國立政治大學資訊科學系
Department of Computer Science
National Chengchi University

碩士論文
Master's Thesis

基於群組模式下之大型網路語音會談系統
A Cluster-Based Transmission Scheme for Large Scale VoIP Conferencing

研 究 生：邵育晟
指導教授：連耀南

中華民國九十八年十月

October 2009

基於群組模式下之大型網路語音會談系統

A Cluster-Based Transmission Scheme for Large Scale VoIP Conferencing

研 究 生：邵育晟　　Student：Yuh-Sheng Shao

指導教授：連耀南　　Advisor：Yao-Nan Lien

國立政治大學

資訊科學系

碩士論文

A Thesis

submitted to Department of Computer Science

National Chengchi University

in partial fulfillment of the Requirements

for the degree of

Master

In

Computer Science

中華民國九十八年十月

October 2009

基於群組模式下之大型網路語音會談系統

中文摘要

利用 VoIP 進行網路會談(VoIP Conference)在現今的社會已有越來越流行的趨勢，尤其是遠距離語音會議的應用方面，不但能節省費用，還能同時允許多人上線會談，但隨著上線人數的增加，網路的頻寬需求與連線數量相對倍增，通話延遲時間也因頻道擁塞而難以控制，此時會議的品質(QoS)往往大打折扣，網路的負擔也會大幅增加。現行的主要解決方案為選擇網路會議中處理能力較強的使用者設備，將來自個別說話者的聲音經過混音疊加後再轉送給其他使用者，再利用不同架構的 multicasting tree 廣播語音封包至所有成員，藉此減少網路頻寬與連線數量的負擔，達到增加同時說話人數的目的。但這些作法在使用者所在位置過於分散、遙遠時，仍然會造成許多節點無法達到所需的通話品質。

本研究針對在大型網路語音會談中，與會人數過多造成通話品質不佳之問題，提出了解決方法，我們假設會議中大部份時間僅有一個說話者，首先為每位使用者加上靜音消除機制，只有發話者的封包直接送給其他所有與會人員，如此不但降低網路負擔，更可以刪除混音的需求，大幅降低傳輸時間，此外並以分群組（Cluster）的方式將位於同一地區的使用者分為同一群組，群組內以 Full Mesh 機制相連，如此便能建立一複雜度較小的 MLDST (Minimum Loss Diameter Spanning Tree) 樹狀結構的 multicasting tree 用以廣播語音封包至所有成員，可以進一步控制傳輸時間及封包遺失率。我們根據使用者之間彼此的連線速度與實際距離來分群組，從群組中選出連線能力較佳的節點作為群組轉送點(Cluster Head)，透過此轉送點收到聲音後做群組內的發送，並藉由一個 multicasting tree 將封包在嚴格控制延遲時間及封包遺失率之下，將語音封包廣播至其他所有 Cluster Head，以確保 VoIP 網路會議之品質。

我們在 PlanetLab 這種實際網路的實驗平台上進行實驗，評估本方法的效能，實驗結果顯示我們的方法可以在與會人數達到 20 人時，仍能有效控制延遲時間在 350ms 左右，若利用 Google 雲端運算系統的協助，平均延遲時間將可減少 50ms 至 70ms。由於 PlanetLab 上的電腦性能普遍不佳，我們預期本技術應用於實際網路時，可以增加與會人數。

A Cluster-Based Transmission Scheme for Large Scale VoIP Conferencing

Abstract

VoIP conferencing is becoming more and more popular in modern life, especially for long distance conferences. It can save a great deal of cost and traveling time. However, the increasing number of online users not only induces an explosive growth of bandwidth demand, but also reduces the quality of voice. Traditional multi-party conferencing systems select those computers that have larger capacity to combine the voice streams from all participants and to distribute the aggregated voice stream back to all participants by various multicasting schemes. Although these solutions can alleviate the burden of network bandwidth demand, the extended long service time remains the biggest obstacle to the improvement of voice quality.

This research proposes a Cluster Multicasting Tree (CMT) aiming to increase the size of VoIP conferencing with acceptable voice quality. Assuming that there is only one speaker in the conferencing for most of the time, CMT applies a silence suppression mechanism to block non-speech voice streams to reduce the number of voice streams injected into the network. CMT employs a special multicasting tree allowing each participant broadcast his/her voice directly to all other participants. By eliminating the need of voice aggregation (mixing), the service delay can be greatly reduces. Furthermore, CMT reduces the zigzag effect existing in the transmission paths by dividing the participants into clusters according to their physical distances. Each member of a cluster can broadcast its voice stream directly to all others members in the same cluster. CMT then employs a multicasting tree called MLDST (Minimum Loss-Diameter Spanning Tree) allowing the head of each cluster to forward and multicast the voice streams generated by cluster members to all other clusters. MLDST is able to control the transmission time as well as loss rate.

We evaluate the performance of CMT on the PlanetLab testbed against a few existing VoIP conferencing schemes. The experimental results show that the service delay can be effectively controlled under 350ms when there is no more 20 participants in a VoIP conference. With the assistance of Google cloud service system, the average delay can be reduced by 50ms to 70ms. In reality, the performance of the computing devices on the PlanetLab testbed is generally far behind their counter parts in the real world. Therefore, we anticipate our transmission scheme will be able to allow more participants in a real world VoIP conference.

誌謝辭

在這兩年多來的研究中，首先我最誠摯感謝的人就是我的指導教授連耀南教授，老師總是能在細節中找到大學問，「魔鬼藏在細節裡」更是老師耳提面命的座右銘。從老師身上我不僅學習到做學問的方式，更在老師身上看到積極的處世哲學，讓我未來的道路必定能更加平坦。而本論文的完成亦得感謝蔡子傑和張宏慶兩位教授的教導和提點。因為有你們的幫忙和討論，使得本論文能夠更完整而嚴謹。

兩年多來的實驗室生活，充滿著學術研究的討論、同學們間的情感，非常感謝卡比在各方面對我無私的幫助，以及推薦我好耳機，智賢學長對我的論文提出許多有價值之建議，小 P 學長總是在我需要幫助時適時出現，怡萱學姐在我研一時常常關懷我，諭祺往往也在不經意之時，提出絕妙的思路解決我的問題，玉潔常幫助我留心各種小細節，也會搞笑，啟禎學長是最好的 Leader，帶領實驗室一起出征，還有山高在口試時幫我跑了許多地方，筱慈、小 M 也為實驗室帶來朝氣。當然其他間實驗室的同學，文卿、東諺、文捷、國淵及其他學弟妹等也不能忘記，有你們的幫忙、搞笑、陪吃飯讓我銘感在心，祝福將要結婚及追求另一半的各位一切順利。在這裡再一次感謝我的指導教授連耀南教授，謝謝老師。

最後，謹以此文獻給我摯愛的雙親及妹妹，謝謝你們 。

邵育晟 台北 October 2009

目錄

圖目錄

表目錄

第一章　緒論

1.1大型網路語音會談

近年來，為響應全球減碳運動，並節省差旅費，許多公司內部召開之大規模跨國會議均採用網路會談方式進行。通常在網路會議進行時，並不會特別採購新的硬體設備與租用網路服務，而是採用現行的終端設備(如 PC，PDA 或 Laptop 筆記型電腦等等)、VoIP 軟體，例如 Skype [20]、Google Talk[16]等軟體，而網路頻寬也多採取現有 ISP 公司所提供的服務(通常為 ADSL 或 3G 速率，通常為下行頻寬 2~3Mbps，上行頻寬 384Kbps 左右)。目前常見的網路會議都是在此架構上進行多人的 VoIP (Voice Over Internet Protocol) 會談。雖然雙人對話的 VoIP 之品質與傳統電話相去不遠，但在進行大型網路語音會談時，由於上線人數增加，耗用網路資源也大幅增加，隨之所必須面對的挑戰也更為艱鉅。本研究希望在不影響通話品質下，盡可能提高與會人數的上限。

1.2研究動機及方法

採用 VoIP 進行大型網路語音會談的主要一個問題就是頻寬需求會隨著與會人數增加而上升導致品質下降，此外，在大型網路語音會談中，回音及噪音干擾之機率也大幅增加。由於這些問題存在，導致提高與會人數成為一艱鉅的挑戰。

目前相關研究中有多種利用 P2P 來傳送串流資料之方式[1][4][5][12][14]，也發展出多種多人發話系統[8][10][11]，多著重於如何提高同時發話人數的上限。但多半的會議中，同時多人發話的機率很低，而且同發話時的語音內容通常視為無效語音，因此本研究假設

多數情況下只有單一發話者，並為每位使用者加上靜音消除機制，只有發話者的封包能送給其他與會成員，因此不需要混音節點，由此建立一傳輸機制，在不影響通話品質下提高與會人數。

1.3VoIP 原理

VoIP 運作方式是將使用者的聲音訊號包裝為訊框(Frame)，再透過網路封包(Packet)送至對話的對象，並且以相反的程式取出資料，將數位訊號轉為類比聲音[3]，如圖 1。

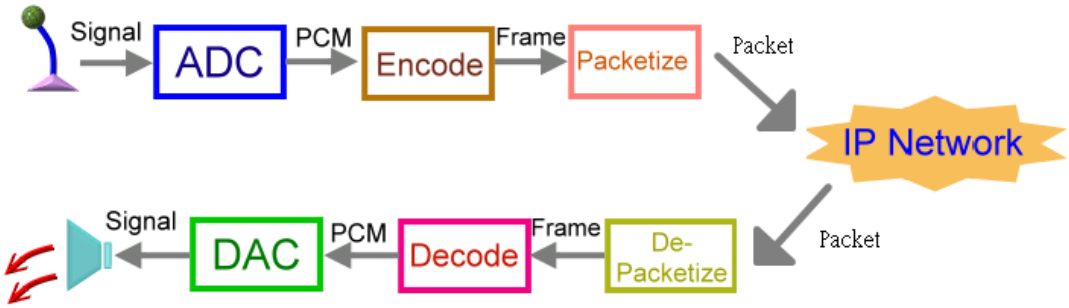

圖 1：VoIP 原理

1.3.1 取樣及編碼

人類說話的聲音是空氣的震動，經過麥克風擷取後，會成為具有強弱幅度變化的電壓信號(Signal)，此訊號是一種連續的類比信號，無法直接被數位系統辨認與處理。因此需要經過類比至數位轉換(Analog-To-Digital)的過程，轉為數位訊號。此過程中最重要的過程即為取樣(Sampling)。

取樣的過程為：將電壓訊號由 0 至最高電位根據編碼長度細切為許多等級(即 level，例如以 8Bits 編碼，則可細分為$2^8 = 256$個等級)，並且根據取樣頻率(Sample Frequency)，每隔固定時間抓取一次電壓訊號，判定其電壓等級，並且以數位資料表示。

接著再將數位訊號傳送給編碼器進行編碼。VoIP 所用的編碼通常會加入壓縮的動作以減少頻寬的消耗。 編碼器的效能是影響 VoIP 品質及效能的一重要因素。壓縮的執行，必須從音訊串流中，取得一段段的資訊再進行壓縮，再輸出一段段壓縮過的資訊[3]。

1.4混音

當利用 VoIP 進行多人通話，同時有多位使用者發話時，為了將聲音傳送給系統中所有收聽者，上傳/下載頻寬需求遠比一對一通話來的可觀，聲音播放時序之問題也較一對一通話來的複雜。因此系統利用混音處理（mixing）將多位發話者的聲音合而為一，如圖 2。當混音器（mixer）收到各個發話者傳送過來的聲音封包後，必須先將封包解開（depacketization），再將編碼過後的訊框解碼（decoding）還原為 PCM 串流，接著再將多個聲音串流疊加（superposition），得到混音後的聲音[3]。

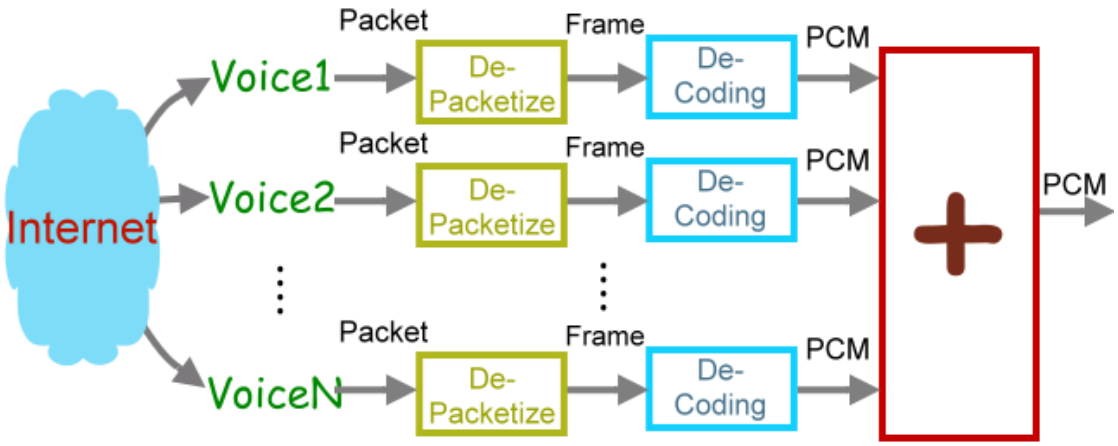

圖 2：混音程序

疊加後的聲音包含所有發話者的聲音資料，系統對其編碼（encode）並包裝成封包之後，就可以將此混音過的聲音封包傳送給其他收聽者，如此減少頻寬的需求並使得語音能

更加的同步。但也因編碼、解碼的過程，並依照其壓縮方式的不同，每經過一個混音的節點時將會增加 30ms 以上不等之時間，如表 1 所示[9][17][18]。

表 1：編碼對延遲時間之影響

Codec	Frame Size	Frames/pkt	Packetization	Encoding	Total Time
G.723	30ms	1	30ms	17.5ms	47.5ms
G.729	10ms	2	20ms	15ms	35ms

混音程序可以在接收端或網路端進行。接收端收到多位發話者之聲音封包後進行混音處理，再透過音效卡播放讓使用者同時聽到多位發話者聲音，如圖 3。

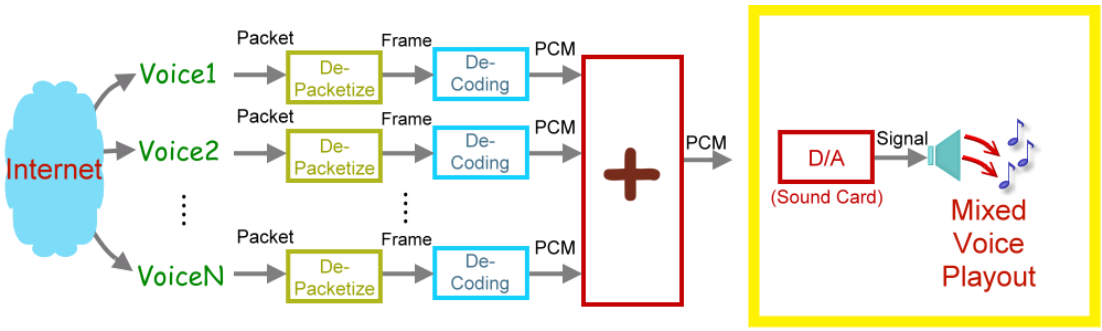

圖 3：接收端之混音程序

雖然此方式僅增加一個 mixing delay，但會談中每位接收者都要接收來自其他使用者的聲音，頻寬需求極大，每個接收端都需有足夠的下載頻寬及系統資源來處理所有發話者的聲音封包。另一種方式則在網路上的節點進行混音。在大型網路會談中，挑選系統資源較豐富的電腦做為混音節點，由其將混音後的聲音封包傳送給其他與會成員，但每個混音節點都會增加一段延遲時間（mixing delay + retransmission delay），如圖 4。

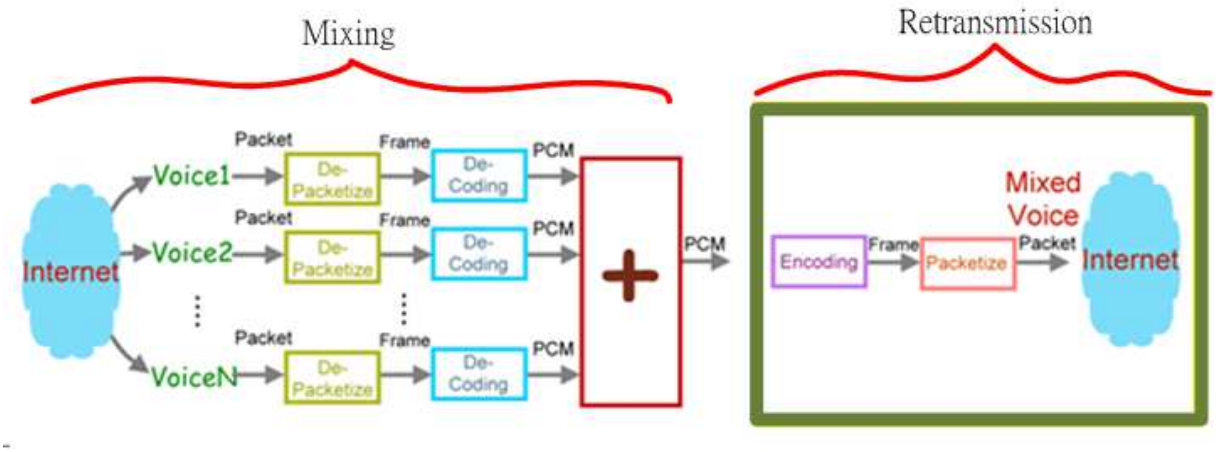

圖 4：網路節點之混音程序

1.5影響 VoIP 通話品質之參數

語音由說話者傳至收聽者所需時間會因上述各種狀況的變動而受影響，例如不同編碼器(Codec)具有不同的單位訊框時間，不同的網路條件也會造成封包傳遞的延遲(Latency)、封包遺失(Packet Loss)、Jitter、回音(Echo)、噪音(Noise)等情形，特別諸如此類問題皆可能影響 VoIP 的服務品質(Quality of Service)。

封包傳輸的延遲、遺失與 Jitter，可能讓使用者在通話時感覺聲音斷斷續續、聲音播放時序（Playout Scheduling）之錯亂等，影響到通話雙方之表達與接收。另一方面，使用者所在的環境也不盡相同，麥克風可能會接收到許多環境存在的噪音，例如工地施工的噪音、旁人看電視的聲音等等而干擾通話。而發話端所說出的聲音，到接收端的喇叭播放後，有可能被接收端的麥克風再次收音，而使得發話端聽到自己的聲音，而造成回音的產生。除此之外，VoIP 中收聽者無法由說話者的表情、肢體動作等來判別發話者的語意，常因前述因素降低溝通的效率，甚至使得會談無法進行。

1.6語音會談常見問題

當參與會談人數增加時，許多網路會談的問題就顯得更為嚴重。例如頻寬需求量，連線人數限制，封包遺失率或者聲音品質等等因素，以下提出幾個常見的問題：

- 總頻寬需求與 stream 數量問題：當使用者增加時，stream 數目隨會議成員增多而呈N^2成長。若與會人數為 N，則總共需要存在的 stream 數量為$2 * C(N,2)$，當連線人數上升至某個程度時，此數量即相當可觀。
- Echo/Noise epidemic 問題：當參與會談人數增加時，產生回音或噪音的機率也隨之增加。且回音會有遞迴的效應，即回音本身還會產生回音，如此反覆產生回音。圖 5 係不同與會人數及不同 echo/noise 產生率下，probability of echo/noise free 的關係圖。若單一使用者故障率為 20%，則十人參與會談時，沒有回音存在的機率僅為$(1-0.2)^{10} = 0.107374$，約為 10%，由此可見參與人數越多時，能夠不受回音干擾成功進行會談的機率越低。

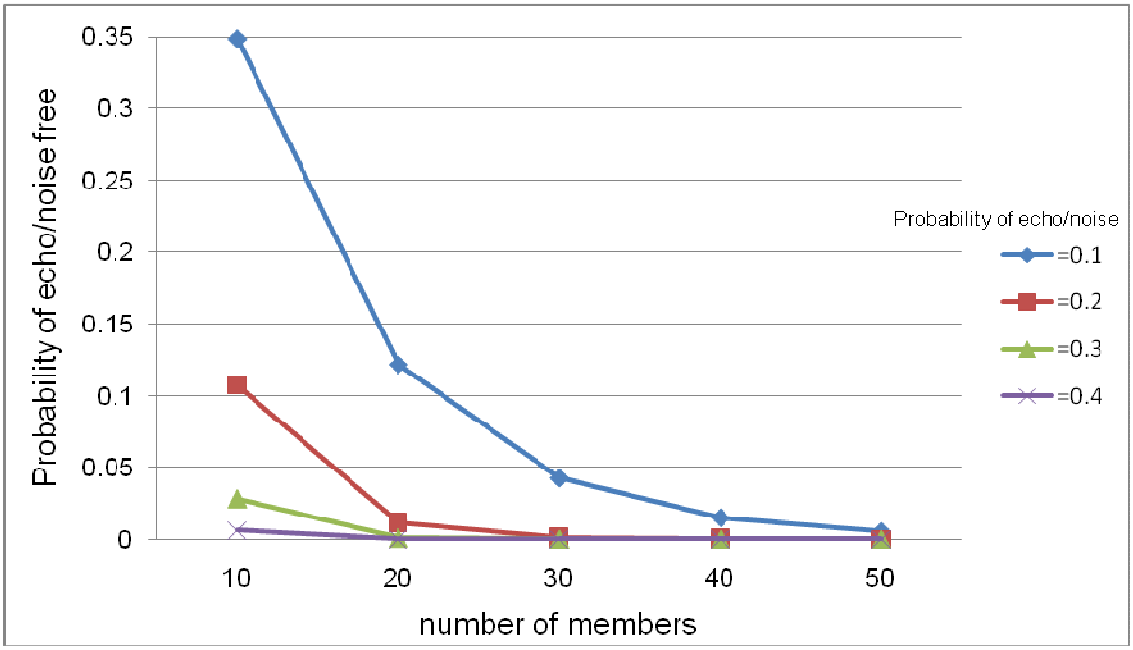

圖 5：Echo/Noise Epidemic 與會談人數之關係

- Time sequence disorder 的問題：亦即聲音不依照發送順序到達接收端，而發生先發言卻較慢被聽到的狀況。例如 A 成員比 B 成員更早發言，但 B 成員的聲音卻比 A 成員的更早被收到。

諸如以上問題，都是有可能在大型網路語音會談中所發生的，而針對以上種種問題，目前分別都有提出一些在應用程式端或網路架構上的解決方案，整體的概觀可由表 2 說明，△代表這種解決方法能有效改善該問題，◇代表此方法有機會改善此問題，但不排除加重此問題的可能性，▽則代表此方法對此問題有負面影響：

表 2：會議通話常見問題及解法（Client Side Solutions）

VoIP Conferencing Issues and Solutions		Key Issues							
		Bandwidth Consumption	No. Of Streams	Echo/Noise Epidemic	Time Sequence Disorder	Delay	Packet Loss	Jitter	Voice Quality
Client Side Solutions	Silence Suppression(VAD)	△	△	△					
	Echo Cancellation			△					△
	Adaptive Sampling Rate	◇							△
	Adaptive Packet Payload	◇				◇			
	Redundant Voice Streams (without Piggyback)	▽	▽				△		△
	Redundant Voice Streams (with Piggyback)	▽					△		△

表 3：會議通話常見問題及解法（Network Side Solutions）

VoIP Conferencing Issues and Solutions		Key Issues			
		Bandwidth Consumption	No. Of Streams	Node Stress	Delay
Network Side Solutions	Full Mesh Broadcasting	▽	▽	▽	△
	P2P Multicasting	△	△	△	▽

針對網路會談所發生的種種問題，在使用者軟體(client)端與網路(network side)端各自有一些解決方案，以下分別進行分析。

Client Side Solutions

1.VAD：此方法能夠偵測聲音訊號中實際是否包含語音資料，假設 VAD 偵測到一個聲音訊框中並不包含任何有意義的語音資料，那麼就不將這個訊框傳送至網路上。通常在網路會談中一次只有一位發話者需要上傳資料，其餘使用者僅需接收，如此即可節省頻寬使用並減少連線數量。同時 VAD 也能夠避免一些不必要的雜音或回音混入會談聲音中，提高會談的品質。

2.Echo cancellation：此方法專門用來消除與會者麥克風收到喇叭聲音所造成的回音(acoustic echo)。通常採用聲波相減的方式抵銷回音訊號，加入此方法能夠有效減少回音遞迴對音質與會談品質造成的干擾。

3.Adaptive sampling rate：採用可變動的取樣頻率進行聲音的數位化轉換。越高的取樣頻率可以得到越佳的音質，但同時也會提高資料量。因此在網路資源允許時(有足夠頻寬可使用時)採用此方法，提高取樣頻率，即能夠提昇會談音質，當網路資源不夠時，可採用較低的取樣頻率，得到較低品質但資訊量較少的語音封包。

4.Adaptive packet payload：每個網路封包內通常可裝載數個聲音訊框(frame)。若將一個封包裝滿，雖可提高網路的使用效率，但卻可能增加聲音的延遲(delay time)。相反地，若為了節省時間而將每個訊框裝載在一個單獨的封包中時，聲音即時性較好，但卻降低封包的使用率(增加 overhead)。因此採用可變動的方式，根據網路可用頻寬調整每個封包內裝載

的訊框數量，頻寬足夠時，則降低 packet payload，反之則將多個訊框放於同一個封包內以節省頻寬。

5.Redudant voice streams (without piggyback) ：網路會談進行時，語音資料均採用 UDP 封包傳送，因此並不能保證封包必然到達，一旦封包遺失，可能對聲音品質造成影響。此方法將同一語音封包以多個網路串流(stream)重複傳輸數次以降低封包遺失率(packet loss rate)，提昇音質。但此方法必然會提高頻寬使用量與串流連線數量，對網路造成較大負擔。6.Redudant voice streams (with piggyback) ：將同一個語音封包傳輸多次減少遺失率，提昇音質。但此方法不建立額外的串流連線，而是將前一個語音封包附加於下一個封包內一併傳輸，如此雖網路頻寬使用量會提高，但卻不用增加串流數量。

Network Side Solutions

為了解決網路會談的相關問題，在網路傳輸架構方面，有別於傳統需要伺服器協助轉送資料的 centralized 架構，相關研究提出了兩種不需伺服端的傳輸架構(serverless)。這兩種方法分別為：

1.Full Mesh Broadcasting：參與會談的節點之間直接建立連線，每位使用者在說話時都將語音封包直接傳送給會談中的所有與會者。此方法需要耗費相當大的網路頻寬與連線數量，對於每一個節點而言負擔相當的沈重，但由於資料不需轉送，故此方法最大的優點為延遲時間較短。

2.P2P Multicasting：此方法根據網路會談中每個節點的資源，每位發話者建立一個用來傳輸聲音資料的 multicasting tree， 用以廣播封包到所有其他成員。此方法使得發話者僅需將資料傳送給下一個轉送點，對於每個與會者而言需要建立的連線數量通常也較 Full

Mesh 架構少，整體節點負擔較輕。但其最大缺點為封包必須經過多個轉送點，延遲時間必然增加。

1.7語音會談品質評量指標

- MOS

一般常用來衡量 VoIP 通話品質的指標為 MOS（Mean Opinion Score）[19]，如表 3。

表 4：MOS 定義

MOS	Quality	Impairment
5	Excellent	Imperceptible
4	Good	Perceptible but not annoying
3	Fair	Slightly annoying
2	Poor	Annoying
1	Bad	Very annoying

分數由高至低分別代表品質最好至最壞，分數的計算是由特定的發話者與聽話者在特定的環境下，透過收集測試者在各種不同情景下的主觀感受，再根據建議的分析法則得出該語音的品質。

- E-Model

E-Model 概念是將語音信號傳輸過程中，對話質造成影響的負面因素綜合成一參數 R，來評估該通話的客觀品質 [21]。R 值越大表示品質愈好，R 值由下公式計算而得：

$$R = R_o - I_s - I_d - I_e + A$$

o R_o表示噪音的影響，如環境雜音、雜訊的干擾。

o I_e表示傳送語音訊號時所產生的干擾，如 PCM 量化失真。

o I_d表示延遲造成的品質影響。

o I_e表示引入其他設備造成的響，如不同編碼器對封包遺失的復原程度等。

o **A**為期望數值，當R_o扣除掉其他負面因素後，依照使用者設備、環境之不同對R_o補償，例如傳統電話 **A**值為 0，在市區大樓間使用行動電話**A**值為 5。

E-model 評分方式由訊號品質，扣除其它負面因素並依照其設備、環境加上期望數值**A**求得通話品質分數。

- Number of members

與會人數多寡為語音會談中一可客觀量測之指標，計算方式為在不影響使用者通話品質下，該語音會談網路架構同時能容納多少使用者參與此會談。

- Service delay

在語音會談中，封包延遲時間包含傳送延遲時間（propagation delay）、排隊等候時間（queueing delay）、編解碼所需時間及經過的節點數所花費時間。因此[10]中定義 service delay 為「從發話者發話至收聽者聽到聲音所需時間為 service delay」。

大型網路語音會談中，參與會議人數可能分散於世界各地，然而隨著會談人數增加，頻寬需求也隨之上升，分散世界各地的使用者讓延遲時間難以控制，本研究之目標在保證 service delay 下盡可能提高與會人數，我們提出一語音會談網路架構，依照使用者位置資訊將其分組，群組內使用 Full Mesh 機制連接降低傳送延遲時間，並配合 Maximum

Engergy Tracking Voice Activity Detection (VAD) [6]技術阻擋非發話者的聲音，如此當任一使用者在會談中沒有發話時，便不會傳出聲音封包佔用頻寬。再以群組為輔來建立一個 Minimum Loss Diameter Spanning Tree 用來廣播封包到其他所有 Cluster Head，透過此傳輸機制，期望在不影響通話品質下，盡可能提高與會人數上限。

第二章　背景與相關研究

目前在大型網路語音會談的應用方面，各種相關研究提出了不同的網路傳輸架構，常在重疊網路（overlay network）上利用 Peer-to-Peer Multicasting 或 Centralized Server 兩種架構來進行網路會談。overlay network 是建立於底層實體網路之上，在 overlay network 上的每個點是以 logical links 相連，任兩點間的路徑實際上可能是藉由數條底層網路的連線所構成，如圖 6，在本章節中，將針對各架構的原理以及優缺點分別進行論述與比較。

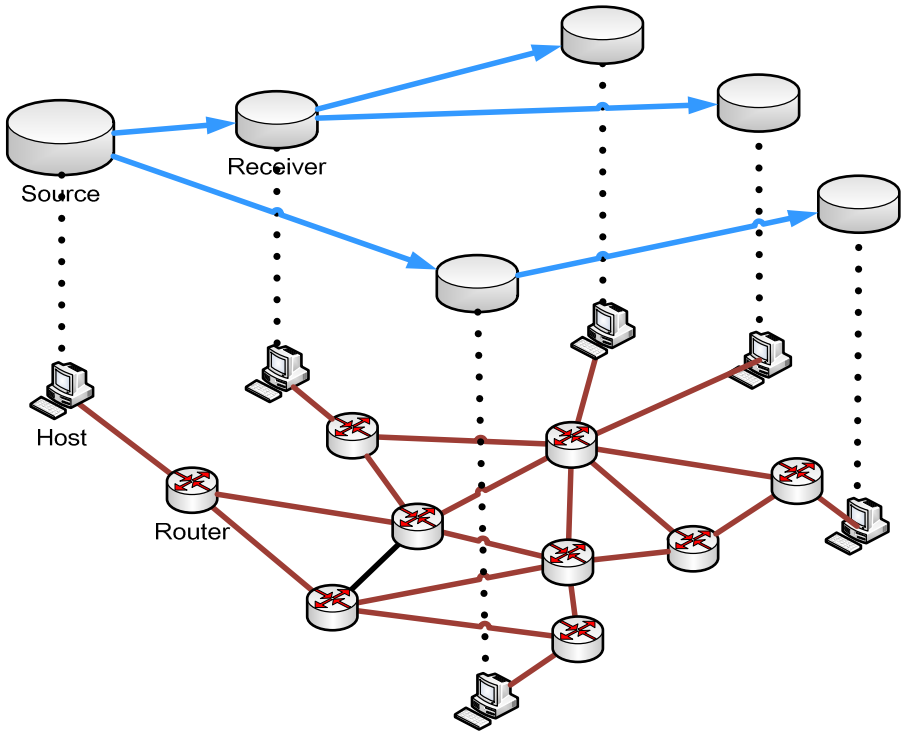

圖 6：Overlay Network Architecture

2.1語音會談網路架構

2.1.1 Centralized Server

其概念為在系統中建置一部伺服器，而每位參與會議的使用者將各自語音封包送往此伺服器，再由此伺服器將語音封包做混音處理並發送給所有使用者（如圖 4 架構）。此作法之優點在於能減少上傳頻寬，每位發話者僅需將聲音傳送給伺服器而不需發送給系統中所有收聽者，但付出的代價為此伺服器須要有足夠的負載能力（網路頻寬，運算效能等等方面），伺服器也無法負擔同時多個會議通話之進行；同時，若參與會議的某一使用者與伺服器之間連線能力不佳時，無法藉由其他使用者給予協助（例如：作為中介者，轉送來自伺服器的訊息），此架構在 large scale 會議環境中，對距離伺服器較遠的使用者極為不利。比起發話者直接將聲音傳送給其他使用者，此架構多了伺服器的 mixing delay，以及伺服器將聲音傳送給其他使用者的傳播延遲時間，而且信號所行經的距離大大增加了 propagation delay time。以圖 7 中的跨國會議通話為例，伺服器位於美國，則台灣的每一位使用者都必須個別透過長距離的 Internet 與伺服器通訊，不但網路負擔大，且穩定性也不佳(如 latency/jitter 等)。除此之外，當發話者及伺服器將聲音傳送出去或者接收等動作時，都必需做編碼、解碼等動作，皆需花費 30ms 以上之不等時間。定義系統中，節點將聲音編碼（Encoding）所需時間為 T_e，將聲音解碼（Decoding）所需時間為 T_d，當任一使用者發話後，並由伺服器轉送給其他收聽者所需之編碼/解碼之時間為

$T_e + (T_d + T_e) + T_d = 2 \cdot (T_e + T_d)$，易造成伺服器負擔太重，增加延遲時間，若伺服器故障時整個會議系統將無法使用。

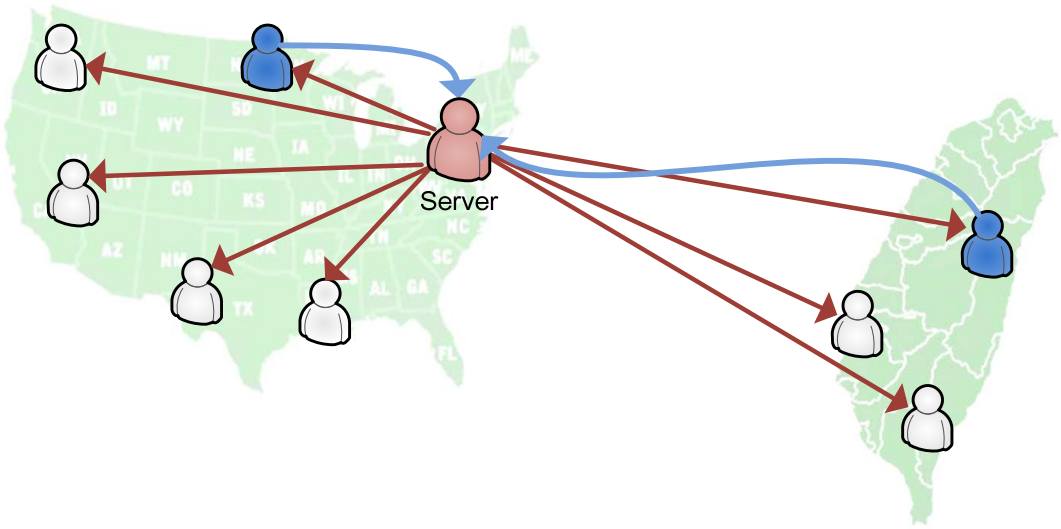

圖 7：VoIP Conferencing By Centralized Server

2.1.2 Full Mesh Broadcasting

在此模式下，每位參與會議的使用者兩兩互相連接，發話者將語音封包傳送給參與會議的其他使用者，如此即使沒有伺服器也能夠進行會議通話，且不需要混音程序，因此可省下混音延遲時間及 propagation delay time，而在會談進行時，若有一個或數個節點突然離開時，也較不會影響到其它節點 [13]。但每位使用者至少須付出$N * 8kbps$（N 為使用者個數）的上傳/下載頻寬來傳送接收聲音封包。

但隨著參與會議通話的使用者數量增加，每位使用者的網路負擔都會加劇，若與會人數為N，則總共需要存在的 stream 數量為$2 * C(N,2)$，如此即限制了參與會議的人數，且網路的總流量極為龐大。此架構如圖 8 所示，因為發話者直接將聲音傳送給使用者，所以編碼/解碼所需時間為$T_e + T_d$。

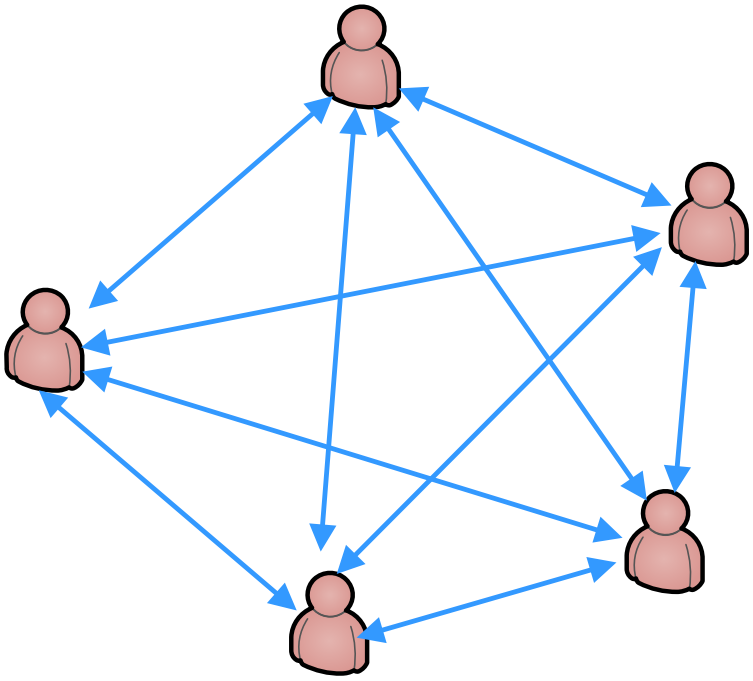

圖 8：VoIP Conferencing By Full Mesh Broadcasting

2.1.3 Coupled Distributed Processing

此方式的概念為，每台伺服器皆負責一定數量之使用者，當有使用者發話時，僅需將聲音傳送給附近的伺服器，由其做混音，再由此伺服器將聲音傳給更上一層之伺服器，與其它聲音再做混音，如此建立一個 tree，最後由 root 負責將所有資料做混音處理，再利用同樣的 tree 將聲音發送給系統中所有使用者 [2] [15]。如圖 9，發話者 A、D 兩位將聲音傳送給 M6 做混音，再由 M6 傳送給上一級的 M2，最後交由 root M0 把所有聲音做混音處理，得到 A+D+F+H 聲音，再沿著同樣的 Tree 將此聲音廣播給所有的使用者。

此架構雖能減少伺服器之負擔，有效的減少每台伺服器上傳/下載頻寬的需求，但是每做一次混音就必需花費一段 mixing 的時間，因此整體延遲時間將會大幅增加，而 VoIP 通話中使用者對於延遲時間的容忍最高為 350ms，因此此方法常造成嚴重的延遲問題。

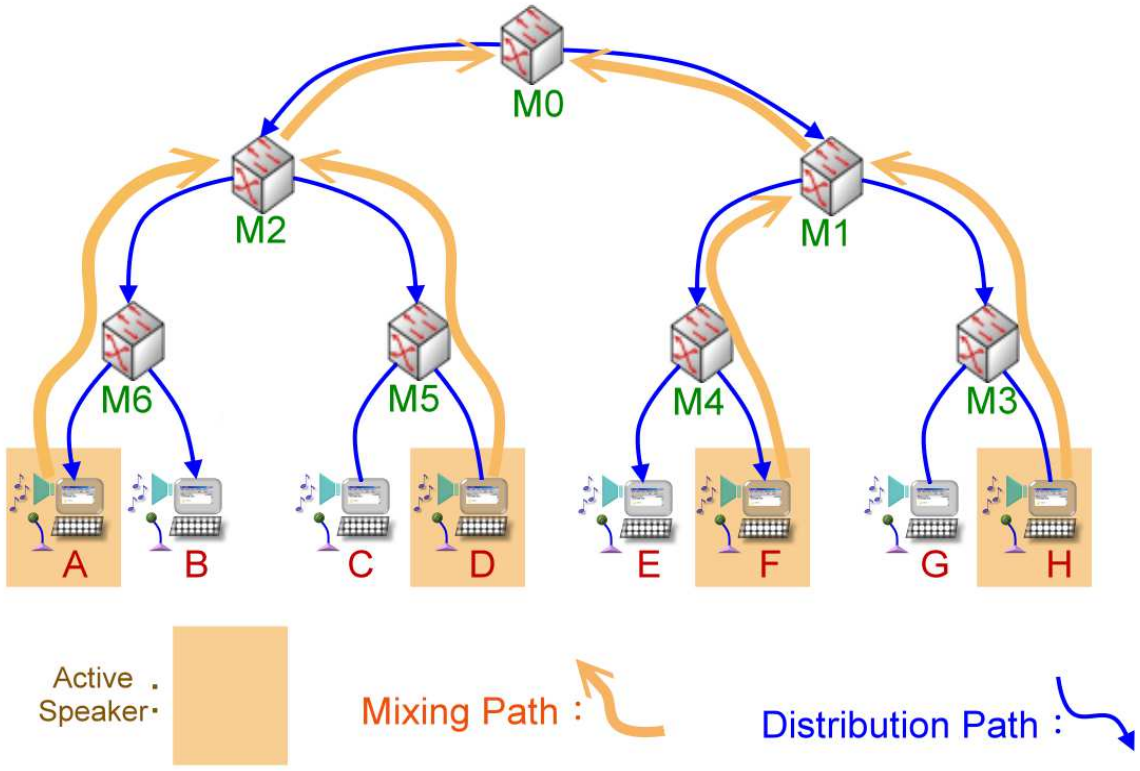

圖 9：VoIP Conferencing By Coupled Distributed Processing

2.1.4 P2P Multicasting

圖 10 為多人通話利用 P2P Multicasting 傳輸之情形。當任一使用者發話時，發話者必須自行將聲音傳送給鄰近使用者，再由這些使用者將聲音傳送給其它人，如此建立一個 Multicasting Tree [8]。發話者會優先選擇延遲時間較少的節點當作轉送節點，將發話者的聲音封包轉送給其他使用者或是轉送節點，以此方式建立一 P2P Multicasting Tree。

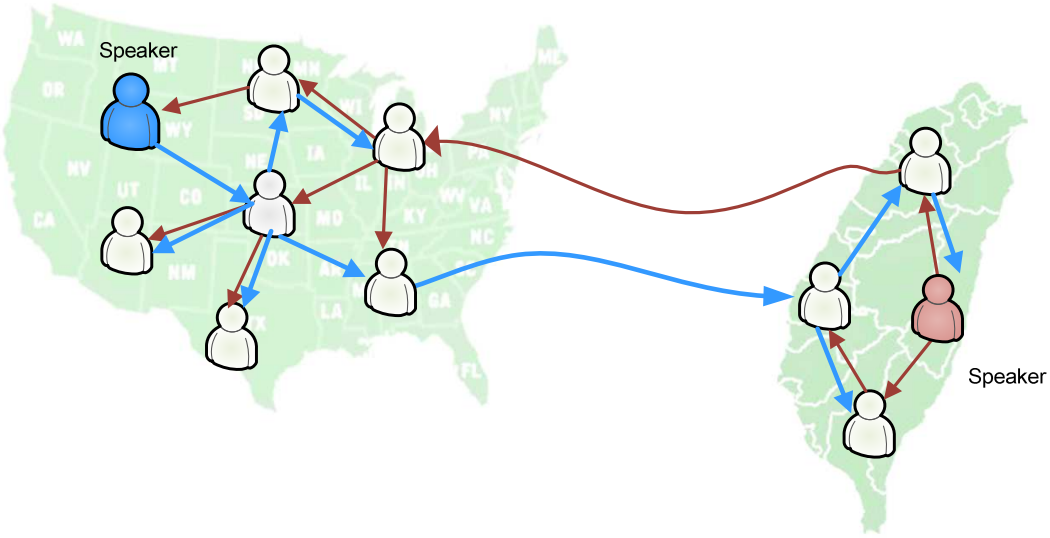

圖 10：VoIP Conferencing By P2P Multicasting

此方法不需要對網路層做任何更動，並能有效減少發話者傳輸聲音串流所需之頻寬。但由於每位發話者皆需建立一個 multicasting tree，因此隨著說話人數增多，系統中網路的負擔將隨之增長，且若與會人數過多，multicasting tree 長度可能會太長，造成不佳的通話延遲時間。

2.1.5 peerTalk: A Peer-to-Peer Multi-Party Voice-Over-IP System

peerTalk 為一個多人通話系統，特別適用於多人同時說話的環境。peerTalk 從 Coupled Distributed Processing（CDP）演變而來，與 CDP 不同的是，peerTalk 採用兩個不同的 mixing、distribution tree (multicasting tree) ，可進一步減輕中間節點的負擔，當同時發話人數增多時，mixing tree 節點個數隨之增加，此種作法能有效降低系統中節點負擔。

系統初始時，系統中每位成員先測量彼此間的延遲時間，選擇一延遲時間最小的節點作為 mixing、distribution tree 的共同 root，並先建立一個 distribution tree，每當同時說話人數增加時，再動態新增 mixing tree 的節點個數。

peerTalk 先進行混音程序（mixing），之後再將混音後的 stream 傳給所有與會者（distribution），如圖 11。在圖 11 中發話者有四位，發話者會將自己的聲音傳送給鄰近的 mixer（M1, M2），先做第一次混音，接著再由 M1, M2 兩者把聲音傳送給 root mixer M0 (4)。

peerTalk 在說話人數非常多時，之所以能有不錯的效果，主要是因其採用 dynamic mixing tree，能隨著說話人數之不同來變動 mixing tree 的高度，有效的減少單一節點負擔，mixing 結束之後，才交由 distribution tree 將聲音傳送出去，但是若在同時說話人數較少的情形下，peerTalk 仍需維護 mixing、distribution tree，依舊無法擺脫進行混音程序所需的時間以及轉送聲音所需的 propagation delay，因而增加過多的延遲時間而影響通話品質。

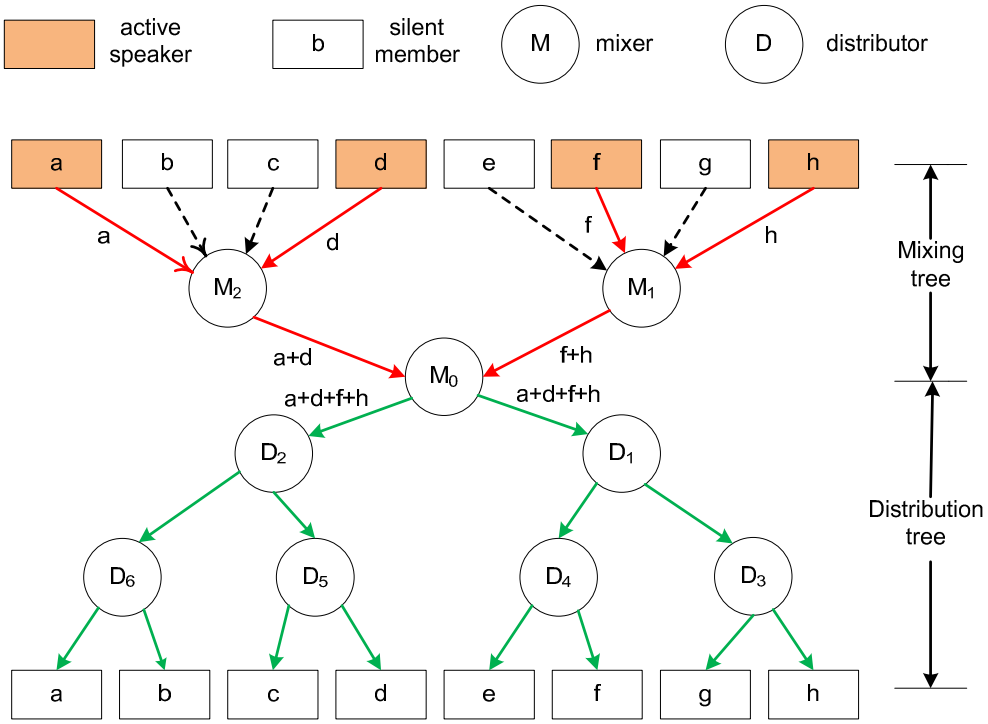

圖 11：VoIP Conferencing By peerTalk

2.2 現有傳輸機制比較

表 5：現有架構分析比較

P2P Multicasting	• Maintain a large number of multicasting trees for all participants. • Overloaded by processing many audio streams concurrently.
Centralized Server	• Has a large overhead.

	• Hard-controlled delay. • Less scalability.
peerTalk	• Mixing phase. • Delay Time.

P2P Multicasting 在大型網路語音會談中，multicasting tree 數量的增長所隨之增加的網路流量可能給系統帶來過大的負擔，multicasting tree 的高度可能過高而影響延遲時間；Centralized Server 無法應付過多的使用者，且發話者要將聲音傳送給伺服器，再由其傳送給其他使用者，大幅增加了 propagation delay time；peerTalk 雖然較其它方式更能有效分擔負載，但仍未能擺脫混音程序所增加的 mixing delay 及 propagation delay time。

第三章　Cluster-Based Multicasting Scheme

本章節將詳細介紹我們所設計的「基於群組模式下的傳輸機制」（Cluster-Based Multicasting Scheme），包括設計理念、細部的設計方法及適用範圍。

3.1設計理念

一般會議進行時若與會者有兩位以上時，由於人與人之間的溝通會話是一種語音一來一往不斷交替的訊號，人在說話時，語句與語句之間比不會緊密的連接在一起，而會有中斷的間隔存在。且多人同時說話的機會極小，即使多人同時說話，通常也因無法有效溝通而必須循序重講一次才能順利溝通。

因此本研究假設會議中大部份時間僅有一位說話者，而採用 P2P Multicasting 模式，並為每位與會成員加上靜音消除機制，阻擋非語音封包進入網路，只有發話者的聲音訊框才會藉由網路封包傳送給其他與會成員，如此能有效的降低網路負擔，避免 P2P Multicasting 方法的缺陷。此外，本方法不需要混音節點，可以省去混音延遲時間。

除此之外，過去所提出的大型網路語音會談架構中，在 overlay network 上所建構的拓樸架構並沒有考慮到節點彼此間的實際距離，如此建構的 multicasting tree 各段鏈結(link)可能在實際網路上是一條很長的路徑。圖 12 為一個在 overlay network 上建構的樹狀結

構，與圖 13 相比可以看出，封包可能在不同國家間反覆來回傳遞，導致傳遞路徑過長而浪費網路資源。

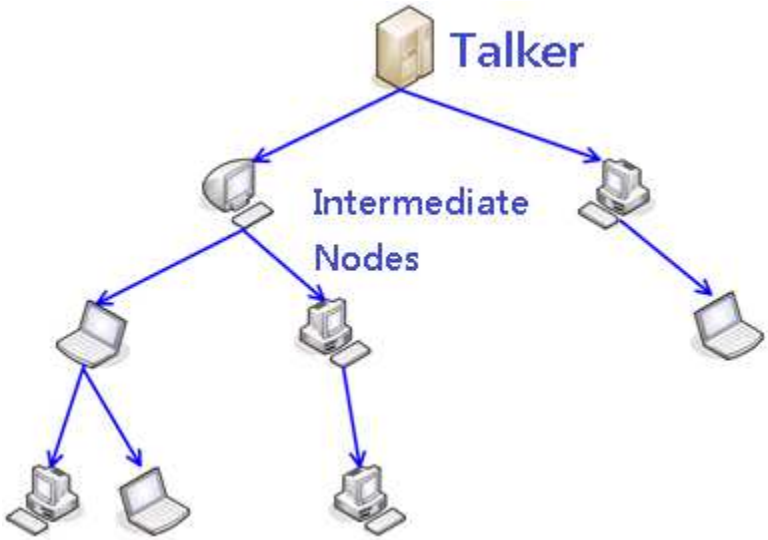

圖 12：Logical Mixing/Distribution Tree

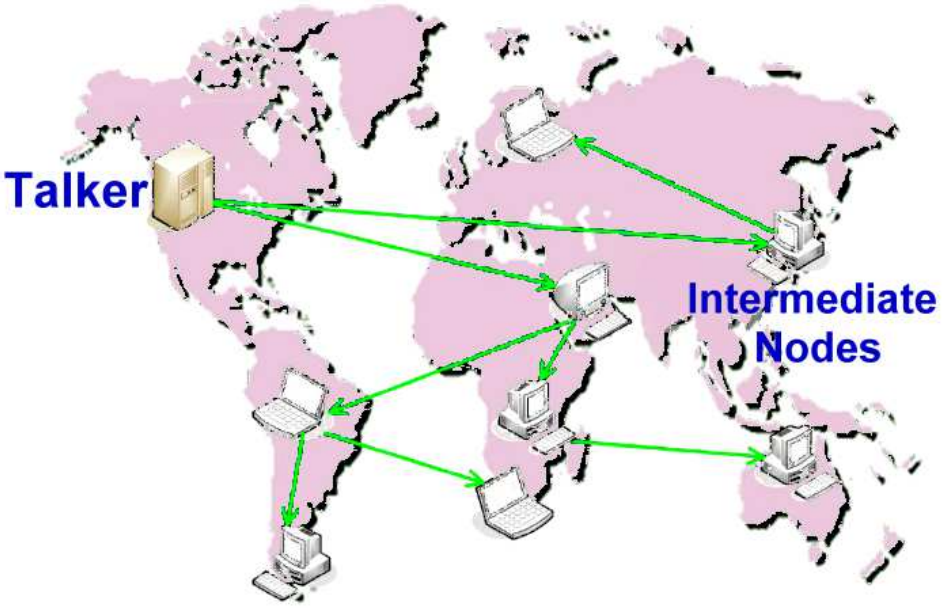

圖 13：Physical Mixing/Distribution Tree

為避免前述缺陷，本研究提出群組概念，將同一地區內彼此間有充足頻寬的節點歸納成一群組，而使用 Full Mesh Broadcasting 機制讓群組內之節點可以利用充足頻寬彼此互傳，以減低群組內傳送延遲時間，群組之間再利用一個拓樸複雜度較低的 multicasting tree 互連。

我們採用具有回音消除能力的最大能量追縱 VAD (Maximum Energy Tracking Voice Activity Detection) 來做靜音消除[6]，此方法會紀錄每位使用者說話的語音能量變動範圍，以此做為分辨語音訊框的臨界值。如此當語音會談中沒有人發話時，則不會有使用者傳出聲音封包佔用頻寬，此 VAD 僅需根據振幅計算能量值，對系統運算量負擔極少。

群組間的 multicasting tree 是以 Minimum Loss Diameter Spanning Tree (MLDST) [7]所建構，此樹狀結構能限制每個節點分支的數量（node degree）及最長路徑的累積延遲時間，每條傳輸路徑的封包遺失率在最佳解中也會是最小，透過此 multicasting 機制，期望在不影響通話品質下，盡可能提高與會人數。

本方法分為兩階段執行，在第一階段先求得使用者與登入伺服器間的封包往返時間（Round-Trip Time, RTT），利用 RTT 將使用者分為若干群組，稱為 cluster，再利用群組內使用者間的 RTT 做群組切割，並做群首篩選，圖 14 為分群成果。第二階段再利用群首間的 RTT 為依據，來建立一個複雜度較小的 MLDST。

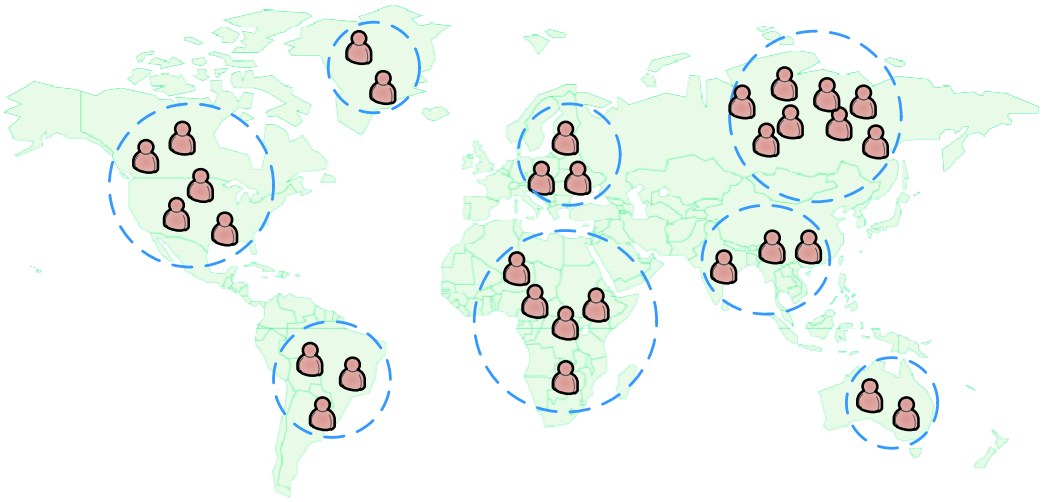

圖 14：Clustering By Physical Distance

本研究提出以群組方式建立一個 MLDST 的語音會談網路架構，稱為 Cluster Multicasting Tree（CMT），能進一步控制傳輸延遲時間及封包遺失率，以確保所有使用者都能夠在時間限制之內收到來自其他使用者的語音封包。圖 15 為 CMT 架構示意圖，圖中綠色節點為一群組，群組內採用 Full Mesh 機制連接。

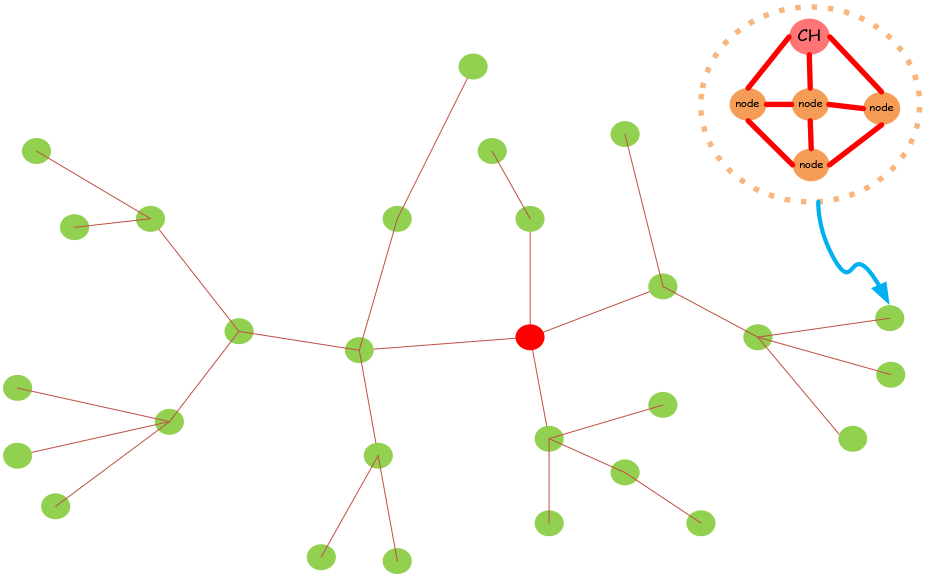

圖 15：Cluster Multicasting Tree

除此之外，本研究針對大型網路語音會談中，使用者分散世界各地的特性，結合雲端計算－Everything as a Service 概念，透過其龐大的頻寬將聲音封包傳送給世界各地的與會成員，來降低平均通話延遲時間，並提高與會人數上限。

3.2CMT 設計流程

本系統依照地理位置來分群，在每個群組內篩選出 Cluster Head，採用 MLDST (Minimum Loss Diameter Spanning Tree) 為骨幹將 Cluster Head 連接，並配合靜音消除減少網路負擔，以此建立一網路拓樸，控制封包傳送延遲時間及封包遺失率。以下為 CMT 設計流程。

```
size bound is the allowed largest size of a cluster
For each participant P
      Join a cluster in accordance with the RTT time between P and login
      server
End
For each cluster C
      Divide cluster C into smaller ones according to RTT time among users
      Employ full mesh broadcasting
      If cluster size > size bound
         Then divide cluster C into smaller one
End
Execute cluster header election procedure
Build MLDST Tree.
```

3.2.1. 分群機制

我們採用分群機制之目標在於找出位於同一地區彼此間有寬頻網路連接之節點，群組內使用 Full Mesh Network 連接，群組間再以一個複雜度較小之 multicasting tree 連接。首

先利用登入伺服器做為標的節點（landmark），當有節點加入此網路時，便測量節點到此登入伺服器的 RTT 做為 network distance，將 network distance 相近之節點歸為同一群組（Cluster），如圖 16。

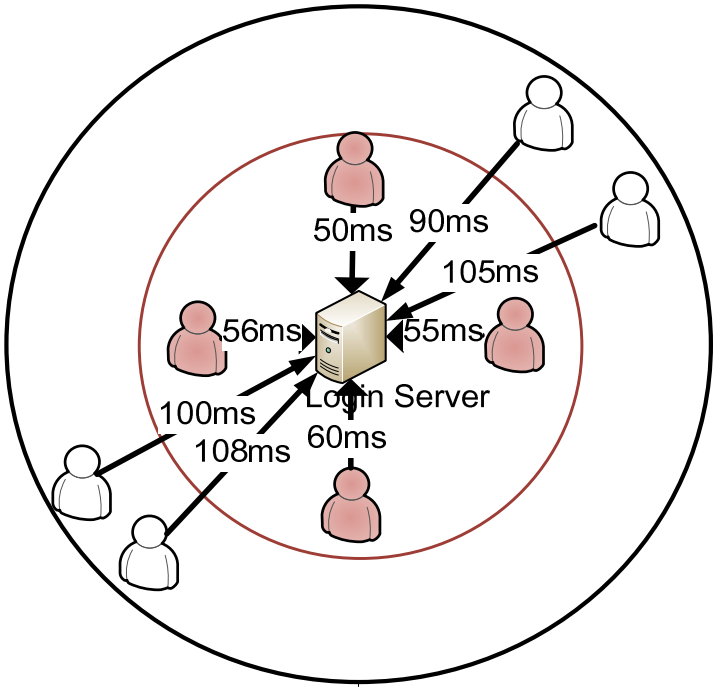

圖 16：Initial Clustering

但初次分群時並無考慮使用者彼此間的距離，可能造成同一群組中的使用者彼此相距過遠，並非在同一地區之內，如圖 16 中白色使用者，因此群組中每位成員必需測量彼此間 RTT，藉此判斷相互間距離之遠近，將鄰近節點再次切割成較小的群組。群組內採用 Full Mesh 機制連接，當群組內任一成員發話時，就自行廣播給其它群組成員，以減少群組內傳送延遲時間，若某節點發生問題也不至於影響會談進行。群組內部每個節點最多連接數（node degree）為 K，若超過則再做一次群組的切割。

3.2.2. 群首篩選

分群結束後，群組內部將進行群首篩選（header election procedure），以選擇出一個最佳的點做為群首（Cluster Head）。首先，login server 將於每個群組中隨機挑選節點，由這些節點來測量群組間的 RTT 時間，如圖 17。圖 17 中，深藍色使用者測量到自己與其他群組使用者的 RTT 後，將依照 RTT 來選擇所要連接的使用者，圖 17 中藍色使用者選擇 RTT 為 250ms 的使用者做為連接對象。

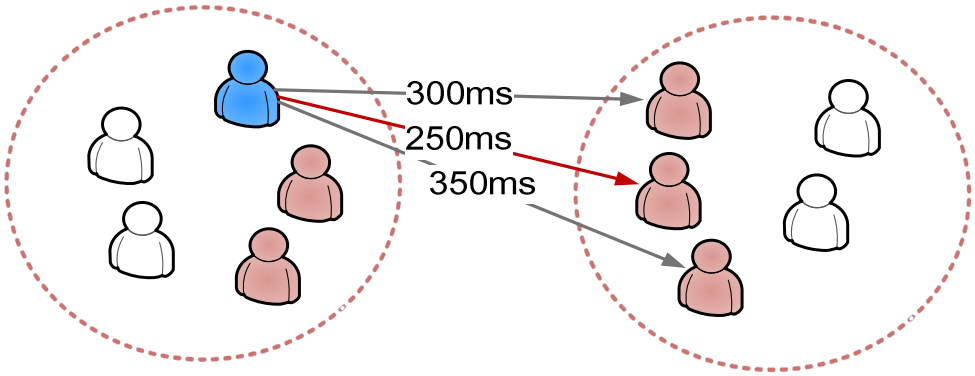

圖 17：群首篩選第一步

群組中隨機挑選的使用者在經過此機制後，皆會篩選出若干連線路徑，如圖 18。圖 18 中篩選出三條連線路徑，login server 將選擇 RTT 最小之路徑將其兩節點做為群首。在圖 18 中系統將選擇 200ms 路徑的兩藍色節點為群首。由此機制選擇出群首之後，群首將會發送一通知訊息給群組內其它節點告知自己具有群首的身份。

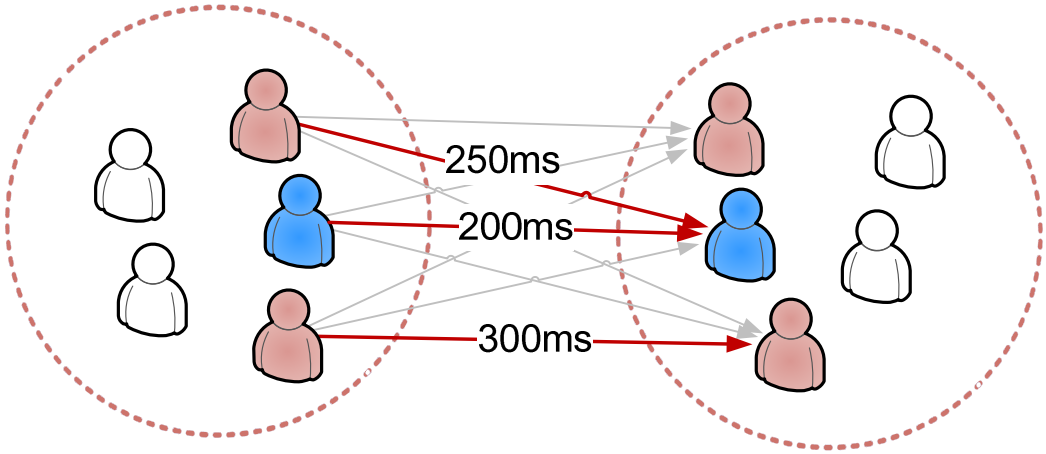

圖 18：群首篩選第二步

3.2.3. 建立 Multicasting Tree

spanning tree 中最長的路徑稱之為 diameter，若權重為 delay time，則稱為 delay diameter，若權重為 loss rate 則稱為 loss diameter。而 MLDST 是一個 delay diameter 及各節點的 degree 數有限制，而 loss diameter 為最小之 spanning tree。

接著以 Minimum Loss Diameter Spanning Tree 建立一個 multicasting tree 將群首連接起來以廣播聲音封包到所有群首，為了要建立 MLDST 此拓樸架構，我們以 Single Source Shortest Path 演算法為基礎，加入延遲時間（delay time）及節點分支數量（node degree）的限制，建構 Constrained Dijkstra 演算法，藉此求得一個 MLDST 的次佳解。

Constrained Dijkstra 演算法是一種改良過的 Bellman-Ford 演算法，首先設定 delay diameter 上限，讓此 spanning tree 中任兩點的 delay time 小於此上限，並限制每個節點分支數量，在此兩點限制之下去尋找一個 loss diameter 最小的 spanning tree [7]。透過此 MLDST 將 Cluster Head 連接起來，每個 Cluster Head 最大連接數小於一定值 H，並選定合適的 delay bound 來篩選任兩點間的 delay diameter（$d_{i,j}$）。圖 19 為一 MLDST 範例，此 spanning tree 為唯一解。

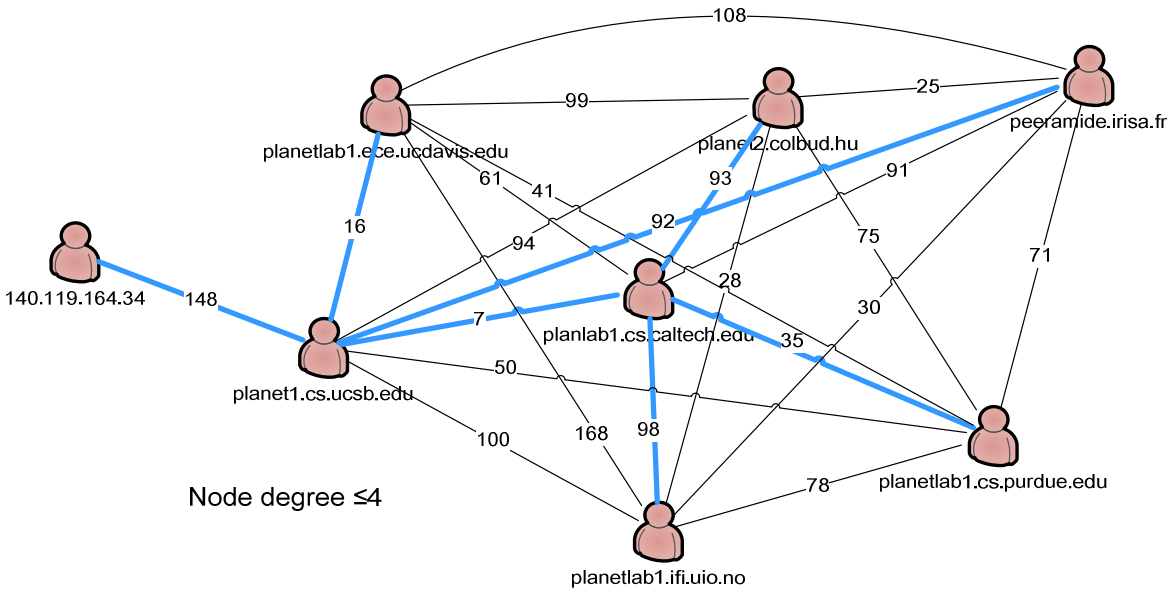
108
99
25
planetlab1.ece.ucdavis.edu
planet2.colbud.hu
peeramide.irisa.fr
61
41
93
91
16
92
94
75
71
140.119.164.34
148
28
30
7
planlab1.cs.caltech.edu
35
50
planet1.cs.ucsb.edu
100
168
98
78
planetlab1.cs.purdue.edu
Node degree ≤4
planetlab1.ifi.uio.no

圖 19：MLDST 範例

以下為 Constrained Dijkstra's Algorithm。

$d[v]$ is the length of the shortest path from root s to node v

$w(u,v)$ is the weight of the link between node u and v

delay bound is the allowed maximum service delay

degree bound is the allowed maximum number of outgoing links of each intermediate node in the MLDST

$degree[v]$ is the current number of outgoing links of node v in the MLDST

Initialize –Single –Source(G,s)

for $i \leftarrow 1$ to $|V(G)| - 1$

do for each edge $(u,v) \in E[G]$

do $RELAX(u,v,w)$

for each edge $(u,v) \in E[G]$

do if $d[v] > d[u] + w(u,v)$

then return FALSE

do if $d[v] > delay\ bound$

then return FALSE

do if $degree[v] > degree\ bound$

因為 Single Source Shortest Path 演算法求得的最短路徑必定比 Minimum Spanning Tree 來的小，因此 MLDST 選擇 Single Source Shortest Path 演算法來做改進，以下利用矛盾證法來證明此理論。

Given a graph $G = (V, E)$. T_x is a spanning tree of G, rooted at s, and $P(s,u)_x$ denotes the path from s to u of T_x and $|P(s,u)_x|$ is the length of $P(s,u)_x$. The longest path of T_x is defined as $ecc(s)_x$.

Theorem: If T_j is a spanning tree generated by Dijkstra algorithm and its source node is s. Then $ecc(s)_j \leq ecc(s)_x$, for all T_x belong to G.

3.3雲端計算

雲端計算是一富有擴展性（scalable）之架構，其提供之虛擬化資源是以服務形式（as a service）提供給使用者，使用者不需要對雲端內部架構、控制流程有明確的了解即可使用，本研究利用其核心概念－everything as a service 來協助大型網路語音會談的進行。在雲端計算中需要有足夠的頻寬讓分散世界各地的使用者來利用雲端上的資源，而大型網路語音會談中語音成員常分佈於世界各地，因此我們利用雲端內充裕的頻寬將聲音封包送至通話延遲時間過高的使用者。圖 20 為雲端概念圖，現今提供雲端服務的公司有 Google, Amazon 等多家公司，本研究採用 Google 提供之雲端。

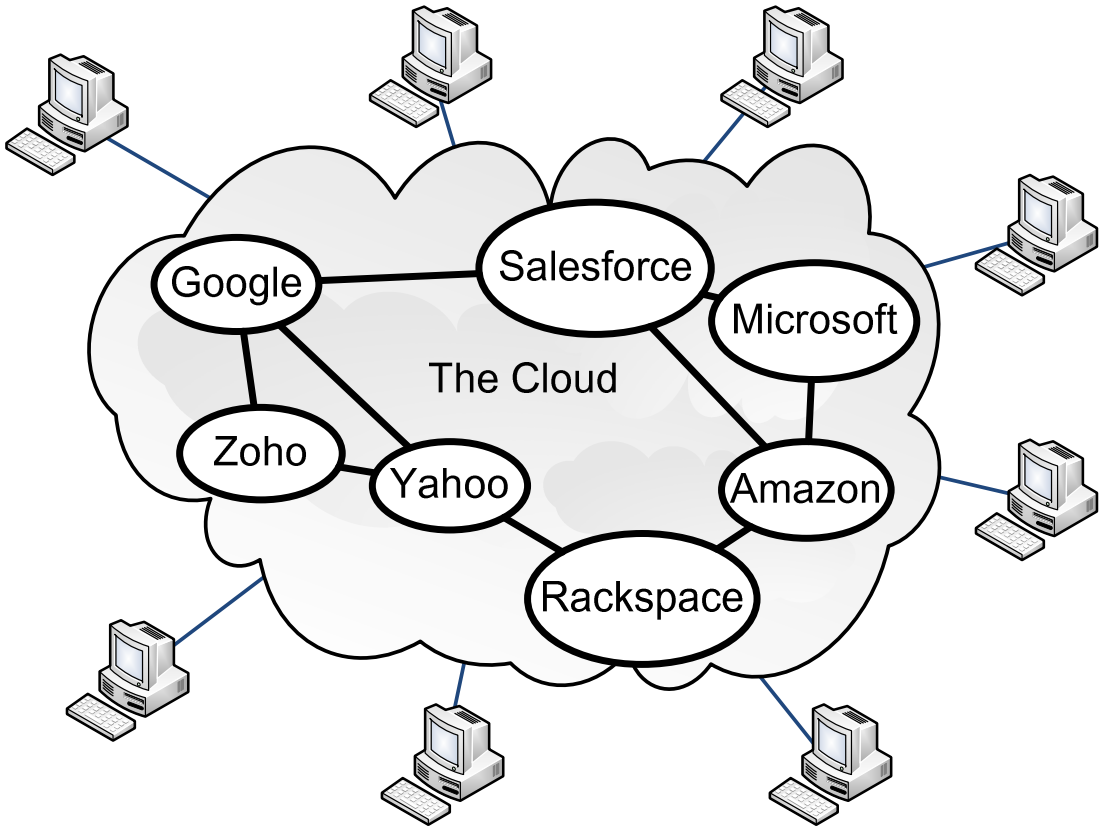

圖 20：雲端計算概念

3.3.1 系統架構

我們利用雲端計算有著充裕網路頻寬的特性來協助轉送使用者的聲音封包，在大型網路語音會談中使用者分佈世界各地，會談進行時若有使用者與發話者間的網路距離過遠，造成其通話延遲時間過長，login server 便會要求使用者從雲端抓取使用者聲音封包，而不經過系統建置的 tree，如圖 21 範例。

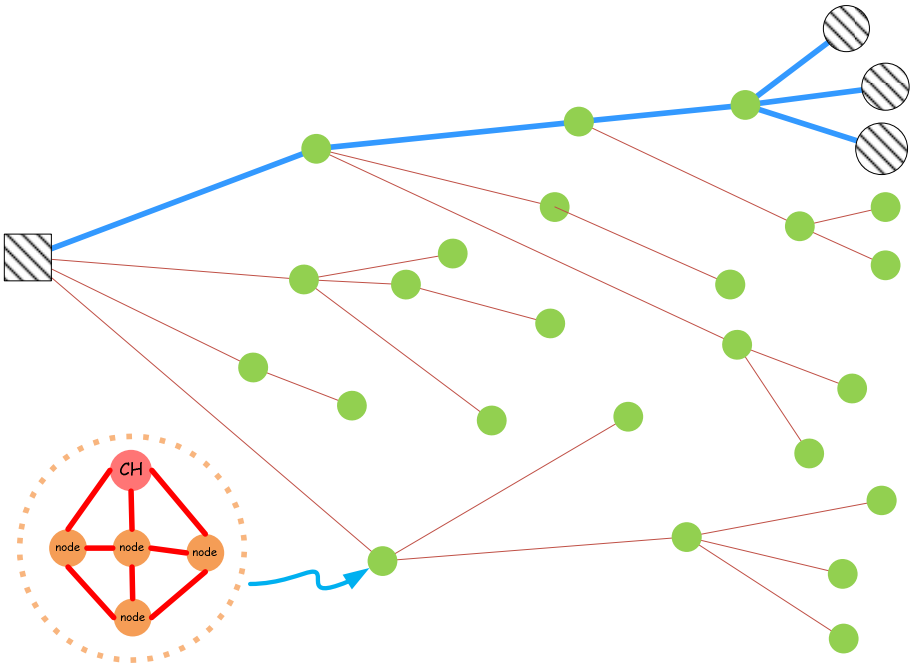

圖 21：延遲時間過長之範例

圖 21 中，若方形發話者將聲音傳送至圓形斜紋使用者所需時間超過 350ms，將影響使用者的收聽品質，因此發話者將聲音傳送至 Google 雲端上，藉由 Google 雲端龐大的網路頻寬，將聲音封包快速傳向世界各地伺服器，讓圖中圓形斜紋使用者轉而向 Google 雲端收取聲音封包，藉此大幅地降低通話延遲時間，如圖 22。

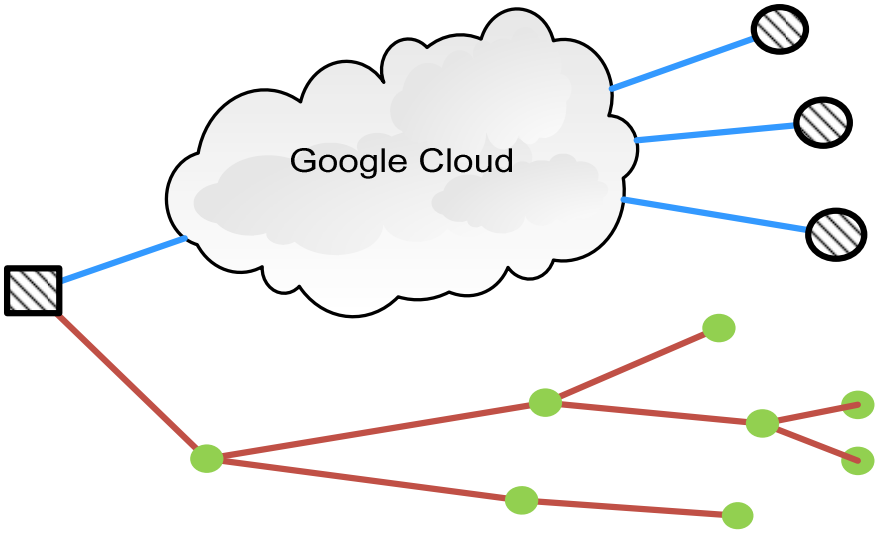

圖 22：藉助雲端計算資源之協助架構

第四章　效能評估

本研究以實驗來評估所提出之傳輸機制，藉由在接收端觀察封包延遲時間及與會人數多寡來了解其效能。

4.1實驗評估指標

- service delay [10]：發話者發話至收聽者聽到的時間，愈小愈好。
- 與會人數：在不影響通話品質下參與會談的人數，愈多愈好。

4.2實驗環境

實驗是在 PlanetLab 平台進行，藉由此平台我們可以操作世界各地的電腦，來實際測試大型網路語音會談並與其它傳輸機制比較。在此實驗環境中，說話者人數為一人，與會人數從 5 至 20 人不等，並依照 peerTalk、P2P Multicast、Centralized Server 三種不同方式來改變本程式的傳送拓撲，以比較其效能。

但由於在 PlanetLab 上的電腦性能普遍不佳，且同一電腦在不同時間的同時使用人數並不固定，難以掌握電腦所能使用的頻寬、系統資源，因此我們預期本技術應用於實際網路時，可以有更佳的成效。

4.3實驗一

4.3.1 實驗目標

由於 PlanetLab 上測試平台上電腦性能參差不齊，不同時間所能使用的網路頻寬、系統資源不盡相同，且使用者彼此間的實際距離長短不一，導致發話者至每位使用者的路徑長短可能差異極大，因此本實驗在各種網路會談語音架構中，隨機挑選兩位使用者，量測兩者間的 service delay (one-stream service delay) ，以突顯不同架構間的優缺點。此外，本實驗也計算發話者至所有使用者的平均 service delay (aggregated average service delay) ，並統計其 packet loss rate(aggregated average packet loss rate) ，來觀察在大型網路語音會談中，peerTalk、P2P Multicast、Centralized Server 及本研究所提出傳輸機制之效能表現。

4.3.2 實驗結果與分析

表 6 及圖 23 至圖 25 統計不同語音會談網路架構中，隨機挑選出兩位使用者在不同與會人數下的 service delay。

表 6：實驗結果 One-Stream Service Delay

	Number of Participants	CMT	peerTalk	Multicast	Centralized
Max Service Delay(ms)	5	186	308	255	372
	10	264	486	450	475
	15	331	470	490	580
	20	415	538	643	712
Min Service Delay(ms)	5	157	260	218	311
	10	218	348	342	401
	15	265	390	419	472
	20	344	492	532	614
Avg Service Delay(	5	159	265	220	317
	10	224	363	348	416
	15	272	396	424	494

	20	349	495	536	620

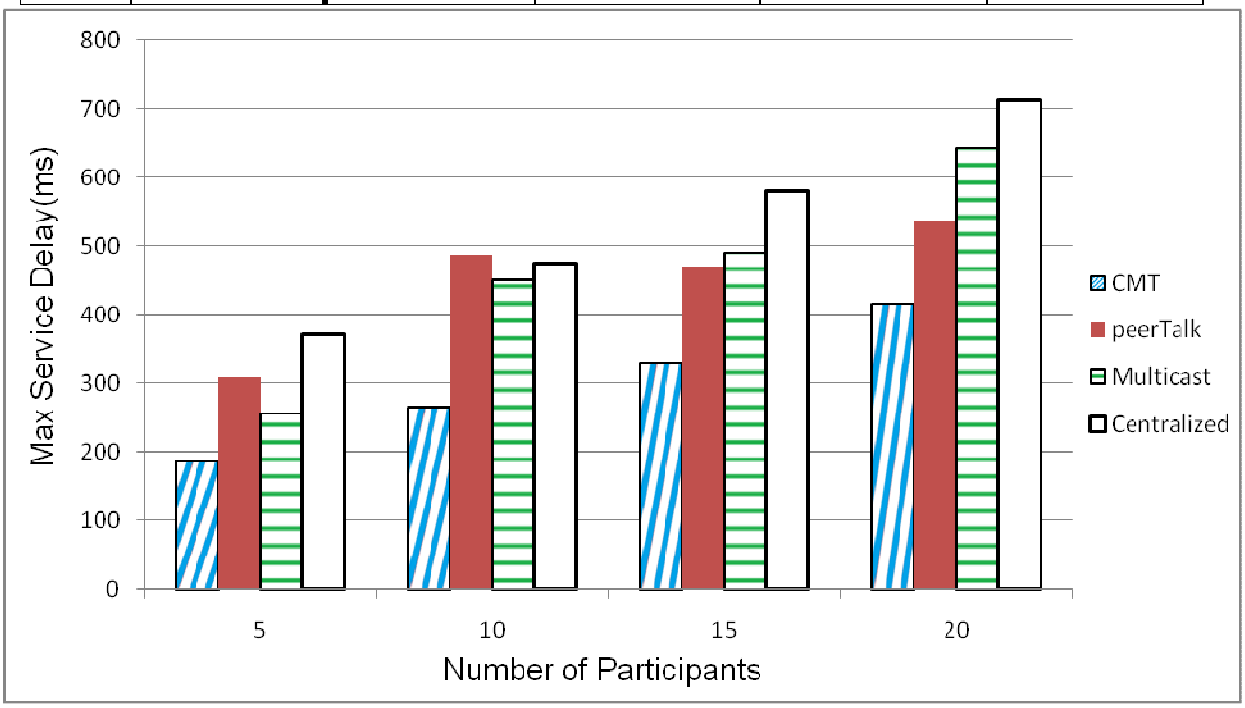

圖 23：One-Stream Max Service Delay

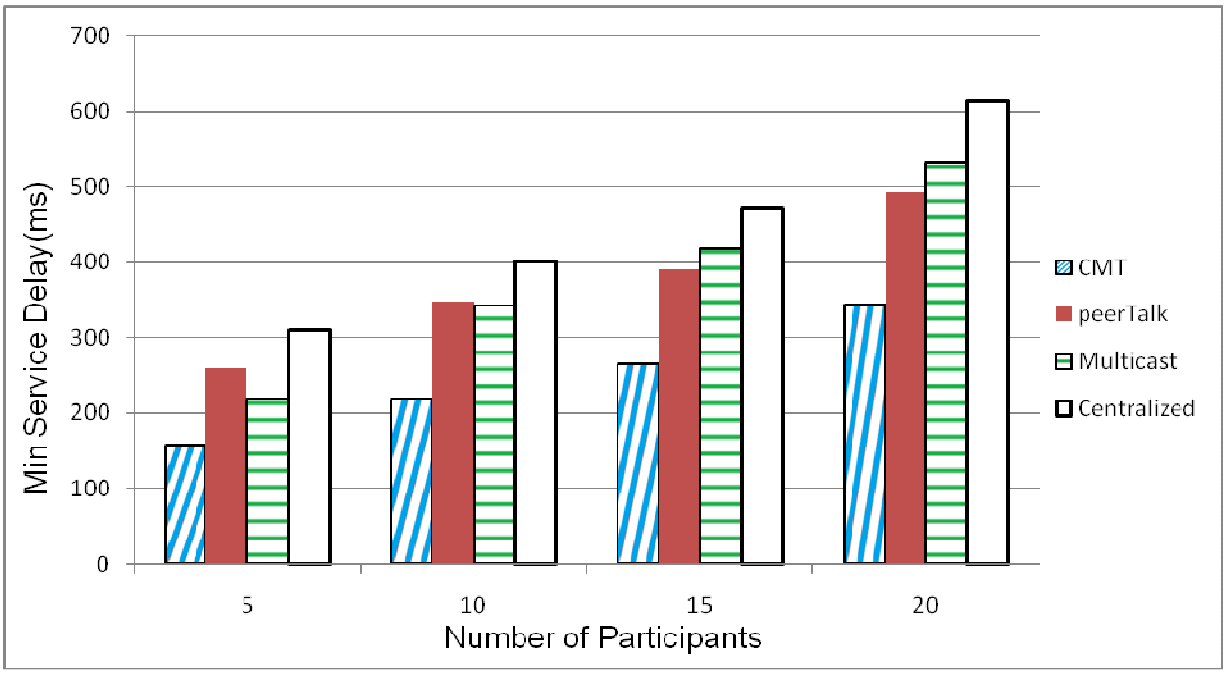

圖 24：One-Stream Min Service Delay

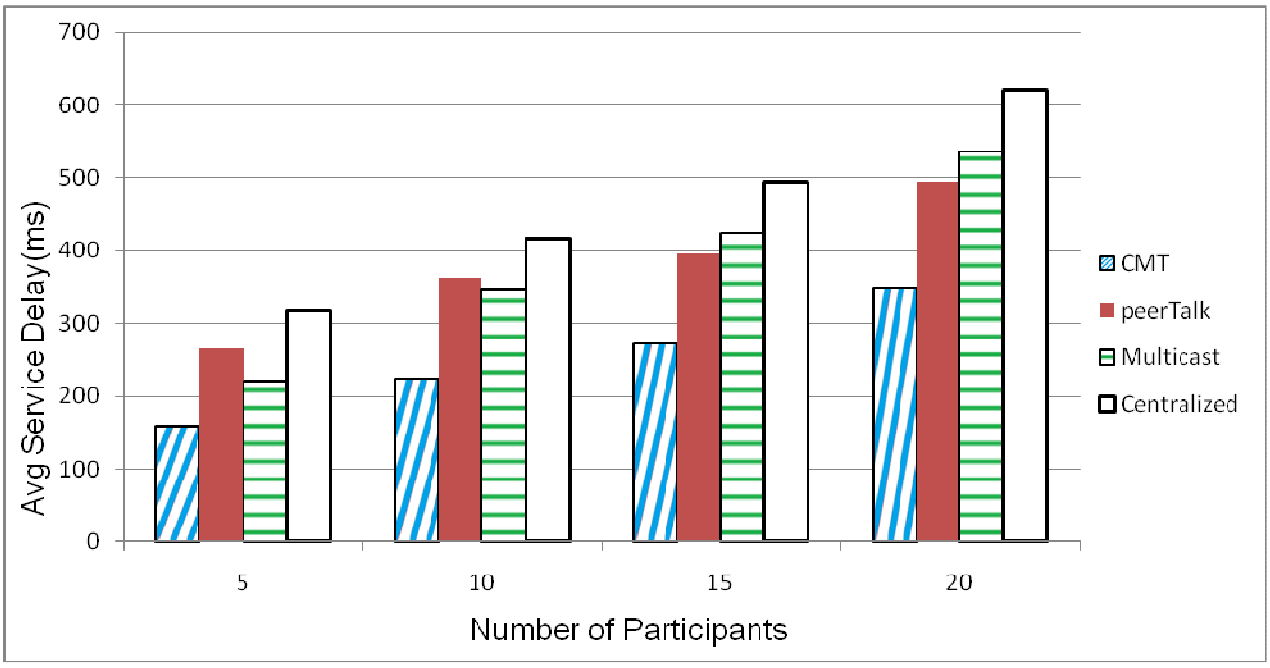

圖 25：One-Stream Avg Service Delay

圖 26 至圖 29 分別為與會人數 5、10、15、20 人下 one-stream service delay。

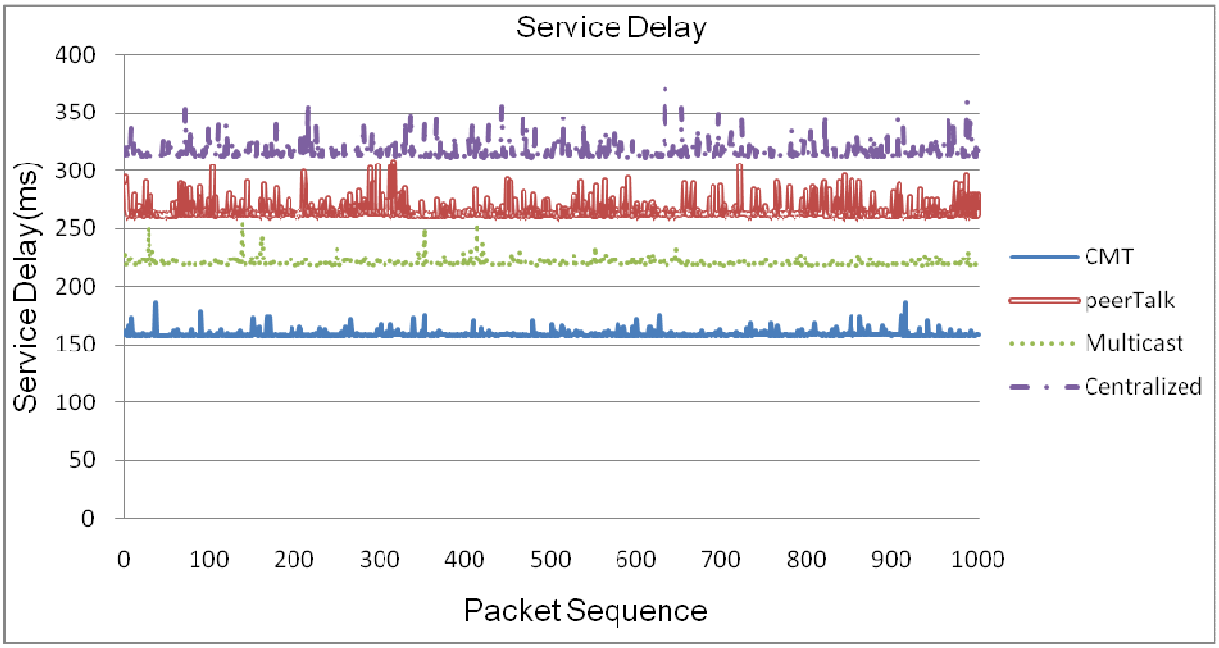

圖 26：One-Stream Service Delay（5 人）

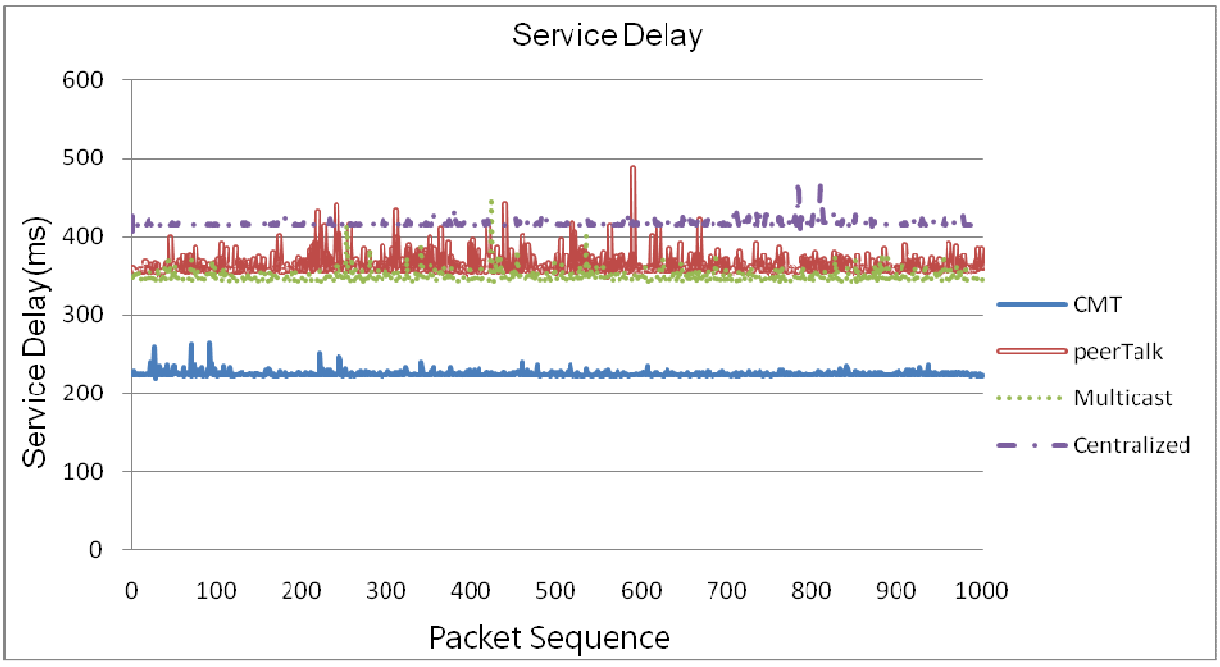

圖 27：One-Stream Service Delay（10 人）

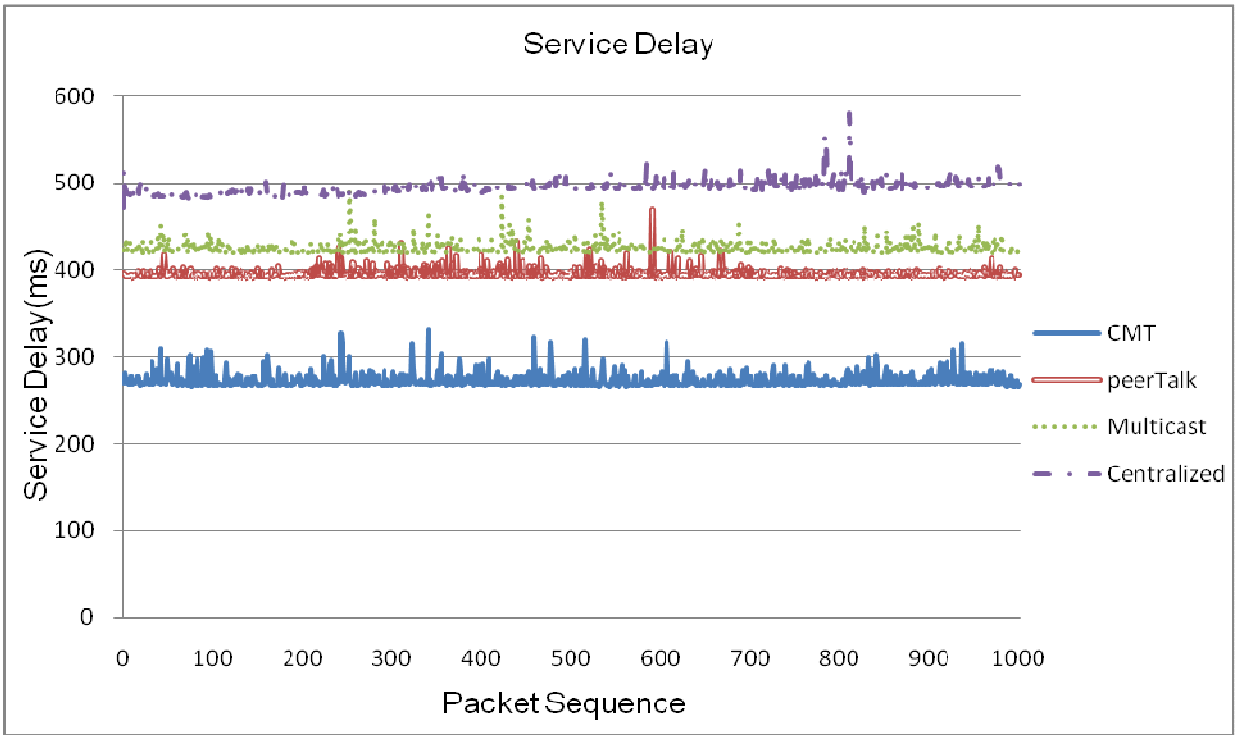

圖 28：One-Stream Service Delay（15 人）

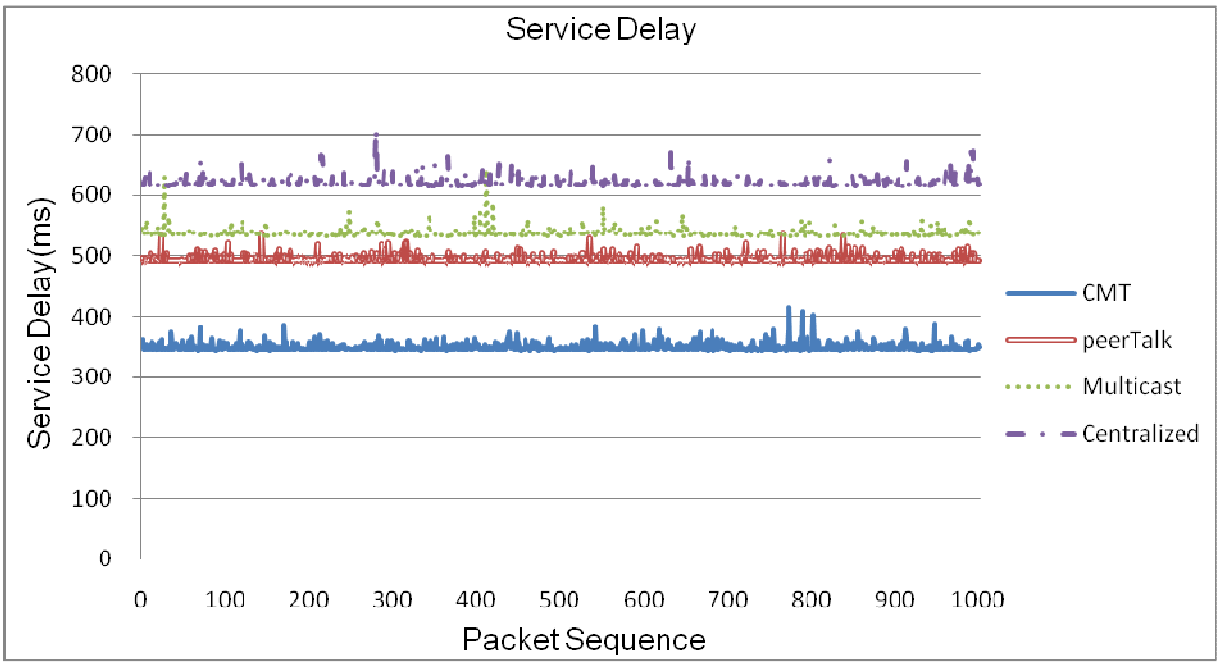

圖 29：One-Stream Service Delay（20 人）

表 7 及圖 30 至 32 統計不同語音會談網路架構，在不同與會人數下的發話者至其他與會成員的平均 service delay (aggregated average service delay) 及 packet loss rate(aggregated average packet loss rate) 。

表 7：實驗結果 Aggregated Average Service Delay

	Number of Participants	CMT	peerTalk	Multicast	Centralized
Max Service Delay(ms)	5	189	309	257	407
	10	311	400	374	466
	15	388	493	593	543
	20	464	529	589	660
Min Service Delay(ms)	5	158	260	197	311
	10	199	293	313	341
	15	239	356	389	467
	20	318	443	477	554
Avg Service Delay(ms)	5	162	271	205	321
	10	217	305	323	405
	15	266	386	418	486
	20	347	464	516	602

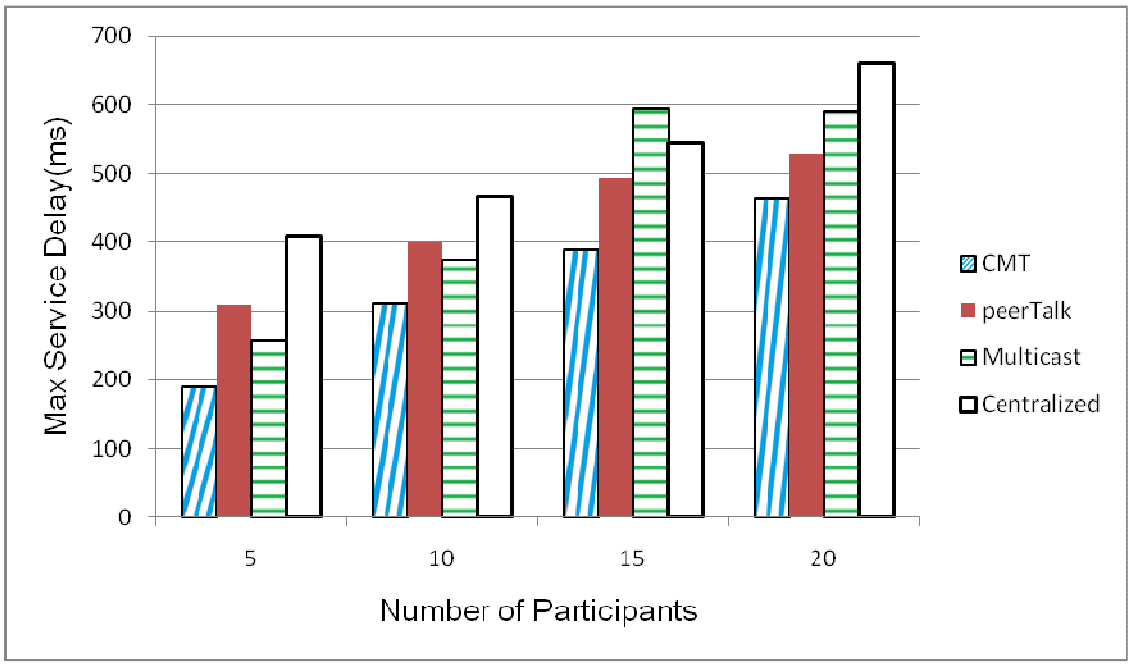

圖 30：Aggregated Max Service Delay

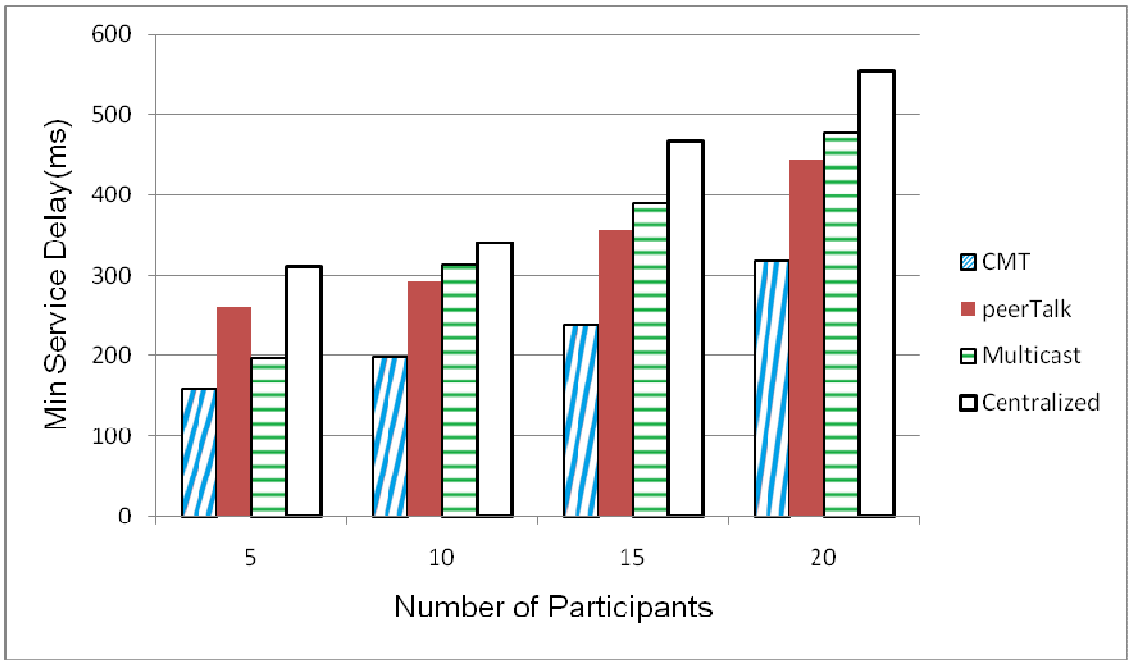

圖 31：Aggregated Min Service Delay

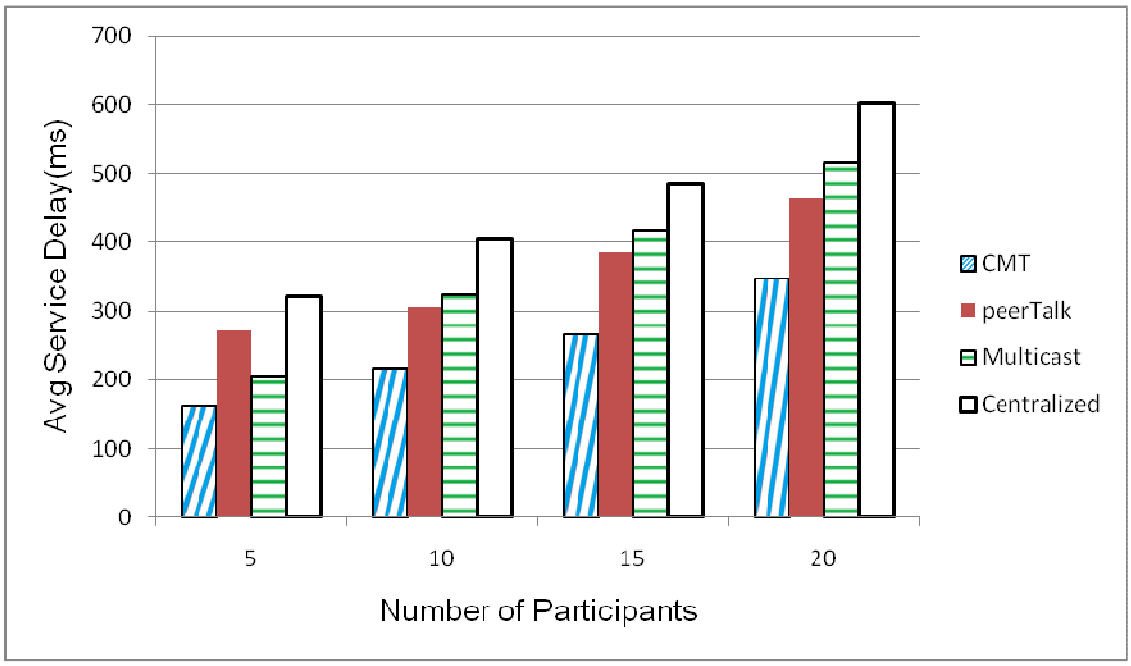

圖 32：Aggregated Avg Service Delay

圖 33 統計不同語音會談網路架構，在不同與會人數下的平均 loss rate。

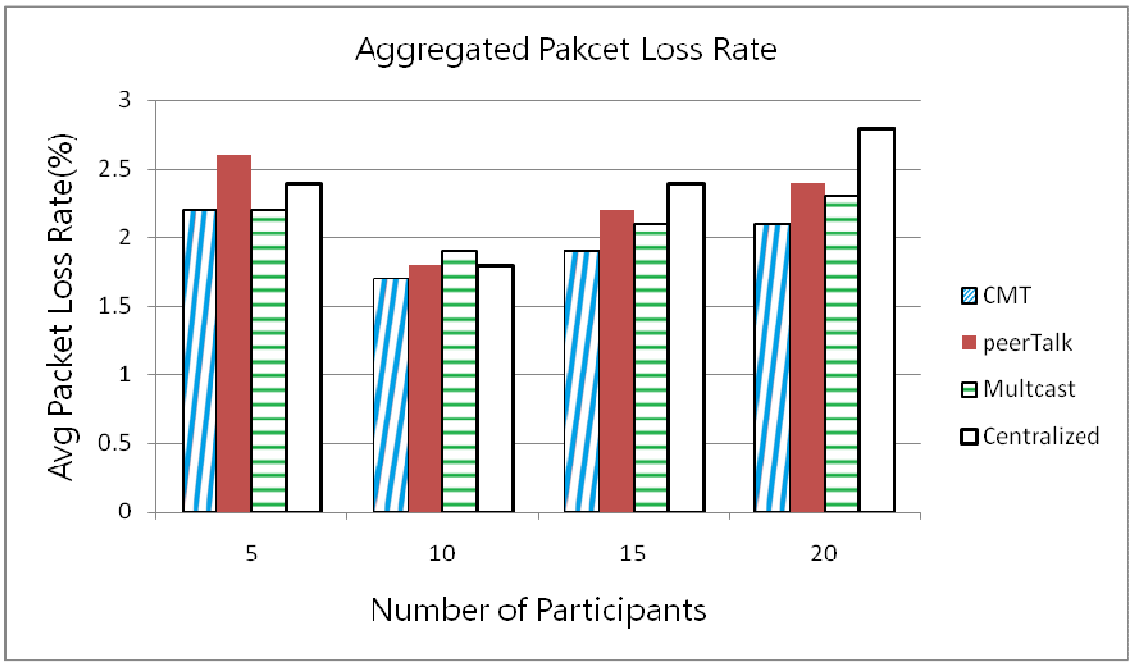

圖 33：Aggregated Average Packet Loss Rate

圖 34 至圖 37 分別為與會人數 5、10、15、20 人下的 average service delay。

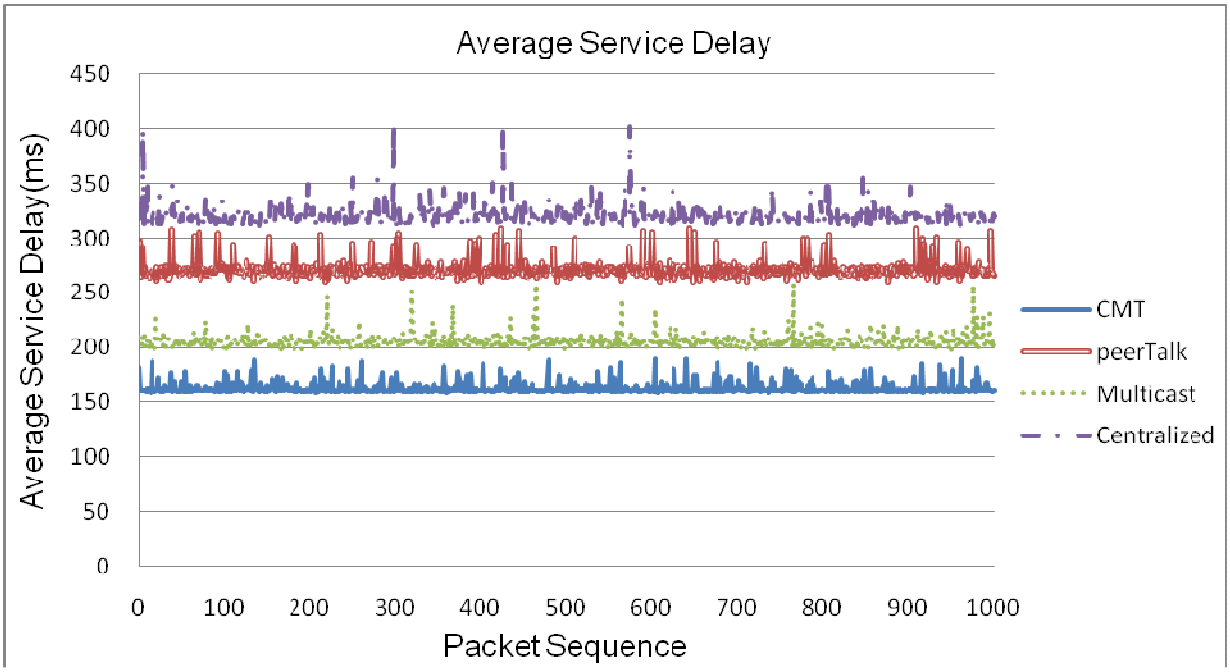

圖 34：Aggregated Average Service Delay（5 人）

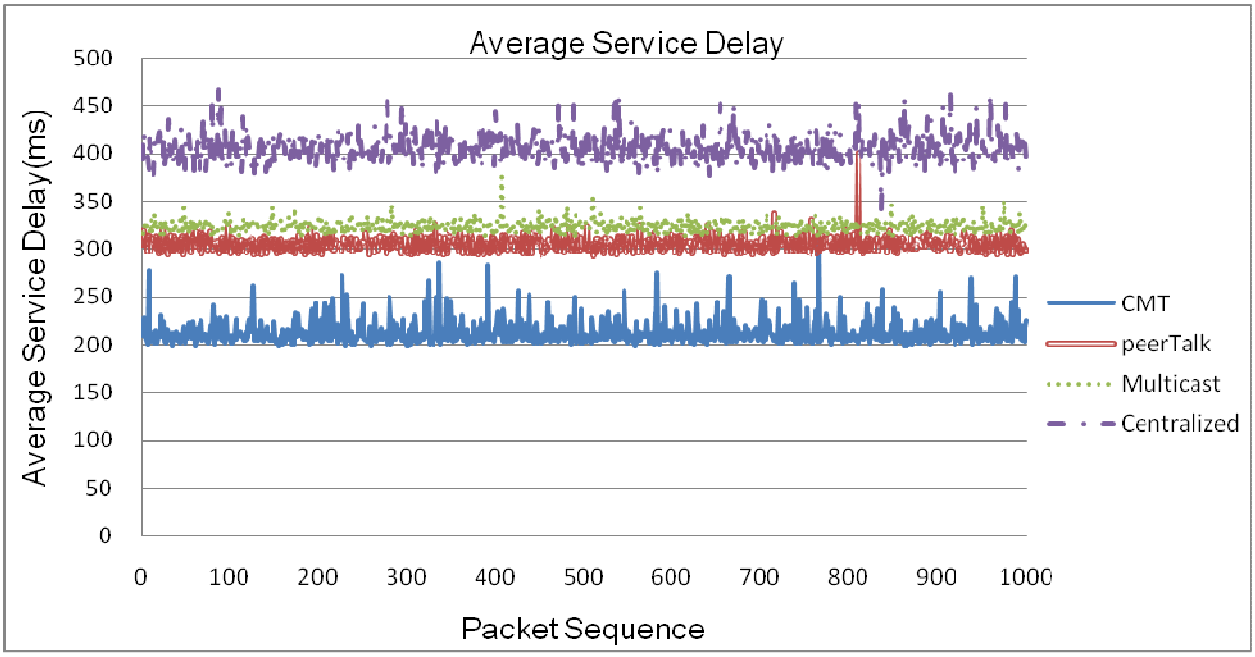

圖 35：Aggregated Average Service Delay（10 人）

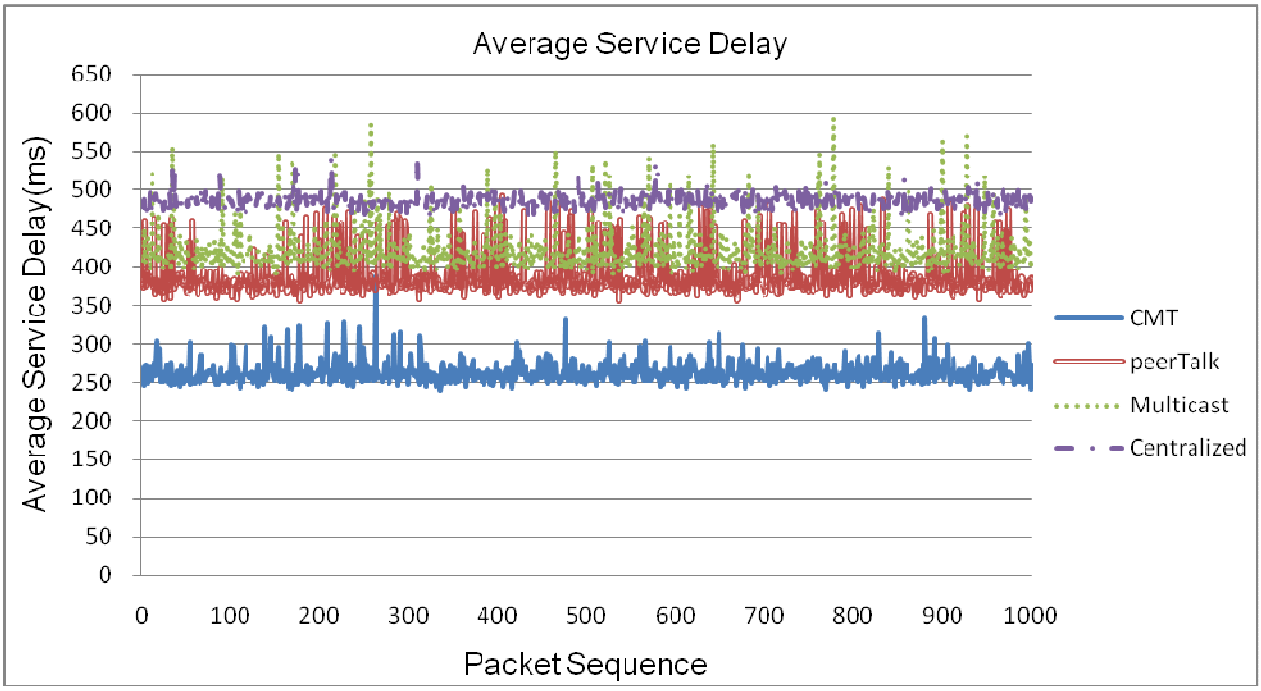

圖 36：Aggregated Average Service Delay（15 人）

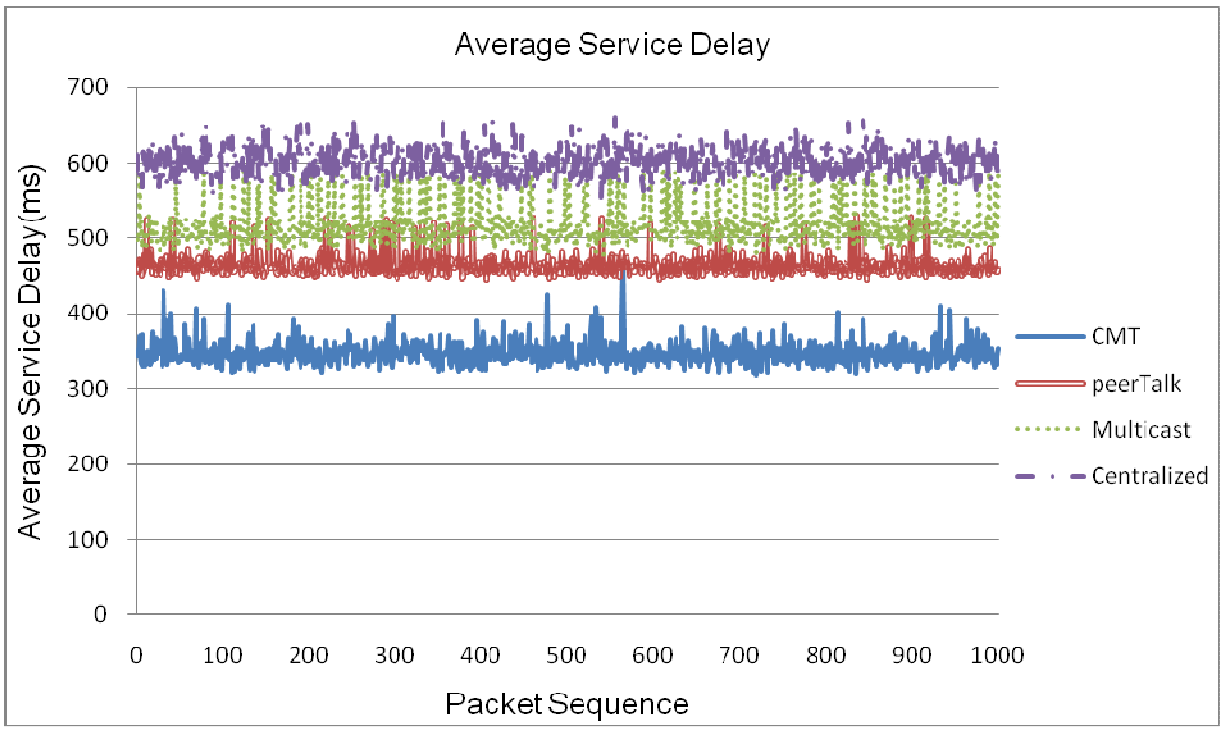

圖 37：Aggregated Average Service Delay（20 人）

由實驗結果可以看出，本研究依照使用者實際位置建置的 Cluster Multicasting Tree，配合靜音消除機制，能有效節省頻寬需求，加上刪除混音程序的結果，可大幅降低 service delay，在與會人數高達 20 人時，仍可將封包 service delay 控制在 350ms 上下。Centralized server 由於其上下傳頻寬、效能負擔過大，在與會人數超過十人後 service delay 漸漸超過 350ms；peerTalk 雖然利用 mixing、distribution 兩個不同樹狀架構來分散系統負擔，但 mixing tree 仍為其增加混音延遲時間的代價，在與會人數為十五人時，service delay 已達到 386ms；P2P Multicast 在與會人數超過十人後，因 multicasting tree 過長而增加了傳送延遲時間，service delay 漸漸超過 350ms。表 8 統計 service delay 小於 350ms 時，各個語音會談架構的最高與會人數。

表 8：最高與會人數（Service Delay < 350ms）

	CMT	peerTalk	Multicast	Centralized

Number of Participants	20	10	10	5

在封包遺失率方面，雖然 CMT 比其他方法有著較少的封包遺失率，但因為會議中所選取的節點都是比較穩定性能相對較佳的節點，以致改善情形不顯著，大部份情況下可控制在 2.5%以下。

4.4實驗二：雲端計算

4.4.1 實驗目標

實驗中利用 Google Code 及 Google Site 轉送聲音資料，讓 service delay 過大的使用者轉向 Google 雲端收取聲音資料，藉以展示雲端協助之擴充性，以及對效能的顯著提升。

4.4.2 實驗結果與分析

表 9 及圖 38 至圖 40 統計不同語音會談網路架構，在不同與會人數下的發話者至其他與會成員的平均 service delay (cloud-assisted VoIP conferencing aggregated service delay)。

表 9：實驗結果 Cloud-Assisted VoIP Conferencing Aggregated Service Delay

	Number of Participants	CMT	peerTalk	Multicast	Centralized
Max Service Delay(ms)	5	177	326	243	370
	10	199	367	402	448
	15	242	487	499	516
	20	307	517	594	675
Min Service Delay(ms)	5	136	271	186	322
	10	149	290	322	380
	15	183	367	408	451
	20	272	443	507	566
Avg Service Delay(ms)	5	139	275	200	327
	10	155	300	330	407
	15	194	375	424	483
	20	278	456	520	612

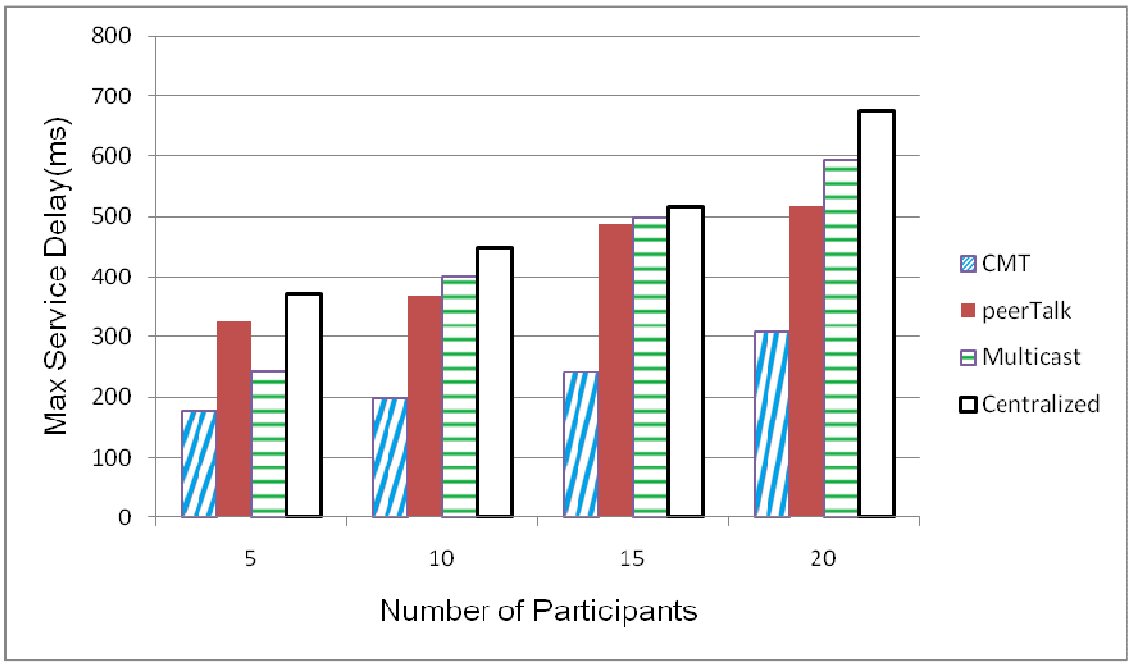

圖 38：Cloud-Assisted VoIP Conferencing Aggregated Max Service Delay

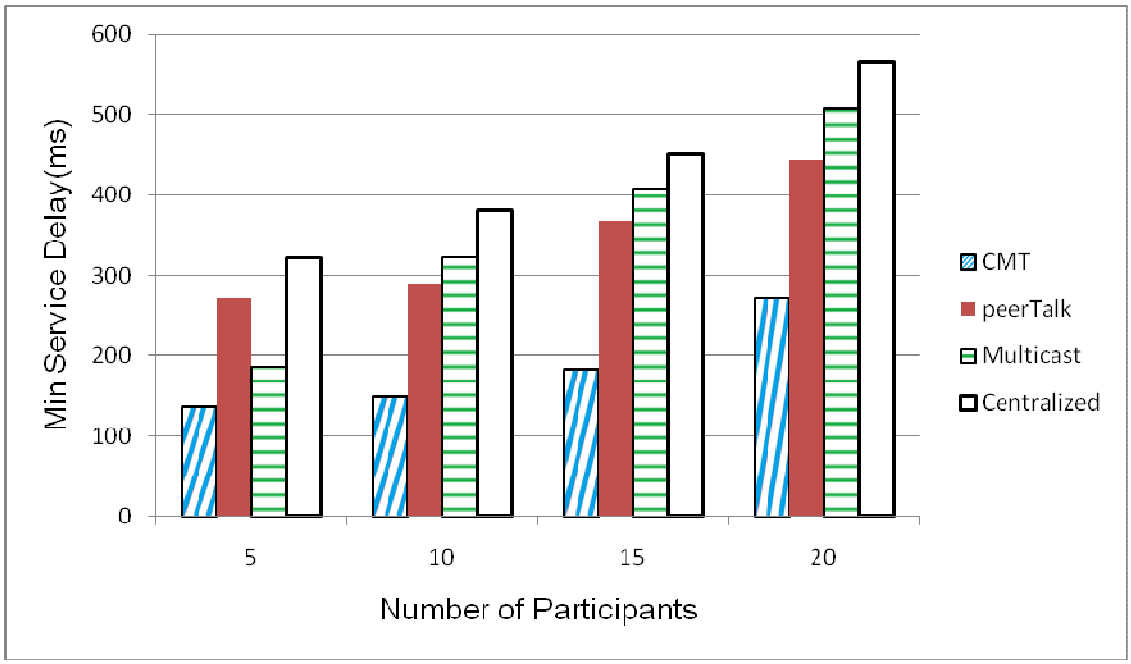

圖 39：Cloud-Assisted VoIP Conferencing Aggregated Min Service Delay

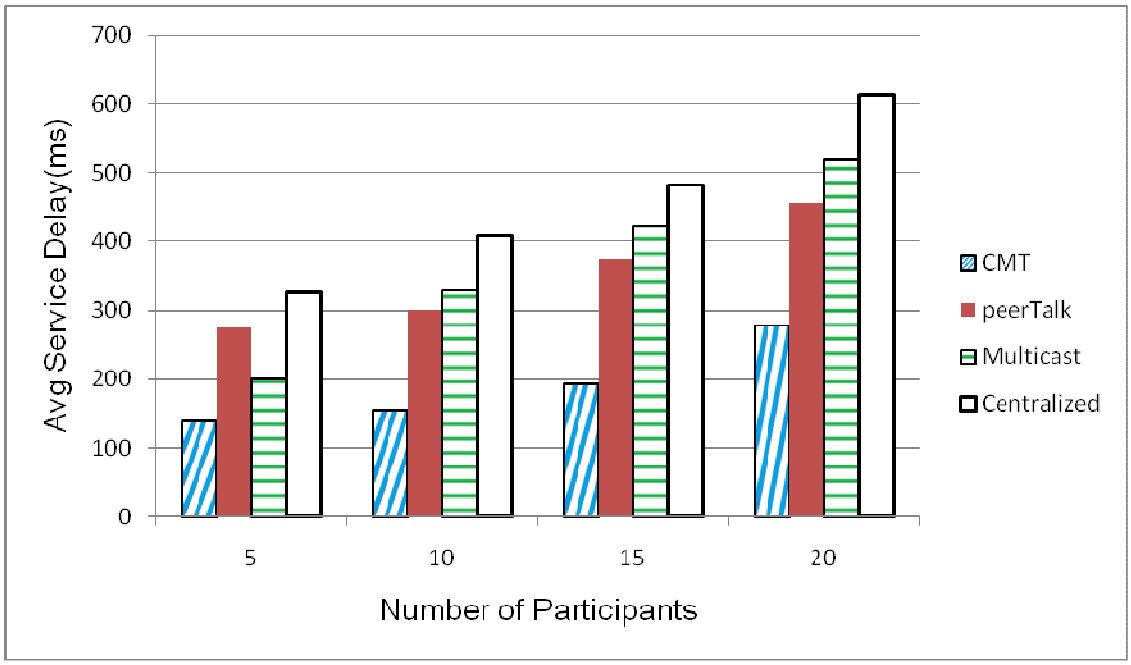

圖 40：Cloud-Assisted VoIP Conferencing Aggregated Avg Service Delay

圖 41 至圖 44 分別為與會人數 5、10、15、20 人下的 average service delay。

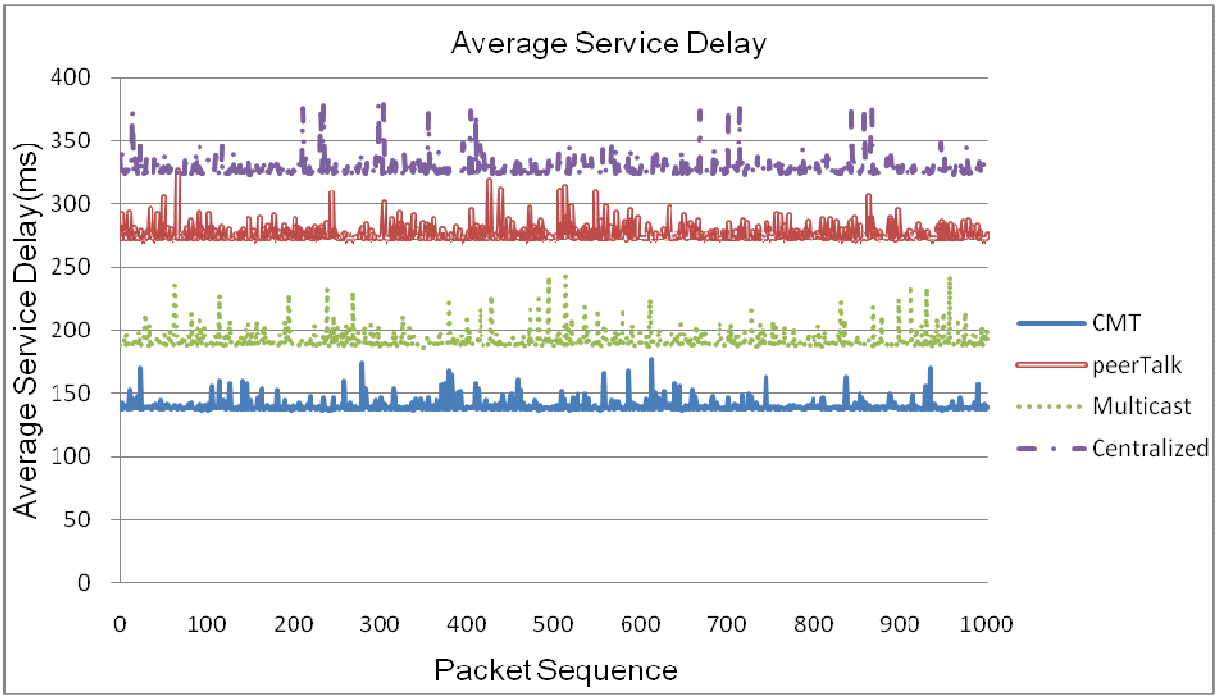

圖 41：Cloud-Assisted VoIP Conferencing Aggregated Service Delay（5 人）

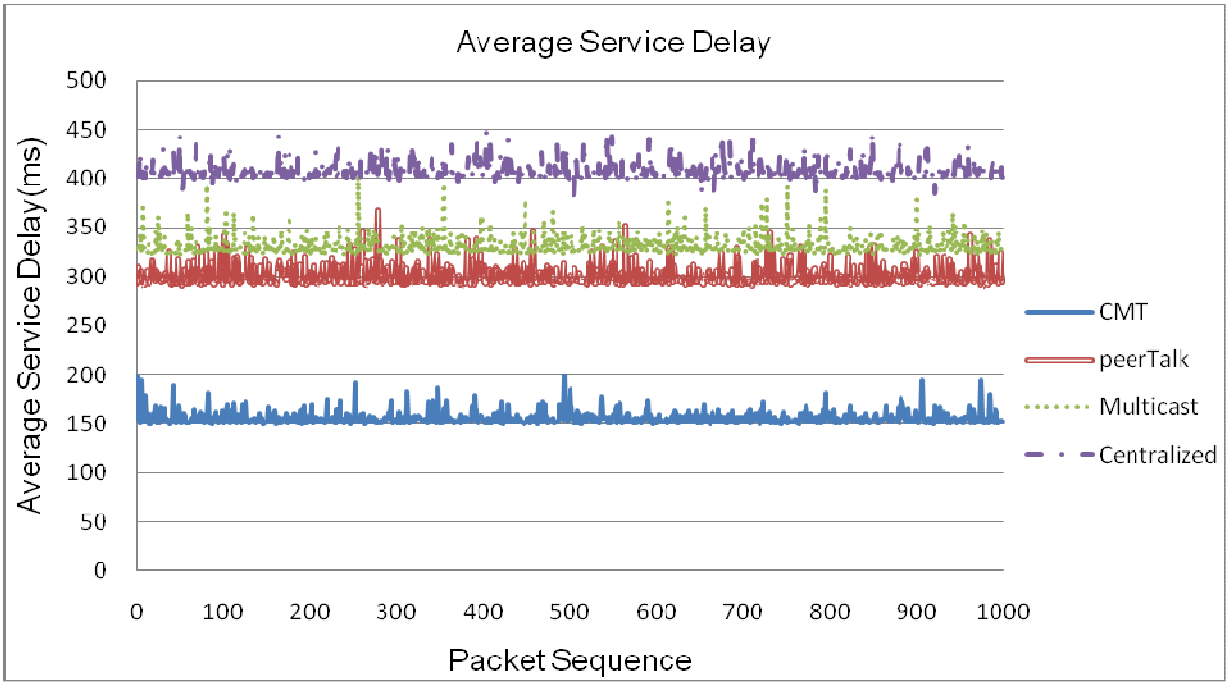

圖 42：Cloud-Assisted VoIP Conferencing Aggregated Service Delay（10 人）

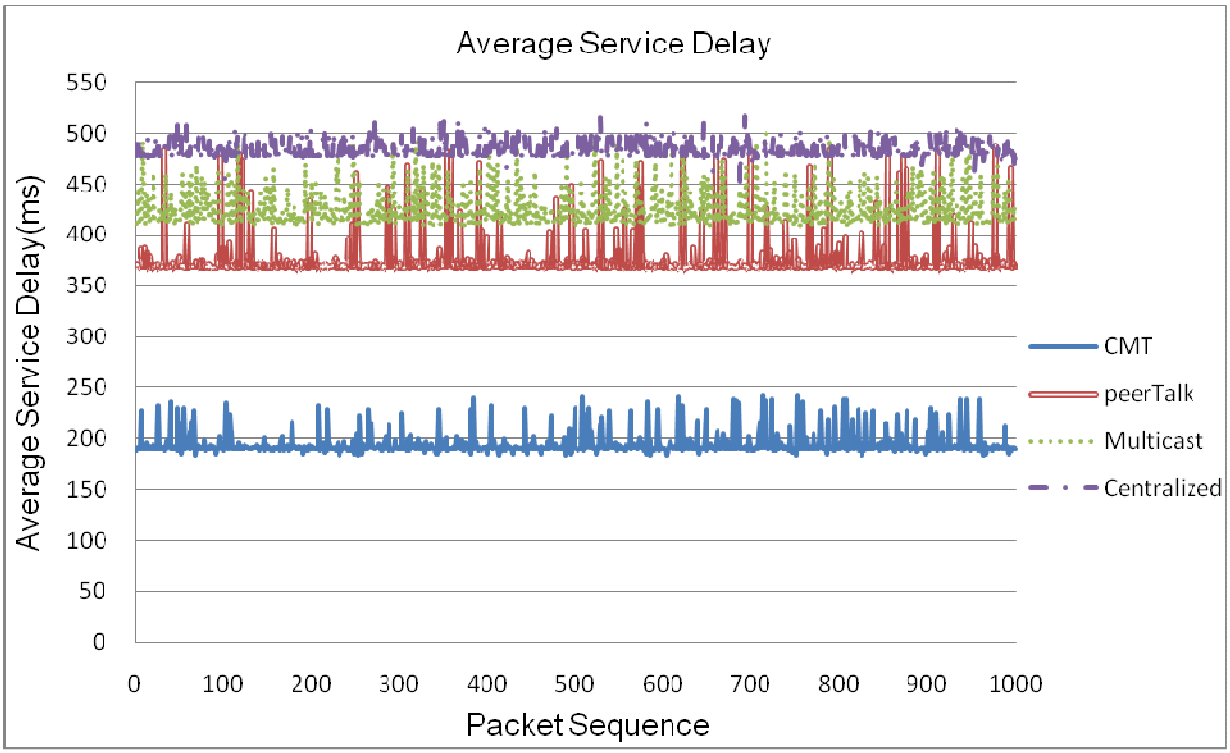

圖 43：Cloud-Assisted VoIP Conferencing Aggregated Service Delay（15 人）

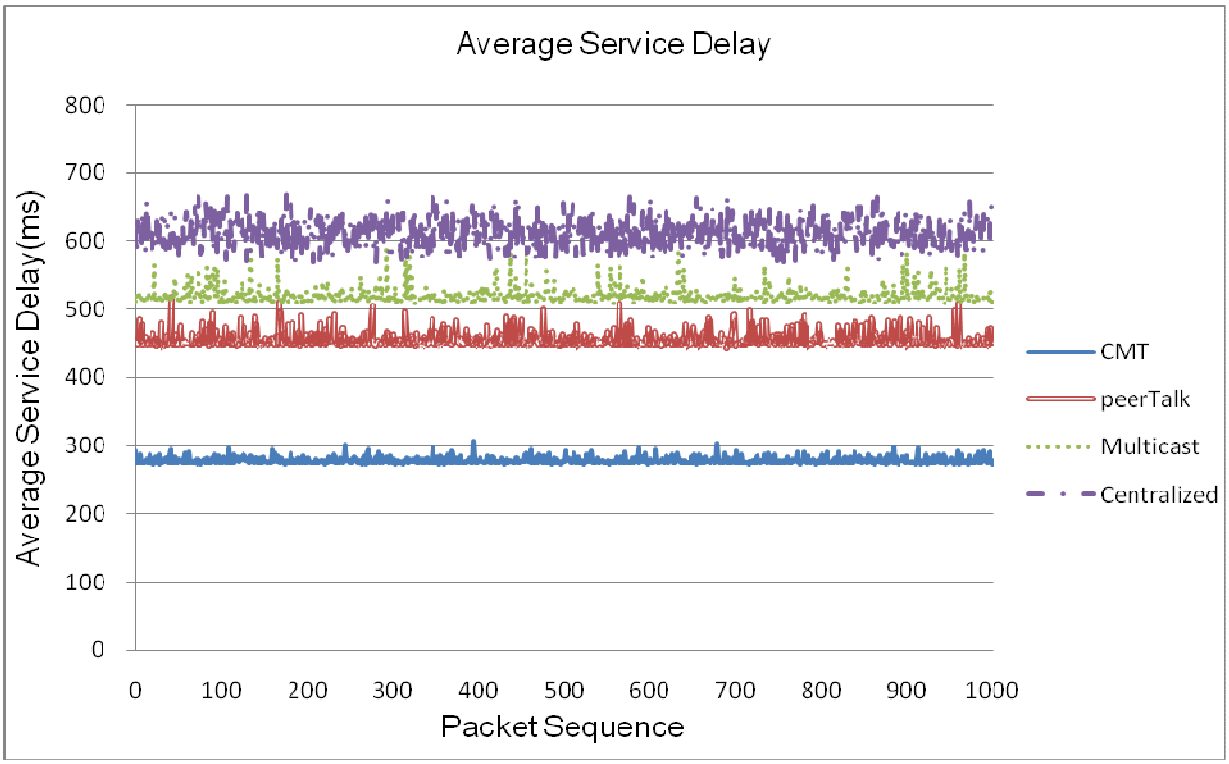

圖 44：Cloud-Assisted VoIP Conferencing Aggregated Service Delay（20 人）

由實驗結果可以看出，利用 Google 雲端運算的協助，可以幫助網路品質較不穩定之節點改善整體的 service delay，在實驗中可降低平均 service delay 約 50ms 至 70ms。

第五章　結論

本研究針對大型網路語音會談中，與會人數過多與距離過遠造成通話品質不佳之問題，就現有解決方法、需求面及會談成員間的互動等各種角度進行分析，歸納現有解決方法不足以支援大型網路語音會談之原因。在一般會議進行時，若有兩人以上同時發話，通常無法有效溝通而必須循重講一次才能順利溝通，因此本研究假假設會議進行中大部份時間僅有一位說話者，並為每位使用加上 VAD 機制消除靜音，阻擋非發話者的聲音封包，降低網路負擔，並刪除混音需求。

在網路拓樸方面，我們將地理位置相近之使用者分群(Cluster)，群組內採用 Full Mesh Network 相連，並將各群組以一個複雜度較小的 Minimum Loss Diameter Spanning Tree 作為 multicasting tree 連接起來，進一步控制傳輸時間及封包遺失率。我們並研究採用雲端計算資源之協助架構進一步提升本傳輸機制的擴充性，幫助網路品質較不穩定之節點改善整體 service delay。

在 PlanetLab 上的實驗結果顯示我們的方法在與會人數達到 20 人時，仍能有效控制延遲時間在 350ms 上下，封包遺失率在 2.5%以下，若利用雲端計算資源之協助架構，則能將傳輸延遲時間減少 50 至 70ms 左右，我們可以預期運用本方法於實際網路時，將有更佳的效能表現。未來本研究除了以提高與會人數上限為目標以外，還可以針對雲端運算的協助，探討雲端運算在大型網路語音會談中，是否有更良好的運用方式，更進一步的改善整體通話品質。

參考文獻

[1] S. Banerjee, B. Bhattacharjee, and C. Kommareddy, "Scalable application layer multicast", in Proc. of ACM SIGCOMM, Pittsburgh, PA, August 2002.

[2] A. Bharambe, V. Padmanabhan, and S. Seshan, "Supporting Spectators in Online Multiplayer Games", in Proc. of HOTNETSIII, San Diego, November 2004.

[3] Uyless Black, "VOICE OVER IP", New Jersey: Prentice Hall, 2000.

[4] M. Castro, P. Druschel, A.-M. Kermarrec, A. Nandi, A. Rowstron, and A. Singh, "SplitStream: High-Bandwdith Multicast in Cooperative Environments", in Proc. of ACM SOSP 2003, New York, October 2003.

[5] M. Castro, P. Druschel, A.-M. Kermarrec, and A. Rowstron, "SCRIBE: A large-scale and decentralized application-level multicast infrastructure", in Proc. of ACM SIGCOMM, San Diego, CA, August 2001.

[6] Li-Chen Chi, "Echo Cancellation in Large-Scale VoIP conferencing", 2009.

[7] Wei-Chung Chiu, "Quality Assurance of Streaming Data Dissemination over P2P Network", 2009.

[8] Y. H. Chu, S. G. Rao, S. Seshan, and H. Zhang, "Enabling Conferencing Applications on the Internet using an Overlay Multicast Architecture", in Proc. of ACM SIGCOMM, San Diego, CA, August 2001.

[9] B. Goode, "Voice over Internet Protocol (VoIP)", in Proc. OF THE IEEE, VOL.90, NO.9, Sep. 2002.

[10] Xiaohui Gu, Z. Wen, Philip S. Yu, and Zon-Yin ShaeZhen, "peerTalk: A Peer-to-Peer Multi-Party Voice-Over-IP System", in Parallel and Distributed Systems, IEEE Transactions, April 2008.

[11] Xiaohui Gu, Z. Wen, Philip S. Yu, and Zon-Yin ShaeZhen, "Supporting Multi-Party Voice-Over-IP Services with Peer-to-Peer Stream Processing", in Proc. of ACM Multimedia, Singapore, November 2005.

[12] D. Kostic, A. Rodriguez, J. Albrecht, and A. Vahdat, "Bullet: High Bandwidth Data Dissemination Using an Overlay Mesh", in Proc. of ACM SOSP 2003, New York, October 2003.

[13] J. Lennox and H. Schulzrinne, "A Protocol for Reliable Decentralized Conferencing", in NOSSDAV'03, Monterey, California, USA, June. 2003.

[14] B. Li and H. Yin, "Peer-to-Peer Live Video Streaming on the Internet: Issues, Existing Approaches, and Challenges", in IEEE Communications Magazine, Toronto, Ont., Canada, June 2007.

[15] M. Radenkosvic and C. GreenHalgh, "Multi-party Distributed Audio Service with TCP Fairness", in Proc. of ACM Multimedia, 2002, Juan-les-Pins, France, December 2002.

[16] Google Talk, http://www.google.com/talk/intl/zh-TW/.

[17] G.723, http://www.itu.int/rec/T-REC-G.723/e.

[18] G.729, http://www.itu.int/rec/T-REC-G.729/e.

[19] Mean Opinion Score, http://en.wikipedia.org/wiki/Mean_opinion_score.

[20] Skype, http://www.skype.com.

[21] The E-Model, http://www.itu.int/rec/T-REC-G.107.