BitTorrent P2P 檔案系統下載端網路資源之可調式配置方法與效能實測

呂宗憲　胡誌麟*　梁瑞麟
國立中央大學 通訊工程系
995203021@cc.ncu.edu.tw, clhu@ce.ncu.edu.tw, 100553013@cc.ncu.edu.tw

摘要

同儕網路技術提供使用者快速且高延展性的多媒體內容分享服務，迄今同儕網路服務所產生的數據量已成為今日網際網路流量的主要來源。本篇論文的研究首先針對主流的 BitTorrent 同儕檔案分享協定及其衍生應用，以效能實測方式來分析 BitTorrent 同儕系統中下載端本身之特點；再者，本論文根據分析所得之觀察提出一套動態頻寬分配機制，網路實測的結果顯示所設計的機制能夠有效地控制下載端對於同儕節點的選擇以及網路資源使用，因此能夠維持下載端在平均頻寬使用率和節點利用性等方面之效能。

關鍵詞：BitTorrent、同儕網路、網路資源配置、網際網路量測

Abstract

Peer-to-peer networking technologies provide users rapid and high scalable content distribution services. The astounding increasing population of peers contributes a major fraction of today's Internet traffic in the global networks. The study of this paper considers the popularity of BitTorrent protocol and its derivatives among many peer-to-peer file sharing applications, and conducts performance measurements upon an extensive collection of real traffic logs as downloading various media files in BitTorrent systems. According to many newly observed features, this paper designs a dynamic bandwidth allocation mechanism which can be employed to efficiently maintain the average bandwidth utilization and the maximal number of uploading peers at the downloading side in BitTorrent systems.

Keywords：BitTorrent, Peer-to-peer (P2P) networks、Network resource allocation、Internet measurement

1. 簡介

我們見證了網際網路中多媒體內容與用戶群的驚人成長，這成長的關鍵在於如何使用有效率和可靠的方法來加速網際網路中多媒體內容的散佈。鑒於傳統主從式(Client-Server; C/S)架構與內容傳送網路(Content Delivery Networks; CDN)架構易受到系統層次性因素而導致效能降低等問題；對此，同儕網路技術採以高度分散式的組織架構，藉由同儕節點之間的互助和資源分享，避免系統層次性問題而維持系統服務[1]。

P2P 網路與 C/S 和 CDN 在架構設計上有著不同之別，諸如系統組織、節點的多重角色扮演以及服務能力等方面。C/S 架構主要藉由單一伺服器來與其它節點交換伺服器上的資源，為了避免當單一伺服器服務過載或中斷，通常會保守性地限制整體的工作量、頻寬使用和資料吞吐量，以避免系統過載之可能。至於 CDN 則是增加多餘的核心項目和後援的伺服器數量，用以提高整體系統的能力並減緩骨幹網路及節點的工作負載和流量。不過，CDN 的系統建置成本極高，之後仍需花費大量成本來升級後端系統以及系統管理與維護。相較之下，P2P 網路模式並非結合特定的伺服器來運作，與前兩者本質上的差異在於節點的雙重角色扮演，每個節點同時扮演提供服務的伺服器節點(Server peer)和使用服務的終端節點(Client peer)這兩種角色。也就是說，當節點總數增加時，即使整體服務負載量會增加，同時間可提供服務的節點數量也伴隨地增加[2]。因此，在 P2P 網路中每個節點可利用眾多其他節點所提供的大量儲存、傳輸和計算資源來協同進行內容傳輸和分享服務。

近期的網際網路量測報告[3]指出目前網際網路流量大部分來自於 P2P 檔案分享應用，其中以 BitTorrent 及其衍伸應用所產生的流量最多。這樣的現象相當引起我們的注意，我們審視先前許多 BitTorrent 效能量測的相關研究，大部分的文獻主要關注於以下幾點：第一、整體系統的服務能力[4][5]，第二、針對 Free-riding 情況下的公平性問題[6]，第三、網路中節點 Churning 和檔案可用性問題[7]，第四、不同節點選擇方法的差異[8]，第五、BitTorrent 系統內 Tracker 與上傳及下載節點之間的交互行為[9]。因此，過去的研究貢獻主要著力於系統端與上傳端兩方面的效能量測和分析。

然而，若僅分析系統端或上傳端的量測結果，實際上仍不足以詮釋 BitTorrent 協定對於下載端之影響。由於 BitTorrent 系統內檔案的上傳與下載並

This work was supported in part by the National Science Council of Taiwan, R.O.C., under Contracts NSC100-2221-E-008-085-MY3 and NSC101-2420-H-004-006-MY3.

非同步運作，並且過去的文獻仍然少有下載端效能的研究。對此，本論文的研究特別針對 BitTorrent 系統下載端之行為和傳輸效能進行網路實測，以期能更深入了解 BitTorrent 系統。我們經由大量的量測分析提出許多值得參考的資訊及新的量化指標，例如檔案的下載時間、可建立的上傳連線數和頻寬使用率。經由這些指標所顯示出的資訊，我們剖析現有 BitTorrent 下載端的機制和功效，進一步提出一套可調式頻寬分配機制，並且我們成功地將此機制結合現有 BitTorrent 下載端應用軟體，網路實測的結果顯示此機制能夠有效地控制下載端對於同儕節點的選擇以及網路資源使用，維持下載端在平均頻寬使用率和節點利用性等方面之效能。

本論文的章節組織如下所述，第二章節簡述 BitTorrent 背景知識、相關文獻探討以及我們先前的研究，第三章節提出動態頻寬分配機制及其運作流程，第四章節說明所提機制的量測結果與觀察，最後於第五章節總結本論文之研究成果。。

2. 背景知識和相關文獻

本章節簡述 BitTorrent 協定及其量測研究的相關文獻，接著說明本論文所擬方法的基礎知識。

2.1 BitTorrent 協定

BitTorrent 檔案傳輸系統的組成主要包括 Tracker 伺服器、Torrent 網站以及同儕節點三種角色。在 BitTorrent 系統中每個檔案的原始擁有節點一開始會將預備分享的檔案切割成許多個大小相同的檔案片段(介於 32KB 至 4MB)，並為該檔案建立一個名為 Torrent 的 meta-data 文字檔，這個 meta-data 內容描述有關於這個檔案的參考資訊，例如檔案名稱、檔案大小、片段數量、編碼核對碼、註冊的 Tracker 主機位置等，之後節點可將 Torrent 檔案發佈至 Torrent 網站以提供其他節點下載。

Tracker 伺服器的主要工作是負責管理和協調檔案在系統內以何種方式來分散與下載等工作[9]，Tracker 不僅會追蹤所有的檔案擁有者資訊，還可以命令節點如何與其他節點連結以交換和分享檔案片段。因此，當某節點想要下載檔案時(亦即為“下載節點”)，會先到 Torrent 伺服器上尋找和下載這個檔案所對應的 Torrent 檔案，從中取得該檔案已註冊的 Tracker 網路位址，隨後與其建立 TCP 連線，在連線 Session 的有效過程中，Tracker 伺服器將會記錄該下載節點的資訊，並且分批回應系統中當下擁有該檔案片段的“上傳節點”名單，至此，下載節點便可透節點名單來分別與不同的上傳節點建立 TCP 連線，以獲得不同的檔案片段，這樣的連線與下載的工作會反覆持續地進行直到下載完整個檔案。最後，當檔案下載完畢後，該節點也將會成為系統內的「種子節點」，爾後能貢獻檔案片段給其他有需要的節點。

BitTorrent 協定為了能夠管理檔案散佈、頻寬使用和服務公平性，採用以下幾種所有節點必須共同遵守的機制：

Rarest First Strategy：當節點在選擇與其他節點交換檔案片段之時，節點會盡可能優先分享網路中存在最稀少的檔案片段，目的是為了提高每個片段存在於網路中的多樣性，提高整體的檔案分享速度，同時也可避免受到 Flash-Crowd 影響而造成網路中不再有種子節點擁有完整檔案，即斷種現象。

Tit-For-Tat (TFT)：為了鼓勵節點彼此分享，防止僅下載而不提供上傳的節點，意即 Free-Rider，TFT 機制會對這樣的節點進行懲罰，減少分配到的下載資源，以傳達：付出更多上傳的節點能夠得到更快的下載速度，以降低平均下載時間，增加資料交換的公平性。

Optimistic Unchoking (OU)：每個節點會與其他節點保持疏通(Unchoking)狀態(預設為 4 個節點)，BitTorrent 會選擇提供較高的傳輸速率優先疏通下載。做法上，BitTorrent 每隔一段時間(預設為 10 秒)執行一次狀態運算，每 30 秒定期與其他的節點(無論能力好壞)疏通來交換片段，藉以發掘網路中存在更好能力節點。

2.2 文獻探討

誠如前述，過去許多 BitTorrent 量測研究[4-10]主要是從系統端或上傳端節點的角度來分析 BitTorrent 系統的功能與效能。

[4]提出幾項參考性的效能指標，譬如上傳利用率和公平性，並發展 BitTorrent 模擬環境將指標應用到不同的模擬環境，例如同質性結構與異質性結構網路對整體性能所造成的影響，最後並利用 Smartseed 、 Pairwise Block-level TFT 及 Bandwidth-matching tracker 機制來改善系統性能。

[10]著重於討論 BitTorrent 網路中檔案種子的可用性、系統的完整性、系統處理 Flash-Crowd 以及下載的性能，量測結果觀察到以下幾點：BitTorrent 能有效的處理非常大的 Flash-Crowd 情況；當檔案的熱門程度下降時，檔案的可用性將可能會變得不可預測；最後，在實際量測發現高達百分之九十以上的節點下載速率是低於 520 KB/s。

[12]分析 Multi-torrent 的 BitTorrent 系統，對於節點的離開以及服務的公平性等多方面進行探討。[13]提出 Multi-torrent 系統中的頻寬分配方法，以常見 TCP 流量策略(Additive increase multi-plicative decrease）控制節點之間的連線頻寬，這樣的做法雖可以調節整體系統能力，但是無法顧及到個別節點的負載服務能力。

[24]提出動態群組節點管理方法，藉由 Tracker 來切割過大的服務群組以維護負載的公平性，或合併較小服務群組來提高節點群組的資料吞吐量，然而這機制必須建置入所有的 Tracker 和節點，實際上的部署問題會限制此方法的使用性。

2.3 先前的研究結果

本論文的先導研究[11]針對 BitTorrent 系統進行實地的量測與分析，不同於先前的研究[5][10][12]多是關注於 BitTorrent 追蹤伺服器與 Client 端間的紀錄、BitTorrent 系統中檔案散佈的完整性以及選擇上傳節點的演算法等；相較之下，我們特別以接收端所關心的檔案下載時間為出發點，量測並分析在取得一個完整檔案所需的下載時間過程中，實際使用到的下載節點數之變異，以及這變異對於使用者端傳輸頻寬利用的影響。

研究採行的方式是以實際佈建十組 BitTorrent Client 使用者端裝置，直接下載網際網路中的檔案，從 Client 端擷取的數據中分析 BitTorrent 系統內的行為，並調控不同的量測參數，包括檔案大小、頻寬限制和連結點總數等，揭露 BitTorrent 環境下檔案散佈過程中所潛藏的現象和特徵，包括：

- BitTorrent 僅藉由 TFT 與 OU 機制來篩選與替換節點是相當緩慢，因此需有效的分配節點頻寬與連線數量來達到更好的效能。
- BitTorrent 系統受到網路拓撲不匹配、節點 Churning 和節點頻寬不對稱等之影響，導致下載端的頻寬使用率變動相當大。
- 下載端過度地增加對外連線量並不能有效地提高資料吞吐量。
- 下載端頻寬的遞增無法得到等比例的頻寬使用率。

特別說明，我們的先導研究進行維持約半年的大規模量測，揭露出 BitTorrent 系統下載端之網路資源使用情形，以及 BitTorrent 系統中不同角色之間的互動況；按於篇幅限制，詳盡的量測方法和分析報告留置於甫發表的論文[11]，本文於此僅以代表圖 1 說明：當依照原始 BitTorrent 協定進行檔案下載時，預設的最大對外連線數額度與下載過程中實際被使用到的連線數量兩者之間的變異。

基於[11]的量測研究成果，本論文所擬設計的可調式網路資源分配方法(Downloading-side Network Resource Adaptation Method；簡稱 DsNRA)其設計考量不同於過去諸多研究多是上傳端或系統的角度來控制網路資源的使用。同時，所擬設計的量測指標亦不同於過去研究，我們主要的量測是在於“在檔案下載完成所經歷的時間過程當中，如何以更有效分配連線頻寬和連結到來源節點的數量，以期能改善下載端的頻寬使用功效”。因此，本論文的研究及方法將針對不同檔案下載的頻寬需求，動態調節和分配下載端可使用的連線數量，讓下載端可以根據不同檔案的下載情況有效地利用網路資源，期能最大化頻寬使用率與成本效益。

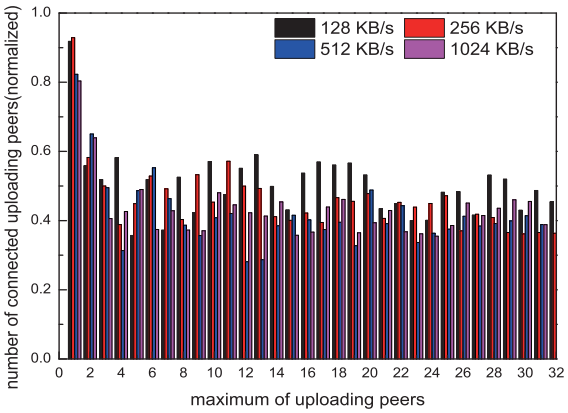


圖 1：實際連線數(Normalized)

3. 動態頻寬分配機制之設計與運作流程

本章節提出一套可整合入 BitTorrent 下載端之可調式網路資源配置方法（DsNRA），此方法可即時地根據當下不同連線的流量情況，動態的替換不同的上傳節點以期能提高資料傳輸量並且縮短整體的下載時間。

3.1 機制設計

誠如我們在[11]所觀察到 BitTorrent 下載端的許多特徵（圖 1 所示），根據 BitTorrent 協定，下載端節點僅被動地接受來自 Tracker 所提供無任何排序依據的上傳節點清單，從中依序或隨機選取節點並與其建立連線；此外，對於下載過程中連線或傳輸品質不良的上傳節點，下載端節點也不會有任何拒絕或終止的動作；再者，也因為 BitTorrent 在下載過程期間不會對連線的情況有所管控，因此下載端節點未能充分發揮其最大的傳輸能力，舉例來說，實際的量測研究發現 BitTorrent 下載端實際建立的對外連線數量幾乎都未達到最大的可使用額度，這是因為 BitTorrent 受限於 TFT 與 OU 機制的被動處理影響，導致替換上傳節點的過程會相當緩慢，為了能有效地使用下載端網路資源，我們的研究在現有 BitTorrent 協定的基礎上加入 DsNRA 機制，以改善下載端在節點量的分配，動態地最大化網路資源使用效益。

圖 2 描繪 DsNRA 機制的功能架構圖，此架構設計可相容於原型的 BitTorrent 運作流程。當使用者在 Torrent Website 獲得有興趣的 Torrent meta-data 描述檔案，經由 BitTorrent 下載端軟件解析 Torrent 取得 Tracker 網路參考位置，自 Tracker 取得上傳節點的 Peer List 清單。DsNRA 機制首先可以從清單中初選出一部分的上傳節點，與其建立連線和下載檔案片段，隨後 DsNRA 機制反覆地持續與 Tracker 互動，獲取更新及更多的 Peer List 清單，同時篩選和替換掉傳輸情況不良的節點，並利用回收或是剩餘的可用連線額度來和新的上傳節點建立連線，持續下載不同的檔案片段。

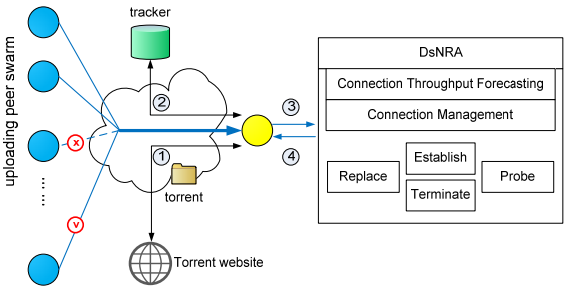


圖 2：DsNRA 機制的功能架構

DsNRA 機制主要包含兩個功能程序，如圖 2 所示：

1) 實際使用的下載連線數量與資料輸出量之預測（Connection Throughput Forecasting）：下載端節點週期性地量測當下實際使用的對外連線數量，亦即真正接觸到的上傳節點數，並量測每一條連線的單位時間內平均資料輸出量和負載情況，藉以反應出不同上傳節點的上傳品質資訊，提供連線管控所需之參考。
2) 下載連線數量之管理與控制（Connection Management）：下載端節點將可依據上述功能所提供的資訊，適時的判斷各個上傳節點的服務能力，並根據持續更新的 Peer List 清單來選擇新的替代節點；同時也可以根據下載端當下可使用或剩餘的網路傳輸資源，動態地調節當下使用的上傳節點數量，期能將有限的連線數適時地分派到符合傳輸效益的上傳節點。

3.2 機制運作流程

圖 3 描繪 DsNRA 機制運作方式和流程，表 1 說明 DsNRA 機制所使用的量化參數。我們在本文此處特別說明，為了能進行實機的網路量測，我們採用常見的 BitTorrent 下載端軟件 Vuze 所開放的原始碼[15]，經過部分修改以便整合所開發的 DsNRA 功能軟件。

DsNRA 運作流程包含兩部分：初始設定階段及下載調節階段，以下說明對應於 DsNRA 實機操作程序。

首先在機制運作的初始階段，使用者可以設定下載節點在本機端可以使用到的最大下載頻寬（M）以及可以建立對外連線數量的最大值（C_max）節點量，緊接著 DsNRA 執行如下的初始工作設定：

- 每秒可允許的連線量：Vuze 軟件是以每秒為一個周期，持續地量測當下已經使用的對外連線數量與可用的最大連線數量之差值。
- 連線超時或中斷：我們在 Vuze 中加入可以強制中斷連線的功能，因此當一部分上傳節點受到網路拓撲變動或是檔案片段傳送發生錯誤等影響時，下載端可經由持續性的連線流量計算，發現某些不良節點，進而中斷超時的連線。
- 更新 Tracker 給予的 Peer List 清單：DsNRA 將持續地維護本機端所擁有的 Peer List 清單，因此 DsNRA 將會驅使 Vuze 軟件很規律、週期性地和 Tracker 主機溝通，要求更新的清單，並且也可以在下載單備用節點數量不足時立即發出要求。
- 開啟下載任務：當一個新的下載任務被啟動時，DsNRA 將根據 Torrent 的資料，與所屬的 Tracker 聯繫並進行下載準備。

表 1：符號表

符號	描述
M	The total download bandwidth (set by the user)
C_{max}	The maximum of outward connections to uploading peers.
C_{before}	The data throughput in the previous duration
O_{now}	The data throughput in the current duration
C_{now}	The number of currently used outward connections.
O_{Δ}	The increment of data throughput.
O_c	The average increment of data throughput per connection
C_{Δ}	The increment of outward connections to uploading peers

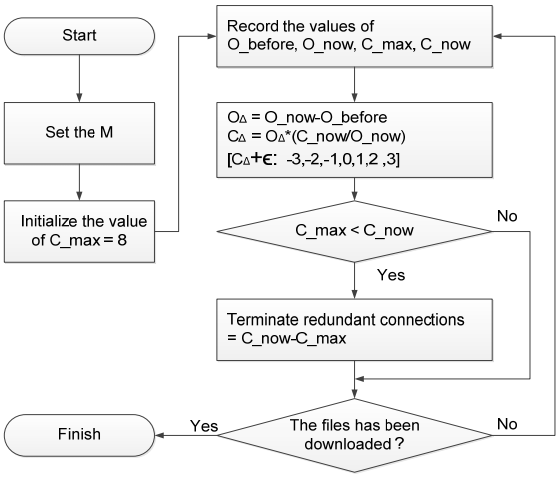


圖 3：DsNRA 機制流程圖

另一方面，在檔案下載過程中 DsNRA 機制將依照實際使用的下載連線數量與資料輸出量之預測資料，動態地啟動連線與頻寬分配的調整程序。原則上，DsNRA 會在每一個周期時間內紀錄儲存“先前”與“目前”的下載頻寬和即時連線量等資料，當下載端具備有這些資料後，DsNRA 機制在檔案下載的執行期間，將可以在不同的階段啟用不同的程序，以下說明三種條件程序（conditional procedures）供 DsNRA 擇一運作：

1) 檔案下載的初期或下載接近完成的末期（初期的連線數量逐漸遞增；末期的連線數量逐漸遞減）：為了避免造成連線數量的明顯巨幅變化，在此條件程序中對外的最大連線數量

並不做任何變化，讓檔案下載在初期與結束末期時可以維持既有、穩定的連線而運作，俾利於量測的觀察與分析。

2) 當即時的下載頻寬量接近使用者設定上限值 M 時，若此時再持續與其他新的上傳節點建立 TCP 連線下載檔案片段的話，將容易違反 M 值的限制，迫使網路流量控制軟體針對所有進行中的對外連線進行節流的控制，反而容易傳輸量不穩定，因此為了避免這樣的情況，DsNRA 機制可以在連線數達到某適度的門檻值之際，刻意地減少幾條不良的對外連線量，這可以讓 DsNRA 機制有機會回收或以剩餘的連線來跟新的上傳節點建立連線，如此，DsNRA 可以逐漸地提高下載端節點的下載頻寬使用效率。

3) 當即時的下載頻寬量距離使用者設定的上限值 M 仍有相當的差距時，DsNRA 可以透過下列三個算式獲得週期性的量測結果，包括以算式 1 計算出資料傳輸增加量或遞減量（O_Δ）、以算式 2 計算現有連線的平均下載資料量（O_c）、另外算式 3 綜合以上兩算式而得到在下一個週期需要增加或減少的連線數量之估測 (C_Δ)。

$$O_\Delta = O_{now} - O_{before}, \quad (1)$$

$$O_c = O_{now} / C_{now}, \quad (2)$$

$$C_\Delta = O_\Delta / O_c . \quad (3)$$

特別說明的是，當 DsNRA 決定下一周期的最大連線數量之後，該數值可能受到負值減量的影響，導致最大連線數少於即時連線數，對此由[11]的量測研究發現連線數量變化的標準差約為正負 3，因此在 DsNRA 的設計上，節點增減量將適度地控制在正負 3 區間，而這樣的設計也便於和原始設計進行效能評比，以避免因為過度地增加或減少同時運作的連線數量而造成比較上的偏頗。

另外，正如前述所提常見的 BitTorrent 下載端軟體如 Vuze 本身並不會即時中斷過多的連線，為此 DsNRA 機制須增設與 Vuze 軟件介接的及時中斷過多連線的程序；值得注意的是，對於中斷連線的方式，DsNRA 設計嘗試兩種不同中斷連線的做法：

- 隨機選擇連線來強迫中斷
- 中斷資料傳輸速率較慢的連線

關於這兩種方法的效能結果，在下一章節中我們將會透過實機量測的數據來呈現兩者的比較。

4. 量測實驗與結果

實驗環境佈建五台硬體規格相同的個人電腦連結在一Gigabit Ethernet高速區域網路，所有電腦皆搭載BitTorrent Vuze Client軟體並結合DsNRA機制軟件，實際連結至外部網際網路下載BitTorrent P2P系統內的檔案，同時間亦以NetLimiter頻寬限制軟體[16]來限制每一下載節點的對外總頻寬。檔案下載過程會以Wireshark網路封包分析軟體[17]收集完全的封包監測紀錄檔(Trace Log)進行後續的過濾與數據分析。因此，檔案下載完成後，我們可以得到DsNRA對下載端的動態調整及檔案傳輸的實測數據結果。

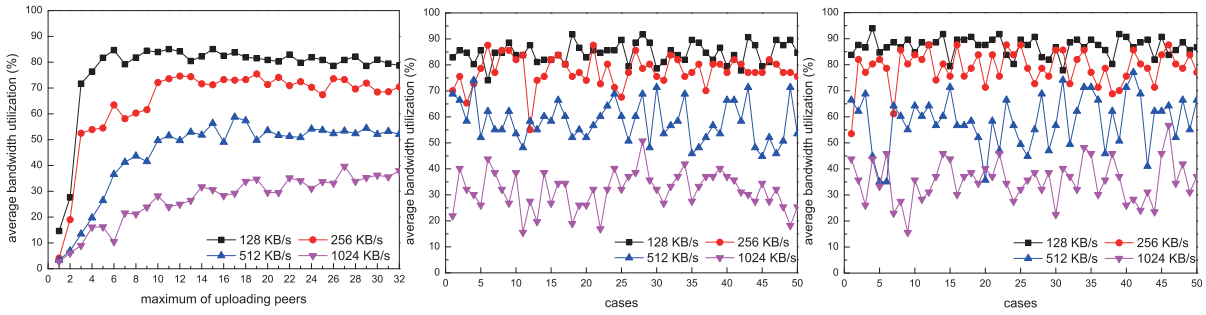
由於整個檔案下載時間會受到下載的檔案大小、下載端的總頻寬和可配置的連線數量等操縱變因之影響。量測實驗過程依序包含採用原型 Vuze、Random 方式中斷連線的 DsNRA (簡稱 DsNRA (Random))以及 Slowness 方式中斷連線的 DsNRA (簡稱 DsNRA (Slowness))這三種模式，之後交叉比對所觀察到的現象分別在不同檔案大小與在不同中斷連線方法之下的效能表現，以凸顯不同方法之特色。因此，整體的量測數據可區分為兩大部分：第一部分是下載一個檔案小於 10MB 的音樂檔，第二部分是下載一個檔案大小約 100MB 的音樂檔，表 2 簡略地列舉出本章節中圖表內容所對應的下載檔案資料，然而礙於篇幅限制，本文此處僅綜合呈現 DsNRA 的效能表現，詳盡的量測數據另放置在網頁 http://mpclab.ce.ncu.edu.tw/tanet2012 以供參閱。

表 2：Torrent 檔案相關數據

項目	小型檔案量測	大型檔案量測
檔案名稱	Fun. - We Are Young (feat. Janelle Monae) {2012-Single} [NL]	Foster the People - Torches [2011-mp3-VBR][TJ]
檔案類型	Audio	Audio
檔案大小	9865.9863 KB	85094.4 KB
上傳節點的頻寬	8192 KBps	8192 KBps
下載節點的頻寬	1024、512、256、128 KBps	1024、512、256、128 KBps
量測時間	2012/05/07-10	2012/05/22-25
量測次數	10	5

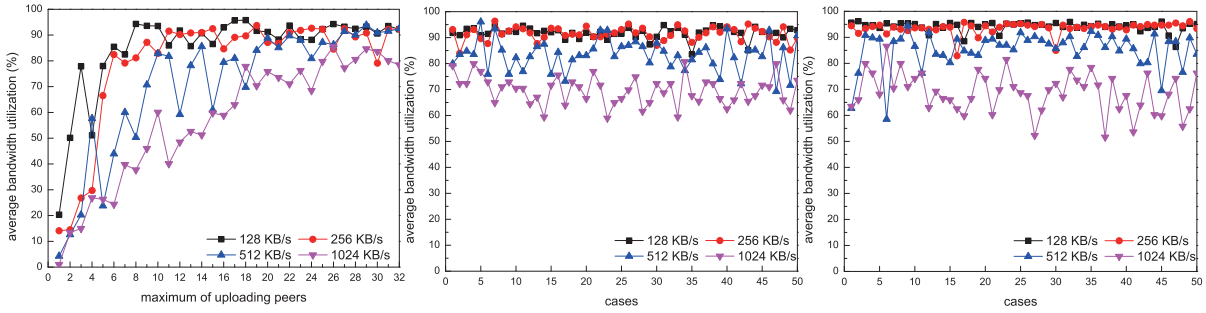
4.1 小型檔案量測

在小檔案量測實驗，下載端對外的下載總頻寬分別限制為 1024 KB/s、512 KB/s、256 KB/s 與 128 KB/s 四種情況，並限制原型 BitTorrent Vuze Client 對外可連結的上傳節點總數分別為 1～32，是以上共有 128 種設定組合，每一組設定值均執行十次求其平均值。實測分析比較原型 Vuze 與 DsNRA 兩方法效能，量化指標包括平均頻寬使用率(Average Bandwidth Utilization)以及實際對外連線數與最大連線數量之差異關係。



(a)origin Vuze (b)Vuze with DsNRA (Random) (c)Vuze with DsNRA (Slowness)

圖 4 小型檔案之平均頻寬使用率(%)



(a)origin Vuze (b)Vuze with DsNRA (Random) (c)Vuze with DsNRA (Slowness)

圖 5 大型檔案之平均頻寬使用率(%)

圖 4 表示小型檔案在平均頻寬使用率上的量測結果，由圖 4(a)中可看出當原型 Vuze 中所限制的上傳節點數大於 8 時，不同頻寬條件下的平均頻寬使用率(意即，使用率越高下載所需時間越短)趨於穩定狀態；相反地在最大連線數小於 8 時，平均頻寬使用率明顯受到節點量的影響，這是因為在系統初始過程中，由 Tracker 所給的上傳節點清單內的節點數量很少，剛開始建立連線時並不會考慮該上傳節點是否能夠提供充足的上傳頻寬(對應於下載節點端的可用頻寬)，此時下載端節點僅能被動地藉由 TFT 和 OU 機制來篩選及尋找其他擁有更好上傳能力的節點，因此平均頻寬使用率的提升速度速度較為緩慢。

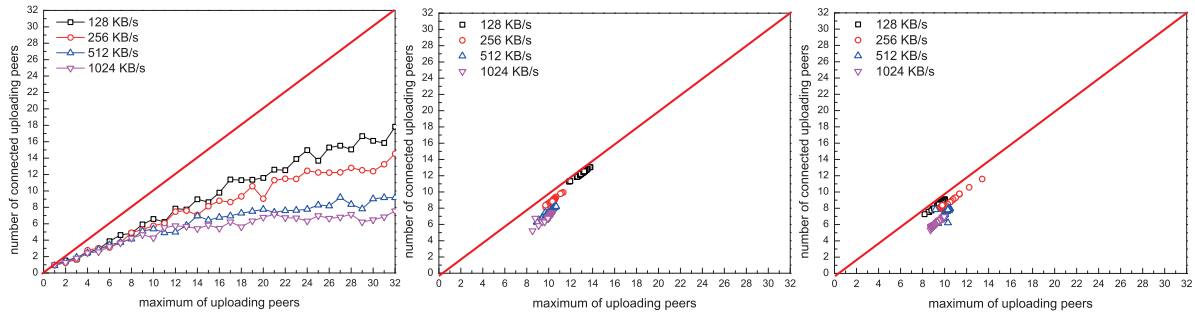
然而，如圖 4(b)和 4(c)所示，當加入 DsNRA 機制後，不論是採用隨機方式刪除節點或是採用刪除最慢速節點的方式，這兩種方法都能在小型檔案量測結果上獲得明顯提高的平均頻寬使用率。其中，對於 512KB/s 與 1024KB/s 的例子而言（相較於 128KB/s 與 256KB/s 的例子），由於每個節點能夠提供的下載頻寬增大，此時頻寬使用率震幅較為劇烈且低於 50%，分析其主因為小檔案所需的下載時間較短（節點從初始至下載結束之間的時間較短），BitTorrent 系統初始化的溝通訊息過程明顯會拖累頻寬使用率，因此 DsNRA 機制的中長期效益不易被突顯出來。

此外，DsNRA 機制的設計著重在縮小「最大連線數」和「實際對外連線數」兩者之間的差距。如圖 6(a)所示，我們可以看到在原型 Vuze 在不同頻寬分配下，隨著最大連線數之額度的增加（亦即可以同時連結之上傳節點量的增加），其所獲得的實際節點量並沒有等比例地提高。關於這一點，理想上兩者之間的比值應該為 1（或是曲線的斜率應該為 1 的直線），而我們觀察到當採用 DsNRA 之後，如圖 6(b)和 6(c)的結果所顯示，在 128KB/s 與 256KB/s 的例子中，DsNRA 量測結果上都緊鄰斜率為 1 的直線，而在 512KB/s 與 1024KB/s 的例子中，則是因為小檔案下載時間較短之影響，致使所有節點尚未完全被使用到之前，檔案即已接近下載完畢階段，所以有剩餘少許的可以連線。對此，在第二部分的大檔案量測中，我們將進一步檢驗 BitTorrent 系統在連線數上的使用程度以及頻寬使用率這兩項指標上的表現。

4.2 大型檔案量測

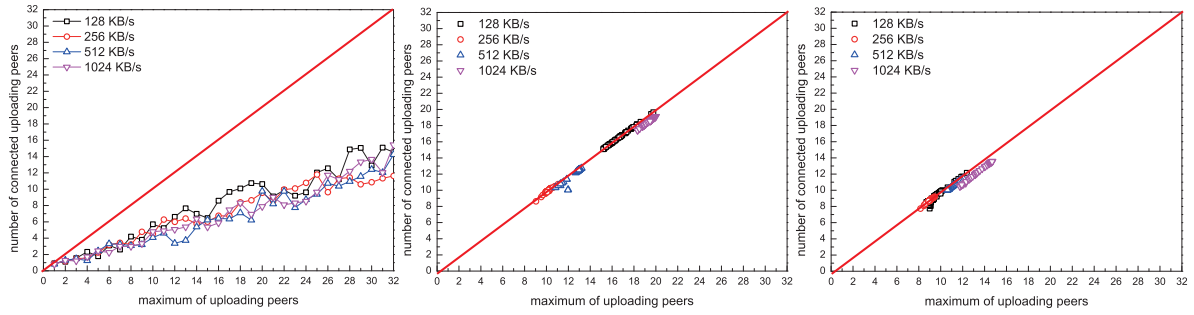
第二部份的量測下載 100MB 的音樂檔，圖 5 和圖 7 分別顯示在原型 Vuze 與包含 DsNRA 兩種機制在平均頻寬使用率和最大連線量與實際連線量差異之效能比較。

如圖 5(a)所示，我們可以明顯看到平均下載時間在節點數限制大於 8 時，已經可以達到穩定的頻寬使用率；而當節點數量很少時，下載容易因為選到上傳能力不佳的節點而導致整個檔案在頻寬使用率上非常的不穩定，此時 BitTorrent 僅能仰賴



(a)origin Vuze (b)Vuze with DsNRA (Random) (c)Vuze with DsNRA (Slowness)

圖 6 小型檔案之最大連線量與實際連線量



(a)origin Vuze (b)Vuze with DsNRA (Random) (c)Vuze with DsNRA (Slowness)

圖 7 大型檔案之最大連線量與實際連線量

OU 機制定期去發掘其他更合適的節點，然而僅僅依賴 OU 機制並不能讓下載端更快速的替換這些不佳的節點。

相較之下，我們設計的 DsNRA 機制可持續地和 Tracker 主機溝通，將可以主動的替換掉不佳的上傳節點，圖 5(b)和 5(c)分別顯示 DsNRA (Random)以及 DsNRA (Slowness)方法的量測結果，對於頻寬的限制越低(例如 128 KB/s 和 256 KB/s 的例子) DsNRA 能夠獲得更為穩定的頻寬使用率(意即，更快地達到所給定的使用頻寬上限)。而當頻寬上限增加為 512KB/s 與 1024 KB/s 時，仍有 10%左右的變動量，這原因或許歸咎於網際網路中的常態性變化或 BitTorrent 網路拓撲的變動等等下載端無法掌握的因素；另一個可能的原因是當檔案的熱門度不高時，其他節點擁有目標檔案片段數相對較少，這也可能導致整體的平均頻寬使用率降低。

最後我們由圖 7 的結果可以觀察到 DsNRA 動態分配節點的使用情況，首先如圖 6(a)和圖 7(a)所示，當下載大檔案時原型的 Vuze 在實際節點數的使用情況亦無法等比例的增加，而在加入 DsNRA 機制後，DsNRA (Random)以及 DsNRA (Slowness)這兩方法所得到的結果之線型都能緊貼斜率為 1 的直線，這說明 DsNRA 機制可以有效率地分配節點提高頻寬使用率。

4.3 效能提升分析

本小節進一步量化小檔案與大檔案的量測數據並探討 DsNRA 效能的優化表現。表 3 和表 4 分別為下載小型檔案和大型檔案的量測結果之綜合比較表，我們從原型 Vuze 在連線數 8~32 的所有量測結果裡面找出平均頻寬使用率為最大與最小的這兩組結果（簡稱為 Vuze (Maximum)和 Vuze (Minimum)）用來與 DsNRA 機制比較所得的結果。我們可以看到在不同的頻寬限制之下，DsNRA 機制能夠以較少的連結數量達到（接近或超過）原型 Vuze 在頻寬使用率上的最佳結果，此外，DsNRA (Slowness)平均使用的節點量更略低於 DsNRA (Random)，這樣的結果可說明 DsNRA (Slowness)之設計優於 DsNRA (Random)。

5. 總結與未來工作

我們審閱過去諸多 BitTorrent 系統與協定的量測報告，知悉過去的研究多是以系統或上傳端的角度來進行實測與分析，但是這樣的結果尚不足以說明 BitTorrent 系統中下載端的特徵、功能和效能。本論文的研究接續我們先前在[11]所彙整的 BitTorrent 下載端實測分析，據所提出一套可相容於 BitTorrent 協定的下載端可調式網路資源配置機制，該機制可以估測下載端與上傳節點之間的連線傳輸品質，並且管理、控制和分配下載端的可用、剩餘和回收的連線資源，進而適時地調節網路資源的使用，以期能最大化下載端的傳輸效益。同時，

我們實作整合 DsNRA 和 BitTorrent Vuze 下載端軟體，並進行實際的量測實驗，數據結果顯示 DsNRA 機制可以有效的控制下載端網路服務資源，獲得較高的成本效益。

表 3 小型檔案效能表

Bandwidth		average bandwidth utilization (%)
128 KB/s	original Vuze (Maximum)	85.07508
	original Vuze (Minimum)	78.57086
	Vuze with DsNRA (12 peers,Random)	84
	Vuze with DsNRA (8.6 peers, Slowness)	87.1
512 KB/s	original Vuze (Maximum)	58.74849
	original Vuze (Minimum)	41.5291
	Vuze with DsNRA (7.5 peers,Random)	58
	Vuze with DsNRA (7.5 peers, Slowness)	58.4
1024 KB/s	original Vuze (Maximum)	39.64919
	original Vuze (Minimum)	21.17528
	Vuze with DsNRA (6.8 peers,Random)	31
	Vuze with DsNRA (6.2 peers, Slowness)	35.5

表 4 大型檔案效能表

Bandwidth		**average bandwidth utilization (%)**
128 KB/s	original Vuze (Maximum)	95.79251
	original Vuze (Minimum)	85.71429
	Vuze with DsNRA (16.6 peers, Random)	92
	Vuze with DsNRA (9.4 peers, Slowness)	94.2
512 KB/s	original Vuze (Maximum)	93.89831
	original Vuze (Minimum)	50.33313
	Vuze with DsNRA (12 peers, Random)	83
	Vuze with DsNRA (10.5 peers, Slowness)	85.3
1024 KB/s	original Vuze (Maximum)	85.49383
	original Vuze (Minimum)	37.70417
	Vuze with DsNRA (18.5 peers, Random)	70
	Vuze with DsNRA (12.4 peers, Slowness)	68.8

參考文獻

[1] J. Buford, H. Yu, and E. K. Lua, P2P Networking and Applications. San Francisco, CA, USA, Morgan Kaufmann Publishers, 2008.

[2] Y. Xiangying and G. de Veciana, “Service capacity of peer to peer networks,” in Proceedings of IEEE INFOCOM’04, vol. 4, pp. 2242–2252, 2004.

[3] Ipoque GmbH, “Internet study 2008/2009.” Http://www.ipoque.com/resources/internet-studies/internet-study-2008_2009, 2009.

[4] A. R. Bharambe, C. Herley, and V. N. Padmanabhan, “Analyzing and improving a BitTorrent networks performance mechanisms,” in Proceedings of IEEE INFOCOM’06, 2006.

[5] D. Qiu and R. Srikant, “Modeling and performance analysis of BitTorrent-like peer-to-peer networks,” in Proceedings of ACM SIGCOMM’04, Sep. 2004.

[6] L. Minglu, Y. Jiadi, and W. Jie, “Free-riding on BitTorrent-like peer-to-peer file sharing systems: Modeling analysis and improvement,” IEEE Transactions on Parallel Distributed Systems, vol. 19, no. 7, pp. 954–966, 2008.

[7] S. Kaune, R. C. Rumi, G. Tyson, A. Mauthe, C. Guerrero, and R. Steinmetz, “Unraveling BitTorrent’s file unavailability: Measurements and analysis,” in Proceedings of the IEEE 10th international conference on peer-to-peer computing (P2P’10), pp. 1–9, 2010.

[8] C.-L. Hu and D.-Y. Chen, “Distributed pairing for file sharing in large-scale peer-to-peer networks,” in Proceedings of the 13th international conference on advanced communication technologies (ICACT’11), pp. 492–497, 2011.

[9] M. Bardac, G. Milescu, and R. Deaconescu, “Monitoring a BitTorrent tracker for peer-to-peer system analysis,” Proceedings of the 3rd International Symposium on Intelligent Distributed Computing, vol. 237, pp. 203–208, 2009.

[10] J. Pouwelse, P. Garbacki, D. Epema, and H. Sips, “The BitTorrent p2p file-sharing system: measurements and analysis,” in Proceedings of the 4th international workshop on peer-to-peer systems, pp. 205–216, 2005.

[11] C.-L. Hu and Z.-X. Lu, “Downloading trace study for BitTorrent P2P performance measurement and analysis,” Peer-to-Peer Networking and Applications, DOI: 10.1007/s12083-012-0146-6, 2012.

[12] L. Guo, S. Chen, Z. Xiao, E. Tan, X. Ding, and X. Zhang, “A performance study of BitTorrent-like peer-to-peer systems,” IEEE Journal on Selected Areas on Communications, vol. 25, no. 1, pp. 155–169, 2007.

[13] J. Choi, J. Han, T. Chung, E. Cho, T. Kwon, and Y. Choi, “Bandwidth allocation for BitTorrent under multi-torrent environments,” in Proceedings of IEEEGLOBECOM’11, Dec. 2011.

[14] G. Dan and N. Carlsson, “Dynamic swarm management for improved BitTorrent performance,” in Proceedings of the 8th international conference on peer-to-peer systems (IPTPS’09), 2009.

[15] Vuze Version 4.6.0.2 (Windows), http://www.vuze.com/.

[16] NetLimiter 3 Version 3.0.0.10 (Windows 32 bit), http://www.netlimiter.com/download.php

[17] Wireshark Stable Release (1.6.1 Windows 32 bit), http://www.wireshark.org/download.html.