以對等式網路傳輸技術為基礎之即時多媒體串流系統

蔡文能、柯開維*、吳和庭、廖梓彤
國立台北科技大學資訊工程所

摘要 — 本文為設計一個以對等式網路串流傳輸的系統，將伺服器在大量影音傳輸的負載量分散至網路中的成員，在本系統中使用拉式的傳輸模式要求取得傳輸，並透過非同步傳輸、排序與資料緩衝等演算法改善傳輸品質，達到降低影音串流來源伺服器的負載量並提供穩定傳輸的品質[1]。

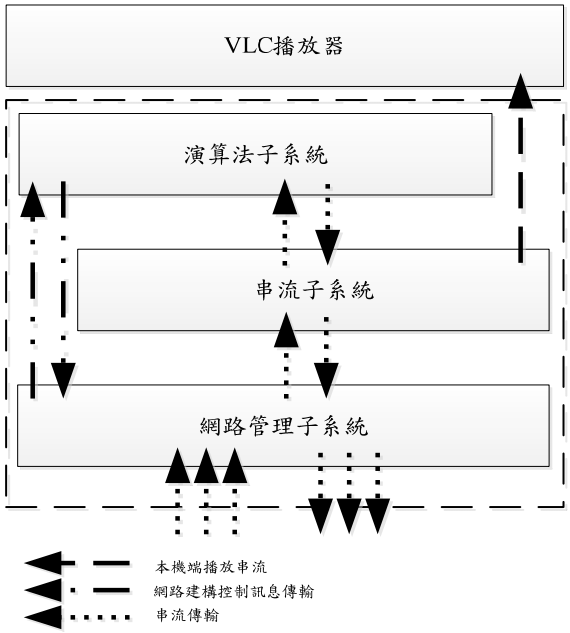
一、 簡介

科技的進步使得網路傳輸能力也不斷地提升，從早期 56kbps 提升到使用 ADSL 可以有數 Mbps 的傳輸速率，到現在使用光纖等技術可以達到數十 Mbps 甚至 100Mbps 的傳輸速率。在早期傳輸率的限制，使用者無法即時存取大量的資料，僅能夠存取文字、圖片與靜態資料，然而現今高傳輸速率的環境下，還能夠存取高品質的內容，甚至是即時多媒體資料；然而當網路使用者傳輸使用量逐漸增加，造成傳輸瓶頸導致塞車、傳輸不穩定等網路的問題，這些問題都將侷限網際網路的發展，影響的原因可分為兩部分，一為網路設備傳輸能力的不足，二為伺服器端瓶頸的限制，網路設備不斷的提升也逐漸的使伺服器端的問題凸顯出來，因此在本系統中使用對等式網路的方式達到降低伺服器端發生網路傳輸或是處理能力的問題。

改善大量存取的方式有建構內容傳的網路(Content Delivery Network, CDN)[1]與對等式網路(Peer-to-Peer, P2P)兩種方式，本系統設計以 P2P 網路架構為基礎的串流傳輸系統，採用 Pull-based 的方法[2]，使系統能夠以 P2P 分散式與叢集式有效率的方式存取網路，並參考使用 BitTorrent 的網路模型，達到有效分散網路傳輸的流量與負載，並針對串流傳輸進行最佳化，提供較佳的品質與播放的穩定度。

本系統分為演算法子系統、串流子系統與網路管理子系統三個部分，如圖一所示，演算法子系統中實作傳輸的路徑與策略，並操作串流子系統進行傳輸，演算法子系統與串流子系統的網路連線皆由網路管理子系統進行維護。網路緩衝區管理子程序將串流接收，並由串流緩衝管理子程序進行暫存保留其他成員的可重複存取性，緩衝的內容由封包排序子程序將順序進行調整，並在調整完成後進行串流的播放與傳輸，串流傳輸經由演算法子系統決定傳輸的路徑送交本地端播放器與遠端要求發送片段的成員。

[1] 本研究由國科會贊助，計畫編號 NSC102-2219-E-027-003。



圖一： 系統方塊圖

二、 網路管理子系統

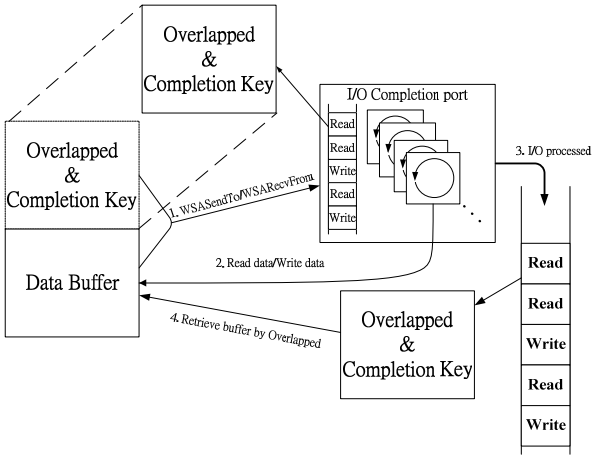
網路管理子系統分為系統 Socket 的管理與傳輸的管理兩個子程序，維護網路狀態與資料的傳輸。作業系統的 Socket 使用，首先項系統要求建立一個新的 Socket，之後必須先對 Socket 初始化設定，才能綁定系統埠號、建立連線然後開始傳輸，在傳輸完成後則必須關閉連線，若是在 Windows 環境中使用 Socket 之前必須先使用 WSAStartup 函數向系統註冊引用 WinSock 函數[3]，使用完之後則需使用 WSACleanup 函數向系統通知釋放 WinSock 引用。

系統能被要求的資源有限，如果大量使用且不加以管理在該釋放的時候將資源釋放，將造成系統資源不足等情況發生，在本系統中由網路管理子系統進行連線的管理，重複使用連線，避免過度要求系統資源。

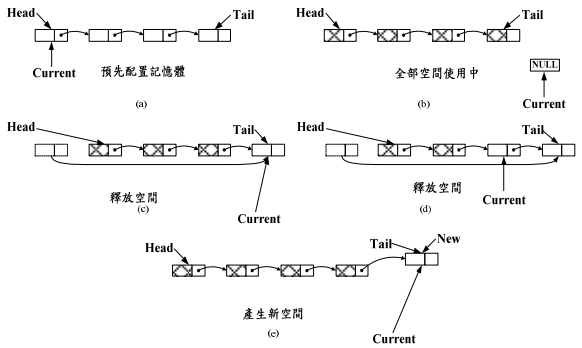
另一方面，因即時影音傳輸的高傳輸率與低時間延遲的需要，使用同步傳輸等待每一次傳輸完成後再進行下一次的傳輸，再大量的影音資料傳輸也會造成大量的等待延遲，因此在本系統中影音串流的必須使用非同步傳輸之機制，採用 Windows 的 IO 完成端口(I/O Completion Port, IOCP)[4]的非同步傳輸架構實作非同步的串流傳輸使串流能夠順暢的播放，IOCP 只使用 Overlapped 的結構與一個 Completion key 當作識別管理傳輸項目，並不管理傳輸的資料，如圖二所示，因此使用 IOCP 的架構會占用記憶體直到傳輸結束，在系統處

Corresponding author:kwk@csie.ntut.edu.tw
DOI:10.6159/IJSE.2014.(4-2).55

理的過程中不可修改或釋放其資料內容，否則將得到不可預期的結果，但使用這樣的方式可以減少一次大量資料的複製減少系統的負擔，而提高系統的效率。因此 IOCP 使用時必須自行管理緩衝區，在 IOCP 傳輸的過程會交由作業系統完成傳輸，在傳輸之前配置記憶體空間直到傳輸完成時才能夠釋放。當系統傳輸完成時會將完成的傳輸項目存入完成佇列中，程式透過建立工作線程等待傳輸完成，工作線程中使用阻塞輪詢的方式等待完成佇列中是否有傳輸完成的項目，如沒有完成的項目將會被阻塞直到取得完成的工作項目為止，取得完成的工作項目包含傳輸項目的 Overlapped 結構與 Completion key 的資訊，必須使用這兩個資訊得知已被完成的傳輸，完成資料的傳輸並釋放記憶體要求。IOCP 緩衝區管理設計使用鏈結串列動態調整要求的空間，如圖三所示，管理記憶體要求與釋放的動作，當要求記憶體空間時會從 Head 開始依序從串列中取用記憶體，當記憶體釋放則將串列中的元素拆下放到 Tail，達到循環使用記憶體避免造成記憶體洩漏(Memory leak)的發生。



圖二： IOCP 操作概念圖



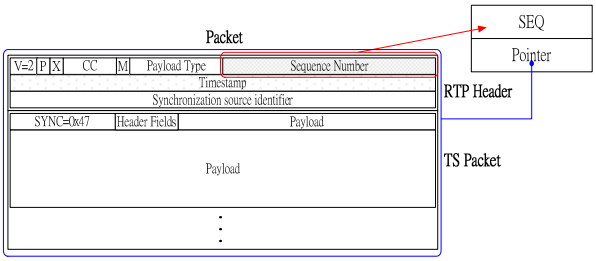
圖三：IOCP 緩衝區管理

三、 串流子系統

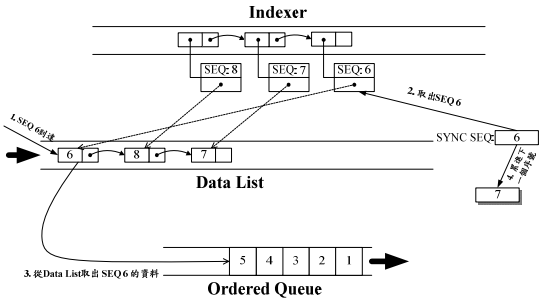
串流子系統中為了確保影音的品質，由串流源發送 RTP[5]的影音內容，由本系統接收串流影音的內容並轉送給其他的接收者，本系統使用 UDP 的方式傳輸 RTP 封裝的封包[6]，因為 UDP 並不保證資料的可靠性與順序，再加上本系統使用 IOCP 非同步的傳輸架構，使得封包順序不可靠而有嚴重錯亂產生的問題，因此為了確保影音的連貫性，在本子系統中將針對影音的順序與影音內容的緩衝進行處理，讓影音可以較順暢的播送而分為封包排序與串流緩衝管理兩個子程序。

緩衝管理子程序由一個資料串列與一個資料佇列組成，第一個資料串列為可以隨機取出的鏈結串列與第二個資料佇列為先進先出的佇列，當資料被接收時將會直接被放入第一個串列當中，經由排序子程序依序號順序放入佇列，當資料要被送出的時候則從第二個佇列取出。

排序子程序流程，由一個插入排序的串列組成，將存入接收緩衝區中的封包建立關聯索引，記錄第一個封包的序號，如圖四，並使用插入排序法放置到串列正確的順序，然後從串列尾部依序取出，當串列中取出的序號與同步參考的序號相同時則為正確的封包，此時則將封包放入第二個佇列當中等待資料發送，並且將紀錄的序號累進，用以預測下一個要接收的封包。並重新從串列的尾部取出比較，如不符合參考的序號時將不累進且將持續等待至正確的序號到來，當接收到正確的序號後則繼續執行排序子程序，如圖五所示。



圖四：紀錄 RTP 封包的序號與參照指標



圖五：記錄 RTP 封包的序號與參照指標

因 UDP 傳輸有不確定性，可能封包過久到達或遺失的情況發生，因此必須限制其範圍，確保不因為過久封包未到達而產生系統嚴重停擺等待的情況發生，但相反的，如果為了更快反應封包遺失的情況而降低等待的封包數，則會造成提早發生放棄封包的等待降低影音的品質。

四、 路由演算法子系統

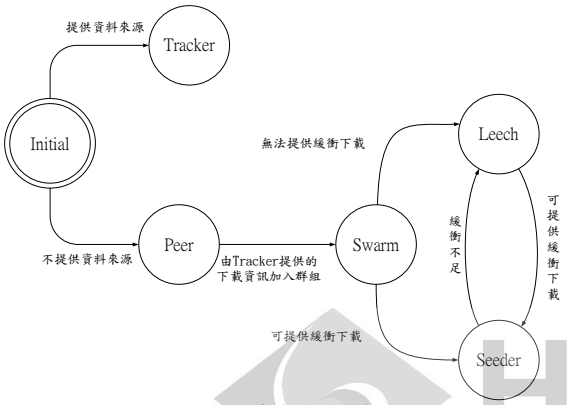
路由演算法子系統參考採用 BitTorrent[7]的 P2P 網路架構加以延伸，由於 BitTorrent 針對檔案傳輸進行設計，然而檔案與影音串流傳輸上有許多的差異，並無法透過原始的演算法達到最佳的串流傳輸之目的，為了達到最佳的影音串流傳輸效率必須修改其演算法符合串流

傳輸的環境，在本系統中針對串流傳輸修改了 BitTorrent 的機制，在 BitTorrent 中必須要有明確的資源大小才能夠完成細切的動作，在串流影音的傳輸無法表示串流的大小，本系統使用 RTP 封包的時間戳記作為串流的細切，並為每一個串流產生一組 ID，由時間戳記與 ID 這兩個參數紀錄每一個片段，當要求端送出要求時會與接受要求的 Peer 第一次的同步，即從被要求的 Peer 擁有的片段中時間最早的片段給要求者，為確保品質必須從最早的時間點同步，避免時間過於接近而來不及到達產生經常性的緩衝，在要求者取得同步時間後使用時間戳記發送給其他 Peer 要求片段，每個要求都經過一次時間戳記的偏移，一旦無回應且超過等待時間即以新的片段重新同步。

參考 BitTorrent 的機制與角色，修改應用於串流傳輸，將 BitTorrent 基本的機制策略修改為：

1. 最少的片段最優先(Rarest First)，原本為了讓檔案片段的副本數增加的策略，提高被存取的機率，在本系統中除了增加最少片段的還要考慮片段時間為最新的
2. 反饋機制(Tit for tat)，原本為了讓上傳與下載達到平衡的一種機制，將從其他成員接收的資料量決定傳輸的優先順序，並主動隨機發送提高其在其他成員的傳輸優先順序，串流傳輸中則考慮延遲時間，與成員在網路上存在的時間。
3. 終結機制(End Game)，為了在傳輸即將完成的時候將參與者挽留，由於成員即將完成時代表其擁有大部分的片段，當成員完成後立即離開將使得成員中擁有較多片段的成員減少降低可傳輸的來源，為了達到保留參與者數量的效果，但此一機制在即時串流中造成串流影音不順暢，因此本設計中不採用此一機制。

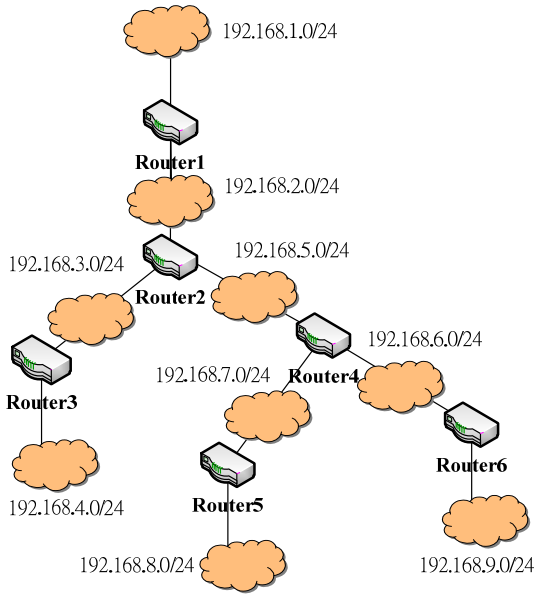
修改 BitTorrent 的角色，在系統初始狀態由是否提供來源資訊可分為 Tracker 與 Peer 兩種身分，Peer 加入至下載群組後成員 Swarm 的身分，並依緩衝是否能夠提供下載分為 Leech 與 Seeder 兩種身分，由於串流傳輸會有明確且唯一來源之特性，在本系統中來源可扮演 Tracker 的角色提供參與傳輸之成員，並採用如同 Torrent File 方式提供來源連線資訊。



圖六：角色演變狀態

五、 實作成果比較

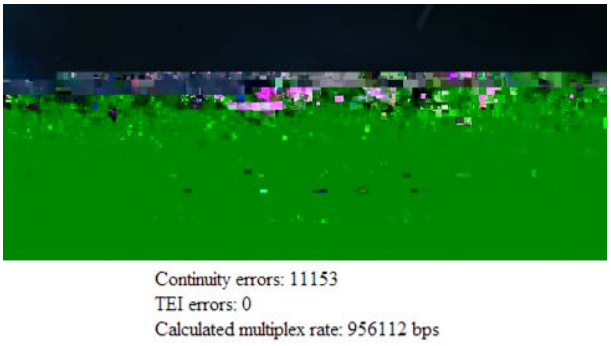
本實作之驗證之測試環境使用 6 個路由器建立 9 個網路區段進行測試，透過本系統進行串流之傳輸，如圖七所示。



圖七：網路拓樸

非同步傳輸與排序功能實作量測與比較，分為已排序、未排序、非同步傳輸與一般傳輸四種情況之比較，並使用 TSReader 工具量測比較傳輸錯誤的資料量。使用一般傳輸接收速度太慢會造成資料來不及接收而遺失的問題，順序則會造成解碼順序不正確而導致解碼失敗。四種情況之結果比較如下：

1. 一般傳輸且未排序

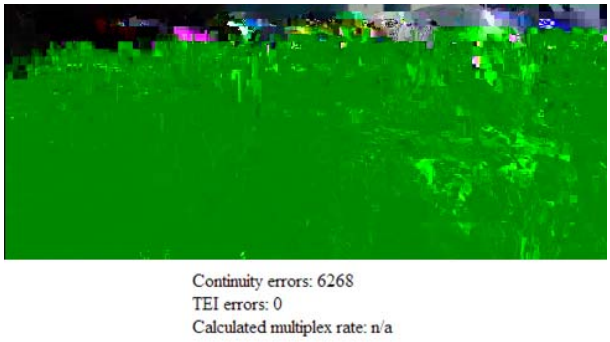


圖八：畫面與量測傳輸錯誤量

2. 一般傳輸且已排序

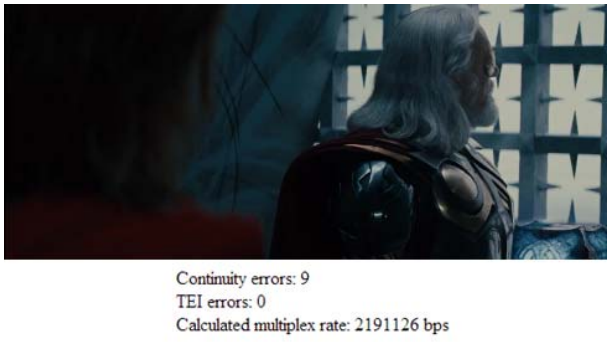
圖九：畫面與量測傳輸錯誤量

3. 非同步傳輸且未排序



圖十：畫面與量測傳輸錯誤量

4. 非同步傳輸且已排序

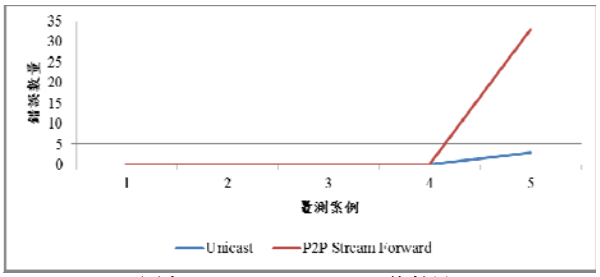


圖十一：畫面與量測傳輸錯誤量

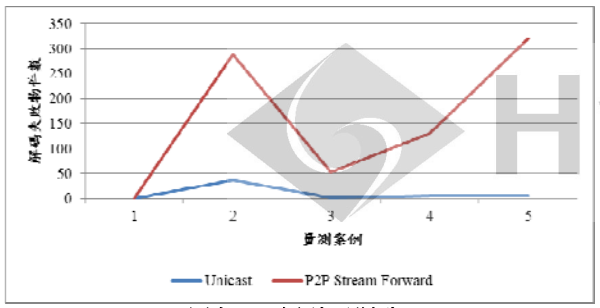
本系統設定來源資料量比較 Unicast 與本系統五種案例進行量測之傳輸效能，分別為一下五種案例之資料量：

1. 視訊 500kbps,音訊 128kbps
2. 視訊 1000kbps,音訊 128kbps
3. 視訊 1500kbps,音訊 128kbps
4. 視訊 2000kbps,音訊 128kbps
5. 視訊 3000kbps,音訊 128kbps

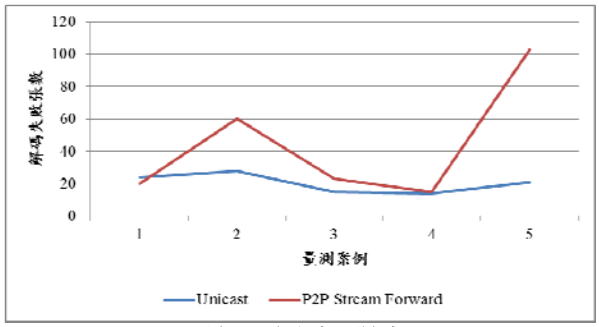
分別進行 TS 封包發生 Continuity Error 的數量、音訊解碼遺失與視訊解碼遺失進行量測，並統計計算平均之結果如下：



圖十二：Continuity Error 的數量



圖十三：音訊解碼遺失



圖十四：視訊解碼遺失

本測試針對本系統串流之壓力測試，量測之數據隨著使用位元率解碼錯誤的機率也逐漸攀升，其解碼也隨著位元率提高產生較多明顯的失真，但仍可順利播放與觀看，可透過提高緩衝區大小，滿足高位元率資料傳輸之需要，但提高緩衝區相對的會造成較長的時間延遲，必須在時間延遲與傳輸資料量之間取得平衡，本系統中設定之參數在資料位元率 1500kbps 或以下的情況可得到較為良好的品質。

六、 結論

本系統之研製為 P2P 串流網路傳輸進行分析，並設計較有使用彈性之串流系統，透過演算法的設計，與本系統之傳輸策略，達到提升網路傳輸能力之效，在本系統中參考 BitTorrent 之演算法時作拉式的 P2P 存取，完成並進行實際傳輸之驗證，透過串流傳輸分析工具量測其傳輸之品質，並實際由播放器驗證其播放之流暢。本系統主要之成果有：

1. 分析並研究現有之串流傳輸之設計，與影音內容網路傳輸之原理與方法。
2. 建構一個具演算法抽換能力，較有使用彈性之串流傳輸系統。
3. 使用 P2P 之演算法進行分散式之傳輸，提高傳輸之品質並降低骨幹網路傳輸資料量

此傳輸機制尚待後續在更大的網路上測試才能彰顯 P2P 共享機制之優點。

七、參考文獻

[1] Content Delivery Network, URL: http://en.wikipedia.org/wiki/Content_delivery_network
[2] T. Locher, R. Meier, S. Schmidt, and R. Wattenhofer. Push-to-pull Peer-to-Peer live streaming. In DISC, 2007.
[3] Winsock Reference, URL: http://msdn.microsoft.com/en-us/library/ms741416.aspx
[4] I/O Completion Ports, URL: http://msdn.microsoft.com/en-us/library/aa365198(VS.85).aspx
[5] Real-time Transport Protocol, URL: http://tools.ietf.org/html/rfc3550
[6] Boris Felts, Yuval Fisher Alex MacAulay, *IP Streaming of MPEG-4:Native RTP vs MPEG-2*, Envivio Inc., October 2005
[7] BitTorrent, URL: http://www.bittorrent.com/