A BOTTOM-UP APPROACH TO PEER-TO-PEER LIVE MULTICAST

by

Başak Öztaş

B.S., Electrical and Electronics Engineering, Boğaziçi University, 2009

Submitted to the Institute for Graduate Studies in
Science and Engineering in partial fulfillment of
the requirements for the degree of
Master of Science

Graduate Program in Electrical and Electronics Engineering

Boğaziçi University

2011

A BOTTOM-UP APPROACH TO PEER-TO-PEER LIVE MULTICAST

APPROVED BY:

Prof. Bülent Sankur .....
(Thesis Supervisor)

Assoc. Prof. Burak Acar .....

Assoc. Prof. Oğuz Sunay .....

DATE OF APPROVAL: 05.09.2011

ACKNOWLEDGEMENTS

First and foremost, I am grateful to Assoc. Prof. Oğuz Sunay for his understanding and courteousness in the course of this thesis. He is a brilliant and comforting academician, and this thesis would not be complete without his help.

I would like to thank to Prof. Bülent Sankur for sharing his wisdom and knowledge during our discussions. I feel fortunate to get to know such a great mind.

I deeply thank to Prof. Fatih Alagöz for all the guidance and friendship he has offered me. I truly wish to have the privilege to work under his guidance in the near future.

I also thank to Şükrü Kuran for sharing his experience, saving me from months of misery.

I should not forget to thank to both of my families for their sharing my troubles and cheering me up. I hope I deserve to claim their name.

Beyond all, special thanks go to my significant other, Ahmet Burak Yoldemir, for he always was and will be with me.

ABSTRACT

A BOTTOM-UP APPROACH TO PEER-TO-PEER LIVE MULTICAST

The implementation difficulty of IP multicast in terms of both structural and economical reasons, headed the multicast functionality towards higher layers. Peer-to-peer multicasting have aroused as a potential solution for these services and it has established a presence in the market rapidly. However, it came with several challenging design and deployment issues, and unresolved problems. One of the key concerns of P2P protocol designing is the overlay construction phase. However, neither tree based systems nor mesh based systems are robust enough to provide a resilient and efficient service. In this study, we propose a new overlay topology for P2P multicasting to eliminate the problems caused by the best effort nature of the Internet infrastructure and unpredictable behavior of peers. In this approach, we are handling the tree based architecture of P2P live video streaming with a bottom-up perspective. By allowing reverse transmission of video packets, i.e., packet transmission from descendants through ascendants, the disruptions caused by the packet losses and peer churn are effectively reduced. By implementing the proposed method, we consistently achieved better video quality than a well known tree based P2P protocol (SPPM), notably in the cases of high packet loss ratio and peer churn.

ÖZET

EŞLER ARASI CANLI ÇOKLU GÖNDERİME AŞAĞIDAN YUKARI YAKLAŞIM

IP çoklu gönderimin yapısal ve ekonomik nedenlerden kaynaklanan uygulanma zorlu-ğu, çoklu gönderim işlevlerinin üst katmanlara taşınmasına yol açmıştır. Eşler arası çoklu gönderim bu servislere bir potansiyel çözüm olarak ortaya çıkmış ve pazar içerisindeki yerini hızla almıştır. Fakat, birçok dizayn ve yayılma mevzularını ve çözülmemiş problemleri de beraberinde getirmiştir. Eşler arası protokol dizaynının en önemli meselelerinden birisi, katman kurumu aşamasıdır. Fakat, ne ağaç temelli sistemler ne de veri bazlı sistemler yeteri kadar gürbüz ve verimli servis sağlayacak kadar dayanıklı değildir. Bu çalışmada, internet altyapısının en iyi çaba ilkesinin yarattığı problemleri ve eşlerin tahmin edilemez davranışlarının sonuçlarını ortadan kaldırmak için, yeni bir eşler arası çoklu gönderim katman topolojisi önerilmiştir. Bu yaklaşımda, eşler arası canlı video aktarımının ağaç temelli mimarisi, aşağıdan yukarıya doğru bir bakış açısıyla ele alınmaktadır. Paketlerin tersine aktarımına izin vererek, paket kayıpları ve kullanıcıların giriş çıkışlarından kaynaklanan bozulmalar etkili bir şekilde azaltılmıştır. Önerilen metodu uygulayarak, özellikle yüksek paket kaybı ve eş dalgalanması durumlarında, bilinen bir ağaç temelli eşler arası aktarım protokolüne kıyasla, düzenli bir şekilde daha iyi video kalitesi sağlandığı gözlenmiştir.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

LIST OF ACRONYMS/ABBREVIATIONS

ADSL	Asymmetric Digital Subscriber Line
CC	Check Children
CDF	Cumulative Distribution Function
CDN	Content Distribution Network
CP	Check Parent
DVMPR	Distance Vector Multicast Routing Protocol
Gbps	Gigabits per Second
IP	Internet Protocol
IPTV	Internet Protocol Television
JPS	Joint Parent Selection
Kbps	Kilobits per Second
Mbone	Multicast Backbone
Mbps	Megabits per Second
MDC	Multiple Description Coding
MOSPF	Multicast Open Shortest Path First
ms	Milliseconds
MTU	Maximum Transmission Unit
P2P	Peer-to-Peer
PSNR	Peak Signal to Noise Ratio
QoS	Quality of Service
QP	Quantization Parameter
RMSE	Root Mean Squared Error
RMTP	Reliable Multicast Transport Protocol
RPM	Reverse Path Multicasting
RTT	Round Trip Time
SPPM	Stanford Peer-to-Peer Multicast
SPS	Single Parent Selection
SRM	Scalable Reliable Multicast
Tbps	Terabits per Second

TV	Television
UDP	User Datagram Protocol
VoD	Video on Demand

1. INTRODUCTION

1.1. Multimedia Over the Internet

Since there is an explosive growth and tremendous success in the Internet technology, more and more people became users of the Internet and it spread all over the world. This success is enlarged by the increasing variety of applications provided to the users, causing a vicious circle of penetration.

The increased demand to the Web paved the way towards a better Internet infrastructure. The rapid increase of bandwidth, computing power and reduced prices of both hardware and software technologies, created a huge flow of multimedia content to the Internet. There are countless number of emerging applications that require the transmission of multimedia content over the Internet such as interactive TV, online games, VoD, IPTV, digital video libraries, distance learning, document imaging and broadcast of sports events, to name a few. Apparently, this phenomenon, the growing usage of demanding applications, was not expected prior to the deployment of Internet protocol and service model. The best effort nature of Internet protocol has been the main bottleneck for demanding applications requiring QoS guarantees such as multimedia systems. Together with multimedia systems, other applications including distributed simulation and distributed computing, have been the main motives behind the path breaking research conducted by academia and industry throughout the last 20 years.

Among several types of multimedia applications, the most bandwidth consuming ones are related to video transmission. Recent studies estimate the share of video traffic over the Internet and the video traffic is expected to be the main source of the Internet traffic [1–3]. YouTube has a data volume exceeding 45 terabytes and it has reached over 3 billion views per day [4]. Due to this massive cost of video applications, much work has also been done on video coding and packet video delivery. These novelties have drawn excessive attention from users, industry and academia. With the contribution of both evolving technology and academic studies, streaming video over the Internet has been transforming from being a novelty to a commonly used technology, a reality.

Although other video transmission techniques, e.g., VoD and video file sharing, penetrated the market to a large extent, live video broadcasting is still a challenging issue. It has additional stringent constraints those are not required by VoD and file sharing applications. It is obvious that, TV-like viewing experience is expected from video broadcasting over the Internet eventually. To achieve this goal, quite a few approaches are proposed in the literature covering a wide spectrum of objectives, architectures, and infrastructures, because the traditional client-server technique is beyond the reach of sufficiency for providing such a technology.

Peer-to-peer multicasting have aroused as a potential alternative to these systems and it has established a presence in the market rapidly, like in the case of standard P2P file sharing systems. However, it came with several challenging design and deployment issues, and unresolved problems such as stability and resistance.

1.2. Contributions of the Thesis

In this study, we propose a new overlay topology for P2P multicasting to eliminate the problems caused by the best effort nature of the Internet infrastructure and unpredictable behavior of peers. In this approach, we are handling the tree based architecture of P2P live video streaming with a bottom-up perspective. We integrate the redundancy in mesh systems into the assuring nature of tree based architectures. By allowing reverse transmission of video packets, i.e., packet transmission from descendants through ascendants, the disruptions caused by the packet losses and peer churn are effectively reduced. This method exploits the resistance of mesh based architectures to peer churns and the highly structured nature of tree based systems. The proposed method is generic, and it is applicable to any tree based topology. The ultimate goal is to provide the highest possible video quality in a peer-to-peer live video streaming system. As the base application layer protocol, we have implemented Stanford Peer-to-Peer Multicast protocol which is specifically developed for low latency, self scaling and robust transmission of video in peer-to-peer networks [5]. The performance of the proposed method notably outperforms the original system.

1.3. Thesis Outline

This thesis is organized as follows. In Chapter 2, the background information on the two main building blocks of this thesis, video multicasting over the Internet and P2P networks, is provided. Chapter 3 explains the SPPM protocol structure by illustrating it with flow diagrams, and presents the proposed method. Chapter 4 describes the simulation setup and evaluates the proposed bottom-up approach applied to the SPPM protocol. In Chapter 5, conclusions of the thesis are drawn and future research directions are proposed.

2. BACKGROUND

2.1. Video Streaming over the Internet

There are several video streaming technologies differing in their capacity of concurrent viewers and the number of simultaneous sessions they can provide[1] . In [6], authors categorize these technologies as shown in Figure 2.1. For a vast number of users, TV broadcasting (analog/cable) has been used for decades, which can only serve a limited number of channels in a specified region. Although its endless scalability in terms of the number of concurrent receivers it can support, this technology has proved to be outdated as the surveys demonstrate [7]. Also, it may not be feasible to offer service to wide areas in the case of sparsely populated networks because this technology depends on the area coverage and deploying a transmission system on an area for a small number of users is a costly operation.

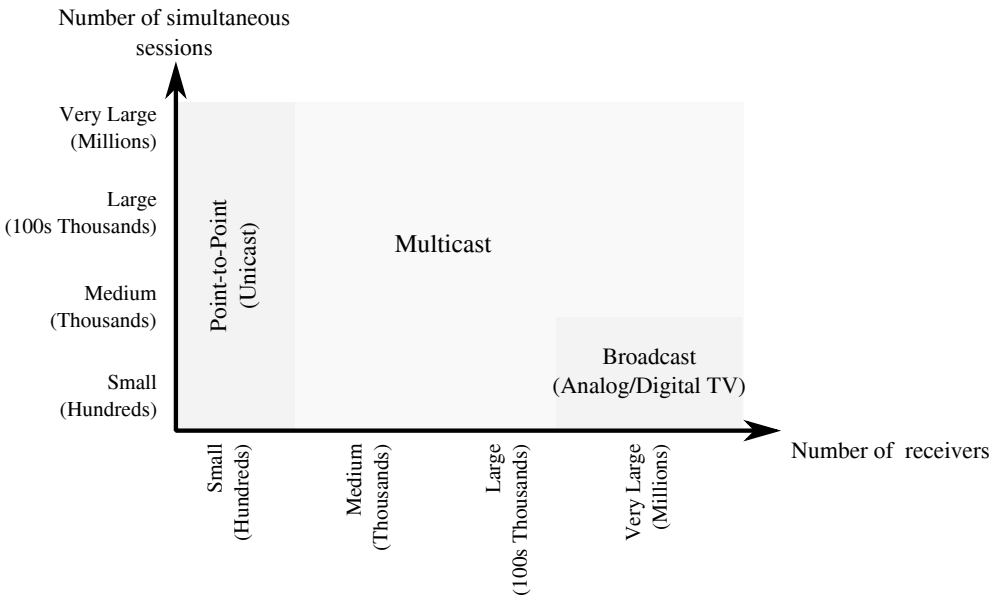


Figure 2.1. Video delivery technologies according to their scales.

Unlike TV broadcasting, point-to-point video delivery over the Internet (standard unicast system) is used to support a large number of channels with a service region covering a large area, even including the whole Internet network. In these systems, a

[1]Please note that video streaming does not imply video file transfer which is unrelated to the video streaming properties addressed in this thesis.

user may select a channel or a single video among thousands of others. Mostly, this technology is used for VoD and interactive video. The server creates and sends a separate copy of each video packet to every client with an overlay approach. However, the basic problem of these systems is the scalability issue. They inherently suffer from the lack of scalability since each user adds an additional load on the system's infrastructure. For example, a server with a minimum bandwidth of 300 Mbps is needed to serve an 300 Kbps video stream to 1000 clients, which is a moderate number for the current circumstances. In Figure 2.3, the network and logical topologies for unicast systems are illustrated.

Neither TV broadcasting, nor point-to-point delivery would be sufficient, and a compromise should be reached. The market needs a system which can support both a significant number of parallel sessions, which gives the flexibility to the user to view whatever she wants, and an unsteady number of concurrent viewers who may be sparsely distributed around the world, or very dense in a specific region. This is where video multicasting comes into play. In video multicasting, the receiver only gets the video she desires unlike TV broadcasting. In these systems, the users who desire to view the same channel are brought together and create a multicast group where they may interact directly with each other or the server. For video broadcasting over the Internet, the term "multicast" will be a better taxonomy since it is technically not broadcasting, i.e., only the demanding users receive a session[2] . These systems are mostly used in live and VoD multicasting.

Obviously, Internet video delivery has been a substitute for the standard TV broadcasting because of its flexibility and adaptability to the trends, the changing habits of users about video consumption. The community has met with the chance of an endless number of channels where they are able to choose the content to be delivered. This growing community have started to share video content, even their own productions by means of the increased usage of digital cameras and webcams. The users have gained the chance of distributing their own media to the community they intended.

[2]Please note that throughout this thesis, the term multicasting will be used interchangeably with video broadcasting for the Internet context.

Unlike the file transfer and web services, video streaming is a challenging technology since it is much more vulnerable to network deficiencies. Despite the improvements over the best effort nature of the Internet technology, additional efforts are required to provide high quality video streaming to users. The performance of this structure depends on many conditions such as the route taken, processing speed and capacity of the nodes and the utilization rates of the network. There are two basic issues regarding the service degradation and user frustration in these systems. Firstly, video transmission is bandwidth sensitive unlike the file transfer, regardless of the size of the content. It is much more vulnerable to packet losses and network congestions since overload and delay stemming from retransmissions are unsuitable for media streaming. From the compression side, there is an increasing standardization of coding techniques which provides various advantages such as lower bandwidths and better error resilience. Although there are various video coding techniques to eliminate the packet losses for highly congested networks, e.g., multiple description coding, it still stands as a hot research topic. The other problem is the end-to-end transfer latency which is extremely unpredictable in IP networks. The transmission time of sequential packets are important for VoD systems, determining the current status of a received packet, i.e., a received packet cannot be used until all of the prior packets become available regardless of the waiting period. Although the transfer latency is not crucial in terms of usability of a packet in other applications, a late packet is worthless in live media streaming applications.

2.2. Live Video Multicasting

As we discussed in the previous section, video streaming over the Internet implies both live and VoD streaming, which have common as well as distinct properties. In live video streaming, all connected users are intended to receive the live video content in a synchronous manner. Contrarily, the VoD systems aim to disseminate the content to the users whenever they desire, regardless of their synchronization. We will be focusing on live systems for the rest of this thesis.

To provide a taxonomy, the multicast functionalities are divided into two main groups below, similar to the classification in [6].

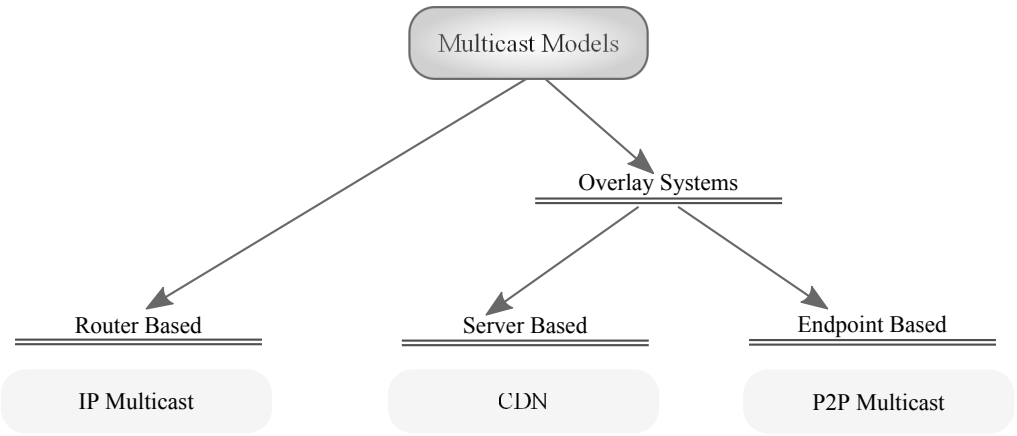


Figure 2.2. Classification of video delivery architectures for video multicasting over the Internet by the responsible layer and infrastructure requirements.

Membership management: An entity in the multicast network should be responsible for keeping track of a membership list, and be able to maintain it up-to-date. The list should be as exact as possible. There is a trade off between the correctness of the list and the overload in the system caused by protocol packets during flash crowds.

Video delivery path maintenance: The dissemination graph should be spanning all nodes during the multicast session. In other words, there should be at least one path from the server to each requesting receiver. Depending on the structure of the network, each node should be informed properly about their packet replication and forwarding tasks.

2.2.1. Router Based Systems

Through 1990's, academia and industry tried to determine the layer at which the multicast should be implemented, and different ideas were suggested. Figure 2.2 shows the structure of this classification. While some people supported lower layer architectures such as router based, others encouraged end-to-end systems. Implementation of these systems at lower layers has considerable advantages in terms of scalability and traffic load, with increased complexity in return. IP multicast, a delivery architecture proposed earlier, relies on router based data dissemination. In IP networks, *router based* has the same meaning with network layer because the routing task is handled in this layer. In [8, 9], authors proposed the IP multicast for the first time as a solution to

large scale video multicast over the Internet, which attracted considerable attention from the research community. In the suggested approach, the routers are responsible for both packet forwarding and replication, acting as the interior nodes of the graph. Every video packet is disseminated from the server to each client through the multicast routers. The end users stand as the leaves of the system, being released from the dissemination responsibility. As a matter of fact, a packet passes over each link only one time. Figure 2.3 illustrates the packet paths for IP multicast in comparison with unicast (point-to-point) method. In such a simple scenario, the total number of traversed links for the unicast example is 22, however it is only 11 for the IP multicast case. This method becomes more and more efficient as the users are more densely populated in the network layer.

Each multicast group has a unique group address which should be known by both the server and the receivers of the session. The advantage of this property is that, this address creates a rendezvous point between two end points of the application, i.e., an unlimited number of receivers can join the session without explicitly informing the server. This feature increases the scalability of the system in terms of peer churn load on the server. Various protocols are developed for routing and control mechanisms such as RPM, DVMRP [8], MOSPF [10], RMTP [11], SRM [12]. MBone [13] was deployed in 1992 as the earliest example (only audio-cast) of this system. In 1995, it had already been used for more than 250 conferences and events in 25 countries [14]. It is used as a basis of several applications such as *vic* [15] and NPSNET [16].

Despite the heavy demand and effort towards IP multicast systems, it could not survive due to the obstructing disadvantages. First of all, the deployment step of the model requires extreme complexity and cost. To be an effective solution, it needs a long deployment phase of IP multicast capable networks all over the world. Although it seems scalable in terms of bandwidth, the requirement of routers to maintain groups state extremely increases the complexity of the system where routers need to keep huge multicast forwarding tables and update the entries as users join or leave the sessions. Such a list is a large and unsteady table creating a costly operation. Because of these limitations, IP multicast service has almost disappeared.

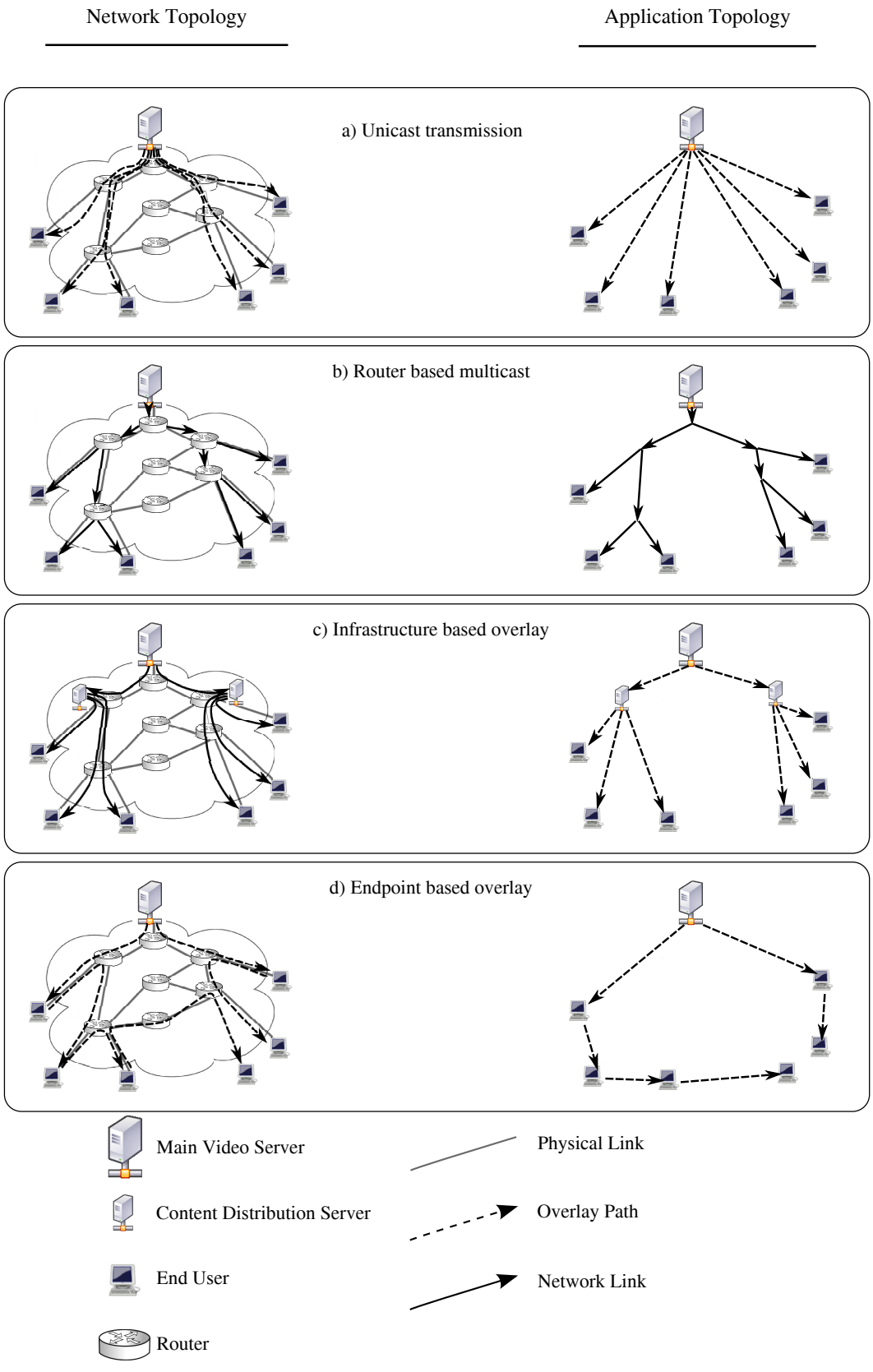


Figure 2.3. Topology illustrations for several video delivery architectures.

2.2.2. Overlay Systems

The implementation difficulty of IP multicast in terms of both structural and economical reasons, headed the multicast functionality towards higher layers as suggested in [17, 18]. The principle idea behind the overlay systems is that each delivery is performed as a unicast transmission without interfering with the underlying network infrastructure. Therefore, multicast elements consist of the end users and servers. In contrast to IP multicast, where all interior nodes are routers and the leaves of the graph are end systems, endpoints (servers and end users) act as both interior and leaf-nodes in overlay multicasting. The routers perform only the unicast functions. Similarly, while the links of the IP multicast scenario represent real network links, the links between the nodes implies the unicast path for the overlay systems. These unicast paths include all network links and routers participating the unicast transmission. The topologies are illustrated in Figure 2.3 with a comparison of unicast, IP multicast and overlay systems. A point-to-point system can be interpreted as a special case of an overlay system, with only one interior node.

In overlay systems, several multicast tasks, such as packet forwarding and group management, are carried out at the end systems. With this approach, the disadvantages of the IP multicast are tried to be addressed. Most importantly, the deployment phase of overlay systems is very fast and simple. Also, it preserves the stateless nature of the IP network. However, the decrease in the performance arises as a trade-off to the mentioned advantages. For example, a packet disseminated from a sender would probably traverse the same link multiple times, generating an excessive traffic on the physical links which cannot be avoided easily. Also, the packet dissemination is performed without concerning the underlying system, and eventually the overall packet delay of the system increases.

There is a wide spectrum of overlay applications, differing in the multicast functionality assignments on system elements. One end of this spectrum represents CDNs, the purely infrastructure based architectures. In a CDN, several content distribution servers are positioned on important locations at the initial deployment step. Strate-

gically most efficient spots to maximize the network efficiency are near to the routers serving the densely populated customers. A small scenario is depicted in Figure 2.3. Every new client requesting the video service is automatically directed to the nearest content delivery server from which it will receive the video stream during the whole session. Firstly, each video packet is sent from the main video server to the content distribution servers. Then each CDN server distributes the packets with a unicast manner to the clients connected to itself during the multicast session. CDN is an effective solution for reducing packet transmission and protocol delays, and for balancing the total multicast load on the network.

Akamai [19], which is a well known CDN company, is a successful example for this technology. The company launched its service in 1998 and it has a total number of 84000 servers in 72 countries in 2011. It is stated that Akamai constitutes between 15 and 30% of all Web traffic, reaching more than 4 Tbps [19]. Today, lots of globally known companies such as Apple, Yahoo!, MSNBC.com are using Akamai to serve live and on demand video to their customers.

Despite its success, CDN has several major disadvantages which prevent it from being a monopoly. Although it is a good solution for small number of multicast channels with large populations, unfortunately it is not a feasible solution for small businesses because its deployment and maintenance are extremely costly operations. Moreover, it is not scalable. The total data rate of the server in a CDN is directly proportional to the number of users, which may be an overpriced agreement for content providers with a fluctuating customer population.

To dramatically point out the scaling problem of video delivery, consider the following real life example. On 7 July 2009, CNN.com reported a peak of 781000 concurrent live video streams during the Michael Jackson's memorial [20]. Assuming a low bandwidth video of 400 Kbps is broadcasted, the total bandwidth required for the multicast was at least 310 Gbps. This is much larger in comparison with Akamai's network capacity, since the largest CDN has a total capacity of 200 Gbps.

Overlay multicasting also involves purely end system based P2P networks and

way point multicasting techniques. Way point multicasting [21] is a special case of P2P networks, supported with interior provisioned nodes within the overlay architecture. Therefore, we do not explain it in detail here. P2P multicasting will be explained exhaustively in Section 2.3, including P2P basics, multicasting issues and architectures.

2.3. Peer-to-Peer Video Broadcasting

In this section, first we briefly discuss P2P networks and its background. Then, we focus on live video broadcasting over P2P networks and the design issues. Lastly, we explain the architectures used in P2P multicast systems.

2.3.1. Peer-to-Peer Networks

In traditional server/client (centralized) systems, all of the users directly communicate with the server. Every task in these systems consumes resource from the server. Regardless of the required data type, server is the only responsible side for data dissemination. This single-acting topology works quite adequately for the systems where the number of users scales up to a predictable amount and the provided service does not require a high bandwidth. However, in such systems, the increasing number of users degrades the performance, even causing the server to be unresponsive. Collapsed web sites which we often come up are common examples of this problem. To address this scaling problem of server/client systems, P2P networking was developed and has been widely used since its emergence.

P2P networks are decentralized systems which unifies the network elements by assigning the serving tasks also to users unlike the traditional centralized networks. In other words, clients also cover the resource expenses of the network along with the server. In these networks, all participants donate some of their resources to be a part of the network, which may include bandwidth, storage space or processing power. The contribution of every participant increases the overall resource of the system, hence resolves the scaling problem of a traditional centralized network. Therefore, P2P networks do not need a dedicated network infrastructure.

Another advantage of P2P networks is their decentralized nature, which increases the robustness against node failures. In centralized networks, the survival of the network solely depends on the server node. Contrarily, a failed node in a P2P network affects the network only up to a certain extent, probably a few of other nodes, during the recovery process.

In recent years, P2P services have been widely used for file sharing applications. Numerous examples can be given to these applications such as Napster [22], Gnutella, e-Mule [23], BitTorrent [24] etc. Detailed studies on these applications are reported in the literature [25]. Alarming surveys have pointed out very interesting statistical information which reflects the influence of P2P networks on today's technology. Only BitTorrent traffic occupies about 35% of all Internet traffic [26] and P2P file transfer represents more than 60% of total Internet traffic [27]. Currently, more than 150 million people are using P2P systems [28].

Besides its advantages, there are also several disadvantages of P2P networks. For example, copyright issue has been a substantial problem due to uncontrolled nature of the P2P communication and sharing. This uncontrolled structure also brings along some security threats. Also, the excessive amount of bandwidth consumption is a limitation for the Internet service providers. Last but not least, the overlay topology of P2P networks does not properly match to the underlying network's topology. In [29], authors depict the mismatch and show that P2P networks severely degrades the network efficiency.

2.3.2. Challenges

Various studies have been carried out to ensure a high QoS and satisfying experience for consumers of media. To provide a high quality level, all of the related research aims to achieve low end-to-end delays, reliability, low start-up latencies, robust and scalable systems without a dedicated infrastructure. Following the great interest in P2P file sharing applications and VoIP such as BitTorrent and Skype [30], the implementation of P2P principles for live video multicasting is proposed in [17]. Since video

Table 2.1. Requirements for P2P application types.

P2P Application Type	Bandwidth	Delay	Scaling
File Sharing	×	×	✓
Video Conference	✓	✓	×
VoD	✓	×	✓
Live video streaming	✓	✓	✓

broadcasting is a highly bandwidth consuming service, realizing it with a traditional client/server network may result in enormous prices. High number of potential clients brings up the scalability problem. For these services, P2P networks are very appropriate since they can resolve the scaling problem by consuming the participants' resources. The main motive was its scalability where each additional participant is expected to enlarge the overall bandwidth resource of the system by sacrificing some of its uploading bandwidth. In such a system, the multicast functionalities including packet replication and forwarding, administration and maintenance are undertaken by end users, as well as servers. Another advantage of this system is its independence from the underlying infrastructure, making it an extremely economical and easy-to-deploy solution.

Although the principle idea is the same, live video multicasting brings challenges different from other P2P solutions because video broadcasting needs stringent QoS requirements intrinsically. Table 2.1 summarizes the requirements of various P2P applications. For example, the bandwidth is not a crucial requirement in file sharing applications because variations in the total download duration are acceptable unless it takes a very long time, which would frustrate the users. The download duration may vary from several hours to days depending on the file size. The P2P file sharing applications serve to thousands of users all over the world and locating the files distributed among these peers is a challenging issue. For example, users exchange the segments of the content to download a file, while the order of the segments is not important in BitTorrent. However, they should be informed about the available segments in each node. There are numerous solutions for this problem in the literature such as [31]. In addition to the challenges faced by file sharing such as indexing and search issues, live streaming has additional specific constraints different from those of file sharing applications.

For video conferencing applications, the upload and download bandwidth of end users are very important to acquire a certain level of quality. In such systems, the heterogeneity in the bandwidth of the users should be handled properly, as in the case of live streaming. Also, delay is a crucial problem for video conference applications due to their interactivity. However, conference calls are not affected by the scaling problem since the number of participants rarely exceeds a small number, differentiating it from live video multicasting. Narada [32] can be given as an example work for this type of applications.

VoD and live multicasting have several characteristics in common which make them sensitive to bandwidth, delays and scaling problems. In live multicasting, all users are simultaneously demanding the same segment of the video content, whereas the users may be demanding disjoint parts of the video in VoD applications. Exploring the users holding the required segment is one of the most challenging issues and there is an extensive research ongoing to solve this problem such as [33]. Although the delay can be tolerated by buffering for a certain time for VoD applications, it is still frustrating for the users and it obstructs the general acceptability of the system. These features transfer VoD applications to a different domain than live streaming where an uninterrupted streaming is compulsory.

The insufficient bandwidth and highly dynamic population of peers prevent high quality video streaming on P2P networks. For example, peers connected via asymmetric lines such as ADSL have much more downloading capacity compared to their uploading capacity. The ADSL architecture is suitable for traditional systems where clients have no responsibility of information delivery. However, P2P networks are based on the contribution of the participants, which makes asymmetric connections unsuitable.

Peer churn, i.e., the behavior pattern of the participants indicating their connections and disconnections from the system, is also an important problem for P2P networks. It can be described as the dynamism in the network environment which arises from the random participations and separations of the peers. Whether a peer leaves the system gracefully or not, every additional peer action reduces the overall perfor-

mance of the network, because of the connection establishment durations, unanswered requests, topology changes etc. Other peers depending on the disconnected peers suffer from short term or long term connection problems. Even the overlay structure has to be renewed in some cases.

Another problem is the increasing delay at the end nodes. Since P2P networks are decentralized systems where peers are connected to other peers for receiving data, most of the peers are several hops away from the main server. At the nodes away from the source, the latency caused by the increasing number of hops becomes a serious problem, preventing a high quality video experience.

These challenges of live video streaming requires specific solutions. The challenges of video broadcasting on P2P networks can be eased by strategic approaches. Although there are many approaches in the literature including video coding techniques leveraging the performance of these systems such as MDC, or network coding, our focus is especially on the overlay construction techniques of P2P live multicasting applications.

2.3.3. Overlay Construction Techniques

One of the basic problems, or maybe the most important one, of P2P live video multicasting is the overlay construction technique of the system. Overlay structure implies the organization of the peers in the system and the relationships of these peers between each other. An efficient structure must achieve the following goals [34]:

- Overlay efficiency: The overlay should be constructed efficiently, i.e., the overall video quality should be as high as possible with the limited bandwidth of peers. The delay should be lowered to provide a live video experience.
- Scalability: The overlay should be able to support a very large number of users, reaching hundreds of thousands without extra provision over the network or the server.
- Self organizing ability: The overlay should be constructed in a distributed manner. It should be immune to peer churn and network deficiencies, and it should rebuild

itself without an interference from an authorized entity such as the server.

- Allowance to heterogeneous peer bandwidth: The overlay should not be strictly dependent on a bandwidth constraint. Considering today's heterogeneous bandwidth distribution of end users, each peer who wants to join the session should be hosted in some way. For example, different qualities of video may be served according to the download capacities of peers or they may be assigned uploading responsibility to other users as much as they can supply.

There are numerous proposals for the overlay construction techniques in the literature. These techniques are mainly grouped under two main categories: tree based and mesh based methods [34]. Regardless of the structure, all types of overlay topologies are rooted at the video source.

In tree based overlays, the structure is explicitly defined and there are established parent-child relationships between the peers. To receive the full stream, each peer should have a definite parent which is responsible for the packet delivery to its child. Each video packet generated by the source traverses the same links and interior nodes with exactly the same order, unless there is a change in the structure. A parent must forward the arriving video packets to all of its children. Since the packets are forwarded without requests from the other peers, this type of dissemination can be considered as *push-based*. In Figure 2.4, an example tree based scenario is demonstrated with the parent-child relationships. In this scenario, there are 9 peers and a server. When the server generates a video packet, it sends three copies of the packet to *Peer 1*, *Peer 2* and *Peer 3*, because the server is directly parents of these peers. Then each of these peers, forwards the packets to their children, e.g., *Peer 1* forwards the packet to *Peer 4*, *Peer 8* and *Peer 5*.

Tree based overlays are highly organized systems which makes them convenient for live video streaming and they are natural solutions for such services. However, the optimization of the structure is crucial for an efficient system because the structure is not dynamic. The reorganization of the structure in tree-based overlays is a costly operation in terms of control traffic overhead and streaming pauses in the peers. Also, the

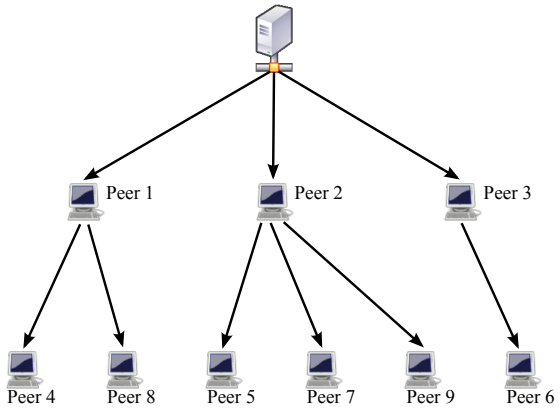


Figure 2.4. An example tree based scenario.

imperative dependency to ascendants is a drawback for tree based systems. Returning to the example, *Peer 1* must be online and continuously uploading the packets, for *Peer 4* to receive the video stream. This dependency to the peers is more critical for peers that are close to the server as the total number of descendants below a hierarchically important peer may be very large. Unless a redundancy or retransmission technique is implemented on such a protocol, descendants of a left ancestor cannot receive the video packets until the structure is renewed. Therefore, maintaining the structure is a very critical problem for tree based systems where the structure is affected with each peer leave or participation. To reduce the dependency to a peer, the fan out degree should be limited for each peer. Although the single point of failure may be relaxed up to a certain extent, such a limitation brings along increased heights, i.e., the number of hops separating a peer from the server, eventually causing the tree to be more fragile. The probability of a peer to be unable to receive the video packets increases with the number of ascendants between the peer and the server. Furthermore, the unused uploading bandwidth of the leaf peers (peers with no children) is another concern for this structure. Favoring the leaf peers is an undesirable property of P2P systems, which rely on uploading devotion from peers.

There are many studies in the literature for tree based overlay systems such as [17, 18, 35]. However, a single multicast tree for the whole video stream is proved to be insufficient and implementing multi-trees for a single video stream is proposed in [36]. In such a structure, the video is split to a few parts and each part is disseminated through each tree separately where the trees are managed distinctly. In this way, the

resiliency of users to peer churn is increased, because, even if a parent leaves the session, its children continue to receive the packets sent over the other multicast trees unless they are connected to the same parent in each tree. Also, preventing the peers to become a leaf at least in one multicast tree, increases the bandwidth utilization and fairness among the end users. In [36,37], multi-tree structures are used within different P2P multicast protocols. Also in [37], the protocol utilizes the multipath diversity with improved video coding techniques.

On the other hand, most of the studies in the literature are built on mesh based systems. There is no or little structure in such systems. Each peer maintains relationships with several other peers called neighbors. Unlike tree based overlays, these relationships rely on availability information of video packets rather than parent-child relationships. The data exchange between the peers can be maintained by two means, which are push-based and pull-based approaches. In push-based systems, a peer sends a packet to another peer without having the availability information in the potential receiver. A peer may send a packet to one of its neighbors who is already holding that packet, hence both uploading bandwidth of the sender and downloading bandwidth of the receiver are wasted. The massive redundancy created by push-based approach is a drawback for video delivery systems due to the large bandwidth demanding nature of video content. Push-based solutions are proposed in the literature such as epidemic dissemination [38], however they are eventually deprecated due to their inherited disadvantage. On the other hand, video packets are sent upon the requests of peers from their neighbors indicating the lack of requested packet in pull-based approaches. To maintain a list of available packets in their neighbors, peers periodically exchange buffer maps. Pull-based approaches increase the efficiency in terms of bandwidth, i.e., they do not introduce redundancy. In other words, a peer may receive a packet at most one time. Exchange of buffer maps resembles the file sharing applications, however the real time constraint converts this routine operation into an obstructive problem. Buffer map exchange messages and pull requests constitute a high percentage of overall traffic in mesh based systems, making it more costly than it looks.

There are several advantages of mesh based systems. First of all, the structure maintenance costs are much lower than those of tree based structures because these systems do not require frequent heartbeat messages due to their inherent redundancy. In mesh based systems, a peer is not totally dependent to only one parent and the burden created with the peer churn is eased. The structure does not need to be reconstructed with each leave or participation of the peers. Hence, the mesh structure increases the robustness against failures. Another advantage arises from the changing routes at each packet. Unlike the tree based structures, each packet may follow a different path depending on the network conditions and protocol specifications. This dynamic topology causes path diversity and increases the resilience against peer failures and network congestions. In addition, it resolves the bandwidth utilization problem in leaf nodes because they cannot steadily preserve their leaf locations within the dynamic overlay. However, the dynamic nature of mesh based systems introduces unpredictable efficiency as a trade off. Since there is no dedicated parent-child relationship, timely delivery cannot be guaranteed even if the peers are in online state. Degradations in the quality and increased packet latency become the main bottlenecks of the mesh based protocols. The start-up delays may be too long in these systems and the user is frequently faced with freezing screens. The majority of recent P2P multicast studies are based on mesh based overlays [39–46].

To overcome the difficulties arising from aforementioned types of overlay structures, namely tree based and mesh based, hybrid systems are proposed in several studies [47, 48]. The basic motivation behind the hybrid systems is using the highly structured nature of tree based systems which guarantees efficient delivery with well-defined relationships, and overcoming the sensitivity to peer churn by introducing the mesh based approach. Putting some degree of redundancy, where peers can demand the missing packets or peers may have diverse paths to receive the video packets, is the only solution to existing problems.

2.4. Deployment Scenarios

For the last 15 years, live video broadcasting via P2P multicasting systems did not remain as a theory. There have been many successful experimental and commercial attempts. The widely known efforts include PPStream [49], PPLive [50], TVAnts [51], TVUPlayer [52], GridMedia [44], Zattoo [53], SopCast [54], and [17,42]. Peak concurrent users are given below for several of these applications:

- PPLive attracted more than 200000 users on 28 January 2006.
- SopCast reached a peak simultaneous viewers of more than 100000 [55].
- CoolStreaming [42] multicasted to more than 80000 concurrent users [26].

From now on, P2P live multicasting is not only a promising technology, because, the self scaling ability of P2P multicasting has proved its feasibility and it has been found satisfactory by thousands of people all over the world.

3. STANFORD P2P MULTICAST PROTOCOL

In this section, we explain SPPM protocol [5] in detail which is used as the basis of our simulations. The protocol enables the construction and maintenance of multiple trees which are routed at the server. The protocol is completely push based and trees are managed distinctly. The video is split into distinct streams where each one is sent over a different tree. A peer must participate in every tree to reconstruct the video in highest possible quality by acquiring path diversity. Since the approach is self scaling, a server with a very limited bandwidth capacity is enough for the survival of the network. This protocol is especially designed for low end-to-end latency and minimum start-up delay.

3.1. Protocol Structure

The control protocol is executed on each participant. In this section, we first explain the protocol used for peers by demonstrating it with flow charts, and then we present the server's protocol structure. It is assumed that peers know the IP address of the server priorly. Application consists of two types of processes, namely the main process and tree process. Main process starts to run with the peer's participation in the session until it leaves the multicast. A tree process is generated by the main process for each multicast tree of the session. The tree number and data rate of video are kept constant for each multicast session. Protocol timers used in the protocol are written in capital letters. The content of each packet type and their lengths will be discussed later.

3.1.1. Main Process

Main process consists of OFFLINE, JOIN, PROBE and ONLINE states. The OFFLINE state represents an inactive peer which may have not yet started the participation task or may have left the session. A new peer immediately switches from OFFLINE to JOIN state by running the application. The diagram for this state is

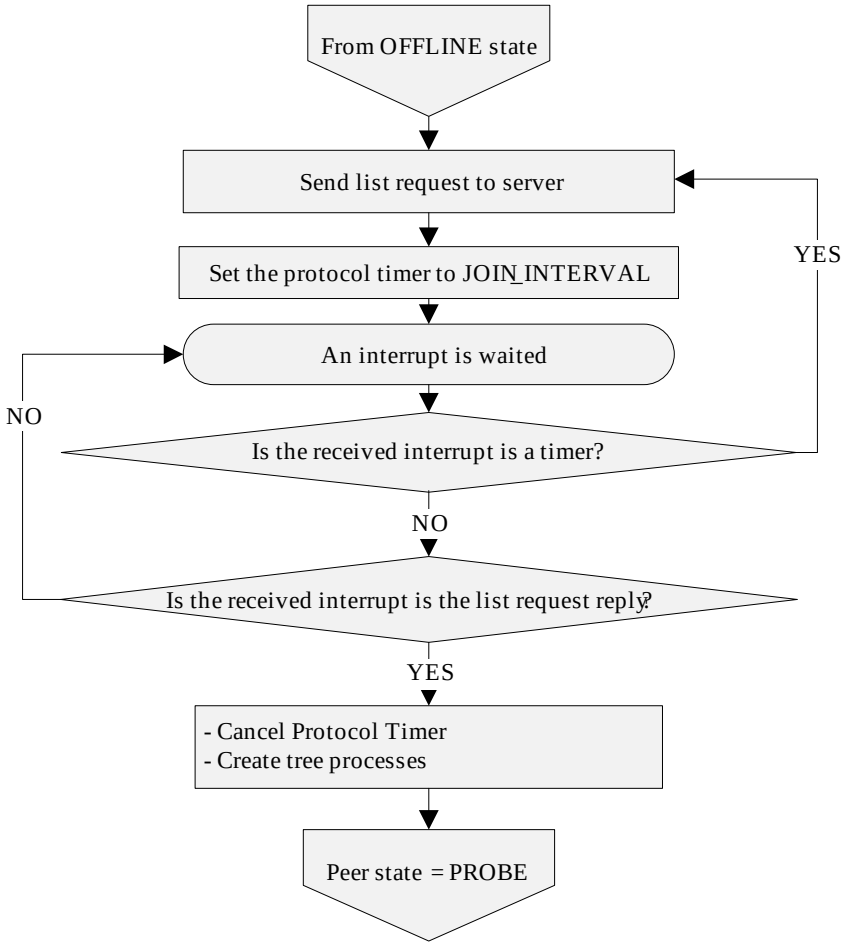


Figure 3.1. JOIN state of main process.

given in Figure 3.1.

The peer requests a list of online peers from the server and sets its protocol timer to JOIN_INTERVAL indicating that a reply is expected within this duration. The peer waits until a list request reply packet arrives. If a reply does not arrive within this duration, the peer sends a new request message and repeats this process until it receives the packet. The list request reply message consists of a list of online peers in the session at that moment, the source rate, and the number of trees used for the multicast session. After receiving the reply, main process constructs tree processes according to the number of trees and also cancels the protocol timer. The tree processes are initialized in OFFLINE state[3] . The main process leaves the JOIN state by switching to PROBE state. From this moment on, each tree process operates separately and it is responsible for the control of that multicast tree.

[3]The tree process will be explained in Section 3.1.2.

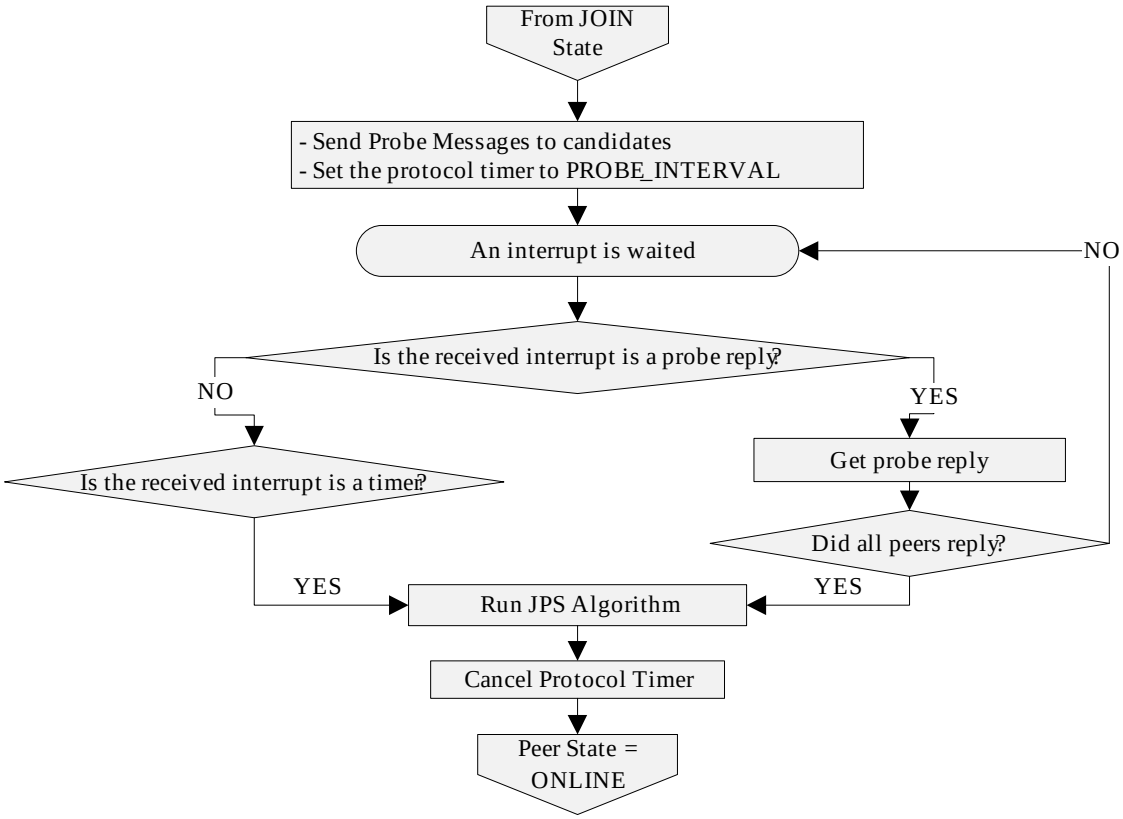


Figure 3.2. PROBE state of main process.

In PROBE state, the protocol aims to collect the condition information of each candidate peer obtained through the list request reply. Firstly, the peer sends probe messages to each candidate in the list request reply send by the server to request information and sets the protocol timer to PROBE_INTERVAL. As the probe replies arrive, the peer handles these packets and collects information on the replied peers. A probe reply packet indicates its owner's available throughput and height in the multicast trees. Height refers to the number of hops separating a peer from the server. The peer continues to wait for additional probe replies until all probed candidates transmit their reply packets or the protocol timer expires. As the number of replied candidates increase, the peer can locate itself in a more secure position in the overlay. However, to control the overhead introduced by the probe packets, the list length is limited by the server as explained in Section 3.1.3. Also, the protocol timer prevents the peer from unnecessarily waiting for congested or departed candidates.

After the peer fulfilled the aforementioned condition, it cancels the protocol timer and runs the JPS algorithm depicted in Figure 3.3 to jointly select the best candidate

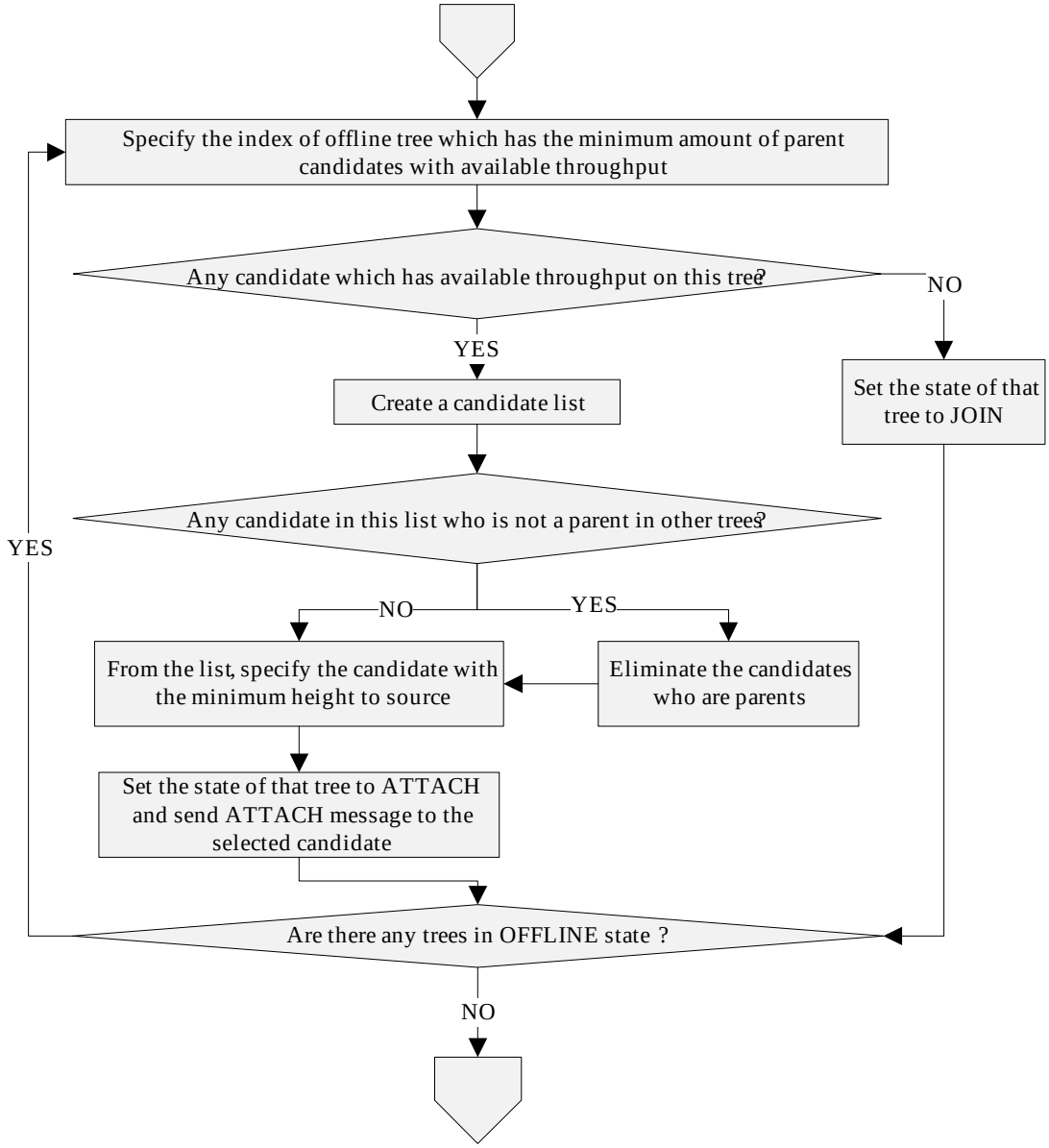


Figure 3.3. Joint Parent Selection algorithm of main process.

parents for all multicast trees. In JPS algorithm, the protocol tries to find a candidate parent having available throughput, for each tree. At every iteration, the tree with the minimum number of candidates are handled to efficiently use the bandwidth.

A child is seriously affected with the leave of a peer who is the parent in multiple trees due to the interruption in the video stream, and also the overhead of relocating in several trees. Since the main motive behind the multi-tree protocols is to provide path diversity and resiliency to peer churn, the peer prefers selecting different candidates for each tree and creates a list of candidates who are not already parents in other trees. Obviously, if there is not such a candidate, the peer considers the ones who are already parents in the other trees. Then, the peer selects the candidate with the minimum height, i.e., the candidate which is hierarchically highest, from the list to firmly cling to the tree. For each tree in which a potential parent is assigned, attachment requests are sent to the potential parent indicating the request of connection and the corresponding tree process is transitioned to ATTACH state. If there is not any candidate for a tree, its process is switched to JOIN state.

Generally, the total number of candidates would be the same for each tree because each tree in SPPM protocol carries approximately the same data rate in our study. However, this is not valid if the server is also a potential parent because the server reserves equal bandwidth and accepts the same amount of children for each tree to guarantee a balanced multi-tree structure.

After the iteration for every tree is completed, the peer switches to the ONLINE state which is demonstrated in Figure 3.4 and remains in this state until it leaves the session. The arriving packets are processed according to their types. If the packet belongs to one of those types which are list request reply, probe reply, attach request, attach reply, hello, hello reply, leave notice and video, the packet is directly forwarded to the corresponding tree process and handled there. On the other hand, the probe packets are handled in the main process since the reply conveys information about the overall condition of the peer in all trees. A probe reply is sent to the probing peer, containing the available throughput information and heights in every multicast tree.

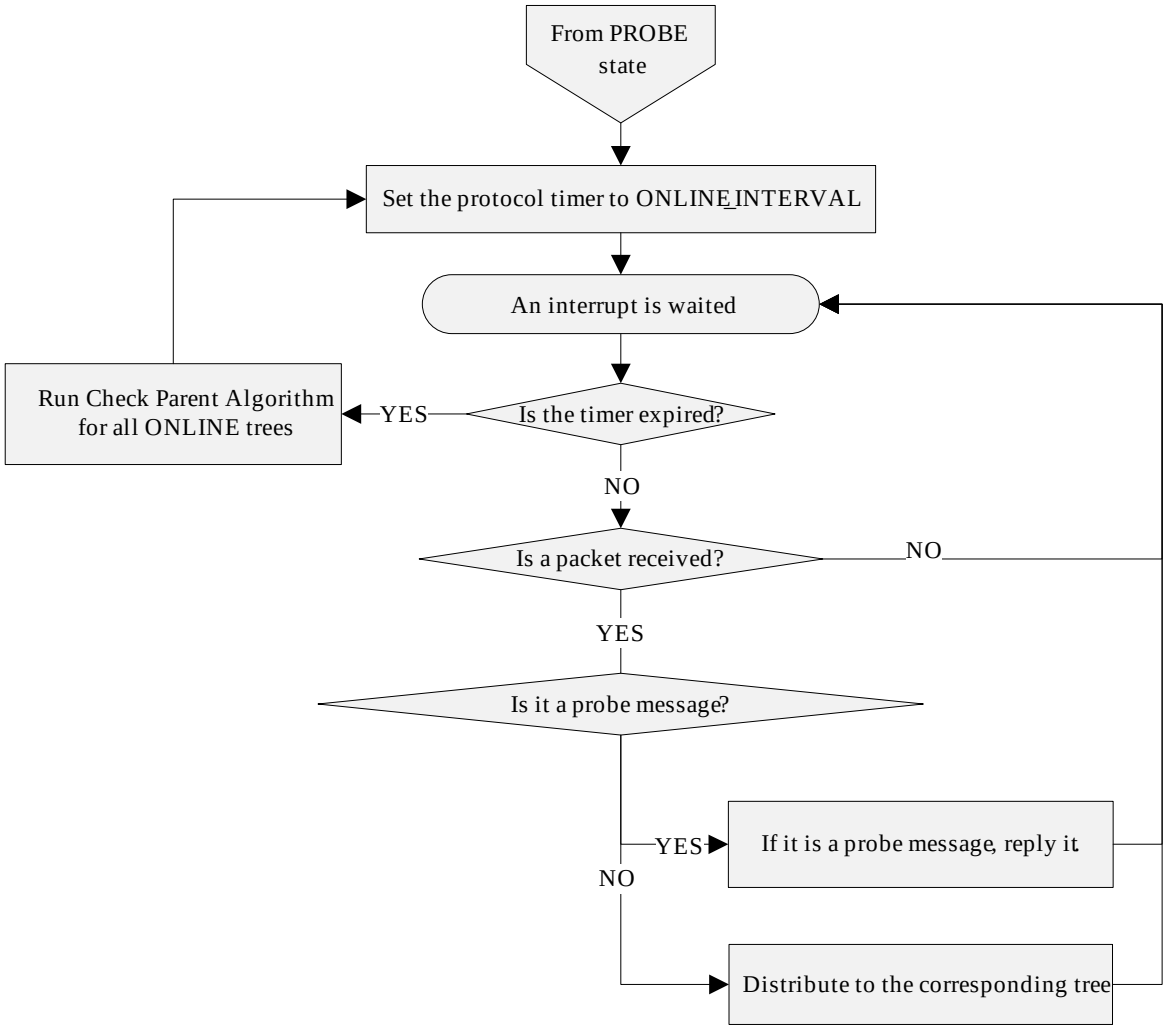


Figure 3.4. ONLINE state of main process.

In addition to packet management, the peers periodically check the connections with their parents by monitoring the hello messages and video packets. CP algorithm given in Figure 3.5 is used to determine if the parents are online and the established connections are working properly. If a specified time interval has elapsed after the last video packet was sent over a tree and the parent is not responding to hello packets by sending hello replies, the parent is regarded as disconnected from the session and corresponding tree is informed that the parent is offline and a new parent should be found for this tree. Even if a parent is online and replies the hello messages, it is discarded if it has not forwarded video packets for a large interval. The reason of this phenomenon is that, the congestion over the network is also a penalty for live video streaming and the control protocol must be capable of generating a solution for the problem. In this way, the disconnected parents are detected in a decentralized manner and the overlay repairs itself as needed.

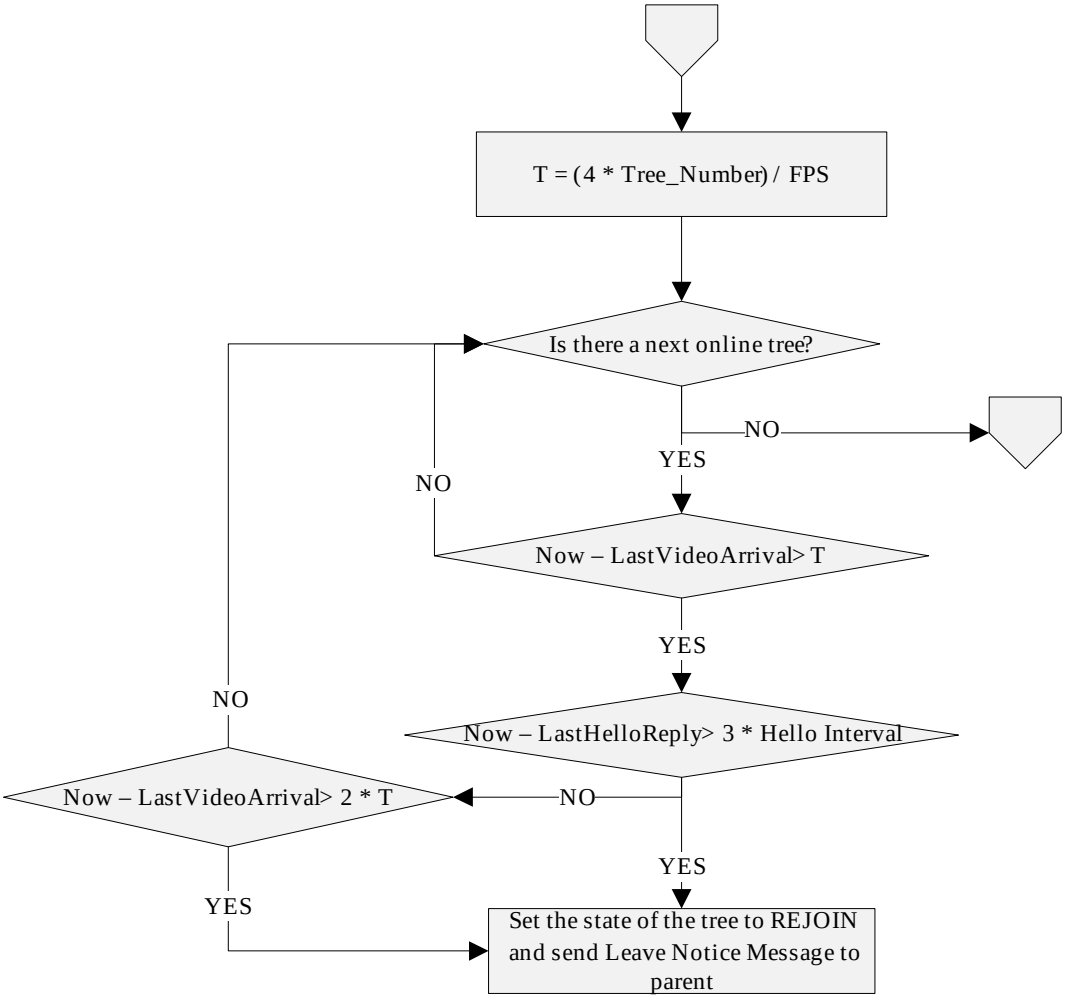


Figure 3.5. Check Parent algorithm of main process.

3.1.2. Tree Process

When a tree process is created by the main process upon the list request reply, it is automatically initialized at the OFFLINE state. In the OFFLINE state, the tree process waits for the results of JPS algorithm running in main process. If the peer is assigned to a potential parent in the tree through JPS algorithm, the main process sends an attach request message to the potential parent and interrupts the tree process for transition to ATTACH state. If there is not any candidate parent for this tree, then its state is set to JOIN by the main process. Once a tree is initialized and left the OFFLINE state, it never switches back to this state again.

The tasks in the JOIN state of tree process in Figure 3.7 are very similar to those of main process. However, this time the server replies with a peer list who are not only

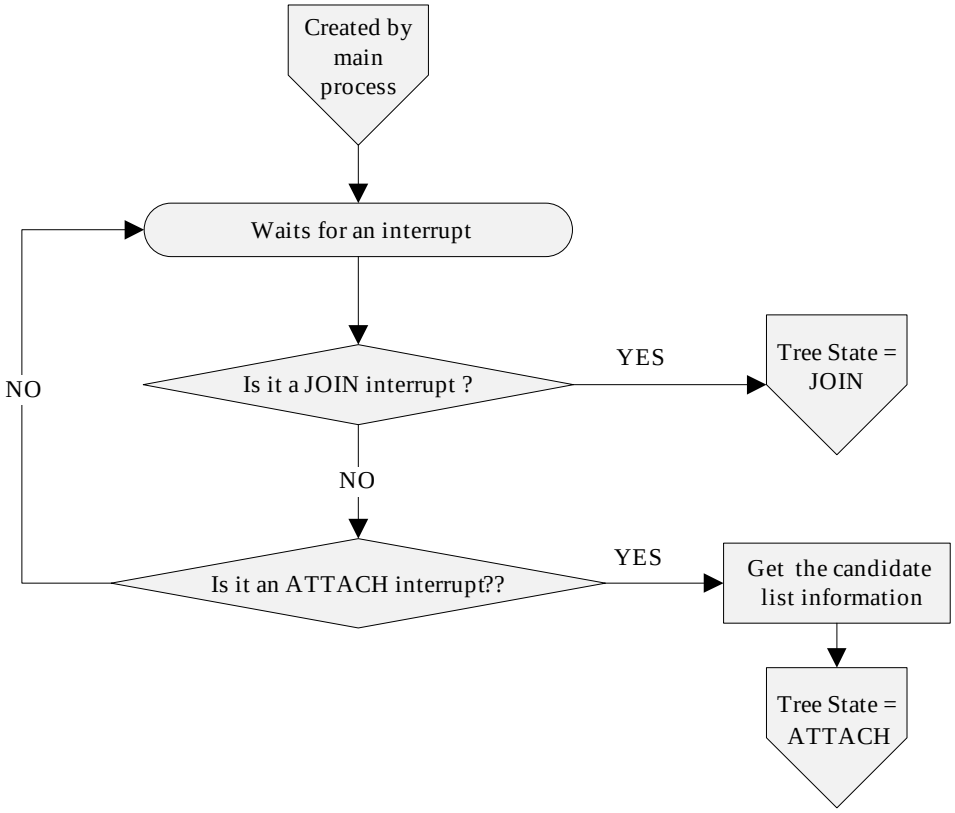


Figure 3.6. OFFLINE state of tree process.

participating in the multicast session, but also having an active connection with their parents, i.e., the peers in ONLINE state for this multicast tree. The tree process may transition to this state from Single Parent Selection, ATTACH, OFFLINE and REJOIN states when a fresh list of candidates is required.

In PROBE state shown in Figure 3.8, the tree sends probe messages to the peers in the list to get information. As many as probe replies are collected in a specified time interval. There is a trade off between the control overhead on the system and the structure efficiency. As the number of probed peers increases, the probe packets create a burden on the system while the trees gain the chance to select a better parent in terms of structure efficiency.

Along with quiting the PROBE state, the tree process executes the SPS (Single Parent Selection) algorithm which is described in Figure 3.9. In this state, the algorithm evaluates the feasibility of probed peers and tries to select a potential parent to establish a connection. SPS algorithm resembles JPS algorithm except that it only considers the

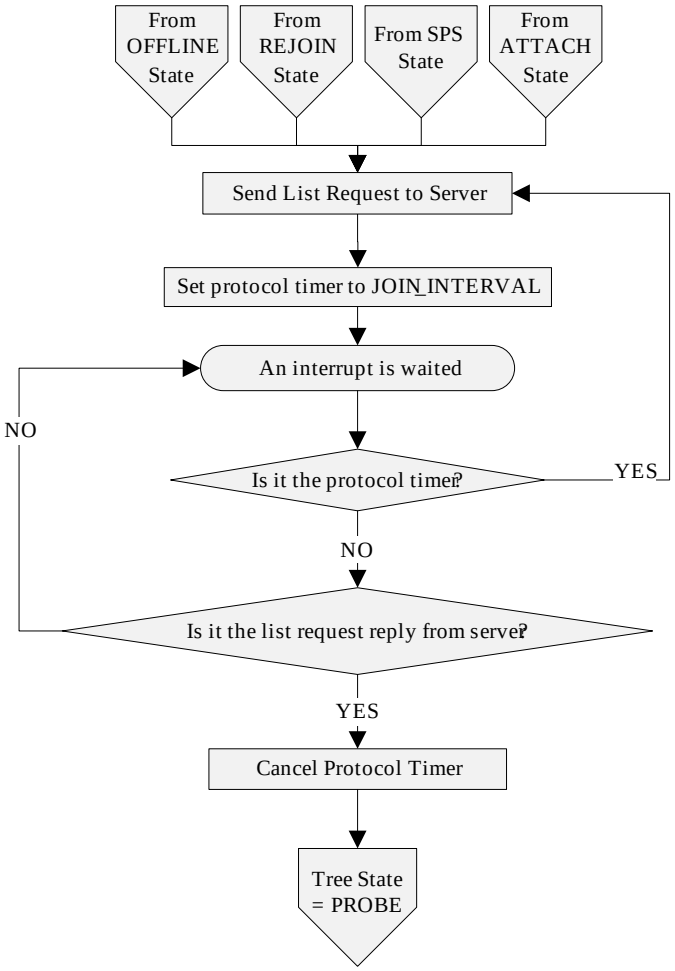


Figure 3.7. JOIN state of tree process.

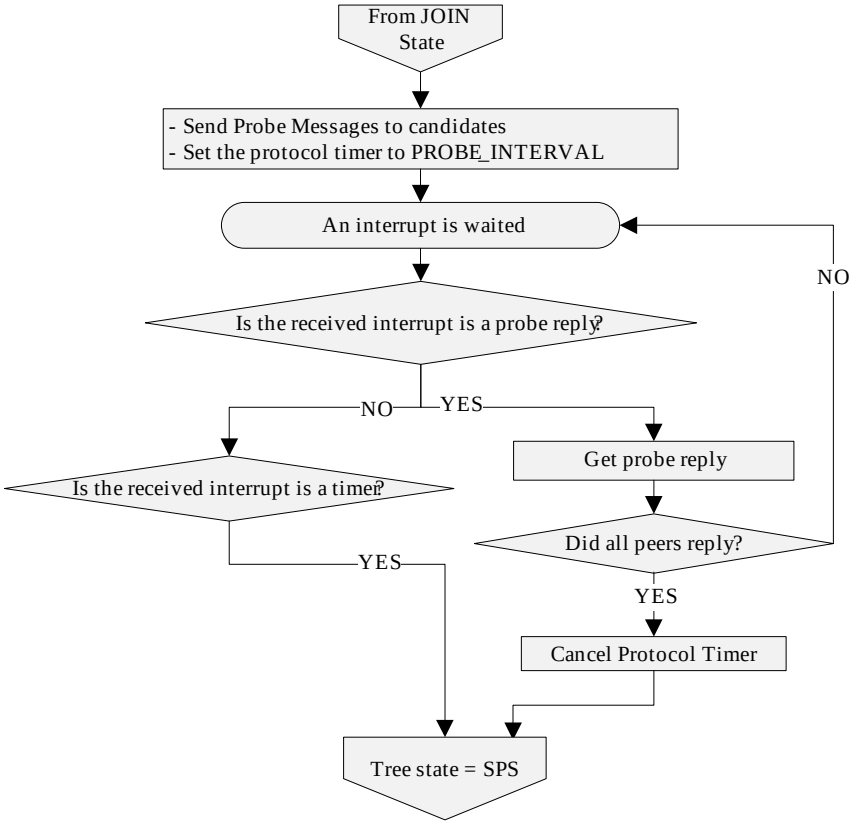


Figure 3.8. PROBE state of tree process.

subject tree. First, the peers with no available bandwidths are eliminated from the list. To provide a first check, the children of this tree are also eliminated from the list to prevent loops in the structure. The loop problem of tree based overlays is demonstrated in Figure 3.10. Initially, peer 1 serves a large number of descendants in the overlay. When a peer detects a disconnected parent, it starts searching for a new potential parent by probing the candidates sent by the server. In other words, the children of peer 1 request an updated peer list from the server following the disconnection of peer 1 from the tree. However, there is a high possibility that the list contains at least on of the descendants of peer 1 who has peer 4 in its upstream and the peer may blindly try to connect to one of those peers. For example, the peer 4 may attempt to establish a connection with peer 2 in Figure 3.10b. If there is no protection mechanism against loop emergency, peer 2 accepts the peer 4 as a child, creating a loop. Since a peer has the information about upstream peers, i.e., the peers separating it from the server, through the exchange of heartbeat messages, the loop avoidance is assured in the descendant peers when an ascendant tries to attach to one of those.

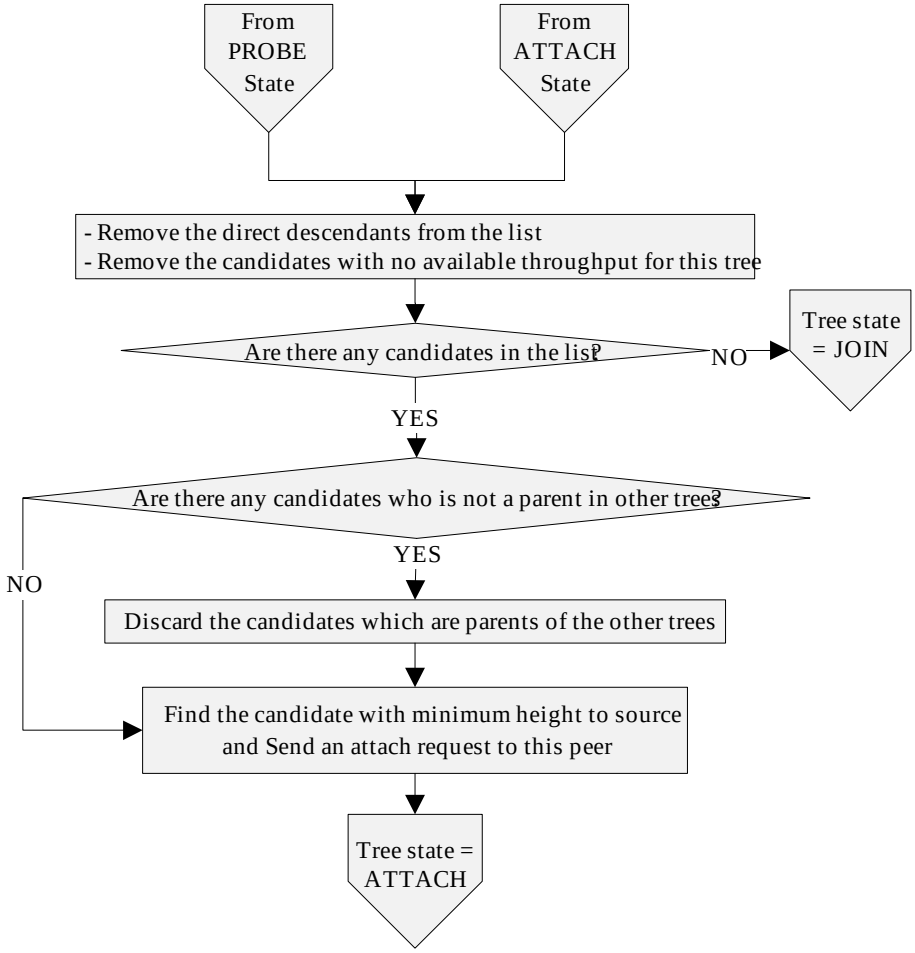
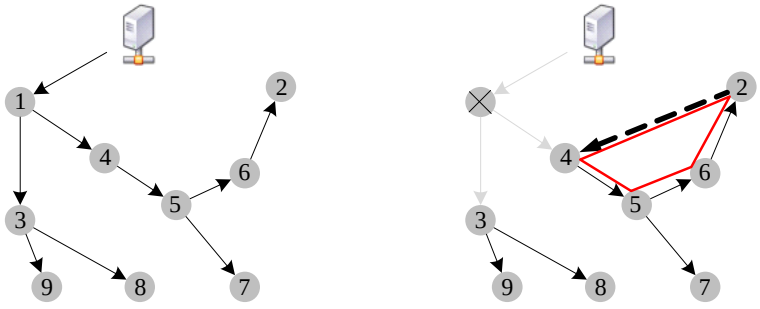


Figure 3.9. SPS state of tree process.



(a) Initial overlay. (b) Loop emergence.

Figure 3.10. Loop problem in tree based overlays.

As in the case of JPS, SPS prefers the candidates who are not already parents of other trees to provide diversity and resilience. From the final candidate list, SPS selects the peer with minimum height as the potential parent and sends attach request message to initiate a connection. Obviously, if there is not any candidate, the tree process switches to JOIN state to acquire a fresh list of online peers.

After sending attach request messages, the tree process switches to ATTACH state to wait for a reply from the potential parent. An attach reply is expected within a specified interval and the protocol timer is set accordingly to decrease the latency. If the potential parent positively replies the request, the information regarding the parent is initialized and the connection is established. In the attach request reply message, a parent also indicates its available throughput, height in the tree, and the addresses of its ascendants. These data are updated through the hello and hello reply messages in the session for each parent. Also, a join message is sent to the server stating that the connection for this tree is successfully established and the peer can serve as a parent in this tree. This information is used by the server for the tracking of online peers. Then, the tree process sets the timer and switches to ONLINE state. On the other hand, if the candidate negatively replies the attach request, the peer transitions to JOIN state to get a new list of online peers. For unanswered requests, the corresponding candidates are removed from the list and tree process reactivates the SPS algorithm to find another appropriate parent. However, the first requests generally become successful unless there is a flash crowd, because the available bandwidth of the peers are already known prior to parent selection algorithms.

Upon a successful connection establishment, the peer transitions to the ONLINE state as in Figure 3.12 and remains in this state until its parent is disconnected from multicast detected by CP algorithm running on the main process. In ONLINE state, each peer periodically exchange *heartbeat* packets with their parents to check if the other one is alive. Peer sends hello messages to its parent which include the RTT of the connection to the parent. As long as a child sends these periodic messages, its parent assures that the uploading bandwidth is efficiently utilized. Upon the arrival of a hello packet, parents send hello reply messages indicating that it is still connected

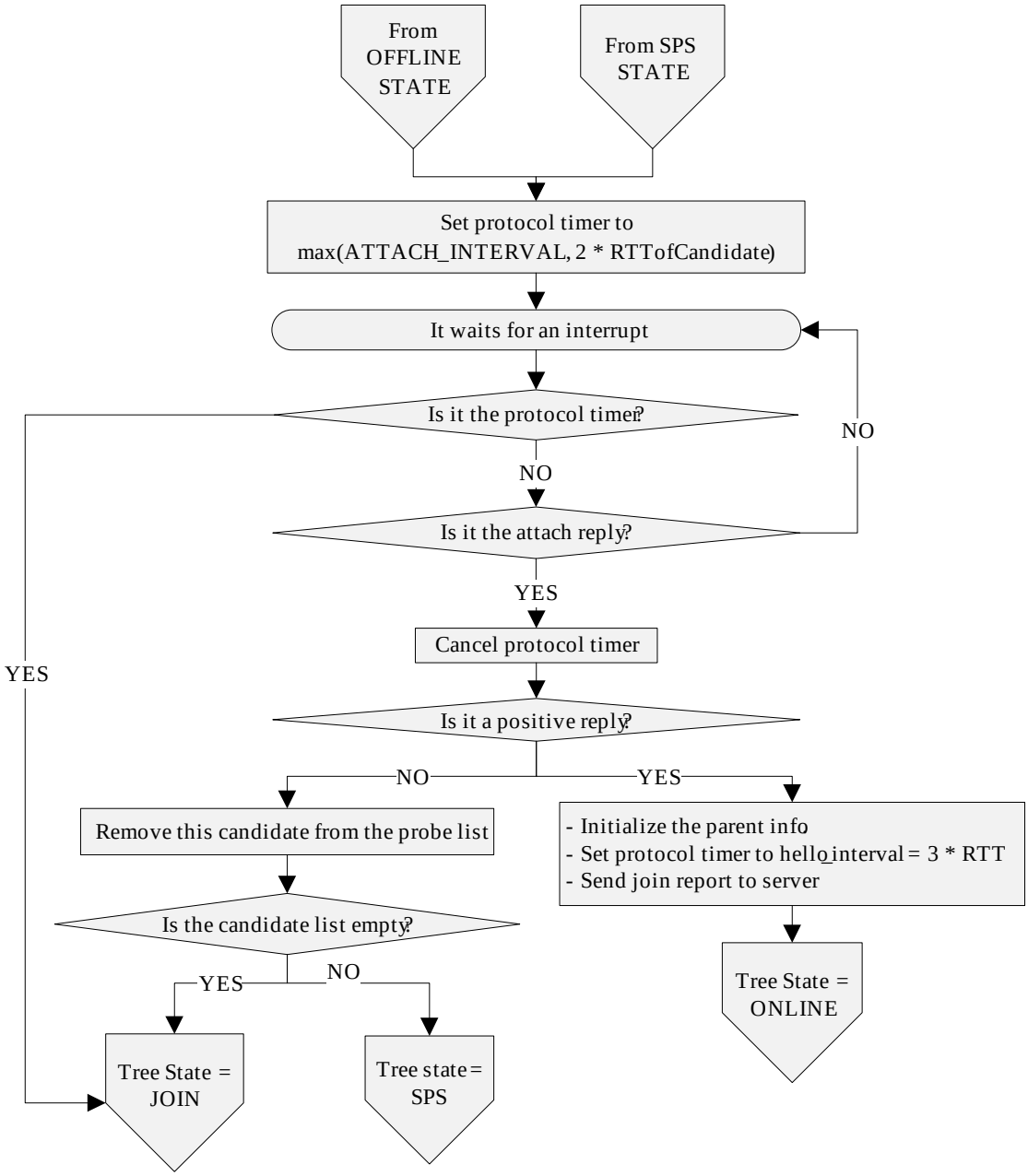


Figure 3.11. ATTACH state of tree process.

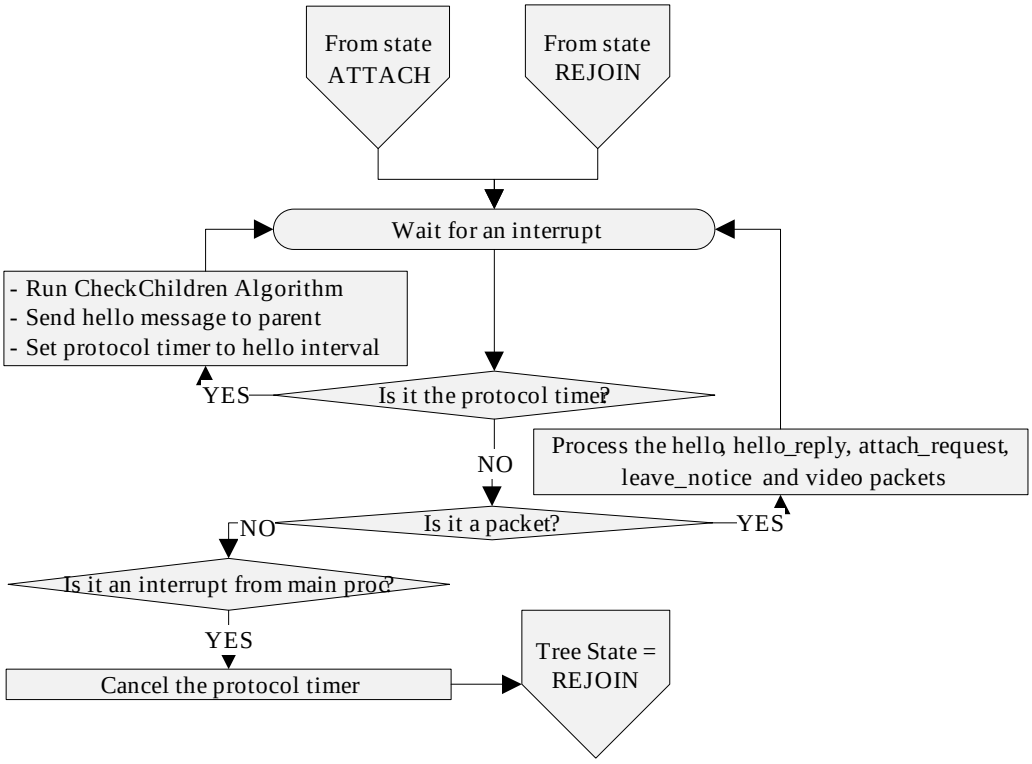


Figure 3.12. ONLINE state of tree process.

to the multicast stream. Hello reply packets update the information initialized with attach request reply packets, representing the height in the tree, available throughput and ascendant peers. Upon the arrival of a hello reply packet, a child is assured that its parent is still connected to the multicast. Through the exchange of these messages, RTT is calculated as in [5] at the child peer using a moving average as follows:

$$RTT_n = 0.7RTT_{n-1} + RTT_{last} \tag{3.1}$$

where RTT_n is the updated RTT information, RTT_{n-1} is the previously stored RTT information and RTT_{last} is the RTT of last exchanged hello messages. The configuration of heartbeat frequency requires an updated RTT value.

Also, a peer checks its children in regular intervals to prevent temporary waste of upstreaming bandwidth. Since a false alarm may be a serious penalty for the overlay, CHILD_INTERVAL, the maximum allowed time for an unresponsive peer, is kept larger than those used for CP algorithm. CC (Check Children) algorithm is described in Figure 3.13 where the parent checks its children one by one and the disconnected ones

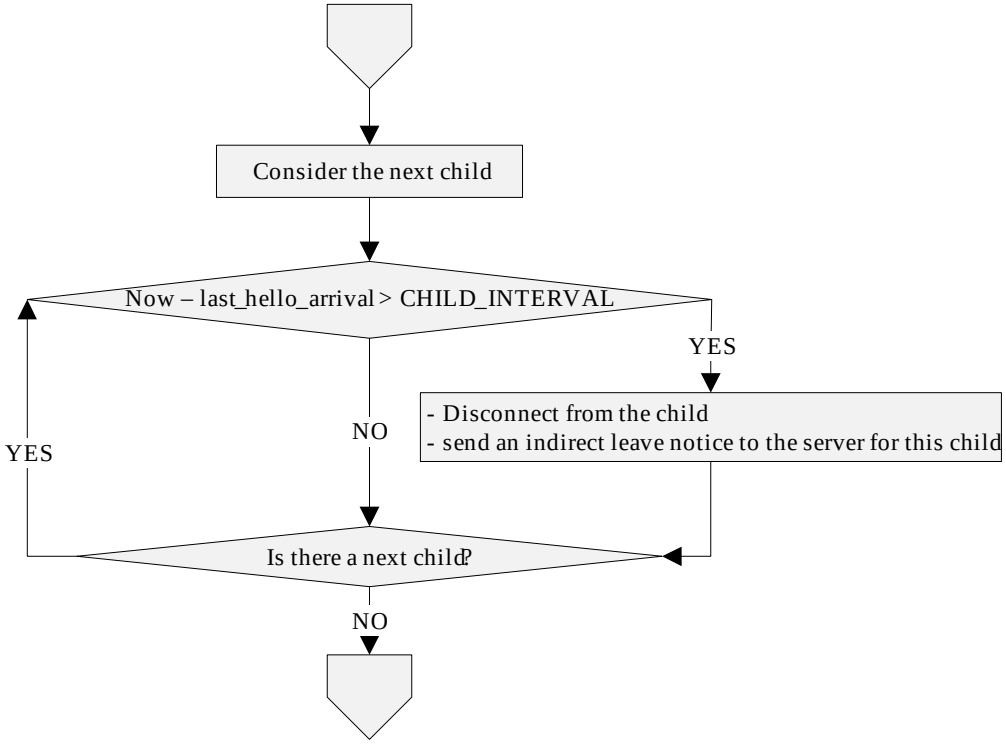


Figure 3.13. CC algorithm of tree process.

are removed from the parent's children list. Also, the peer sends an indirect leave notice to the server so that it can correspondingly update the online peers list.

In ONLINE state, the tree process is responsible for handling the packets forwarded by the main process. If the incoming packet is an attach request, it generates an attach request reply, content of which depends on the available uplink throughput of the peer. If the peer has enough available throughput to host one more child in a tree, the request is approved by setting the flag to 1, otherwise the flag is set to 0 indicating that the peer is unavailable. Along with the request response, the peer also sends additional information including available throughput, height in the tree and upstream peers. Upstream peers are very important in the tree based structures to guarantee loop avoidance. If the incoming packet is a leave notice from a child, the peer removes this child from its forwarding list.

If an interrupt is received from the main process indicating the disconnection of the parent, the tree process transitions to REJOIN state given in Figure 3.14 to provide a fast recovery mechanism for the tree. In this state, a parent of another tree with available bandwidth is randomly selected. If such a parent does not exist, the tree

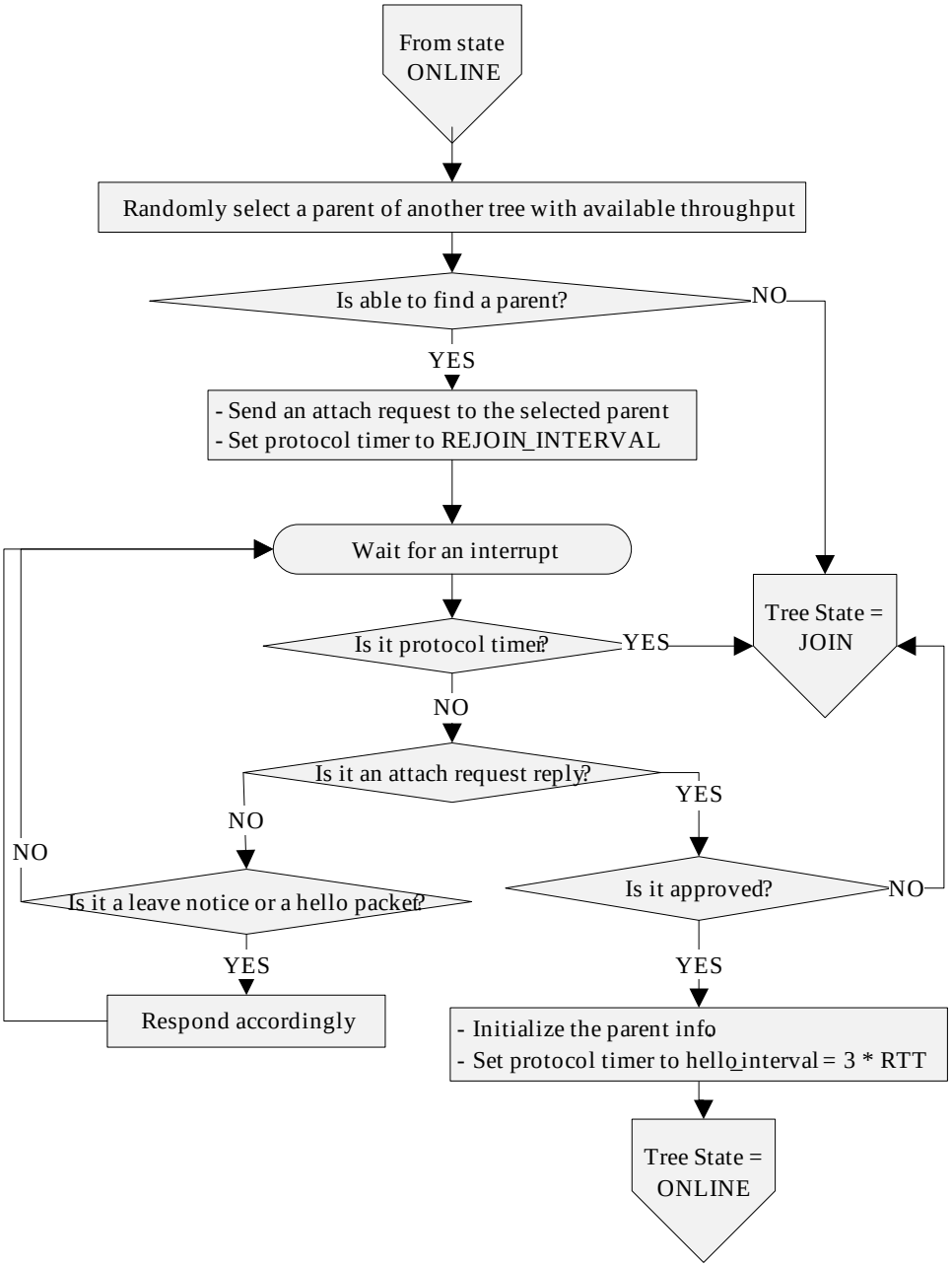


Figure 3.14. REJOIN state of tree process.

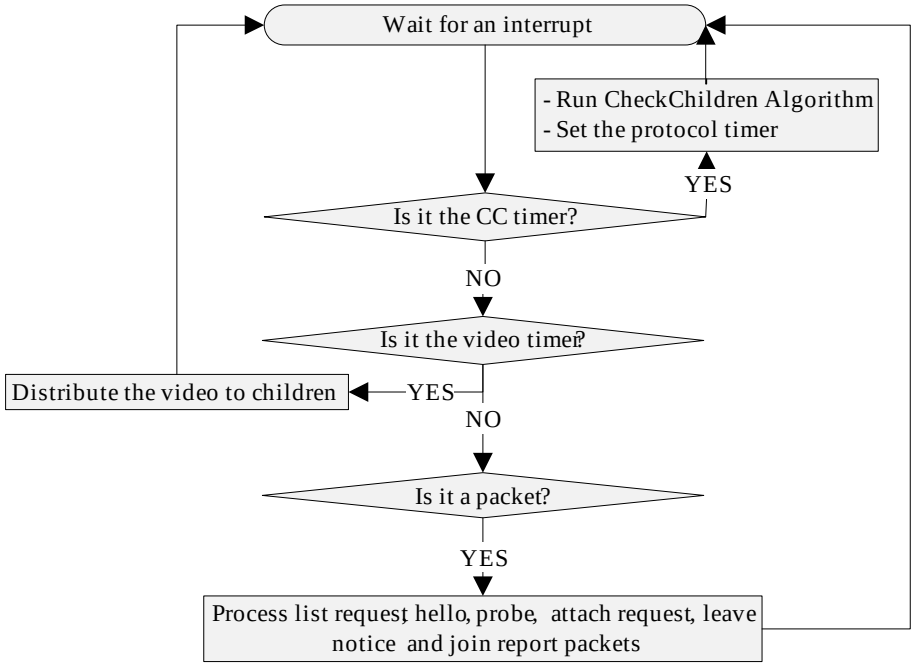


Figure 3.15. ONLINE state of server process.

process switches to JOIN state to request an updated peer list from the server. If there is a potential candidate, an attach request message is sent to this peer to establish a connection. If a positive reply is received in a specified interval, the peer initializes the connection with the new parent. Otherwise, the tree transitions to JOIN state.

3.1.3. Server Process

The server process is executed on the server for overlay management tasks and it also covers the most of the functionalities performed in a standard peer. The server process continuously runs in the ONLINE state shown in Figure 3.15 and covers all multicast trees. Besides being the key point of the system, it acts as a peer serving to several children, without a connection to a parent. Server replies probe packets, attach request packets, and hello packets in the same manner with a peer. However, server sends available throughput messages for all trees distinctly because it is expected to accept the same number of children for each tree to prevent imbalance in the quality of multicast trees. In this case, waiting interval is more restricted because a temporary waste in the uploading bandwidth of server extremely affects the overall structure of network, causing long disruptions and increased hop numbers.

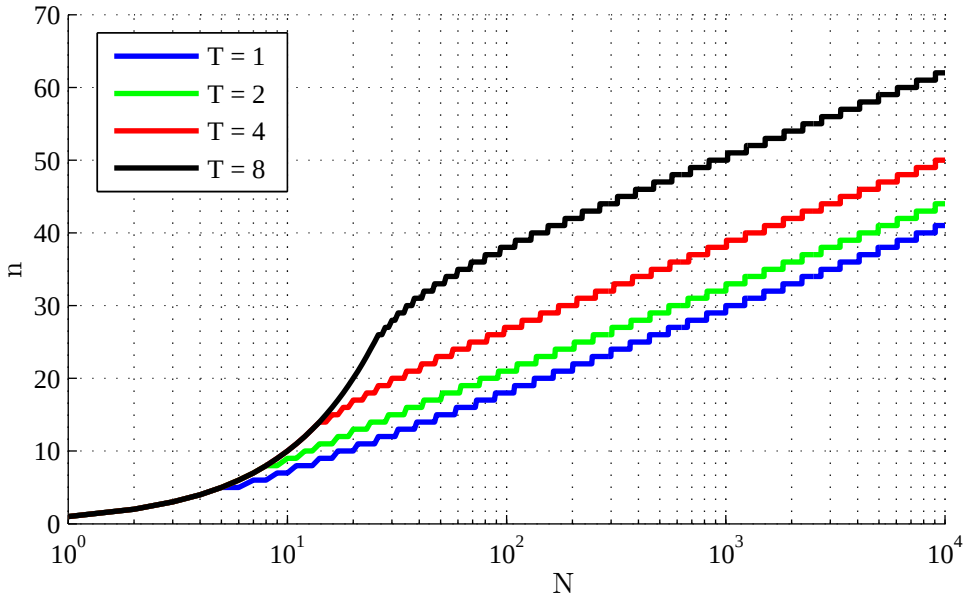


Figure 3.16. Size of candidate list for different number of trees.

The server creates and updates its online peer list by means of leave and join packets. As we discussed in Section 3.1.2, a peer sends a join report to the server indicating that it is connected to the multicast session and available to serve to the other peers. The server also processes the leave notice packets which are sent by parents detecting a disconnected child, and updates the list by removing the disconnected peer.

One of the key functions of the server is replying the list request packets by generating a fresh list of peers. However, to keep the control traffic over the network at a certain point, the number of candidates in the packet is specified in [5] as

$$n = \begin{cases} N & \text{if } N < 3T, \\ \lfloor 5\ln(N-(3T-5)) + 3T - 5\ln(5) + 0.5 \rfloor & \text{otherwise,} \end{cases} \tag{3.2}$$

where n is the length of the candidate list, N is the total number of online peers, and T is the number of trees in the multicast session. The list size for different number of trees are plotted in Figure 3.16.

3.2. Performance Evaluation

The simulations are carried out for a set of parameters mentioned in the previous sections, which are same as those used in [27]. In Table 3.1, we provide the intervals used for the protocol timers throughout the simulations. The protocol timers provide a safety against unnecessarily waiting a congested peer for a long time and it is important to adjust these parameters short enough for a low latency service but long enough to ensure an efficient utilization of the overlay structure.

The packet types used for protocol control are depicted in Table 3.2. It is assumed that UDP/IP protocol stack is used in the network and the control packets are encapsulated in IP layer. We assume that a typical IP packet has a 20 bytes header length whereas a typical UDP packet has a length of 8 bytes. A total of 40 bytes for very short packets, namely list request, probe, attach request, join report, leave notice, hello, and hello reply packets is more than enough for such an implementation.

Since the list request reply packets may include a large number of candidate addresses, its packet length varies between 50 and 500 bytes. However, it is assumed that the average length of a list request reply packet is 120 bytes.

Probe reply messages include an additional height information encoded in 4 bytes for every tree. Therefore, the packet length of this type is assumed to be 80 bytes, which is longer than the short packets.

The attach reply packets include the available throughput information and the upstream peers. Although the number of upstream peers is variable, it typically varies between 1 and 15, and an 80 bytes length would be sufficient for such packets.

Throughout the simulations, the link delay is assumed to be constant and 5 ms for each link. For this section, we assume that there is no packet loss over the links. We use a server with 1.4 Mbps upload and download bandwidth. Peer bandwidth distribution is derived from [56] as given in Table 3.3. It can be seen that as the number of multicast trees increases, the uplink bandwidth of the peers can be more efficiently utilized since

Table 3.1. Protocol timers.

JOIN_INTERVAL	0.5 sec
PROBE_INTERVAL	0.5 sec
ATTACH_INTERVAL	0.5 sec
ONLINE_INTERVAL	0.033 sec
CHILD_INTERVAL	2 sec
REJOIN_INTERVAL	$\max(0.5, 3RTT + 0.3)$ sec

Table 3.2. Typical packet lengths.

Type	Length (bytes)
List Request	40
List Request Reply	120
Probe	40
Probe Reply	80
Attach Request	40
Attach Reply	80
Hello	40
Hello Reply	40
Join Report	40
Leave Notice	40

Table 3.3. Bandwidth distribution of peers.

Percentage	Uplink	Downlink
56%	256 kbps	512 kbps
21%	384 kbps	3 Mbps
9%	896 kbps	1.5 Mbps
3%	2 Mbps	20 Mbps
11%	5 Mbps	20 Mbps

they can use their residual bandwidths. For example, 56% of all peers are unable to serve a 300 Kbps stream to a child if there is only one multicast tree. In the case of four multicast trees, such a peer can now serve up to three children, increasing the total bandwidth resource of the system.

The protocol assumes that the peers can measure and know their approximate uplink and downlink bandwidths. There are many techniques proposed by academia and commercial tools providing this service. In [57], authors investigate the bandwidth measurement techniques and provide a comprehensive taxonomy.

During the simulations, the dynamic behavior of peers is modeled with Poisson distribution. After a node is switched on, it remains awake for an exponentially distributed interval with an average time of 4.5 minutes. At the end of this interval, the node is switched off by the system, and the peer goes offline, representing an ungraceful leave. To create a cycle, it is again switched on after an average time of 30 seconds. This interval is also exponentially distributed. For initialization, we simulate a flash crowd by letting all peers join the system in the first 60 seconds. Simulations are carried out for 30 minutes.

The control overhead of video multicasting systems is one of the most important characteristics. The protocol must be able to maintain itself without creating overwhelming control traffic which lowers the overlay efficiency. In Figure 3.17, the aggregate traffic of the network is shown for both video traffic and control traffic served for different number of participants with a four-tree multicast session of 240 Kbps. The

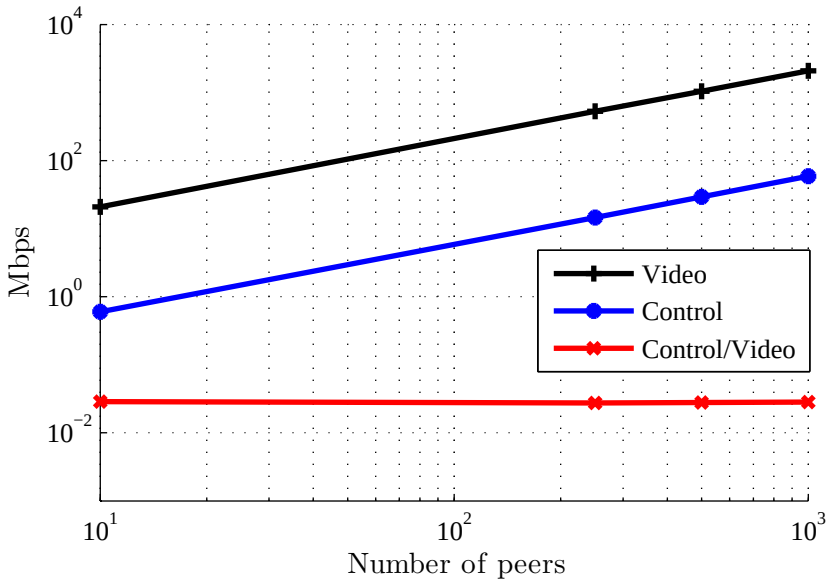


Figure 3.17. The aggregate traffic of the network for different number of participants in a multicast session with four trees and 270 kbps data rate.

control traffic only constitutes of two per cent of total video traffic, which is a very good ratio for a live streaming system showing that SPPM protocol exploits the regularity of tree based structure. The constant ratio between the control and video traffic shows the scalability of the system because it can maintain a highly populated structure with the same effort required to maintain a small group. According to the measurements, the hello and hello reply messages make up 99% of the total control traffic, which is a major fault which should be fixed. Adaptive timing strategies should be deployed to minimize the unnecessary hello messages generated for already stable connections.

The control traffic is probed for different number of trees to examine the most suitable tree number for a multicast session of 240 Kbps data rate. A total of 250 nodes are used throughout the simulations. In Figure 3.18, the statistics mentioned in the preceding paragraph are also provided for T = $\{1, 2, 4, 8\}$ where T is the number of trees. As T increases, the control traffic follows an increasing trend whereas the aggregate received video traffic remains nearly constant. However, this is not the case for single tree where the aggregate received video traffic is also lower than the other results. Therefore, we can conclude that the best performance can be achieved with a moderate number of trees.

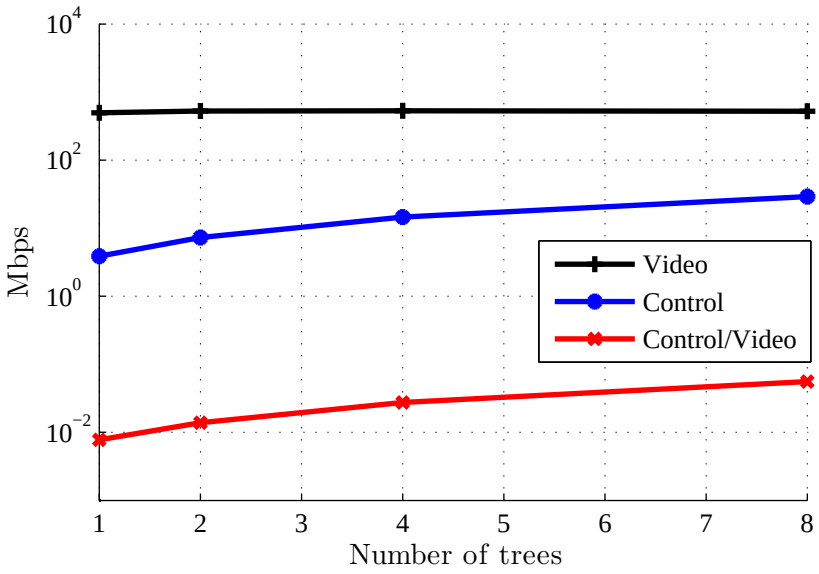
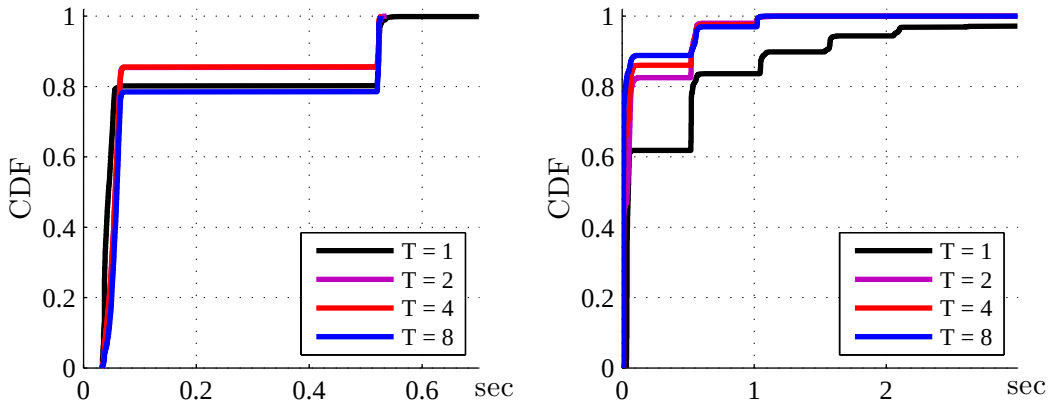


Figure 3.18. The aggregate traffic of the network for different number of trees in a multicast session with 250 participants and 270 kbps data rate.



(a) CDF of join durations.

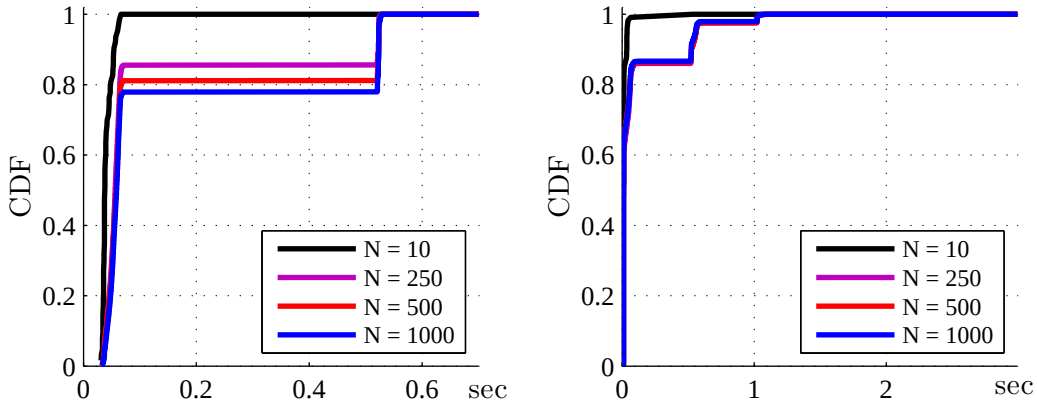
(b) CDF of rejoin durations.

Figure 3.19. Distribution of the start-up durations for different number of trees in a multicast session with 250 participants and 270 kbps data rate.

Since we are dealing with low latency systems, the start-up latency becomes much more stringent. In Figure 3.19, we have analyzed the latency of the protocol with respect to the number of trees used in the multicast session. The join duration represents the interval between the time when the application is launched and the peer creates a successful connection to at least one of the trees. On the other hand, rejoin duration is the time elapsed between a disconnection from a tree and its reestablishment. Simulated network consists 250 peers and the server node, and the CDF of the durations are plotted for different number of trees, denoted by T. For only one tree, the initial connection duration to the multicast tree is larger because the diversity cannot be exploited and the structure is more sensitive to individual peers constructing it. On the other hand, network performance degrades for very large number of trees, because the possibility of disconnections increase proportionally with tree number. We observed that for a moderate number of trees (two and four), the start-up latency of the system is smaller than 0.1 sec with a probability of 90%.

For rejoin procedure, the latency decreases with increasing number of trees as shown in Figure 3.19b. The reason for this fact is that, as the number of trees increases, there is a higher chance to recover a disconnected tree with the fast recovery mechanism executed in the REJOIN state of tree process. Although a very high number of trees seems reasonable from this point of view, this statistic does not provide the total number of disconnections, therefore it is not directly related to video quality. However, the advantage of the rejoining procedure is obvious for the multiple trees with respect to a single tree.

To provide a comprehensive view, the distributions of aforementioned latency measures are given for different number of participants in Figure 3.20. Since a high scalability is expected from a P2P system, the system latency should not be seriously affected from increasing number of participants. As shown in Figure 3.20a, a slight increase in latency occurs for a large viewer population. The basic reason is that the list size sent by server does not scale linearly with increasing number of viewers, to keep the probe messages at a certain level. The reduced options lower the chance of a successful parent selection. On the other side, we observed that the rejoin latency does not change



(a) CDF of join durations.

(b) CDF of rejoin durations.

Figure 3.20. Distribution of the start-up durations for different number of participants in a multicast session with four trees and 270 kbps data rate.

with varying number of receivers, except extremely low numbers. For very small numbers of peers, the P2P system's behavior converges to a server-client system. Therefore, we deduce that the main pattern behind rejoin latency is fast recovery mechanism.

The resiliency of the network structure is highly affected by the diameter of the network, i.e., the maximum hop number in the overlay. To reduce the effect of peer churn and also for lower end-to-end latencies, the network diameter should be small enough. We have collected the diameter information with a period of one second. The tree with the largest diameter is used for the statistics to illustrate the worst case scenario. In Figure 3.21, CDF of diameter is given for different number of participants, on a 240 Kbps multicast session disseminated through four trees. Although increasing diameter is inevitable for highly populated networks, a certain level of tolerability is expected from the structure. We use the tree with the maximum hop number, which is the worst case, for the statistics. For a large number of viewers, the network diameter is below 20 for most of the time, however the maximum hop number may reach to 30 for short durations and it should be avoided for real life scenarios.

The same statistics are also illustrated for different number of trees in Figure 3.22 simulated with 250 participants. The general trend of network diameter follows the number of trees. There are two main reasons behind this degradation. First, the

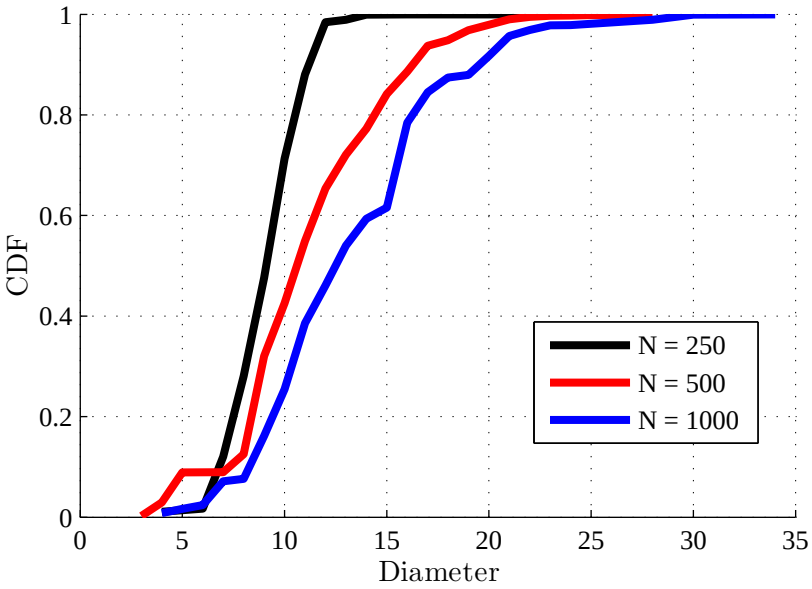


Figure 3.21. Distribution of the network diameter for different number of participants in a multicast session with four trees and 270 kbps data rate.

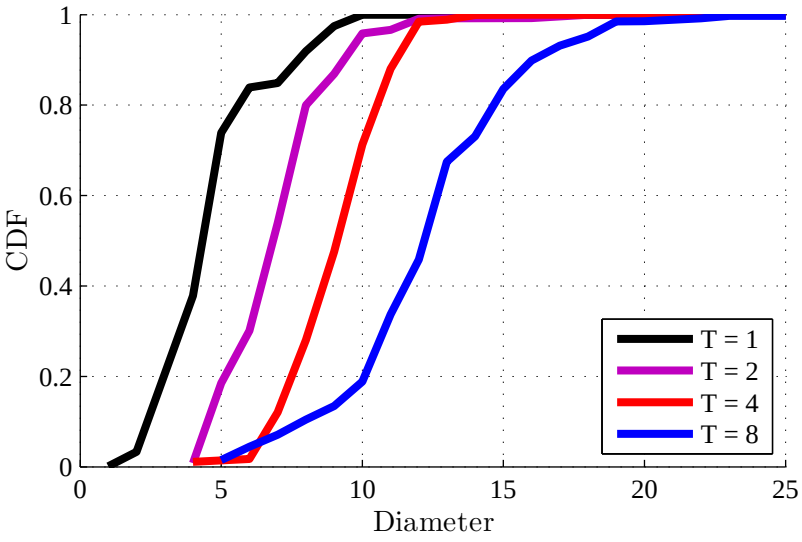


Figure 3.22. Distribution of the network diameter for different number of trees in a multicast session with 250 participants and 270 kbps data rate.

increased number of trees reduce the chance to minimize network diameter for all the trees at the same time, because the worst tree is regarded in the plot. The other reason is that, the JPS and SPS algorithms aim to provide path diversity for different trees, avoiding a small number of hops in exchange for parent diversity. This strategy may create dramatic problems for a large number of trees.

4. A BOTTOM-UP APPROACH TO TREE STRUCTURE

4.1. Motivation

One of the key concerns of P2P protocol designing is the overlay construction phase, hence the structure types used for P2P live multicasting networks were discussed in Section 2.3.3. Despite their advantages, the tree based overlays are disfavored due to several reasons. Although these disadvantages are actually affecting the network performance, one may seek possible solutions for these problems.

The fundamental concern has been the unused uplink throughput on the leaf-nodes because most of the peers are located at the network edge, similar to the trees in graph theory. Serving a very high number of peers with a fairly small population turns out to be very inefficient in terms of network capacity as more participants join the system.

To solve this problem, we propose a bottom-up approach instead of the traditional tree structure. It merges the advantages of tree based structures and push-based mesh systems. First of all, there is no pull requests in our overlay, which constitutes a high percentage of total traffic of mesh based systems. Since there are definite relations between the peers, i.e., parent-child relations, additional pull requests are not necessary.

The second power of this system is its robustness to peer churn. As we discussed in previous sections, tree based systems are extremely sensitive to peer churn because there is only one path from the server to each peer. Therefore, the video stream is interrupted whenever a parent on the path is disconnected from the system. On the other hand, the proposed approach dynamically reconstructs the overlay by minimizing the effect of an ascendant disconnection. This reconstruction is created by pushing approach and guarantees high resilience to peer churn without introducing very high redundancy as in push-based mesh systems.

In Figure 4.1a, an example tree based overlay is illustrated which includes 10 peers. After the *peer 2* is disconnected from the tree, the expected video dissemination

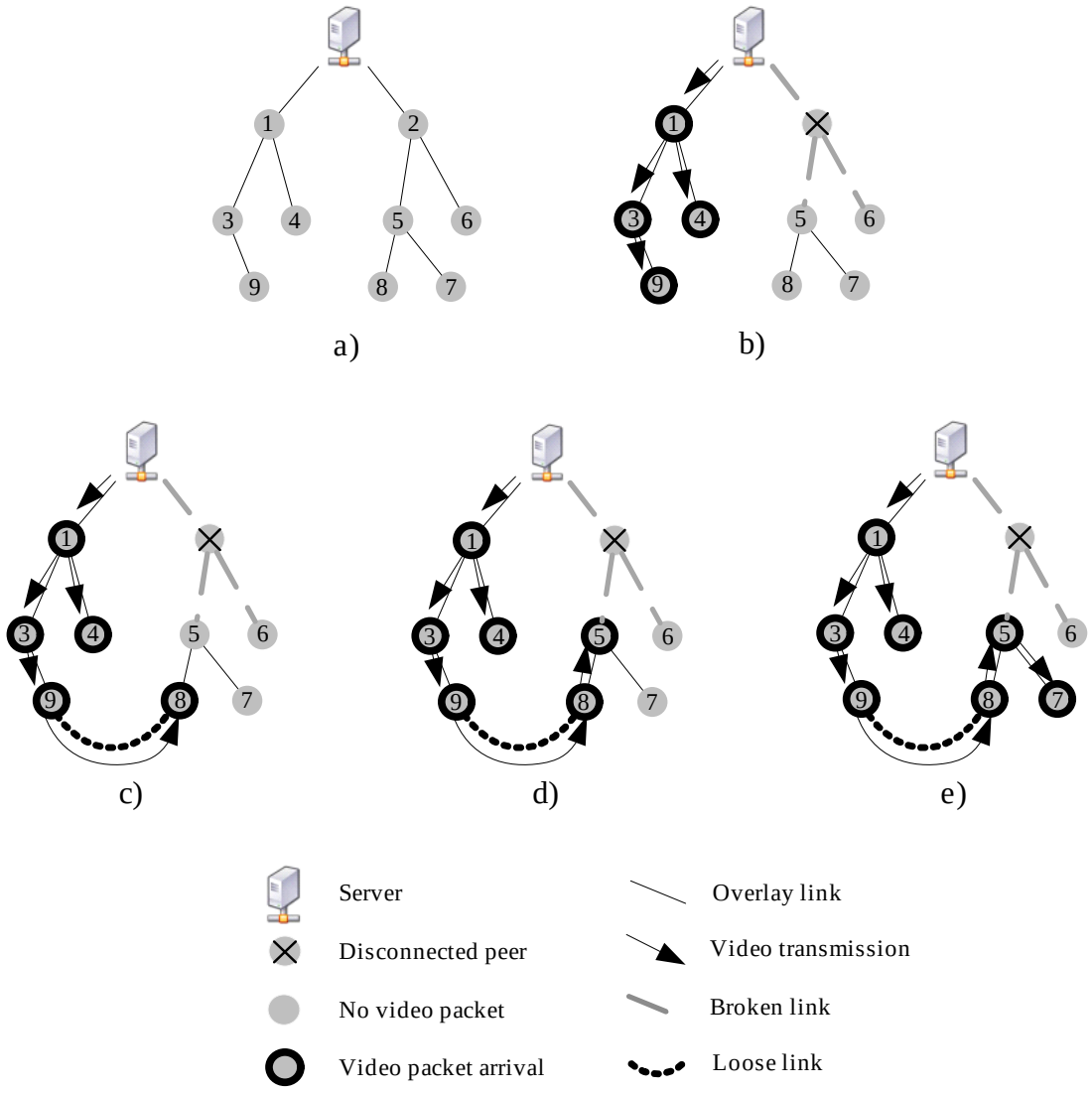


Figure 4.1. Video packet forwarding in proposed approach with a comparison to SPPM protocol with an illustration of loose links used in proposed method.

is illustrated in Figure 4.1b for standard SPPM protocol. Actually, this mechanism is valid for any basic tree based approach. For this example, the descendants of *peer 2*, namely *peers 5, 6, 7, 8* are unable to receive the video stream.

The proposed method basically includes creating loose connections between the leaf-peers which have enough uplink bandwidth. These peers are called *siblings*, and they support each other by sending the arrived video packets to the other one. In the example given in Figure 4.1c, a loose connection is created between *peers 9* and *8*, prior to any disconnections. For every packet arrival, *peer 9* sends a copy of the packet to its sibling *peer 8*, without any information of the packet availability on *peer 8*. If *peer 8* has already received the packet, it simply discards the same packet coming from its sibling peer. The advantage appears when an ascendant of *peer 8* is disconnected as in the case of Figure 4.1c. In this case, *peer 8* keeps the packets sent by *peer 9* because these packets are not available at *peer 8*. To disseminate the video packet through the defective regions of the overlay graph, it forwards the arrived packet to its parent as in Figure 4.1d. Then, *peer 5* receives the new packet and disseminates the packet to its parent and children. In Figure 4.1e, this forwarding mechanism is depicted, where *peer 5* sends the packet to its children, namely *peer 7* in our case. Since the parent of *peer 5* is disconnected from the multicast session, the data cannot be transmitted to *peer 6*.

By the help of loose connections, the defected parts of the overlay network can be recovered up to some extent in exchange for the uplink and downlink throughput of leaf peers. To create healthy links, only peers with available uplink bandwidth are considered as forwarding siblings. Since this throughput was already a waste from P2P point of view, using it is not a overhead for the system. However, the downloading capacity of receiving siblings introduce some overhead to the system because they have to download twice of the original video data rate. To avoid any congestions over these peers, the server assigns *forwarding siblings* to the peers having at least the double of the session data rate. According to Table 3.3, even the peers with minimum downlink bandwidth can afford such a redundancy for a session data rate of 250 Kbps.

Another expected benefit of this method is increased resiliency of structure to

the network congestions. In the case of a congestion on the path leading to a peer, assuming that the peer 5 in our example, a packet may not be received before its play-out deadline. The proposed method creates a chance for peer 5 to receive the video packet prior to its play-out deadline by letting the packet traverse an alternative path.

The method is very flexible and can be applied to any tree based protocol. In our testbed, we have used SPPM protocol as a tree based overlay protocol.

4.2. Video Transmission

For the video transmission part of the protocol, we have used video streams encoded with H.264 standard. We have used 30 minutes video streams. The decoded stream is disseminated from the server through the participants. The frames are periodically released from the server according to the FPS property of video stream. For example, the server sends a frame with a period of 0.033 second for a 30 FPS video sequence.

The server is responsible for packaging each frame prior to its transmission for providing a balance between multicast trees and preventing temporary data rate peaks to stabilize the system. Each frame exceeding the MTU size is fragmented into several packets, provided that the maximum packet size does not exceed MTU size. We assume that UDP is used in the transport layer and MTU size is 1500 bytes. Each packet of a fragmented frame is sent over different trees in sequence. Therefore, the k^{th} packet of the video sequence is disseminated over the tree T= $mod(k)$. We assume that network does not introduce bit errors. All packet losses occur due to packet drops and network congestions. A peer must successfully receive all packets belonging to a frame to be able to reassemble it.

Since the multi-tree structure brings along the path diversity, different packets belonging to the same frame may not arrive in order. Therefore, additional fields are required for the reassembly of the frame. Each video packet includes the fields in Table 4.1.

Table 4.1. Video packet structure.

Field	Explanation
Packet sequence number	The order of the packet within the overall video sequence
Video frame number	The order of the video frame that the packet belongs
Number of packets in the frame	This is used for reassembly of the video frame
Order of the packet in the frame	This is used for reassembly of the video frame
Play-out deadline of the packet	The last time that the video is allowed to be decoded

The received frames are buffered on the peers until they are played. Therefore, the buffer size depends on the data rate of the video stream and the play-out parameter. We assume that the memory size on a today's PC is large enough to buffer such an amount of video content.

Since decoding a real video stream over hundreds of peers through the simulations can be very expensive in terms of processing time, we have used video traces for our testbed. Although the video traces do not include the actual video content, the necessary performance information is embedded into the traces. The traces used for the simulations are extracted from [58] and the video content is a part of the movie Gandhi. The trace file basically includes the frame index, display time, frame type, frame length, and PSNR for each frame. In addition, these files also include the general statistics, such as average bit rate, maximum frame length etc., for each video trace published by the authors. Therefore, the video quality of overall network can be evaluated without requiring the actual video content.

We use video streams with the GOP number of 16. Although it is not mandatory, traditional IBBBPBBB... pattern is used for encoding of the video streams. I frames represent intra-coded frames which are not dependent on another frame for decoding. P frames are predicted from the prior I or P frame, depending on the position in the GOP. We also used bidirectionally predicted B frames, which rely on prior and posterior I/P

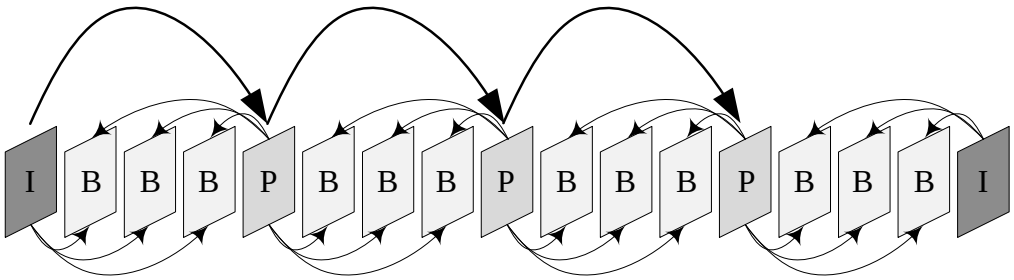


Figure 4.2. GOP pattern used throughout the simulations and prediction relations between the frames.

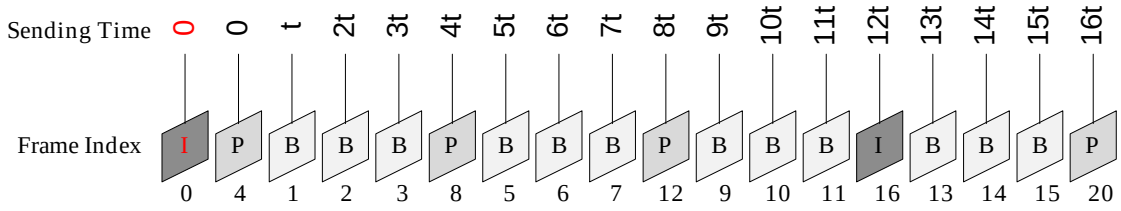
frames. Although I frames are standalone, meaning that they do not rely on any other frame to be decoded, the other types are affected by the availability of other frames. The dependency schema is given in Figure 4.2 for a 16 frame GOP with three B frames between I/P frames.

For the aforementioned pattern, the scheduling mechanism is illustrated in Figure 4.3 where t represents the interval between successive frames of the video sequence and p is the play-out latency of peers. In Figure 4.3a, frames are sent with a period of t. However, the frames are sent in an unordered fashion because some frames must be sent prior to the others for a successive decoding. For example, the B frames having the indices 1, 2, and 3 require the frame 4 (P frame) to be decoded because these B frames are predicted using previous I frame and posterior P frame. Therefore, the P frame with the index 4 should be sent in advance of the frames 1, 2, and 3.

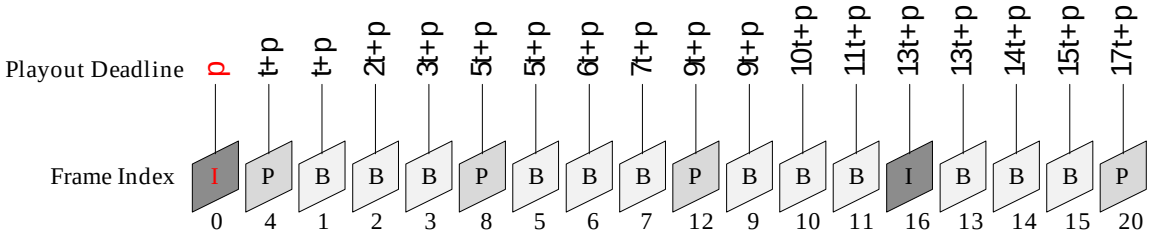
The play-out deadline of each frame is determined according to inter-frame dependencies. Although the frame 4 is sent at time 0, its actual play-out deadline is the same with the first frame depending on itself to be decoded. Therefore, the play-out deadline of frame 4 is set to the summation of t and p.

The arrived frames are viewed according to their decoding times. First, the received frames are ordered for a successful decoding. In this case, the frame 4 is lined up behind frames 1, 2 and 3 while the viewing. This scheduling is depicted in Figure 4.3c for a better clarification.

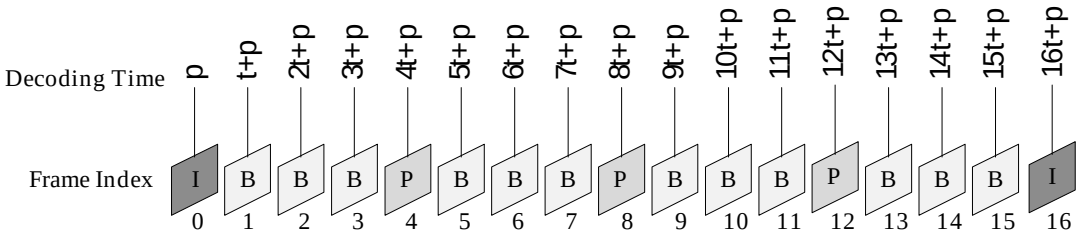
For the measurement of video quality, PSNR is used, which is computed from



(a) Releasing times of frames.



(b) Play-out deadlines of frames.



(c) Decoding order of frames.

Figure 4.3. The transmission and decoding scheduling for a low latency system.

RMSE value. These metrics are traditionally used as they provide a good reflection of perceived video quality. RMSE represents the distortion and it is the error between the pixels of the desired frame and the decoded frame. RMSE is calculated by:

$$RMSE = \sqrt{\frac{1}{XY}\sum_{0}^{X-1}\sum_{0}^{Y-1}(F(x,y) - f(x,y))^2} \tag{4.1}$$

where F is the desired frame, and f is the decoded frame. The difference between the individual luminance value of each pixel located at the position x and y is used as the error value. PSNR value of a frame can be easily calculated by using the RMSE value as

$$PSNR = 20\log_{10}\frac{255}{RMSE}. \tag{4.2}$$

The overall video quality is measured by the average PSNR of all frames in the stream. To increase the video quality at the end points, we replaced missing frames, which may not arrive before its play-out deadline due to uplink congestions, or packet drops, with the last successfully decoded frames. This error concealment mechanism is widely used in video applications. The last decoded frame is displayed until a new video frame is successfully decoded. We use error concealment for the previous 16 frames for a missing frame. If there is not a successfully decoded frame in the last 16 frames, the PSNR value is set to the minimum of last error concealment value and 20 dB.

4.3. Performance Evaluation

To evaluate the performance of proposed method, we carried out experiments for several protocol variables. The collected metrics are the PSNR of the video received by each participant, and the pause statistics of each peer. Although PSNR value is a good metric defining the video quality, video pauses are also very crucial in terms of user experience for a live system. Therefore, to provide an overall assessment, these metrics should be considered together.

To test the system's resilience to congestions, we used a network of 250 peers,

connected via links without packet losses. The default tree number is assumed to be four, unless otherwise stated. PSNR statistics of both original system (SPPM) and the proposed system are given in Table 4.2. For different data rates, our proposed method consistently performs better than the original method in terms of average PSNR of receivers. Although the average PSNR of the network increases with decreasing QP, this may not always indicate a good overall performance. The main reason behind this result is the effect of data rate on the streamed video quality as the PSNR value of a coded video is expected to increase with reducing QP level. On the other hand, the variance of PSNR values increases with decreasing QP level, showing an unstable overlay structure which does not fairly serve the content. With the increasing data rate, unfair distribution of the video gradually increases the total pause duration among the participants.

Along with a higher PSNR value, we also acquired lower pause durations for our proposed method compared to SPPM protocol. The high average value of pause duration is a result of effectively decreased total number of pauses in the session. This behavior shows our method's resiliency to congestions.

Up to now, we assumed a network without packet losses. However, this is not a realistic testbed for a live multicasting system using UDP/IP stack. UDP is not able to cover the defects caused by the best effort nature of IP protocol, hence it allows packet drops over the links, especially in the case of a congestion. The packet drops turn out to be an extremely serious problem in multi-hop networks, where the probability of not receiving a packet increases with the number of hops separating a peer from the source of the packet. In Table 4.4, the video qualities of both the proposed and the original methods are evaluated for lossy network conditions with a video stream of 270 Kbps. Our method constantly performs better than the original SPPM protocol, especially if there is a high packet loss probability. Even with a packet loss probability of two per cent, the video quality of the original protocol is nearly unacceptable. On the other hand, our system can achieve the same performance even with a eight per cent drop probability. Tree based structuring is an ill conditioned method for networks where there is a high packet probability because it prevents the chance of recovery for a

missing packet. However, our method resolves this problem up to a certain degree. As the drop probability increases, the variance of the PSNR value is obviously increases because network defects emerges in the overlay in spite of our protection mechanism.

The proposed method also achieves a good performance in continuous streaming. Although, average pause duration seems to be higher in the proposed system, it stems from the fact that, the total number of pauses per a peer is much more lower than the original system's result. The high number of short pause intervals frequently occurring in the original system simulates the effect of a lower average pause duration.

Another important parameter of the protocol is the number of trees used in the multicast session. The characteristics of the protocol depending on this variable are already discussed in previous sections. To provide a video quality perspective to the evaluation and to measure the response of our method to the number of trees, the experiments are carried out for different number of trees and results are given in Table 4.6 and Table 4.7. The performance results are worst for a multicast session with only one tree as expected, because only one tree cannot exploit the advantages of multi-tree based architectures. Still, the proposed method presents much better results than those of the original system. For high number of trees, our method consistently outperforms SPPM protocol, however, performance of the original method is also acceptable. Also, the low PSNR variance shows that our mechanism provides fairness regarding of the height in multicast tree, unlike traditional systems.

The total number of pauses in our system is nearly one third of the SPPM protocol, due to the elimination of short duration pauses. Although the average pause duration seems higher in our system, the total pause duration of the network is much more lower. The increasing number of trees in the multicast session brings over a high probability of a ascendant disconnection, which may result in a short or long term disconnection from the tree. The pause statistics of multicast session with the eight multicast trees reflects this effect in Table 4.7.

Lastly, the performance is measured for different play-out deadlines. The play-out deadline is the parameter of the multicast session which guarantees a live delivery

system. For a very stringent deadline, the overlay suffers from packet drops at the end users. On the other hand, the system goes further from being a live system as this parameter is more relaxed. The smallest value providing an acceptable quality level must be chosen as the system parameter. In Table 4.8, the PSNR values at different play-out deadlines are depicted. At very small values such as 0.5 sec, neither system is able to serve an acceptable video quality. As the play-out deadline relaxed, the PSNR value gradually increases until the point where system starts to saturate. It is obvious that there is not a large performance increase between two seconds and four seconds. For such a scenario, setting the deadline to two seconds results in a better live multicasting system.

Our method consistently performs better than the original method, providing a higher average PSNR and lower PSNR variance values. Although there is not much difference for lower deadlines, the total pause duration of the system decreases with relaxed deadlines. The reason behind this phenomenon is that, our mechanism does not provide a solution for optimizing end-to-end delays, which also suffers from stringent deadlines in turn. However, as the deadlines become more flexible, the proposed method tries to find alternative paths to reach every peer, even though it is not the shortest path to that peer. On the other hand, the original method still suffers from peer churn and it is not able to recover from this effect even with large deadlines.

Table 4.2. PSNR results of both proposed method and original method for different data rates.

QP / Rate (Kbps)	Proposed Method		SPPM	
	Average PSNR (dB)	PSNR Variance (dB)	Average PSNR (dB)	PSNR Variance (dB)
28 / 270	39.0825	1.1035	38.4272	1.5771
24 / 420	41.4256	1.1345	40.3614	2.6538
22 / 535	41.9935	2.6307	40.8382	2.9371

Table 4.3. Pause durations of both proposed method and original method for different data rates.

QP / Rate (Kbps)	Proposed Method			SPPM		
	n_p	d_{ave} (sec)	d_{max} (sec)	n_p	d_{ave} (sec)	d_{max} (sec)
28 / 270	26.2600	0.0064	1.7433	115.3320	0.0037	1.4959
24 / 420	36.8360	0.0096	1.5508	112.1320	0.0075	1.6047
22 / 535	86.5320	0.0129	2.1751	116.9560	0.0092	1.6697

Table 4.4. PSNR results of both proposed method and original method for different packet drop probabilities.

	Proposed Method		SPPM	
Drop Probability	Average PSNR (dB)	PSNR Variance (dB)	Average PSNR (dB)	PSNR Variance (dB)
0.00	39.0825	1.1035	38.4272	1.5771
0.02	36.4461	2.6445	30.5108	8.9343
0.04	34.0824	5.6345	25.9328	13.0917
0.06	31.8766	9.6640	23.7319	11.0704
0.08	30.3384	12.8840	21.7086	8.8018
0.10	28.8460	13.7638	20.7412	7.2701

Table 4.5. Pause durations of both proposed method and original method for different packet drop probabilities.

	Proposed Method			SPPM		
Drop Probability	n_p	d_{ave} (sec)	d_{max} (sec)	n_p	d_{ave} (sec)	d_{max} (sec)
0.00	26.26	0.0064	1.7433	115.3320	0.0037	1.4959
0.02	819.64	0.0016	1.6601	1825.9	0.0019	1.5119
0.04	1296.2	0.0016	1.5236	1769.3	0.0026	1.6425
0.06	1493.6	0.0019	1.7380	1518.8	0.0037	2.0932
0.08	1601.7	0.0022	1.4868	1096.1	0.0055	1.9951
0.10	1622.8	0.0024	1.6900	834.1240	0.0075	1.7273

Table 4.6. PSNR results of both proposed method and original method for different number of multicast trees.

The Number of Trees	Proposed Method		SPPM	
	Average PSNR (dB)	PSNR Variance (dB)	Average PSNR (dB)	PSNR Variance (dB)
1	33.9323	8.1896	28.0302	10.5564
2	39.1274	1.0922	38.8845	1.1560
4	39.0825	1.1035	38.4272	1.5771
8	38.8244	1.4674	38.0589	1.8693

Table 4.7. Pause durations of both proposed method and original method for different number of multicast trees.

The Number of Trees	Proposed Method			SPPM		
	n_p	d_{ave} (sec)	d_{max} (sec)	n_p	d_{ave} (sec)	d_{max} (sec)
1	14.2640	0.0896	6.6287	18.0560	0.1986	6.6287
2	18.7440	0.0090	1.2420	53.1320	0.0061	1.2185
4	26.2600	0.0064	1.7433	115.3320	0.0037	1.4959
8	63.2000	0.0038	1.3087	182.0960	0.0028	1.3657

Table 4.8. PSNR results of both proposed method and original method for different play-out deadlines.

Play-out Deadline (sec)	Proposed Method		SPPM	
	Average PSNR (dB)	PSNR Variance (dB)	Average PSNR (dB)	PSNR Variance (dB)
0.5	25.5494	4.4162	25.2042	4.7962
1	36.6522	3.5774	35.4191	4.0255
2	39.0825	1.1035	38.4272	1.5771
4	39.1867	1.0560	38.5340	1.5080

Table 4.9. Pause durations of both proposed method and original method for different play-out deadlines.

Play-out Deadline (sec)	Proposed Method			SPPM		
	n_p	d_{ave} (sec)	d_{max} (sec)	n_p	d_{ave} (sec)	d_{max} (sec)
0.5	1554.2	0.0039	1.8895	1488.5	0.0041	1.5497
1	650.0560	0.0041	1.7631	835.6080	0.0036	1.6041
2	26.2600	0.0064	1.7433	115.3320	0.0037	1.4959
4	24.8680	0.0018	0.0137	113.0160	0.0021	0.0943

5. CONCLUSION

In this thesis, a bottom-up approach to tree based multicast systems is proposed which can be applied to any tree based architecture. The main aim of this approach is to solve the problems stemming from the overlay structure type. One of the considerations for such systems is their lack of uplink efficiency, the leaf-peers do not contribute uplink bandwidth to the system. This effectively reduces the overall efficiency. On the other hand, since tree based overlays are extremely vulnerable to peer churn, alternative solutions must be produced to ensure the resistance of the system. However, the proposed solution should not bring along a heavy overhead and high redundancy.

We propose a method to solve the aforementioned problems. The method uses the highly structured nature of tree based architectures. In the proposed method, random loose connections are created between the leaf-nodes as new peers join the system. These loose connections are propagated through the lower heights as the number of peers increases. A peer sends each arriving packet to its sibling peer. Therefore, the uplink capacity of leaf-peers are efficiently used. The sibling peer discards packets that are already available. If the packet is not available at this peer, it sends the packet to its parent, therefore the packet is disseminated in a bottom-up fashion. By this approach, a high resilience can be achieved with only a small redundancy.

The performance of the proposed method is evaluated and compared to a standard tree based structure, namely the SPPM protocol. The traditional video quality metrics, which are PSNR and video pause duration, are used throughout the simulations. The protocol is extensively tested with by changing several parameters of the system, such as the number of trees, packet loss probability, video data rate and play-out deadline. In all of these tests, our method consistently achieved a higher performance compared to the SPPM protocol. It is shown that average PSNR increases and the variance in PSNR decreases with the use of the proposed extension. Besides, when the bottom-up approach is exploited, the number of pauses per peer also decreases, resulting in a high quality user experience. The gains are more pronounced for relaxed play-out deadlines and less for stringent requirements. Also, our gain may vary depending on the video

content streaming on the overlay.

Although the proposed method provides a high performing redundancy over the system, it still needs additional supporting methodologies. Since prediction based video coding is used in today's video transmission technologies, a retransmission algorithm relying on the importance of frames should be merged with the proposed solution. The packets belonging to the important frames from the coding view should be favored by the protocol.

Also, a further study may investigate the transmission of different rates over each tree to provide a wide range of video quality to the heterogeneous user population. Although the overlay construction techniques and protocols are widely examined, the combination of these protocols with alternative coding techniques such as multi-rate coding and multiple description coding are still in their infancy and will eventually lead to a better performance efficiently exploiting the network resources.

Another significant problem in P2P networks is the free-riding behavior of the participants. We have assumed a rational customer population throughout this study. However, users generally have an inherent disincentive towards cooperating to avoid consuming resources of them. This approach degrades a such system's performance eventually, relying on the voluntary contributions of the participants. To resolve this problem, there are many proposed incentive mechanisms in the literature [59]. These techniques should be combined with the existing spectrum of multicast protocols.

REFERENCES

1. Mervana, S., "Cisco Visual Networking Index: Forecast and Methodology, 2010–2015", CISCO, White Paper, 2011.

2. *Ipoque Internet Study 2008/2009*, http://www.ipoque.com, Accessed at June 2011.

3. Ihm, S. and V. S. Pai, "Towards Understanding Modern Web Traffic", *Proceedings of the ACM Joint International Conference on Measurement and Modeling of Computer Systems,* San Jose, California, USA, 2011, Vol. 1, pp. 143–144, 2011.

4. *Youtube*, http://www.youtube.com, Accessed at June 2011.

5. Setton, E. and B. Girod, *Peer-to-Peer Video Streaming*, Springer, 2007.

6. Ganjam, A. and H. Zhang, "Internet Multicast Video Delivery", *Proceedings of the IEEE*, Vol. 93, No. 1, pp. 159–170, 2005.

7. *Internet and Society Study*, http://www.bsos.umd.edu, Accessed at July 2011.

8. Deering, S. E., "Multicast Routing in Internetworks and Extended LANs", *ACM Computer Communication Review*, Vol. 18, No. 4, pp. 55–64, 1988.

9. Deering, S. E. and D. R. Cheriton, "Multicast Routing in Datagram Internetworks and Extended LANs", *ACM Transactions on Computer Systems*, Vol. 8, No. 2, pp. 85–110, 1990.

10. Moy, J., "Multicast Routing Extensions for OSPF", *Communications of the ACM*, Vol. 37, No. 8, pp. 61–66, 1994.

11. Paul, S., K. K. Sabnani., J. Lin and S. Bhattacharyya, "Reliable Multicast Transport Protocol (RMTP)", *IEEE Journal on Selected Areas in Communications*, Vol. 15, No. 3, pp. 407–421, 1997.

12. Floyd, S., V. Jacobson, C.-G. Liu, S. McCanne and L. Zhang, "A Reliable Multicast Framework for Light-weight Sessions and Application Level Framing", *IEEE/ACM Transactions on Networking*, Vol. 5, No. 6, pp. 784–803, 1997.

13. Casner, S. and S. Deering, "First IETF Internet Audiocast", *ACM Computer Communication Review*, Vol. 22, No. 3, pp. 92–97, 1992.

14. Thyagarajan, A. S., S. L. Casner and S. Deering, "Making the MBone Real", *Proceedings of the International Networking Conference,* Honolulu, Hawaii, 1995, Vol. 1, pp. 112–119, 1995.

15. McCanne, S. and V. Jacobson, "vic: A flexible Framework for Packet Video", *Proceedings of the ACM International Conference on Multimedia,* San Francisco, California, USA, 1995, Vol. 1, pp. 511–522, 1995.

16. Macedonia, M. R., M. Zyda, D. Pratt, P. Barham and S. Zeswitz, "NPSNET: A Network Software Architecture for Large Scale Virtual Environments", *Teleoperators and Virtual Environments*, Vol. 3, No. 4, pp. 265–287, 1994.

17. Chu, Y., R. Yang-hua, G. Sanjay and H. Zhang, "A Case for End System Multicast", *Proceedings of the ACM International Conference on Measurement and Modeling of Computer Systems,* Santa Clara, California, USA, 2000, Vol. 1, pp. 1–12, 2000.

18. Jannotti, J., D. K. Gifford, K. L. Johnson, M. F. Kaashoek and J. W. O'Toole, "Overcast: Reliable Multicasting with an Overlay Network", *Proceedings of the* 4^{th} *Symposium on Operating Systems Design and Implementation,* San Diego, California, USA, 2000, Vol. 1, pp. 197–212, 2000.

19. *Akamai*, http://www.akamai.com, Accessed at July 2011.

20. *Featured Articles from CNN*, http://articles.cnn.com, Accessed at July 2011.

21. Chu, Y.-H., A. Ganjam, T. S. E. Ng, S. G. Rao, K. Sripanidkulchai, J. Zhan and H. Zhang, "Early Experience with an Internet Broadcast System Based on Over-

lay Multicast", *Proceedings of the Usenix Annual Technical Conference,* Boston, Massachusetts, USA, 2004, Vol. 1, pp. 155–170, 2004.

22. *Napster*, http://www.napster.com, Accessed at July 2011.

23. *e-Mule*, http://www.emule-project.net, Accessed at July 2011.

24. *BitTorrent*, http://www.bittorrent.com, Accessed at July 2011.

25. Lua, E. K., J. Crowcroft, M. Pias, R. Sharma and S. Lim, "A Survey and Comparison of Peer-to-Peer Overlay Network Schemes", *IEEE Communications Surveys & Tutorials*, Vol. 7, No. 2, pp. 72–93, 2005.

26. Li, B. and H. Yin, "Peer-to-Peer Live Video Streaming on the Internet: Issues, Existing Approaches, and Challenges", *IEEE Communications Magazine*, Vol. 45, No. 6, pp. 94–99, 2007.

27. Setton, E., P. Baccichet and B. Girod, "Peer-to-Peer Live Multicast: A Video Perspective", *Proceedings of the IEEE*, Vol. 96, No. 1, pp. 25–38, 2008.

28. *Internet Filter Review*, http://www.internetfilterreview.com, Accessed at July 2011.

29. Ripeanu, M., A. Iamnitchi and I. Foster, "Mapping the Gnutella Network", *IEEE Internet Computing*, Vol. 6, No. 1, pp. 50–57, 2002.

30. *Skype*, www.skype.com, Accessed at July 2011.

31. Stoica, I., R. Morris, D. Karger, M. F. Kaashoek and H. Balakrishnan, "Chord: A Scalable Peer-to-Peer Lookup Service for Internet Applications", *Proceedings of the Conference on Applications, Technologies, Architectures, and Protocols for Computer Communications,* San Diego, California, USA, 2001, Vol. 1, pp. 149–160, 2001.

32. Chu, Y., S. Rao, S. Seshan and H. Zhang, "Enabling Conferencing Applications on

the Internet Using an Overlay Multicast Architecture", *ACM Computer Communication Review*, Vol. 31, No. 4, pp. 55–67, 2001.

33. Vlavianos, A., M. Iliofotou and M. Faloutsos, "BiToS: Enhancing BitTorrent for Supporting Streaming Applications", *Proceedings of the IEEE Global Internet Symposium,* Barcelona, Spain, 2006, Vol. 1, pp. 1–6, 2006.

34. Liu, J., S. G. Rao, B. Li and H. Zhang, "Opportunities and Challenges of Peer-to-Peer Internet Video Broadcast", *Proceedings of the IEEE*, Vol. 96, No. 1, pp. 11–24, 2008.

35. Tran, A. D., K. A. Hua and T. Do, "ZIGZAG: An Efficient Peer-to-Peer Scheme for Media Streaming", *Proceedings of the Annual Joint Conference of the IEEE Computer and Communications Societies,* San Francisco, California, USA, 2003, Vol. 1, pp. 1283–1292, 2003.

36. Castro, M., P. Druschel, A.-M. Kermarrec, A. Nandi, A. Rowstron and A. Singh, "SplitStream: High-Bandwidth Multicast in Cooperative Environments", *ACM Operating Systems Review*, Vol. 37, No. 5, pp. 298–313, 2003.

37. Padmanabhan, V. N., H. J. Wang and P. A. Chou, "Resilient Peer-to-Peer Streaming", *Proceedings of the IEEE International Conference on Network Protocols,* Atlanta, Georgia, USA, 2003, Vol. 1, pp. 16–27, 2003.

38. Eugster, P. T., R. Guerraoui, A.-M. Kermarrec and L. Massouli, "From Epidemics to Distributed Computing", *IEEE Computer*, Vol. 37, No. 5, pp. 60–67, 2004.

39. Liao, X., H. Jin, Y. Liu, L. M. Ni and D. Deng, "AnySee: Peer-to-Peer Live Streaming", *Proceedings of the IEEE International Conference on Computer Communications,* Barcelona, Spain, 2006, Vol. 1, pp. 1–10, 2006.

40. Kostić, D., A. Rodriguez, J. Albrecht and A. Vahdat, "Bullet: High Bandwidth Data Dissemination Using an Overlay Mesh", *ACM Operating Systems Review*, Vol. 37, No. 5, pp. 282–297, 2003.

41. Pai, V., P. Kumar, K. Tamilmani, V. Sambamurthy and A. E. Mohr, "Chainsaw: Eliminating Trees from Overlay Multicast", *Proceedings of the International Workshop on Peer-To-Peer Systems,* Ithaca, New York, USA, 2005, Vol. 3640, pp. 127–140, 2005.

42. Zhang, X., J. Liu, B. Li and T. P. Yum, "CoolStreaming/DONet: A Data-Driven Overlay Network for Peer-to-Peer Live Media Streaming", *Proceedings of the Annual Joint Conference of the IEEE Computer and Communications Societies,* Miami, Florida, USA, 2005, Vol. 1, pp. 2102–2111, 2005.

43. Liang, J. and K. Nahrstedt, "DagStream: Locality Aware and Failure Resilient Peer-to-Peer Streaming", *Proceedings of the SPIE Multimedia Computing and Networking Conference,* San Jose, California, USA, 2006, Vol. 1, pp. 23–31, 2006.

44. Zhang, M., L. Zhao, Y. Tang, J.-G. Luo and S.-Q. Yang, "Large-Scale Live Media Streaming Over Peer-to-Peer Networks Through Global Internet", *Proceedings of the ACM Workshop on Advances in Peer-to-Peer Multimedia Streaming,* Singapore, 2005, Vol. 1, pp. 21–28, 2005.

45. Magharei, N. and R. Rejaie, "PRIME: Peer-to-Peer Receiver-Driven Mesh-Based Streaming", *IEEE/ACM Transactions on Networking*, Vol. 17, No. 4, pp. 1052–1065, 2009.

46. Hefeeda, M., A. Habib, B. Botev, D. Xu and B. Bhargava, "PROMISE: Peer-to-Peer Media Streaming Using CollectCast", *Proceedings of the ACM International Conference on Multimedia,* Berkeley, California, USA, Vol. 1, pp. 45–54, 2003.

47. Venkataraman, V., K. Yoshida and P. Francis, "Chunkyspread: Heterogeneous Unstructured Tree-Based Peer-to-Peer Multicast", *Proceedings of the IEEE International Conference on Network Protocols,* Santa Barbara, California, USA, 2006, Vol. 1, pp. 2–11, 2006.

48. Wang, F., Y. Xiong and J. Liu, "mTreebone: A Hybrid Tree/Mesh Overlay for Application-Layer Live Video Multicast", *Proceedings of the International Confer-*

ence on Distributed Computing Systems, Toronto, Canada, 2007, Vol. 1, pp. 49–56, 2007.

49. *PP Stream*, http://www.ppstream.com, Accessed at July 2011.

50. *PPTV*, http://www.pplive.com, Accessed at July 2011.

51. *TvAnts*, http://www.cache3.tvants.com, Accessed at July 2011.

52. *TVU Networks*, http://www.tvunetworks.com, Accessed at July 2011.

53. *Zattoo*, http://www.zattoo.com, Accessed at July 2011.

54. *SopCast*, http://www.sopcast.com, Accessed at July 2011.

55. *Newest Export from China: Pirated Pay TV*, http://europe.wsj.com, Accessed at July 2011.

56. Sripanidkulchai, K., A. Ganjam, B. Maggs and H. Zhang, "The Feasibility of Supporting Large-Scale Live Streaming Applications with Dynamic Application End-Points", *Proceedings of ACM Conference on Data Communications,* Portland, Oregon, USA, 2004, Vol. 1, pp. 107–120, 2004.

57. Prasad, R., C. Dovrolis, M. Murray and K. Claffy, "Bandwidth Estimation: Metrics, Measurement Techniques, and Tools", *IEEE Network*, Vol. 17, No. 6, pp. 27–35, 2003.

58. Seeling, P. and M. Reisslein, "Video Transport Evaluation With H.264 Video Traces", *IEEE Communications Surveys and Tutorials*, submitted.

59. Habib, A. and J. Chuang, "Incentive Mechanism for Peer-to-Peer Media Streaming", *Proceedings of the IEEE International Workshop on Quality of Service,* Montreal, Canada, 2004, Vol. 1, pp. 171–180, 2004.