定期(30 秒)送 heartbeat(HB) chunk，來測試此路徑是否還有效；經過五次連續的封包(DATA 或 HB)遺失後，就會認定此路徑失效。

1.2.3.2 SCTP 分析討論

由於 SCTP multi-homing 的特性，可同時使用多個網路介面；SCTP 內建 heartbeat testing 的方法，提供了一個可用以偵測網路連線狀態的手段。不過 SCTP 對於連線狀態的探知是很緩慢的：通過計算資料重傳次數是否已到達某個值，所以需要經過幾次的資料遺失與重

傳，才能知道連線是否已中斷；又或者是通過每 30 秒一次的 heartbeat 來測量，也是相當的緩慢，因此並達不到快速反應時間的需求。

1.2.4 改良之 SCTP

基於 SCTP 的技術，目前也已經有相當多的改良方法，以下介紹數種相關的改良方法。

1.2.4.1 多路徑傳送演算法

多路徑傳送研演算(Multi-path Transmission Algorithm)[7]提出一個運用多條線路的演算法：利用 SCTP heartbeat 的機制，頻繁的傳送 heartbeat 以測試路徑是否穩定。當路徑不穩定時，則會多條線路同時傳送同樣的資料(並傳)，以確保資料的傳，如圖 1-7 所示。

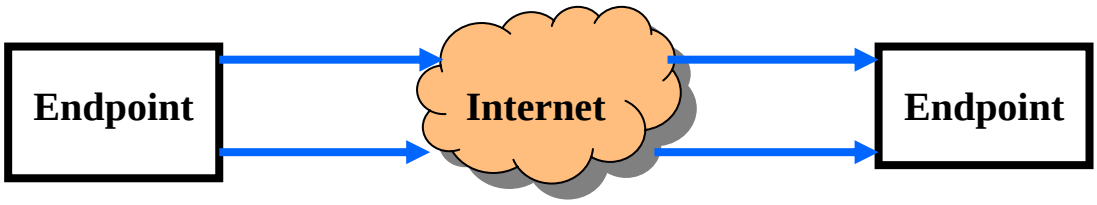

圖 1-7 多路徑傳送演算法運作示意圖

這方法由於頻繁的測試路徑的穩定度，因此增進了對路徑狀態的反應速度；並且在不穩定狀態下使用並傳，對網路的 throughput 有所增進。不過此方法針對 unreliable transmission，所以並未考慮到路徑的反應時間，而且挑選傳送路徑的方式也並不會挑選最低反應時間的

路徑。

1.2.4.2 訊號強度偵測方法

訊號強度偵測方法[24]針對 WLAN 與 GPRS 兩種網路之間的切換，利用 WLAN 高頻寬、低反應時間的特性，選擇以 WLAN 為優先使用的主要傳送路徑。並且偵測 WLAN 的訊號強度，以其協助判斷 WLAN 的穩定程度。當 WLAN 的傳送發生封包遺失或是訊號強度變弱時，就會利用 GPRS 一起並傳，以降低封包再度遺失的機率。運作方式如圖 1-8。

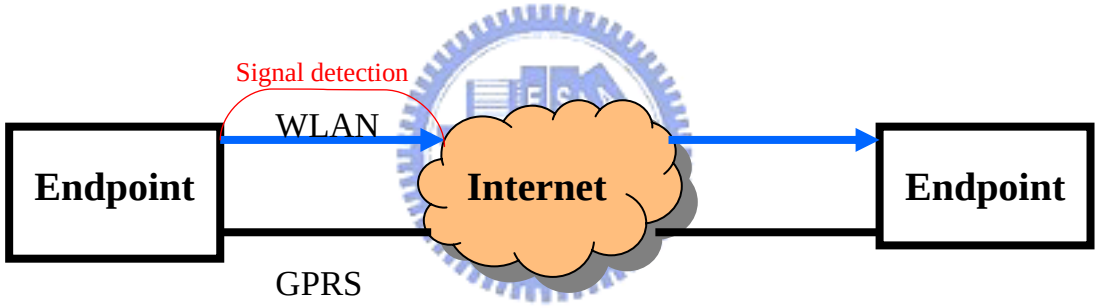

圖 1-8 訊號強度偵測運作示意圖

這個方法以偵測訊號強度的方式來加快對路徑狀態的反應速度；並配合在主要路徑不穩定時使用並傳的機制，可以增進在整體傳輸在反應時間方面的效能。不過此研究完全是針對 WLAN 與 GPRS 的特性來設計的，而且著重在訊號強度的測試。當傳輸路徑上所發生的問題不能由訊號強度來判斷時，僅能依靠封包遺失與否來做路徑的轉換，這樣的方式並不能確保傳輸能得到最低的反應時間。

1.2.4.3 延遲取向轉換

延遲取向轉換(Delay-Based Handoff)[8]希望能選出反應時間最低的路徑來傳送資料，如圖 1-9 所示。因此去測量各個路徑的 RTT，再加以計算各個路徑的 SRTT(Smoothed Round Trip Time)。

$$SRTT' = SRTT*(1-\alpha)+ RTT*\alpha,\ \alpha=1/8$$

SRTT 可視為是路徑 RTT 平均值的一種計算方式，依此值決定路徑反應時間的快慢，並選擇 SRTT 最低的路徑來傳送資料，來達到低反應時間的要求。

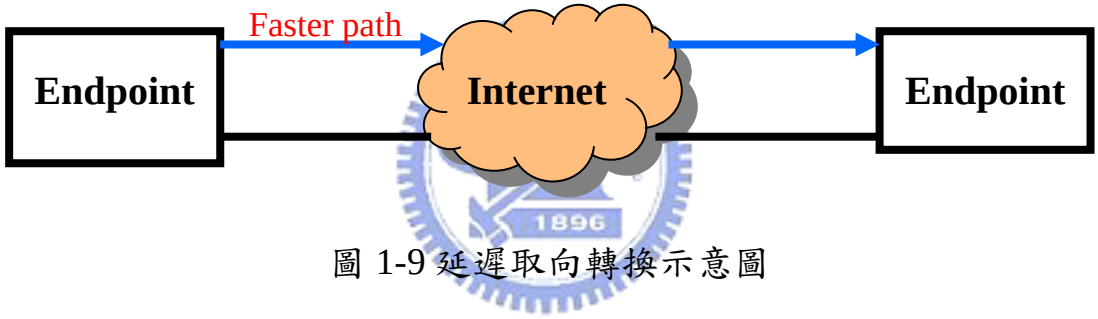

圖 1-9 延遲取向轉換示意圖

這個方式基本上是想要選擇最低反應時間的路徑來傳送，但是 SRTT 是一個平均值的計算方式，並不能完全即時的反應目前實際的路徑反應時間。

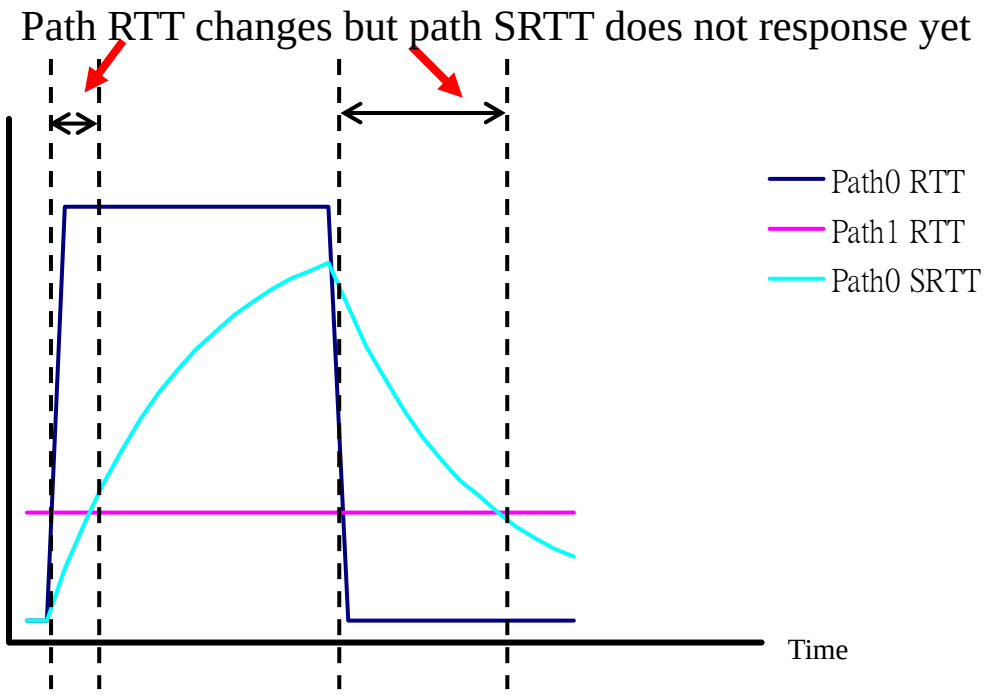

圖 1-10 SRTT 曲線示意圖

如圖 1-10 所示，Path1 是一條反應時間相當穩定的路徑，其 SRTT 因而也持平在同一個值；而 Path0 的反應時間由低變高再落回低值。可以觀察到，即使 Path0 的反應時間一瞬間變高或一瞬間變低，其 SRTT 值的變化將必須經過一段時間才會趨近實際的反應時間。因而造成路徑的轉換判斷會不夠即時，無法完全達到選擇最低反應時間路徑的需求。

1.3 研究目標

本論文是為了提供使用者一個能達成無縫式轉換並擁有快速反應時間的環境。我們以 SCTP 為基礎，並設法增進其反應時間方面的效能。本論文的方法利用 SCTP multi-homing 的特色，使用多個網路

介面建立連線，搭配 SCTP heartbeat 測試的機制，定期測量網路的 RTT(Round Trip Time)，以即時取得各個網路反應時間的資訊，並藉此資訊來選擇最快速的網路傳輸資料。且在網路速度不穩定而無法決定何者最快的時候，利用並傳(simulcast)的機制加以協助，以取得快速的反應時間。

1.4 論文大綱

本論文之其他章節架構如下：

1. 第二章說明本論文詳細的設計，包括取得路徑 RTT 資訊的方法、數種可能的判斷網路快慢的方式和藉以選擇傳送路徑的方法。
2. 第三章說明實驗的環境與流程，提出實驗數據並分析其結果。
3. 最後，第四章為結論和未來展望。

第二章、系統設計

為了提供使用者一個能達成無縫式轉換並擁有快速反應時間的環境，利用 SCTP multi-homing 的特色，基於延遲取向轉換的原則，來選擇最快速的網路傳輸資料。這章節將敘述此方法的設計，包括測量網路路徑 RTT 的方式、可能的延遲取向轉換路徑選擇方法與其問題並提出更合用的方式、以及針對資料封包逾時重傳路徑選擇的特殊處理。

2.1 路徑 RTT 測試

一般而言，路徑的 RTT 就代表著路徑反應時間的長短；因此要求快速的反應時間，路徑的 RTT 資訊是相當重要的依據。所以如何測量路徑 RTT 將是本論文的第一個重點

。在原本的 SCTP 中，路徑的 RTT 測量只透過封包的 ACK 來計算，其中包括一般在傳送的資料封包和不傳送資料的路徑上的 heartbeat 封包。這樣的方法使得每條路徑上的 RTT 測量並不定時、不同步。而且當封包沒有 ACK 的時候，不論是當封包遺失或者 ACK 封包遺失，RTT 測量也就不會運作。因此在需要以路徑 RTT 為主要資訊來轉換路徑的延遲取向轉換中，一個穩定的 RTT 測量方法是必要的。因此我們利用 SCTP 的 heartbeat 機制，定期在每一條路徑上傳送一個 heartbeat 封包，以此即時的取得所有路徑的 RTT 資訊。

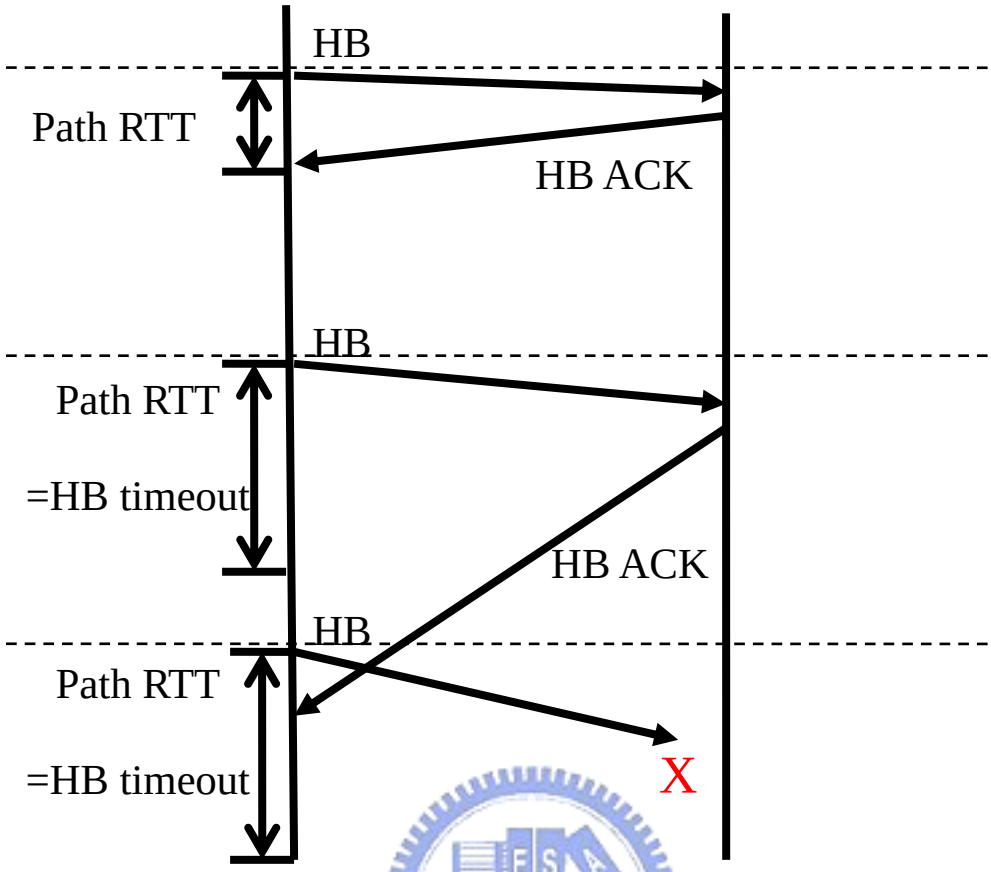

圖 2-1 heartbeat 測試方法

以 heartbeat 測量 RTT 的機制如圖 2-1 所示。虛線表示固定的時間間隔，每個時間間隔(譬如 1 秒)就傳送一個 heartbeat 作為 RTT 測試。一般是以 heartbeat 送出到 ACK 回來之間的時間，作為路徑的 RTT，如圖 2-1 的第一個區間。但有兩種特殊情況需要處理：heartbeat ACK 時間太長或 heartbeat 並無回應，這兩種情況通常發生在路徑狀況不良的時候，如圖 2-1 的第二及第三區間。由於希望測量到的 RTT 資訊具有足夠的即時性，才能即時反應路徑的狀態，所以希望每一個測試區間都要取得一次 RTT 的資訊；而無論是 ACK 時間太長或是 heartbeat 無回應都將造成其區間內無法計算 RTT。因此必須對 heartbeat 設定一個固定的 timeout 時間，此時間小於或等於每個測試區間的長度，當在此 timeout 時間內得不到 ACK 時，就以 timeout 時間作為路徑的 RTT。這樣的做法可以確保每一個測試區間之中都能取得一次 RTT 的資訊，而且也可以用 timeout 的時間值來反應路徑處

於不良好的狀態之中。

2.2 延遲取向轉換之路徑選擇方式

在取得各個路徑的 RTT 之後，就以此作為主要資訊來選擇要使用的路徑。以下先敘述一些可能的選擇方式，並提出其存在的問題，從而引出我們的設計。

2.2.1 RTT 方法

RTT 方法是最基本的路徑選擇方法。既然路徑的反應時間一般就以 RTT 來表示，所以要求較快的反應時間，就選擇 RTT 較低的路徑。

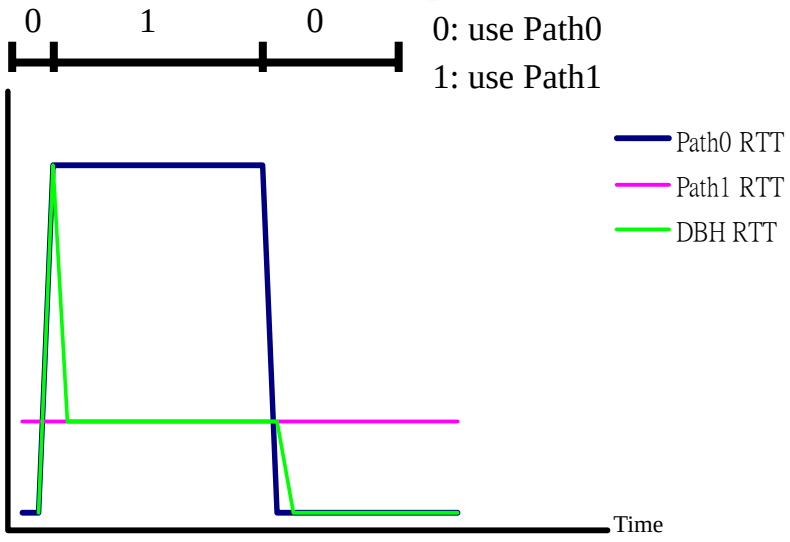

圖 2-2 RTT 方法運作示意圖

以圖 2-2 為例，Path1 RTT 穩定不變，Path0 RTT 先變高，持續一段時間之後變低。當使用 RTT 方法選擇路徑時，在發現 Path0 RTT 有所改變之後，就會馬上轉換到 RTT 較低的路徑。這樣的路徑選擇方法相當直觀，對於網路反應速度的變化也反應得相當迅速，但實際上卻容易發生誤判的情形。

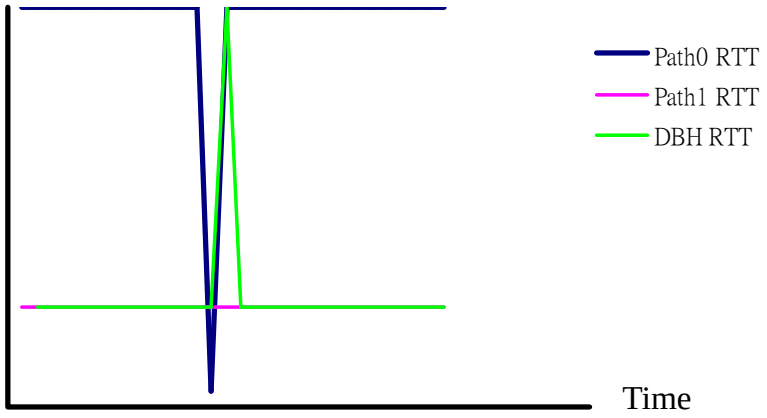

圖 2-3 RTT 方法誤判範例

如圖 2-3 的狀況，Path0 原本處於比較慢的狀態，所以選用 Path1 來傳輸資料。當偵測到 Path0 RTT 下降了，就會轉換到 Path0 來傳輸，但是此 Path0 RTT 下降僅在一瞬間，之後依然回復到原本緩慢的狀態，所以這次的轉換反而造成資料傳輸反應時間的上升。總結來講，RTT 方法的路徑轉換會在偵測到路徑 RTT 有所改變之後才會反應，但此改變可能只是一個暫時的狀態，並不完全足以反應這條路徑整體的狀況，因而可能發生誤判的情形。所以需要一種方法，用以表示路徑整體的情況，才能做到準確的路徑轉換。

2.2.2 SRTT 方法

2.3.3 節中所提到的延遲取向轉換方法，便是基於 RTT 方法的缺

陷，提出一個用以表示路徑整體反應時間的方法，以 SRTT 為選擇路徑的主要依據：

$$SRTT' = SRTT*(1-\alpha)+ RTT*\alpha,\ \alpha=1/8$$

SRTT 可視為一種路徑 RTT 平均值的計算方式。若以 SRTT 為基準，選擇 SRTT 較低的路徑，則可以避免一些 RTT 發生誤判的情形。以與圖 2-3 同樣的 RTT 變化模式，SRTT 方法的運作如圖 2-4：

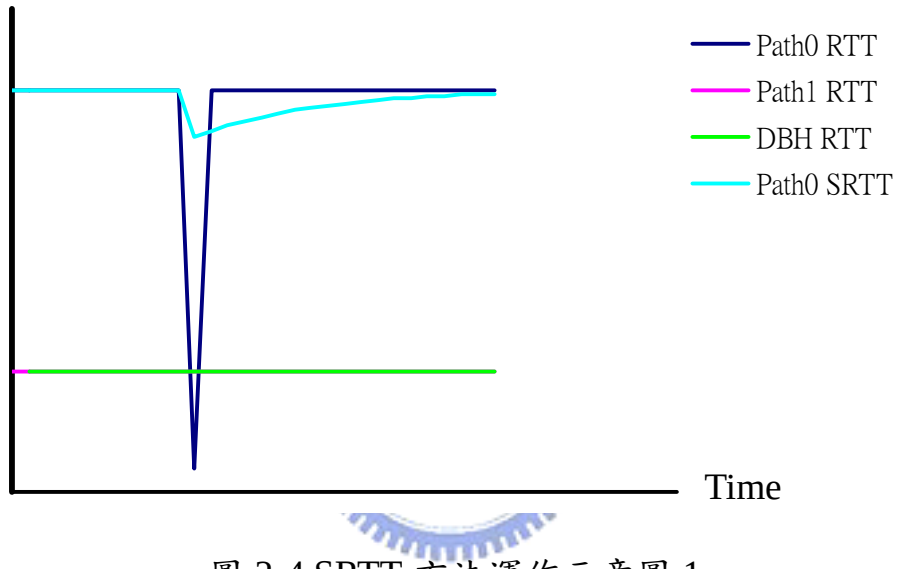

圖 2-4 SRTT 方法運作示意圖 1

可以看見，當 Path0 RTT 變化之時，Path0 SRTT 並未隨之大幅改變，因此就不會因為僅一瞬間的 RTT 變化而造成路徑的轉換。但也如同 2.3.3.1 中所分析，SRTT 並不能完全即時的反應路徑 RTT。以與圖 2-2 同樣的 RTT 變化模式，SRTT 方法運作如圖 2-5：

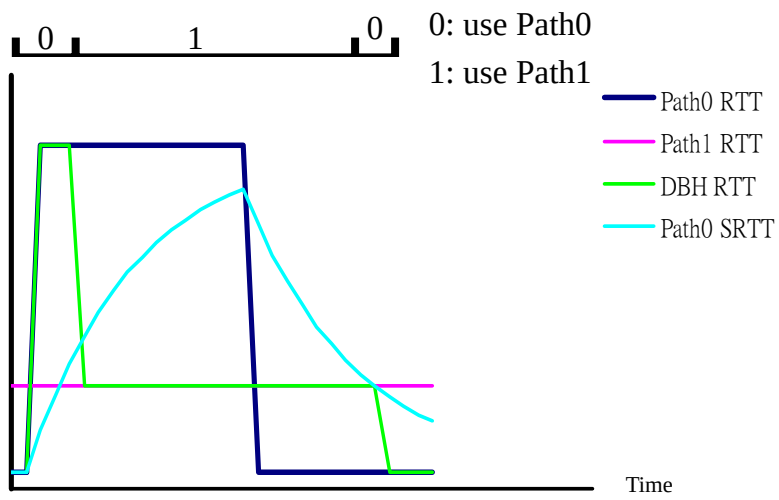

圖 2-5 SRTT 方法運作示意圖 2

由圖可見，由於 SRTT 變化幅度較緩，路徑 RTT 的改變要過一段時間才會由 SRTT 反應出來，因此就會造成路徑轉換時機的延遲，無法即時利用到最快速反應時間的路徑。基本上，這是利用平均值計算來選擇路徑都會遇到的問題，因為平均值並不能完全表現出當前的狀態。

2.2.3 RTT thresholding 方法

根據 3.2.1 及 3.2.2 的敘述，可以了解到需要有一種路徑轉換方式，既能夠即時的對路徑 RTT 的變化作出反應，又要能避免暫時性的變化可能造成的誤判。基於這樣的需求，本論文提出 RTT thresholding 的方式，主要分為路徑狀態表示法及路徑選擇原則兩部分。

2.2.3.1 路徑狀態表示法

在 RTT 方法跟 SRTT 方法中，都只使用了一個數值，藉以表現路徑的狀態，並依此選擇路徑；然而從結果論，其所使用的數值僅能表現一種暫態或是常態，以致於選擇路徑時有所缺失。因此本論文採用一個〝區間〞的方式，以更合理的表示路徑的狀態。所使用的方式稱之為 RTT thresholding，藉由紀錄一段時間的路徑 RTT 資訊，從中界定一個區間，這個區間以一個高值和一個低值來表現，以表示 RTT 可能的變化範圍。本論文中，這個高值和低值是由最新的數次 RTT 資訊中選取其最高值(Recently Highest RTT, 簡稱 RH)和最低值(Recently Lowest RTT, 簡稱 RL)來決定。

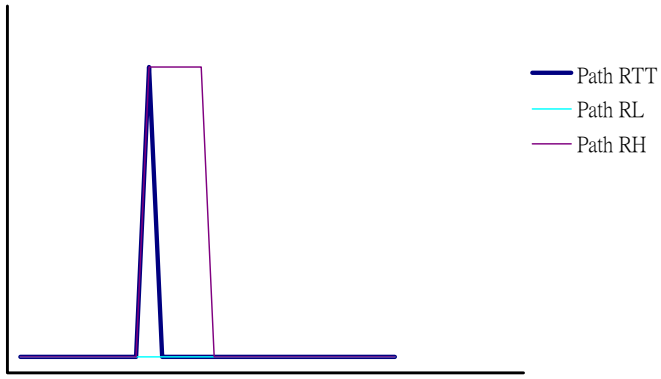

圖 2-6 RH/RL 區間示意圖 1

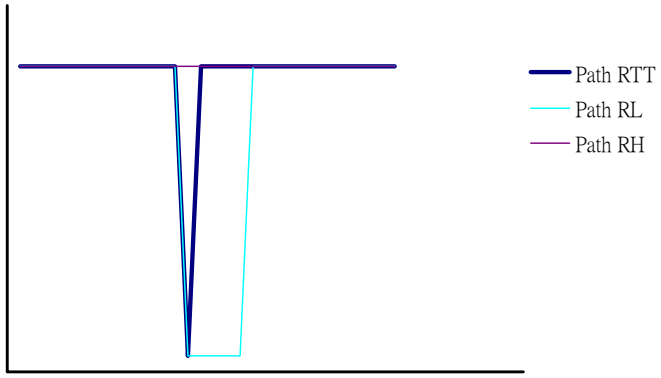

圖 2-7 RH/RL 區間示意圖 2

圖 2-6 和圖 2-7 表示 RL/RH 區間的運作方式：圖 2-6 裡路徑 RTT 平常保持低值，只有一次暫時性的高起，則選取最新幾次 RTT 內最低值的 RL 會一直維持著低值，而選取最高值的 RH 則在高起發生後的一段時間內表現此高值，這段時間的長度即等於用來挑選 RL/RH 的 RTT 紀錄次數所經歷的時間。例如說 RL/RH 是由最新 5 次 RTT 中所選出，而每秒會測試一次 RTT，則 RH 保持高值的時間大概為 5 秒，此時間會受實際測量到 RTT 的時間而有所變化。圖 2-7 裡的路徑 RTT 平常保持高值而只有一次暫時性的低落，則 RH 會一直維持高值，RL 在低落發生後的一段時間內表現此低值。藉由這樣的設計，既有一個值可以表現路徑的常態，也有一個值可以表現路徑即時的變化，才能更完整的表現路徑的狀態。另外，RL/RH 區間也很適合表示路徑處於不穩定狀態，如圖 2-8。

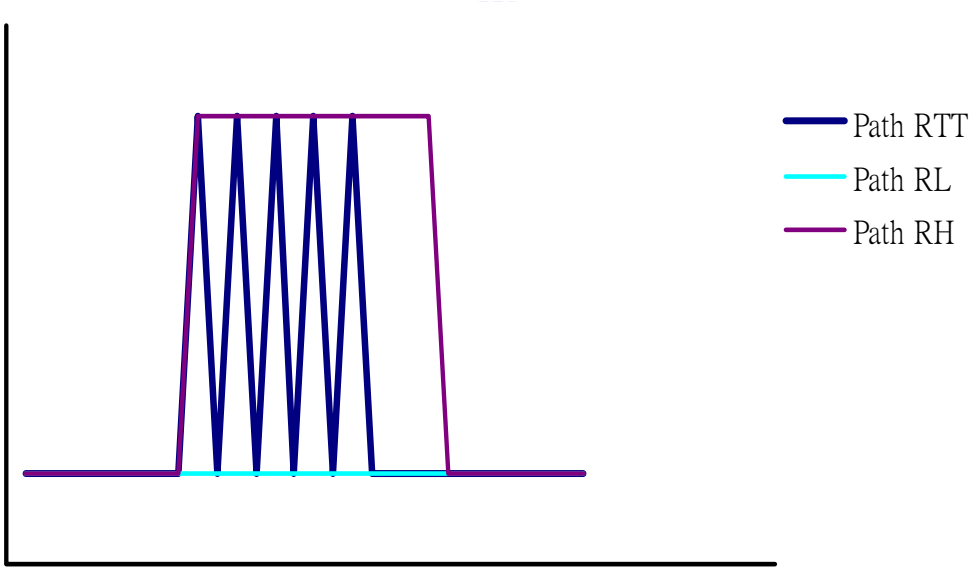

圖 2-8 以 RL/RH 表現路徑的不穩定狀態

2.2.3.2 路徑選擇原則

在用 RL/RH 區間表示出路徑的 RTT 可能的變化範圍之後，便以

此作路徑的選擇的依據：

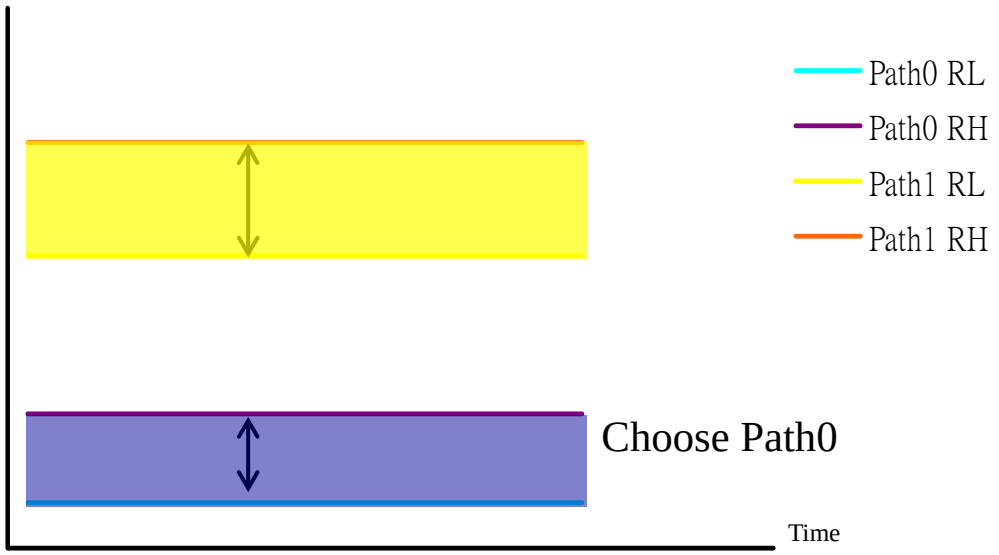

圖 2-9 RTT thresholding 選擇單一路徑示意圖

當 Path0 和 Path1 的 RL/RH 區間分布如圖 2-9 所示，可以看出 Path0 的區間完全低於 Path1 的區間，這就代表了 Path0 的 RTT 儘管可能有變動，但都會比 Path1 RTT 還低，因此就選擇只使用 Path0 來傳輸，就可以得到最快的反應時間。

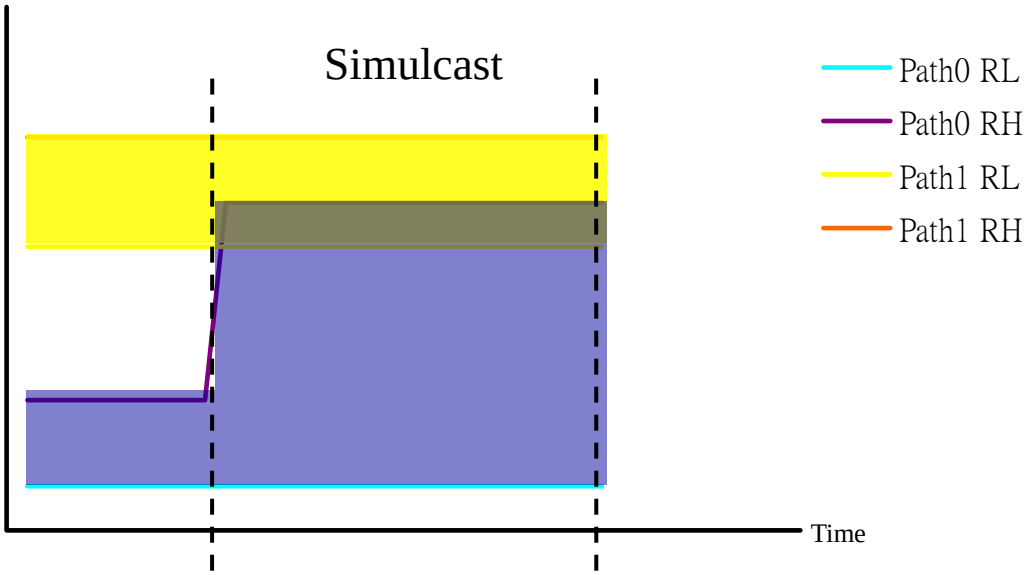

圖 2-10 RTT thresholding選擇並傳示意圖

當 Path0 和 Path1 的 RL/RH 區間分布如圖 2-10 所示，有一段時間內 Path0 和 Path1 的區間是互相重疊的，這代表著 Path0 的 RTT 有可能高於 Path1 的 RTT。在這種情況下，並無法直接斷定使用哪一條路徑可以得到最快的反應時間，所以此時就採用並傳(simulcast)的方式，兩條路徑同時傳送同樣的資料。這樣一來無論是哪一條路徑的反應時間較低，都能使用到最快速的路徑。圖 2-11 提供一個 RTT thresholding 方法選擇路徑的範例。以和圖 2-2 和圖 2-5 同樣的 RTT 變化模式，在偵測到 Path0 RTT 有所變化之後，由於 Path0 的 RL/RH 區間和 Path1 重疊，因此就會進入並傳的模式；經過一段時間，當確認了 Path0 的 RTT 並非暫時性變化之後，就停止並傳而選用 RTT 較低的路徑。RTT thresholding 便是基於這樣的原則來選擇路徑以取得最快速的反應時間。

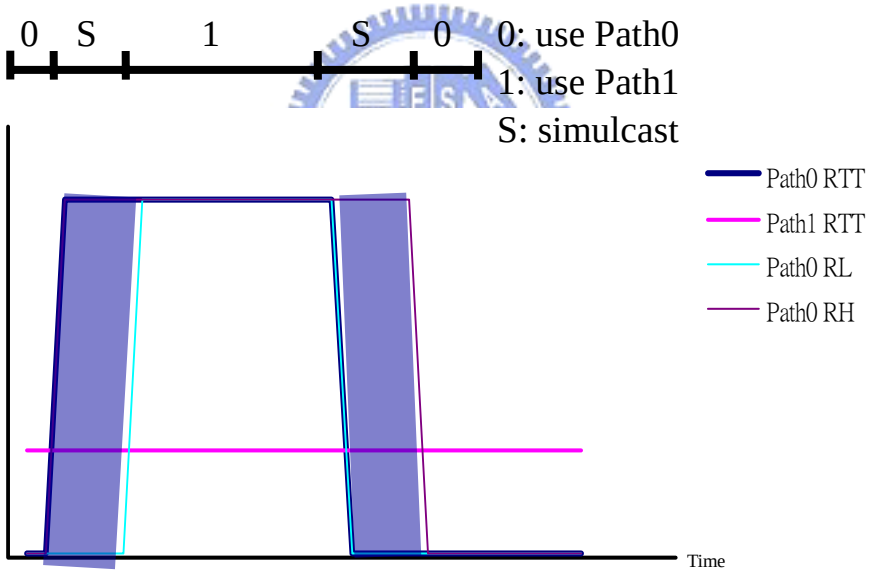

圖 2-11 RTT thresholding 運作模式示意圖

2.3 封包逾時後的重傳路徑選擇方式

除了按照延遲取向轉換的方法來進行路徑之間的轉換，還有一種情況也需要選擇資料傳輸所使用的路徑，那就是在發生了資料封包逾時(timeout)之後。在原本的 SCTP 中，當資料封包逾時，則視為封包遺失需要重傳；而這也代表著目前所使用的路徑有不良狀況，因此會選擇另一條路徑來進行重傳。在本論文所敘述的方法之中，也將會保留這個性質，並且和延遲取向轉換選擇路徑的方法相互結合。

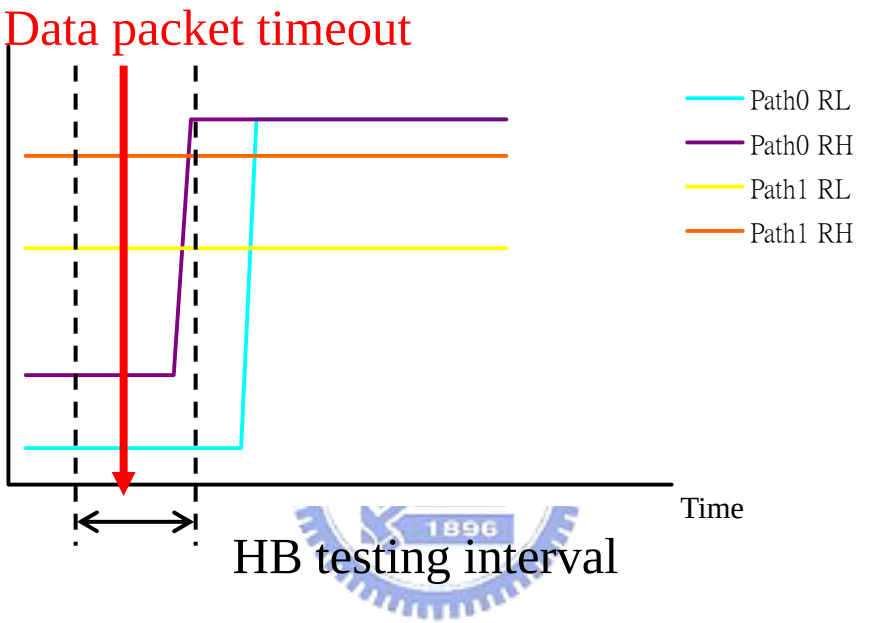

圖 2-12 資料封包逾時發生時之路徑狀態示意圖

首先要說明的是資料封包逾時可能發生的時機。基本上在 delay-based handoff 中，主要是依據每次測量到的 RTT 資訊來進行路徑的轉換，但路徑 RTT 測試每一段時間才會有一次，因此路徑的 RTT 資訊其實是離散的，而資料封包的逾時可能就發生在兩次相鄰的 RTT 測試之間。如圖 2-12 所示，原本的 RTT thresholding 方法會在一次 RTT 測試之後才造成了 RL/RH 區間的變化，從而開始並傳。但對於實際上資料傳輸而言，當路徑狀態變差時，在 RTT 測試測得路徑狀態之前可能就發現了資料封包逾時，此時就應該立即對重傳的路徑作出選擇。而既然資料封包逾時代表著路徑有不良狀況，路徑有不良狀況時作為 RTT 測試用的 heartbeat 也很可能會發生 timeout 的狀態，所

以為了保險起見，便如同 heartbeat timeout 之後的運作模式，就會立即開始並傳(若是像 RTT 方法及 SRTT 方法並不支援並傳，也就只能直接轉換到另一條路徑，以避免資料封包可能連續的遺失)，直到 RTT 測試到路徑狀態為止，才依照原本的方式來選擇路徑。

在實作上，則將發生資料封包逾時的路徑註明為”不穩定路徑”，只有在從此路徑上接收到 HB ACK 時才會取消這個標記。當一條路徑被註明為不穩定路徑的時候，這條路徑就不會被選擇為唯一用來傳輸資料的路徑(即使 RTT 資訊顯示它比較快)，直到取得下一個路徑的 RTT 資訊為止。此時可能有以下兩種狀況：1) RTT 資訊是由 HB timeout 產生，此路徑依然是不穩定路徑，不過也表示路徑狀況真的很差，因此的確應該進入並傳的狀態，而當 HB timeout 持續發生數次，便會轉換到另一條路徑而不再使用原本的路徑，這時候就算原本路徑還是不穩定路徑也無所謂。2) RTT 資訊是由 HB ACK 回來所產生，此時此路徑不再是不穩定路徑，然而路徑的選擇也就可以交還由原本的機制做選擇。基於以上的方法，就對資料封包逾時之後重傳路徑的選擇制定出一套合理的方法。

第三章、實驗設計與分析

這個章節將會敘述實驗所用的方式，並分析討論其結果，以驗證本論文所提出的方法有其優越性。將依序介紹實驗環境、實驗流程、和實驗結果及分析。

3.1 實驗環境

本論文使用 Network Simulator Version 2(NS-2)[19]來架構整個實驗環境，並利用其上的 SCTP 模組，對其進行修改，實作出實驗用到的各種方法。實驗所使用的網路環境如圖 3-1 所示，包括兩個網路端點，其間由兩條網路路徑所連接。其中一條 Path0 用以模擬 WLAN 的狀態，頻寬為 11Mbps、路徑延遲為 60ms；另一條 Path1 用以模擬 3G 的狀態，頻寬為 2Mbps、路徑延遲為 440ms。以這樣的設計來模擬各種方法在同時擁有 WLAN 及 3G 網路時的運作方式。

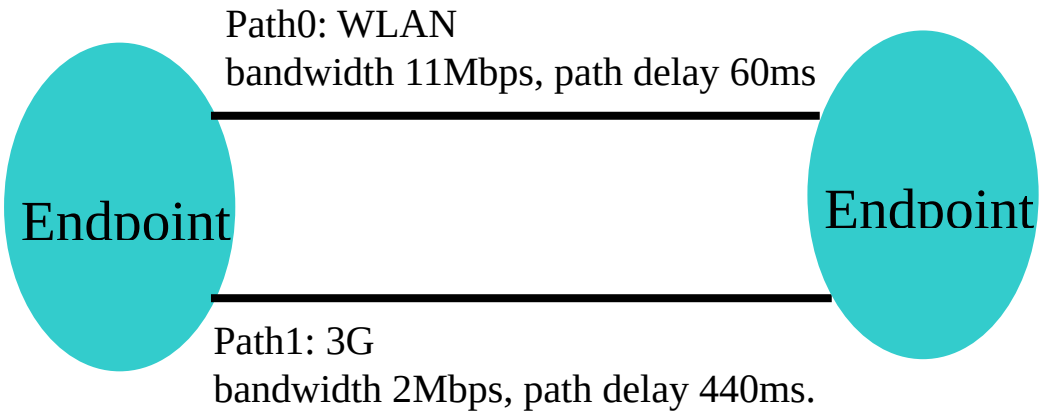

圖 3-1 實驗網路環境

3.2 實驗流程設計

初始狀態下資料將透過 Path0 來傳輸，然後透過 Path0 的狀態變化，觀察 5 種不同的路徑選擇方法的運作結果來加以比較。這 5 種方法包括有：

- 原始 SCTP 方法。
- RTT 方法。
- SRTT 方法。
- RTT thresholding 方法，其中更調整其所使用的 RTT 紀錄數量，分別從最近的 3 個、5 個、及 10 個 RTT 紀錄中選取其最高跟最低值來界定 RL/RH 區間。
- Simulcast 方法，為永遠都使用兩條路徑同時傳送資料的方式。

Path0 的狀態變化，以 60ms 為基礎，每秒為一區間，每區間的資料傳輸都有可能遭受同一延遲；延遲發生率從 0%到 100%，以 10%為一階段，觀察不同延遲發生率下的運作狀態。延遲長短也分為 3 種模式，以涵蓋各種可能的狀態變化類型。這 3 種模式包括有：

- 固定延遲模式，每次延遲的長度皆為固定。
- 隨機延遲模式，每次延遲的長度在一區間內隨機產生。
- 遺失模式，每次延遲的長度都相當長，將會造成封包逾時，模擬路徑不通而造成封包遺失的狀態

在我們的實驗中，將會測試兩種固定延遲模式，延遲長度為 300 ms 及 800ms、一種隨機延遲模式，延遲長度為 0~800ms、及遺失模式，如圖 3-2 所示。

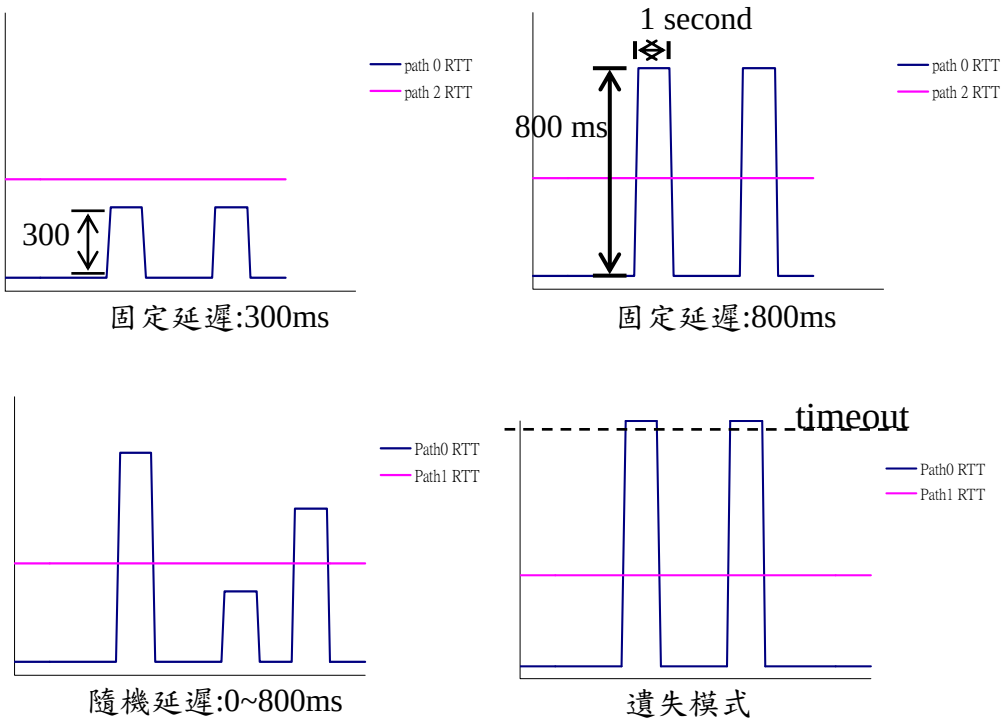

圖 3-2 實驗所用各種延遲模式

實驗所使用的測試資料流，基於本論文針對網路遊戲應用，網路遊戲資料流的特性是小封包、低頻率的傳輸。因此每個封包大小選定為 200bytes，每 200ms 傳送一個封包，從 Path0 狀態變化開始後傳送 60 秒的資料，並紀錄這些封包在網路間往來的資訊，用以分析運作狀態。且由於網路狀態變化是隨機產生，每種狀況的組合都跑 20 次再做平均。

3.3 實驗結果及分析

由封包往來的資訊我們將計算出以下幾個數值，包括反應時間平均值(average response time)，最高 10%反應時間的平均值(average of largest 10% response time)，以及額外網路負載比率(overhead%)。

首先要對”反應時間”定義，一般而言，路徑的反應時間就以 RTT 表示，是為封包送出去到 ACK 之間的時間長短。但對於網路使用者而言，例如說網路遊戲的玩家，其感覺到的反應時間應該是他輸入指令到得到回應之間的時間。這和 RTT 的差別在於，指令輸入後有可能會在 SCTP 的 send buffer 裡等待一段時間才從網路上送出。這是因為 SCTP 的 congestion control 機制，為了避免路徑上有太多的資料同時在傳輸而造成壅塞，從而限制送出卻尚未 ACK 的封包資料量，因此造成某些資料必須在 send buffer 之中等待之前的封包 ACK 之後才能進行傳送。這樣的現象特別容易發生在難以取得封包 ACK 情況下，如封包太常遺失。因此為了反應使用者的感覺，本論文的反應時間長度將採用資料輸入的時間到資料有 ACK 之間的時間間隔。

以下敘述我們所計算的三個數值所代表的意義。

- 反應時間平均值: 這個數值計算所有傳送封包的反應時間的平均值，用以表示整體反應時間的高低。基本上，反應時間平均值越低，表示此狀況下運作效能越好。

- 最高 10%反應時間的平均值: 這個數值計算所有傳送封包的反應時間中最高 10%數值的平均值，用以表示此狀況下運作時最差狀況。這數值越低，表示此狀況下即使是最差狀況也有越好的表現。

- 額外網路負載比率: 這個數值計算額外傳送的封包量與原本需要傳送的封包量的比值，以百分比表示。額外傳送的封包包括並傳所多傳的封包和封包遺失之後重傳的封包。此數值用以表示為了讓封包能確實與即時送達所要付出的代價。

在觀察實驗數據之前，需要先說明兩個路徑選擇方法所代表特殊意義。第一個是原始 SCTP。原始 SCTP 對路徑狀態的感知相當緩慢，只有在連續發現封包逾時之後才會進行路徑的轉換，並未針對路徑反

應時間作優化。所以從原始 SCTP 方法所測量到的反應時間數值，表示一種並未進行最佳化的狀況，理論上可視為最差的狀況。第二個是 simulcast 方法。simulcast 方法永遠都使用兩條路徑同時傳送資料，因而總是可以得到最佳的反應時間，所以可以視為最好的狀況。

3.3.1 固定延遲模式 300ms 之實驗結果

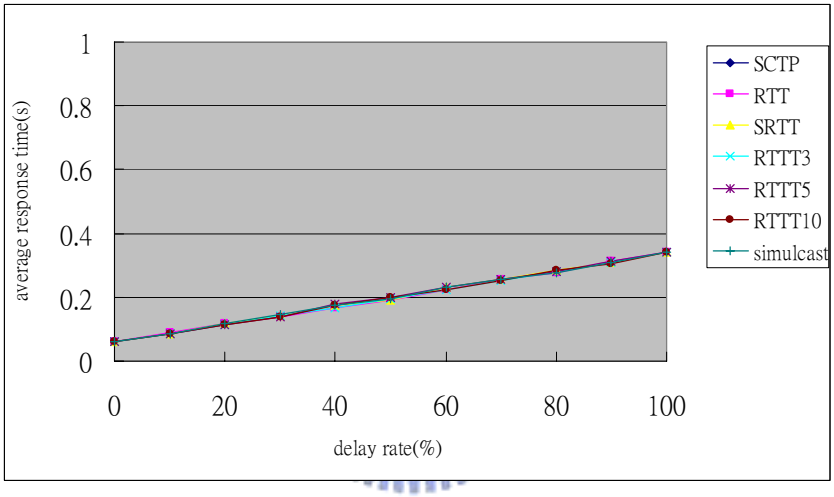

圖 3-3 固定延遲 300ms 的反應時間平均值

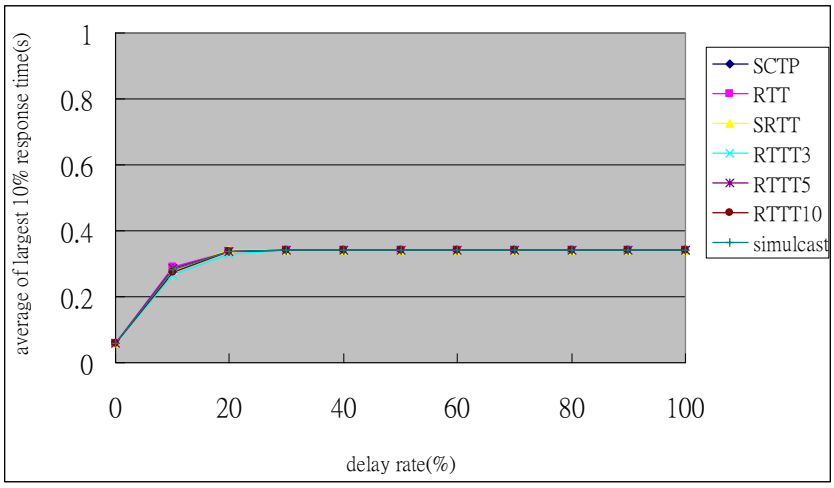

圖 3-4 固定延遲 300ms 的最高 10%反應時間的平均值

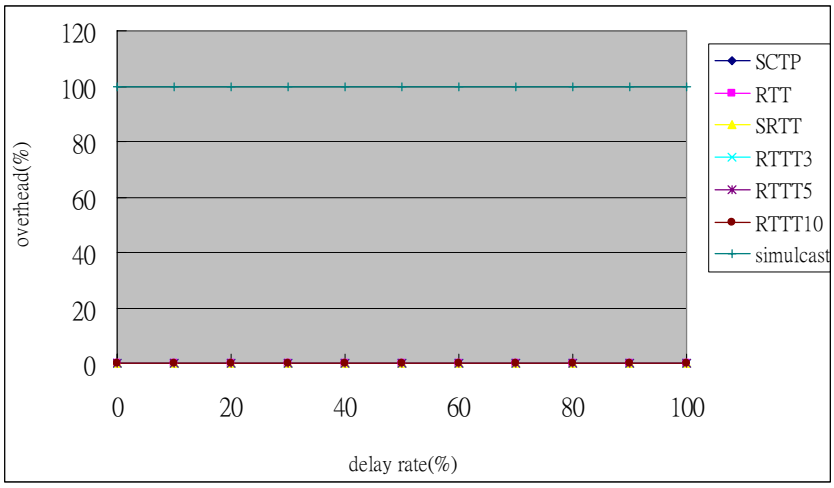

圖 3-5 固定延遲 300ms 的額外網路負載比率

首先觀察的是固定延遲 300ms 時各種方式的運作模式的實驗數據，如圖 3-3、圖 3-4、及圖 3-5 所示。可以發現所有的方法得到的反應時間平均值以及最高 10%反應時間平均值都相同。這是因為 Path0 的延遲只有 300ms，不論延遲發生的機率為何，即使發生延遲，Path0 的反應時間依然低於 Path1 的 440ms。所以對 RTT, SRTT 或是 RTT thresholding 方法而言，就會繼續使用 Path0，不會進行任何路徑的轉換。而原始的 SCTP 本來就不會因網路反應時間作路徑轉換。Simulcast 方法必然可以得到最佳的結果，但是在不需要作路徑轉換的情況下，永遠並傳就造成了一種無謂的浪費，這點可以從額外網路負載比率中觀察出來，只有 simulcast 方法有著 100%的額外網路負載，其他都是 0%。

3.3.2 固定延遲模式 800ms 之實驗結果

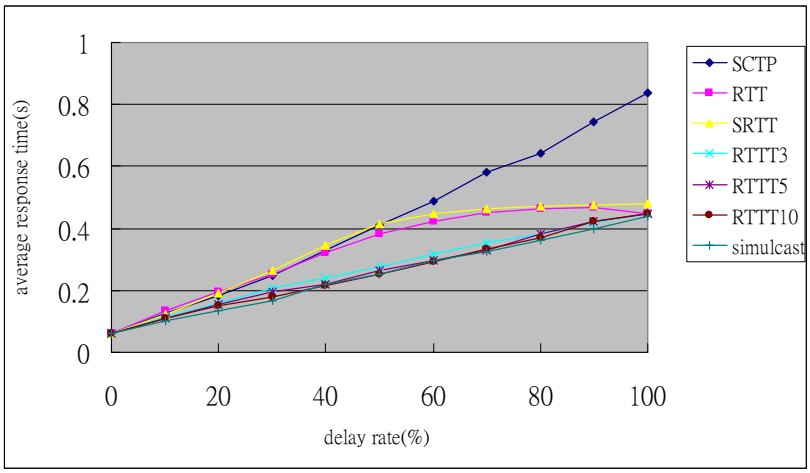

圖 3-6 固定延遲 800ms 的反應時間平均值

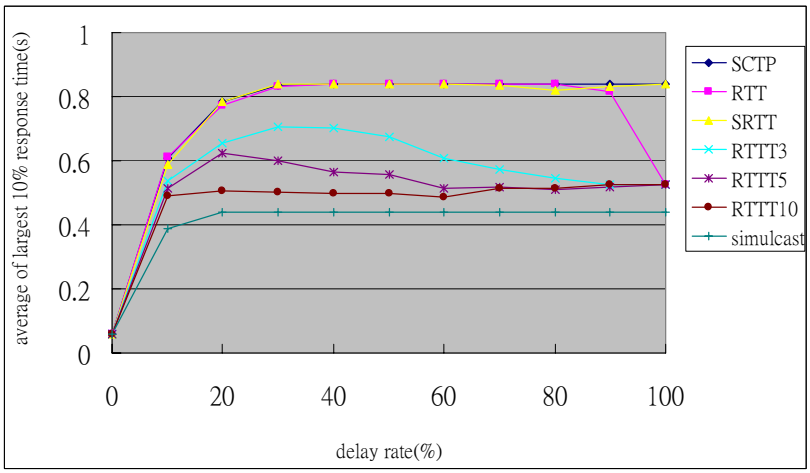

圖 3-7 固定延遲 800ms 的最高 10%反應時間的平均值

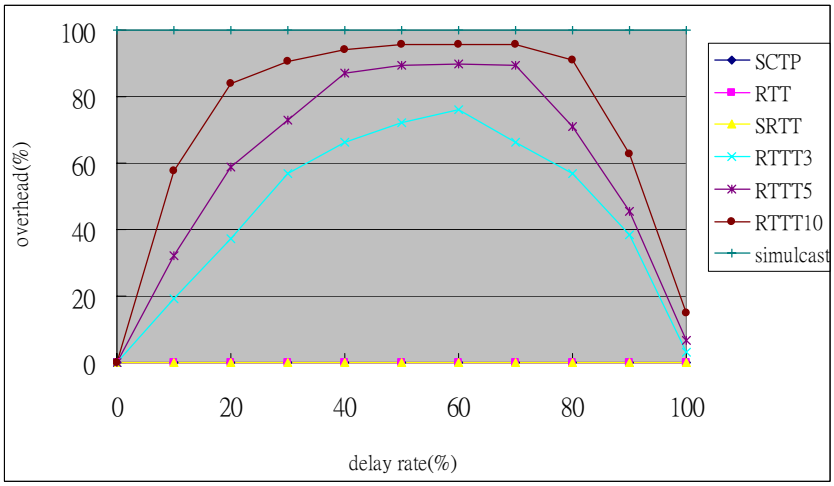

圖 3-8 固定延遲 800ms 的額外網路負載比率

再來觀察固定延遲 800ms 的實驗數據。如圖 3-6、圖 3-7、及圖 3-8。800ms 的延遲將會使 Path0 的反應時間高於 Path1，因此延遲發生的時候就有路徑轉換的必要。

首先要注意的是原始 SCTP 與 simulcast 方法的數值。在反應時間平均值方面，原始 SCTP 由於不會做路徑轉換，所以隨著延遲發生機率的升高，反應時間平均值從 0%時的 60ms 以固定斜率上升到 100%時的 860ms；而 simulcast 方法，由於能取得最佳的反應時間，因此在 path0 發生延遲時，就能通過 Path1 取得 440ms 的反應時間，因此反應時間平均值從 0%時的 60ms 以固定斜率上升到 100%時的 440ms。而在最高 10%反應時間平均值方面，除了 0%代表沒有延遲發生，其餘延遲發生的機率都在 10%以上。原始的 SCTP 會遭受到所有的延遲，所以取最高的 10%反應時間平均值都會得到趨近 860ms 的值。而 simulcast 則是在延遲發生時會得到 Path1 的反應時間，因此取最高的 10%反應時間平均值就會得到趨近 440ms 的值。從圖形中可見，在反應時間平均值與最高的 10%反應時間平均值上，這兩個方法的得到的實驗數據如同上下界，其他方式的數值幾乎都被夾於其中。這結果符合對這兩種方法運作狀況的推測。

所以 RTT 與 SRTT 方法的反應時間平均值在延遲發生率較低的區間是趨近於原始 SCTP 的數值，這代表這兩個方法在這些狀況下的運作結果如同沒有處理一般糟糕，而在延遲發生率較高的情況下就向 simulcast 方法趨近，表示路徑選擇趨向正確的方向；但觀察到其最高 10%反應時間平均值，就算是到了高延遲發生率的區段，無論是 RTT 還是 SRTT 方法大都仍與原始 SCTP 相同，這代表仍有許多時間會遭受到 Path0 延遲的影響，其原因就是由於 RTT 方法可能誤判與 SRTT 路徑轉換緩慢所造成的影響。

而 RTT thresholding 方式，無論是從最近的 3、5、或 10 個 RTT 紀錄來選擇 RL/RH 區間，其反應時間平均值都接近於 simulcast 方法

的結果，這就代表著此方法的運作方式堪稱良好；只有在最高 10% 反應時間平均值上可見到低延遲發生率區段的依然有些許高起，但依然低於 860ms，這表示承受 Path0 延遲的比率小於 10%，而且可以見到選用紀錄的時間越長，其數值就越快開始下滑而趨近 simulcast 方法的數值，這一方面是由於越長的紀錄可以反應越低比率的延遲發生，不過同時也就造成了並傳時間的延長，這點就可以由額外網路負載中觀察到，其間的取捨就依照使用者的需求來做選擇。

3.3.3 隨機延遲模式之實驗結果

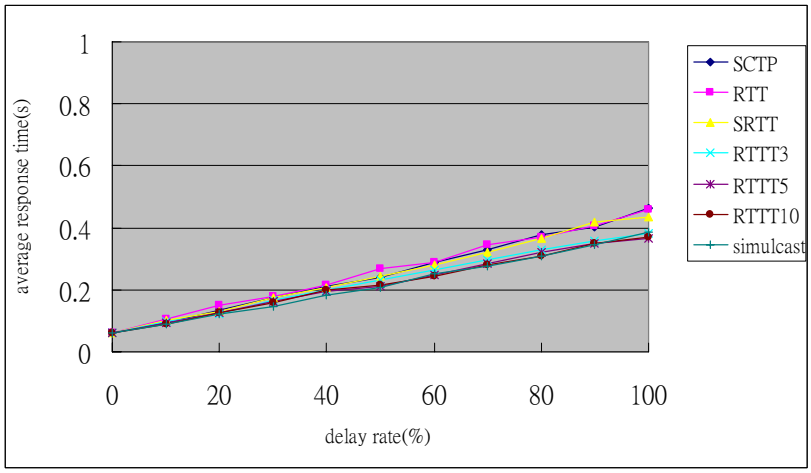

圖 3-9 隨機延遲的反應時間平均值

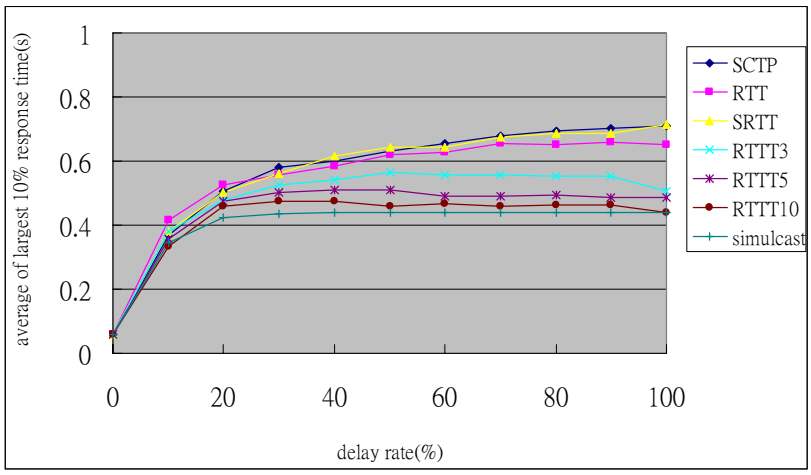

圖 3-10 隨機延遲的最高 10%反應時間平均值

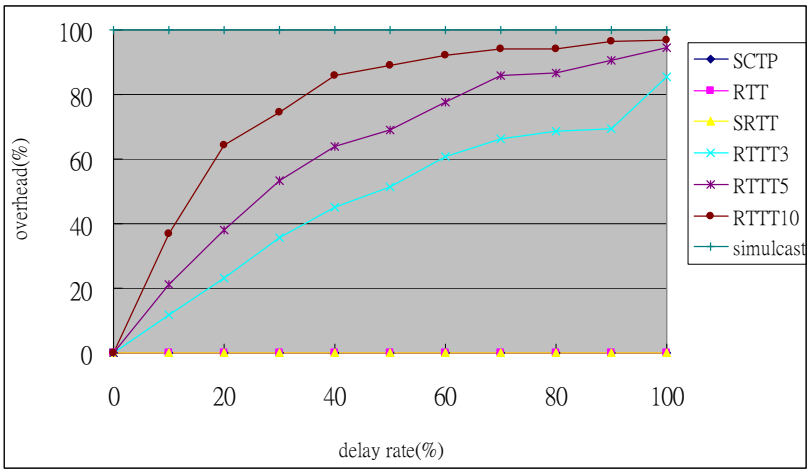

圖 3-11 隨機延遲的額外網路負載比率

圖 3-9、圖 3-10、與圖 3-11 是隨機延遲模式的實驗數據。從反應時間平均值來看，不會做路徑轉換的原始 SCTP，從 0%的 60ms 以固定斜率上升到 100%的(60ms+860ms)/2 = 460ms，這是由於 100%的隨機延遲代表所有的反應時間值會平均分布在 60ms~860ms 之間；而 simulcast 方法則是由 0%的 60ms 以固定斜率上升到 (440-60/800)*[(60+440)/2]+(860-440/800)*440 = 349.75ms，這是由於儘管隨機延遲讓 Path0 有延遲時的反應時間在 60ms~860ms 之間跳動，不過在 60ms~440ms 區間，由於低於 Path1 的 440ms，所以會得到 Path0 的反應時間，只有在 440ms~860ms 區間，才得到 Path1 的 440ms。最高 10%反應時間平均值，有延遲時原始 SCTP 應該得到 (860*90%+860)/2 = 818ms 的值，不過由於隨機分布的影響，而較此值略低；simulcast 則仍是得到最佳的 440ms。

從圖上可見，其他方法的反應時間及最大 10%反應時間依然分布在原始 SCTP 與 simulcast 方法之間，SRTT 方法與 RTT 方法依然接近原始 SCTP，RTT thresholding 也是接近 simulcast，而且整體趨勢類似於固定延遲 800ms 的較低延遲發生率的區段。其實這是可預期的結果，因為延遲取向轉換是在比較相對的大小，就算是 100%產生隨機延遲，只要不跨越 Path1 的 440ms 就不會發生路徑的轉換，也就類

比於低頻率發生超過 440ms 的固定延遲模式。這樣的運作方式從額外網路負載也看得出來。

3.3.4 遺失模式之實驗結果

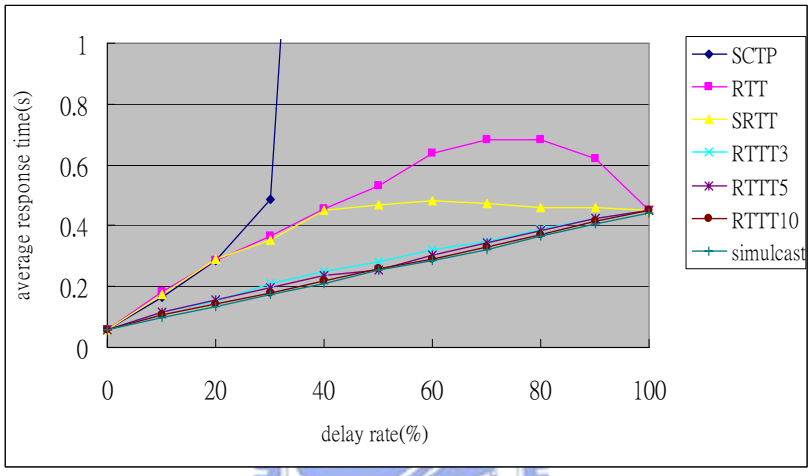

圖 3-12 遺失模式的反應時間平均值

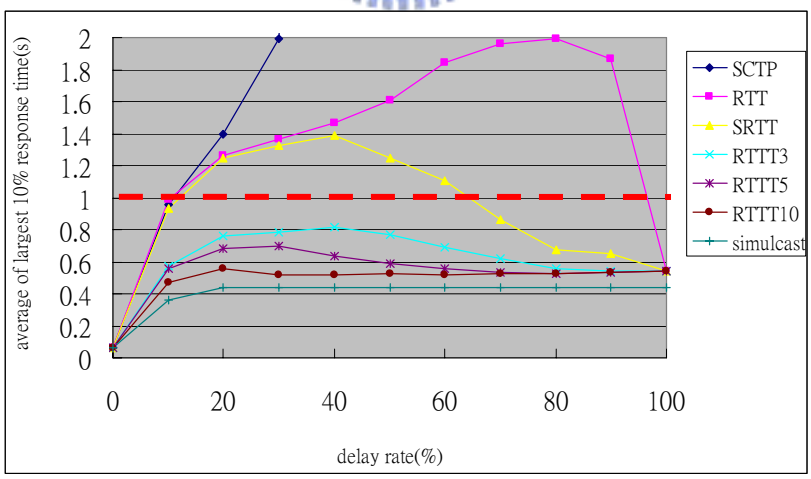

圖 3-13 遺失模式下的最高 10%反應時間平均值

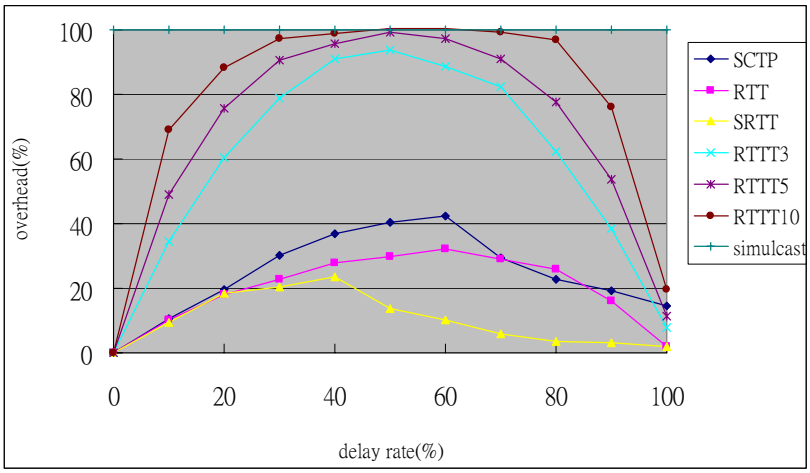

圖 3-14 遺失模式下的額外網路負載比率

最後圖 3-11、圖 3-12、及圖 3-13 顯示遺失模式下的實驗數據。可以見到原始 SCTP 在 Path0 開始有不通情形發生的時候，反應時間平均值就快速的上升，最大 10%反應時間平均值更是急遽增高，這是由於路徑不通將造成封包遺失，原始 SCTP 只會在資料封包逾時之後才重傳遺失的封包，而封包逾時的等待時間長度又是以 2 倍指數成長，等待期間若是 congestion window 滿了，還會有越多的封包在 send buffer 之中等待之前的封包 ACK 完之後才能傳送，因此造成了超長的反應時間。然而 simulcast 方法在 Path0 上路徑不通的情形下，依舊能使用 Path1 將封包送達，其運作結果就與固定延遲 800ms 的模式完全相同，因為無論是延遲與封包遺失同樣會都被視為慢於 Path1 的情形。

其餘方法的結果雖然仍在這兩個方法之間，不過若與固定延遲 800ms 的結果相比較，有些情形的曲線變化模式不太相同。首先是 RTT 方法在 Path0 不通的情況偏多卻非完全不通的時候，無論是反應時間、最高 10%反應時間，甚至是額外網路負載比率，都是比較高的情形，這就是由於 RTT 方法容易誤判所造成的結果。當 Path0 通暢的機會偏低，一但因為快速反應轉換回 Path0，造成封包再度的遺失機率就會很高，而且封包逾時的等候時間也會拉長，所以反應時間或是重傳的資料量都會上升。而 SRTT 方法在反應時間平均值的變化

與固定延遲 800ms 相似的;但是最高 10%反應時間平均值,以圖 3-13 上的紅色虛線：1 秒為界線，這是封包逾時等待時間的下限，在不通比率偏高的時候反而呈現下滑的趨勢，甚至低於此界線，即代表封包遺失的量相當少,這點也可以從額外網路負載中看出。這是因為 SRTT 方法考量路徑平均值的特性,不會因為稀少的通暢而切換到不通的路徑上，而且加上封包逾時之後立即轉換路徑重傳的機制，也降低了 SRTT 反應緩慢的影響。

然而 RTT thresholding 方法，依然也保有接近 simulcast 方法的結果，整體的曲線變化模式也相近於固定延遲 800ms 的狀況，只是最高 10%反應時間平均值與額外網路負載的數值都略高，顯示出封包遺失造成的高延遲與重傳所帶來的影響。

基於上述實驗，我們證明了 RTT thresholding 方法在大多數路徑不穩定情形下，將會有效利用並傳來取得快速的反應時間，其結果優於原始 SCTP 及同為 delay-based handoff 的 RTT 方法與 SRTT 方法。

3.3.5 效率分析

除此之外，以下將對 RTT thresholding 方法從不同數量的 RTT 紀錄中取得 RL/RH 值的效率進行分析，並提供一種標準來證明此方法的效率。

各種方法的效率將通過” 額外網路負載比率”及”延遲發生時受其影響的程度”來觀察。這是由於理想狀態中，當額外網路負載比率提高時,延遲發生時受影響的程度便該下降,才能反應出額外負載所帶來的利益。在這裡我們選用了兩個方法，以其之間的差距為基準，用以標準化其他各種方法的數值。第一個方法即為並傳方法，此方法永遠使用兩條路徑同時傳送資料,也因此永遠可以得最快速的反應時

間，將延遲發生的影響降到最低；所以將此方法的額外網路負載比率定義為 100，而延遲發生時受其影響的程度定義為 0。另一個方法則是假設網路狀況已知，在連線初始時便挑選出一條整體狀況受延遲影響較低的路徑，傳輸過程中就不再進行路徑的轉換。所以此方法的額外網路負載比率為 0。至於延遲發生時所受影響的程度，儘管是選定一條整體狀況受延遲影響較低的路徑，依然必須承受部分延遲帶來的影響，只是在這樣的選擇之下，能避免遭受不必要的延遲，因此將延遲發生時受其影響得程度定義為 100。在此定義下，兩標準方法之效率總合數值皆為 100。又，延遲發生所造成的影響將反應在整體的反應時間上，即可用反應時間平均值來表示，因此由兩種方法反應時間平均值差即可以得出用以標準化受延遲影響程度的數值。

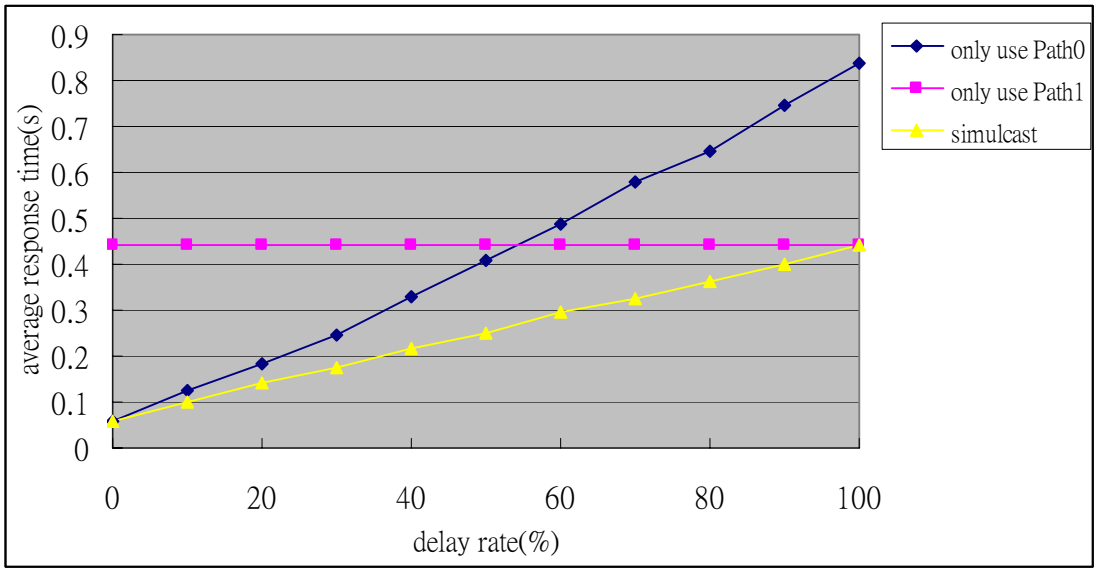

圖 3-15 固定延遲模式 800ms 用以標準化受延遲影響程度之數值

以固定延遲模式 800ms 的情況為例，由圖 3-15 所示，在 Path0 延遲發生率 50%以下的時候，這些情況下只使用 Path0 受延遲影響程度是在一半以下，而只使用 Path0 的反應時間平均值也較只使用 Path1 為低，表示這些情況下若要選定單一路徑應該使用 Path0 為佳；而 Path0 的延遲發生率在 50%以上的情況下則反之，表示此時應該選用 Path1。而其數值與並傳方法平均反應時間之間的差值，則為各種延

遲發生率下用以標準化受延遲影響程度之數值。

有了標準之後，將數種方法的反應平均值與並傳方法的反應時間平均值相減，經過標準化，即得圖 3-16，反應出各種延遲發生率下受影響之程度。

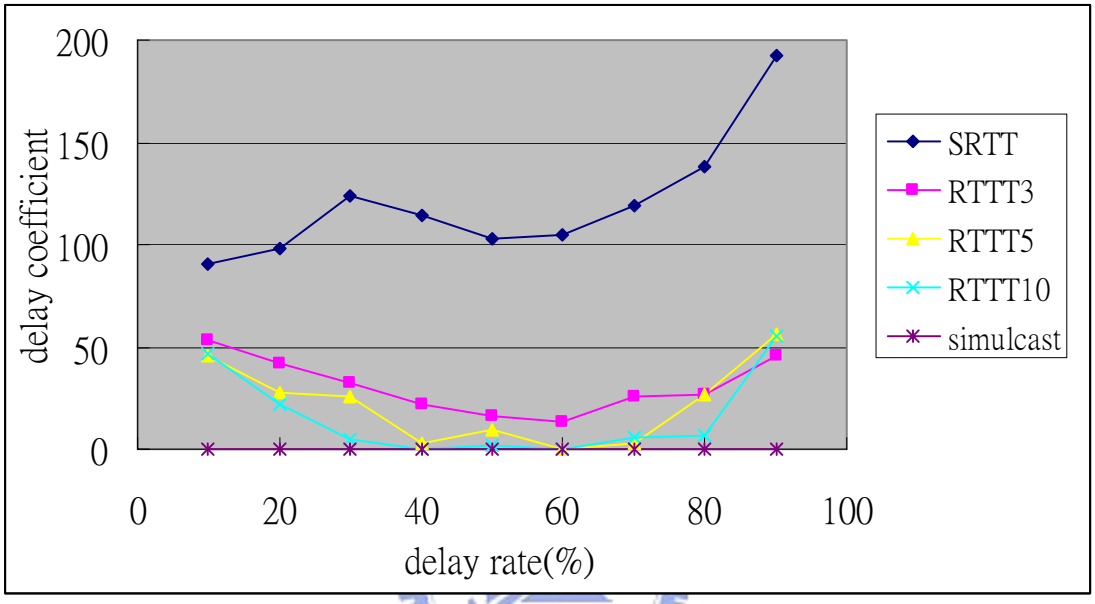

圖 3-16 固定延遲模式 800ms 下各種方法的受延遲影響之程度

受延遲影響程度只有 10%~90%之間的數值，是由於 0%與 100%時只使用單一路徑的反應時間平均值與並傳方法相同，相差為 0 則無法標準化，這次因為此時只使用單一路徑就是最佳解，不需要並傳，自然也沒有隨時作路徑轉換的必要。在 10%~90%之間，可以看出 SRTT 方法受延遲影響的程度甚至大過於 100，這便是由於 SRTT 反應路徑狀態太慢時額外遭受的延遲所造成的影響。而其他方法則多在 50 以下。

若是再加上額外網路負載比率來觀察整體之效率，則如圖 3-17 所示。

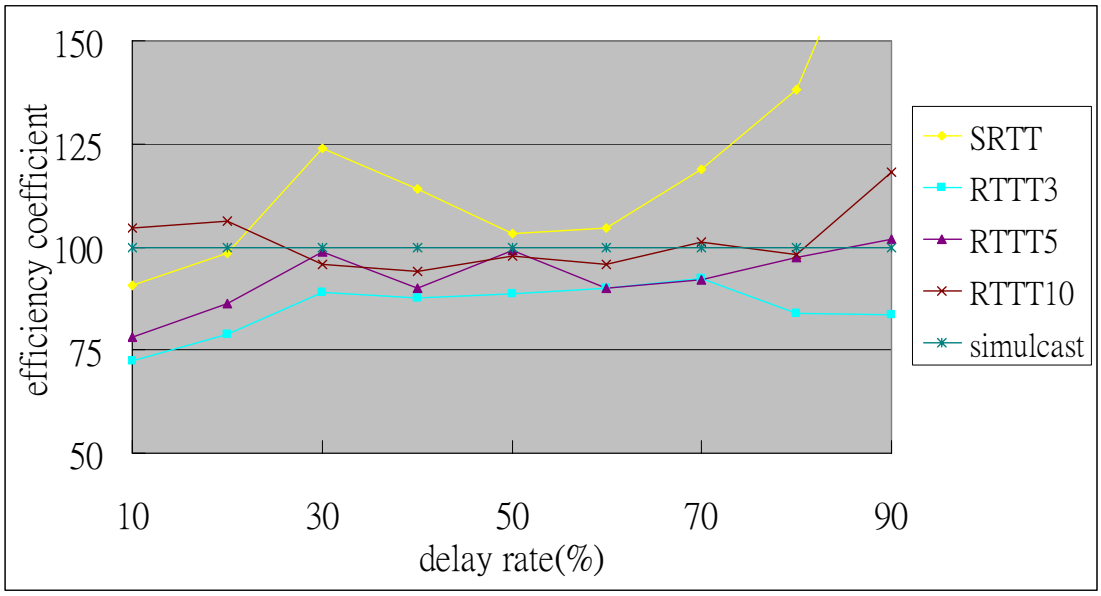

圖 3-17 固定延遲模式 800ms 下各種方法的效率

由圖中可觀察到並傳方法在各種狀態下，其總合數值皆為 100，是為一標準。SRTT 方法儘管額外網路負載比率為 0，但卻因遭受額外的延遲影響，所以總合數值依然多在 100 之上。而 RTT thresholding 方法的數值則多在 100 之下，表示其並傳所造成的負載能包容更大比例的延遲影響，整體效能堪稱良好。更細微的觀察，可以見到從不同數量的 RTT 紀錄中選取 RL/RH 區間，也各有不同的效率。其中數量為 10 的方法在低比率不穩定的狀況下，數值是大於 100，表示這樣的選取區間並不夠適當，會有效率不好的情形發生；而以效率論，則數量為 3 的方法是所比較的方法裡面數值最低，因此是效率最好的。

基於以上分析，提供了一個方法來評斷各種方法的效能，同時證明 RTT thresholding 的運作效率是相當良好的。

第四章、結論與未來發展

在本論文中，以測量網路 RTT，輔以並傳的方式，加以利用所有可用的網路介面，提出一個能有效率擁有快速反應時間的無縫式轉換方法。此方法有以下的特性：

- **用 RTT thresholding 方法表示路徑的狀態：**
  以 RTT 資訊，界定出路徑反應時間可能變化的範圍，能良好的表示路徑的狀況，為路徑轉換的抉擇提供合理的依據。

- **以反應時間最快的路徑來傳輸：**
  依照 RTT thresholding 所描述的路徑狀況選擇最快速的路徑，達到最快反應時間的需求。

- **使用並傳來改善反應時間：**
  當發現路徑狀況不穩定，難以抉擇較快速的路徑時，使用並傳的方法，以降低只使用單一路徑可能導致的延遲。

基於所設計方法的性質，可以歸納出以下的未來發展：

- **更精準的路徑狀態表示法：**
  本論文提出的 RTT thresholding 方法，是用 RL/RH 區間來表示路徑狀態。用以選擇 RL/RH 的紀錄時間越長，能容忍越低比例的暫態所造成的影響，卻也容易造成並傳時間的拉長。因此是否有更好的方法來決定路徑反應時間可能變化的範圍？

- **超過兩個以上的網路路徑：**
  本論文的路徑轉換方法是依照兩條路徑的相對狀況加以抉擇。但若擁有超過兩個以上的網路路徑，其相對狀況將更加

複雜，此時在不穩定狀況下該選擇使用哪些網路介面來並傳也是相當值得研究的主題。

- **可移動性：**

  行動裝置的漫遊可能會造成可使用的網路路徑動態的狀態改變或增減。因此未來可以整合 Mobile SCTP[15]的規格，以達到對可移動性的支援。

參考文獻

[1] I. Aydin and C. Shen, “Cellular SCTP: A Transport-Layer Approach to Internet Mobility”, October 2003.

[2] R. Chakravorty; P. Vidales; K. Subramanian; I. Pratt;J. Crowcroft, “Performance issues with vertical handovers - experiences from GPRS cellular and WLAN hot-spots integration”, Pervasive Computing and Communications, 2004, Proceedings of the Second IEEE Annual Conference on, March 2004.

[3] L. Coene , “Stream Control Transmission Protocol Applicability Statement”, RFC 3257, April 2002.

[4] Tom Goff and Dhananjay Phatak, “Unified Transport Layer Support for Data Striping and Host Mobility”, To appear in the IEEE Journal of Selected Areas in Communications, Special Issue on Wireless IP networks, 2004.

[5] H.-Y. Hsieh and R. Sivakumar, “A Transport Layer Approach for Achieving Aggregate Bandwidths on Multi-Homed Mobile Hosts,” ACM/Springer Wireless Networks, vol. 11, no. 1-2, pp. 99-114, Jan. 2005.

[6] Z, Jiang, K, Leung, Byoung-Jo J. Kim. P. Henry, “Proxy Servers Based Seamless Mobility Management”, WCNC 2002, Orlando, FL, 2002.

[7] Shigeru Kashihara, Katsuyoshi Iida, Hiroyuki Koga, Youki Kadobayashi and Suguru Yamaguchi, “End-to-End Seamless Handover using Multi-path Transmission Algorithm”, Proc. Internet Conference 2003, October 2003.

[8] Andrew Kelly, Gabriel Muntean, Philip Perry and John Murphy, “Delay-Centric Handover in SCTP over WLAN”, Transactions on Automatic Control and Computer Science, Vol.49, May 2004.

[9] LAN/MAN Standards Committee of IEEE Computer Society,

"802.11g-2003", IEEE standard, http://standards.ieee.org/getieee802/download/802.11g-2003.pdf

[10] Li Ma, Fei Yu, Victor Leung, Tejinder Randhawa, "A New Method to to Support UTMS/WLAN Vertical Handover Using SCTP", IEEE Wireless Communications, vol. 11, no. 4, pp. 44-51, August 2004.

[11] D. Maltz and P. Bhagwat, "MSOCKS: An architecture for transport layer mobility", in Proc. IEEE INFOCOM, April 1998.

[12] L. Ong, J. Yoakum, "An Introduction to the Stream Control Transmission Protocol", RFC 3286, May 2002.

[13] Hemish Parikh, Hemant Chaskar, Dirk Trossen, Govind Krishnamurthi, "Seamless handoff of Mobile Terminal from WLAN to cdma2000 Network", 3G World - Wireless congress, San Francisco, May 2003.

[14] Charles E. Perkins, "IP Mobility Support", IETF RFC 2002, October 1996.

[15] M. Riegel and M. Tuxen, "Mobile SCTP", October 2005, http://tools.ietf.org/id/draft-riegel-tuexen-mobile-sctp-03.txt

[16] Nirmala Shenoy, Punita Mishra, Bruce Hartpence, "Performance of a Framework for Seamless Integration of Cellular and WLAN, Proceedings of Annual OPNET Technology Conference, July 2004.

[17] H. Sivakumar, S. Bailey, R. L. Grossman, "PSockets: The Case for Application-level Network Striping for Data Intensive Applications using High Speed Wide Area Networks", Proceedings of the 2000 ACM/IEEE Conference on Supercomputing, 2000.

[18] R. Stewart, Q. Xie, K. Morneault, C. Sharp, H. Schwarzbauer, T. Taylor, I. Rytina, M. Kalla, L. Zhang, and V. Paxson, "Stream Control Transmission Protocol", RFC 2960, October 2000.

[19] UC Berkeley, LBL, USC/ISI, and Xerox PARC, Network Simulator NS2, http://www.isi.edu/nsnam/ns/

[20] Wikipedia, "General Packet Radio Service",

http://en.wikipedia.org/wiki/General_Packet_Radio_Service

[21] Wikipedia, “High-Speed Downlink Packet Access”, http://en.wikipedia.org/wiki/High-Speed_Downlink_Packet_Access

[22] I-Wei Wu, Wen-Shiung Chen, Ho-En Liao, Fongray Frank Young, “A seamless handoff approach of mobile IP protocol for mobile wireless networks”, IEEE Transaction on Consumer Electronic, Vol. 48, No.2, May 2002.

[23] M. Ylianttila, M. Pande, J. Makela and P. Mahonen, “Optimization scheme for mobile users performing vertical handoffs between IEEE 802.11 and GPRS/edge networks”, IEEE Proceedings of Global Telecommunications Conference, Vol. 6, November 2001.

[24] 王志祥，“WLAN 與 GPRS 之間無縫式轉換之研究”，國立交通大學資訊工程學系，碩士論文, June 2005.

[25] 林義欽，“行動無縫式 TCP 連線環境架構之研究”，國立交通大學資訊工程學系，碩士論文, August 2004.

[26] 陳伯綱，“整合 GPRS 與 Wireless LAN 資料網路之兩階層式架構與通訊協定”，國立交通大學資訊工程學系，碩士論文, June 2000.

[27] 邱文岳，“WLAN/GPRS 整合網路下無接縫換手的方法”，國立交通大學，碩士論文, 2003.