

應用於 BitTorrent 之通透快取機制

^a陳宜凡、^{ab}游象甫、^a曾黎明、^a白淳元

^a中央大學電子計算機中心

^b中央大學資訊工程系

摘要

由於同儕網路的蓬勃應用，其流量已成為 WWW 外最大宗。然而同儕網路中使用者之間直接的資料傳輸，與底層的拓撲並不相符，在路由器之間造成了重複的流量，降低了頻寬的使用效率。本論文主要分析，如何讓區域網路的使用者能夠彼此優先分享檔案，從而減少頻寬的浪費，使用者也能得到快速下載的服務品質。藉由在區域網路中設置一台通透的網路代理伺服器，讓使用者取得修改過之 Metainfo 檔案，而連上內部所設置的一台 BitTorrent Tracker，由內部此台 Tracker 告知使用者快取的位址而可以向快取下載檔案，並且區域網路中有相同檔案下載請求的使用者也能透過內部的 Tracker 彼此優先分享檔案，而不需連至外部網際網路的使用者索取檔案。

關鍵字：同儕網路、快取、防火牆、網路代理伺服器、BitTorrent

1 緒論

利用同儕網路技術進行檔案分享已經受到廣泛的使用，使用者可以很方便地由網際網路上的其他使用者取得想要的檔案，因此也產生了相當大的流量，除了 WWW 的資訊流量外，也成為佔用網路頻寬的一項主要服務。根據威斯康辛大學一項網路流量持續性測量(<http://wwwstats.net.wisc.edu>)，P2P(KaZaa, Gnutella, and eDonkey)的流量佔了校園網路的 30%。

同儕檔案分享 (Peer-to-Peer file sharing) 應用上，代表性的軟體產品有 Napster [1]、Gnutella [2]：

Gnutella，是完全同儕化的網路系統。相對於 Napster 集中式的索引查找，Gnutella 在搜尋上是分散式的查找，其協定是以廣播的方式在 Gnutella 網路上詢問檔案所在，接收詢問的點如果擁有該檔案

便回應詢問者，反之則代為轉發該訊息。在下載層面也是點對點直接連線，接收到回應的詢問者向擁有檔案的那些點個別連線，取得檔案。不管是 Gnutella 或者是 Knapster 在擴展性上的表現都有其限制，knapster 因其集中式索引，Gnutella 則因其廣播機制造成詢問訊息的氾濫，浪費頻寬，另外在查找距離遠時，時間便會增長，因此查找效率較低。所以結構化的查找架構是最新的研究方向，以分散式雜湊表 (Distributed Hash Table, DHT) 的方式，在各個點上維持一張輕量級的表格，經過適當的雜湊，將每個物件 (檔案) 對應到其他點，經由在幾個點之間跳躍的詢問，逐漸接近擁有者。例如麻省理工學院的 Chord [3][4]、AT&T 的 CAN [5]、加州柏克萊大學的 Tapestry [6][7][8]、史丹福大學的 YAPPERS [9][10] 及微軟倫敦康橋軟體研究院的 Pastry [11][12]。

除了 Gnutella 之外，BitTorrent [14] 也是一種 P2P 檔案交換技術，採取其特殊的獎懲機制來鼓勵使用者能夠貢獻上傳頻寬，使得檔案的分享能夠更為快速，是目前非常熱門的檔案交換應用軟體。

在 Stefan 等人的研究 [13] 中，模擬在 KaZaa 網路中加入 P2P cache 的效能。在連外的交通方面，快取命中隨著時間的而成長可至 85% 連內便只有緩慢成長至 35%。而快取命中率在僅有一千參與點時，便有高達 40% 的符合率，隨著用戶數的增加可成長至 85%。在參與點眾多時，快取的效率一般可見。

同儕網路是屬於應用層群播 (Application layer Multicast)，應用層群播是由終端使用者充當網路的功能，類似路由器轉送封包，由終端使用者替彼此轉送內容資訊。這樣的方式比起網路層群播 (Network layer Multicast, IP Multicast) 在建置上容易許多，也不需要耗費路由器去維持並建立路由資訊。但是相對應便會犧牲掉一定程度的網路頻寬，

造成網路上重複的流量。而這端看應用層路由之拓撲與底層網路拓撲的契合度如何。在區域網路經由閘道器連外至網際網路的環境下，點對點覆層網路 (Peer-to-Peer Overlay network) 很明顯地會造成相當大的頻寬浪費(圖 1)。

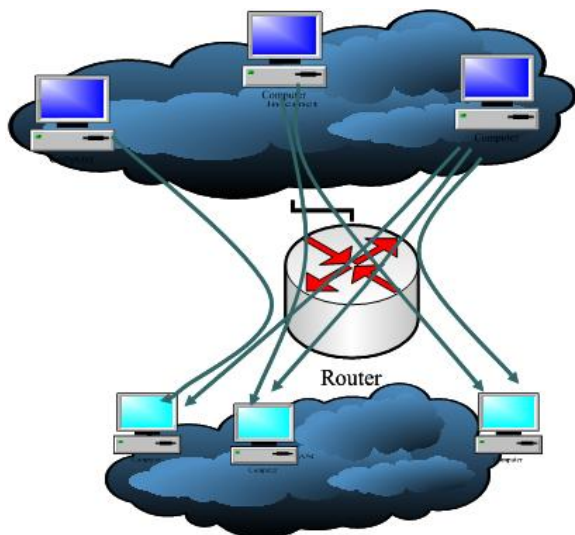


圖 1：同儕網路傳輸示意圖

由圖中可知，同儕網路點對點直接連線的特性，造成相同資料的重複傳輸，不但使用者等待時間較長，供檔端(server peer)負荷也較大。實際上如果區域網路中有使用者擁有相同的檔案，或許正在下載，使用者可找到此近端的使用者快取，優先從近端使用者處下載檔案，以節省網路頻寬，增加效能。

我們可用架設防火牆的方式來監聽所有的連線，將所有的連線加以攔截限制，而在本文中提出一個機制，藉由通透的代理伺服器來攔截並修改使用者至網站伺服器上所下載 BitTorrent 的 metainfo 檔案，將使用者與原 tracker 的連線導向至我們在區域網路中設置來替代裡的內部 Tracker，藉由此內部 Tracker 來提供使用者供檔者清單，包括負責由網際網路下載檔案的快取點以及其他索取相同檔案的使用者，使區域網路內的使用者能夠由彼此進行分享，也不會浪費網路的頻寬，也因此能夠偵測並攔截到區域網路使用者與網際網路使用者資料傳輸的連線。

本論文分為五個部份，在第二章列出相關研究。第三章分析問題且提出系統架構。第四章為系統實

作，最後是結論及未來研究方向。

2 相關研究

本章分析同儕網路的特性，比較網路頻寬使用效能之策略，以及相關研究之改進

Giraud [16] 中認為BitTorrent覆層網路的建構是藉由隨機地選擇點的方式，具有嚴重影響每個使用者的系統效能以及整個網路的資源的潛質，因此提出建立一個較佳的覆層網路的策略，並相信建立一個根據位置遠近來建立的覆層網路是能夠改善整體網路的頻寬使用、以及終端使用者的負載。採用了 synthetic network coordinates 以及 probabilistic flooding 來降低搜尋近點的多餘工作負載。

GW[17]為Gnutella網路提出集中式的查詢流量解決之道。這篇提出以集中式的查詢快取機制來收集使用者的查詢封包，藉由在閘道器的地方設置該機制，將所有查詢封包攔截，由快取機制回答查詢，如果查詢有命中則回應，反之則仍舊轉送查詢封包。見圖 2，Gnutella點之間，所公開默認的埠號為 6346，進行初始化。

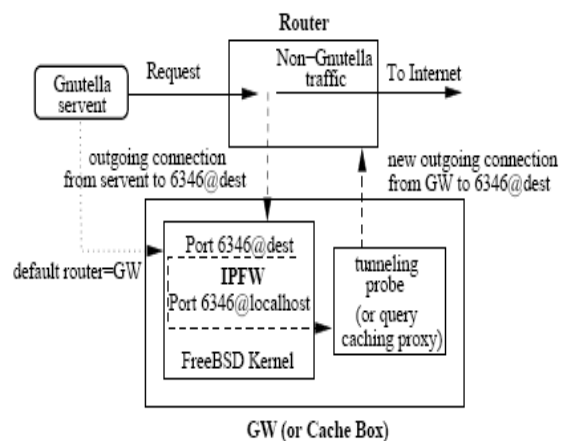


圖 2：查詢快取模組圖 [17]

GW 的方式為將區域網路內的 Gnutella 點 (Servent) 發出的請求在經由閘道路由器連往網際網路時，啟動 Layer-4 的功能把屬於 Gnutella 查詢請求(目標埠號為 6346 之封包轉向至 GW(or Cache Box)，而不屬於 Gnutella 的封包則放行。使用者也可自行設定 GW 為預設閘道器。被轉向至 GW 的封包，會再被 GW 內的防火牆以 IP Forwarding 導

向原本到目的端 6346 埠的封包到本地端的 6346 埠。聽取此埠號連線的為查詢快取模組，為一修改過的 GTK-gnutella 開放源碼程式。此模組便會執行以上的查詢轉向或回應動作。

Uniform index caching(UIC)的查詢快取方式是由查詢回應的返回路徑上的所有參與點都快取一份下來。DiCAS [18] 的實驗中顯示UIC的方式會在各點產生過多重複、不必要的訊息，進而提出減輕快取負載的方法。在改善搜尋效率方面，以分散式的快取 (Distributed Caching) 機制將快取結果分配到附近的點，根基於分散式快取之外，也是能具有調適性的搜尋機制 (Adaptive Searching) 能將查詢訊息轉發(Forwarding)到較有可能得到回應的節點，選擇性的散佈查詢訊息，而非盲目地選擇目標。

然而 DiCAS 仍有其侷限，有可能會造成某些查詢訊息永遠無法到達某些點，即使該點擁有該份文件。因為該點所屬群組不等於該文件運算出的群組 ID，該查詢訊息只會在所屬層上蔓延。

我們於下表 1 將上述三篇有關同儕網路檔案交換的快取系統作一份比較。首先比較系統的架構，還有所快取的資料。在第三項比較使用該系統對使用者增加的額外負擔。第四項為系統所使用的方法，最後一項則為該同儕快取系統是否能找到近端使用者。

項目比較	GW	DiCAS	Giraud
系統架構	Tree	Distribute	Distribute
物件快取	Metadata	Metadata	Metadata + content
使用者額外負擔	None	small	Medium
物件搜尋方法	interception	Layers	DHT
近端同儕搜尋	Yes	no	yes

表 1 各個 P2P 快取比較

3 系統設計

在系統設計部份我們進行方法的分析與特性

比較。在提出我們的系統架構、以及運作流程。我們假設使用者皆是透過 WWW 的方式取得 BT 的種子檔案(metainfo)。

3.1 BitTorrent

Bittorrent [14] 是目前相當流行的P2P檔案交換方式。Bittorrent (BT) 是一種P2P檔案交換協定，其架構屬於混合型(hybrid)，在搜尋檔案、查找點上採取不同的策略。以下為Bittorrent的兩個特殊的組成元件：

- The static 'metainfo' file
- The BitTorrent tracker

Metainfo 檔之副檔名是.torrent，顧名思義，為分享文件的索引檔案，內容包含，「tracker」之位址(URL)、文件的各項資訊、以及每個文件片段以 SHA1 函數運算的雜湊結果。文件的發佈者將製作出公佈於網站上同時也發佈該文件至 tracker。而 tracker 為一個特殊的站台，所負責的工作僅為回應使用者欲查找的檔案供檔者清單的詢問，即分配文件的供檔者清單。避免集中式設計的 tracker 因為必須服務的工作過多而發生錯誤。

文件的分享過程是以在網站上公開發佈欲分享文件之 metainfo 檔案，當使用者下載.metainfo 檔之後，啟動的 BT 的用戶端軟體來進行檔案之下載。用戶端軟體在讀取 metainfo 檔之後，首先確認該份之完整性(integrity)，通過之後與指定的 Tracker 連接，詢問文件的提供者之位址(list of peer)。而 Tracker 本身已由該原始檔案之提供者(種子, seed)發佈有關原始檔案之資訊，便會回應目前擁有這份原始檔案完整的使用者位址，以及僅取得部分原始檔案片段的使用者位址清單。取得清單之後，用戶端再一一與清單上的下載者進行連接，經下載者同意上傳之後，由一個以上的來源取得文件。

3.2 系統說明

我們在 BitTorrent 網路中分層攔截由一開

始下載 metainfo 檔，Tracker 通訊，至覆層網路的建立 (peer connection)。我們設置 Torrent Agent 於網頁伺服器與使用者瀏覽器之間，作為攔截區域網路使用者下載 metainfo 檔案的 Torrent filter。當使用者啟動其 BitTorrent 用戶端軟體之後，Tracker Agent (TA) 負責攔截使用者對外通訊 (Tracker 通訊)。使用者下載資料的通訊透過快取由外界取得，是由 Peer Agent 負責在外部向網際網路抓取檔案，內部的 TA Client 向內部 tracker (即 TA) 發佈檔案，圖 3

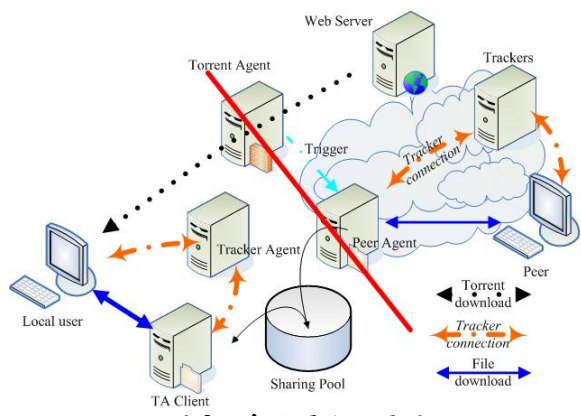


圖 3：系統環境配置圖

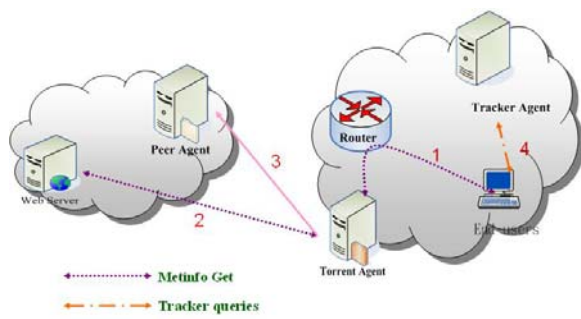


圖 4：系統初始時之示意圖

圖 4 所顯示為系統與網路節點互動之步驟示意圖第一部份，第一位下載文件的使用者產生的系統運作。區域網路內的使用者在網頁瀏覽器上點擊有興趣的物件，伺服器會將物件的 metainfo 檔 (.torrent) 回傳給使用者，但使用者所取得之 metainfo 檔實為修改過之版本。在 Tracker 代理者部分當 Torrent 代理者在收到有使用者下載新的 .torrent 檔的 HTTP 請求時便已通知 Tracker 代理者，Tracker Agent 會取得原始 metainfo 所記載之 trackers，並於自身 tracker 發佈該文件，並向原登記 tracker 詢問下載者清單。在當第一位使用者取

得修改過之 metainfo 檔案，便會與 tracker 代理者 (TA) 連線，而 tracker 代理者已經發佈了該 metainfo 檔案。便會回覆使用者一張目前的下載者清單，此清單是目前 tracker 代理者所登記過的區域網路使用者。

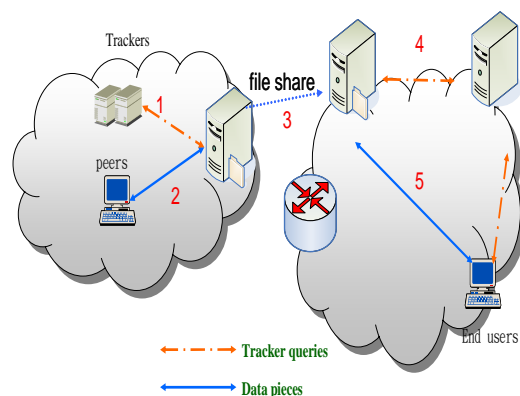


圖 5：系統中外外部 peer 下載檔案示意圖

圖 5 所示為外部 Peer Agent 在被 Torrent Agent 驅動之後，持有原來的 metainfo 檔案，索取相同的來源檔案。並將資料寫入至一與區域網路中的 TA Client 共享的磁碟區中，由內部的 TA Client 讀取分享的檔案並發佈於內部 Tracker Agent，此時區域網路中只有 TA Client 有文件本身資料，內部的使用者便會嘗試連接上 TA Client，並在連接之後發出索取文件片段 (piece) 的請求。

圖 6 當有相同的 (object ID) 文件請求連線至 Tracker 代理者索取供檔點清單，Tracker 代理者也同樣將目前所維護的清單回覆給使用者；使用者便自動根據此清單與各供檔點連接，索取文件片段。

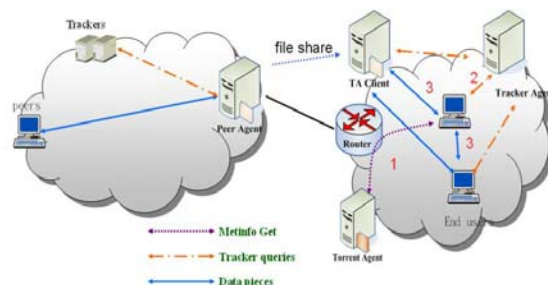


圖 6：第二位使用者加入之系統運作示意圖

3.3 系統模組

本節依據上一節所設計的系統運作方式，詳細說明系統運作所需之功能，以方塊模組表現之；

整個系統分為導向以及代理者兩大區塊。

在圖 7 中使用者設置 Torrent Agent 為網路代理伺服器，將 HTTP 的請求送至 Torrent Agent，Torrent Agent 正在運行的快取伺服器服務在聽取到 HTTP 請求之後會將請求轉向至我們的 Torrent filter 進行分析，篩檢是否有 .torrent GET 的資訊在內。如結果為非，會將流程轉回快取伺服器服務，由該服務回應使用者之請求。如篩檢出 .torrent GET 之資訊且為新的 .torrent GET 請求，則會呼叫抓取模組至使用者原目的地網站抓取該 metainfo 檔。

接著需檢查抓取回來的 metainfo 檔案之完整性，正確無誤才呼叫修改的模組；經過 modification 模組修改之後，將之存放在自身的磁碟，同時也發佈新 metainfo 檔事件，並且返回 Torrent filter。由 Torrent filter 傳回 modified metainfo 之回應。反之，如果此 .torrent GET 資訊已有使用者請求過，Torrent filter 便直接由本地端快取回覆使用者 modified metainfo 之回應。圖中右下角之模組 New Torrent Awareness 為負責通知 Tracker 代理者新 metainfo 下載事件。

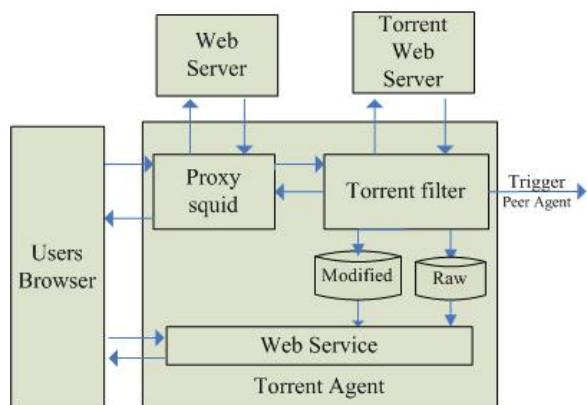


圖 7：Torrent Agent 模組示意圖

使用者可使用代理伺服器分流程式 (proxy.pac) 來設定代理伺服器，在程式中檢查 HTTP 的請求，跟據請求的類型來選擇特定的代理伺服器，以本題為例，即可分流使用者下載 Metainfo 檔案之請求至 Torrent Agent，其餘 HTTP 請求則根據效率需求選擇其他網頁快取伺服器提供服務。

如果使用者沒有設置網路代理伺服器的習慣，仍可以通透式代理伺服器PWFS [19] 的方式運作，藉路由器將HTTP請求轉向至Torrent Agent，於Torrent Agent設置DNAT將連線導向至快取代理服務(Squid)。下圖 8 為Torrent Agent 之運作流程圖。

我們設計的子系統 Tracker-Cache 架構 (圖 9)，由一模組負責聆聽新的 metainfo 檔通知。在收到通知之後，由 Torrent Agent 處取得未修改的 metainfo 檔 (即原發佈網站上的 metainfo 檔)。

Peer Agent，接收由 Torrent Agent 送達的訊息，向 Torrent Agent 取得原始未修改的 metainfo 檔，我們設定每一秒鐘檢查一次是否有新的 metainfo 檔進入將取得的 metainfo 檔載入程式，進行新的下載任務。首先，根據拿到的原始 metainfo 檔連上原發佈所提供的 Tracker 進行供檔點的索取。

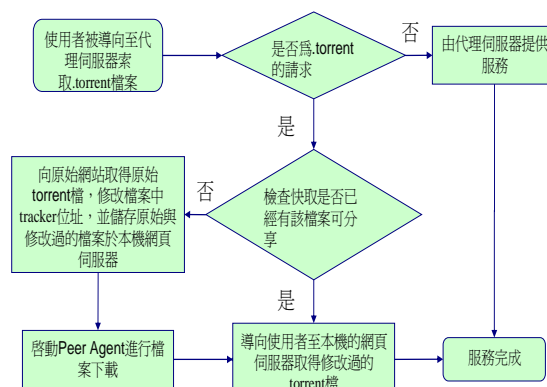


圖 8：Torrent Agent 流程

Peer Agent 在取得供檔點之後會嘗試著與供檔點進行連接。而後 Peer Agent 會將檔案寫入至磁碟。內部的 TA Client 與 Peer Agent 共享磁碟槽，並以唯讀的方式輪詢讀取檔案，發佈於 TA Tracker 之上，如此使用者才能得知快取的位置。由於如果不先至原 Tracker 取得供檔者清單就無法提供服務。根據測量我們設定大約二十秒才發佈資訊至 TA Tracker。

TA Tracker 聽取區域網路被導向至 Tracker 代理者的使用者請求，維護登錄於 TA Tracker 本身的供檔者清單並回應使用者。如有使用者連接上 TA Client 並且要求檔案片段的下載，TA Client 端也會提供服務。

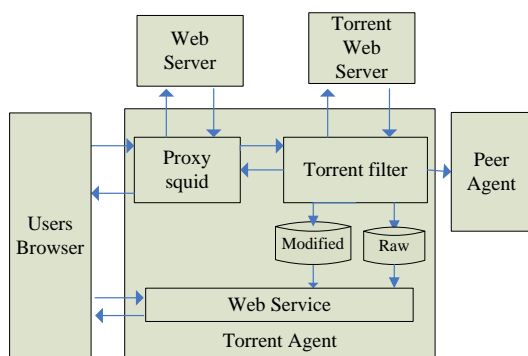


圖 9：通透開道快取其餘的模組示意圖

第4章 系統實作及測試

本章節我們針對前一章所設計的方式，展示系統的運作情形。

4.1 實作環境

我們的系統共分兩個部份，在兩部電腦上實作完成。

本系統的使用方式為，我們將 Torrent Agent 佈置成 Transparent Proxy（通透式代理伺服器），設定開道路由器負責內外網路所經過的流量，處理封包交換，以及執行特殊的轉向，如：將 HTTP 的連線轉向至網路代理伺服器。所以也需於 Torrent Agent 上啟動 NAT 機制，將 80 埠號的流量導向至本地端的 3128 埠。使用者可自行設定代理伺服器（Torrent 代理者）或者使用代理伺服器分流程式（Proxy.pac）來選擇快取代理伺服器，便不需更動路由器之設定，HTTP 的導向便會被過濾，當使用者的請求為一 metainfo 檔時，便會將內容裡的 tracker 列表修改掉，以我們 Tracker Agent 的位址取代上去。

當我們的使用者啟動用戶端軟體欲開始自動下載，便會連至 Trackers Agent，取得擁有者簽單，並從中選取點進行連線，根據 BitTorrent 的協定下載檔案。

4.2 使用流程

使用者於網頁瀏覽器，尋找 BitTorrent 分享群組，於分享群組所架設之網站上瀏覽分享的文件清單。

於有興趣的文件上點取連結，請求下載該文

件之 metainfo 檔。Torrent 代理者會篩檢使用者之請求。回傳使用者修改過之導向用 metainfo 檔。其中有普通文字檔案、圖片檔案、以及 metainfo 檔案的下載請求。

系統之 Torrent Agent 在篩檢到使用者的資訊後，至原發佈 metainfo 的網站下載該索引文件，並存放在磁碟之中。這份資料將不會由使用者所取得。而是讓 Tracker Agent 使用，作為對外窗口所用。可看到 metainfo 檔案前端所記錄的明碼文字為原發佈之網站 URL 及埠號，後端所銜接的部份為發佈文件之資訊，如創件日期、創建者、切割片段大小、片段總數、檔案路徑等。

經過處理之後的 metainfo 檔案內容。區域網路的使用者所取得的檔案此修改過的版本。檔案前端之 Tracker 列表，已經被修改為 Tracker Agent 的位址：`http://140.115.52.149:6969/announce` 當使用者在下載完畢之後，啟動 BitTorrent 用戶端軟體，便會詢問 Tracker Agent 進行通訊。

經過 Torrent Agent 的攔截處理後，使用者回如正常由網頁上下載東西一般，秀出檔案下載的進度視窗，並於下載完全之後詢問使用者 metainfo 檔案的儲存位置。當儲存完畢，可選擇是否直接開啟這份檔案，使用者本身的作業系統會呼叫相關的程式來執行這份檔案，在本環境即為 BitTorrent 之用戶端軟體。見伺服器列表上，可知用戶端根據所開啟的修改過之 metainfo 檔連上的 tracker 是 140.115.52.149，是內部的 tracker。

當使用者連上 Tracker Agent 此時 Tracker 上已發布該份文件。使用者可收到 Tracker Agent 所告知的供檔者訊息，即我們所提供的 Cache。此時所顯示 Cache 已成為區域網路中的種子。

使用者下載過程中，如果系統中有其他區域網路使用者加入，便會彼此分享檔案，藉由 TA Tracker 告知使用者對方的存在。經過一段時間，區域網路的快取提供點下載完畢，並成為區域網路中的種子（seed）。

5 結論及研究方向

本文討論在區域網路內部如果能夠設置一機制，區域網路使用者能夠查找到區域內部的使用者，並優先由使用者處取得資源，並設置快取輔助使用者快取的效能。應可改善外部網路中內容提供者或是路由器的負載，以及網路頻寬使用的效率。所以我們以 BitTorrent 為基礎，在區域網路設置 Tracker Agent 來代替原始 Tracker 分配下載者清單 (Peer list)，並也設置了 Torrent Agent 攔截使用者的 HTTP 的 GET 請求，修改了 metainfo 檔 (.torrent) 來將使用者導向至內部的 Tracker Agent。區域網路內部的使用者透過快取由網際網路取得檔案，並且互相分享檔案

綜合上方所述作法擁有的優點主要是：

- 不需維護龐大的攔截用表
- 使用者不必有明顯的特別設定
- 有效限制對外的流量
- 降低使用者、路由器之負擔
- 提供特定點快取輔助

未來可能的研究方向，除了能搭配自身儲存空間的快取替換法則之外，在佈置多台快取時也提供策略能動態地選擇快取作為特定點快取。例如使用 DHT 技術來為使用者分配快取伺服器的選擇。

參考文獻

- [1] Napster, <http://www.napster.com/>
- [2] Gnutella 0.6 protocol draft, <http://rfc-gnutella.sourceforge.net/>, 2002.
- [3] I. Stoica, R. Morris, D. Karger, M. F. Kaashoek, and H. Balakrishnan, "Chord: A scalable peer-to-peer lookup service for internet applications," in *Proceedings of ACM SIGCOMM*, San Diego, pp. 160–177, August 2001
- [4] F. Dabek, E. Brunskill, M. Frans Kaashoek, D. Karger, R. Morris, I. Stoica, and H. Balakrishnan., "Building Peer-to-Peer Systems With Chord, a Distributed Lookup Service," *8th Workshop on Hot Topics in Operating Systems*, Germany, May 2001.
- [5] S. Ratnasamy, P. Francis, M. Handley, R. Karp, and S. Shenker, "A scalable content-addressable network," in *Proceedings of ACM SIGCOMM*, San Diego, pp. 149–160, August 2001
- [6] B. Y. Zhao, J. Kubiatowicz and A. Joseph, "Tapestry: An Infrastructure for Fault-tolerant Wide-area Location and Routing," *UCB Tech. Report UCB/CSD-01-1141*
- [7] B. Y. Zhao, Y. Duan, L. Huang, A. D. Joseph and J. D. Kubiatowicz, "Brocade: landmark routing on overlay networks," *The First International Workshop on Peer-to-Peer Systems (IPTPS)*, Cambridge, MA. March 2002.
- [8] B. Y. Zhao, A. D. Joseph, and J. D. Kubiatowicz, "Locality-aware Mechanisms for Large-scale Networks," *Workshop on Future Directions in Distributed Computing Bertinoro*, Italy, June 2002
- [9] P. Ganesan, Q. Sun, and H. Garcia-Molina, "YAPPERS: A Peer-to-Peer Lookup Service Over Arbitrary Topology," *INFOCOM*, 2003.
- [10] B. Yang and H. Garcia-Molina, "Improving Search in Peer-to-Peer Systems," *ICDCS*, 2002.
- [11] P. Druschel and A. Rowstron, "PAST: A large-scale, persistent peer-to-peer storage utility," *HotOS VIII, Schloss Elmau*, Germany, May 2001.
- [12] A. Rowstron and P. Druschel, "Pastry: Scalable, distributed object location and routing for large-scale peer-to-peer systems," *IFIP/ACM International Conference on Distributed Systems Platforms, Heidelberg, Germany*, pp 329-350, November 2001.
- [13] Stefan Saroiu, Krishna P. Gummadi, Richard j. Dunn, Steven D. Gribble, and Hebnry M. Levy, "Analysis of Internet Content Delivery Systems" In. *Proceeding of OSDI '02*, 2002.
- [14] Bittorrent, <http://www.bittorrent.com/protocol.html/>
- [15] BramCohen, "Incentives Build Robustness in BitTorrent", *Workshop on Economics of Peer-to-Peer Systems*, May 22 2003
- [16] AsfandyarQureshi, "Exploring Proximity Based Peer Selection in a BitTorrent-like Protocol", *MIT 6.824. student project*, May 7,2004
- [17] S. Patro and Y. C. Hu, "Transparent Query Caching in Peer-to-Peer Overlay Networks," in *Proceedings of the 17th International Parallel and Distributed Processing Symposium (IPDPS)*, 2003.
- [18] Chen Wang, Li Xiao, Yunhao Liu, and Pei Zheng, "Distributed Caching and Adaptive Search in Multilayer P2P Networks," in *Proceedings of the 24th International Conference on Distributed Computing Systems (ICDCS'04)*, 2004.
- [19] 郭廖軒,游象甫,曾黎明, "以網域名稱伺服器為基礎之色情網站過濾系統," *TANET 2004*.