4 系統效能分析

為了驗證我們所提出的視訊串流系統的優點，以及實際觀察階層式傳輸在 peer-to-peer 網路中的運作情形，我們在本章提出了數種不同的情境，並以我們的模擬環境來模擬各種情境，得到各種數據。藉著分析這些數據，我們能夠更深入了解階層式傳輸所帶來的效能改進。

4.1 模擬環境說明

在呈現我們的模擬數據之前，首先必須先說明的是整個模擬環境的設定。在我們的模擬環境中，我們採用 100 個節點的 Transit-Stub Topology 作為 backbone，在每個 backbone node 上我們會隨機連結上 0 至 2 個 Leaf Node，因此整個模擬的 peer 數目大約是在 100 個上下。在 peer-to-peer 系統的部分，整個模擬時間長度為 70 秒，。在使用者端的部分，所有的 Leaf Node 均設定為 ADSL 的頻寬大小，其上傳/下載頻寬分別為 512Kbps/2Mbps。在視訊串流的部分，共分為兩個群組及一個 base layer，base layer 的大小為 40Kbps，而每個群組的完整大小(8 層)為 320Kbps，也就是說每一層的大小為 40Kbps；視訊的解析度大小為 512 x 512，每一個視訊串流長度為 3 秒。從這樣的參數我們可以得知，當一個傳送端對一個接收端送出一個群組的完整服務時，上傳頻寬是足夠負荷的。當第二個接收端前

來要求服務，此時若無使用階層式傳輸，則我們需送出 640Kbps 的資料，此時即會超出上傳頻寬之負荷。

我們在整個 peer-to-peer 系統中將所有的 peer 之集合定義為 P，在 P 中我們指定擬觀察之 Server 及 Client 的集合，這部分定義為 P_1；而其餘的 peer 之集合則定義為 P_2，這些 peer 雖不在我們觀察的範圍，但其仍會進行視訊串流的傳輸，對於我們所觀察的 P_1 仍有可能造成影響。這樣的一個集合關係我們可以圖 4.1 來表示：

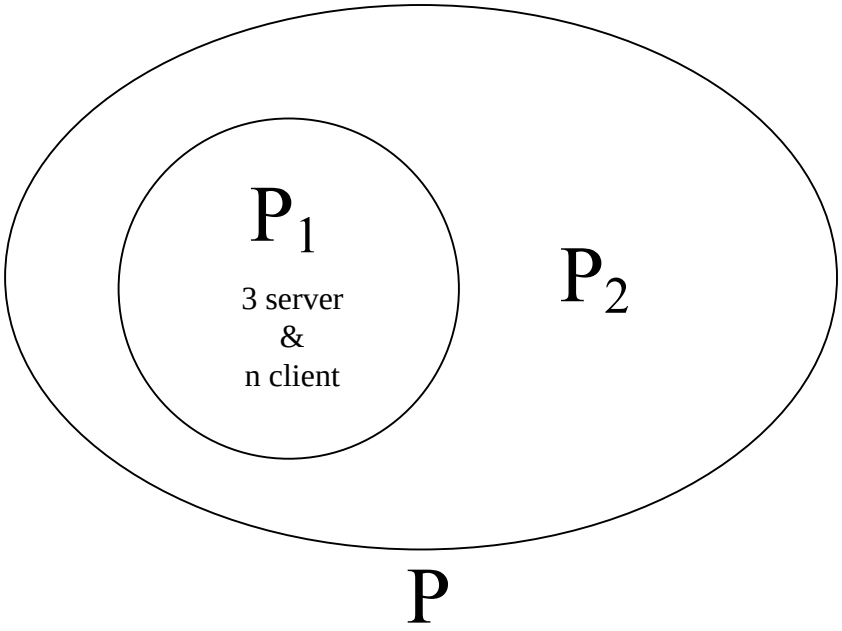


圖 4.1 測試 case 在整個系統中的相對關係圖

由圖中可以看到在整個 peer-to-peer 系統中我們挑選任意的節點作為所指定的角色，P_1 的部分我們固定為 3 個 Server 及 n 個 Client。在模擬過程中，我們在指定的 peer 中，將其 ActivityController 加入一特殊事件，模擬開始時，P_2 的所

有 peer 均按照所設定的機率隨機進行視訊串流，當到達特定時間點，P_1 的接收會在同一時間內發出 Query，當 Query 結束後，P_1 的接收端會對指定的傳送端發出視訊串流的要求。在這個環境中，我們固定了傳送端的數目為 3 個，並以不同的接收端數目(1 至 8)分別下去模擬多次，以得到一個平均值。如此我們即可觀測在不同接收端要求服務之下，階層式傳送對於視訊品質所造成的影響。

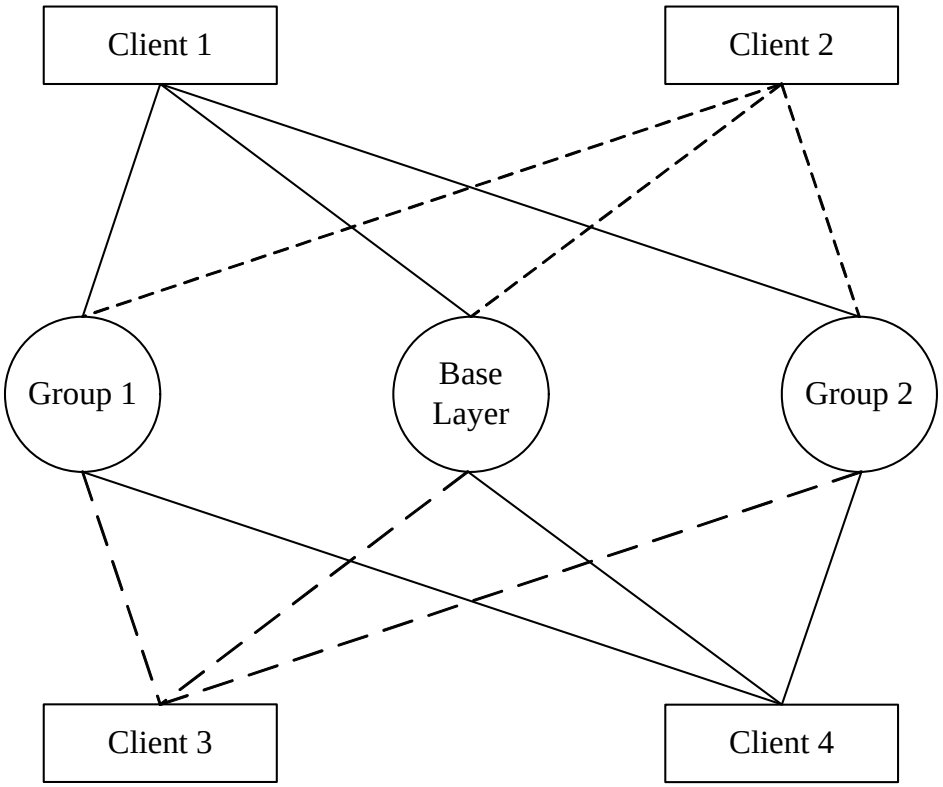


圖 4.2 P_1 中傳送端與接收端之相對關係

圖 4.2 是 P_1 內的傳送端與接收端之間的相對關係，我們以 Client 數目為 4 作為例子，本系統模擬環境中的三個傳送端分別各自擁有一個視訊來源的某一群組。當在特定時間時，四個 Client 會同時發出 Query，而三個傳送端會回傳 QueryHit 給這四個 Client，經過一定時間後，四個 Client 會同時向三個傳送端發

出視訊串流要求。而在我們的模擬設定中，Client 數目為測試的變因。

依據這樣的一個環境，我們將整個 P_1 及 P_2 之內的傳送端及接收端以時間軸的方式來呈現為圖 4.3：

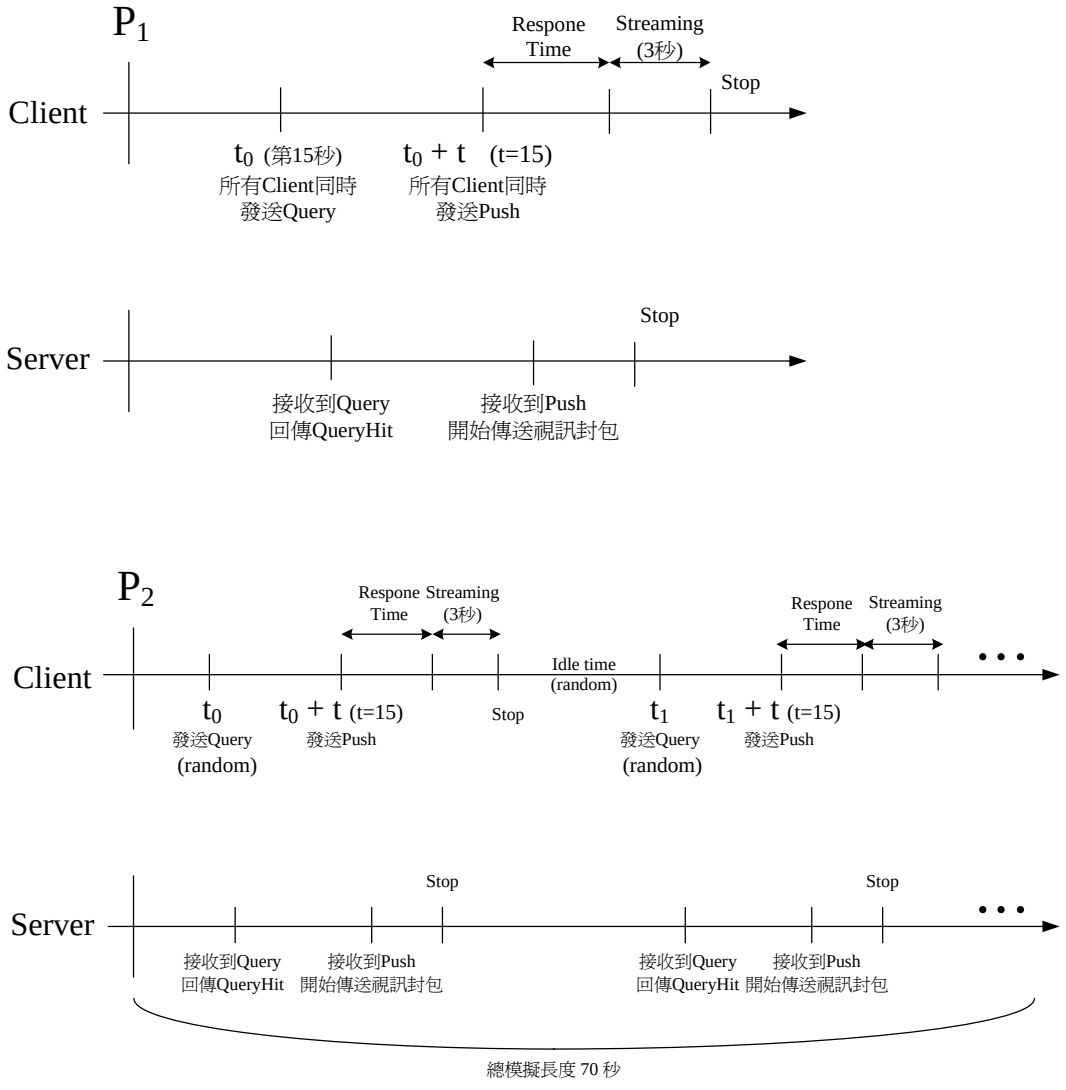


圖 4.3 P_1 及 P_2 的傳送端與接收端之時間軸表示圖

圖中可以看到 P_1 是我們的測試環境，P_1 的 Client 均在第 15 秒時同時發送 Query，經過 t (t = 15)後可得到一 QueryHit List，在其中選擇一最佳視訊來源發

送 Push，經過一定的 Response Time 之後接收到視訊封包，開始進行 streaming。當 P_1 的此一流程結束後即不再重覆，而 P_1 則轉變為 P_2 中的 peer，仍繼續執行一般狀況的模擬。

在整個模擬中，階層式傳輸部分是適用於 P，也就是說，當 P_1 進行階層式傳輸模擬時，P_2 也同時使用到階層式傳輸技術；當 P_1 進行不使用階層式傳輸時，P_2 也同時不使用階層式傳輸。意即在數據分析中，階層式傳輸之使用所影響的是整個系統 P，而不僅是觀察對象 P_1。

4.2 Response time 分析

在本節中我們首先針對本系統的 Response time 進行分析。首先我們先針對 Response time 進行定義。在本節中我們定義了 Request Time 以及 Transfer Time 兩個時間，分別為 t_{req} 以及 t_{tr}，則我們定義

$$\textbf{Response time} = t_{req} + t_{tr}$$

我們以圖 4.4 來說明這兩種時間的定義：

在圖 4.4 我們可以看到一個從 Client Node 向 Server Node 發送視訊串流要求的流程，整個視訊串流流程可參照圖 3.4，上圖即為圖 3.4 的 Stage 2 以及 Stage 3 部分，我們以更詳細的方式來對這兩個階段作解說，並藉此定義 Request Time

以及 Transfer Time 的起始範圍。

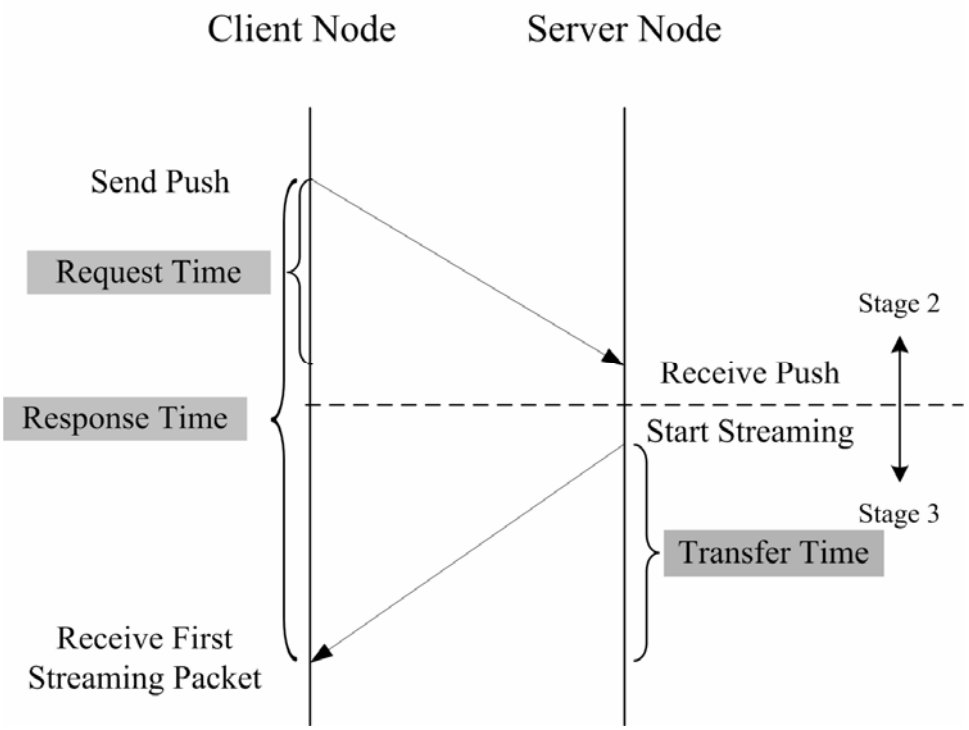


圖 4.4 Request Time 及 Transfer Time 之範圍定義圖

由圖 4.4 中我們可以很清楚的得知 Request Time 的定義為：**從發送視訊串流要求到傳送端接收到視訊串流要求的時間。**在這段時間中包含了視訊串流流程的階段 2。在這個階段中，點與點之間的封包繞送是在 peer-to-peer 網路上所進行的。而到了階段 3，也就是開始傳送視訊串流時，此時是兩點之間的直接連線，已不在 peer-to-peer 的網路之上，兩者間的封包便是直接經由一般網路進行繞送。基於這兩者的不同，我們將階段 3 中，也就是**傳送端開始傳送視訊串流的時間到接收端收到第一個封包止**定義為 Transfer Time。在之後我們將根據模擬所得到的數據來分析這兩種時間在不同服務量下的變化。

表 4-1 即是我們根據 4.1 節所描述的模擬環境，以 1 至 8 個不同的 Client 數目去模擬所得到的數據。

Client 數目	階層式傳送		無階層式傳送	
	Request Time	Transfer Time	Request Time	Transfer Time
1	2.666735702	0.779958826	4.866128182	0.788038119
2	2.982605385	0.867956179	5.274996667	0.878343098
3	3.139809516	0.882275734	5.804629433	0.952623773
4	3.319497869	0.927126459	5.690614828	1.077588698
5	3.475622521	0.969976756	5.778984159	1.114494963
6	3.60276319	0.996591647	5.871178793	1.284335483
7	3.634802051	1.020600752	5.759943404	1.336614117
8	3.790150128	1.040846564	5.379475971	1.395301892

表 4-1 使用階層式傳輸的 Request Time 與 Transfer Time

在上表中我們可以發現在 Request Time 比 Transfer Time 大很多，主要的 Response Time 都花在 peer-to-peer 網路傳送 Push 訊息上。為了方便分析數據，我們將表 4-1 中的 Request Time 繪成圖 4.5：

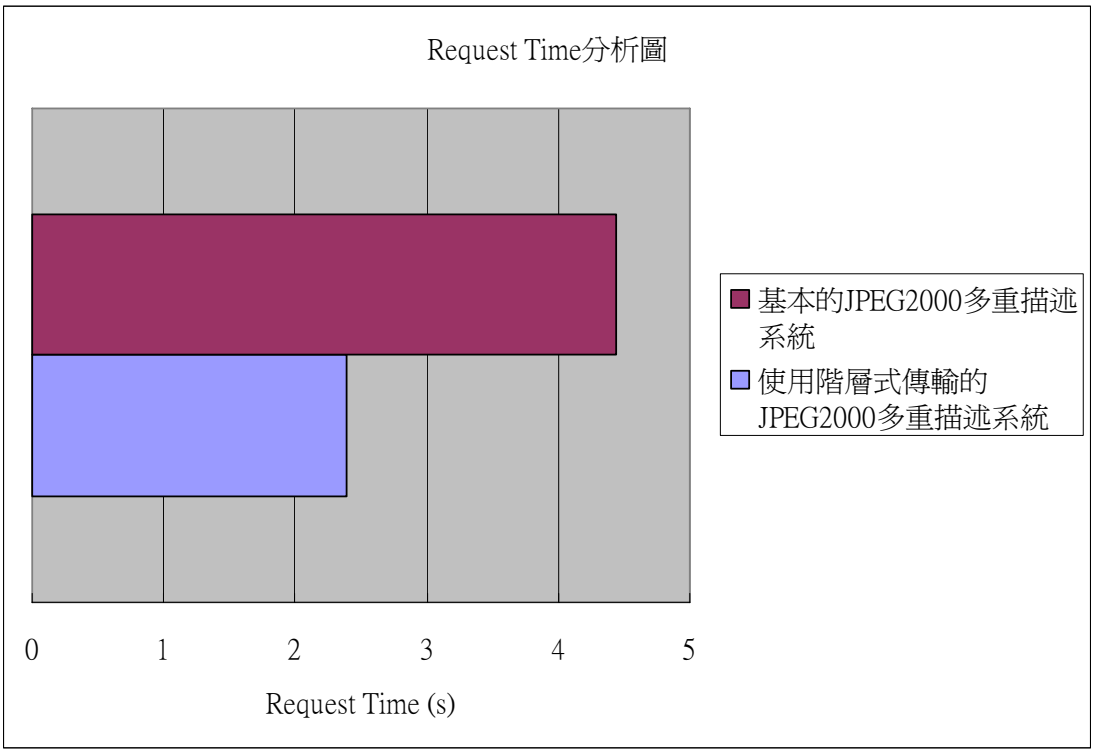


圖 4.5 使用階層式傳輸的 Request Time 分析圖

圖 4.5 是各種 Client 數目平均所得之 Request Time 之分析圖表。我們可以看到在使用階層式傳輸的系統中，整個 Request Time 平均低於沒有使用階層式傳輸的系統約 2 秒左右，其主要原因是整個網路(P)上的節點均在傳送視訊串流封包，當 P_1 中的 peer 要傳送 Push 訊息時，必須經過 P_2 中的 peer 才可送達，而 P_2 中的 peer 有可能正忙於傳送視訊串流封包。當沒有使用階層式傳輸技術時，peer 在同時服務兩個 Client 時即會用盡上傳頻寬，若此時 P_1 內的 Clinet 所發出的 Push 訊息由該 peer 經過，就需要更多的時間等待才可被送出，這樣累積起來，讓接收端發出的 Push 訊息需要經過更多的時間才可到達傳送端，所以圖中使用階層式傳輸的 Request Time 是較低的，這顯示了本系統可以有效的減低整個網路的流量，並降低封包傳送的 response time。

接下來所說明的是 Transfer Time。我們可以從 Transfer Time 的數據中來觀察 Client 數目所產生的影響，經由表 4-1 中我們可以發現 Transfer Time 是較為隨著 Client 數目而成長，這是因為 Transfer Time 所量測的即為第一個封包從發送到被接收所花的時間，這就是完全受到 Server 端的上傳頻寬、Queue 的狀態等因素而決定的，當 Client 數目增多時，封包傳送至接收端便會花費較多的時間。

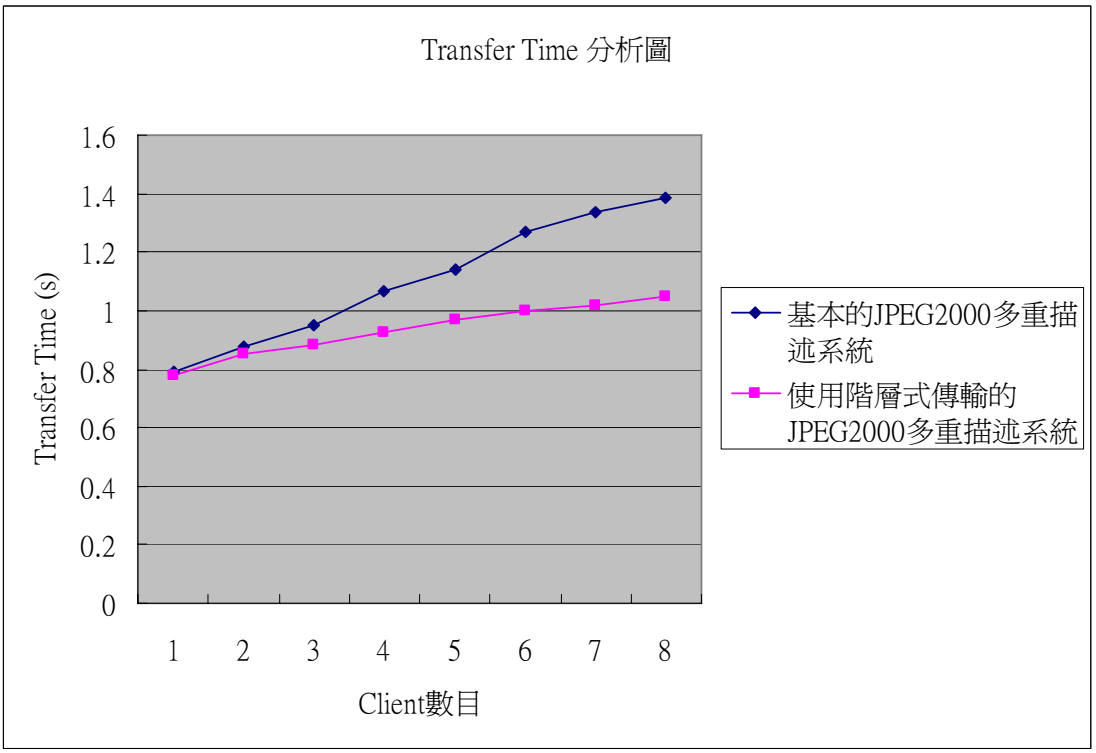


圖 4.6 使用階層式傳輸的 Transfer Time 分析圖

在圖 4.6 中我們可以看到使用階層式傳輸的 Transfer Time 上升相當緩慢，這是因為上傳資料量均能維持在一定程度，不會將上傳頻寬使用殆盡，所以當 Client 數目到達 8 時，Transfer Time 仍舊與 Client 數目為 5 的時間十分接近。在非使用階層式傳輸的部分，我們可以看到 Client 數目為 1 及 2 的 Transfer Time 與使用階層式傳輸之時間幾乎相同，但在 Client 數目變為 3 時，Transfer Time 是

持續的上升，整個成長曲線大於使用階層式傳輸的系統。這是由於當 Client 數目超過 2 個時，上傳頻寬被視訊串流封包所使用完畢，來不及送出的資料均送至 Queue 中等待發送，因此 Transfer Time 會隨著 Client 的數目增加而越來越大。

4.3 PSNR 分析

在串流系統中，最重要的還是服務使用者實際上所感受到的視訊品質。所以在本節中，我們依據前一節所描述的相同環境，實際去分析所有串流封包的接收與否，進而計算出每個 Frame 的 PSNR 值，再將視訊中每個 Frame 的 PSNR 值作平均，得到如表 4-2 的結果：

Client 數目	階層式傳送	無階層式傳送
1	34.53889504	33.19388881
2	32.61103419	32.16669798
3	31.35754839	27.6338773
4	30.60459508	25.36581207
5	29.6287958	24.29930783
6	28.64886897	22.43115172
7	27.4480812	20.50584238
8	26.82654192	19.31889137

表 4-2 使用階層式傳輸的視訊串流 PSNR 值

從上表的結果我們可以看出階層式傳送比無階層式傳送平均高出了 5db 左右，為了方便比較，我們將表 4-2 以圖 4.7 的方式來呈現：

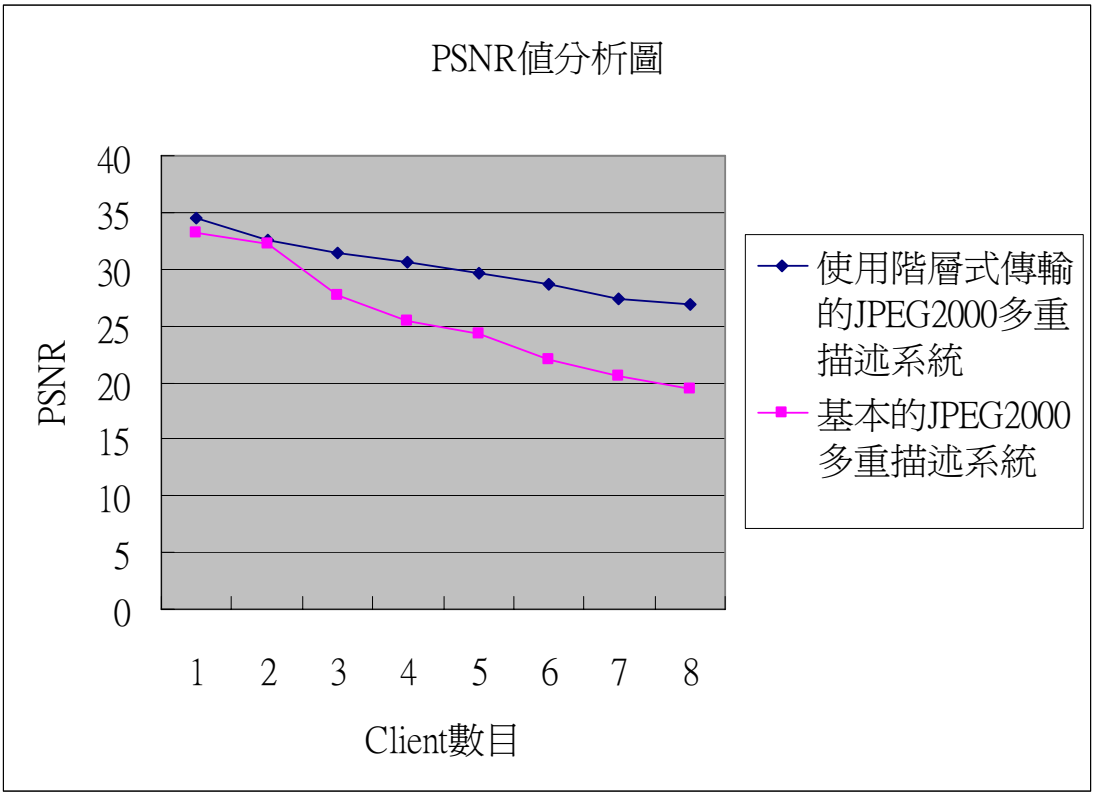


圖 4.7 使用階層式傳輸的視訊串流 PSNR 分析圖

根據前一節所描述的模擬環境可以得知，當接收端越來越多時，沒有使用階層式傳輸的系統此時已超出上傳頻寬之負荷，要送出的大量資料可能超出 Queue 的大小而遭到丟棄，亦或是經過一很長的時間才送至接收端，超出 Playback 時間而視為封包遺失。從圖中我們可以看到在 Client 數目為 3 時，非使用階層式傳輸的系統之 PSNR 值有大幅下降的趨勢，根據這個情形我們可以推測得知當 3 個接收端同時要求服務時，傳送端的 512Kbps 頻寬無法負荷高達 960Kbps 的資料傳送量，造成許多封包無法正確的被傳送，因而造成這種 PSNR 大幅下降的情形。而隨著 Client 的數目增多，所有 Client 所接收到的視訊品質也越來越低。我們可以看到在 8 個接收端要求服務的時候，所有接收端所得到的平均 PSNR 值只有 19.46，根據我們的實際編碼所得到的 PSNR 值，19.46 相當於只接收到了 0

至 1 層之間的編碼資料。實際上我們仔細的去分析整個封包接收的情形，發現在整個接收過程當中，通常是在一開始的幾個 Frame 當中能夠正確的收到所有層數的資料，接著封包即開始大量遺失，這是因為傳送端同時要傳送大量的資料，無法負荷，所以在後面的 Frame 幾乎都接收不到任何資料，我們將以上所描述的情形以圖 4.8 來表示：

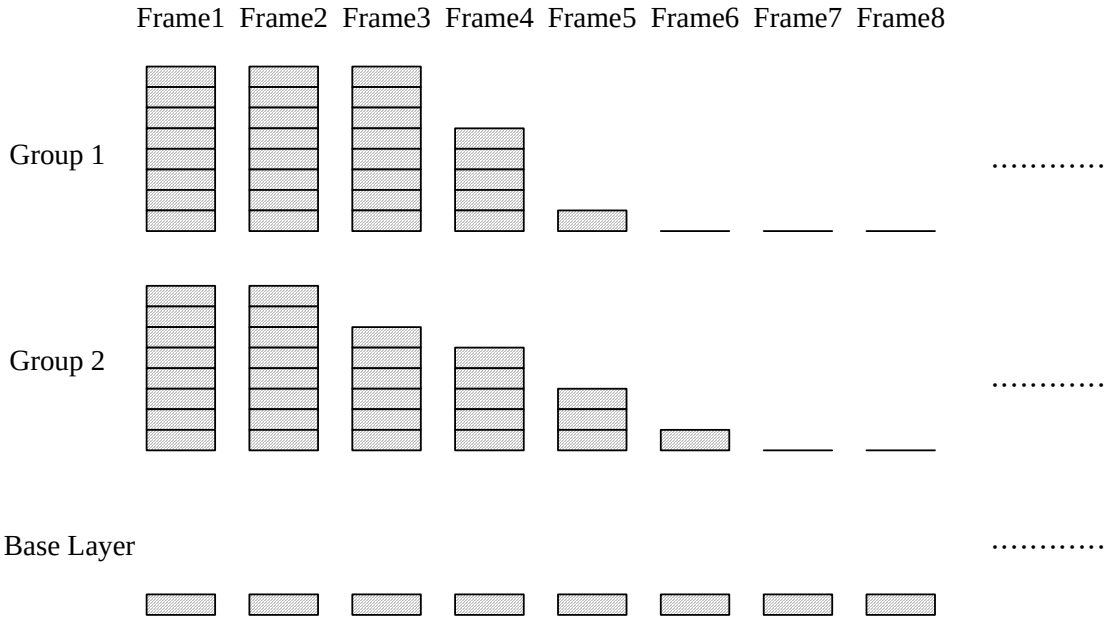


圖 4.8 基本 JPEG2000 系統傳送端大量傳送資料下其接收端接收之資料狀況

在圖 4.8 中我們可以看到一開始的兩個 Frame 中是可以完整的接收到 8 層的資料，但接著便開始有封包遺失的狀況，到第 7 個 Frame 之後就完全無法收到底層的資料了，而雖然均收不到各群組的封包，但由於最重要的 Base Layer 是視為獨立的一個群組，且其僅占一層的資料量(40Kbps)，我們藉此來加強 Base Layer 的重要性，即使在多個 Client 的要求下，Base Layer 仍是幾乎都能夠送到接收端，至少能還原出一個最基本的影像品質。因此經由所有 Frame 的 PSNR 值去平均，

可以得到這樣一個介於 0 至 1 層之間的 PSNR 值，其詳細的原因為此。

反觀有使用階層式傳輸的部分，整個 PSNR 值是穩定下降的趨勢，這樣的品質下降是由於傳送端在傳送時即自行削減服務層數，以維持一定的資料傳送量。我們可以看到在 Client 數目為 8 時，此時每個接收端雖然僅能分得一層的資料，但這一層資料均能夠順利的傳送到接收端，而得到平均為 26.53 的 PSNR 值。所以雖然階層式傳輸在影像品質上作了部分的犧牲，但其能確保傳送端所送出的資料能正確的到達接收端，PSNR 值比起沒有使用階層式傳輸的系統高出了 7db，顯示階層式傳輸編碼法則應用在視訊串流系統上是具有相當程度的影像品質提昇之效用。

4.4 封包流失率分析

在本論文的串流系統中，封包流失具有以下兩種情形：

第一種情形是封包丟棄，這樣的狀況通常發生於網路頻寬用盡，許多封包來不及送出，會先置於 Queue 中等待送出，若是 Queue 的空間也耗盡，則封包會遭到丟棄。在我們的串流架構中，此情形多發生於 peer-to-peer 使用者，在 backbone node(見 3.1 節)中則較少發生，這是由於 ADSL 使用者上傳頻寬不高，若是同時有多個接收端要求提供服務，且未使用階層式傳輸來動態調整資料傳輸量的話，則上傳頻寬很容易達到滿載，而造成封包在傳送端即被丟棄的情形。

第二種情形是封包延遲，這是指在串流系統當中，封包應於視訊播放前即到達接收端，否則應視為封包遺失，即使封包是正確的被送達接收端。在前一段提到封包會被置於 Queue 中等待送出，當等待時間過長，就有可能造成封包傳輸的延遲。

在本節中我們將比較使用及不使用階層式傳送的系統其封包流失率的差異，再針對非階層式傳送的封包流失原因進行分析，進而探討整個視訊串流系統在面對到大量視訊串流負載所可能發生的情形。在本節中所提到的封包流失率代表的是從來源端所送出的所有視訊串流封包其流失的比率。

Client	階層式傳送			無階層式傳送		
數目	封包遺失	封包延遲	總流失率	封包遺失	封包延遲	總流失率
1	0%	0%	0%	0.685%	0.587%	1.272%
2	0.0617%	0.0123%	0.074%	7.084%	1.830%	8.914%
3	0%	0%	0%	17.678%	6.817%	24.496%
4	0.0087%	0%	0.0087%	26.550%	7.049%	33.599%
5	0.025%	0%	0.025%	30.791%	7.984%	38.776%
6	0.0035%	0%	0.0035%	39.074%	7.317%	46.392%
7	0.6394%	0.1948%	0.8343%	50.284%	4.643%	54.926%
8	0.681%	1.033%	1.714%	57.437%	2.613%	60.050%

表 4-3 使用階層式傳輸的 Packet lost rate

在表 4-3 中是使用及不使用階層式傳送的系統其各自的封包流失率。我們可以發現在使用階層式傳送的系統中幾乎是沒有任何封包流失的。為了方便比較，我們以圖 4.9 來表示：

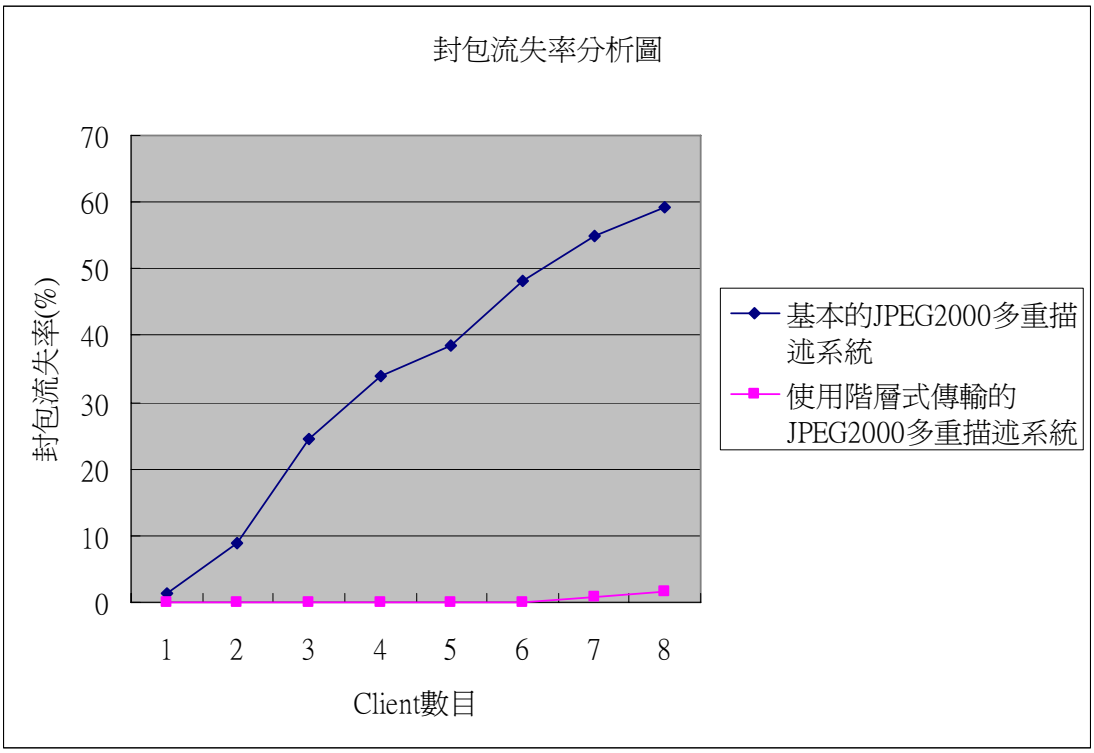
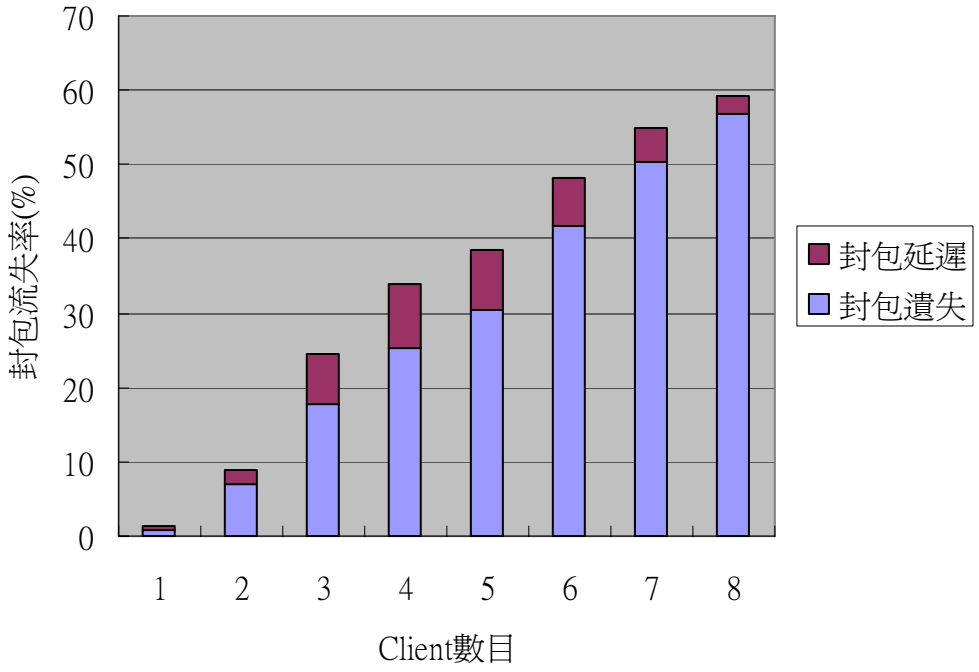


圖 4.9 使用階層式傳輸的 Packet lost rate 分析圖

在圖中可以很清楚的看出使用階層式傳輸的系統在進行視訊串流的過程中幾乎沒有任何封包流失的情形，其最主要的原因為我們的系統在傳送視訊服務時便將服務的層數削減，讓傳送端保持一定的資料傳送量，雖然乍看之下是犧牲了一定的影像品質，但以長遠的角度來說，超過負荷的資料傳送量會使得 Queue 滿載，而滿載之後的新進封包幾乎是無法傳送出去，而會導致一連串的封包遺失。我們的系統即是在避免這樣的情形發生，在傳送端傳送資料出去時便自行將資料量減低，讓所有 Client 都能夠接收到最低品質的封包，而不是說大家都想要高品質的服務，而互相爭搶有限資源，最後使得大家均分享不到任何資源。

封包流失原因分析圖



在圖中我們可以發現非階層式傳送主要的封包流失情形均為封包遺失，大約占了整體封包流失中的 83%，而封包延遲僅占一小部分。這顯示了 Response Time 並不會嚴重影響到視訊串流的品質，我們只要在播放視訊之前先預先保留適當數量的 Frame 作為緩衝，即可大大減低封包延遲所帶來的視訊品質下降影響。主要影響視訊品質的因素仍在於封包遺失的問題，也就是本論文使用階層式傳輸的原因，藉由削減服務層數達到控制頻寬使用的目的，從封包流失率的分析我們可以很明顯的看出我們的階層式傳輸編碼法則是能夠有效藉由降低封包流失率來達到視訊串流品質上升的效果。