# 國 立 交 通 大 學

## 資訊科學與工程研究所

## 碩 士 論 文

點對點行動即時視訊廣播服務
之設計與實現

The Design and Implementation of P2P Live Video
Broadcasting Services in Mobile Environment

研 究 生：黃名杰

指導教授：曹孝櫟 教授

中 華 民 國 九 十 六 年 七 月

# 點對點行動即時視訊廣播服務之設計與實現
# The Design and Implementation of P2P Live Video Broadcasting Services in Mobile Environment

研 究 生：黃名杰　　Student：Ming-Chieh Huang

指導教授：曹孝櫟　　Advisor：Shiao-Li Tsao

國 立 交 通 大 學

資 訊 科 學 與 工 程 研 究 所

碩 士 論 文

A Thesis

Submitted to Institute of Computer Science and Engineering

College of Computer Science

National Chiao Tung University

in partial Fulfillment of the Requirements

for the Degree of

Master

in

Computer Science

July 2007

Hsinchu, Taiwan, Republic of China

中華民國九十六年七月

# 國　立　交　通　大　學

## 博碩士論文全文電子檔著作權授權書

（提供授權人裝訂於紙本論文書名頁之次頁用）

本授權書所授權之學位論文，為本人於國立交通大學 資訊科學與工程 系所 ＿＿＿＿＿＿組， 九十五 學年度第 二 學期取得碩士學位之論文。

論文題目：點對點行動即時視訊廣播服務之設計與實現

指導教授：曹孝櫟

■ 同意　□不同意

本人茲將本著作，以非專屬、無償授權國立交通大學與台灣聯合大學系統圖書館：基於推動讀者間「資源共享、互惠合作」之理念，與回饋社會與學術研究之目的，國立交通大學及台灣聯合大學系統圖書館得不限地域、時間與次數，以紙本、光碟或數位化等各種方法收錄、重製與利用；於著作權法合理使用範圍內，讀者得進行線上檢索、閱覽、下載或列印。

| 論文全文上載網路公開之範圍及時間： | |
|---|---|
| 本校及台灣聯合大學系統區域網路 | ■ 中華民國 99 年 7 月 19 日公開 |
| 校外網際網路 | ■ 中華民國 99 年 7 月 19 日公開 |

**授 權 人：黃名杰**

**親筆簽名：＿＿＿＿＿＿＿＿＿＿＿＿**

**中華民國 九十六 年 七 月 十九 日**

# 國　立　交　通　大　學

## 博碩士紙本論文著作權授權書

（提供授權人裝訂於全文電子檔授權書之次頁用）

本授權書所授權之學位論文，為本人於國立交通大學 資訊科學與工程 系所 ______________ 組， 九十五 學年度第 二 學期取得碩士學位之論文。

論文題目：點對點行動即時視訊廣播服務之設計與實現

指導教授：曹孝櫟

■ 同意

本人茲將本著作，以非專屬、無償授權國立交通大學，基於推動讀者間「資源共享、互惠合作」之理念，與回饋社會與學術研究之目的，國立交通大學圖書館得以紙本收錄、重製與利用；於著作權法合理使用範圍內，讀者得進行閱覽或列印。

本論文為本人向經濟部智慧局申請專利(未申請者本條款請不予理會)的附件之一，申請文號為：________________________，請將論文延至 99 年 7 月 19 日再公開。

**授 權 人：黃名杰**

**親筆簽名：________________________**

**中華民國 九十六 年 七 月 十九 日**

# 國家圖書館博碩士論文電子檔案上網授權書

ID:GT009455590

本授權書所授權之論文為授權人在國立交通大學 資訊 學院 資訊科學與工程 系所＿＿＿＿＿組 九十五 學年度第 二 學期取得碩士學位之論文。

論文題目：點對點行動即時視訊廣播服務之設計與實現

指導教授：曹孝櫟

茲同意將授權人擁有著作權之上列論文全文（含摘要），非專屬、無償授權國家圖書館，不限地域、時間與次數，以微縮、光碟或其他各種數位化方式將上列論文重製，並得將數位化之上列論文及論文電子檔以上載網路方式，提供讀者基於個人非營利性質之線上檢索、閱覽、下載或列印。

※ 讀者基於非營利性質之線上檢索、閱覽、下載或列印上列論文，應依著作權法相關規定辦理。

**授權人：黃名杰**

**親筆簽名：＿＿＿＿＿＿＿＿**

**民國 九十六 年 七 月 十九 日**

1. 本授權書請以黑筆撰寫，並列印二份，其中一份影印裝訂於附錄三之二(博碩士紙本論文著作權授權書)之次頁；另一份於辦理離校時繳交給系所助理，由圖書館彙總寄交國家圖書館。

點對點行動即時視訊廣播服務之設計與實現

學生：黃名杰 指導教授：曹孝櫟 博士

國立交通大學資訊科學與工程研究所碩士班

# 摘 要

基於無線網路與行動運算技術的快速發展，我們可以在任何時間任何地點，將周遭有趣的事物拍攝下來，並透過無線網路即時與好友們分享。為了實現此一服務，傳統主從式架構，也就是所有收看端直接向發送端取得影音串流，將遭遇可擴充性問題。為解決此問題，一種可行的方式為使用點對點技術，使用應用層群體廣播來發送即時影音。然而，此服務中，由於各節點可能為具有高度可移動性且收訊不穩定的行動節點，這樣的情況下要建構穩定的應用層群體廣播樹，是個極具挑戰的研究課題。本論文中，我們設計且實現行動即時視訊廣播服務，使用點對點應用層群體廣播架構來提升服務之可擴充性，並提出最佳化機制來建構行動環境下穩定的應用層群體廣播樹。模擬結果顯示，本論文所提方案確實減少接收端平均收看延遲時間及服務中斷次數。

The Design and Implementation of P2P Live Video Broadcasting Services in Mobile Environment

Student：Ming-Chieh Huang Advisors：Dr. Shiao-Li Tsao

Institute of Computer Science and Engineering
National Chiao Tung University

# ABSTRACT

Advances in wireless networks and mobile computing technologies, it becomes possible to use a mobile device taping video at anytime and anywhere, and share the live video with friends in real-time through wireless networks. To realize this service, conventional client-server approaches which all receivers have to connect to the live video source, i.e. the mobile device, suffer from serious scalability problems. One possible solution to resolve the scalability issue is to apply peer-to-peer technologies and implement the system by using application layer multicast (ALM) scheme over receiver nodes. However, considering receivers that could be relay nodes and mobile nodes with high mobility, unstable wireless channels and bandwidth, to construct a stable ALM tree for relaying live video to all receivers becomes a very challenging research topic. In this thesis, the design and implementation of a live video sharing service in a mobile environment is presented. We apply P2P ALM schemes to improve the scalability of the services and propose ALM tree optimization schemes for constructing a stable ALM tree in mobile environment. Simulation results demonstrate that the proposed scheme reduce the average initial playback delays and service disruption during playbacks on receivers.

# 誌　　謝

本論文得以順利完成，首先要感謝我的指導教授 曹孝櫟 博士，兩年來，不僅給予研究上紮實的指教，更以身作則教導我們待人處世的道理，良師益友莫過於此。同時也要感謝口試委員 彭文志 教授和 徐玉青 博士，他們給予許多寶貴的意見，使本論文更趨完整。另外也要感謝實驗室的學長、同學和學弟們，在研究過程中給予許多鼓勵與幫助。在此特別感謝建明學長，將一甲子的功力毫不保留地傳授我們，並不厭其煩提供許多寶貴經驗；一正學長則給予許多背景知識的指導；中暉和誌謙也在我最需要的時候，給予各方面適時的鼓勵和幫助；雅筑對於論文內容更提出許多精闢的見解；彥筑和政龍則是研究當中互相砥礪的好夥伴；宥霖學長、建臻、家祥和世永則給予許多精神上的支持與幫助，謝謝大家。此外，還要感謝教會和團契大哥大姐和學弟妹們的關心，讓我在研究過程中更有衝勁。最後，要感謝我的家人，有家人的全力支持，讓我得以無後顧之憂地完成研究。當然，還有許多尚未提到的人事物，快炒店老闆好吃的便當是我體力的來源、書桌旁的二手吉他是我紓解壓力的良伴、耐操的小 90 機車使我得以迅速往返博愛和光復校區進行研究、博愛校區路旁的鳳凰花樹提醒我要更加倍努力進行研究、……等，要感謝的人太多，就感謝天吧！感謝主。

# 目　　錄

# 表　　目　　錄

# 圖　目　錄

# Chapter 1. Introduction

In recent years, the population of mobile device users is tremendously increasing. The computation power and multimedia functions on mobile devices are improved as well. We can take pictures, watch videos, and even record videos with our mobile devices easily. At the same time, broadband wireless networks, such as 3G, WLAN, WiMAX, etc., are fast developed and put into practice widely. As a result, it is easy to share or get real-time multimedia with mobile devices anywhere, at any time. Certainly, the combination of multimedia mobile devices and broadband wireless networks will be a killer application in the near future. And we call this service LIVING (LIve Video sharING). Figure 1 shows an example of LIVING service. While watching a baseball game, Charles can use his mobile phone to record live video and share the real-time live video with remote friends, Rita and Mel, through wireless networks.


Figure 1: The imagination of live video broadcasting service

To realize this kind of services, the simplest way is to record and upload real-time live streaming to a centralized server, which then sends duplicates to subscribers. However, it is not scalable when more and more users join the service so that we have to find other ways to achieve our goal. After surveying, we collect possible choices: server-based, multi-unicast, broadcast, IP Multicast, application-level infrastructure, ALM (Application Layer Multicast) [1], and chunk-driven multicast [2]. Table 1 shows the comparison of these solutions. For better scalability, less complexity and less peer lags, we choose ALM using P2P (Peer-to-Peer) idea to broadcast information to specific users, as our solution. In P2P idea, each user contributes some of their resources, like computation power and bandwidth. With more and more users' joining, the overall capability is increased. Meanwhile, it is not necessary to set a server which may take high maintenance cost. Figure 2 shows a sketch of P2P live video broadcasting service.

Table 1: The comparison of live video broadcasting service solutions

| | Pros | Cons | Examples |
|---|---|---|---|
| Server-based | simple | scalability | N/A |
| Unicast | simple | scalability | N/A |
| Broadcast | simple | mass trash | N/A |
| IP Multicast | good performance | complex state maintenance, lack of higher level features, infrastructure cost and billing issue | N/A |
| Application-level Infrastructure | centralized control | scalability | Akamai, Read Broadcast Network |
| Application Layer Multicast | scalable and simple | worse performance than IP Multicast | Yoid, Narada, Overcast, CAN, NICE, HMTP, Zigzag, Coopnet |
| Chunk-driven | playback continuity and resource usage rate | long start-up delay and peer lags | CoolStreaming, PPLive, ppStream, VVSKY, TVAnts, FeiDian |

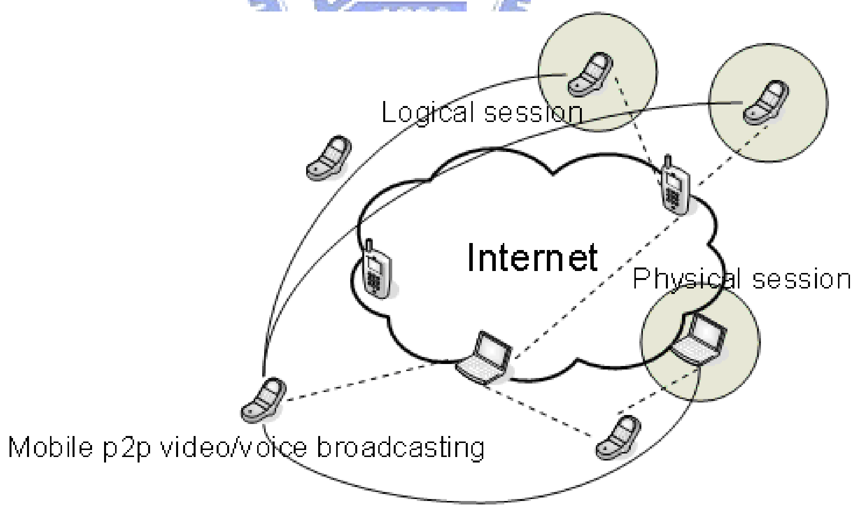


Figure 2: P2P live video broadcasting service

Unfortunately, most existing ALM designs are developed over wired networks rather than wireless networks. However, LIVING runs in mobile environment and some mobility characteristics, such as handoff interruptions, dynamic bandwidth and limited resources, etc., may make them unstable. In this thesis, we analyze the mobility problem and design a live streaming ALM system based on mobile environment.

The rest of this thesis is organized as follows. Chapter 2 reviews related work of ALM. Chapter 3 presents theoretical analysis and design of LIVING. Then we describe simulation and results in Chapter 4. In Chapter 5, we implement LIVING on mobile devices. Finally, we conclude with a summary and discussion of future work in Chapter 6.

# Chapter 2. Related Work

LIVING can be briefly divided into two procedures: resource lookup and resource retrieval. For resource lookup method, related file-sharing systems, such as Napster [3], Gnutella [4], Chord [5], CAN [6], etc., are well-analyzed and well-designed. And we directly apply existing P2P file-sharing resource lookup methods in our system. For resource retrieval method, ALM technology is chosen as mentioned above. In this chapter, we introduce ALM related works, and then apply the idea into our LIVING design.

## 2.1. Background and Brief History of ALM

If some data is going to be sent to specific group members, we can simply unicast many times or broadcast to everyone. However, these methods are not scalable and may waste lots of internet resources. To solve these problems, the idea of multicast is proposed. The question is: should multicast be implemented at network layer or at application layer? In 1988, Deering proposed IP Multicast [7], and related protocols, such as IGMP [8], DVMRP [9], PIM [10], etc., were proposed in few years. However, some technical and non-technical limitations point out the drawbacks of IP Multicast. These limitations include the complexity of maintaining per group status and routing tables, the replacement or upgrades of existing large number of routers, and the pricing model between different ISPs.

As a result, researchers started to pay attention to application layer multicast, also known as ALM. Similar to IP Multicast, ALM builds multicast trees at application layer using P2P mechanism. Each peer stores part of information and provides some of its resources. To be compared with unicast or broadcast, ALM not only saves unnecessary waste of internet resources, but also accurately sends data to each group members. On the other hand, to be compared with IP Multicast, ALM always has worse performance. Nonetheless, ALM does overcome all the limitations from IP Multicast. In 2000, Yoid [11] and Narada [12] were proposed respectively as the beginning of ALM researches. And many different ALM designs were proposed in few years. For example, Overcast [13] was proposed in 2000, TBCP [14] and CAN [15] were proposed in 2001, switch-trees [16], HMTP [17] and NICE [18] were proposed in 2002, and Zigzag [19] and CoopNet [20] were proposed in 2003. The evolutionary timeline is shown in Figure 3.

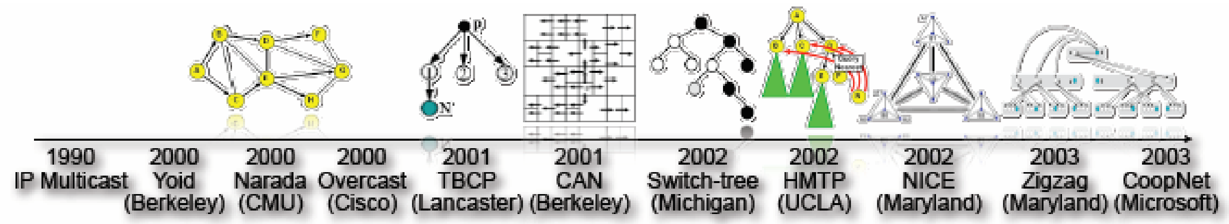


Figure 3: Timeline of several ALM systems

## 2.2. Classification and Comparisons of ALM Systems

In order to have a clear concept, some researchers surveyed, classified and compared existing ALM systems. [21], [22] and [23] classified ALM systems by overlay construction, data delivery, maintenance and optimization. And [24] did a complete performance comparison on these various systems. Figure 4 depicts the idea of different overlay construction approaches. In mesh-first approach, group members form a mesh network first, and a multicast tree is established by some rules, such as Reverse Path Forwarding (RPF), while multicast is proceeding. In tree-first approach, group members form a multicast tree directly, and a mesh network is built based on the multicast tree for fault tolerance or optimization. In implicit approach, group members form a structured control topology, and the multicast tree is implicitly defined in the topology. In this thesis, we use tree-first approach in our system design.

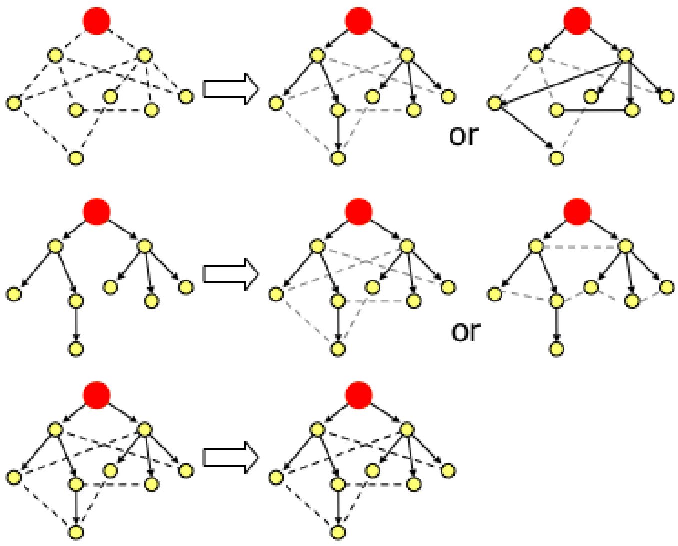


Figure 4: The idea of mesh-first, tree-first and implicit approaches

Generally, ALM may send data multiple times over one physical link or extend data transmission delay. For analysis, there are three metrics commonly used to evaluate the performance of an ALM system: stress, stretch and resource usage. Stress means the times the same data been sent through one physical layer link. The larger the stress is, the more the internet resources are wasted. Stretch is the ratio of delay in ALM system compared with unicast delay. The larger the stretch is, the more the delay time. Resource usage is the amount of total data flows, which can be presented as $\Sigma_{i = 1 \text{ to } L}$ $d_i$ * $s_i$, where L is the number of active physical links covered by the overlay tree, $d_i$ is the delay of link i and $s_i$ is the stress of link i. Figure 5 shows the concept of stress and stretch.

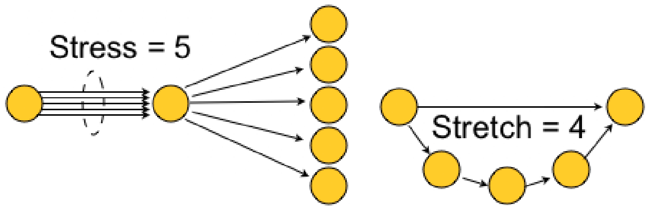


Figure 5: The concept of stress and stretch

On top of the three metrics, two widely studied performance goals are cost and delay optimizations. Tree cost is the summation of all tree links' delay, which can be viewed as resource usage. And delay, which can be roughly viewed as stretch, is a critical issue for real-time applications. The minimum cost spanning tree and star topology provide the best solutions to these goals respectively. However, these problems are proven to be NP-hard when a degree constraint is enforced [25][26][27]. In this thesis, our performance goals are mobility and delay optimizations, where mobility is an important issue for mobile environment, and delay is a vital issue for real-time live streaming applications.

# Chapter 3. LIVING Design

LIVING (LIve Video sharING) is a P2P live video broadcasting service in mobile environment. When someone wants to share her/his real-time live video to others, she/he can publish it onto the P2P network. Meanwhile, when someone wants to watch specific real-time live video, she/he can search for it, join the multicast system, and start watching it. Figure 6 shows a usage scenario of LIVING. Someone is watching a baseball game and wants to share the view from her/his seat with others. She/he records the game with camera phone and publishes it onto the P2P network. Her/his friends or others can search for it through the P2P network, join the multicast tree, and then start watching the real-time live video.


Figure 6: A usage scenario of LIVING

With mobility, LIVING has more characteristics needed to be concerned. First, when some mobile device is moving away from the area covered by one AP and entering the area covered by another AP, a handoff takes place and causes service interruption. In other words, mobile devices have more probability to have service interruptions which lower down the viewing quality. Moreover, the interference or variant distances between devices and access points may cause data loss or bandwidth variations. That is to say, it is more probable for mobile devices to have dynamic bandwidth or unpredictable failures. What's more, mobile devices are usually small-sized embedded systems so that system resources, such as CPU frequency and memory size, are poorer than general personal computers. To sum up, mobility causes handoff interruptions, dynamic bandwidth, and more failure rate to mobile nodes. These characteristics produce undesirable changes to the system, especially in real-time live streaming service, which is sensitive to data loss. As a result, we have to minimize the effects of mobility.

## 3.1. Problem Statement

To have a clear view on mobility problem, we try to describe the system in graph theory. Because LIVING runs at application layer, each pair of nodes can be viewed connected. And the topology G(V, E) of N nodes can be viewed as a complete graph $K_N$. Based on this graph, we can get spanning forest rooted from source S as our application layer multicast trees T. For example, in Figure 7, six nodes form a complete graph $K_6$. The dotted lines between each pair of nodes are application layer links and the spanning tree T rooted from S is shown in solid arrows, where the thickness of the lines represents the available bandwidths between each pair of nodes.

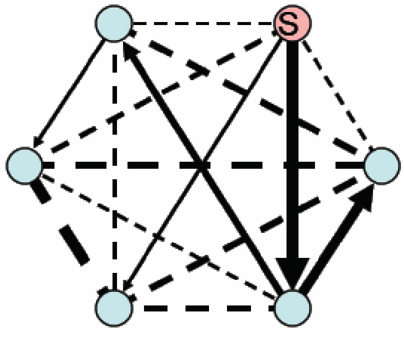


Figure 7: ALM topology

For a single node, we assume that the bandwidth between each pair of nodes is more than or equal to the playback rate R. At the same time, the in-degree allocated bandwidth $albw_{in\_i}$ should be equal to R, and less than or equal to the in-degree bandwidth $bw_{in\_i}$, i.e., $R = albw_{in\_i} \leq bw_{in\_i}$. The out-degree allocated bandwidth $albw_{out\_i}$ should be more than or equal to the playback rate R, and less than or equal to the out-degree bandwidth $bw_{out\_i}$, i.e., $R \leq albw_{out\_i} \leq bw_{out\_i}$. Figure 8 illustrates a live streaming multicast model of a single node.

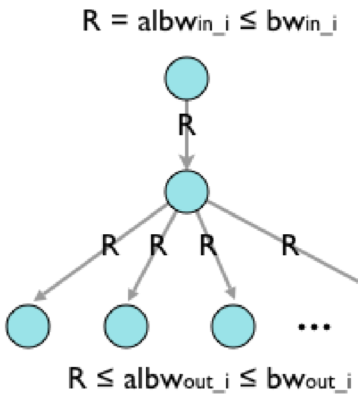


Figure 8: A live streaming multicast model of a single node

To analyze the effect of mobility, we define $m_i$ as the mobility probability of node i. Furthermore, because the movement of any nodes on root path of node i may make streaming unstable to node i, we define path mobility probability $M_i$, which can be expressed as $M_i = 1 - (1 - M_{parent(i)})(1 - m_i)$. And average path mobility probability P, which can be expressed as $P = (\Sigma_{i = 0 \text{ to } N-1} M_i) / N$, is further defined. To make the system more stable, we have to work out how to minimize the average path mobility P.

To make a conclusion, we simplify the mobility optimization problem of P2P live video broadcasting service in mobile environment as follows:

*Given G(V, E), find a spanning tree T rooted at source S with minimum average path mobility probability P, which fits the in/out-bandwidth of each node and the available bandwidths between each pair of nodes.*

Similar to delay optimization problem, a star topology is the best solution to mobility optimization problem. However, while a degree constraint is enforced, we conjecture that computing a tree with minimum P and bounded degree is NP-hard as well. Consequently, we do not have further best solution discussions, but focus on heuristic algorithm design.

## 3.2. System Design

In this section, we present LIVING, a protocol designed to implement P2P live video broadcasting service in mobile environment. In designing LIVING, some issues should be taken into consideration. First, basic ALM metrics, i.e., stress, stretch and resource usage, should be tuned as fine as possible so that the basic performance could be acceptable to users. Second, we have to consider some live media streaming characteristics, such as live, sensitive to data loss and timeliness constrains. Besides, in mobile environment, mobility causes handoff interruptions, dynamic bandwidth, and more failure rate. We must minimize the impacts of these situations. Figure 9 illustrates some mobility issues. In panel (a), the yellow node indicates a mobile node. In panel (b), the available bandwidths between the mobile node and its neighbors shrink while it is moving. In panel (c), the links between the mobile node and its neighbors are broken due to handoff or unexpected failure.

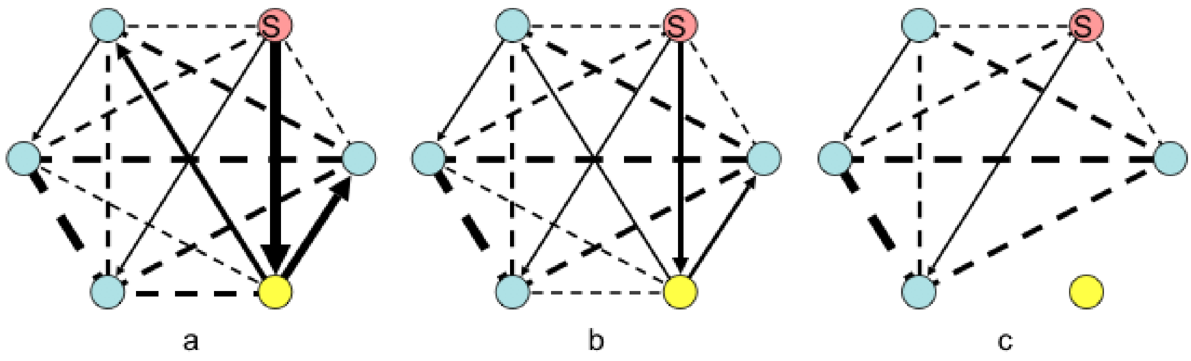


Figure 9: Some mobility issues.

For this kind of services, the most important things users may concern are start-up delay, peer lags and viewing quality. As a result, we primarily focus on these design considerations. Start-up delay is the time interval between joining and watching. If start-up delay takes too long, users may loss their patience and stop using this service. Peer lags are the streaming lags between source and destinations. For real-time live services, long lags make users unwilling to continue watching. In mobile environment, viewing quality mainly depends on mobility, which causes service interruptions to users. Therefore, to better viewing quality, we have to reduce the effect caused from mobility, especially handoff interruptions.

For high scalability, LIVING is designed using P2P mechanism in both resource lookup and resource retrieval procedures. For resource lookup, structured P2P search, which bounds the search time in log(n), is a good choice. For resource retrieval, an ALM protocol is designed with consideration to the goals mentioned above. Briefly, the system flow can be divided into following steps: 1) join P2P search network, 2) search for the live streaming, 3) join the ALM tree, and 4) start receiving the streaming. For example, in Figure 10, node 1 wants to watch node 15's streaming, she/he searches for it and gets node 15's location. Then in Figure 11, node 1 joins node 15's ALM tree and start watching the streaming.

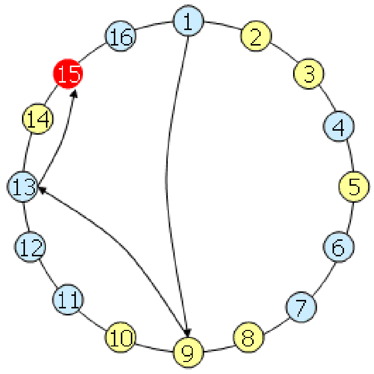


Figure 10: Finding the resource with structured P2P search

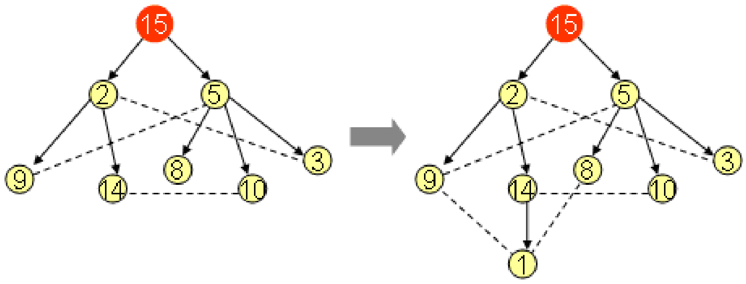


Figure 11: Joining the ALM tree and start watching the streaming

### 3.2.1. Maintenance

To evaluate system performance, make decisions and do adjustments, each node maintains some information, including mobility probability $m_i$, path mobility probability $M_i$, peer lags $D_i$, available out-degree $aod_i$, aggregated information $Aggr_i$ and some backup links.

To explicitly evaluate the effect of service interruptions caused by mobility, we view handoffs as the main character of mobility. Consequently, for each node i, we define mobility probability as $m_i = t_{m\_i} / t_{lifetime\_i}$, where $t_{m\_i}$ is the total service interruption time of node i caused by handoffs of itself, and $t_{lifetime\_i}$ is the total service time of node i. If the $j^{th}$ handoff takes $t_{int\_i\_j}$, we can define $t_{m\_i} = \Sigma_{j = 1 \text{ to number of handoffs of node i}}\, t_{int\_i\_j}$. As a result, $m_i = (\Sigma_{j = 1 \text{ to number of handoffs of node i}}\, t_{int\_i\_j}) / t_{lifetime\_i}$. Figure 12 shows the concept of mobility probability.

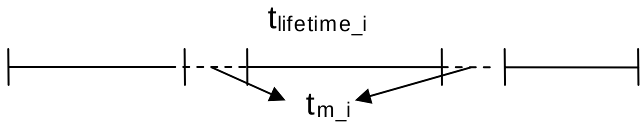


Figure 12: The concept of mobility probability

Besides the handoffs of node i itself, the handoffs of each node on node i's root path also cause service interruptions to node i. To evaluate the effect of service interruptions caused by path mobility, we define path mobility probability of node i as $M_i = 1 - (1 - M_{parent(i)})(1 - m_i)$, which is mentioned in problem description. However, because the system is decentralized, periodic calculation of each node computes approximate $M_i$ rather than exact $M_i$.

To evaluate peer lags of each node, we define peer lags of node i as $D_i = D_{parent(i)} + d_{i, parent(i)}$, where $d_{i, j}$ represents peer lags between node i and node j. Because the system is decentralized, periodic calculation of each node computes approximate $D_i$ rather than exact $D_i$.

To avoid bandwidth overload, each node maintains available out-degree. It can be calculated from $aod_i = [(bw_{out\_i} - albw_{out\_i}) / R]$, where $bw_{out\_i}$ and $albw_{out\_i}$ are described in problem description and shown in Figure 8.

To improve the performance during optimization and make multicast tree balanced, we apply the concept of [28], where each node aggregates its children's information, to get approximate aggregated information $Aggr_i$ as shown in Figure 13. Aggregated information $Aggr_i$ is the overall information of the sub-tree rooted from node i. In our design, the aggregated information is the number of nodes of the sub-tree rooted from node i in order to maximize performance improvement.

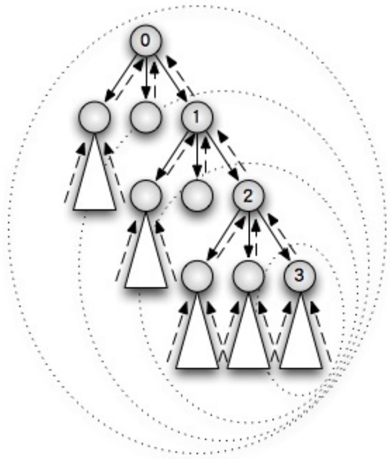


Figure 13: The concept of aggregation

When graceful leave or unexpected failure takes place, nodes are supposed to recover the service themselves. To reduce the variation of peer location after leave or failure recovery, each node maintains some backup links based on its location. The links include every nodes located in n-hop region except the nodes in the sub-tree rooted from itself. Figure 14 shows an example of n-hop region nodes from node 9.

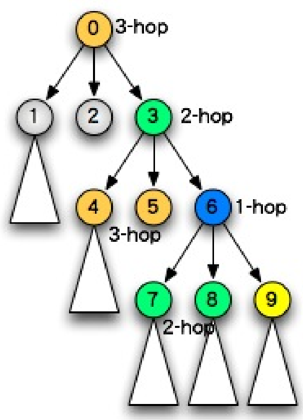


Figure 14: An example of n-hop region nodes

### 3.2.2. Join

In the join procedure, a well-known Rendezvous Point (RP), which periodically crawls and maintains parts of node list from multicast trees, is setup. When a newcomer i wants to join a multicast tree, it gets N random nodes from RP first. We assume this step takes $T_{RP} = RTT_{i, RP} + T_{proc\_RP}$ time, where $RTT_{i, RP}$ is the round trip time between node i and RP, and $T_{proc\_RP}$ is the process time of RP. Then, it sequentially queries the nodes. The queried node replies its parent, children list, peer lags and available out-degree. There will be at most N candidates with available out-degree and the newcomer joins the node with shortest peer lags. If none of the N nodes has available out-degree, the newcomer extensively queries from the N nodes using BFS until meeting a node with available out-degree. The newcomer's parent does optimization after join procedure. We assume each query takes $T_Q = RTT_{i, j} + T_{proc\_j}$ time, where $RTT_{i, j}$ is the round trip time between node i and node j, and $T_{proc\_j}$ is the process time of node j. Figure 15 shows an example of the join procedure. First, the yellow node gets three random nodes 1, 7 and 11 from RP. Then, it sequentially queries the three nodes. If only nodes 7 and 11 have available out-degree, the newcomer joins the one with less peer lags. If all the three nodes have no available out-degree, the newcomer first queries node 1's parent and children list. If node 6 is queried and it has available out-degree, the newcomer joins nodes 6.

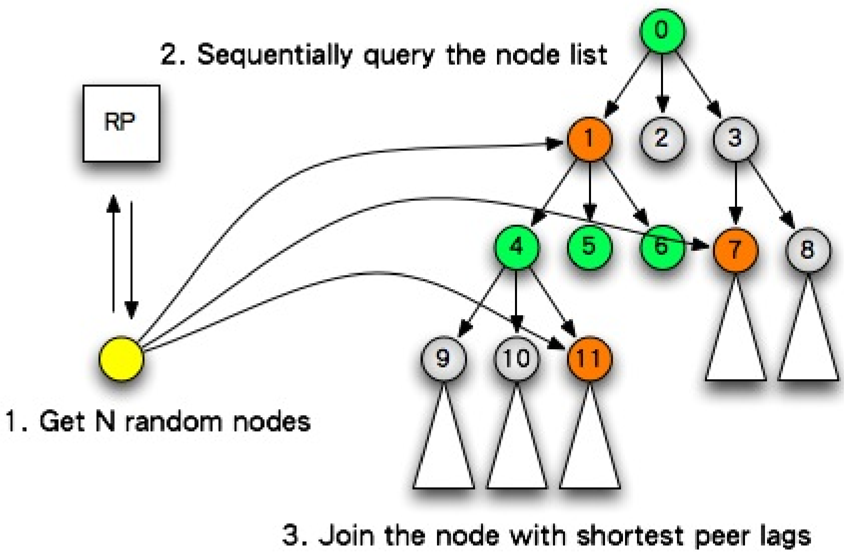


Figure 15: The concept of join procedure

### 3.2.3. Leave

A node notifies and rearranges its parent and children before leaving. We assume the leaving node has $c_i$ children and its parent has $pav_i$ available out-degree including the link between them, where $pav_i = aod_{parent(i)} + 1$. If $pav_i \geq c_i$, every children reconnect to the parent directly. Else if $pav_i < c_i$, only $pav_i$ children with most available out-degree reconnect to the parent, and the rest $c_i - pav_i$ children reconnect to the connected children in available out-degree order using FCFS scheduling. We assume all the children have $cav_i$ available out-degree, where $cav_i = \Sigma_{v \in \text{children of } i}\, aod_v$. If $cav_i \geq c_i - pav_i$, all the children can reconnect to each other. Otherwise, if $cav_i < c_i - pav_i$, there will be $c_i - pav_i - cav_i$ unconnected children. Finally, the unconnected nodes seek and reconnect to their backup links with available bandwidth or rejoin the multicast tree. The parent and all the children do optimization after leave procedure.

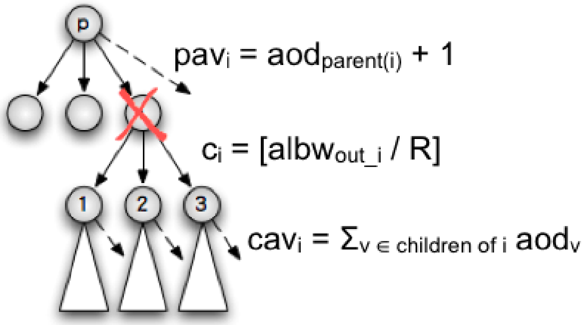


Figure 16: The concept of leave procedure

### 3.2.4. Failure Recovery

While there is something wrong, node i first checks its parent. If parent(i) is still alive, node i waits for parent(i)'s recovery. However, if parent(i) is not alive, node i checks its backup links. If one of the backup links has spare available out-degree, it reconnects to the backup node. If none of the backup links have spare available out-degree, node i rejoins the multicast tree.

### 3.2.5. Optimization

As the system is dynamically changing, some adjustments are needed to make it stronger and more stable. The major principle for optimization is to improve the overall system performance with least negative effects to other nodes. Optimization takes place when periodic events, including information updates and links checks, and join and leave procedures occur.

While optimization takes place, node i first queries the descendants in n-hop region from itself using BFS. When $m_j < m_i$, where node j ∈ descendants of node i in n-hop region from node i, there exists some node i's descendant more stable than node i. To reduce the overall path mobility probability, it is better to promote node j. In this case, node j reconnects to parent(i) and node i reconnects to node j. If $aod_j = 0$, node j makes one of its children with greatest mobility probability reconnects to node i. Figure 17 shows the concept of promotion.

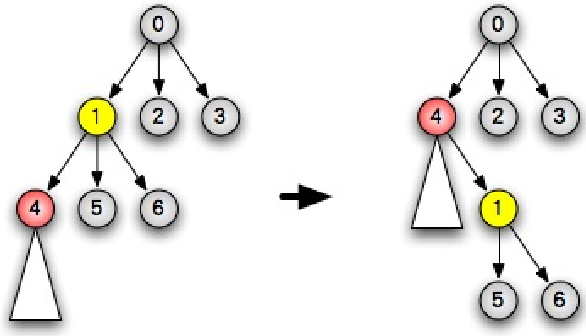


Figure 17: The concept of promotion

Proof:

Figure 18 shows the normal case of promotion.

$M_1 = 1 - (1 - m_1)(1 - M_0)$

$M_n = 1 - (1 - m_n)(1 - m_{n-1})\ldots(1 - m_1)(1 - M_0)$

$M_1' = 1 - (1 - m_1)(1 - m_n)(1 - M_0)$

$M_n' = 1 - (1 - m_n)(1 - M_0)$

$M_1 + M_n = 2 - (1 - m_1)(1 - M_0)(1 + (1 - m_n)(1 - m_{n-1})\ldots(1 - m_2))$

$M_1' + M_n' = 2 - (1 - m_n)(1 - M_0)(1 + (1 - m_1))$

$\because m_2, \ldots, m_{n-1} > m_1 > m_n$

$(1 - m_1) < (1 - m_n)$ and $(1 - m_n)(1 - m_{n-1})\ldots(1 - m_2) < (1 - m_1)$

$\therefore M_1' + M_n' < M_1 + M_n$

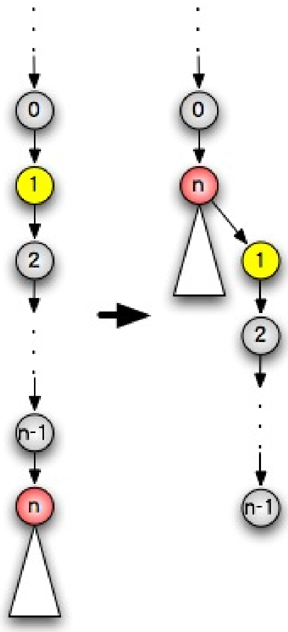


Figure 18: Formal case of promotion

In addition to promotion, when $aod_i \geq 1$, node i has spare available out-degree bandwidth and can carry one descendant up to improve the overall performance, including peer lags and path mobility probability. In our design, node i carries up the grandchild with greatest aggregated information in order to balance the multicast tree at the same time. Node i asks node j, the child with greatest aggregated information, about node k, which is node j's child with greatest aggregated information. Finally, node i makes node k reconnect to node i. Figure 19 depicts the concept of vacancy filling.

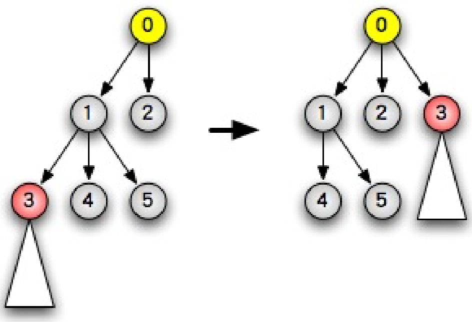


Figure 19: The concept of vacancy filling

Proof:

$\forall$ v $\in$ subtree of node k

$D_v' = D_v - d_{j,k} < D_v$

$M_v' = 1 - (1 - M_v) / (1 - m_j) < M_v$

$\because$ subtree of node k $\supset$ subtree of child of node k

$\Sigma_{\forall v \in \text{subtree of node k}} (D_v - D_v') < \Sigma_{\forall v \in \text{subtree of child of node k}} (D_v - D_v')$

$\Sigma_{\forall v \in \text{subtree of node k}} (M_v - M_v') < \Sigma_{\forall v \in \text{subtree of child of node k}} (M_v - M_v')$

$\therefore$ it is better to bring up upper layer descendant.

### 3.2.6. Playback Adjustment

To prevent discontinuity of streaming playback during location adjustments, each node adjusts the playback speed to make streaming video smooth to users. There are two possible situations.

When node i is relocated from parent with less peer lags to parent with more peer lags, i.e., $D_{i,\, old\_parent(i)} < D_{i,\, new\_parent(i)}$, it disconnects to the old parent and slows down the playback speed until the timestamp meets the new one as shown in Figure 20.

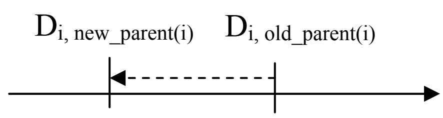


Figure 20: Playback adjustment (slow down)

When node i is relocated from parent with more peer lags to parent with less peer lags, i.e., $D_{i,\, old\_parent(i)} > D_{i,\, new\_parent(i)}$, it saves the streaming received from the new parent to the buffer first. And node i disconnects to the old parent when the timestamp catches the beginning of the buffer. At the same time, node i accelerates the playback speed until the timestamp meets the new one as shown in Figure 21.

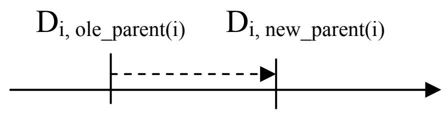


Figure 21: Playback adjustment (accelerate)

# Chapter 4. Simulation

To evaluate the performance of our design, we developed a discrete event simulator that can simulate the behavior of some static and mobile nodes running the service on top of physical layer routers and wireless access points. The simulation, results and some discussions are presented in this chapter.

## 4.1. Simulation Setup

### 4.1.1. Performance Metrics

Our main performance metrics are average viewing quality (VQ), average relative delay penalty (RDP), average start-up delay (SD) and control overhead.

Since the viewing quality is mainly affected by handoff interruptions, we define viewing quality as the proportion of total service interruption time. That is to say, $VQ_i = t_{M\_i} / t_{lifetime\_i}$, where $t_{M\_i}$ is the total service interruption time caused from handoffs of any node on the root path, and $t_{lifetime\_i}$ is the lifetime of the service. If the $j^{th}$ handoff of node i takes $t_{INT\_i\_j}$, we can define $t_{M\_i} = \Sigma_{j = 1 \text{ to number of handoffs}}\, t_{INT\_i\_j}$. And $VQ_i = (\Sigma_{j = 1 \text{ to number of handoffs}}\, t_{INT\_i\_j}) / (t_{lifetime\_i})$. Less handoff leads to better viewing quality.

To evaluate peer lags, we can simply apply stretch, which is defined as (end-to-end delay of node i and the root using the overlay tree) / (end-to-end delay of node i and the root using unicast), as our performance metric. However, it is difficult to conjecture unicast paths. What's more, unicast paths are generally equal to the shortest paths. To have an explicit evaluation, we define another performance metric $RDP_i$ = (end-to-end delay of node i and the root using the overlay tree) / (end-to-end delay of node i and the root using network layer shortest path), where the delay of two nodes is the physical layer hop count between them in simulations.

To evaluate start-up delay, we define $SD_i = T_{RP} + k * T_Q$, where $T_{RP} = RTT_{i, RP} + T_{proc\_RP}$, $T_Q = RTT_{i, j} + T_{proc\_j}$ as mentioned in system design, and k is the query times. NCHC TWAREN NOC's reports [29] show that the average end-to-end RTT in Taiwan area, where hop count is about less than or equal to 10, is about 4ms to 8ms. And the RTT between Taiwan area and international area, where hop count is about more than 10, is about 150ms to 200ms. As a result, we assume the RTT of hop count of 10 or less is 10ms. Otherwise, the RTT of hop count of 11 or more is 200ms. As regards $T_{proc\_RP}$ and $T_{proc\_j}$, the process is to execute instructions, fetch the data in memory and then return. The memory access time is about 5ns to 7ns [30] and the CPU execution time is less than 1ms for most mobile devices that support multimedia functions. Consequently, we can assume $T_{proc\_RP}$ and $T_{proc\_j}$ as 1ms.

To evaluate the amount of control signals, the number of queries during join, leave, optimization and failure recovery are recorded as control overhead.

#### 4.1.2. Simulation Environment

To calculate RDP, we need physical layer information to get end-to-end delay of each pair of nodes in simulations. We use a number of routers, APs and links to describe physical layer topology. [31] indicates that internet topology follows the characteristic of power-law. Consequently, we choose power-law model as our topology model. Routers and links are generated from Inet-3.0 topology generator [32], which is supported by National Science Foundation, Office of Naval Research and AT&T. As to the number of routers, we refer to [24] and use thousands of routers in simulations. To evaluate the impact of the number of routers, we first simulate 5000 to 10000 routers, and then use default 5000 routers in other simulations.

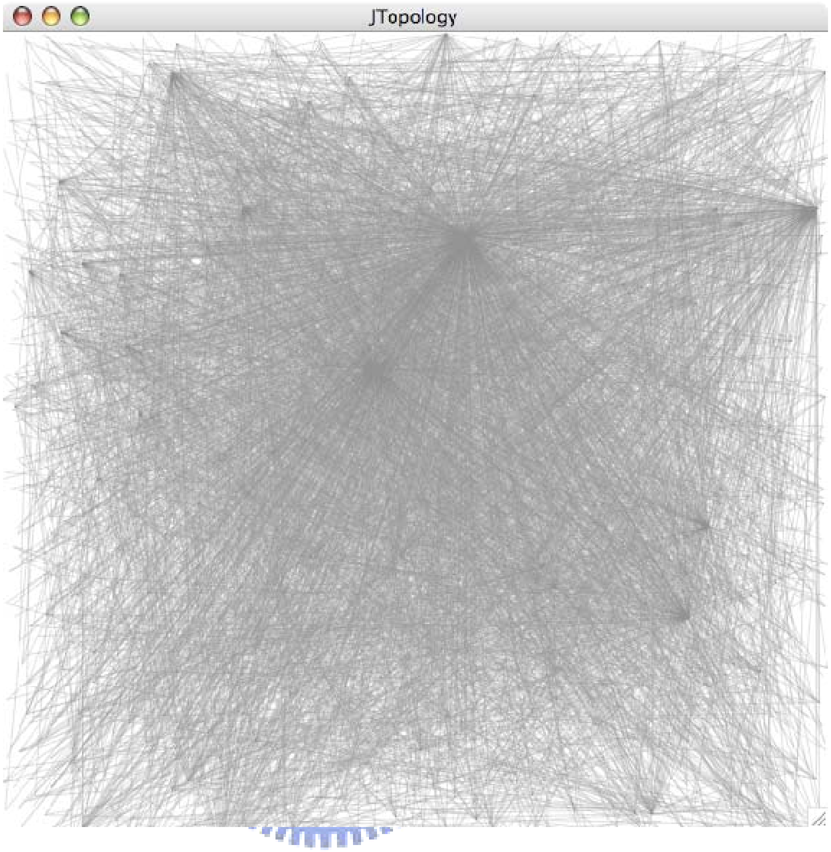


Figure 22: Power-law model topology generated by Inet-3.0

To calculate VQ, handoffs in mobile environment are simulated. We first construct a 10km x 10km 2D coordinate wireless cellular environment, where the radius of AP cells is 100m. Besides, every 12 contiguous APs are grouped as a WLAN hotspot and all the APs in a WLAN hotspot connect to the same randomly selected router. We assume the bandwidths between routers and APs are sufficient to provide the service. The mobile nodes move and change their locations in this environment, and handoffs take place when nodes move into the coverage area of different AP.

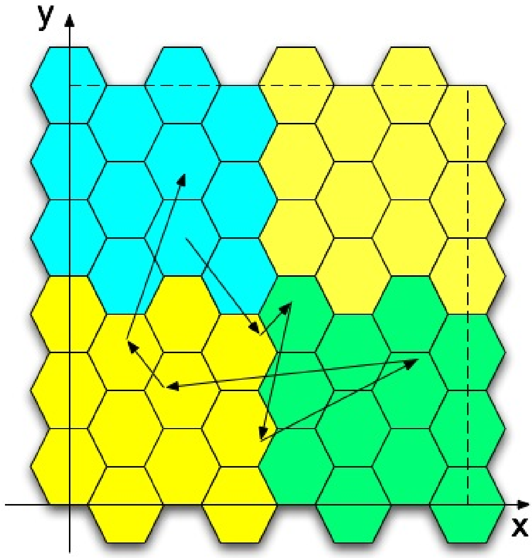


Figure 23: The concept of 1km x 1km wireless cellular environment, WLAN hotspot and waypoint mobility model

To simulate our system, a number of peer nodes are generated and operate on the internet topology. A proportion of nodes are static nodes, which randomly connect to routers, and others are mobile nodes, which connect to APs based on their locations of AP cell coverage area. For each mobile node, waypoint mobility model [33] is applied as the movement policy, where [minspeed, maxspeed] = [0m/s, 20m/s (72km/hr)] and the pause time is randomly selected from 0s to 40s. Besides, [34] indicates that the handoff delay using different protocols is about 1s, so we apply 1s as the handoff delay in simulations.

As regards join procedure, when two nodes attempts to join simultaneously, RP responds node lists one by one. Consequently, we assume that two nodes would not join multicast tree simultaneously. To simulate the case of a concert or a ball game, we make nodes join the multicast tree one at a time every 1s following exponential probability distribution, where λ = 1.0. And the live streaming continues for 7200s, i.e., 2hr, which is about the duration of a concert or a ball game.

### 4.1.3. Benchmarks

We compare our design with three algorithms that construct overlay trees based on complete topology information, including no optimization, delay greedy heuristic algorithm and mobility greedy heuristic algorithm. No optimization represents a scheme that construct a multicast tree using the same join algorithm but do no optimization. For delay greedy heuristic algorithm, we use Compact Tree algorithm [35] as our benchmark. It grows a spanning tree from the root. And a new node with the smallest increment in tree delay is attached to the tree at each round. For mobility greedy heuristic algorithm, we use an algorithm which grows a spanning tree from the root. At each round, a new node with the smallest mobility probability is attached to the lowest level node of the tree.

### 4.1.4. Simulation Parameters

To evaluate and compare our design with benchmarks under different conditions, we adjust some parameters in simulations. While a parameter is evaluated, other parameters are set to the default values. Firstly, to evaluate the effect of number of routers, we simulate the system using different number of routers. This parameter is set to 5000 by default. Secondly, to observe the scalability of our design, we simulate the system using different group sizes. This parameter is set to 500 by default. Moreover, as we can not estimate the proportion of mobile nodes in real world, we simulate the system using different proportion of mobile nodes. This parameter is set to 0.5 by default. Besides, in real world, each peer node has different number of out-degree, which is the maximum number of children a node can serve, i.e., [$bw_{out\_i}$ / R]. And it is difficult to estimate the distribution of number of out-degree. However, this parameter affects the height of multicast trees, as well as the final performance. To reduce the uncertainty from this parameter, we refer to [24] that use fixed number of out-degree in simulations, and have an individual discussion to the effect caused from this parameter. This parameter is set to 8 by default. Furthermore, optimization period, which indicates the period between two periodic events of each peer node, could affect the performance convergence speeds and optimization overhead. Therefore, we simulate the system with different optimization periods. This parameter is set to 50s by default. In addition, in our system design, the range of promotion hop number limits the scope of promotion during optimization. We also simulate the system using different promotion hop number. This parameter is set to 2 by default. Finally, join parameter N is the number of nodes provided by RP at join procedure. We simulate the system using different join parameter N to observe the impact to RDP, SD and join overhead. This parameter is set to 3 by default.

## 4.2. Results

### 4.2.1. Number of Routers

The results in this subsection show the impact of different number of physical layer routers. Figure 24 and Figure 25 plot average VQ and average RDP using different number of routers respectively. We observe that average VQ and average RDP are not affected by number of routers for all the four algorithms. This indicates that we can view our simulation results as the case of running over real-world scale of routers.

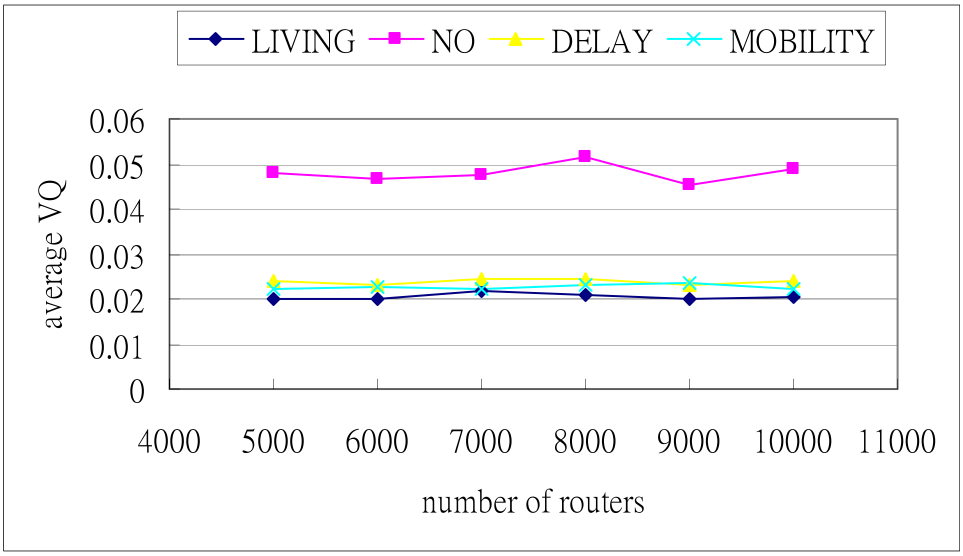


Figure 24: Average VQ using different number of routers

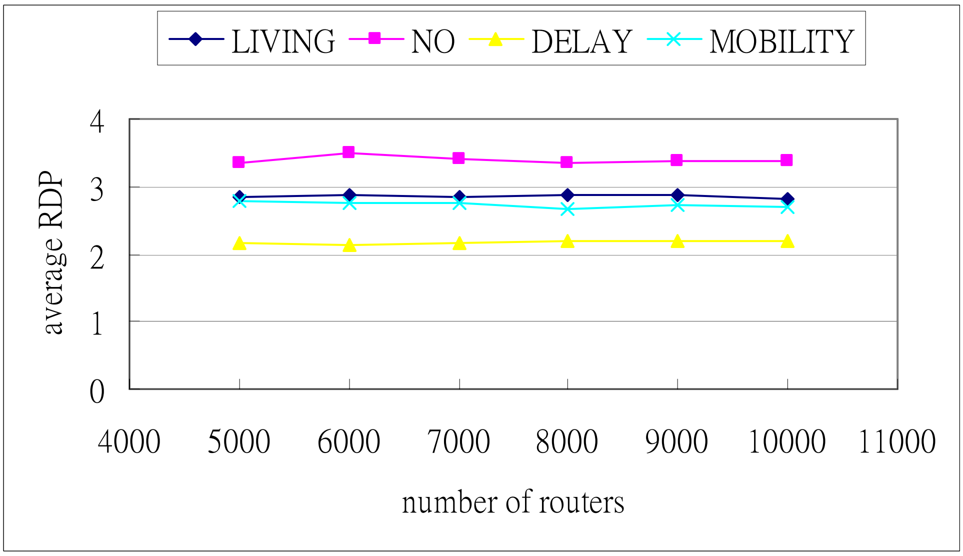


Figure 25: Average RDP using different number of routers

### 4.2.2. Group Size

We now examine the performance with different group size to test scalability of the four algorithms. Figure 26 depicts average VQ with different group sizes. We can see that our LIVING design performs worse than delay and mobility optimizations when the group size is small. However, when the group size grows larger than 400, it performs well, even better than delay and mobility optimizations. For RDP, Figure 27 illustrates average RDP with different group size and average RDP of all the four algorithms grows slowly when the group size is large. Consequently, we conclude that our LIVING design is scalable in terms of group size.

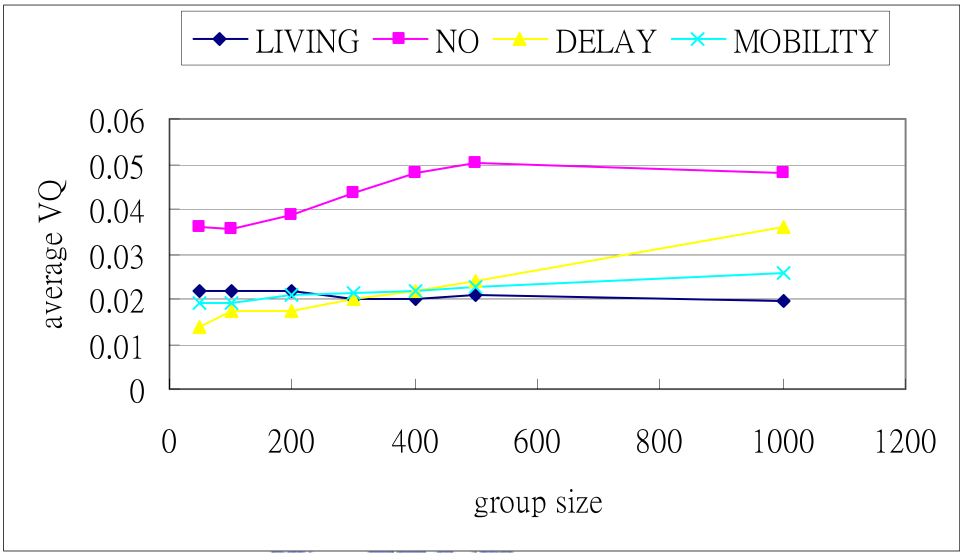


Figure 26: Average VQ with different group size

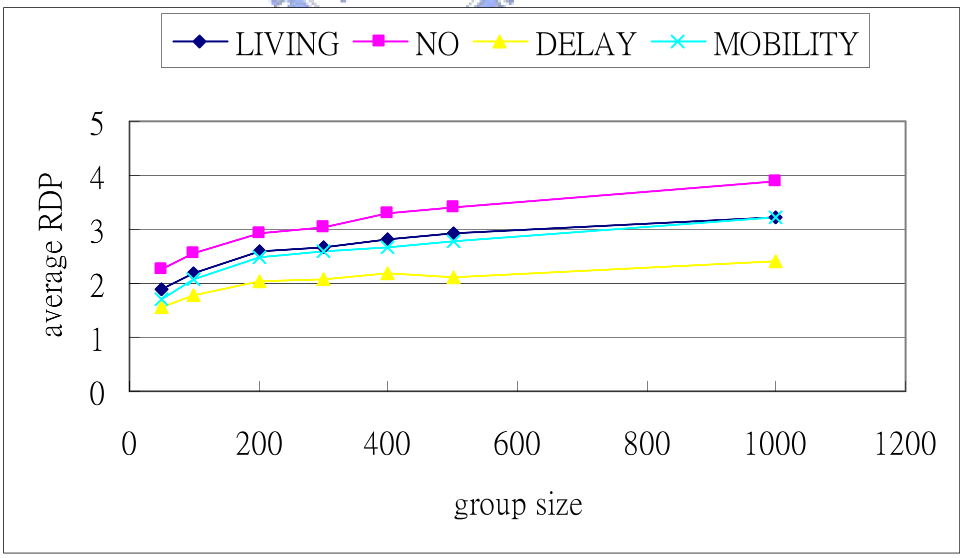


Figure 27: Average RDP with different group size

4.2.3. Proportion of Mobile Nodes

This section investigates the effects on performance with different proportion of mobile nodes. In Figure 28, average VQ of all the four algorithms grows while more proportion of mobile nodes uses the service. The growing rate of LIVING is similar to the rates of delay and mobility optimizations, while the rate of no optimization is pretty higher. On the other hand, Figure 29 shows that average RDP is not affected by proportion of mobile nodes. We conclude that LIVING can fit the situations with different proportion of mobile nodes.

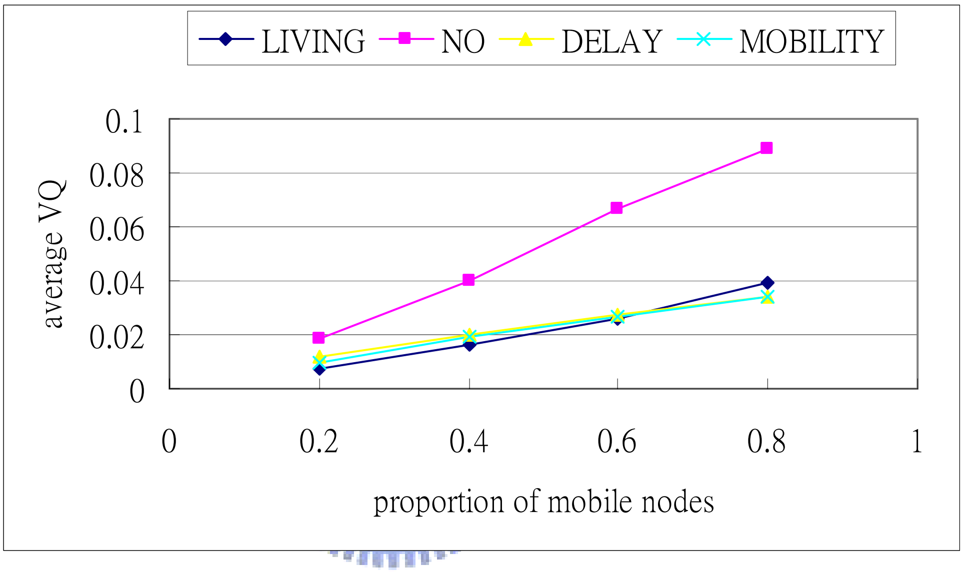


Figure 28: Average VQ with different proportion of mobile nodes

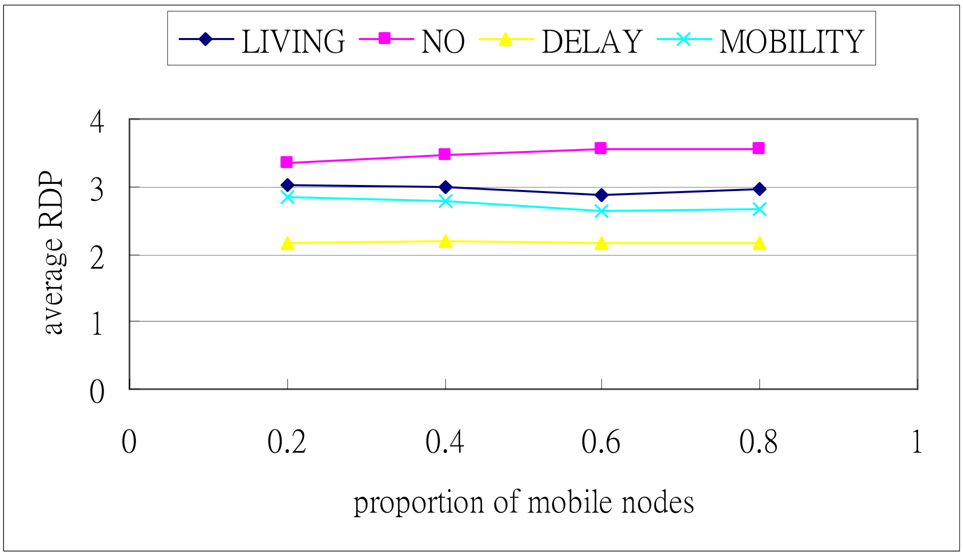


Figure 29: Average RDP with different proportion of mobile nodes

4.2.4. Number of Out-degree

In this simulation, we vary the number of nodes' out-degree from 4 to 16. Figure 30 and Figure 31 show that both average VQ and average RDP are improved with larger number of out-degree. We conclude that the number of out-degree affects the performance.

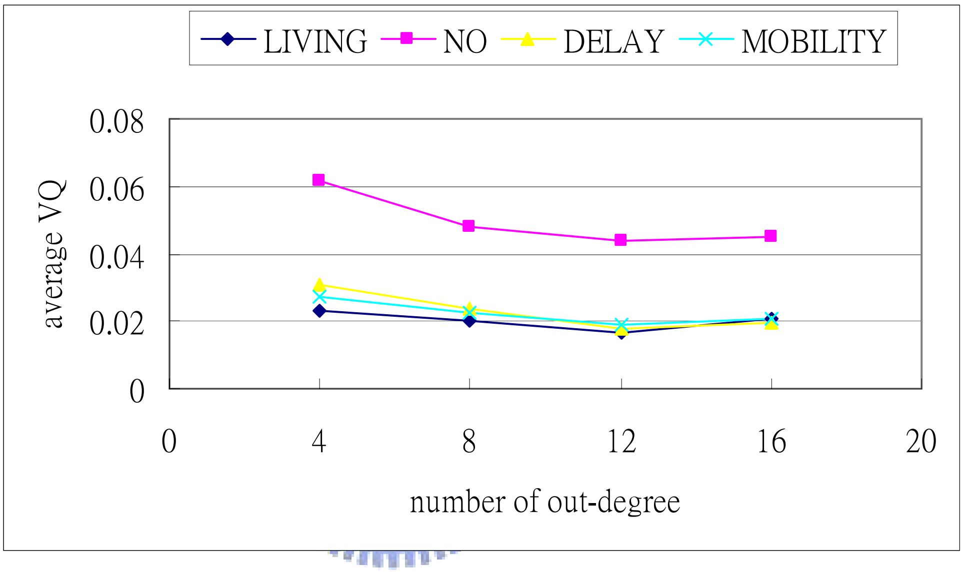


Figure 30: Average VQ with different number of out-degree

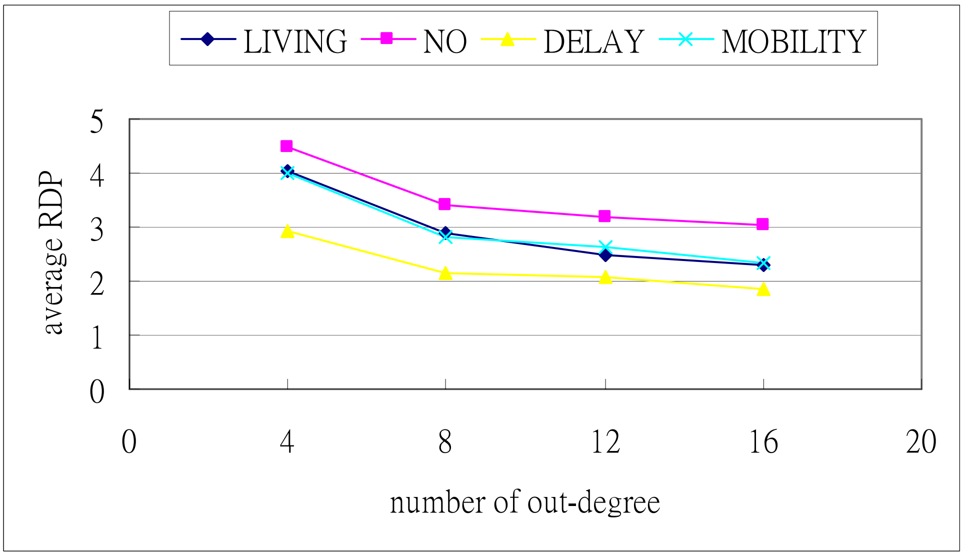


Figure 31: Average RDP with different number of out-degree

4.2.5. Optimization Period

Since the optimization takes place periodically, the performance using different optimization periods should be evaluated so that we can decide an appropriate period in our system. Figure 32 and Figure 33 show the evolution of average VQ and average RDP using different optimization periods. We can see that the convergence speeds of performance using periods between 50s and 150s are similarly fast. However, Figure 34 shows that the optimization control overhead increases when the period is short. Consequently, it is suitable to choose larger periods between 50s and 150s.

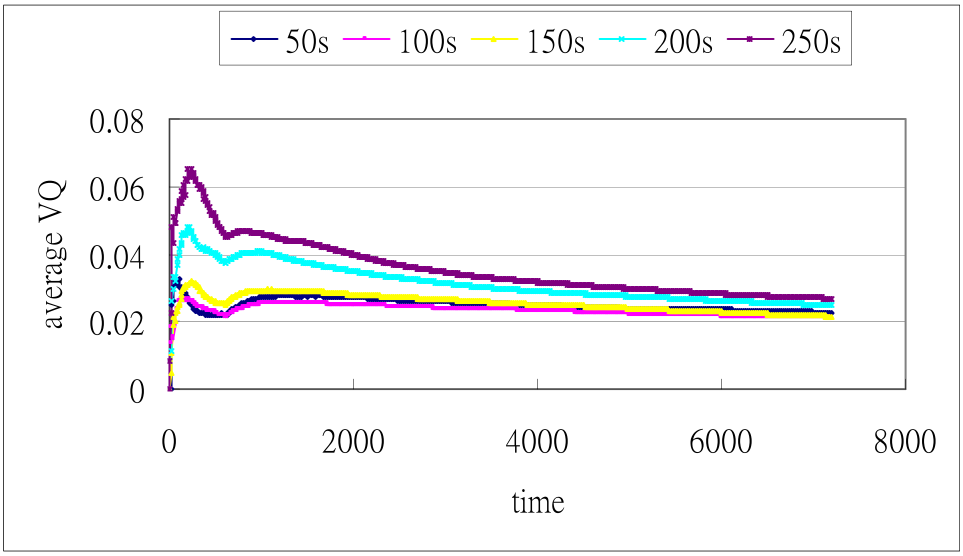


Figure 32: Average VQ using different optimization periods

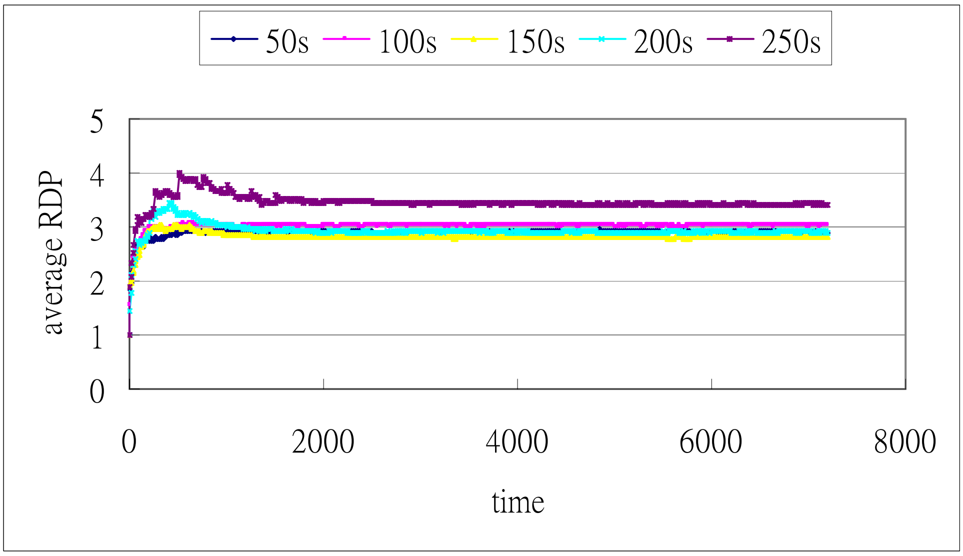


Figure 33: Average RDP using different optimization periods

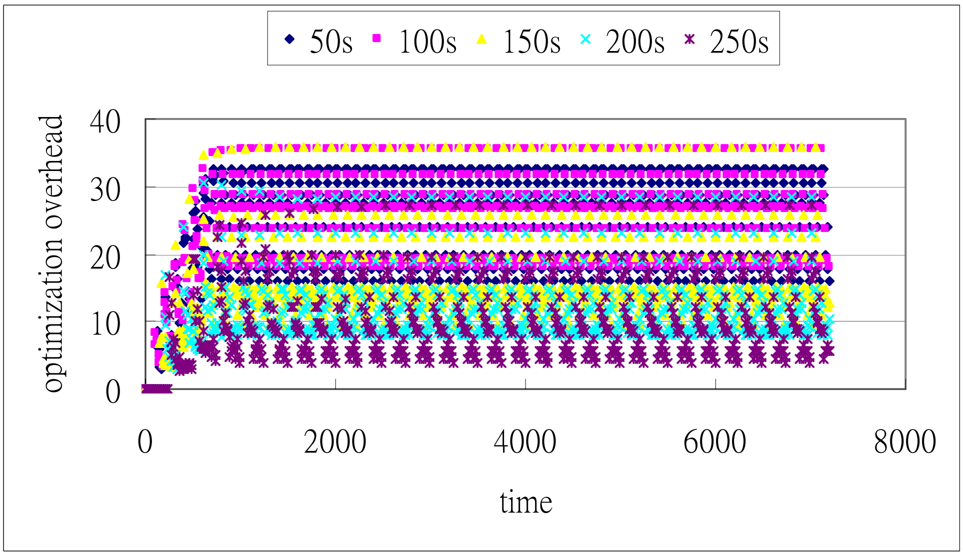


Figure 34: Optimization overhead using different optimization periods

### 4.2.6. Promotion Hop Number

In our system design, promotion hop number is proposed for limiting overhead. We evaluate the performance using different promotion hop number so that an adequate value can be found and applied in our system. Figure 35 and Figure 36 show the evolution of average VQ and average RDP using different promotion hop numbers. We can see that the convergence speeds of performance using hop numbers between

1-hop and 5-hop are similar. However, Figure 37 shows that the optimization control overhead tremendously increases while the hop number is large. Consequently, it is suitable to choose smaller hop number between 1-hop and 5-hop.

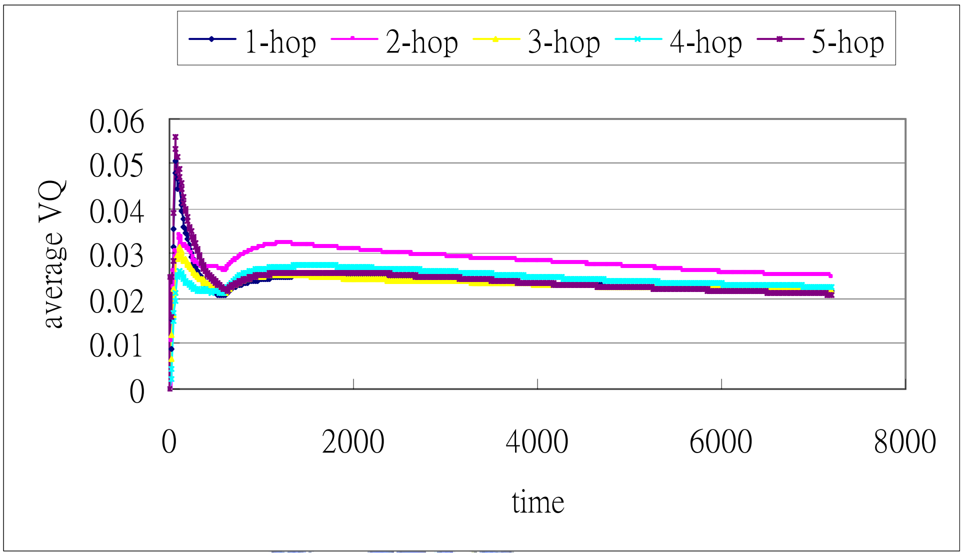


Figure 35: Average VQ using different promotion hop number

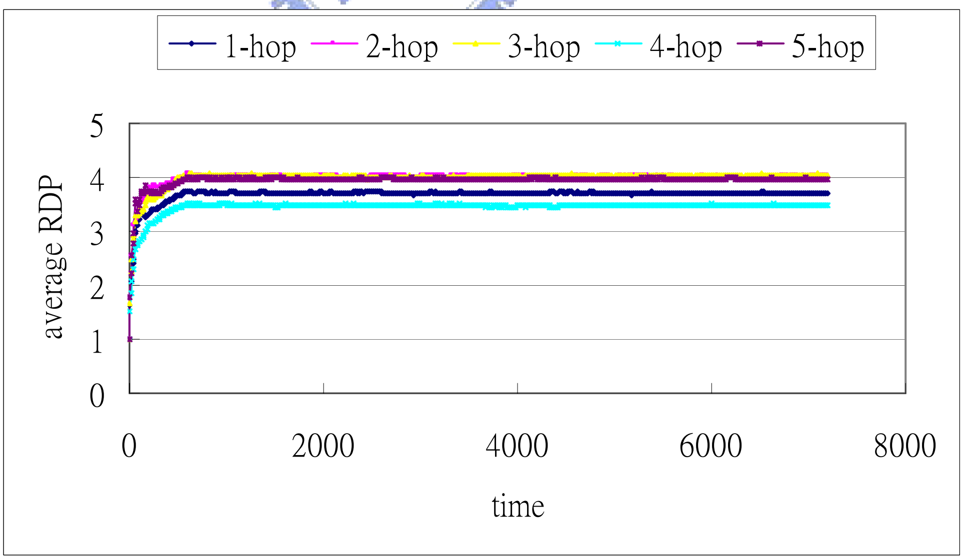


Figure 36: Average RDP using different promotion hop number

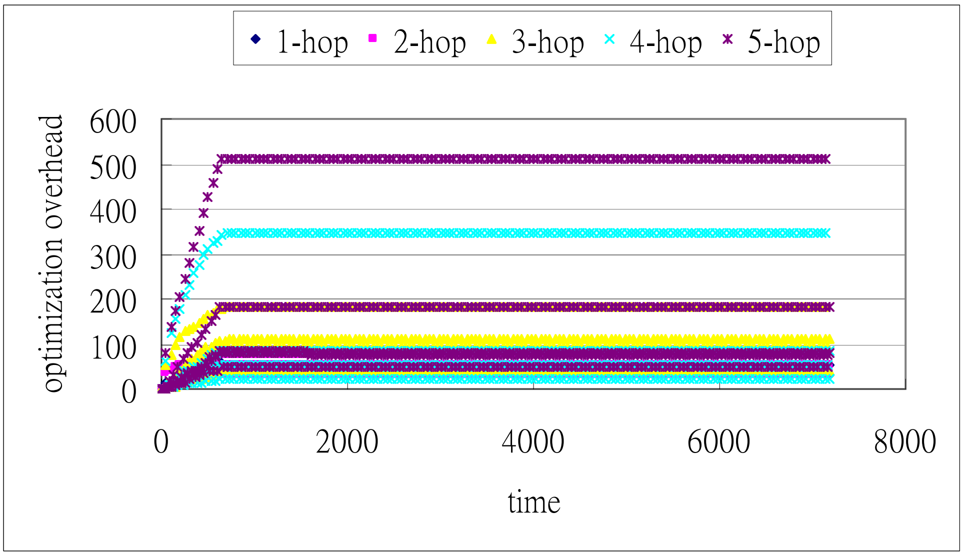


Figure 37: Optimization overhead using different promotion hop number

### 4.2.7. Join Parameter N

In the last simulation, we vary join parameter N to observe the impact to the performance. Figure 38 and Figure 39 depict that start-up delay and join overhead are positive relative to join parameter N. However, in Figure 40, average RDP slightly decreases while larger join parameter N is applied. The tradeoff between start-up delay, join overhead and average RDP should be taken into consideration in the system.

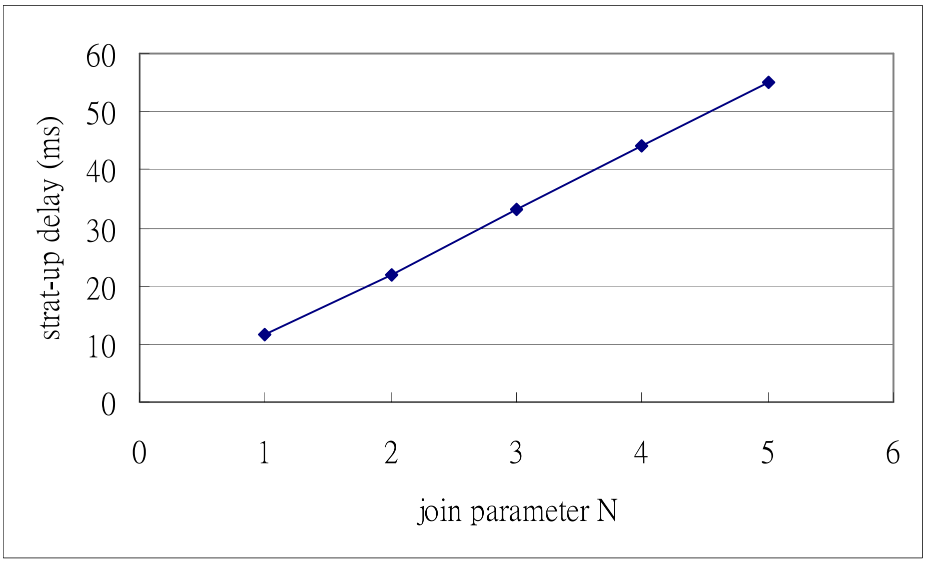


Figure 38: Start-up delay with different join parameter N

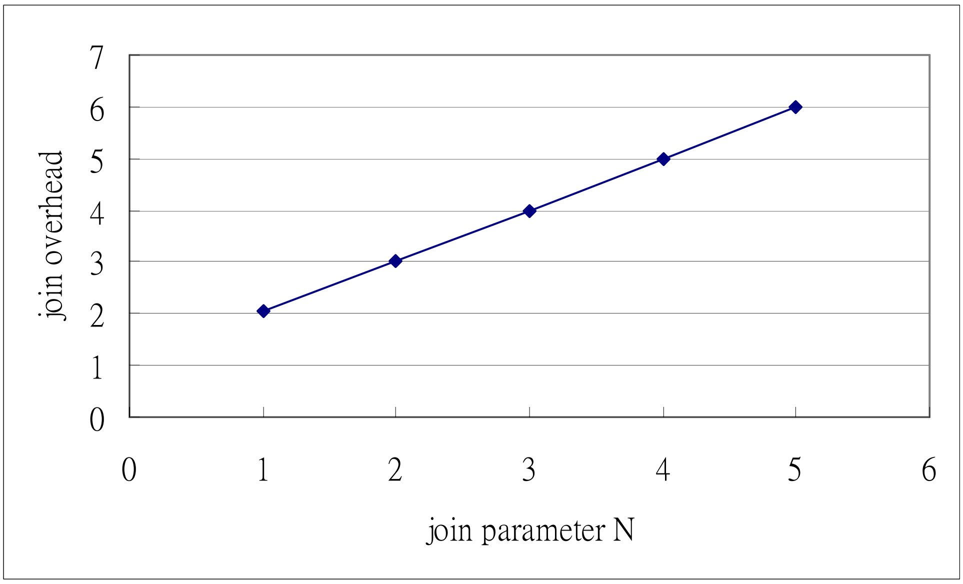


Figure 39: Join overhead with different join parameter N

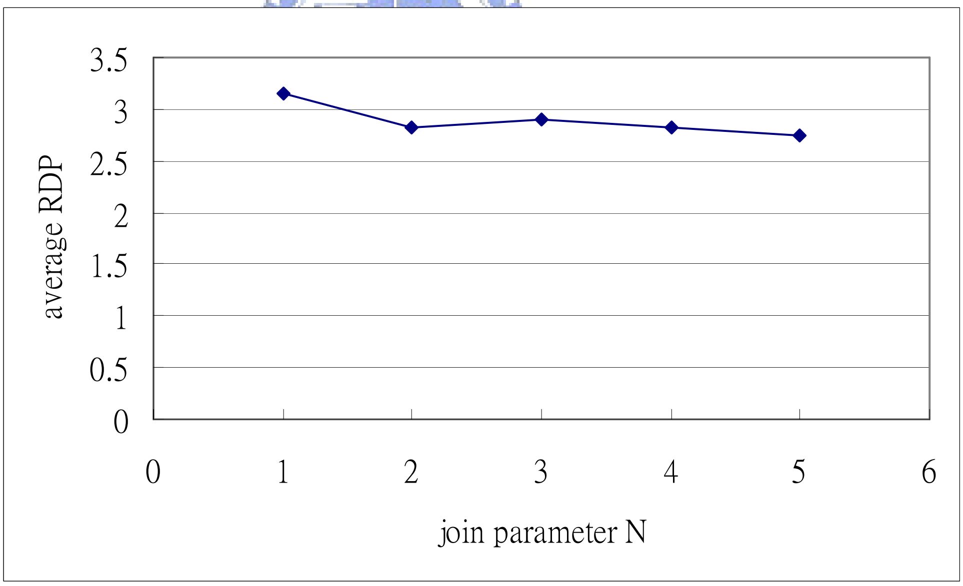


Figure 40: Average RDP with different join parameter N

## 4.3. Conclusions

In this chapter, we simulate our LIVING design using different parameters. And we summarize our main conclusions as follows. First, comparing to other algorithms, LIVING is scalable in terms of group size. Second, the proportion of mobile nodes affects VQ, but not RDP. More proportion of mobile nodes leads to more VQ. Moreover, node out-degree has impact on both VQ and RDP. Larger out-degree results in less VQ and RDP. Furthermore, while the performance convergence speed performs similar using different optimization periods between 50s and 150s, the optimization overhead grows high using shorter period. Hence, it is suitable to choose larger optimization periods between 50s and 150s. Besides, system with larger promotion hop number generates more overhead, while the convergence speed is not apparently improved. Consequently, it is suitable to choose smaller promotion hop number between 1-hop and 5-hop. Finally, there exists tradeoff between start-up delay, join overhead and average RDP while adjusting join parameter N.

# Chapter 5. Implementation

To verify our idea, we implemented LIVING using Java platform. The system performs well on Windows XP, RedHat Linux and Mac OS X with 5 fps 320 x 240 pixels resolution JPEG images, and uncompressed audio streaming. The overall data flow rate is about 50-120 kbps per overlay link. Figure 41 shows our system architecture and a picture with 6 mobile peers. Start-up delay and peer lags of all the peers are less than 1s.

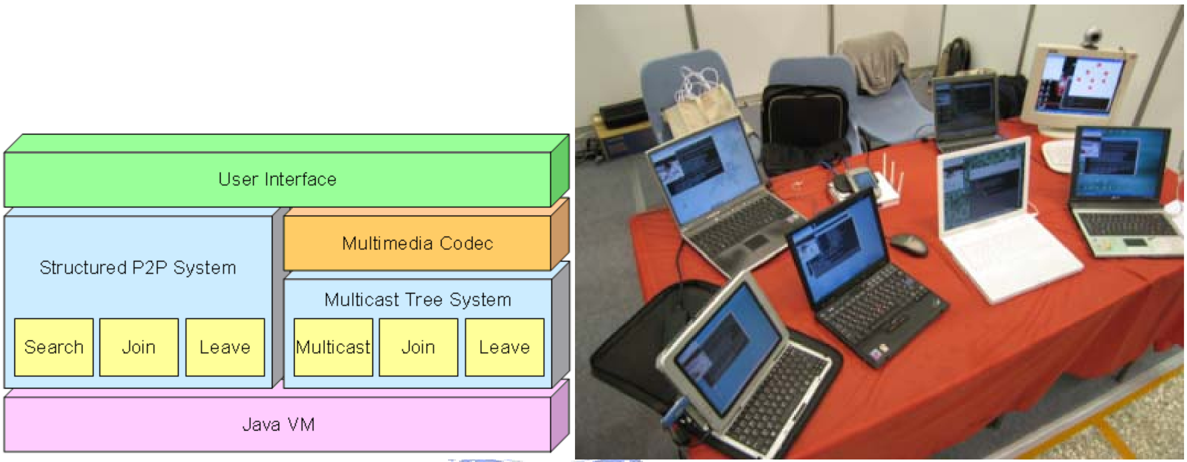


Figure 41: Implementation using Java platform

Furthermore, we also successfully implemented LIVING on ITRI (Industrial Technology Research Institute) PCA (Personal Communication Agent) system, which uses SIP (Session Initiation Protocol) [36] as control signals, H.263 and MPEG-4 as video codec, and QCIF (176 x 144) as video resolution. The overall data flow rate is about 250 kbps per overlay link. Figure 42 shows the demonstration of one live streaming source (the notebook) and 4 receivers (the PDAs) in WLAN mobile environment. The peer lags of all the peers are less than 1s.

Figure 42: Implementation on ITRI PCA

# Chapter 6. Summary and Future Work

In this thesis, a new live video broadcasting service, called LIVING, is introduced. And we find that ALM is a good solution to realize this service. However, because LIVING runs in mobile environment, there exist some mobility problems which make ALM systems unstable. To minimize the impacts of mobility, we analyze and describe the problem in mathematics form. Then, a system design is proposed based on issues users may concern. The simulations show that our design is scalable and suitable for different proportion of mobile nodes. What's more, some parameters are evaluated and the results help us decide appropriate parameter values in the system. Finally, we implement the system on Java platform and ITRI PCA respectively. It performs well and we conclude that this service is interesting and practicable.

Although LIVING performs well, there are interesting topics left for future work. First, we have to consider the lack of CPU computation power, memory or power supply on low level mobile devices. It is important to improve the performance so that LIVING can work well on these devices. What's more, the combination of LIVING and IM (Instant Messaging) is an attractive application. When we login IM, real-time live videos can be seen directly from friends list. That will make LIVING more convenient and easier to use. Additionally, there are other exciting applications we can or cannot imagine, such as personal real-time live TV stations, real-time live 3D baseball games, etc. We believe that the age of LIVING is coming, and LIVING is living in the living world!

# 參 考 文 獻


[1] A. Ganjam, H. Zhang, “Internet Multicast Video Delivery,” the IEEE, 2005.

[2] X. Hei, C. Liang, J. Liang, Y. Liu, K. W. Ross, “A Measurement Study of a Large-Scale P2P IPTV System,” IPTV Workshop, International World Wide Web Conference, 2006.

[3] Napster, <http://www.napster.com/>.

[4] Gnutella.com, <http://www.gnutella.com/>.

[5] I. Stoica, R. Morris, D. Karger, M. F. Kaashoek, and H. Balakrishnan, “Chord: A scalable peer-to-peer lookup service for Internet applications,” ACM SIGCOMM San Diego, CA, Aug. 2001.

[6] S. Ratnasamy, P. Francis, M. Handley, R. Karp, and S. Shenker, “A scalable content-addressable network,” ACM SIGCOMM, San Diego, CA, Aug. 2001, pp. 161–172.

[7] S. Deering, "Multicast routing in internetworks and extended LANs," ACM SIGCOMM, Aug. 1988.

[8] Internet Group Management Protocol, RFC 1112, RFC 2236.

[9] Distance Vector Multicast Routing Protocol, RFC 1075.

[10] Protocol Independent Multicast, RFC 2362, RFC 3973.

[11] P. Francis, “Yoid: Extending the Internet multicast architecture,” <http://www.icir.org/yoid/>, 2000.

[12] Y. Chu, S. G. Rao, H. Zhang, “A case for end system multicast,” ACM SIGMETRICS, 2000.

[13] J. Jannotti, D. Gifford, K. L. Johnson, M. F. Kaashoek, and J. W. O'Toole, Jr., “Overcast: Reliable multicasting with an overlay network,” OSDI, 2000.

[14] L. Mathy, R. Canonico, D. Hutchison, “An overlay tree building control protocol,” Intl. workshop on Networked Group Communication, 2001.

[15] S. Ratnasamy, M. Handley, R. Karp, S. Shenker, “Application-level multicast using content-addressable networks,” NGC, 2001.

[16] D. A. Helder, S. Jamin, “End-host multicast communication using switch-trees protocols,” GP2PC, 2002.

[17] B. Zhang, S. Jamin, L. Zhang, “Host multicast: A framework for delivering multicast to end users,” IEEE INFOCOM, 2002.

[18] S. Banerjee, Bobby Bhattacharjee, C. Kommareddy, “Scalable application layer

multicast," ACM SIGCOMM, Pittsburgh, PA, 2002.

[19] D. Tran, K. Hua, T. Do, "ZIGZAG: An Efficient Peer-to-Peer Scheme for Media Streaming," tech. rep. Orlondo: Univ. Central FL, CS-UCF 2002, 2002.

[20] V. N. Padmanabhan, H. J. Wang, P. A. Chou. "Resilient Peer-to-Peer Streaming," Technical Report MSR-TR-2003-11, Microsoft Research, Redmond, WA, 2003.

[21] S. Banerjee, B. Bhattacharjee, "A comparative study of application layer multicast protocols," unpublished report.

[22] A. El-Sayed, V. Roca, L. Mathy, "A survey of protocols for an alternative group communication service," IEEE Network, 2003.

[23] C. Abad, W. Yurcik, R. H. Campbell, "A survey and comparison of end-system overlay multicast solutions suitable for network centric warfare," SPIE, 2004.

[24] S. W. Tan, G. Waters, J. Crawford, "A performance comparison of self-organising application layer multicast overlay construction techniques," Computer Communications, 2006.

[25] J. Konemann, "Approximation Algorithms for Minimum-cost Low-degree Subgraphs," PhD Thesis (Computer Science). CMU, 2003.

[26] S. Y. Shi, J. S. Turner, M. Waldvogel, "Dimensioning server access bandwidth and multicast routing in overlay networks," Proc. NOSSDAV, 2001.

[27] N. M. Malouch, Z. Liu, D. Rubenstein, S. Sahu, "A graph theoretical approach to bounding delay in proxy-assisted, end-system multicast," Proc. IWQoS, 2002.

[28] K. Albrecht, R. Arnold, M. Gahwiler, R. Wattenhofer, "Aggregating Information in Peer-to-Peer Systems for Improved Join and Leave," Proc. of the Fourth International Conference on Peer-to-Peer Computing (P2P'04), 2004.

[29] National Center for High-Performance Computing, Taiwan Advanced Research & Education Network, Network Operation Center <http://noc.twaren.net/>.

[30] The PC Guide <http://www.pcguide.com/ref/ram/timingAccess-c.html>.

[31] C. Faloutsos, P. Faloutsos, M. Faloutsos, "On power-law relationships of the internet topology," ACM SIGCOMM, 1999.

[32] J. Winick, C. Jin, Q. Chen, S. Jamin, Inet Topology Generator, available from: <http://topology.eecs.umich.edu/inet/>.

[33] T. Camp, J. Boleng, V. Davies "A survey of mobility models for ad hoc network research," Wireless Communications and Mobile Computing, 2002.

[34] N. Banerjee, W. Wu, S.K. Das, "Mobility Support in Wireless Internet", IEEE Wireless

Communications, 2003.

[35] S. Y. Shi, J. S. Turner, M. Waldvogel, "Dimensioning server access bandwidth and multicast routing in overlay networks," 11th International Workshop on Network and Operating Systems Support for Digital Audio and Video (NOSSDAV), 2001.

[36] IETF SIP working group, <http://www.ietf.org/html.charters/sip-charter.html>.

# 自傳

黃名杰，成功大學資訊工程系學士，現就讀交通大學資訊科學與工程研究所，主要的研究興趣為點對點同儕網路（Peer-to-Peer Networks）。