為提供視訊快取功能而整合多重代理伺服器之高效能視訊片段鏈結策略

An Efficient Video-Segment Chaining Strategy for Integration of Multiple Proxy Servers for Video Caching

王佳龍　　童曉儒
國立屏東科技大學 資管系
{ m9256005 ; strong }@mail.npust.edu.tw

摘要

影音串流技術(Video Streaming)已廣泛地被運用在現今 Internet 中，然而由於廣域網路(WAN)之頻寬有限，且傳輸延遲過長，往往造成冗長之啟始等待與不穩定之串流品質。為改善以上問題，最常見之方法為在影片伺服器(Video Server)與使用者間，使用代理伺服器(Proxy)來暫存部分影片區段(Video Segment)，進而提升資訊分享之效益與網路回應速度。本篇論文著重在由多個獨立之 Multicast Domain 所組成之網路環境，我們提出 Video-Segment Chaining Strategy (VSCS)。本機制能跨不同之網域(Domain)建立視訊片段鏈結(Video-Segment Chaining)，讓不同網域內之 Video Segment 得以像在同網域中互相分享。經我們模擬實驗證實，VSCS 能提高影片之擊中率(hit-ratio)進而改善系統效能，並在適度增加網域內部 LAN 流量下，有效降低 WAN 頻寬。

關鍵詞：影像伺服器、視訊片段鏈結、鏈結、Prefixing / Suffixing。

1. 前言

隨著網路資訊技術之不斷發展，網路硬體傳輸速度日趨增長，軟體壓縮、播放技術也不斷地在進步當中，隨選視訊(Video on Demand, VoD)的應用已日漸普及，對於此種聲音、影片串流資料，由於檔案容量很大，在網路中會有連續一段播放之時間（約數分鐘至數小時），且影片在傳輸時也會佔用一定之網路頻寬，這類資料我們通稱為連續媒體(Continuous Media, CM)。

隨著網路硬體設備不斷地建置，LAN 跟 LAN 間不斷地做結合，在學校、稍具規模之公司、大型建築物中，我們常會發現 LAN Clustering 之硬體架構(圖 1-1)，也就是在一較大範圍之網路環境(Domain)中，利用路由器(Router)來分隔出不同之子網域(sub-domain)，sub-domain 中有可能包含一個或一個以上之 LAN，而各自之 sub-domain 中有各自之佈署與所有權，最後在 Domain 出口處共用一線路來連接至 WAN 中，共享其頻寬。

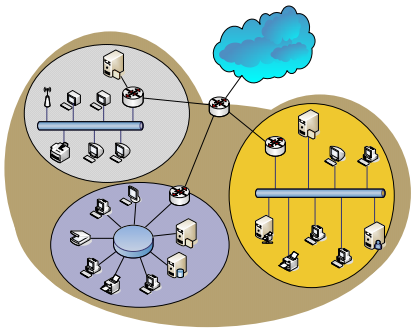

圖 1-1：LAN Clustering 架構部署圖

在上述LAN Clustering之環境當中，而於各sub-domain之所有權、流量控管、安全性等因素考量之下，通常各sub-domain內multicast是可以正常運作，而在不同sub-domain間資料只能透過unicast來進行傳送，由於現今學者提出之cache police多須藉由multicast機制來輔助其運作，因此在不同sub-domain內之Proxy Server資料除服務自己sub-domain內使用者外，並無法服務其它sub-domain內使用者以進行資料共享，造成寶貴之WAN頻寬(bandwidth)白白浪費。為了能將既有之single domain / single proxy架構下之cache policy擴充至能提供在多重Multicast Domain /多重Proxy下運作，並兼容在異質環境中傳輸CM，我們除了參考先前所提出之ASW(Adaptive Suffix-Window) / ASW$^+$(Adaptive Suffix Window$^+$)演算法[2][4]外並參考Sheu, S.等人[8]所提出之Chaining strategy；我們主要之目的是將ASW/ASW$^+$與Chaining之概念結合，將原本不同sub-domain下之Proxy Server組成一chain，如此一來相異sub-domain間Proxy中資料就能共享，用來服務不同sub-domain下之使用者，我們稱此種Chaining技術為「Video-Segment Chaining Scheme, VSCS」。

2. 文獻探討

2.1 Video Caching Policy 之介紹

由於 proxy server 所能提供的儲存空間有限，如何妥善管理與利用這些空間以達到最佳的共享效果，進而降低 WAN 的交通量，就成了極重要的

NSC 93-2219-E-020-001

研究課題。過去幾年一些 segment-based caching policy 相繼被提出。表 2-1 列舉一些較具代表性的演算法，並加以歸類標示。他們的內容概述如下。

表 2-1：Segment-based caching policy 分類表

SCHEMES	LOCAL NETWORK SERVICES		TYPE OF CACHED CONTENTS			ADAPTIVE	MULTI DOMAIN	MULTI PROXY
	Multicast	Unicast	Prefix	Patch	Suffix			
Prefix Caching[3][6]		◎	◎					
Adaptive Prefix[10]		◎	◎			◎		
Batching[7]	◎							
SBatch[9]		◎	◎					
MPatch[9]	◎	◎	◎	◎				
Suffix-Window Caching[4]	◎	◎	◎	◎	◎			
Adaptive Suffix-Window[2]	◎	◎	◎	◎	◎	◎		
VSCS	◎	◎	◎	◎	◎	◎	◎	◎

Prefix Caching：綜觀國內外有關在代理伺服器上為連續媒體傳輸所設計的快取策略之研究中[5]，發現最近幾年的有關代理伺服器快取策略的做法都是在代理伺服器中儲存一段影片[1][10]，其假設區域網路回應的速度，遠快於在廣域網路環境中，因此在代理伺服器中暫存開頭影片（prefix）的方法則能夠有效地降低使用者起始等待時間[3][6]。

Adaptive Prefix Caching：Wu, K. L.等學者[10]，認為對於大多數的影片點選行為，使用者不見得會看完整部影片，所以統計上來講以開頭片段的讀取率最高；再者影片愈是熱門，其被看完的機率愈高；因此提出此方法將多媒體物件平均切割成固定大小之 Block 方式經由網路傳送，切割後的 Block 放入不同大小之 Segment 內。若影片之點閱率越高則 Segment 越長，反之若點閱率低則 Segment 越短。

Batching：Shachnai and Yu[7]提出了 batching 的策略，其運作的方式就是讓一部影片擁有 k 個頻道，每隔 $1/k$ 的影片時間就 multicast 一次影片，server 會累積 client 的 request，直到下一個播放點到達時，以 multicast 的方式將影片一次播放給所有等待的 client。然而這種作法雖然能節省頻寬，但是也造成使用者等待最長的時間為 $1/k$ 的影片時間。

SBatch：Wang[9] 等學者，將 Prefix Caching 與 Batching 結合，而提出了 Unicast suffix batching (SBatch)的策略。SBatch 策略中，每部影片會在 proxy 中事先 cache 一段長度為 v_i 的 prefix，某部影片的第一個點選者會從在 proxy 下載 prefix。但是，對於要從 server 下載的後續影片（suffix）則盡量慢點傳輸，只要能趕上 prefix 播放完後再接續上即可。這樣做的好處是可以將多個不同時間發出影片需求的 client 一次服務完。SBatch 策略因為有 Prefix Caching 機制，比起傳統的 Batching 少了 initial delay 的問題。

MPatch：此外 Wang[9]等學者更進一步假設區域網路能提供 multicast 服務的情況下，將 SBatch 中加入 Patching 技術，提出 Multicast patching with prefix caching（MPatch）。當影片被第一個點選者選取時，就會在 proxy 上啟動一個新的 multicast stream (master client)。這個 multicast stream 會在 prefix 耗盡前自 server 下載 suffix stream 到 proxy 中，以便接續 stream。假設這個 multicast stream 啟動時間為 t=0，之後若還有同樣需要此部電影的使用者，便可依使用者緩衝區與加入時點等不同條件考慮加入此 multicast stream。這樣的做法可提升 multicast stream 共享的機率，而降低 WAN 的流量。

Suffix-Window Caching：前述之演算法多為利用 Proxy Server 使用 buffer 來 cache 影片之 prefix，企圖縮短使用者起始等待時間、降低 WAN 頻寬使用率，不過使用 prefix 之先決條件為 client 本身必須擁有足夠之緩衝區來暫存影片片段方可順利執行。這點並無法支援異質環境中緩衝區有限之使用者。林青[4]提出了 Suffix-Window Caching 之機制。也就是在 Proxy Server 中除了儲存 prefix 外（其長度為 v_i），設法 cache 一段最近播放過之 suffix 片段，稱之為「suffix window」（其長度為 w_i）來輔助後續加入之使用者，使其能有更大的機會加入 batch 中。

ASW 與 ASW$^+$ Caching：江家宏[2]進一步將 Suffix-Window Caching 之觀念加以延伸，探討當有數部影片同時存在時 suffix window 之大小應如何調配？其設計了一套即時之 suffix window 自動調適機制：ASW 與 ASW$^+$，當某部影片點播集中時，suffix window 就會自動延伸；反之如果點播分散，suffix window 就會自動縮小，釋放資源給其他的較熱門之影片，ASW 與 ASW$^+$ 差異主要在於當使用者 buffer 不足以加入 batch(sub-batch) 時，ASW$^+$ 將會以補 patch 方式幫助使用者順利加入 batch(sub-batch)中。

2.2 Chaining Strategy

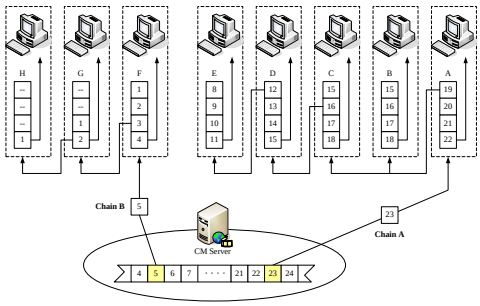

圖 2-1 Chaining strategy

為了達到不同 sub-domain 下之 Proxy Server 能資源共享之目的，我們參考 Sheu, S.等學者[8]所提出稱之為 Generalized Batching Technique 之 “Chaining”機制，其基本原理為利用各 client 端播放裝置(PC)在播放影片同時也會在本機記憶體儲存快取之特性，並且各使用者在要求服務之時間點上會有先後不同之差異性，將其快取記憶體 Chaining 在一起(圖 2-1)，經由 Chaining 後，CM Server 只需送

出一份資料，再經由 Chaining 機制不同節點之傳遞過程，就能讓連結在 Chain 中之每個 client 端節點皆能正確接收並播放同一份資料，此舉除了能降低 CM Server 之負擔外，還能節省 CM Server 至 chain 間之 WAN 網路頻寬與 LAN 端資料之有效再利用。

3. 提出之「Video-Segment Chaining Strategy (VSCS)」

3.1 VSCS/VSCS$^+$基本原理

隨現在LAN Clustering架構所需，多重Multicast Domain /多重Proxy Server支援架構已為必然之趨勢，因此我們以結合ASW/ASW$^+$與Chaining Strategy為基礎，設計一套可適用此種架構下之演算法為目標，設計出「視訊片段鏈結策略」(Video Segment Chaining Strategy, VSCS)。

我們將其傳統Chaining節點(node)[8]由PC來擔當，轉化為使用Proxy Server來運作(圖 3-1)，利用Proxy Server間彼此之合作來達成Chaining機制，由於一般之Proxy Server其系統效能、穩定性、磁碟儲存空間、I/O能力均較使用者終端設備為佳，因此利用Proxy Server擔任chain node此舉有助於提高系統穩定性、簡化Chaining之管理且有助於適用在異質環境之優點，並再結合ASW/ASW$^+$演算法，藉由演算法中Expanding Suffix Window來決定是否增加chain-node與node所需記憶體大小。

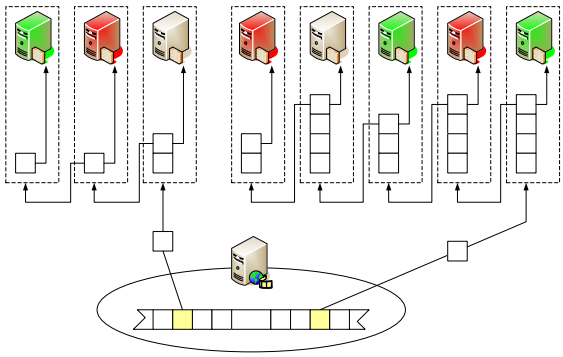

圖 3-1：Video-Segment Chaining

3.2 VSCS Algorithm

由於 VSCS 演算法是基於 ASW 演算法擴充而來，因此在 Proxy Caching 機制方面相當雷同，在單一 Domain/單一 Proxy Server 之條件下，兩機制會得到完全相同之效果。其兩演算法之效果差別要在多重 Multicast Domain 或多重 Proxy Server 支援架構下 VSCS 方能顯現出其優勢。圖 3-2 表示出當使用者於不同時間進入系統時 VSCS 演算法中 Expanding Suffix Window 之過程。以下將介紹本研究 VSCS 演算法之詳細流程。

初始設定:在同Domain中一開始每部影片可用之Suffix Window $w_i = 0$，且如果沒有任何一道master stream在系統中，則第一個使用者進入時將產生一新的master stream。

◆ **Case 1 : ($t_a \in (0, v_i + w_i]$ and $b_j \geq (t_a - w_i)$)**

圖 3-2，此圖代表影片位置對於系統時間之關係圖(Location-to-Time)，橫軸Time代表系統之時間，t_0為系統之起始時間(第一個使用者(master client)在時間t_0要求了一部影片)，t_a為後續要求相同之影片使用者(slave client)所加入之時間。縱軸為影片之播放位置，灰階區域代表是某個時間點Domian中Proxy Server所存放之資料。假設sub-domain中有一使用者(master client)在時間t_0要求了一部影片並產生一新的master stream，當sub-domain中隨後有一slave client發出影片需求之起始點落在(v_i+w_i)內，這時slave client可以加入由CM Server產生之Master multicast stream，或也可加入由Proxy Server為slave client產生之slave stream，除從Proxy Server中下載正在進行之影片外，錯過之部分則用unicast自Proxy Server下載至本身buffer中。因此slave client必須具備有可容納長度片段(t_a-w_i)之buffer。此部分機制與Prefix Caching相同。若此時Video Segment Chaining (sub-domian 1) 與使用者 (sub-domian 2) 不同sub-domain，則Local Proxy將考慮依Case 2 條件，為其Expanding Suffix Window 並與 t_0 之 Video Segment Chaining 作連結，加入 Video Segment Chaining尾端，若Proxy Server允許，將為其使用者保留Suffix window w_i(w_iSize=t_a-w_i-1)，slave client可以透過Proxy Servery在sub-domian 2 啟動一群播群組，同時可以讓sub-domain 2 後續要求服務使用者加入其群組。

◆ **Case 2 : ($t_a \in (0, v_i + w_i]$ and $b_j < (t_a - w_i)$)**

當sub-domain中，slave client發出影片需求的起始點落在$(0,v_i+w_i]$內，但是slave client之buffer size $b_j<(t_a-w_i)$，這時VSCS與ASW一樣，也會啟動R function機制依該部影片被點選頻率和$(t_a- w_i)$size，Proxy Server決定是否該為此使用者Expanding Suffix Window。(t_a-w_i)為Proxy Server需減少之buffer，當Proxy Server為此使用者儲存w_i Size($w_i=t_a-w_{i-1}$)，slave client可以透過Proxy Server啟動一群播群組，稱作salve stream，而不需要直接向CM Server要求一新master stream，這樣可以降低WAN上之傳輸頻寬，同時此新Slave multicast stream可以讓後續要求服務使用者加入其群組。若VSCS/VSCS$^+$依該部影片被點選頻率和(t_a-w_i)size，拒絕為此使用者Expanding Suffix Window，此時這slave client只能透過WAN 直接要求CM Server產生一道全新完整之Multicast master stream。

◆ **Case 3 : ($t_a \in (v_i + w_i, L_i]$ and $b_j \geq (t_a - w_i)$)**

如圖 3-2，當slave client發出影片需求的起始點落在$(v_i+w_i,L_i]$內，client buffer size足夠容納(t_a-w_i)的話，slave client可以選擇加入Last stream群播群組，並且透過WAN向CM Server使用unicast下載path$[t_a-w_i-v_i,t_a]$缺失部份影片，此部分與MPatch相

同；但若此時Video Segment Chaining與使用者不同sub-domain，則此要求將被拒絕。

◆ **Case 4 : ($t_a \in (v_i + w_i, L_i]$ and $b_j < (t_a - w_i)$)**

當slave client發出影片需求之起始點落在$(v_i+w_i,L_i]$內，已經完全無法分享master or slave stream，故只好重新開啟一個multicast session，重新扮演master client之角色，從CM Server中下載整部影片。

圖 3-3 所示為 VSCS 機制當遇有新使用者要求加入服務時所需考慮之各種條件與判斷之演算流程，我們可看到虛線標註起來之部份，為原來 ASW 演算法故有之機制，而虛線外之部分為 VSCS 為支援跨 sub-domain 所新增之機制。

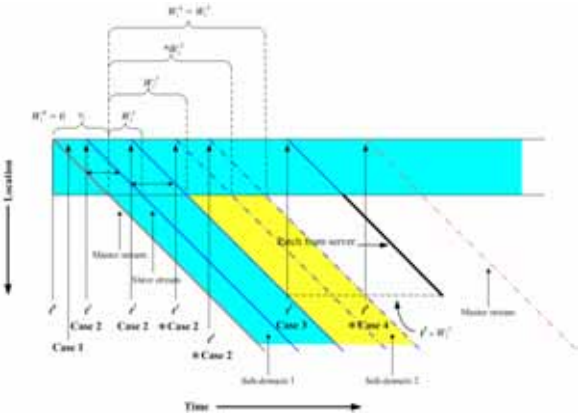

圖 3-2：PVSCS 之 Location-to-Time 圖

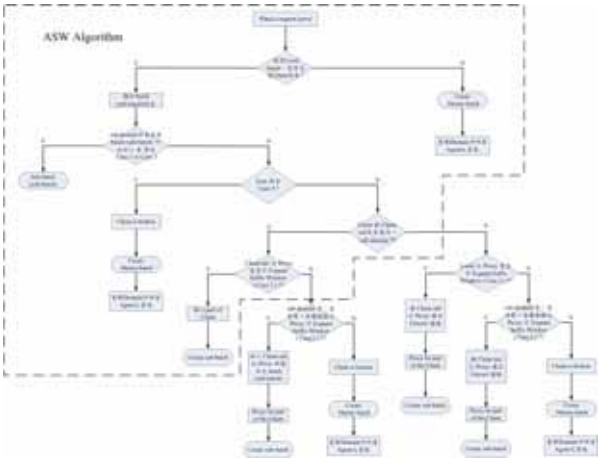

圖 3-3：VSCS 快取儲存演算流程

3.3 VSCS$^+$ Algorithm

與ASW演算法類似，在VSCS之演算法中，當slave client進入時間點落在$t_a \in (v_i + w_i, L_i]$時(參考圖 3-2)，client必須具備有可容納(t_a-w_i)影片長度之buffer，方可加入Last stream且下載path成為slave client。若是client沒有足夠之空間，必須透過WAN從CM Server重新下載影片。在異質環境網路上有些使用者設備之buffer非常有限，在這種情況下，系統可能無法服務使用者，使用者便會遭遇到此問題。

因此設計了VSCS$^+$用來改善VSCS此情況，使其在異質性環境中支援性更好，以下是其說明：

◆ **Case 4 : ($t_a \in (v_i + w_i, L_i]$ and $b_j < (t_a - w_i)$)**

圖 3-4 中，當slave client發出影片需求之起始點落在$(v_i+w_i,L_i]$內，使用者Buffer $b_j<(t_a-w_i)$已經無法分享master or slave stream，此時VSCS$^+$依該部影片被點選頻率和(t_a-w_i) Size，Proxy Server決定是否該為此使用者Expanding Suffix Window。若允許，則slave client可以經由Proxy Server啟動一slave stream，而不是直接向CM Server要求一新master stream，若此時Video Segment Chaining(sub-domain 1)與使用者(sub-domain 2)不同sub-domain，Proxy Server將為其Expanding Suffix Window並與sub-domain 1 之Video Segment Chaining作連結，加入Video Segment Chaining尾端，啟動一slave stream。以圖 3-4 為例，sub-domain 2 在t^3時點有一使用者要求服務，其進入之時間點落在$(v_i+w_i,L_i]$內，使用者buffer $b_j<(t^3-w_i^2)$已經無法分享master or slave stream，此時VSCS$^+$依該部影片被點選頻率和$(t^3-w_i^2)$size，Proxy Server決定是否該為此使用者Expanding Suffix Window，若同意則Proxy Server為其Expanding Suffix Window並與sub-domain 1 之Video Segment Chaining作連結，加入Video Segment Chaining尾端，其為使用者保留Suffix window w_i^3 (w_i^3Size=$t^3-w_i^2$)，slave client可以透過Proxy Server在sub-domian 2 啟動一群播群組，同時可以讓sub-domain 2 後續要求服務使用者加入其群組。

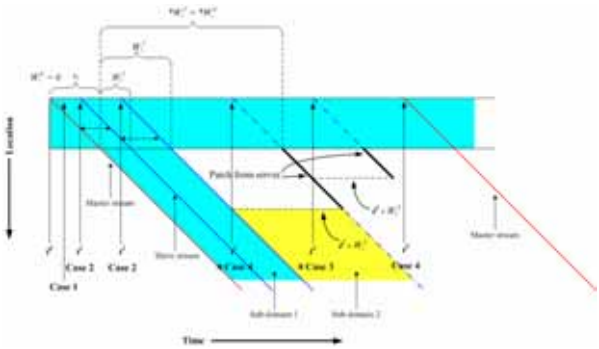

圖 3-4：VSCS+之 Location-to-Time 圖

我們同樣將本機制當遇有新使用者要求加入服務時所需考慮之各種條件與判斷之演算流程整理如圖 3-5 所示。虛線標註起來之部份，為原來ASW$^+$演算法舊有之機制，而虛線外之部分為VSCS$^+$為支援跨sub-domain所新增之機制。

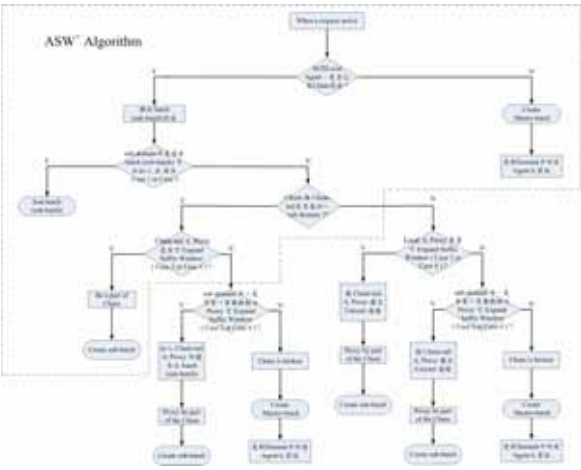

圖 3-5：VSCS$^+$快取儲存演算流程

4. 模擬實驗評估與分析

在本節我們要比較MPatch、ASW/ASW$^+$與VSCS/VSCS$^+$演算法在多重Multicast Domain/多重Proxy Server架構其是否能有效減少WAN流量、節省Proxy Server儲存空間、包容異質性環境使用者，圖 4-1 為本網路實驗架構示意圖，在整個Domain中為一LAN Clustering架構。

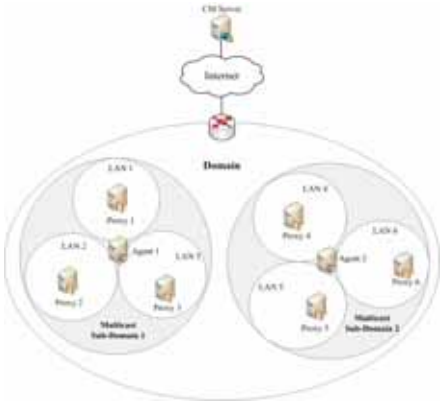

圖 4-1 網路實驗架構示意圖

4.1 系統效能指標之定義

在本模擬實驗中我們將利用WAN及LAN中之網路傳輸資料量比例(Bandwidth Gain)探討MPatch、ASW/ASW$^+$與VSCS/VSCS$^+$演算法在多重Multicast Domain /多重Proxy Server架構、異質環境中其各項特性，以下將描述其定義。

- 網路傳輸獲利值(Bandwidth Gain)：在LAN中傳輸資料量代表使用者透過LAN環境由Proxy Server下載影片，在WAN中傳輸資料量則是透過WAN環境由CM Server下載。我們探討MPatch在LAN和WAN中之資料傳輸總量，扣去ASW/ASW$^+$與VSCS/VSCS$^+$在LAN和WAN中之資料傳輸總量，再除以MPatch在LAN和WAN中之資料傳輸總量，分別以ASW/ASW$^+$ Bandwidth Gain和VSCS/VSCS$^+$ Bandwidth Gain來表示。
  - ASW/ASW$^+$ Bandwidth Gain = (MPatch Bandwidth Cost – ASW/ASW$^+$ Bandwidth Cost) / MPatch Bandwidth Cost
  - VSCS/VSCS$^+$ Bandwidth Gain = (MPatch Bandwidth Cost – VSCS/VSCS$^+$ Bandwidth Cost) / MPatch Bandwidth Cost

4.2 模擬實驗

4.2.1 實驗一：單一 Domain /多重 Proxy 對網路傳輸效能之影響

在實驗一，我們主要討論為在單一Domain中同時有多台Proxy Server協同運作之情況下，分別模擬Proxy Server運用MPatch、ASW/ASW$^+$與PVSCS/PVSCS$^+$等不同之演算法，並觀察其在系統運作期間之WAN與LAN流量變化並比較之，其參數設定如表 4-1 所示。

表 4-1：Proxy Number 影響之參數設定

參　　數	數　　值
執行次數	100
影片長度L_i個單位時間	500
單位時間到達使用者數目	0.2 人
Proxy 的大小(S)	$0 \sim 4L_i$
Mean Client buffer (C_i)	$0.01L_i$
Video Popularity distribution(D_i)	Zipf distribution
Domain Number	1
Proxy Number	**1 ~ 3**

在本文以下之表示圖中，橫軸之座標皆是表示Proxy buffer size之大小，其數據標示表示與影片長度Li之相對比例，越往右代表Proxy buffer size越大，縱軸則代表頻寬獲利(Bandwidth Gain)，四條曲線分別代表使用ASW/ASW$^+$與VSCS/VSCS$^+$演算法對使用MPatch演算法時在WAN & LAN上之頻寬獲利。

1. 由於VSCS/VSCS$^+$是由ASW/ASW$^+$改進而來，因此在圖 4-2 單一Domain、單一Proxy之情況下，WAN流量ASW = VSCS、ASW$^+$ = PVSCS$^+$，所以圖中只能看到兩條曲線。
2. 在圖 4-2 中可以看到剛開始VSCS比VSCS$^+$之獲利值較高，這是因為Proxy buffer有限，假若初期buffer被頻率低之影片佔用，將導致後來點播頻率高之影片無法使用Proxy buffer，而只能夠從CM Server下載整部完整之影片，所以VSCS會比VSCS$^+$之獲利值高，但是當Proxy buffer size再增加時VSCS$^+$明顯比VSCS來的更有效率。
3. 藉由觀察圖 4-2 我們可發現，隨著Proxy Number之增加WAN網路流量皆有減少，但在MPatch、ASW/ASW$^+$演算法中較不明顯，而利用VSCS/VSCS$^+$演算法在圖中可清楚發現WAN流量減少之情況，這是因為VSCS/VSCS$^+$能藉由Video Segment Chaining來達到不同Proxy Server

間之資源共享，因而有效率節省WAN bandwidth。

4. 由圖 4-2 Proxy Number = 2、Proxy Number = 3 中我們可發現VSCS/VSCS$^+$曲線上升速度皆優於ASW/ASW$^+$，顯然在單一Domain/多重Proxy Server架構下藉由Video Segment Chaining之幫助也能夠達到Proxy Server資源分享與有效地節省WAN bandwidth之目的。

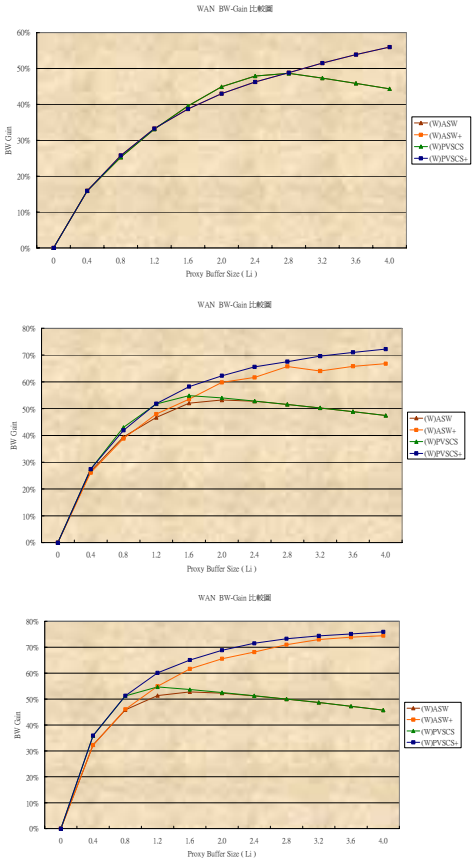

圖 4-2 Proxy Number=1、2、3 之 WAN Bandwidth-Gain 比較圖

4.2.2 實驗二：多重 Multicast Domain 對網路傳輸效能之影響

我們在實驗一已證明單一Domain/多重Proxy Server架構下VSCS/VSCS$^+$能明顯降低WAN流量，在實驗二中，我們將進一步討論當架構轉為多重Multicast Domain時對VSCS/VSCS$^+$有何影響，藉由不同sub-domain同時有多台Proxy Server協同運作之情況下，也分別模擬Proxy Server運用MPatch、ASW/ASW$^+$與VSCS/VSCS$^+$等不同之演算法，並觀察其在系統運作期間之WAN與LAN流量變化並比較之，其參數設定如表 4-2 所示。

表 4-2：Domain Number 影響之參數設定

參　　數	數　　值
執行次數	100
影片長度L_i個單位時間	500
單位時間到達使用者數目	0.2 人
Proxy 的大小(S)	0 ~ 4L_i
Mean Client buffer (C_i)	0.01L_i
Video Popularity distribution(D_i)	Zipf distribution
Domain Number	**1 ~ 3**
Proxy Number	2

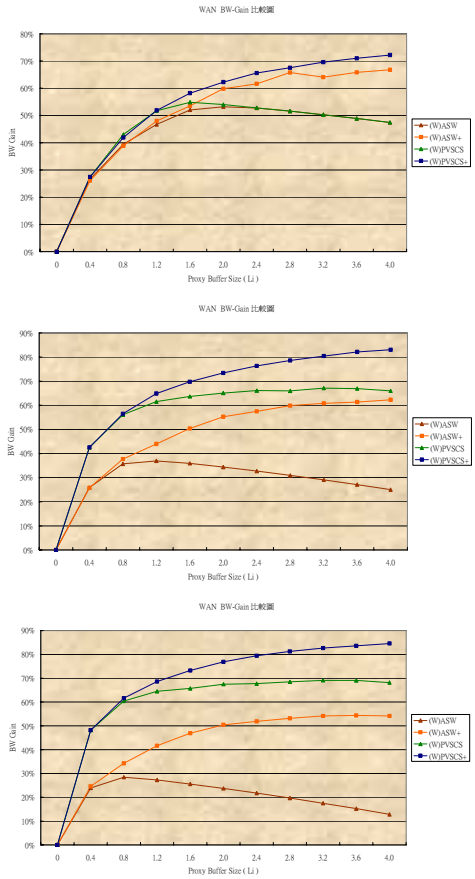

圖 4-3：Domain Number=1、2、3 之 WAN Bandwidth-Gain 比較圖

1. 比較圖 4-3 不同Domain Number下之WAN Bandwidth-Gain比較圖，我們可以清楚看到VSCS/VSCS$^+$曲線均有長足之進步，曲線上升之速度也加快，其在WAN bandwidth之節省大幅領先改良前之ASW/ASW$^+$演算法，大幅領先之原因為完全歸功於Video Segment Chaining機制，此機制不僅在相同Domain中能分享不同Proxy Server中所儲存之資料，在多重Multicast Domain環境中，藉由unicast與Chaining之觀念，順利地為不同Multicast Domain建立資料交換機制，使不同Multicast Domain中之Proxy Server亦能分享所儲存之資料，大幅減少透過WAN網路從CM Serverv下載資料之機率。因此可看出VSCS/VSCS$^+$在多重Multicast Domain/多重Proxy Server架構下有其充分運用Domian中資源之優勢。

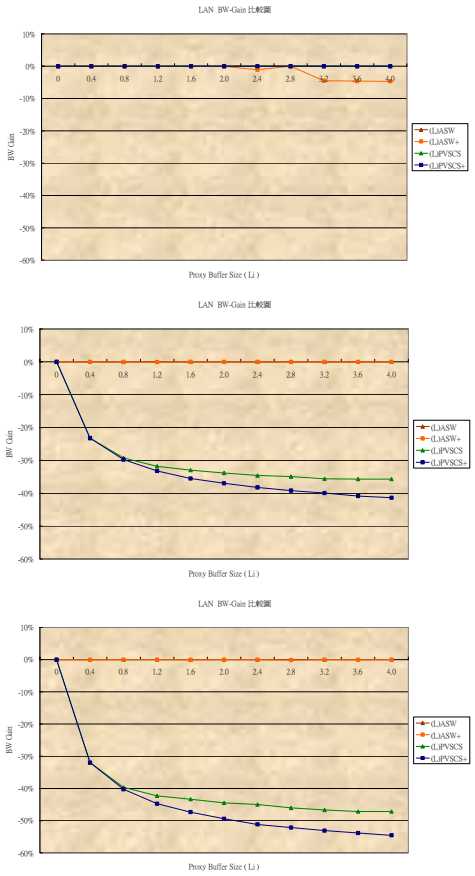

圖 4-4：Domain Number=1、2、3 之 LAN Bandwidth-Gain 比較圖

2. 但在圖 4-4 之LAN Bandwidth比較圖，我們可以發現VSCS/VSCS$^+$演算法也有其缺點，那就是與MPatch、ASW/ASW$^+$相比會佔用較多之LAN bandwidth，其主要原因為當VSCS/VSCS$^+$要發揮Video Segment Chaining機制，需從別的Multicast sub-domain 中取回資料必須在LAN中藉開啟unicast session來達成，因而造成LAN bandwidth之大量增加，這是為節省WAN bandwidth所不得不花費之代價，且我們知道WAN bandwidth價值比LAN bandwidth在經濟面上高出許多，因此我們認為此問題無損於VSCS/VSCS$^+$演算法之優點。

5. 結論與未來方向

基於時代之發展與網路之不斷進步，LAN Clustering 之結合為必要之趨勢，為符合多重 Multicast Domain /多重 Proxy Server 之網路架構，且服務異質性環境使用者，我們設計了一套稱為「高效能視訊片段鏈結策略」(Video-Segment Chaining Strategy, VSCS)。經系統模擬實驗證實，其特點不但具有減少起始延遲、有效節省 WAN 網路頻寬、自動調整 Proxy Server 儲存資源配置，更能達到包容異質環境使用者與跨 Multicast Domain 協同運作與資源分享。

未來我們將考慮在模擬實驗中 LAN 端之網路架構新增入無線網路環境(Wireless)，藉由無線網路較侷促之 bandwidth(802.11g 54Mbps)來實驗並改進我們的 VSCS 使其能有更佳之擴展性。

參考文獻

[1] Almeida, J. M., Eager, D. L., and Vernon, M. K.,“A hybrid caching strategy for streaming media files,” in Proc. SPIE/ACM Conference on Multimedia Computing and Networking, January 2001.

[2] Chiang, C. H., and Tong, S. R., “An Adaptive Suffix-Window Caching Scheme for CM proxy server (爲連續媒體代理伺服器而設計之調變式後置視窗快取演算法),” TANET 2004, Taitung, Oct. 2004.

[3] Lim, E. J., Park, S. H., Hong, H. O., and Chung, K.D., “ A Proxy Caching Scheme for Continuous Media Streams on the Internet,” in Proc. IEEE ICOIN'01, Beppu City, Japan, pp.720-725, January/February 2001.

[4] Lin, C., and Tong, S. R., “Suffix-Window Caching: A Proxy Caching Strategy for Delivery of Continuous Media in a Heterogeneous Environment,” Journal of Internet Technology, vol.5, no.1, pp.27-32, 2004.

[5] Liu, J., and Xu, J., “Proxy Caching for Media Streaming over the Internet,” IEEE Communications Magazine, vol. 42, pp.88-94, Aug. 2004.

[6] Sen, S., Rexford, J., and Towsley, D., “Proxy Prefix Caching for Multimedia Streams,” in Proc. IEEE INFOCOM ’99, New York, NY, Mar. 1999.

[7] Shachnai, H., and Yu P. S., “Exploring Wait Tolerance in Effective Batching for Video-on-demand Scheduling,” ACM Multimedia Systems Journal, vol. 6, pp.382-394, 1998.

[8] Sheu, S., Hua, K. A., and Tavanapong, W., “Chaining: A generalized batching technique for video-on-demand system,” in Proc. of the IEEE Intl Conf. on Multimedia Computing and Systems, pp.110-117, June. 1997.

[9] Wang, B., Sen, S., Adler, M., and Towsley, D., “Optimal Proxy Cache Allocation for Efficient Streaming Media Distribution,” in Proc. IEEE INFOCOM ’02, New York, NY, June 2002.

[10] Wu, K. L., Yu, P. S., and Wolf, J. L., “Segment-based Proxy Caching of Multimedia Streams,” in Proc. World Wide Web Conf., Hong Kong, China, May 2001.