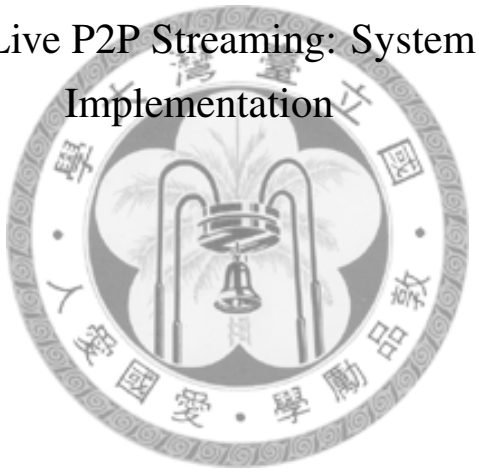
國立臺灣大學電機資訊學院資訊工程學研究所
碩士論文
Department of Computer Science and Information Engineering
College of Electrical Engineering and Computer Science
National Taiwan University
Master Thesis

異質性點對點即時串流轉播系統設計與實作

Heterogeneous Live P2P Streaming: System Design and Implementation



楊鈞羽
Chun-Yu Yang

指導教授：周承復 博士
Advisor: Cheng-Fu Chou, Ph.D.

中華民國 100 年 7 月
July, 2011

致謝

首先，感謝我的指導老師——周承復教授這兩年來的指導，使我漸漸熟悉學術的領域，從一個懵懂無知的大學生，直到培養出能夠完成碩士學位的能力。這兩年來跟著老師meeting，讀paper，做產學案，直到自己寫paper，讓我的能力一步一步向上累積，雖然中間有很多的失敗與挫折，但在老師的指導和鼓勵之下，使我度過重重難關，不論是成果或著是過程，都使我獲益良多。

再者，我要特別感謝論文口試委員——林俊宏教授、林嘉文教授、吳曉光教授和張英超教授所給予我在論文方向的寶貴意見與指導，讓我能開闊研究的視野，並有機會更充實此篇論文。

我也要感謝網路通訊與多媒體實驗室的所有指導教授——吳家麟教授、歐陽明教授、陳文進教授、莊永裕教授、陳炳宇教授和徐宏民教授提供我們良好的實驗室環境，可以從不同組的老師和同學身上學到更多元的知識。感謝我們組上的峻源學長、銘宏學長、靖茹學姊、智程學長、維世學長、銘德學長、庠宏學長等的指導，讓我能夠更快速且順利的進入狀況，步上研究的道路，你們也是我這條路上的榜樣。感謝同組同梯的夥伴們——俊瑋、耀鋒、佑逸、可柔、昱文、尚柏、羽晨、昱志，這兩年的日子，有酸甜，有苦辣，不論是喜是悲，大家一起度過，互相討論、期許與鼓勵，你們是我這兩年來不可或缺的夥伴。也感謝學弟妹們，銘遠、裕博、璽承、明鑫、逸安、家緯等，在這段時間內給我的幫助與鼓勵。

最後，我要感謝我的家人。感謝我的爸爸媽媽，在這段時間對我的照顧，研究的路途很辛苦，比較沒辦法好好陪伴和照顧您們。如今我完成了學業，感謝這些年來您們對我的支持與照顧，接下來是我要回饋您們的時候了。

謝謝你們。

中文摘要

隨著寬頻網路的蓬勃發展，越來越多人會上網觀賞即時的節目轉播。使用同儕式的方式，提供影音串流的服務，可以充分利用使用者的上傳頻寬，達到減輕中央伺服器的負擔，及使系統能容納更多使用者的兩大優點。但同儕式的環境下，我們無法保證每個使用者鄰接的同伴，能夠提供足夠良好的觀賞品質，甚至對於行動裝置的使用者來說，他們自身的下載頻寬不足以得到良好的觀賞品質。我們設計並實作出一個點對點即時影音串流系統，除了參考傳統的作法之外，我們考慮影音串流中資料不同的重要性，讓較重要的資料優先被服務，提升觀賞品質。而針對日漸普及的行動裝置，我們利用其具有多網路介面的特性，使用短距離的網路介面，與鄰近的裝置合作，讓彼此能互助互利，不但得到較佳的觀賞品質，也達到節省電量消耗的效果。

關鍵字：影音串流、觀賞品質、行動裝置、能量使用效率、多網路介面、系統實作

Abstract

With the fast development of broadband Internet access, more and more people watch streaming video over Internet. It is a hard task to implement a live P2P streaming system, which can satisfy the quality-of-service (QoS) of heterogeneous peers. We design and implement a live P2P streaming system with a QoS-aware mechanism, while peers exchange the video data, the more important data is served with higher priority. This mechanism improves the video quality of the peers in bad network condition. For modern mobile devices, they will suffer from the constrained energy supply and communication bandwidth. Since these devices have multiple communication interfaces, we develop an energy-efficient peer-assisted streaming system, peers can cooperative with nearby peers by short-range interface. It can reduce the energy consumption, and improve the video quality of mobile peers.

key words: video streaming, quality-of-service, mobile devices, energy efficiency, multiple interfaces, system implementation

Contents

List of Tables

List of Figures

Chapter 1

緒論

1.1 緒論

隨著網際網路的發達，越來越多人會上網觀賞現場直播的節目，包括像PPLive[7]、PPStream[8]等軟體，不但有現場直播的節目，也提供了一些隨選視訊(Video on Demand)服務。這些軟體使用的是點對點(Peer-to-Peer, P2P)的傳輸方式，讓眾多的網路使用者提供上傳頻寬，可減輕中央伺服器的負擔，並使這系統容易讓數以萬計的使用者加入，共同享受線上觀賞影音的服務。越是熱門的節目，因為能共享的使用者越多，就越容易得到良好的觀賞品質(Quality of Service, QoS)。 然而，在P2P的模式下，每一個使用者依靠的是一群觀賞相同節目的同伴，但我們永遠不能保證，我們隨時都能和一群狀況良好的同伴維持鄰接，並享受沒有缺陷的觀賞品質。先前有些研究指出[13][4][3]，一串連續的影片內容，裡

(a) PPLive (b) PPStream

Figure 1.1: 網路上常用的P2P影音串流軟體

面的資料有不同的重要性。在一個會丟失封包(packet loss)的網路環境，如果在資料的發送端，能優先讓接收者收到比較重要的資料，就可以提供較佳的觀賞品質。但這些研究並沒有討論到視訊和音訊對觀賞品質的差異。

除了考慮發送資料時的策略，接收端也可能有頻寬不足的問題。現今當紅的智慧型行動裝置，例如蘋果公司的iPad和iPhone，以及Android手機，這些裝置具有播放影音串流的能力，但是需要面臨不足且不穩定的3G下載頻寬，以及耗電量大的問題。但行動裝置和一般電腦所不同的是，它們除了行動網路介面(3G)之外，還具有支援短距離傳輸的WiFi及Bluetooth介面。只要用短距離傳輸介面，將鄰近的裝置連接成一個本地點對點網路(Local ad-hoc network)，彼此就能分享影音串流的資料，這個方式可以同時兼顧兩個目標：省電及提升觀賞品質。

1.2 貢獻

我們實作了一個即時同儕式影音串流的系統(live P2P Streaming System)，提供了Windows作業系統上的版本，可以在XP、Vista和win7上面執行，我們也製作了蘋果iOS上的版本，讓iPad、iPhone上能使用我們的程式，觀賞即時影音串流的轉播。 以下列舉這個實作系統的特色：

- 在使用者進行資料交換時，考慮了資料的重要性，讓接收者優先收到較重要的資料，藉此提升使用者的觀賞品質。我們的系統中在考慮資料重要性時，同時考慮了視訊和音訊的感受差異，這是過去的著作中沒有著墨到的。

- 針對行動裝置網路頻寬不足的問題，我們利用其具備多重傳輸介面，以及短距離介面較省電的特性，設計一個鄰近裝置間互助互利的合作機制，讓這些裝置能達到省電和提升觀賞品質的目的。

1.3 論文架構

論文的架構如下：第二章介紹相關的研究著作，第三章介紹我們實作的即時同儕式影音串流系統的細節，第四章介紹行動裝置上的版本實作細節，第五章是我們的成果展示，最後是結論及這個實作系統面臨的問題與限制。

Chapter 2

背景與相關研究

2.1 背景與相關研究

在2005年，X. Zhang等人發表了CoolStream[17]，提出一個同儕式影音串統系統的模型。這是一個拉曳式(pull-based)的模型，每個使用者會用某些方式取得線上使用者名單，然後隨機挑選一些同伴做為鄰居，藉由定時與鄰居交換擁有資料的狀態(data availability)，向鄰居發出自己所需資料的需求。這個根據資料驅動的點對點交換方式，相較於樹狀散播資料的同儕式架構，更能適應使用者進出頻繁的網路環境，較切合現實情況。

這個模型介紹了資料的交換方式，但沒有去考慮到串流中不同資料的重要性。已經有不少的研究指出[13][4][3]，考慮串流中資料的重要性，例如影片中常見的I、P、B編碼方式，或著是多層式編碼(layered coding)的影片，使用者必須先收到最下層的內容，再一層一層依據向上收，影片品質隨著上升，如果下層的內容丟失了，收到上層的內容也是解不回來的。於是在這個同儕式影音串流系統中，加入資料重要性，做為資料服務的參考，可以讓使用者享受到更好的觀賞品質。

而我們的研究中，加入了音訊和視訊的差異，做為觀賞品質的考量。我們發現人耳對音訊的破損是較敏感的，只要稍微丟失掉一些音訊的資料，我們很容易能感受的到；視訊相對比較不明顯，因為人眼在傳遞連續影像給大腦時，會發生視

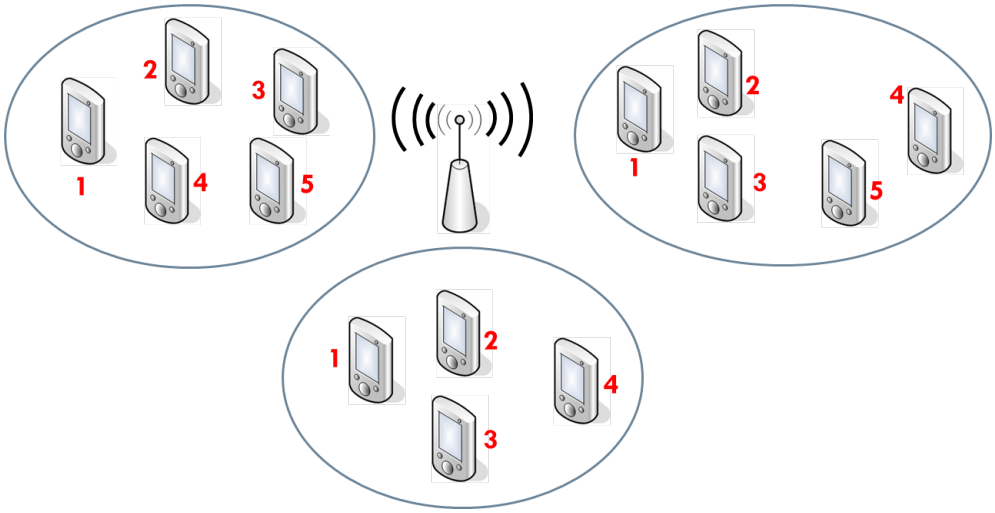


Figure 2.1: Frank H.P. Fitzek的情境

覺暫留，在每秒25到30張畫面組成的影片中，如果中間抽掉幾張畫面，人眼感受較不明顯。而且視訊前後的畫面通常相關性高，遺失一些資訊觀賞者還是可以知道影片發生的是什麼事。於是在音訊和視訊資料之間，我們會優先服務音訊的資料。

在行動裝置的部分，Frank H.P. Fitzek等人[5][2][14]針對具有多傳輸介面的行動裝置所組成的系統，提出了省電的方法。首先先點出行動網路介面(3G)比短距離傳輸介面(如WiFi、Bluetooth)還省電的事實，於是一群位置相近的行動裝置，可以形成一個群組，如圖2.1所示。在這個群組中，k個裝置互相協調好，每個裝置用3G下載k分之一的資料，再用短距離的介面分享，然後最後用模擬的方式驗證。他們所提出的合作模式，是在群組之中，所有人都互相連接得到，但在現實的環境中，不太可能一直維持這麼理想，總是會有A連得到B、B連得到C，但C在A的傳輸範圍之外這種情形。我們的研究另外考慮到下載工作的分配，如果能將下載工作交給目前剩餘電量較充裕的裝置，將可以延長電量較不足的裝置的觀賞時間。

而用多傳輸介面合作的方式，也可以用來提升行動裝置的觀賞品質。G.

Cheung等人的研究中[15]提出了CPR(Cooperative Peer-to-Peer Repair)的方式，在一個無線網路的多媒體廣播或群播(Multimedia Broadcast Multicast Services, MBMS)的環境下，當接收者丟失了資料，可以用短距離傳輸介面，向鄰近的裝置取得修復資料，可以避免造成中央伺服器的負擔，且提供更佳的觀賞品質。兩年後他們又提出網路編碼(network coding)輔助的方式[12]，達到更好的觀賞品質修復效果。不過CPR的方式，因為每台行動裝置還是預訂會下載所有的資料，所以就沒辦法做到省電，我們的方式則能夠兼顧省電及觀賞品質。

Chapter 3

主系統功能介紹(Windows版本)

3.1 概述

我們的主系統設計大致上是參考CoolStream[17]，這是一個拉曳式(pull-based)的點對點系統，每一個使用者會隨機找尋其他的使用者，建立鄰接的關係，形成一個隨機的網狀覆蓋式網路。每個使用者會和鄰居交換資訊，然後向鄰居拉取自己所缺少的影音串流資料。

在CoolStream這個模型之上，我們加入了增進觀賞品質的機制，當使用者回應鄰居的一連串資料請求時，可以依據那些資料的重要性，做為回應順序的參考。讓對方能優先取得重要的部分，即可增進觀賞品質。為了達成這個目標，伺服器在一開始將影音資料輸入這個系統中時，就需要分辨影音串流的內容，讀出其重要性，做一些相關的處理。這一章將要介紹這個主系統(Windows版本)內部的實作細節。包括了如何建立並管理鄰居關係、如何存放影音資料、如何與鄰居交換資料、如何依據資料重要性進行排程、以及如何處理原始的串流內容的五大單元。

3.2 鄰居關係的管理

每個使用者加入這個系統之後，需要維持一定數量的鄰居，使自己更容易取得所需的資料。這一單元介紹使用者如何尋找、建立及管理鄰居關係。

一個使用者新加入系統的流程，如圖3.1所示。伺服器會記錄目前系統中的使

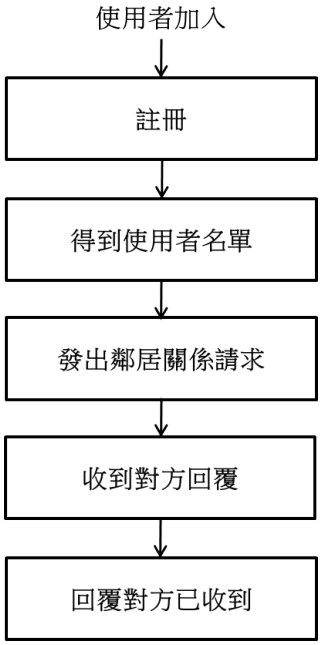


Figure 3.1: 使用者加入的流程

用者名單，包含網路位址及連接埠號。使用者加入後的第一件事情，即是向伺服器註冊。伺服器收到註冊的訊息之後，會將該使用者的資訊存入名單中，回傳給這位註冊的使用者[1]。這個需求使用者名單的動作，在使用者加入系統之後，每隔十秒鐘就會自動重做一次，一來可以取得更新的資料，也可以讓伺服器知道你還在這個系統中，而伺服器如果收不到某使用者的訊息，就可視為已離開這個系統。

使用者收到了來自伺服器的名單後，就可以從中挑選自己想要的鄰居。兩個使用者要建立起鄰居關係，一共會傳三次訊息，第一次是己端發出鄰居請求，第二次是彼端的回覆，己端收到時確認關係建立成功，第三次是再回覆給彼端，使彼端確認關係建立成功。目前使用者名單中只有記錄同伴的網路位置和連接埠，沒有其他額外的判斷資訊[2]，於是己端使用者在找鄰居時，是以隨機方式挑選。一個

[1]使用者名單內可能有很豐富的內容，於是我們一次註冊，伺服器只會從名單中隨機挑選十筆資料，十秒後該使用者重新註冊，伺服器會再給出十個名單。

[2]更詳細的資料可以包含該使用者的頻寬資訊、過去累積的上傳提供量、等等許多的歷史記錄，可以在伺服器端或是使用者端做篩選的工作。但這個部分不包含在我們的實作系統內。

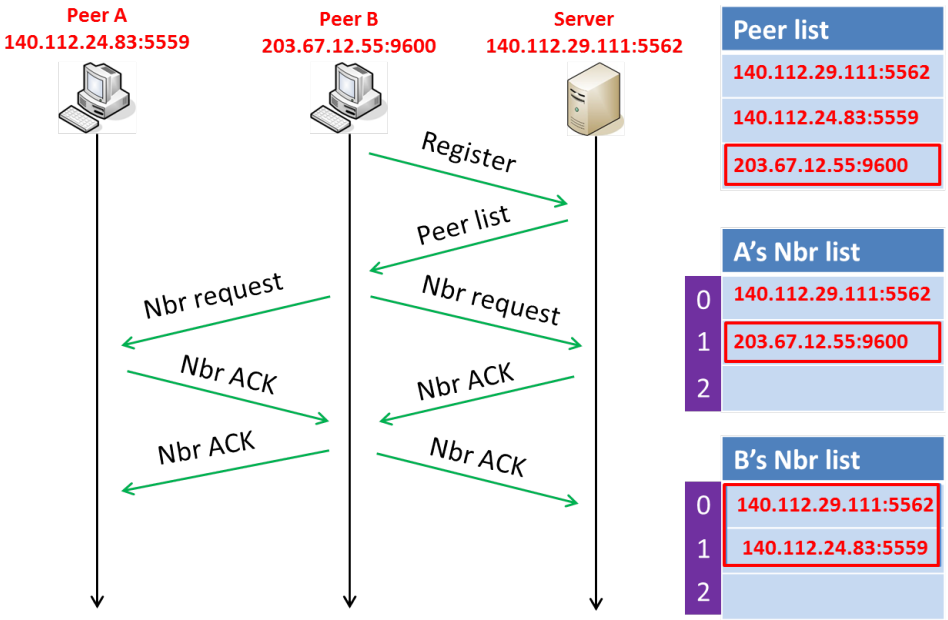


Figure 3.2: Peer B加入系統的情境

使用者最多可以有十五個鄰居，過多的鄰居會增加資訊交換的額外負擔[3]。被挑選到的同伴會被放入自己的鄰居名單中，接者即會發出建立鄰居的請求給該同伴。當一個使用者收到其他同伴的鄰居請求，會先判斷自己接不接受這分請求，目前的系統中，只要鄰居名單中還有空間，就會無條件接受，並回覆給該同伴。接著需要再回覆給該鄰居，這個鄰居關係才建立完成，接下來就可以和鄰居進行交換資訊，分享資料的動作了。

圖3.2顯示Peer B加入系統的情境，圖的左邊是訊息交換的流程，右邊則是使用者名單和兩個使用者的鄰居名單的變化，框框表示的是完成整個左邊的流程後，增加出來的部分。在我們的系統中，產生影音資料的伺服器也視為是一個一般的使用者，等待其他使用者和他連線需求資料。目前伺服器的鄰居名單上限也是十五個。

有關於鄰居名單的管理部分，除了每十秒鐘會向伺服器更新使用者名單之外，

[3]3.4節會介紹與鄰居交換資訊的細節內容。

另外還有以下兩個工作：

- 每秒鐘系統會檢查有無尚未完整建立鄰居關係的情形，如果有這種情形，會嘗試重新發出一份新的鄰居請求。如果十秒鐘過了還無法建立鄰居關係，則這個同伴會被捨棄，會從使用者名單中找尋其他的同伴代替。

- 如果有一個鄰居，在過去的十秒內沒有任何資料傳過來，則視為已經中斷連結，也會將它捨棄。(每一秒鐘鄰居之間會進行資訊的交換，在3.4節中將會介紹)

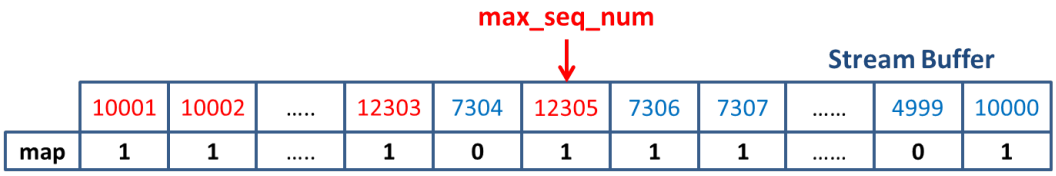
以上便是使用者管理鄰居關係的細節介紹，結束了這一部分，下一節要介紹的是如何存放交換得來的影音串流內容。

3.3 串流資料緩衝區

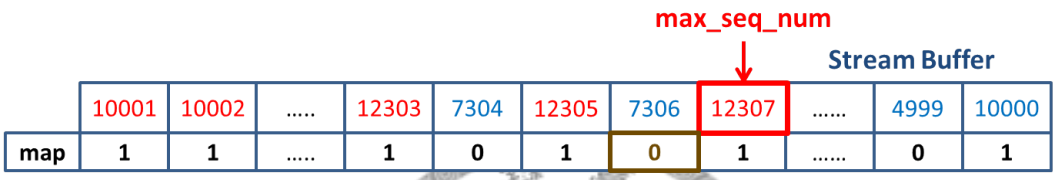
系統中需要有一個緩衝區(Buffer)，存放已下載的影音串流資料。我們將一個單位的串流資料稱為一個「資料塊」(Chunk)，一個資料塊大約是一千個位元組的大小，在後面的章節會仔細來介紹資料塊的內容。緩衝區的大小為五千個資料塊，為了方便了解，我們舉下面的例子說明：例如現在撥放的是位元率400kbps的影片，則這個緩衝區的大小大約相當於一百秒的影片長度。因為這是一個即時的影音串流，過期的資料捨棄即可。每一個資料塊依照產生的順序，系統會給一個編號，我們以該編號除以五千的餘數，決定將這個資料塊放在緩衝區的位置。如圖3.3所示，格子裡的數字就代表資料塊編號。

我們用「緩衝區圖」(Buffer Map)來記錄緩衝區的狀態。緩衝區圖記錄兩件事，第一是目前緩衝區中最大的資料塊編號，代表目前使用者所得到最新的資料。再者，我們用一個位元來表示緩衝區中每一個位置的情況，以最大的資料塊編號，往前算一個緩衝區的長度。如果該位置內存放的資料編號正好在這個範圍

Figure 3.3: 串流資料緩衝區(Stream Buffer)



(a) 第12307塊加入前的緩衝區圖



(b) 第12307塊加入後的緩衝區圖

Figure 3.4: 緩衝區圖的更新

內，表示該資料塊為「此處應存放的最新資料」，則將此位元值設為真，否則為假。舉例來說（見圖3.4(a)），目前緩衝區裡最新的資料塊編號為12305，所以緩衝區的每個位置裡，最新最理想的狀態，是存放編號7306至12305。只要該格裡存的正是在這範圍內的塊，就會將該格代表的位元設為真。由圖3.4(a)到圖3.4(b)顯示了編號12307的資料塊存入緩衝區的過程，首先因為12307為最新的編號，於是更新最大編號值，在更新的同時會將向前移動過的位置（即12306該格）的位元歸零，最後將12307的位置設為真。我們將更新緩衝區圖的步驟記錄在演算法1。

Algorithm 1 Update BufferMap(int received_chunk_num)

```
1: if received_chunk_num > max_sequence_num then
2:     update max_sequence
3: end if
4: A ← received_chunk_num / 8
5: B ← received_chunk_num % 8
6: set the B^th bit of map[A] to 1
```

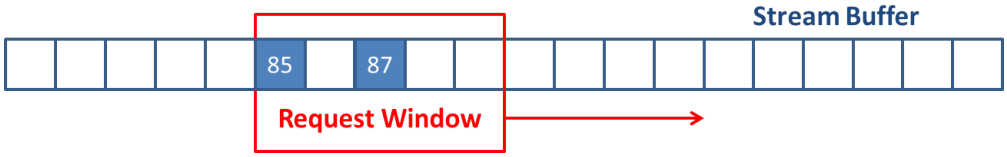


Figure 3.5: 需求窗格(Request Window)

3.4 與鄰居交換資訊

系統中的使用者，每一秒鐘會向鄰居發出請求，更新兩樣東西，分別是鄰居的緩衝區圖，以及自己缺少的串流資料。

3.4.1 更新緩衝區圖

緩衝區圖用來表示一個使用者的緩衝區狀態，於是鄰居的狀態就要透過鄰居的緩衝區圖來了解，有了鄰居的緩衝區圖，才知道可以和鄰居取得哪些資料塊。另外，根據鄰居的緩衝區圖，使用者會記錄目前自己所見到的最大資料塊編號值，用來做為之後需求資料的參考。

3.4.2 拉取串流資料

我們定義了一個「需求窗格」(Request Window)，長度為一千個資料塊（如果串流的位元率為400kbps的話，這個長度代表大約二十秒的影片長度），其範圍的上界為上小節記錄的，使用者所知的最大資料塊編號值。需求窗格的存在主要有兩個意義：第一，這個範圍用來做為影片播放前的緩衝，舒緩網路波動對觀賞品質帶來的衝擊。第二，這個範圍是接下來要播放的最新串流資料，是使用者會向外發出要求的資料塊範圍。隨著需求窗格向前滑動，而脫離範圍的資料塊會被送入播放控制單元，準備進行影片播放的動作。 如圖3.5所示。需求窗格隨著最大資料塊編號值向前滑動，而尾巴脫離窗格的部分會傳送給播放控制單元，舉例說明，當最大編號值從8000前進到8020，同時7001到7020這一段串流資料就會被傳給播放

控制單元。播放控制的部分後面會詳細介紹。如果某個資料塊在脫離需求窗格時，尚未接收到，就視為是資料丟失，丟失的資料即使之後再收到，也已經於畫質無補，因為它應該被播放的時間已經過去了。

現在介紹如何發出資料塊的需求。針對需求窗格中，每一個自己尚未收到的資料塊，從鄰居的緩衝區圖來尋找擁有者。如果有超過一個擁有者，我們隨機挑選其中一個鄰居。看完整個窗格後，將要送出的需求重新整理，每一個鄰居都會有一串需求名單，然後將名單傳出去。接下來就是等待鄰居的回覆，在我們的設計中，鄰居的回應會依據資料塊的重要性做排序，這部分的細節在下一節開始介紹。

3.5 以觀賞品質為基礎的排程

論文開頭已經說明了，在波動及變異性大的網路環境中，優先服務對觀賞品質而言重要的資料塊，能提升使用者的觀賞品質。這裡開始介紹在回應鄰居的一串資料塊請求時，我們的排程方法。

在原始的串流資料當中，有聲音和影像兩個部分，人類的視覺和聽覺，對於資料丟失的感受程度有所不同。視覺的部分因為人眼會視覺暫留，在每秒鐘20張畫面以上的速度，中間穿插掉了一些資訊，感受不一定會很明顯，且因為前後畫面的相關性大，觀賞者通常還是可以解讀影片，但是聲音就不具有這種特性，人的耳朵很容易聽出音訊有破損的情況。而且聲音資料所佔的傳輸量通常是比影像還要小得多，也就是說，讓使用者收到全部的聲音，是比較容易的。即使使用者對聲音和影像的重要度感受相同，也會因為聲音的觀賞品質較容易提供，於是我們給聲音較高的重要性，讓網路不好的情形下，使用者起碼可以在聲音上得到良好的品質。

此外，影像在進行編碼時，會產生不同重要性的幀（frame），影片中的一

組影像，會有主要的幀，及其他次要等級的幀，例如在各種影像編碼中常見的I、P、B分級，次要的幀需要參考上一層重要性的幀才能解碼出來。或著是多層式編碼(layered coding)的影片，使用者必須先收到最下層的內容，再一層一層依據向上收，影片品質隨著上升，如果下層的內容丟失了，收到上層的內容也是沒有用的。

基於上述這樣的特性，在網路頻寬不穩的情形之下，如果能讓使用者優先收到較重要的資料，對於觀賞品質的傷害就會越小。以此為原則，我們在回應鄰居發出的資料請求時，會根據每個資料塊的重要性做排序，先傳出重要的資料塊，來達到改善使用者觀賞品質的目的。下一段會介紹我們如何將串流資料包裝成一個個的資料塊，並且決定它們的重要性。

3.6 影音資訊的解讀與封裝

為了能將不同重要性的串流資料分開來，我們需要先能解讀串流的格式內容，才能以此為依據判斷資料的重要性。如果選用不同的串流格式，或著是用不同的軟體進行轉碼，會有不同的解讀規則。接下來將介紹我們所選用的格式，從解讀、封裝到解封裝，每一個步驟一一做說明。

3.6.1 解讀

我們的影片來源，是在伺服器的電腦上，安裝電視擷取卡，從天線端接收數位電視的訊號，再藉由VLC[9]這個軟體進行轉碼。我們將串流轉碼成我們讀得懂的格式，視訊編碼器我們用的是為H.264[6]，音訊用的是AAC(Advanced Audio Coding)[1]，整個的封裝則是用MPEG transport stream(TS)[16]。影片從最原始的串流資料，直到VLC端包裝完成的過程，如圖3.6所示。編碼器產生出的串流，其中的每一個單元會被包裝成PES(Packetized Elementary Stream)的封包，然後再切割

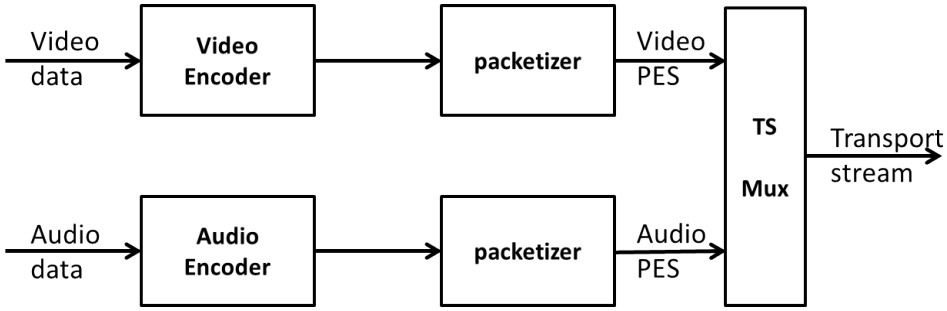


Figure 3.6: 影音串流的封裝

包裝成長度為188位元組的TS封包[16]。我們的伺服器接收來自VLC的TS串流，然後開始進行一連串的解讀及再封裝工作。

判斷視訊和音訊的每個單元，是在圖3.6的PES層上進行，所以我們需要將從VLC接收到的串流，把每個PES封包分辨出來。而分辨的步驟也應由外而內：從TS封包將影音和影像流分開，再讀出兩條串流裡的每個PES封包。接下來介紹處理的細節。

TS封包層的處理

從最外層的TS封包開始講起。每個TS封包的長度固定為188位元組，前面的四個位元組為TS封包的標頭。伺服器要做的第一件事情，是將串流資料中的視訊和音訊分離。TS標頭中有13個位元，是用來表示這個TS封包的ID，不同的ID即代表不同的串流，於是我們可以從這個ID的資訊，將視訊和音訊的兩條串流分開。

其實在一開始的時候，我們無從得知視訊和音訊分別是哪個ID，還需要下面一層的輔助，只要確認視訊和音訊的ID之後，就可以在TS封包這一層，將影音兩條串流分開。

PES封包層的處理

根據MPEG的定義[16]，PES封包前面會有個PES header，而PES header的前面還有一個MPEG header，MPEG header的最開頭三個位元組固定是0x00、0x00、0x01，

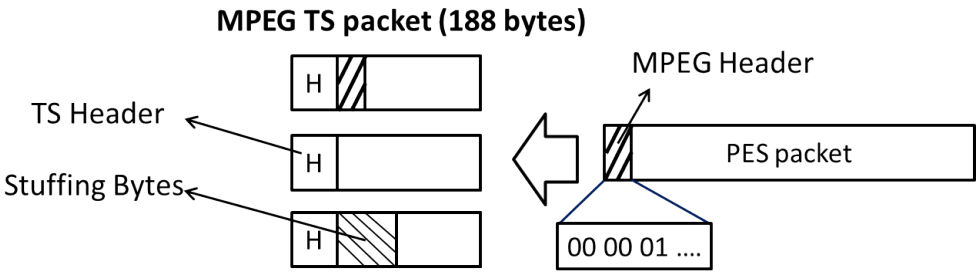


Figure 3.7: PES封包與MPEG TS封包

而第四個位元組是Stream ID，視訊的話一般是0xE0，而音訊則是0xC0[4]。PES封包和MPEG TS封包的關係，如圖3.7所示。圖3.7的右邊是一個PES封包(前面包含了PES和MPEG的兩個header)，會被TS封包封裝起來，而最後一個TS封包會有未被填滿的空間，會用填充位元組(stuffing bytes)填補滿。於是我們每讀到一個TS封包，只要檢查TS標頭後面的三個位元組，如果值分別是0x00、0x00、0x01的話，這就是一個PES封包的開頭。

因為我們可以在PES這一層，辨別視訊和音訊，只要同時對應到上一層的TS封包，做一個比對，就可以確認視訊和音訊的TS封包ID，之後我們就可以在TS封包這一層，將視訊和音訊拆成兩條獨立的串流。截至目前為止，我們已經可以將每個PES的單位分辨出來，並得知是視訊或是音訊。

視訊內容的分類

音訊的部分，每一個PES封包沒有重要性的區別，但在視訊的部分，編碼器產生的串流資料本身就有不同的重要性。圖3.8是視訊編碼中很常見的IPB編碼方式，我們用的H.264也有類似的區分。每一個視訊的PES封包，對應到H.264視訊流格式中的access unit。以下我們將介紹判斷影片內容的方法，而這個方法是依據VLC軟體轉碼出來的串流內容，如果用不同的軟體或方式產生影音串流，或許

[4]這個部分根據MPEG header的定義，視訊的Stream ID範圍是0xE0 - 0xEF，音訊則是0xC0 - 0xDF，而我們從VLC轉碼出來所見到的都是0xE0和0xC0

Figure 3.8: 常見的IPB視訊編碼

看到的串流內容會不相同。

從VLC產生出來的H.264視訊資料，我們看到的access unit可以分成三類：IDR(Instantaneous Decoding Refresh)、P slice、B slice。IDR是一串Code Video Sequence的開頭，而這串連續的影像可能長達數秒，若沒收到IDR對整串視訊會有很大的傷害。P slice則是第二重要，再來是B slice。IDR access unit裡面會先存放Parameter Sets與Supplemental Enhancement Information (SEI)，內容各自為整個壓縮視訊序列的參數，以及影片簡介、版權宣告等等使用者自行定義的資料，後面接著的是這個IDR access unit的畫面部分，使用的是I slice。而P slice和B slice的封包前面就沒有其他資訊，直接存放slice的內容，P slice和B slice的分辨方式，需要找到slice header，裡面會記錄一個slice的型態。於是我們可以讀出這個access unit是屬於IDR、P、B中的那一個種類，而內容的格式詳細記載在H.264的文件之中[6]。

最後我們排出來的重要性是**IDR、A(audio)、P、B**。IDR幾秒鐘才會出現一次，但是它很重要，因為會直接影響幾秒內的視訊畫質，而聲音的重要性排在其他影像封包的前面。

3.6.2 封裝

接下來要將原本的串流，重新再包裝，成為這個系統中使用者互相交流的資料塊(chunk)。圖3.9是伺服器將串流資料再封裝的流程，同樣重要性的資料會被放在同一個暫存區，暫存區長度設為一千位元組，即相當於緩衝區資料塊的大小。當

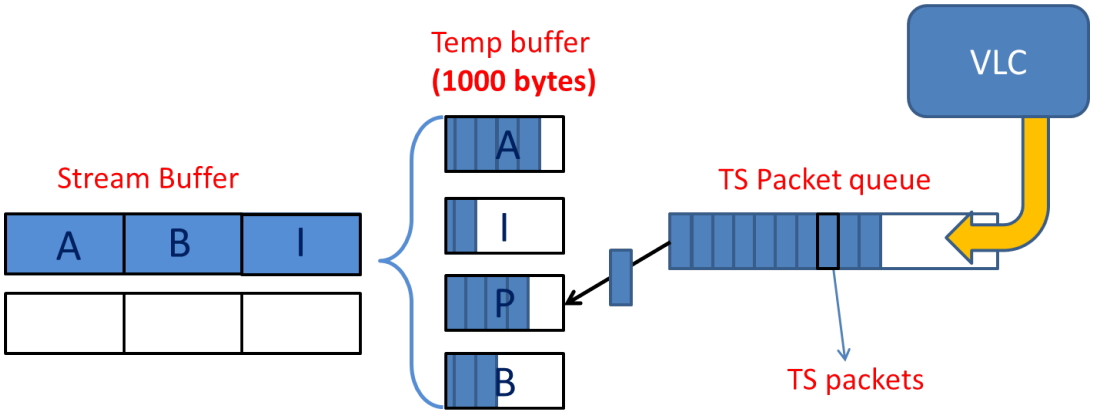


Figure 3.9: 再封裝的進行流程

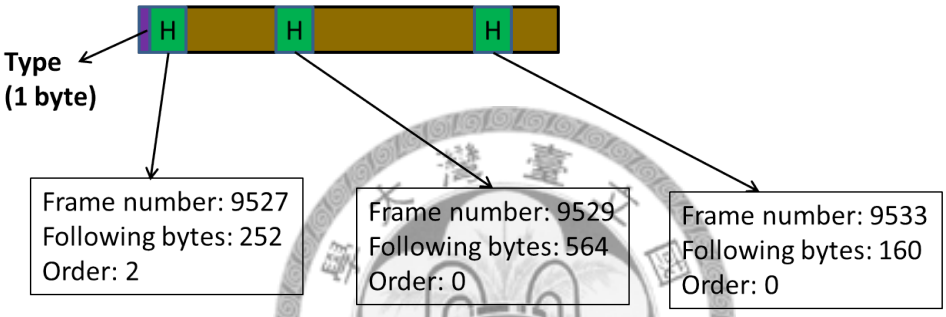


Figure 3.10: 資料塊的結構與內部標頭

這個暫存區存滿資料之後，會給他一個編號，編號的順序同時代表這個資料塊被產生的順序，還有標上他的型別（聲音、IDR、P、B），然後存入緩衝區中，一個新的資料塊即產生完成。[5]在資料塊產生的過程中，TS封包有可能被切割到不同的資料塊裡，不過我們之後會將資料重新組合回來，不傷害到播放時的觀賞品質。資料塊的內部細節介紹如下：

- 一個資料塊的長度，實際上是1001個位元組，除了1000個位元組的資料內容之外，最前面會用一個位元組來表示這個資料塊的類別，他可能是聲音，或著是影像的IDR、P、B其中一種。

[5]因為IDR在這影片中有絕對的重要性，不可能將一個IDR單元的一部分留在緩衝區中，等下一個IDR進來塞滿緩衝區再傳出去，於是IDR的部分我們不會累積在緩衝區中，在確定一個完整的IDR都收到之後，就會直接傳送出去。

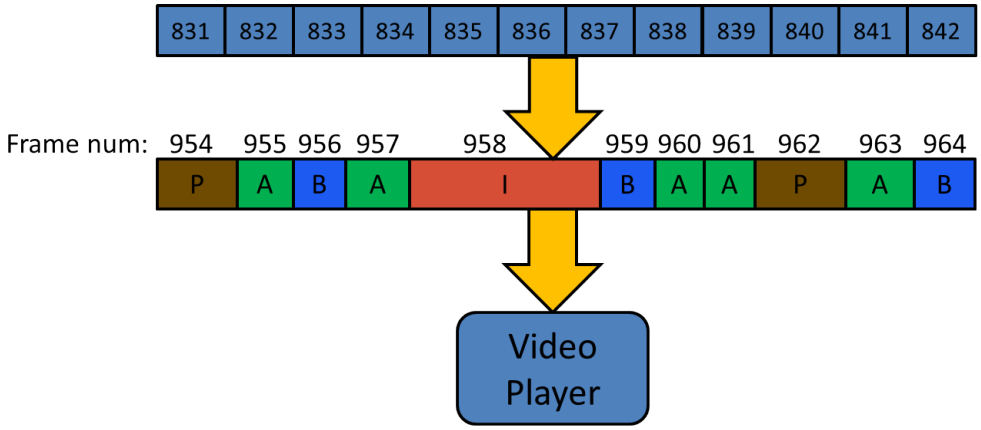


Figure 3.11: 影音串流資料的重組

- 資料塊內容可能包含一或多個片段，每個片段分屬於不同的PES封包，而比較大的PES封包為了放入資料塊，也可能被切割成好幾個片段。圖3.10顯示了資料塊的內部結構，為了標示每一個片段資料的內容細節，在片段之前會加入一個標頭，標頭記錄三個資訊：該片段所屬的PES封包的編號(frame number)、片段資料長度、此片段該PES封包的第幾個片段。

播放控制單元會重組出原始的影音串流，使播放器能順利播放整個影片。下一小節介紹解封裝和重組的過程。

3.6.3 解封裝與重組

在使用者的播放控制單元，收到從串流資料緩衝區(Stream Buffer)送進來的資料塊，就可以根據上小節介紹的資料塊內容格式，將原本的影音串流解回來，如圖3.11所示。在資料沒有丟失的狀態下，解出來的影片會是品質無損的完整影片，但若是發生了資料丟失的情形呢？

因為MPEG transport stream是以每188個位元組的封包做為單位，於是我們會將我們能解出來的TS封包解出來，再傳給影片播放器。如圖3.12所示，一個PES封包會被包裝在數個TS封包裡，一個資料塊丟失可能會使中間的TS封包丟失或收不

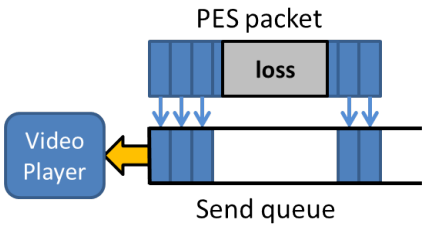


Figure 3.12: 資料塊丟失的情況

完整。收不完整的部分我們會直接捨棄，因為播放器需要辨識每個完整的TS封包。而尚完整的TS封包，我們會維持原本的順序，將它們傳給播放器。[6]

[6]其實我們並不確定播放器收到部分的PES封包可以被解碼，但我們就是盡量將我們看得到的完整部分傳遞給播放器，剩下的就要視播放器的播放能力而定了。

Chapter 4

行動裝置版本介紹

4.1 環境簡介

我們實驗所選擇的行動裝置是iPad第一代，開發在iOS v4.2版本上，只要是蘋果的iPad或iPhone都可以執行我們的程式[10]。我們將Windows版本上的功能完全寫入這個版本裡，讓iPad連上網際網路之後，可以和一般電腦的使用者一樣，加入這個即時影音串流系統，尋找其他鄰居使用者，交換資料等等。除此之外，iPad同時會用短距離傳輸介面，尋找周圍同樣在這系統中的無線裝置。

我們要利用短距離傳輸介面的輔助，讓行動裝置之間進行合作，然後降低電量的消耗，以及提升觀賞品質。接下來會從我們的研究動機開始介紹，然後再講解我們如何達成省電和提升觀賞品質的目的。

4.2 研究動機

像iPad、iPhone這些行動裝置，和一般使用者的桌上型電腦有不同的特性。首先是網路頻寬和傳輸介面的部分，現在的行動裝置，一般來說都具有行動傳輸介面(如3G、3.5G)，以及短距離傳輸介面(如WiFi、Bluetooth)。前者只要在有手機訊號的地方，都可以連接上網際網路，但是頻寬很小，而且訊號不穩定，在比較尖峰的使用時段，有時連想要直播約200kbps的即時串流，都是很困難的事。後者主要是支援短距離的傳輸，頻寬可能稍大一些，可以和其他的行動裝置建立連

Figure 4.1: 在iPad上執行P2PStream

介面	耗電量
3G	83
WiFi	18
Bluetooth	1

Table 4.1: iPad上不同傳輸介面的耗電比例

線，並交換訊息，其中WiFi的傳輸距離和頻寬都比Bluetooth強，但耗電也稍微多一些。

行動裝置另一個重要的特性，就是電量的限制。觀賞即時影音串流對行動裝置來說，是很耗電的一件事，我們可以將耗電來源分成三個部分：螢幕顯示、影片解碼的運算和網路傳輸。其中前兩個部分的耗電是無法避免的，這是行動裝置先天的弱點，可能只能期待未來的技術來克服，但在網路傳輸的部分，我們可以使用較省電的方式來傳輸資料。

表4.1是不同的網路傳輸介面，在接收相同資料量的情形下，耗電量的比例。我們看到了，在傳輸上所造成的電量耗損，3G比起WiFi和Bluetooth增加了許多的

耗電量，尤其Bluetooth展現了優異的省電效能。這個結果顯示，如果我們能減少3G下載的使用量，用短距離傳輸介面代替，可以達到省電的效果。若要利用短距離傳輸介面，就代表需要和位置鄰近的其他行動裝置連線，互相分享資料，才能達到這個效果。

隨著智慧型行動裝置越來越發達的趨勢，不論是在尖峰時刻的捷運火車上，或是下午晚上時段的都市咖啡屋，我們越來越容易在我們的周遭，看見智慧型裝置的使用者。因此，我們很容易能在附近，找到有相同興趣的行動裝置使用者，可以用短距離的傳輸介面和他們連線，運用我們所設計的合作機制，享受觀賞即時節目轉播的服務。

4.3 細部介紹

接下來我們將開始介紹我們如何在iOS上，提供即時影音串流的服務，並介紹我們的合作機制。

4.3.1 短距離網路介面與本地端網路

在iOS開發的資源庫[10]中，提供了一個叫GKSession的元件，用來建立本地端點對點網路(local area ad-hoc network)。GKSession在初始設定時，我們會給予一個固定的sessionID，而系統會產生自己的peerID（peerID在每次執行時都是不一樣的）。開始運作後，便會用短距離網路介面(即Bluetooth和WiFi)[1]，尋找在連線範圍內，有無其他的裝置，和自己擁有相同的sessionID[2]。如果有能連接到的裝置，彼此就能用peerID互相辨識，在我們的設定中，我們會自動和所有連線範圍中的裝置建立連線關係。

關於鄰近裝置的偵測，都是用iOS底層自動進行，我們沒有辦法操作，只能從

[1]使用者需要自己將iPad的WiFi或Bluetooth打開，系統不會主動幫我們做這些事。
[2]如果雙方的WiFi和Bluetooth都開啟，GKSession會優先選擇Bluetooth

系統的通知中，知道現在有哪些裝置在連線範圍內，以及目前有無和該裝置進行連線。在裝置會移動的狀態下，系統會將與其他裝置關係改變的通知傳到應用程式層上，我們因而知道現在有誰在我們的周圍，以及誰離開了。

4.3.2 程式流程

iPad上的程式流程其實很簡單，如演算法2所示。一開始的時候，iPad會先加入這個即時影音串流系統，同時將GKSession啟動，尋找周圍其他的裝置。接下來iPad會依據當下的環境，來決定處於哪一種模式。如果有裝置可以用WiFi或Bluetooth進行連線的話，就會進入EEPS(Energy-Efficient Peer-assist Scheme)模式，使用我們的合作式演算法；否則，iPad的運作就如同一般電腦上的使用者，用3G連上網際網路，找鄰居，交換資料等等。只要iOS系統底層能維持即時的運作，搭配GKSession的功能，我們就能即時將裝置切換到正確的模式。

Algorithm 2 P2P Streaming (iOS Version)

```
1: Join the main streaming system
2: Start the GKSession, search for other peer
3: if there are short-range neighbors then
4:     EEPS()
5: else
6:     act as a normal peer in main P2P system
7: end if
```

4.3.3 省電同儕輔助機制(EEPS)

現在介紹我們設計的省電同儕輔助機制(EEPS)。這個部分的想法，主要來自去年學長的畢業論文[11]。在作者論文中的環境，有一群具有3G和短距離傳輸介面的行動裝置，透過短距離傳輸介面，形成一個本地端點對點網路(local area ad-hoc network)，而這些裝置都想要下載相同的影音串流。作者定義「系統生命時間」為所有的裝置都有剩餘電量，而能待在這個系統中的時間，也就是說，只要其中

一個裝置電量用盡即宣告結束。作者要解的問題是，如何最大化這個系統中的「系統生命時間」。

這篇論文先忽略手機的移動性，以及裝置可能進出系統的問題，只討論在環境固定的情形，而且假設每個裝置的下載能力都是充足的。鄰近的裝置會互相交換剩餘電量的資訊。裝置取得網路串流資料的方式，可以有兩個選項，自己透過3G連上網際網路下載，或著委託給鄰近的裝置，請他透過短距離網路傳給自己。

在這樣的環境下，論文首先證明了，想要最大化「系統生命時間」，這是一個NP-hard的問題。然後設計了一個分散式的啟發式演算法(heuristic algorithm)，每個裝置藉由收集鄰近裝置的剩餘電量，判斷應該將下載委託給哪個鄰近裝置，或是自己下載，來趨近最佳解。

因為目前我們的實驗器材有限，最多只有四台，所以我們沒有考慮比較複雜的情形，只就簡單的環境做討論。我們實作的方法很簡單，就是只考慮一步距離(one-hop distance)的資訊。從裝置自身的短距離介面通訊範圍中，尋找出剩餘電量最多的鄰近裝置，將自己的下載委託給該裝置，該裝置在收到任何影音串流資料後，就會傳遞過來。如果自己的剩餘電量最多，就自己連上網際網路下載。委託的關係示意圖如圖4.2。如果是有很多裝置的複雜環境，學長的論文中設計了很好的演算法，藉由收集更遠鄰居的剩餘電量，來計算出對整體更佳的解法。

圖4.2會看到，這群鄰近的行動裝置，會形成一個樹狀的結構，用推的方式(push)的方式傳遞資料。在同儕式的資料交換系統中，樹狀的結構被認為缺乏面臨環境變化的彈性。但我們的環境中，行動裝置的短距離傳輸介面會一直保持在偵測狀態，他會隨時知道周圍裝置的狀態，並且每五秒鐘就會和鄰近裝置交換電量的訊息。於是裝置能夠對環境的變化，立刻改變自己委託的狀態。

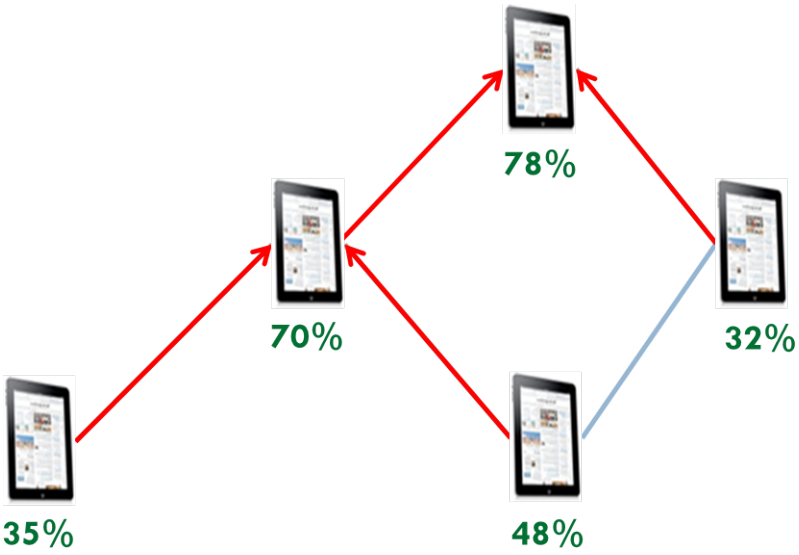


Figure 4.2: 裝置間委託的情形

我們設計了一個簡單的實驗，由三台iPad，同時加入這個系統，其中iPad A和另外兩台分開，而iPad B和iPad C的關係則固定是B委託給C。實驗的結果顯示，B的電量消耗會比A和C少大約25%，詳細內容將在第五章介紹。

4.3.4 提升觀賞品質的同儕輔助機制

上一節我們提到，如何藉由同儕輔助，讓裝置之間達到省電的效果，然而同儕之間還可以進行更進一步的分工合作，若裝置A和裝置B都完整使用自己的下載頻寬，下載不同的資料，再互相分享，就可以可以享受兩倍下載頻寬的觀賞品質。

事實上用單一裝置的3G網路頻寬，是很難達到足夠好的觀賞品質的。在夜深人靜的時段，單一3G大約可以支援到550kbps上下的即時影音串流，但在日常的時段，用戶較多的時候，單一3G下載通常只能支援到350kbps左右，慘的時候會更慘。為了提升裝置使用者的觀賞品質，我們設計了同儕輔助的機制。

這部分還是建築在上一節的委託機制之上。假設現在iPad A受到周圍大家的委託，自己透過3G下載影音串流。因為我們的主系統是一個拉曳式的點對點交換

系統，iPad A每一秒鐘會根據網際網路中的其他鄰居的狀態，發出串流資料的需求。假如目前3G網路每秒鐘可以下載50個資料塊，但A現在卻需要下載80個資料塊，A知道自己頻寬不足，於是他會將另外30個資料塊的需求，傳給其短距離網路中的一個鄰居，請他下載多餘的部分，再傳回給A。這樣的方式即讓系統中的所有裝置能觀賞到更佳的觀賞品質。

當受到委託且需要從3G下載的裝置，要將超出自己負荷的需求量分給其他裝置時，會選擇剩餘電量最多的裝置，這是一個最簡單的方式，也遵循著最大化系統生命時間的方針。當然這樣的方式會增加電量的消耗，但如果選擇的短距離傳輸介面是Bluetooth的話，其傳輸耗損電量幾乎小的可以省略，如此情況總耗電量也不會超過裝置單打獨鬥，沒有合作的情形。

Chapter 5

成果展示

我們的成果主要分以下三個部分展示：

- 實驗一：傳送影音資料時，依據資料重要性做為傳送的優先順序，做為增進觀賞品質的機制（簡稱為QoS機制）。
- 實驗二：在行動裝置上面，利用短距離傳輸介面，將下載的工作委託給剩餘電量較大的裝置，使系統能更節省電量消耗（簡稱為EEPS機制）。
- 實驗三：行動裝置之間下載頻寬的互助，使行動裝置能觀賞到比單獨下載更好的觀賞品質（簡稱為COQoS機制）。

以下如果沒有特別說明的話，我們的影音串流使用的位元率(bit rate)為350kbps，而短距離傳輸介面則都選擇Bluetooth。

5.1 以觀賞品質為基礎的排程（QoS機制）

5.1.1 環境介紹

我們將比較兩種即時影音串流系統，帶給使用者觀賞品質的差異。一個有加入QoS機制，裝置在回覆鄰居的資料請求時，會考慮資料對觀賞品質的重要性，另一個則沒有加入QoS機制。在這兩個系統中，各有一個伺服器，他們接收相同的電視訊號，傳出相同位元率的影音串流，然後將串流資料帶入系統中。伺服器

之外另外開了三台電腦，做為系統中的其他裝置，然後由兩台iPad，分別加入兩個系統，比較接收到的觀賞品質。

為了要做出觀賞品質的差異，我們會調整伺服器輸出串流的位元率，使iPad的頻寬恰好稍微不足於下載全部的串流資料，因為3G的頻寬並不固定，於是最能表現效果的位元率，也會因網路環境不同而改變，經驗上大約是要使發生丟失的機率(loss rate)在0.2至0.3左右。在這個實驗裡，我們設定iPad之間不會互相進行合作。

5.1.2 實驗成果

圖5.1是實驗一的情形，右邊的iPad使用了QoS機制，左邊的iPad則否，圖5.2則是更近拍攝這兩台iPad，可以看到畫面下方顯示的累積資訊，最下面一排顯示了這台iPad目前累積所收下的資料塊(chunk)的情形。我們可以看到，使用QoS機制，明顯能增加對於IDR和音訊資料的傳輸率，而視訊的P、B slice則收得比較少。

累積資訊的上排左邊，顯示的是「丟失資料塊數／應收到資料塊數（丟失率）」。雖然統計所有資料塊的總和，若使用QoS機制，會遭遇到比較大的丟失(loss)，但是因為能收到較多重要的資料塊，於是使用者仍享有較好的觀賞品質。我們的機制會造成較嚴重丟失的理由，是因為較不重要(例如B slice)但已經接近播放期限的資料塊，在我們的機制會被拉到後面的順位，於是使用者最後總是收不到這些不重要的資料塊。

5.2 行動裝置互助的省電效果（EEPS機制）

5.2.1 環境介紹

為了實驗互助的省電效果，我們選用三台iPad，其中一台(iPad A)不和其他兩台合作，另外兩台(iPad B、iPad C)則會互相合作。互連的兩台互相使用委託的方式，

將串流資料的下載工作都託付給剩餘電量較大的裝置，然後觀察這三台iPad的電量消耗。

這三台iPad的都是加入同一個即時影音串流系統，而在使用行動裝置互助的環境中，我們對原本的串流系統做了一些限制，使這些行動裝置不會被加入伺服器中的使用者名單，如此一來這些iPad互相就不會在網際網路上看到。這樣的方式使得短距離傳輸介面形成的本地端網路，和網路網路不會發生重疊，確保不會進行重覆的資料交換。

5.2.2 實驗成果

圖5.3是實驗二的情形，下面兩台iPad是使用EEPS機制，上面那台則是單獨下載，使用EEPS的兩台iPad，觀賞到的是相同的內容，因為他們會互相分享下載的資料。圖5.4是實驗大約80分鐘後，iPad這段時間的累積資訊。我們可以看到，兩台使用EEPS機制的iPad，每台平均可以省下約15%的電量，經過一些換算，將下載託付出去的iPad，他的耗電量大約可以比一般的情形，節省將近30%左右，我們可以預估，對於一個存在n個裝置的系統，當其中只有k個裝置使用3G下載時，整個系統就可以省下$30(1-k/n)\%$的電量。

5.3 行動裝置互助以提升觀賞品質（COQoS機制）

5.3.1 環境介紹

和上一個展示一樣，我們選用三台iPad，一台(iPad A)不和其他兩台合作，另外兩台(iPad B、iPad C)互相合作。互連的兩台同樣使用委託的機制，但是加入了頻寬互助的機制，被委託的裝置下載頻寬不足的部分，會請另一個裝置去下載。

頻寬輔助的部分，需要用3G介面下載資料的裝置，會加入該同儕式系統，每秒鐘根據網路上其他鄰居的緩衝區狀態，發出資料塊的需求。我們設定了一個資

Figure 5.1: 實驗一：右邊為使用QoS機制，左邊則否。右邊得到較佳的觀賞品質

料塊數量的門檻，如果需求量超過這個門檻，就會將超過的部分傳給另一個裝置，請它去下載。我們設定的門檻值為50個資料塊。

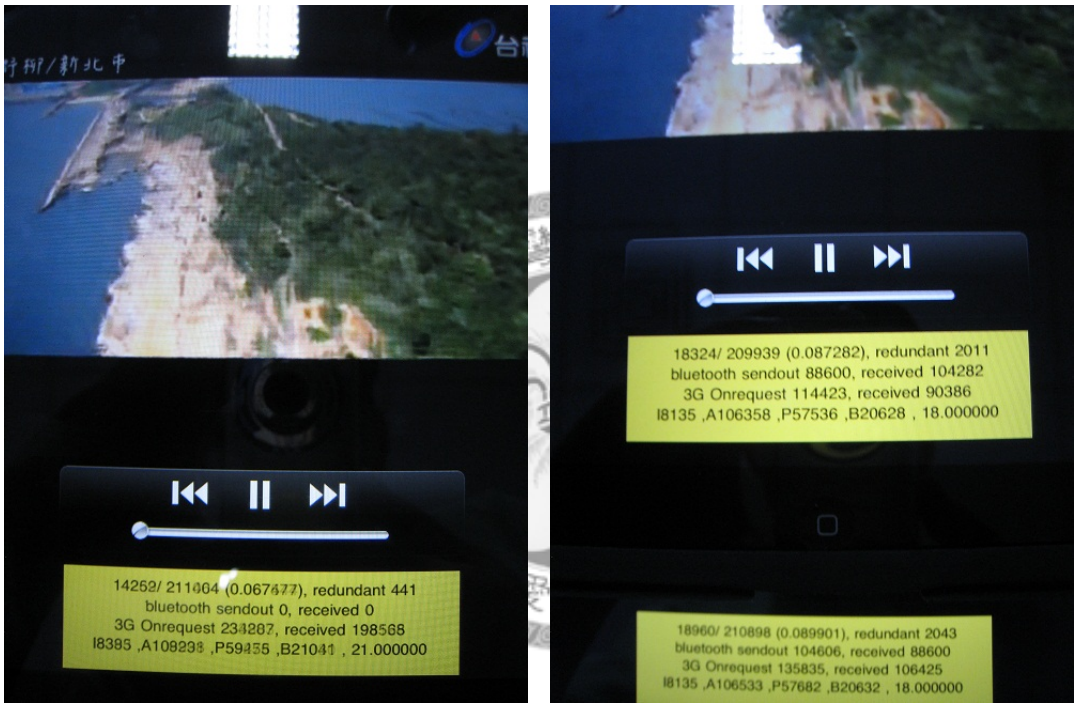
5.3.2 實驗成果

圖5.5是實驗三的情形，和實驗二一樣，下面兩台iPad是使用COQoS機制，上面那台則是單獨下載。我們可以看到，單獨下載的iPad遭遇比較嚴重的丟失情形，連得使畫面比使用COQoS機制的iPad還要延遲。而圖5.6則是比較兩種情形之下，iPad的累積資訊。在丟失率的部分，COQoS機制明顯得表現較好，也可以從累積資訊的第二和第三排看到，第二排顯示的是該iPad使用Bluetooth傳送和接收的資料塊數量，第三排則是發出的3G需求數量（因為資料會丟失，對於沒收到的資料，3G需求可能會重覆發出），可以看到使用COQoS機制，iPad會進行需求資料的合作，兩台iPad會各自下載一部分的資料，再互相分享，因此iPad B和iPad C的觀賞品質會比iPad A還要好，驗證了這是一個有效的機制。

(a) 不使用QoS (b) 使用QoS

Figure 5.2: 實驗一中，比較兩個環境的傳輸情形

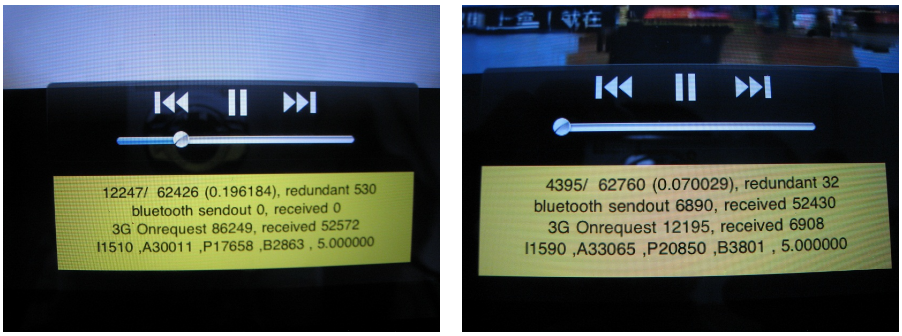
Figure 5.3: 實驗二：下面兩台iPad使用EEPS機制，達到省電的效果



(a) 不使用EEPS　　(b) 使用EEPS的兩台iPad

Figure 5.4: EEPS機制的電量耗損比較

Figure 5.5: 實驗三：下面兩台iPad使用COQoS機制，提供較好的觀賞品質



(a) 不使用COQoS　　(b) COQoS

Figure 5.6: 由累積資料來比較COQoS的效果

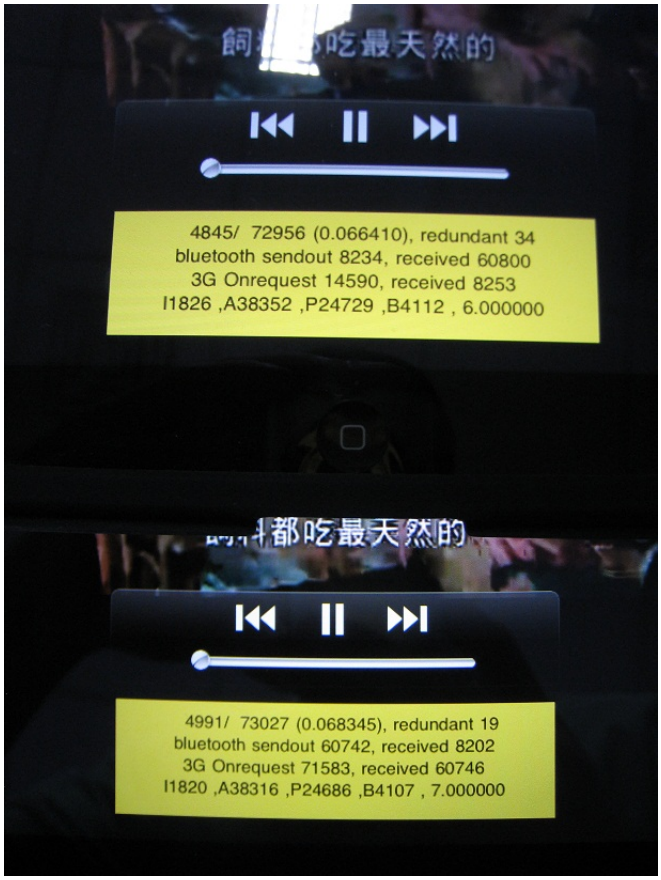


Figure 5.7: COQoS機制的兩台iPad的累積資料比較

Chapter 6

結論

6.1 結論

我們實作出一個即時的同儕式影音串流系統，資料在系統中流通的同時，考慮了資料的優先順序，讓使用者能優先接收到較重要的資料，在一個網路情況較不好的環境下，能提升使用者的觀賞品質。

雖然行動裝置的下載頻寬較為不足，但是可以利用其多重傳輸介面的特性，由短距離傳輸介面將鄰近的裝置連成一個本地端網路，彼此進行互助互利的合作。利用短距離傳輸介面較省電的特性，降低系統的電量消耗，以及利用不同裝置的頻寬合作提升使用者的觀賞品質。

6.2 本系統的缺點

接下來介紹一些本系統中較為不足，或是在實作中面臨到的問題和限制。

1. 在系統中資料流通時，優先考慮較重要的資料，可以提升使用者的觀賞品質，在頻寬不足的環境下，這是一個很好的方式。但是選用這樣的策略，可能使那些不重要的資料，一直處在較低的處理順位，使其很難在這系統中散佈。對於網路狀況較好的使用者，他可能有充足的頻寬，但因為不重要的資料在系統中散佈很慢，導致能享受的觀賞品質會稍差一些。

如果要改善這個缺點，站在比較個體的立場上，在傳輸資料時，可以同時考慮資料的播放期限(playback deadline)，給那些較急迫的資料較高的處理順位，可以提升資料的傳輸率。如果站在宏觀的角度，則應該使整個系統中較稀少的資料，能更優先被散佈，讓系統中避免資料瓶頸(content bottleneck)的情形。

2. 在無線網路的環境下，如果想傳輸相同的資料給兩台以上的裝置，使用廣播(broadcast)或群播(multicast)的方式，無疑是較有效率的方式，只需要傳輸一次資料，所有傳輸範圍內的裝置都能收得到。但根據我們的實驗與觀察，GKSession並沒辦法做到資料的廣播或群播，只能用一對一的方式傳輸，這降低了短距離介面的效率。

3. WiFi因為頻寬較大，所以雖然WiFi傳輸無法使用廣播的優點，但也較不影響整個系統的傳輸效率。但是Bluetooth在有三個以上裝置要傳輸資料時，會產生比兩個裝置的情形下，多兩倍以上的延遲。 這是因為Bluetooth的網路，在一群裝置之中，只會有一個是master，其他裝置都是slave。整個Bluetooth網路中的傳輸都用master負責管理，可是我們無法得知Apple iOS內部如何處理GKSession，進行Bluetooth傳輸的部分。我們只能讓peer A傳資料給peer B，但我們無法知道誰是master，也無法控制傳輸的方式。但可以猜測的是，slave之間不會直接傳遞資料，需要透過master的安排。因為Bluetooth的裝置間不對稱的性質，使得目前我們想要在Bluetooth上，實作三個裝置以上互相合作，是有困難的。

4. 因為Bluetooth和WiFi的頻率互相衝突，所以在實作的時候，這兩個介面只能選一個用，這也是實作上無法避免的限制。

Bibliography

[1] ISO/IEC 13818-7, ISO/IEC 14496-3.

[2] F. Albiero, F. H. Fitzek, and M. Katz. Cooperative power saving strategies in wireless networks: an agent-based model. *IEEE International Symposium on Wireless Communication Systems*, 2007.

[3] J. Chakareski, J. G. Apostolopoulos, S. Wee, W. tian Tan, and B. Girod. Rate-distortion hint tracks for adaptive video streaming. *IEEE Trans. on Circuits and Systems for Video Technology*, 15(10):1257–1269, 2005.

[4] P. A. Chou and Z. Miao. Rate-distortion optimized streaming of packetized media. *IEEE Trans. Multimedia, submitted for publication*, 8(2):390–404, 2006.

[5] F. H. Fitzek, M. Katz, and Q. Zhang. Cellular controlled short-range communication for cooperative p2p networking. *Wireless World Research Forum*, 2006.

[6] ISO/IEC 14496-10.

[7] PPLive. `http://www.pplive.com`.

[8] PPStream. `http://www.ppstream.com`.

[9] VideoLAN - VLC media player. `http://www.videolan.org`.

[10] `http://developer.apple.com/devcenter/ios/index.action`.

[11] K.-H. Lee. *Energy-efficiency Peer-assisted Streaming System for Mobile Devices*. Master Thesis, 2010.

[12] X. Liu, G. Cheung, and C. N. Chuah. Structured network coding and cooperative wireless ad-hoc peer-to-peer repair for wwan video broadcast. *IEEE Transactions on Multimedia*, 11(4):730–741, 2009.

[13] Z. Miao and A. Ortega. Expected run-time distortion based scheduling for delivery of scalable media. *Proc. Int. Packet Video Workshop, Pittsburgh, PA*, 2002.

[14] G. P. Perrucci, F. H. Fitzek, A. Boudali, and M. C. Mateos. Cooperative web browsing for mobile phones. *International Symposium on Wireless Personal Multimedia Communications*, 2007.

[15] S. Raza, D. Li, C. N. Chuah, and G. Cheung. Cooperative peer-to-peer repair for wireless multimedia broadcast. *IEEE International Conference on Multimedia and Expo*, 2007.

[16] ISO/IEC 13818-1, ITU-T Recommendation H.222.0.

[17] X. Zhang, J. Liu, B. Liz, and T.-S. P. Yum. Coolstreaming/donet: A data-driven overlay network for efficient live media streaming. *IEEE Infocom*, 2005.