使用 BT 技術建構隨選視訊傳輸系統

江秉璁、楊宏昌、李科鋒、曾黎明
國立中央大學資訊工程研究所
{earvin, cyht, kofeng}@dslab.csie.ncu.edu.tw, tsenglm@csie.ncu.edu.tw

摘要

隨著網路頻寬的成長，對於影音多媒體串流的需求也隨之增加，隨選視訊系統 (Video-on-Demand；VoD) 的發展被認為是目前網際網路應用的主流。由於影音多媒體串流的資料量大、頻寬需求高、持續時間長，所以有人提出固定耗用頻寬、無限制使用者且低等待時間的各種群播 (Multicast) 傳輸演算法。然而直到現在群播仍然無法普遍被應用，主要因素在於會增加路由器的控制負擔及計算複雜度。因此有人提出使用同儕網路 (Peer-to-Peer；P2P) 技術來利用網路上眾多使用者或同儕 (Peer) 協助傳輸影音多媒體串流，來減輕影音多媒體提供者網路能力與主機能力的負擔。

其中 BitTorrent (BT) 為同儕網路目前最熱門且有效的分散式檔案傳輸技術之一，因此本論文使用 BT 技術來建構隨選視訊系統。由於 BT 是針對傳輸一般檔案所設計，其片斷選擇 (Piece Selection) 方法並不適用於傳輸影音多媒體串流，本論文則是利用原有 BT 的架構另外提出較適合的片斷選擇方法，使得檔案未下載完成前就可以即時播放，並且可以平順且連續收看完影片，減少跳格的機會產生。另外由於實作上不會更動到 BT 的協定，所以可以相容於現有的 BT 環境，即來源 Peer 使用的軟體也可以是一般的 BT 軟體。

關鍵字：影音多媒體串流、同儕網路、BitTorrent、片斷選擇

1. 序論

隨著網路頻寬的成長，對於影音多媒體串流 (Media Streaming) 的需求也隨之增加，隨選視訊系統 (Video-on-Demand；VoD) 的發展被認為是目前網際網路應用的主流。由於影音多媒體串流的資料量大、頻寬需求高、持續時間長，所以有人提出固定耗用頻寬、無限制使用者且低等待時間的各種群播 (Multicast) 傳輸演算法，例如：Fast Broadcasting [1,2]，Recursive Frequency-splitting Scheme [3]及 Harmonic Broadcasting [4]。然而直到現在群播仍然無法普遍被應用，主要因素在於會增加路由器的控制負擔及計算複雜度。還有人提出使用內容遞送網路 (Content Delivery Network；CDN) [5]，但是影音多媒體提供者需要額外花費一筆金錢給內容遞送網路服務業者。因此有人提出使用同儕網路 (Peer-to-Peer；P2P) 技術來利用網路上眾多使用者或同儕 (Peer) 協助傳輸影音多媒體串流，來減輕影音多媒體提供者網路能力與主機能力的負擔。

同儕網路藉由使用者直接連接其他使用者的電腦，進行文件的交換與共用，讓個人電腦同時具備伺服器與終端使用者的功能，使線上數以萬計的個人電腦形成一種類似區域網路的共享平台，各個使用者間能夠彼此分享電腦中的運算、記憶體以及檔案等共享資源，成為另一網路環境型態，有別於傳統集中式系統架構。

其中 BitTorrent (BT) [6] 為同儕網路目前最熱門且有效的分散式檔案傳輸技術之一，並被稱為第三代的 P2P 技術。它跟傳統的 P2P 有兩大分別：(1)沒有中央搜尋系統亦沒有採用分散服務器來追蹤檔案的搜尋，因此在網路上不會像 Gnutella [7] 造成大量的搜尋流量；(2)採用多點對多點傳輸，傳統 P2P 雖然將伺服器的工作分流，但仍然未脫離單點對單點的下載方式，如果 Peer 一多就需要排隊下載，仍然會減低下載效率。但 BT 則善用了用戶在下載時沒用到的上載頻寬，在下載的同時亦進行上傳的動作。換句話說，同一時間的下載者越多，上載者亦越多。

本論文使用 BT 技術來建構隨選視訊系統，讓使用者隨時加入都可以從頭開始看影片。由於 BT 是針對傳輸一般檔案所設計，為了快速分享與下載，其片斷選擇 (Piece Selection) 方法會令已下載的部分非常分散，以致不適用於傳輸影音多媒體串流，因此我們將提出針對使用 BT 下載影音多媒體檔案時，較適合的片斷選擇方法。與一般 BT 採用的分散片斷選擇比較，所提出的片斷選擇法，可以讓影音多媒體檔案未完整下載前就可以即時播放，並且可以平順且連續收看完影片，減少跳格的機會產生。另外由於實作上不會更動到 BT 的協定，所以可以相容於現有的 BT 環境，即來源 Peer 使用的軟體也可以是一般的 BT 軟體。

2 相關研究

2.1 BitTorrent 原理

一般 BT 系統從發佈者發佈檔案到使用者下載檔案的流程為：(1)發佈者將要發佈的檔案做成 "torrent" 檔並且自己當 Seed，然後將 "torrent" 檔

The authors would like to thank the National Science Council of the Republic of China for financially supporting this research under Contract No. NSC 93-2213-E-008-037

發佈到 Tracker 上；(2)發佈者將 "torrent" 檔放在網站上供使用者下載；(3) Peer 下載欲下載檔案的 "torrent" 檔；(4)Peer 根據 "torrent" 檔內的資訊得知 Tracker 位置，接著向 Tracker 索取同樣下載這個檔案的 Peer 清單；(5)根據 Peer 清單向 Seed 與其他 Peer 溝通並且下載檔案。

在 Peer 跟 Seed 與其他 Peer 溝通後欲下載檔案時，由於是跟多個來源請求下載服務，因此會有跟哪一個來源請求下載哪一個片斷會有比較好的效率的問題產生，而 BT 的片斷選擇 (Piece Selection) 法 [8] 主要是要將不同的片斷盡快的分散到不同的 Peer 上，以增加 Peer 彼此之間分享的效率，詳細的幾點原則說明如下：

(1) 隨機第一個片斷 (Random First Piece)：盡快下載一個完整的片斷，使得可和其他Peer交換片斷。
(2) 完整優先 (Strict Priority)：一個片斷完整下載後，才可下載另一個片斷，這樣可以確認片斷的完整與正確性。
(3) 稀有優先 (Rarest First)：優先下載稀有的片斷，防止該片斷消失不見。
(4) 最後階段模式 (Endgame Mode)：同時跟很多 Peer 要求最後幾個片斷，避免跟很慢的 Peer 下載資料，影響到檔案完成時間。

此外，BT 會根據對方 Peer 提供我們的下傳速度給予適當的回報，對於提供較大下載速度的 Peer 則對他提供上傳服務 (Unchoking)，對於其它的 Peer 則暫時停止上傳動作 (Choking)。在任何時間，每個 Peer 都還有一個 "Optimistic Unchoking" 連線，這個連線並不管 Peer 的下載速度如何，藉由此連線有可能找到更適合的合作伙伴，而且對於剛加入下載的使用者，也才能得到下載的機會。也因為有這些傳送規則，可以知道 BT 系統中其它 Peer 加入或離開上傳資料給你的行列是一件頻繁的事情。

2.2 其他同儕網路影音串流的相關研究

2.2.1 On Peer-to-Peer Media Streaming [9]

這篇文章提出 OTS_{p2p} 與 DAC_{p2p} 這兩個方法，一個是分配片斷的演算法，另一個是為了讓整體系統成長較快的協定，與 Peer 選擇有關。但其並未考慮到實際環境中會有 Peer 的突然離開或者 Peer 的速度會改變等情形發生。

2.2.2 群播樹(Multicast Tree)相關方式

在 [10-14] 中提出根據不同的影片來源建立一擴張樹 (Spanning Tree)，樹的根節點為擁有該影片的 Peer。樹中每一個節點負責將資料傳送到它所有的子節點。當有一個新的 Peer 想要取得該影片時，首先由根節點開始往下找，直到找到某一個 Peer 可以滿足其所要求的頻寬時，便加入該樹並且成為該點的子節點，開始接收影片。此方法稱為應用層群播 (Application Level Multicast)。但其並未考慮到真實情況下，每個 Peer 所能提供的頻寬大小不一，甚至有可能需要多個 Peer 聯合起來才能提供足夠的頻寬來服務。

2.2.3 PROMISE [15]

此文章主要想法是將每個片斷經過編碼分成數個封包，然後根據提供下載的Peer之頻寬比例來分配封包數的比例，而且經過特殊的編碼，如果傳輸過程中有幾個封包遺失，影片還是可以順利播放。

2.2.4 P2P 網路電視

CoolStreaming [16] 和 ppStream [17]，這兩套軟體都是利用 P2P 技術來傳輸電視節目，網路電視與我們的隨選視訊不同在於網路電視不是什麼時候加入都可以重頭開始看影片，所以網路電視的重點是在如何選擇 Peer 和分配頻寬，而我們的隨選視訊系統還需要仔細考慮到片斷的選擇。

3 系統設計

3.1 系統設計分析

一般來說，利用 P2P 技術來傳遞影音多媒體檔案的系統會有下列特性：(1)Peer 的上傳頻寬是有限制的，可能會比影片播放速率還要小；(2)每一個 Peer 所能提供的上傳頻寬不一定會一樣；(3)Peer 會隨時加入或離開系統。所以一個影音串流服務必須要同時向多個 Peer 請求服務，因此會有下列問題產生：(1)選擇哪些來源 Peer 來服務；(2)跟哪一個來源請求下載哪一個片斷會有比較好的效率。

現有 P2P 技術中最接近能解決上面問題的就是現在相當熱門的 BT，當使用者越多時，BT 的系統能力就越高，並且由於 BT 的 Choking、Unchoking 的應用，所以使用者上傳越大，下載速度就可能越大，可以算是一種獎勵機制，不過由於 BT 技術原本是設計用來抓取任何檔案，並非針對 Streaming 所設計的，因此要將 BT 技術應用於 Streaming 系統上，就必須要更改某些地方，例如片斷選擇的部份。

使用 BT 技術應用於 Streaming 系統上，Client 端軟體一定不同於一般的 BT 軟體，因為針對影音檔案需要有適合的片斷選擇法，另外還要有播放影音檔案的功能。至於 BT Server 部分需不需要更動，是一個值得探討的問題，BT 的 Server 即所謂的 Tracker，主要的功用就是告訴使用者，目前有哪些 Peer 也在下載同樣的檔案或是這個

檔案有哪些 Seed，標準 Tracker 會給予隨機的 Peer 清單，至於 Streaming 系統上是否要修改 Tracker 給予 Peer 清單的方法呢？若是需要一個有品質保證的 Streaming 系統，則去更動 Tracker 是必要的，因為根據 Peer 清單內所有 Peer 剩餘的服務頻寬是否還可以允許服務使用者，來判斷是否讓目前的 Peer 加入 Streaming 系統，如果剩餘的 Peer 頻寬不夠，則回答新要求的使用者一個空的清單，不過接下來的問題就會變成，如何保證 Peer 所設定的上傳頻寬就是真正的上傳頻寬，以及一個 Peer 同時上傳給多個 Peer 時，如何保證和保持給予多個 Peer 當初所分配的頻寬大小。至於不更動 Tracker 而只更動 Client 端也是可行的，主要缺點就是頻寬無法保證，所以下載速度會有所變化，此外，還需要考慮到有 Peer 隨時離開與加入這種事情發生。我們的實作上不考慮去更改這部分，因為標準 BT 的 Tracker 給詢問者隨機的 Peer 清單，已經擁有負載平衡的功能，不會使得所有的 Peer 都只去跟某幾個 Peer 請求下載資料。

利用 BT 技術建構隨選視訊傳輸系統裡，影片發佈者將要發佈的檔案做成 "torrent" 檔前，會考慮到是否事先按照時間將檔案切成數個小檔案，這樣下載時可以從前面的檔案抓起，主要的考量是目前的播放器幾乎都是要檔案完整抓完才可以開始播，而且支援檔案未下載完整即可播放的軟體所支援的影音格式不多，所以假設將影片事先切割成數個小檔案，這樣就可以將已完成的檔案用支援該影片格式的播放器軟體來播放。事先切割的缺點則是，因為要使得切割出來的檔案都能單獨播放，所以每個切割出來的檔案都會依照格式在檔頭存有影片的資訊，所以切出來的檔案總大小會比原來的檔案大小來的大，而且這種額外付出 (overhead)，如果依照越短的檔案間隔去切，overhead 就會越大。如表 1 所示，我們測試了 MPEG-1 格式的影片檔，將這六個檔案大概每十秒切成一個小檔案，根據結果顯示，大概會有百分之五點多的 overhead，如果檔案較大時，會使得下載頻寬較小的使用者需要更多的時間才能完整的收看影片。因此，我們在實作上不考慮事先將檔案作切割。

表 1：檔案事先切割的 overhead

檔案大小(byte)	秒數	切割數	切割後總大小(byte)	overhead
43,709,792	250	25	46,138,372	5.56 %
41,534,528	238	23	43,451,828	4.62 %
221,105,360	1268	126	234,907,596	6.24 %
236,864,404	1328	132	249,588,304	5.37 %
449,061,872	2576	257	472,992,100	5.33 %
585,585,252	3359	335	616,984,816	5.36 %
			平均	5.46 %

由於 BT 是針對傳輸一般檔案所設計，為了快速分享與下載，其片段選擇 (Piece Selection) 方法會令已下載的部分非常分散，以致不適用於傳輸影音多媒體串流，因此我們在實作上，這部份是需要作修改的。我們將在下一節提出針對使用 BT 下載影音多媒體檔案時，較適合的片斷選擇方法。與一般 BT 採用的分散片斷選擇比較，所提出的片斷選擇法，可以讓影音多媒體檔案未完整下載前就可以即時播放，並且可以平順且連續收看完影片，減少跳格的機會產生。

3.2 片斷選擇法

我們在此針對 P2P 環境提出一個動態的選擇片斷方法，因此不只能用在 BT 上，還可以使用在大多數 P2P 系統上，由於此方法是動態的選擇，所以 Peer 忽然離開或加入新的 Peer 都不需要什麼特殊處理。以下是我們的動態片斷選擇法的分析與說明。

基本原理是根據 Peer 上傳的速度來決定跟它下載哪個片斷，而 Peer 速度是指先前此 Peer 的平均傳輸速度，會用先前的平均速度來決定有幾點理由：(1)Peer 設定的上傳速度不一定是真正最後的傳輸速度；(2)Peer 上傳速度是會改變的；(3)根據自己測的速度是最可信的；(4)任何 P2P 系統都不需要修改任何協定。所以接下來問題就可以分成兩類，第一個是向某個 Peer 發出第一次下載請求時，要如何選擇片斷？第二個是向已知先前速度的 Peer 請求時，如何選擇片斷？

假設已知速度情況下，最簡單的方法都是照順序要，即一開始就同時跟 Peer 清單中的所有 Peer 要片斷，跟第一個 Peer 要第一個片斷，跟剩下的 Peer 照片斷順序一直要下去，若是有任何 Peer 傳完前次分配的片斷或者有新的 Peer，我們就照剩下片斷的順序繼續要下去，直到傳完所有 Peer 為止，但是由於每個來源 Peer 的上傳速度的差異，所以會造成片斷下載完成的順序不是照順序的，假如順序較前面的片斷剛好是跟一個比較慢的 Peer 要求下載的話，就可能會影響到從頭連續播放的片斷即時到達率，即會發生跳格的情形。雖然有一種方法避免這種情形，就是設定限定期限，假設限定期限前，原本分配出去的某個片斷無法下載完成，就丟棄這個未完成的片斷，然後重新跟別的 Peer 要求，但是如果常常遇到這種情況，會造成浪費的下載動作。

而我們的方法是，當向某 Peer 發出第一次下載請求時，若是沒有辦法得知此 Peer 的上傳速度，就要先經過實測速度，所以我們採取先分配最後面的片斷給它，不管速度快慢影響都比較小，當抓完一個子片斷，就有速度的數據來當做選擇片斷的依據。

如果某個 Peer 之前已上傳過片斷了，就有先前速度的數據，接著我們採取門檻法與計算法並行來決定選擇哪些片斷給該 Peer。首先使用 Pipeline 的方式，對於速度快的 Peer，同時跟他們要求多個片斷，對於速度慢的 Peer，則要少點片斷。然

後根據速度和門檻值做比較來決定分配哪些片斷。門檻值指的是目前全部下載速度的平均值的 α 倍，高於門檻值的就從影片最前面的片斷按順序開始選擇，低於門檻值的就去計算最適合該速度傳輸的片斷。至於高於門檻值的不去計算最適合的片斷有兩個理由，一個是怕如果都用計算的，很可能造成最前面幾個片斷，因為都沒有任何一個 Peer 計算的結果適合他，所以都沒選擇到，另一個就是其實一個片斷通常不大，以 BT 為例大概是 16KB 或 32KB，所以對於速度快的 Peer 要完成一個片斷的時間都不長，所以可以不用特別去計算。

表 2 各符號的意義

符號	意義
Si	第 i 個 Piece，總共 m 個
B	影片播放速率（CBR 時）
t	每個 Piece 的播放時間
Lj	每個 Piece 的大小（VBR 時），總共 m 個
Pi	第 i 個加入服務的 Peer
bi	Pi 的上傳速度
A	全部目前正在傳輸的 Peer 的平均速度

表 2 是一些符號所表示的意義。整個演算法針對 CBR 與 VBR 影片可歸納成如下：

(A)假設 Peer P_i 的速度 b_i 接受下載請求且影片為 CBR 時，此時要分配哪一個 Piece 給它？

Case 1：$b_i \geqq \alpha A$

- 直接分配最前面未抓的 Piece 給它。

Case 2：$b_i < \alpha A$

- $t_i = (B*t)/ b_i$ (P_i 抓一個 Piece 所需的時間)
- $j = \lceil t_i / t \rceil$，則分配大於第 j+k 個的片斷給 Pi 較為適合，未開始播放時 k 為影片頭已連續分配的最後一個片斷，若開始播放後，則 k 為目前播放到的片斷編號。
- 上面兩式可合起來，$j = \lceil B / b_i \rceil$。

(B)假設 Peer P_i 的速度 b_i 接受下載請求且影片為 VBR 時，此時要分配哪一個 Piece 給它？

Case 1：$b_i \geqq \alpha A$

- 直接分配最前面未抓的 Piece 給它。

Case 2：$b_i < \alpha A$

- 求出不等式 $h \times t > \frac{L_h}{b_i}, 1 \leq h \leq m$ 的最小 h 值。
- 則分配第 h+k 個以後的 Piece 給 P_i 較為適合，且該 Piece 也要符合不等式，未開始播放時 k 為影片頭已連續分配的最後一個片斷，若開始播放後，則 k 為目前播放到的片斷編號。

圖 1 為我們的 CBR 影片片斷選擇法之範例，假設 α=1，其中會出現四個 Peer，不同的時間加入且 Peer 有可能傳完幾塊片斷就離去，這些 Peer 的上傳頻寬假設為固定分別為，P1=B/2、P2=B/4、P3=B/2、P4=B/4。

- 時間 0 時：P1 加入，P1 頻寬 B/2 大於目前的平均速度，所以跟 P1 抓取片斷 1。
- 時間 3 時：P2 加入，P2 頻寬小於目前平均速度，根據計算 j=B/(B/4)=4、K=1，所以找第 j+k 之後尚未分配的片斷，即片斷 5。
- 時間 4 時：P1 傳完前一個分配的片斷後，我們繼續跟他請下一個片斷，因為 P1 頻寬大於平均速度，所以跟他請求片斷 2。
- 時間 5 時：P3 加入，P3 頻寬大於目前平均速度，所以跟他請求片斷 3。
- 時間 7 時：P4 加入，P4 頻寬小於目前平均速度，根據相同的計算方法後，請求片斷 7 (即第 4+3 之後的片斷)。
- 之後同理類推。

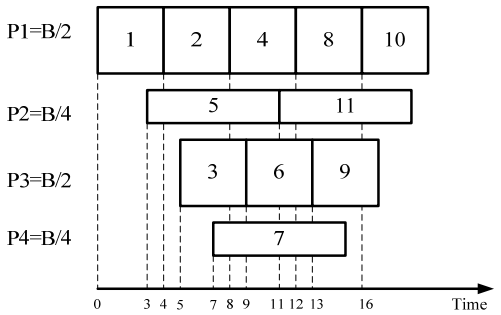


圖 1：片斷選擇範例

VBR 的影片適合於事先將影片依時間切割成數個檔案，而 CBR 事先切不切割都可以，事先切割的話，只要知道切割後每個檔案的播放時間與檔案大小，就可以使用我們的片斷選擇法來選擇要下載的檔案，因為對於速度慢的 Peer，我們會去計算適合下載的片斷，這時需要這些資訊，沒有事先切割的影音檔案，則必須另外想辦法知道整個影音檔案的每個時間點的 Bit Rate，另外預估目前已下載的部分可以連續播放多長時間與還需要多久再開始播放可以完整連續的播放，這也都是必要的資訊。

關於目前已下載的部分可以連續播放多長時間，就是去計算從頭開始有多少個連續的片斷已下載完成，再乘以每個片斷的播放時間所得到的值。而還需要多久再開始播放可以完整連續的播放的計算，就只能以目前的下載速度和已下載的情況去估計，因為下載速度的變化是影響這個時間的最主要的因素，而粗略估計的算法為 $\max(\frac{\text{剩餘未抓的片斷總大小}}{\text{目前的下載總速度}} - \text{影片長度}, 0)$。

4 效能分析

我們依照以下三點因素評估哪一種因素是影響 BT Streaming 可實現性最重要的因素。

1. Peer 提供的平均傳輸速度：下載整個影片過程中，整體平均傳輸速度。
2. 系統平均的 Seed 數目：下載過程中，每個時間平均有多少 Seed 在系統內。
3. 系統平均的 Peer 數目：下載過程中，每個時間平均有多少 Peer 在系統內。

而我們依照上面三點因素評估兩個對使用者來說比較重要的項目。

1. 整體片斷及時到達率：從開始播放影片到影片結束時，過程中有多少比例的片斷會在播放時已經下載完成。
2. 從頭連續及時到達率：從開始播放影片到影片結束時，連續播放多少片斷沒出現中斷的情形，佔全部片斷數的比例。

表 3：實際量測的數據

樣本	第一次得到 Tracker 回應時間(秒)	收到第一個片斷的時間(秒)	整體平均傳輸速度 (KB/S)	系統平均的 Seed 數目	系統平均的 Peer 數目
1	3	10	844.44	28.19	1.07
2	1	21	381.22	46.67	0.41
3	2	64	113.46	42.94	6.42
4	5	20	59.03	52.85	10.95
5	4	67	69.79	23.46	52.39

我們實際使用工具軟體下載網路上別人發佈的檔案，並且測量數據和請求下載的情況，表 3 為測試的五個樣本得到的數據，根據實驗數據，可以知道通常開始下載一個任務幾秒內就可以由 Tracker 得知 Peer 清單，而得到 Peer 清單後平均 30 幾秒後可以開始下載到第一個片斷，蠻符合之前的 Optimistic Unchoking 所敘述的，每 30 秒讓新的使用者 Unchoking。

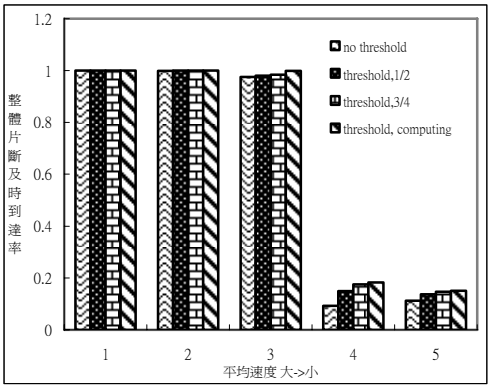
我們比較的片斷選擇法如下。

A. 依片斷順序選擇(no threshold)：根據速度選擇該 Peer 應傳幾個片斷，且按順序從第一個片斷開始分配。

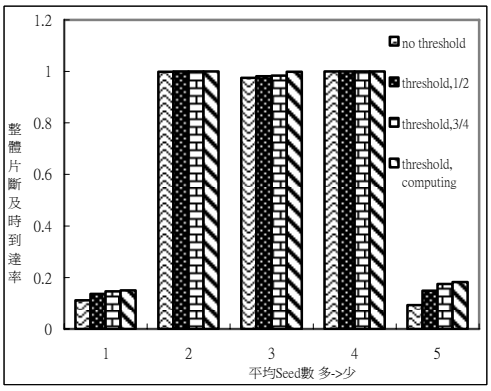
依速度門檻值選擇：速度大於門檻值則按順序從第一個片斷開始分配。

B. 小於門檻值則從影片 1/2 位置往後按順序分配(threshold,1/2)。
C. 小於門檻值則從影片 3/4 位置往後按順序分配(threshold,3/4)。
D. 小於門檻值則計算出最適合片斷後開始分配(threshold, computing) ，即我們的方法。

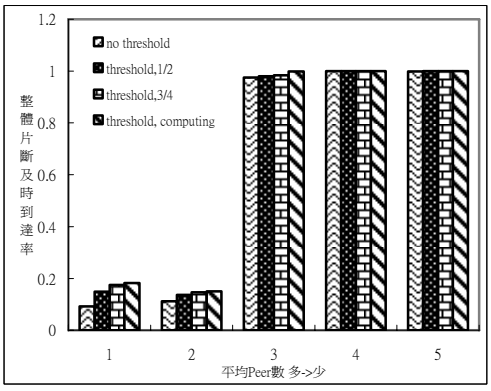
我們使用五個樣本的請求下載情況，假設 α=1，使用這四種片斷選擇法去評估分析整體片斷及時到達率與從頭連續及時到達率，我們分析的目標為切成 2000 塊片斷的影片，開始下載第一塊片斷後的 30 秒開始播放，然後分別計算出整體片斷及時到達率與從頭連續及時到達率，根據 Peer 提供的平均傳輸速度、系統平均的 Seed 數目與系統平均的 Peer 數目來排序，分析這三個因素何者為兩個及時到達率的關鍵因素。如圖 2 為整體片斷及時到達率的比較圖，圖 2(a)為根據 Peer 提供的平均傳輸速度排序有左到右為由大到小，例如最左邊的為平均速度最快的樣本 1，其依序為樣本 2、3、5 與 4，圖 2(b)則是根據系統平均的 Seed 數目排序，圖 2(c)則是根據系統平均的 Peer 數目排序。可以發現我們的片斷選擇法對於整體片斷及時到達率與從頭連續及時到達率皆有比較好的效果。此外，從圖中可發現，Peer 提供的平均傳輸速度是影響到整體片斷及時到達率的最主要因素。圖 3 為從頭連續及時到達率比較圖，我們可以發現一樣的結果。



(a)

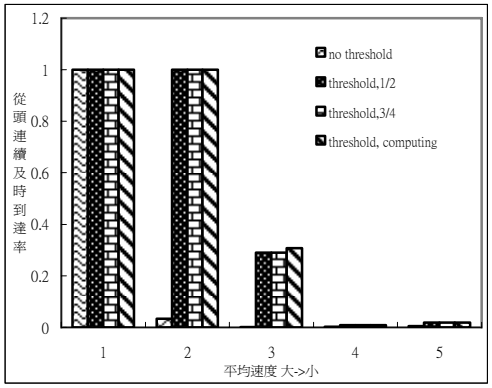


(b)

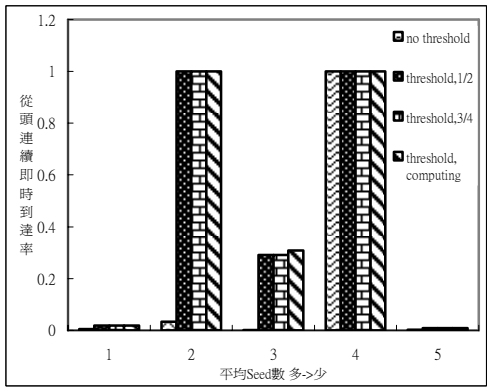


(c)

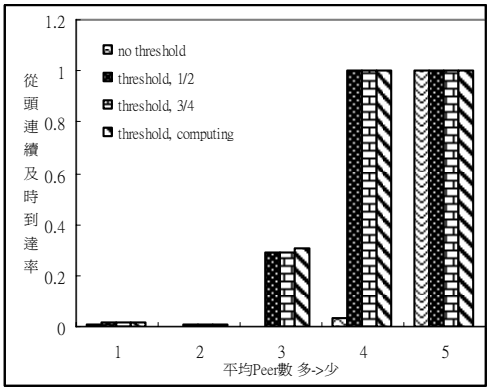
圖 2：整體片斷及時到達率比較圖



(a)



(b)



(c)

圖 3：從頭連續及時到達率比較圖

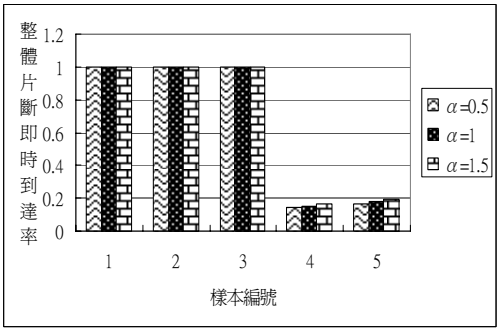


圖 4：門檻值設定比較圖

至於門檻值要設為平均速度的多少倍比較適合呢，我們使用 α=0.5、α=1 與 α=1.5，一樣使用前面的五個樣本來比較我們的片斷選擇法，比較結果如圖 4，發現 α 值改變整體片斷即時到達率變化不大，以樣本 4 與 5 來說，α=1.5 的整體片斷即時到達率略為高一點。

5 系統實作

實作軟體環境如下：

- 以 Azureus_2.2.0.2 [18] (Java Language) 為基礎來改。
- 搭配 VedioLAN Client (VLC) [19] 在影片未下載完前即時播放影片，且 VLC 可以解決檔案讀取被下載軟體鎖住的問題。
  - 某些影片需要先下載片頭才可以播，例如 MPEG-4 和 DivX，因為播放這些類型的影片需要一些檔頭資訊。
  - 支援影音格式：MPEG-1, MPEG-2, MPEG-4, DivX, mp3, ogg, ...
- 編輯與編譯工具為 eclipse [20]。
- 作業系統為 Windows XP

而隨選視訊系統的其他成員，如 Tracker 或其他 Peer 皆是現有的軟體，例如 Client 軟體有 BitComet [21]、ABC [22]、Shareaza [23] 等，而 Tracker 我們所使用的 Azureus 本身內建 Tracker 功能，所以我們直接使用他來測試。

實驗軟體再播放影片時會遇到兩個問題，一個是檔案讀取權限的問題，另一個是影片格式產生的問題。通常下載軟體在檔案未完成前會頻繁的產生寫入的動作，所以會把檔案設定為無法讓其他軟體寫入與讀取，所以會造成大部分的播放軟體無法播放影片，VLC 可以解決這方面的問題。另一個是影片格式的問題，以 MPEG-1 來講，只有抓到影片任何部分都是可以直接播放的，但是對於 MPEG-4、DivX 和 Rmvb 等格式的影片則必須要有檔頭的資訊才可以播放，所以這種情況下一定要先抓取檔頭。

以 MPEG-4 來說通常是 AVI 格式的檔案，而 AVI 算是 RIFF 格式的檔案，RIFF (Resource Interchange File Format)是一個二進位的巢狀結構檔案，它並不代表任何多媒體檔案，而它是把多媒體檔案包裝在其定義的結構之中，RIFF 中的基本單位就是 Chunk,如下：

```
typedef struct _Chunk
{
    DWORD ChunkId;              /* Chunk ID marker */
    DWORD ChunkSize;            /* Size of the chunk data in
bytes */
    BYTE ChunkData[ChunkSize]; /* The chunk data */
} CHUNK;
```

ChunkData 中又可包含別的 Chunck (Sub Chunk)，這樣的關係一直下去就如上所稱的巢狀結構。而 AVI 屬於如上述的 RIFF 格式，AVI 內擺的檔案資料包括了聲音與影像，它的 Chunk 結構如下：

```
struct _RIFF     /* "RIFF" */
{
     struct _AVICHUNK     /* "AVI " */
     {
          struct _LISTHEADERCHUNK     /* "hdrl" */
          {
               AVIHEADER AviHeader;          /* "avih" */
               struct _LISTHEADERCHUNK    /* "strl" */
               {
                    AVISTREAMHEADER StreamHeader; /*
"strh" */
                    AVISTREAMFORMAT StreamFormat; /*
"strf" */
                    AVISTREAMDATA      StreamData;     /*
"strd" */
               }
          }
          struct _LISTMOVIECHUNK    /* "movi" */
          {
               struct _LISTRECORDCHUNK    /* "rec " */
               {
                    /* Subchunk 1 */
                      /* Subchunk 2 */
                    /* Subchunk N */
               }
          }
          struct _AVIINDEXCHUNK    /* "idx1" */
          {
               /* Index data */
          }
     }
}
```

如上所列，這是一個 AVI 的結構檔案，主要 RIFF chunk 包裝一個 AVI chunk，AVI chunk 之中又包含了，LIST-HDR、LIST-MOVI、INDEX 三個 chunk，LIST-HDR chunk 之中主要是紀錄影音資料的資訊，LIST-MOVI chunk 是存放影音資料的地方，INDEX chunk 是紀錄影音長度的索引，可以用來提供影片快轉或倒帶的資訊，所以必須至少要有 LIST-HDR 的資訊，才能知道如何去播放這個影音檔案，而傳輸過程中假使 LIST-MOVI 內某些部分遺失掉，播放過程中就可能會出現某些畫面出現雜訊，但是如果遺失到 chunk 結構的部分，可能就需要藉由 INDEX 來修復，或者如果隨選視訊系統要有影片的快轉與倒帶的功能，我們可以藉由先抓取檔頭知道影片的格式等，然後再抓取 INDEX 部分知道影片的長度及每段畫面的位置，這樣就可以根據 INDEX 抓取使用者欲播放的畫面所在的片斷。

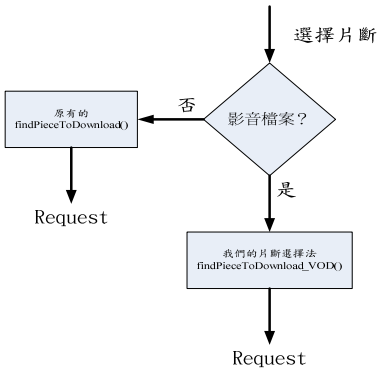


圖 5：實現流程圖

至於程式的修改如圖 5，片斷選擇的部分是改 PEPeerControlImpl.java 內的 findPieceToDownload() 這個函式，使得下載影音檔案時是使用我們的片斷選擇法 findPieceToDownload_VOD()，另外我們在 findPieceToDownload_VOD() 內有 check 前面的片斷是否已經分配給每個 Peer，但是超過 30 秒仍未有下載動作，這時我們就重新分配給其他 Peer。而 UI (User Interface) 的部分，我們新增"即時播放"這個選項，然後去呼叫我們另外寫的函式，這個函式包括計算影片長度等相關時間，另外單一檔案在未下載完成情況下要播放時，假如影音檔案格式為 VLC 所支援的，我們會呼叫出 VLC 來做即時播放，若單一檔案已經完全下載完成，則會使用系統預設軟體去開啟檔案，若影音檔案為多個檔案時，當第一個影音檔案完成後，則使用 windows media player 來做清單播放。

而清單播放是使用 ".wpl" 檔案方式，當執行 ".wpl" 檔， windows media player 會從清單第一個開始照順序播放。".wpl" 檔範例格式如下：

```
<smil><body><seq>
<media src='影音檔案 1 位置'/>
<media src='影音檔案 2 位置'/>
<media src='影音檔案 3 位置'/>
</seq></body></smil>
```

6 結論

隨著網路頻寬的成長，對於影音多媒體串流的需求也隨之增加，隨選視訊系統的發展被認為是目前網際網路應用的主流。因此有人提出使用同儕網路技術來利用網路上眾多使用者或同儕協助傳輸影音多媒體串流，來減輕影音多媒體提供者網路能力與主機能力的負擔。

本論文使用 BT 技術來建構隨選視訊傳輸系統，讓使用者隨時加入都可以在影音多媒體檔案未完整下載前就可以即時播放，並且可以平順且連續收看完影片，減少跳格的機會產生。由於 BT 是針對傳輸一般檔案所設計，為了快速分享與下載，其片段選擇方法會令已下載的部分非常分散，以致不適用於傳輸影音多媒體串流，因此我們的片斷選擇法採取了門檻與計算的方法，根據當時 Peer 的傳輸速度與整體平均傳輸速度動態的計算，並將真實下載情況記錄下後，模擬各種片斷選擇法的效能，可以知道我們的片斷選擇法對於整體片斷及時到達率與從頭連續及時到達率有比較好的效果。另外由於實作上不會更動到 BT 的協定，所以可以相容於現有的 BT 環境，即來源 Peer 使用的軟體也可以是一般的 BT 軟體。

此外，我們評估了幾個可能影響 BT Streaming 的因素，發現 Peer 提供的平均傳輸速度為最主要的因素，所以如何提昇 Peer 提供的平均傳輸速度就是 BT Streaming 可否實現的關鍵。以 BT 來

說，其特性就是下載者越多整體系統速度就可能會越快，並且使用者願意開越大的上傳速度就可能會得到越大的下載速度 (Tit-for-Tat)，已經有稍微的獎勵制度，不過 BT 無法保證使用者可以得到多少下載頻寬，所以可以朝幾個方向去解決。

(1) 如何更改 BT 技術，使得下載速度得到保證，然後下載速度收集不夠的使用者，就直到收集夠頻寬才開始播放。

(2) 怎樣讓使用者盡可能的提供上傳速度，除了適當的獎勵制度外，另外的方法就是使用專用的軟體，並且讓使用者無法設定上傳速度的大小，而是由軟體自己去測出最適合的上傳頻寬。

另外我們前面也都沒提到錯誤修正的問題，意思就是假設某一塊片斷已經分配給某個 Peer，但是快要播到那個片斷時，那個 Peer 還沒開始傳輸那個片斷，這時要怎麼解決這種問題，我們在實驗工具內是設定成靜態的錯誤修正，我們是檢查是否有某一個片斷分配出去，但是經過 30 秒還沒有開始下載，這時我們就會重新分配，不過其實可以改成動態的錯誤修正，即每秒去檢查是否有片斷在該送達的時間未送達，這時我們就直接請求速度最快的 Peer 來補送，至於該送達的時間是指需要播放時還未送來，所以越前面的片斷該送達的時間就越短。

參考文獻

[1] L.-S. Juhn, and L.-M. Tseng, "Fast broadcasting for hot video access", *Proceedings of the 4th International Workshop on Real-time Computing Systems and Applications, pp. 237-243*, Oct 1997

[2] L.-S. Juhn and L.-M. Tseng, "Fast data broadcasting and receiving scheme for popular video services", *IEEE Transactions on Broadcasting, vol. 44, no. 1, pp. 100-105*, March 1998

[3] Yu-Chee Tseng, Ming-Hour Yang, and Chi-He Chang, "A recursive frequency-splitting scheme for broadcasting hot videos in VOD service", *IEEE Transactions on Communications, vol. 50, no. 8, pp. 1348-1355*, August 2002

[4] L.-S. Juhn and L.-M. Tseng, "Harmonic broadcasting for video-on-demand service", *IEEE Transactions on Broadcasting, vol. 43, no. 3, pp. 268-271*, September 1997

[5] Peng, G., "CDN: Content Distribution Network", *Technical Report TR-125*, Experimental Computer Systems Lab, Department of Computer Science, State University of New York, Stony Brook, NY 2003.

[6] Official BitTorrent. http://www.bittorrent.com/

[7] Gnutella. http://www.gnutella2.com/

[8] Bram Cohen, "Incentives Build Robustness in BitTorrent", May 22, 2003

[9] Dongyan Xu, Mohamed Hefeeda, Susanne Hambrusch, Bharat Bhargava, "On Peer-to-Peer Media Streaming", *Proceedings of the 22nd International Conference on Distributed Computing Systems (ICDCS'02), pages 363--371*, Vienna, Austria, July 2002

[10] H. Deshpande, M. Bawa, H. Garcia-Molina, "Streaming live media over a peer-to-peer network", *Work at CS-Standford. Submitted for publication*, 2002

[11] M. Castro, P. Druschel, A.-M. Kermarrec, A. Nandi, A. Rowstron, and A. Singh, "Splitstream ： High-bandwidth content distribution in a cooperative environment", *Proceedings of IPTPS'03*, February 2003

[12] CoopNet. http://research.microsoft.com/~padmanab/projects/coopnet/

[13] Duc A. Tran, Kien A. Hua, Tai Do, "ZIGZAG: An Efficient Peer-topeer Scheme for Media Streaming", *Proceedings of IEEE INFOCOM*, April 2003

[14] Yang Guo, Kyoungwon Suh, Jim Kurose, and Don Towsley, "P2Cast: Peertopeer Patching Scheme for VoD Service", *WWW*, May 20–24, 2003, Budapest, Hungary.

[15] Mohamed Hefeeda, Ahsan Habib, Boyan Botev, Dongyan Xu, Bharat Bhargava, "PROMISE: Peer-to-Peer Media Streaming Using CollectCast", *MM'03*, November 2–8, 2003, Berkeley, California, USA.

[16] CoolStreaming. http://www.coolstreaming.org/

[17] ppStream. http://www.ppstream.com/

[18] Azureus. http://azureus.sourceforge.net/

[19] VLC. http://www.videolan.org/vlc/

[20] eclipse. http://www.eclipse.org/

[21] BitComet. http://www.bitcomet.com/

[22] ABC. http://pingpong-abc.sourceforge.net/

[23] Shareaza. http://www.shareaza.com/