修改 SCTP 壅塞控制以提升換手效率

吳勇霆*[a]、李維聰[a]、張恆耀[a]
私立淡江大學電機工程學系[a]

摘要—近幾年來，網際網路迅速成長，加上行動裝置的進步，使得網路換手問題越來越被廣泛討論，而在眾多方案中，串流控制傳輸協定(SCTP)[1]最為熱門。

然而，即便使用 SCTP 來達到換手，還是有許多問題，如 SCTP 中的緩速啟動與壅塞控制機制就會導致換手時期產生傳輸崩潰現象，在本論文中我們希望針對 SCTP 的緩速啟動與壅塞控制機制去解決問題，使得換手過程中，可以持續擁有良好的傳輸品質。[1]

一、 簡介

近年來網際網路與行動裝置的快速發展，越來越多的問題浮出，像是異質網路間如何整合、無線訊號如何傳遞、如何有效作點對點分群與移動時交換基地台，其中在交換基地台的部分，我們可以把它歸類為網路換手的議題，而這類的問題，在學術上已經被大量的研究，如何有效率的換手，更是網路換手中十分重要的議題。

1.SCTP

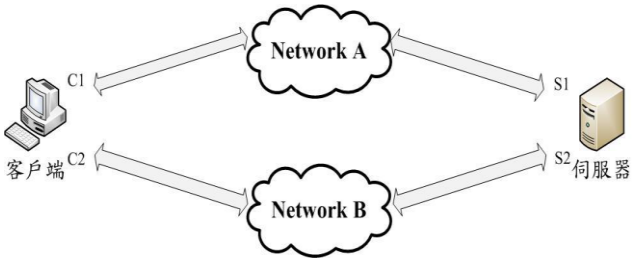
1.1介紹

SCTP (Stream Control Transmission Protocol)是在 2000 年由 IETF 的 SIGTRAN 工作組定義的一個傳輸層協議，提供了終端與終端之間的溝通，其傳送機制類似 TCP(Transmission Control Protocol)，相似的地方在於具有連結導向、可信度、壅塞和流量控制，而改善 TCP 的部分分別是多重連接、連結終止與動態位置配置，使其提供了更多傳輸上的特色。

1.2Multi-homing

在網路機器與裝置介面之間，Multi-homing 提供了重複性與終端對終端的錯誤容忍度機制，與 TCP 和 UDP 不同的是，在 TCP 中，會有四種連結，分別為(C1,S1)、(C1,S2)、(C2,S1)、(C2,S2)，而在 SCTP 中，其連結(Association)是一個點綁定多個 IP 位置，如圖一所示，在 SCTP 中的連結可以看做({C1，C2}，{S1，S2})，而當資料的傳輸完成以後，SCTP 會終止連結。

Multi-homing 的另一項特色是 Dynamic Address-Reconfiguration(DAR)，該特色使得 SCTP 可以在已經運作在一個 Multi-homing 的傳輸中，其中有一條路徑是用來做為負責傳送使用者資料的主要路徑(Primary Path)，而其他次要路徑則是用來重傳及傳送其他訊息，另一方面，當主要路徑發生斷線時，終端就會重新選擇一條新的路徑來當作主要路徑，讓連結(Association)能夠繼續進行；換句話說，也就是除了主要路徑之外， 其它路徑只用來重傳時使用或是當成主要路徑的備用路徑，因此該特色使得 SCTP 在建立連結(Association)時，可以有多重的連線路徑選擇。

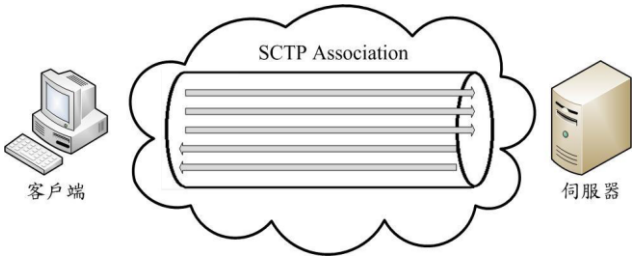


圖一、Multi-homing概念圖

1.3Multi-Streaming

每個 SCTP 的連結(Association)中是由多個串流(Stream)所組成的，每個串流都是單向和彼此獨立的資料流，並且每個資料流都有屬於自己的序列空間，因此當資料流抵達接收端時，接收端不會考慮每個資料流抵達的相對順序，只會依照資料流內的序列來傳遞資料。

在建立 STCP 連結(Association)時，就會先規畫好有多少輸出串流與輸入串流，舉例來說，如圖二所示，在此連結(Association)中，客戶端使用了三條輸出串流和兩條輸入串流，有一點是必須注意的，在此連結中每一個串流都會共享壅塞控制，因此 SCTP 是依附在 TCP 的公平性原則上來運作的，在此特色之下，任何串流中的封包遺失時，只會影響到該串流中的封包，而不會影響到其他串流。



圖二、Multi-Streaming概念圖

1.4架構與原理

SCTP 近似於 TCP，在傳送或接收資料前會先建立連結(Association)，在建立連結(Association)時，兩個終端都會向對方提供一個 Port Number 和一個 IP 位址，而每個連結(Association)是由許多個單向的 Stream 所組成的，一旦資料轉送完成後，兩終端將會終止連結(Association)。

[1]本研究由國科會贊助，計畫編號 NSC 102-2815-C-032-006-E。

Corresponding author:yotiwu@gmail.com
DOI:10.6159/IJSE.2014.(4-1).42

2.無縫換手

行動裝置在交換基地台訊號的同時，基本上會處在一段失去訊號的時間，如何有效率的進行換手之所以十分重要，正是因為一旦失去訊號的時間過長，使用者便會察覺網路中斷，並且為此不滿，而如果能讓使用者感覺不出網路中斷，則網際網路的發展與重要性則會日益提升，因此如何讓使用者無法感覺到換手時的網路中斷成為了目前很重要的一個研究方向－無縫換手。

3.相關研究

關於換手，目前已有多種做法，像是 IEEE 提出的 IEEE 802.21 MIH[3]便是架構在網路七層的第二層與第三層間做為溝通橋樑，藉由 SAP(Service Access Point)對第二層以下的架構進行控制，並允許第三層以上對第二層以下進行詢問，藉由 MIH function 達到無縫換手的效果，不過由於是透過 MIH function 所以只能使用在有支援 MIH function 的環境下，像是 802.3、802.11、3GPP 與 3GPP2…等。

另一個廣為人知的換手解決方案則是使用串流控制傳輸協定(SCTP)作為傳輸協定，因為 SCTP 有著十分適合換手的特性，我們可以更加清楚的知道 SCTP 的特性，像是 Multi-homing、Multi-streaming 與動態位置配置(DAR)，其中在 SCTP 換手相關研究[6][7]中更是指出了如何使用 SCTP 來完成網路換手，並且建立了如何使用 SCTP 進行網路換手的三個步驟。

後來大部分使用 SCTP 進行換手的相關研究大多採用這四個步驟進行換手，如圖三所示，四個步驟分別為：

3.1 獲得新 IP：

當節點進入網路重疊區時，取得新基地台 IP。

3.2 動態建立新 IP 連線：

當節點取得新 IP 時，利用 SCTP 的動態位置配置的特色在已經運行的 Association 中建立新 IP 的連線。

3.3 切換主要路徑：

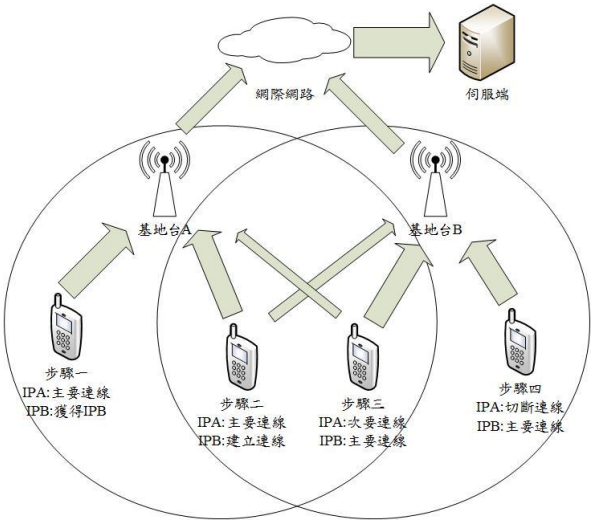
當舊有 IP 的基地台提供的訊號強度低於一定閾值時，將會採用新 IP 的連線為主要連線路徑。

3.4 動態刪除舊 IP：

當節點已經完全離開舊有 IP 的基地台時，如同步驟二，利用 SCTP 的動態位置配置的特色在已經運行的 Association 中切斷舊 IP 的連線。

基於這四個步驟，我們可以藉由 SCTP 進行網路換手。

然而，雖然 SCTP 的特色很適合處理網路換手，但是卻也存在了一些不利於網路換手的因素，其中像是 Slow-Start 與 Congestion Avoidance 這些壅塞處理的演算，將會導致網路換手的行為時分沒有效率，而本篇論文正是要針對使用 SCTP 進行網路換手的效率進行改善。



圖三、SCTP 換手步驟

二、 研究方法

在改善換手效率以前，必須先認識是什麼降低了網路換手時的效率，原因出在 SCTP 為了進行壅塞控制時所建立的壅塞控制演算法，SCTP 類似於 TCP 會對於遺失的封包進行重傳送，但這個機制將有機會造成嚴重的網路壅塞，因為如果當網路狀況十分壅塞的同時，封包遺失量必然會大量增加，一旦遺失量上升，重新傳送的資料又會增加，重新傳送的資料上升，整體的資料量必然又會攀升，緊接著壅塞問題會加重，最後會導致更多封包遺失，為了避免這個問題無窮的相扣下去，所以不論 TCP 或 SCTP 都有設計一套壅塞控制機制，在 SCTP 的部分對於網路狀況的控制主要分為兩個部分，分別為 Slow-Start 與 Congestion Avoidance，在 Slow-Start 的部分是為了確保傳輸量穩定的成長，緩慢增加傳輸量至最大安全傳輸量，因為這個機制將會使 SCTP 傳輸時不會一開始就因過量的傳輸導致網路壅塞，另外 Congestion Avoidance，則是確保 SCTP 發生壅塞的時候能夠自我調節傳輸量以降低網路壅塞的狀況。

不論是 Slow-Start 或是 Congestion Avoidance 都與 SCTP 中的三個參數有關分別為，Congestion Windows(cwnd)和 Slow-Start threshold(ssthresh)與 flightsize，首先 cwnd 為壅塞視窗，這個參數紀錄的是以建立的連結可以累積的最大未認證資料量，根據探測的結果，如果傳輸狀態勢穩定的則會往上攀升一個 MTU(最大傳輸單位，Max Transmission Unit)，而 ssthresh 則是一個用於決定該 SCTP 連結是處於 Slow-Start 還是 Congestion Avoidance 的參數，當連結創立時，初始值為無窮大，每當發生重傳事件時則會成為一個可數值，其判斷條件為 ssthresh 大於 cwnd 時呈現 Slow-start 機制，反之則為 Congestion Avoidance，而 flightsize 則為未認證的封包大小，而 SCTP 在傳輸時存在著一個原則：

如果傳送端送至目標位址的未認證資料大於或等於 cwnd 時，則傳送端不能繼續傳送新資料到給定的傳輸位址[1]。

根據這個原則我們可以得知，cwnd 是控制傳輸量的一大因素，而問題就產生在這裡，當我們進行網路換手時，勢必會進行至 SCTP 換手四大步驟中的第三個步驟，切換主要路徑，這個時候新 IP 的連結因為沒有大量傳輸過資料，因此 cwnd 的數值將會經過 Slow-start 階段，這時將會造成網路傳輸的效率突然的下降，為了解決這個問題，我們想到了一個方法，就是將最大 cwnd 的值進行預存，等到主要路徑切換時，便將預存的 cwnd 給予新的主要路徑並且對於換手後的重傳機制做修改，以提升換手效率。

在我們所提到的方法中，我們設計了一個預存壅塞控制參數的結構，並宣告為 CwndRecord，如圖五所示，pre_cwnd 儲存了在傳輸過程中所達到的最大壅塞視窗參數值，pre_ssthresh 儲存當下了閥值以便判斷是何種壅塞控制機制，另外 changeFlag 則是用來判斷目前是否已切換主要路徑，而在我們方法中的總共會有三個步驟，分別為：

1.預存最大的 cwnd 值：

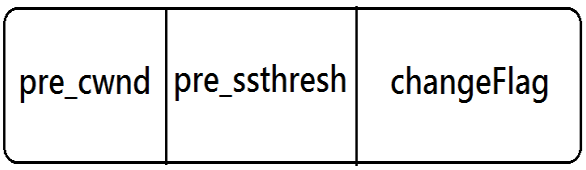
每當壅塞控制機制觸發時，便會調整 cwnd 與 ssthresh，所以就在調整過後立即判斷目前的 cwnd 是否為最大值，如果是的話就將目前主要路徑的 cwnd 與 ssthresh 分別存進 CwndRecord.pre_cwnd 與 CwndRecord.pre_ssthresh，將最大的 cwnd 值記錄下來。

2.將預存的 cwnd 給予新路徑：

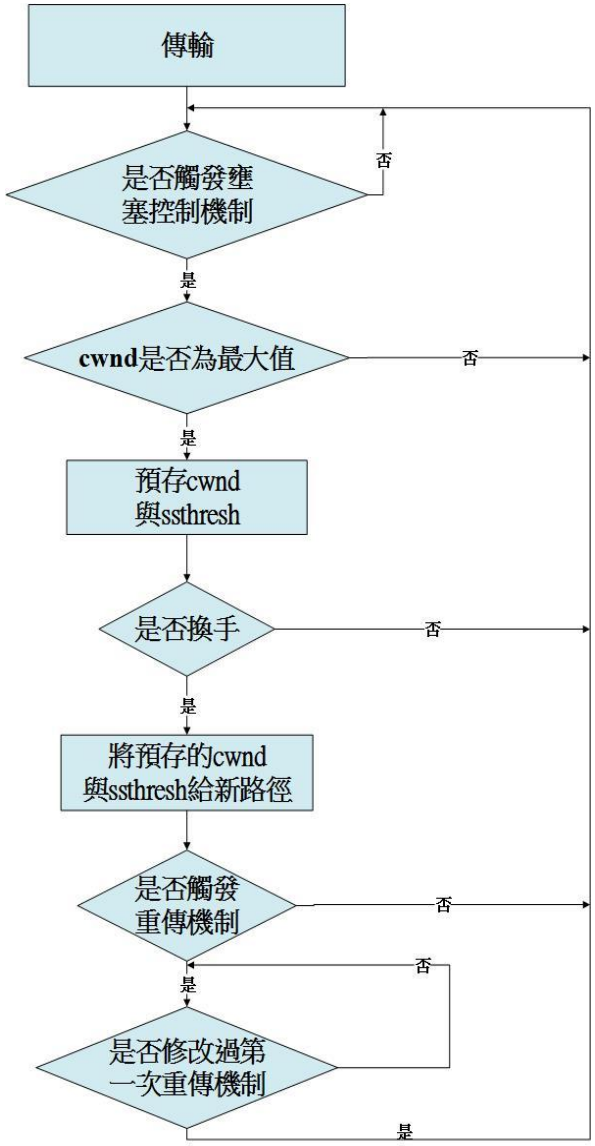
在切換主要路徑的情況發生時，將先前所預存的 ssthresh 與最大 cwnd 值給予新的主要路徑，目的是為了讓新路徑的 cwnd 不會從初始值慢慢爬升，使得傳送的資料不會因此而大幅下降，即使發生了重傳逾時或快速重傳機制，也不會導致 cwnd 過度地減少，另外也會將 CwndRecord 中的 changeFlag 設為 1，表示已經切換主要路徑了。

3.修改換手後第一次的重傳機制：

當換手完成後，會因為封包遺失而觸發重傳逾時或快速重傳機制，在此時會使得 cwnd 減少為原本的一半，在此我們會先藉由 CwndRecord 中的 changeFlag 判斷此時是否是換手後觸發的第一次重傳機制，並且修改機制將 cwnd 減少為原本的四分之三，目的是為了讓傳送的資料量不會因此而大量減少，最後再將 changeFlag 設為 0，讓重傳機制能夠照原本的去執行，方法流程圖如圖六所示。



圖五、預存壅塞控制參數結構



圖六、換手流程圖

三、 模擬與分析

在本論文中，我們使用 NS2(Network Simulator，網路模擬器)來模擬環境，設定的環境如下：

第一：傳送資料時間為 400 秒。

第二：進入重疊區的時間為 1 秒(199.5~200.5 秒)。

第三：由路徑 1 往路徑 2 前進並且在第 200 秒時切換主要路徑，環境規格表如表 I 所示。

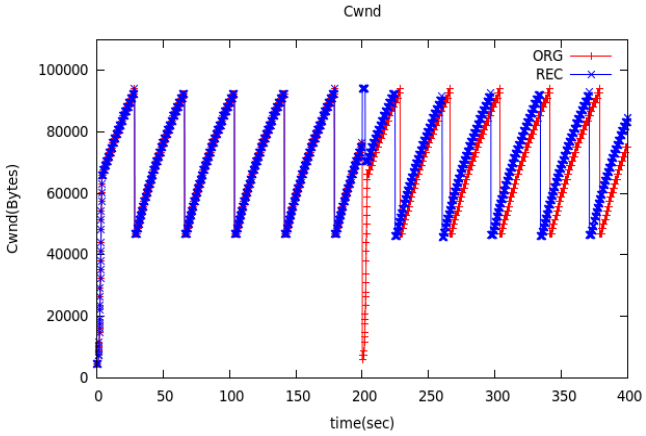
表 I
模擬環境規格表

	頻寬	路徑延遲時間
路徑 1	0.5Mb	150ms
路徑 2	0.5Mb	150ms

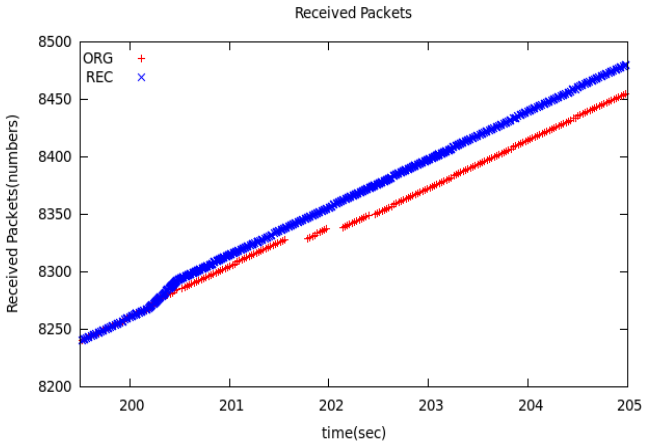
在本實驗中，我們模擬的的情況為相同頻寬下的換手效能測試，採用 SCTP 原始做法與改進做法進行運作，並透過 cwnd 以及收到的封包數來觀察是否有提升

換手的效益，而原始做法我們稱其為 ORG，改進作法為 REC。

圖七與圖八為 cwnd 與收到封包數的比較圖，在此兩圖中可以看到在第 200 秒左右進行換手時，改良的做法透過維持傳輸封包數增進了換手期間的傳輸量，另外，由於在 199.5 秒到 200.5 秒之間處於重疊區，從圖八可發現，在 200 至 200.5 秒間收到的封包數上升較快，原因是在重疊區有兩條路徑同時連結，舊路徑原本已進入傳送緩衝區(queue)的封包會繼續傳送，直到 200.5 秒離開重疊區後，只剩下新路徑傳送的封包被接收。



圖七、壅塞視窗比較圖



圖八、收到封包比較圖

四、 結論與討論

隨著網路與行動裝置的持續進步，在網路換手方面的研究將會更加重要，如何達到無縫換手更是其中最為重要的部分，在本篇論文中我們考慮了 SCTP 的各項參數與規則，進而利用控制 cwnd 來改變壅塞控制對於換手時期的傳輸效率降低，從最後的結果可以得知此方法具有顯著的效果。

在此研究中並沒有特別考慮遺失的封包，而是著重在是否有穩定的傳輸，因此在未來會考量如何減少封包的遺失，我們將會針對各項壅塞控制可以導致換手的規則進行研究，並加強本論文中提到的方法，進而在考慮遺失的環境中比較新舊方法中的差距，另外，更真實的模擬環境也不容易存在著相同傳輸量的換手行為，因此未來也會針對在不同傳輸量的切換上如何避免 SCTP 的規則造成換手效率的下降這個方向繼續努力發展。

五、 參考文獻

[1] R. Stewart, Ed., “Stream control transmission protocol.”, RFC 4960, September 2007.

[2] R. Stewart, Q. Xie, M. Tuexen, S. Maruyama andM. Kozuka,“Stream Control Transmission Protocol (SCTP) Dynamic Address Reconfiguration”, RFC5061,September 2007.

[3] V. Gupta et al, “IEEE 802.21 Media Independent Handover Services”,IEEE Standard for Local and Metropolitan Area Networks Part 21,January 2009.

[4] Keun Jae Lee,Sang Su Nam,and Byung In Mun, “SCTP Efficient Flow Control During Handover”,in *Proc.IEEE Wireless Communicatoins and Network Conference(WCNC)*,Las Vegas,Nevada,USA,Apr.2006,pp69-73

[5] Barry Leiba,“SCTP What，Why and How” IEEE Computer Society,Oct 2009,pp.81-85.

[6] S.Koh,et al.,“mSCTP for Soft Handover in Transport Layer”, IEEE Communications Letters, Vol. 8, No. 3, Mar. 2004, pp.189-191.

[7] Jingji Jin, Qijia Zhang, Lin Cui, Zhaocheng Xuan,“mSCTP Vertical Handover in the Overlap of Heterogeneous Wireless Networks”,2011 Seventh International Conference on Natural Computation, pp. 1537-1540.

[8] “The NetworkSimulator-ns-2”, http://www.isi.edu/nsnam/ns