Università degli Studi di Ferrara

DOTTORATO DI RICERCA IN
SCIENZE DELL'INGEGNERIA

CICLO XXIV

COORDINATORE Prof. Stefano Trillo

MULTIMEDIA ON GEOGRAPHIC NETWORK

Settore Scientifico Disciplinare 09/F2

Dottorando
Dott. Merlanti Danilo

[illegible]

(firma)

Tutore
Prof. Mazzini Gianluca

[illegible]

(firma)

Anni 2009/2011

Abstract

In this thesis we investigate the topic of the multimedia contents distribution on a geographic network which is a rarefied and huge field. First of all we have to classify the main parts necessary in the multimedia distribution on a geographic network. The main aspects of a geographic network that will be highlighted in this thesis are: the mechanism used to retrieve the sources of the multimedia content; in the case of the peer-to-peer network on geographic network one of the most important mechanism is the query flooding protocol. The kind of overlay network (peer-to-peer) used to distribute the multimedia content. The usage of this overlay network in a multicast network. The security of the overlay network over a geographic network.

Therefore the first topic which is investigated in this thesis is the query flooding protocol that can be used in any kind of query operation on a peer-to-peer network. For this protocol we achieve an analytical model through a complex analysis of the proxies network. In this analysis we can see how the proxies permit an improvement in the performance with respect to the routing operations in a generic network of routers. Moreover we address a simple formulation and framework about the performance of the network with and without layer 7 (proxy) and we apply them in three different types of scenarios to show the advantages achieved with the usage of proxies instead of routers.

Through the query flooding operation, each peer of the peer-to-peer network can achieve the list of the peers that hold the desired multimedia content. In a multimedia content distribution system, after the previous step in which the list of the peers that hold the desired multimedia content is retrieved, it is necessary to establish the kind of peer-to-peer network used to distribute this multimedia content to the peers that require it. Therefore the second aspect analysed in this thesis, is how the peer-to-peer network is built so that it is possible to provide the multimedia content to the vast majority of peers (that require this content) with the minimum delay. The construction of the peer-to-peer networks used for the distribution of the multimedia contents is not a very investigated field. Thus in this thesis we produce new algorithms used to build peer-to-peer networks in an incremental way on asymmetric and radio channel and we establish which algorithm is better with respect to the maximum delay of the network, the maximization of the number of peers accepted in the network and the minimization of the bit error probability of each peer of the peer-to-peer network.

In this thesis, we propose an usage of the overlay network (peer-to-peer network) in a multicast network. We introduce an innovative mechanism that exploits the peer-to-peer network to make reliable a standard unreliable multicast network. Moreover we present an analytical model for this innovative mechanism.

Finally the last aspect of a geographic network is the security of the communications among a group of peers. Thus to ensure the maximum level of security with secure communications among a group of three or more peers, in this thesis we propose a new protocol, based on the Massey Omura protocol, which can allow the communications among the peers of a peer-to-peer network in a secure way. Moreover we present the security problems of this Massey Omura Multiple Users Protocol and how it is possible to avoid these issues through a specific encryption function and a specific decryption function by changing the encryption and decryption keys of each peer when the source peer changes. Finally we present a new cryptography protocol which we use to share the decryption shared key that is used in the Massey Omura Multiple Users Protocol.

Preface

The multimedia contents distribution on a geographic network is a rarefied and huge field. First of all we have to classify the main parts necessary in the multimedia distribution on a geographic network. The main aspects of a geographic network are:

- the mechanism used to retrieve the sources of the multimedia content.
- The kind of overlay network (peer-to-peer) used to distribute the multimedia content.
- The usage of this overlay network in different field of the telecommunication network.
- The security of the overlay network over a geographic network.

Thus in a multimedia content distribution peer-to-peer network on geographic network there are many aspect to consider. First of all there is a huge number of works about the protocol used to retrieve the sources of a content in a peer-to-peer network (query flooding protocol), therefore, it would simply not be practical to cite the "most relevant" contributions to a "field" so difficult to define. Anyway all the works, about the query flooding protocol, provide only mechanisms that can be used to optimize the query flooding protocol and reduce the effects of the flooding on the network without specifying a mathematical model (in terms of average access time) for the query flooding protocol. Some of this works are presented in the articles [1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14]. Thus it is important to investigate this topic so that it is possible to achieve an analytical model of the query flooding protocol that can be used in any kind of query operation.

Through the query flooding operation, each peer of the peer-to-peer network can achieve the list of the peers that hold the desired multimedia content. In a multimedia content distribution system, after the previous step in which the list of the peers that hold the desired multimedia content is retrieved, it is necessary to establish the kind of peer-to-peer network used to distribute this multimedia content to the peers that require it. Thus the second aspect of an overlay network (peer-to-peer network) on geographic network is how is built the peer-to-peer network so that it is possible to provide the multimedia content to the vast majority of peers (that require this content) with the minimum delay. The construction of the peer-to-peer networks used for the distribution of the multimedia contents is not a very investigated field. In fact the most important real-time algorithm used to create a simple distribution peer-to-peer network on asymmetric channels is given in article [15] and the issues of performance of peer-to-peer file sharing over asymmetric and wireless networks is addressed in article [16]. Therefore it is very important to analyse the state of the art [15] and generate new algorithms used to build multimedia content distribution peer-to- peer network on asymmetric channel and radio channel.

There are other types of networks which can be used to ditribute the multimedia content. The most popular of these kinds of networks is the multicast network. The most important problem of the multicast network is the reliability of the communications and this issue is solved using reliable multicast protocols. The reliable multicast is a very investigated field. In fact there are different mechanism used to have a reliable multicast network such as: peer-to-peer (P2P) tree based reliable multicast protocol [17], peer-to-peer epidemic algorithms for reliable multicasting [18], reliable peer-to-peer end system multicasting through replication in which it is presented an effective dynamic passive replication scheme designed to provide reliable multicast service in peer-cast [8], and many others [19, 20].

Therefore, it is interesting to see if there is an innovative mechanism that exploits the peer-to-peer network to make reliable a standard unreliable multicast network.

Finally the last aspect of a geographic network is the security of the communications among a group of peers. Thus it is important to investigate the different class of cryptosystems. The cryptosystems are classified in two kinds: the symmetric cryptosystems and the asymmetric cryptosystems. The symmetric cryptosystems use a single secret key shared among a group of users that want to communicate. On the other hand in the asymmetric cryptosystems there is a public key and a private key for each user. The public key is used to cipher the message and the private key is used to decrypt the message. In this case the users do not have to share a private secret key but the computational complexity of the asymmetric cryptosystems is much higher than the symmetric cryptosystem as shown in the article "Symmetric and Asymmetric Encryption" [21]. There is another group of cryptography protocols in which the users do not have to share a secret key. In this way the level of security is much higher than the previous crypography protocols. The number of studies about these protocols are very few and they only allow the communication between users for each execution of the protocol. Therefore these protocols do not allow the communications with one user who sends information to three or more users in a secure way. Examples of these protocols are Shamir's 3-pass protocol [22, 23], Khayat's protocol [24], Massey Omura [25] and other 3-pass protocols like [26]. Thus to ensure the maximum level of security with secure communications among a group of three or more peers it is necessary to create a new protocol, based for example on the Massey Omura protocol, which can allow the communications among the peers of a peer-to-peer network in a secure way.

Acknowledgement

After three years of hard work I have to thank my supervisor, called professor Gianluca Mazzini who gave me the possibility to follow the PhD course in Science of Engineering at the University of Ferrara. I would also like to thank the professor Velio Tralli because he often gave me good advices regarding the PhD. I would also like to thank my best friend and collegue here at the University of Ferrara, Abdul Haseeb who helped me a lot in good and in the bad situations that occurred during my life, as he was a good source of inspiration and encouragement. I would also like to thank my family for their support. And finally I would like to thank my “best English supervisor professor” Titani Nyirenda (my wife) who worked timelessly at correcting my English in this thesis and many of my articles during my postgraduate.

Contents

List of Tables

List of Figures

Chapter 1

Introduction

A geographic network is a Wide Area Network (WAN). To introduce the concept of “Wide Area Network” we have to analyse briefly one kind of network called “Local Area Network” (LAN). The Local Area Network (LAN) stems from the need to interconnect the computers that belong to the same company so that it is possible to share all the resources of the network among all the users of the network. In this case the area in which the computer are interconnected is a limited area such as a school, a computer laboratory, an office building or a company. In a LAN the communication is highly structured it means that has been proposed different protocols for the communications in a Local Area Network. Examples of protocols for the Local Area Network are: IEEE 802.3 (Ethernet) [27], IEEE 802.4 (Token Bus) [28], IEEE 802.5 (Token Ring) [29] and IEEE 802.11 (Wireless LAN or WLAN) [30]. In the case of the Local Area Network the dimensions of this network are between 1 meter to 1 Km and the speed of this network is between 1Mbit/s to 10 Gbit/s.

A geographic network or a Wide Area Network is an extension of the Local Area Network and its dimensions is national or international this means that a wide area network (WAN) is a telecommunication network that covers a broad area. Therefore the Wide area networks (WANs) are used to connect local area networks (LANs) together, so that users and computers in one location can communicate with users and computers in other locations. Moreover a wide area network (WAN) is a network connecting geographically distinct locations, which may or may not belong to the same organization. WAN topologies use both LAN add enterprise-wide topologies as building blocks, but add more complexity because of the distance they must cover, the larger number of users they serve, and the heavy traffic they often handle. Thus the Wide Area Network covers the whole world. In a Wide Area Network the speed of the the links are extremely variable since they are between 64 Kbit/s and 10 Tbit/s. This high speed variability is due to the interconnection technology used (optical fibers, satellite, cables). Even the dimensions of Wide Area Networks are extremely variable and these dimensions are from 1 Km to more than 1000 Km. An example of Wide Area Network (geographic network) is the Internet Network (figure 1.1) and this network is achieved by interconnecting a large number of independent WAN. In a peer-to-peer network the peers are not necessary located in the same Local Area Network but these peers can be located in the same country or in different countries. Thus the peer-to-peer network is an overly network that is built on a Wide Area Network because in this kind of network the computers are located inside of a country or in different countries. Therefore we have to investigate all the aspect of a peer-to-peer network starting from the bases of the peer-to-peer networks. Then we have to investigate the different approaches for the peer-to-peer networks which represent the state of the art. These approaches are used to make peer-to-peer networks used for the distribution of a generic content and they are not used for the distribution of multimedia contents since they don't consider the delays constraints which are important to ensure the best service for all the peers of the network in the case of multimedia content distribution. Anyway these approaches are very important because they represent the bases for the peer-to-peer network used for the multimedia contents distribution. At the end of this introduction, we present a recently peer-to-peer architecture used for the live streaming TV content distribution [31, 32].

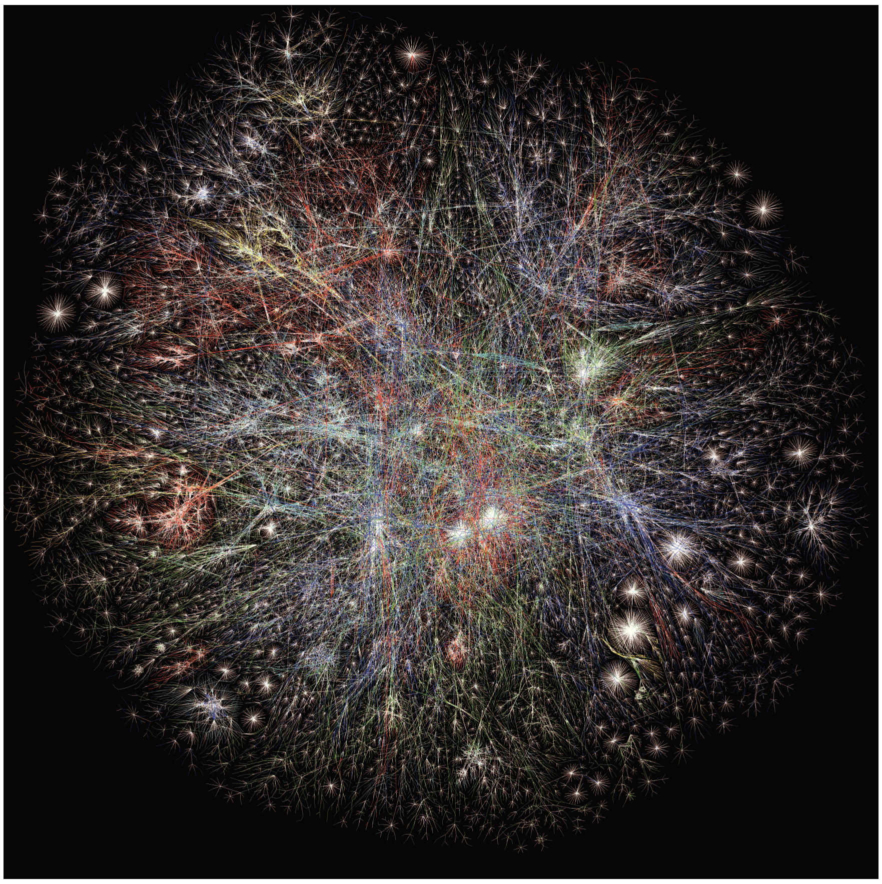

Figure 1.1: Internet Graph

1.1 Communication models: “peer-to-peer bases”

1.1.1 Client-Server Model

The client-server model involves the presence of two entities:

1. the client entities: which is an application that allows a user to access to a server machine (remote) that provides a certain service. To do this the client makes a request of an server application which is resident on the server machine.

2. The server entity: it is an application that waits the arrival of service requests from client applications. When the server receives a service request it satisfies the request by supplying the requested service to the requesting client.

An example of client-server communication is represented by the request of a web page done by a web client to a web server.

1.1.2 Main aspects of the clien-server Model

The client-server model uses an asymmetric way of communication, because the client explicitly designates the recipient server via the IP address of the server and its port, namely when a client has to request a service to a server the client must know the address of the server.

The server does not identify the client, namely the server machine, but instead by definition, waits to receive the service requests and when the server receives a request it replies by providing a particular service to the client. Therefore the server does not have the address since it is waiting for a client service request.

Moreover the kind of communication is many to one, thus indicating that many clients can request the same service from only one server.

1.1.3 Other features of the client-server model: "push and pull"

The client-server model can operate in a push or a pull mode namely:

- pull mode: in this case, the client retrieves the service from the server system namely the client sends a service request to the server system with which picks up directly from the server the requested service.
- Push mode: in this case, the server takes care of sending the response packet to the client and the server decides autonomously when it is necessary to send to the client

the information that the client has requested to the server. So in this case the server, when it is appropriate, transmits the requested information to the client connected to the server. Thus when the client terminates the connection with the server, the server is no longer able to send the data previously requested by the client.

1.1.4 Client-Server Model: “Interaction between processes”

The client-server model determines when it has the interaction between the client process and server process. The communication between the client process and the server process, in general, is subject to the problem of the rendez-vous namely the problem to determine when it is necessary to synchronize the communicating processes (i.e. when the meeting between the client process and the server process has to happen). The client-server model solves the rendez-vous problem by defining the server as a process which is waiting server requests from the client processes.

Therefore, in the case of the client-server model, the synchronization between the client process and the server process occurs with the following type of model: when a client performs a service request to the server, the server (which was waiting for a server request) processes the request and it sends back the results to the client.

The solution of the rendez-vous problem is achieved by using a server process which is executed during the boot time of the server machine. This server process waits for the service requests made by the clients.

1.1.5 Other models

The client-server model is not the only model of communication. Other existing models are the following:

- peer-to-peer model: in this case there is an interaction between peer processes (peers) and thus each entity can be both client and server.

- models event-driven and Publish-Subscribe: in this case the server takes the initiative and it signals the occurrence of an event. These models are based on the Push models.

1.1.6 Peer-to-Peer Model

In the client-server model, the server can represent a bottleneck. The peer-to-peer model avoids the centralized point represented by the server because in the peer-to-peer model the communications are among peers and therefore all the processes are at the same level from the point of view of the protocols and the responsibility. Thus each process can play simultaneously the role of client and the role of server.

In this way there is not anymore the centralized unit represented by the server and thus each peer can exchange files with the other peers without the bottleneck problem of the client-server model. The application targets (goals) of this system are: high availability and high degree of replication.

1.1.6.1 How the peer-to-peer model works

Typically, it is necessary two steps to get a file, in a peer-to-peer system:

1. the first step is the localization. It is performed to locate the interested file. This is done by searching the interested file in all the possible peers available at that time (this topic is very complex and different architectural solutions are possible).

2. The second step is the file transfer. Once it is located the peer or the peers which have the desired file (through the operation of localization), we know the IP addresses of these peers and therefore we can start the file transfer. In this situation the peers, which hold the file, send the desired file to the requesting peer using a direct communication at level 4 (TCP) of the stack OSI.

1.1.6.2 Peer-to-Peer model: contents localizations

The localization methods, used for locating the peers from which we can get the desired file (or content), follow different architectural models which have different degree of distribution:

- in the case of Napster [33] we have a centralized directory. This directory is held in a centralized server which holds the IDs of the files owned by the known users (peers).
- In the case of KaZaA [34] we have a decentralized directory system.
- In the case of Gnutella [35, 36, 37] we have a process used to locate the peers called query flooding.
- In the case of the Kad network (Kademila [38]), uses a distributed hash tables to index files and for keyword searches.

These systems provide several advantages and disadvantages from different points of view:

- fault tolerance.
- Performance.
- Legal responsibilities.

1.1.6.3 Peer-to-Peer model with centralized directory

The peer-to-peer model with centralized directory, such as Napster [33], is characterized by a localization system, which is used to localize the desired file, with a centralized directory which holds the information about the available files and the peers that have those files. Therefore when a peer looks for a specific file, this peer has to connect to the centralized directory system and requests the localization of the file. In this way if the centralized directory finds one or more peers which hold the desired file then the centralized directory provides the IP addresses of the peers which have the desired file to the requesting peer. In this way the requesting peer can get the file directly from the peers that have it.

1.1.6.4 Peer-to-Peer model with decentralized directory

In the case of the peer-to-peer model with decentralized directory (such as KaZaA [34]), the directory functions are directly distributed to some peers of the peer-to-peer network and these peers are called group leaders. Each other peer connects to a group leader and uses it to search the desired file.

Therefore the peer leader of a group holds the directory in which there is the information about the available files and the peers of the group which have those files. In this way, when a peer requests a file, this peer makes a query about the desired file to the peer leader of the group.

If the peer leader knows a peer (of the group) which has the desired file then the peer leader sends to the requesting peer the IP address of the peer which has that file. In this way the requesting peer performs the acquisition of the file from the specified peer of the same group.

If instead the peer leader does not find a peer (of the same group) which has the desired file then the peer leader sends a query, about this file, to another peer leader of a different group thus obtaining the required information. When the peer leader receives from another peer leader the IP address of a peer which has the desired file, then the peer leader sends this IP address to the requesting peer. In this way the requesting peer performs the acquisition of the file from the specified peer of a different group.

When a peer has access to the distributed system, this peer connects to any leader of a group. Thus in this peer-to-peer model there are bootstrap nodes used for the following points below:

- indicate the peer leaders to the peers that want to connect to the peer-to-peer network.

- The selection of the peer leaders. For example if a peer leader goes offline then the bootstrap node designates a normal peer of the group as the peer leader.

1.1.6.5 Peer-to-Peer model with query flooding

The peer-to-peer model with query flooding is the model used in Gnutella system [35, 36, 37]. In this case we do not have a decentralized system based on a decentralized directory system used to find the IP addresses of the users (peers) which have the desired file. The main characteristics of the Gnutella protocol are the following:

- when a peer performs a request research for a particular file, the peer sends this request to all the neighbor peers and they sends this request message their neighbor peers.
- In this way for the localization of the file requires a broadcast operation of the request message. This broadcast operation is called flooding. Therefore the file request propagates to all the peers of the system.
- When the system finds a peer that has the desired file this peer sends back a message to the requesting peer. This message has the specified IP address of the peer which has the file. In this way the requesting peer can get the desired file directly from the peer which has the desired file with a direct connection between them.
- Since the research message does not flood all the peers of the network (where this flooding can generate a high research time and a blockage of the network), the maximum number of hops after which the request message is dropped is arbitrarily set.
- In this protocol there is a bootstrap node. When a new peer N wants to connect to the peer-to-peer network, the bootstrap node sends the list of peers closest to the new peer to N. In this way the new peer connects to the peers specified in the list of peers sent by the bootstrap node.

- If a peer *s* goes offline, there are specific protocols used to update the network so that the peers, connected to the peer *s*, can connect to another active peer.

1.1.6.6 Fault tolerance of these models

In the peer-to-peer model with centralized directory, the level of fault tolerance is low because there is only one server which contains the information about the available files and the peers that have those files. In this case if the server goes offline then the entire peer-to-peer network is offline because the peers cannot localize the peer which has the desired content.

In the peer-to-peer model with decentralized directory, the level of fault tolerance is good because if a peer leader of a group goes offline then the bootstrap node reorganizes the network and it defines (establishes) a new peer leader and in this way the bootstrap connects all the peers of the group to the new peer leader.

In the peer-to-peer model with query flooding, the level of fault tolerance is good because if the neighbor of a peer goes offline then there are specific protocol used to connect the considered peer to a new active neighbor.

1.1.6.7 Performance

In the peer-to-peer networks, the performance of the system depends on the research phase (localization) of the desired file, because in all the peer-to-peer models the transfer of the file is performed directly between the requesting peer and the peer which has the desired file. Therefore in the case of the peer-to-peer model with centralized directory, the localization phase is much faster than the other two models of peer-to-peer.

1.1.6.8 Peer-to-Peer Systems

The most important peer-to-peer system at the moment are: Napster [33], Gnutella[35, 36, 37], KaZaA [34] and Kademila [38]. These peer-to-peer systems allow the file sharing through an architecture characterized by increasing degrees of distribution.

The peer-to-peer networks are the largest contributor traffic on the Internet, therefore the peer-to-peer is the cause of the traffic increasing on the Internet. In this study it is presented that the peer-to-peer networks still produce most Internet traffic worldwide although its proportion has declined across all monitored regions – loosing users to file hosting and media streaming; regional variations in application usage are very prominent; and Web traffic has made its comeback due to the popularity of file hosting, social networking sites and the growing media richness of Web pages [39].

There are many other peer-to-peer systems such as:

- JXTA [40, 41]: SUN environment used to develop peer-to-peer applications in Java.
- Freenet [42]: filesystem with high degree of replication and dissemination resulting from the demands. Propagation strategies inspired by the global routing. System of names based on indexes (via integrity hash).
- Freeheaven [43, 44]: the Free Haven Project aims to deploy a system for distributed, anonymous, persistent data storage which is robust against attempts by powerful adversaries to find and destroy any stored data. This model of decentralized system has been classified as peer-to-peer by recent popular media. In this system there are more information divided by strength and with a high degree of security.
- Publius [45]: sharing of information with different degrees of security and privacy.
- SETI@home [46, 47, 48]: peer-to-peer system used to process and analyse the signals transmitted by the space. This system controls the pattern of such signals to verify whether they are originated from an extraterrestrial intelligence. This research and analysis is performed by distributed nodes (peers). In this case when a peer connects to the peer-to-peer network this peer receives a portion of data, from the central station, and processes it. The received data is processed by each peer during the

idle CPU cycles. When the data processing is completed, the peer sends back to the central station the results. In this way the computational load is distributed among the computers belonging to peer-to-peer the network.

- Jabber/XMPP [49, 50, 51]: support for conversation and cooperation between users.
- Skype [52] is a hybrid peer-to-peer and client–server system, and makes use of background processing on computers running Skype software. The service allows users to communicate with peers by voice, video, and instant messaging over the Internet. Phone calls may be placed to recipients on the traditional telephone networks.

1.2 From the client-server model to the peer-to-peer model

1.2.1 Issues of the client-server model

To understand the issues of the client-server model we consider the following scenario: we have a server which has to send a big file to N different clients. We have the following hypotheses:

- the dimension of the file is equal to F bit.
- we have only one server which has an upload rate equal to u_S.
- for $i = 1...N$, the download rate of the i-th receiver (client) is equal to d_i.

In this scenario the server has to transmit the file to N different receivers and therefore the server has to transmit a total number of bits equal to $N \cdot F$. This operation requires a total time greater of equal to $N \cdot F/u_S$.

But on the side of the receivers, we have to consider that:

- the slowest receiver receives the file at rate equal to: $d_{min} = min_i \{d_i\}$.
- Therefore the time necessary to receive the entire file is greater or equal to F/d_{min}.

Thus the time used by the N receivers to receive the entire file (which has dimension F) is equal to the maximum between the time used by the server to transmit the file to the N receivers ($N \cdot F/u_S$) and the time used by the slowest receiver to get the file (F/d_{min}).

Therefore, in this case, we can speed up the file distribution increasing the upload rate of the server. We can achieve this solution by:

- increasing the link bandwidth at the server.
- Using multiple servers which transmit the same file. In this case each server has an own link used to distribute the file.

The main problem of this solution is that it requires the deploying of more infrastructures.

An alternative solution, in which we don't increase the upload bandwidth of the source node, it is based on the usage of the receivers. In this case we use the receivers to distribute the file to the other receivers using their upload bandwidth:

- the receivers get a copy of the data related to the file.
- Afterward the receivers redistribute the data to other receivers using their available upload bandwidths.

In this way it is reduced the burden on the server namely we reduce the requests (related to the file) received by the server.

1.2.2 Peers Help Distributing a Large File

In this case the system starts with a single copy of a large file:

- we suppose to have a large file with size of F bits and we suppose that the upload bandwidth of the server is equal to u_S.
- We suppose that each i-th peer is characterized by a download bandwidth equal to d_i and an upload bandwidth equal to u_i.

In this case we have two components of the distribution latency:

- the server has to sent each bit of the file to a peer. The minimum time required for the entire transmission is equal to F/u_S.
- The slowest peer receives each bit of the file and the minimum time required to receive the entire file is equal to F/d_{min}.

We can use the previous considerations to achieve the total upload time using all upload resources given by the upload bandwidth of the server and the upload bandwidths of all the peers of the network:

- the total number of bits, which have to be delivered to the N receivers, is equal to $N \cdot F$.
- The total upload bandwidth of the network is given by the upload bandwidth of the server + the upload bandwidths of the peers of the network, namely: $u_S + \sum_i u_i$
- Therefore the total time necessary to distribute the file to the N peers of the network is: $max\left\{\frac{F}{u_s}, \frac{F}{d_{min}}, \frac{N \cdot F}{u_s + \sum_i u_i}\right\}$

1.2.3 Comparison between the client-server model and the peer-to-peer model

In the case of the client-server model, the time required to transfer the file to N clients is:

$$max\left\{\frac{N \cdot F}{u_s}, \frac{F}{d_{min}}\right\} \tag{1.1}$$

In the case of the peer-to-peer model, he time required to transfer the file to N clients is:

$$max\left\{\frac{F}{u_s}, \frac{F}{d_{min}}, \frac{N \cdot F}{u_s + \sum_i u_i}\right\} \tag{1.2}$$

From these formulation, we can see that the peer-to-peer model is self-scaling:

- we have much lower demands on server bandwidth.
- The distribution time grows only slowly with the number of peers N.

But in the peer-to-peer model there are some difficulties such as:

- the peers may easily come and go from the peer-to-peer network because the peers are intermittently connected to the network. The peers may come and go at any time. Moreover the peers can come back with a different IP address if the IP address is assigned to the peers in a dynamic way.
- The peers need to find each other to allow the file sharing. Thus the main problem is how to locate the relevant peers, namely it is necessary to:
  - discover the peers that are online right now in order to ask if they have the desired file.
  - Discover the peers that have the desired content or file.
- The peers need to be willing to help each other. Moreover it is necessary to find a way to motivate the peers to stay in the system, namely:
  - it is necessary to find a motivation which brings the peer to do not leave the system as soon as the download ends.

  - it is necessary to find a motivation which brings the peer to do not be worry to upload and share contents to the other peers.

1.2.4 Introduction to the types of peer-to-peer networks

The main approaches used in the peer-to-peer networks are the following:

- Centralized Directory Model (CDM), such as: Napster [33] and BitTorrent [53].
- Flooded Requests Model (FRM), such as Gnutella [35, 36, 37].
- Selective Queries Model (SQM) (it is even called hierarchical overlay), such as KaZaA [34].

In the following subsections we present the descriptions of these approaches.

1.2.5 Napster

Napster [33] is based on a directory service technology: the user downloads the client software then the user installs the software. The user signs up in the system using the username and the password and specifying the local directory used to share the shared files and to keep the files downloaded from other Napster's users. The client contacts one of the servers of Napster, via TCP protocol, and the client provides the list of music files it will share. In this way Napster's central server updates the directory, with the file list received from the client.

The client uses the directory service to search the desired file by specifying the title of the song of the name of the singer. The client makes this request to the central directory server. The server identifies the online clients with the desired file and provides the IP addresses of these clients to the requesting client. In this way the client gets the list of the IP addresses of the client with the desired file. Afterwards the client requests the required file from the chosen suppliers. These suppliers transmit the file to the client and both client and suppliers report the status of this operation (transaction) to Napster. Therefore in this scenario each peer acts as a server when it sends a file to one or more other peers and the peer acts as a client when it downloads a file from one or more suppliers (peers). Therefore the file transfer is done by a peer-to-peer network.

1.2.5.1 Napster technology: Properties

The server's directory is continually updated with the information related to the file shared by the clients. In this way it is always known which file is currently available. In this system we have a centralized directory server which contains information related to the shared files and the peers that share these files. This mechanism represents a point of vulnerability for legal actions.

During the file transfer, there is a peer (supplier) that sends the desired file to another peer, that in this case acts as a client. Therefore, the file transfer is done by a peer-to-peer mechanism and in this way there is no load on the central server. There is the plausible deniability for legal action, with reference to the mechanism that represents a point of vulnerability for legal action introduced above, but this is not adequate because the file transfer can be traced by observing the communications received by the server from the peers.

Napster uses a proprietary protocol for the following functions: login, files searching, files uploading, files downloading and status operations. In this protocol there is a very low security system because the passwords are transmitted in clear text and there are other system vulnerabilities.

In the system used by Napster there are bandwidth issues because the suppliers (peers that share files) are ranked by their apparent bandwidth and their estimated response time.

1.2.5.2 Napster: limitation of central directory system

The limitations of the system used by Napster are the following:

- there is a single point of failure represented by the central server and therefore the system has a low failure tolerance. This situation is due to the fact that the file transfer is decentralized but the locating of the contents is highly centralized.
- There is a performance bottleneck due to the central server because the file transfer between peers is completely decentralized, but the localization of contents is strongly centralized because there is a central directory service used to locate contents.
- There is a central directory system used to locate the contents and therefore the copyright infringement laws are easily identified by analysing the file managed by the directory service.

Therefore due to these limitations, the recent peer-to-peer systems are more distributed among the peers and some systems, such as Gnutella [35, 36, 37], are completely distributed without any centralized unit.

1.2.6 Gnutella

Gnutella [35, 36, 37] uses a query flooding protocol which provides the following operations:

- Join; the peer uses this operation to contact a few nodes to become neighbors. This is made possible through a bootstrap node. The aim of the bootstrap node is to specify the neighbors to the considered peer.
- Publish; in this protocol, there is no need to publish the shared files of each peer (user) since there is no centralized directory.
- Search; when a peer searches a file this peer asks the neighbors, who ask their neighbors. Each time a new peer, which holds the desired file, is found this peer sends its IP address to the peer which has generated this search.

- Fetch; the requesting peer gets the file directly from another peer which holds the desired file.

Therefore, Gnutella provides a fully distributed system in which there is no central server, using a public domain protocol and there are many Gnutella clients that implement this protocol.

In the Gnutella peer-to-peer network, there is an overlay network that consists of a graph in which each node corresponds to a participant of the peer-peer network. In this overlay network there is a TCP connection between nodes, if these nodes know each other. If a peer knows the localization of another peer of the peer-to-peer network, then there is an edge between the first peer (node) and the second peer (node) in the overlay network. Therefore there is an edge between the peer X and the peer Y if there is a TCP connection between them. All the active peers and all the edges make the overlay network. If we consider a peer, this peer will typically be connected with a number of overlay neighbors less than 10.

1.2.6.1 Gnutella: Query Flooding Protocol

The query flooding protocol is used to search a file (content) in the Gnutella peer-to-peer network. When a node X requests a specific file to the peer-to-peer network, the system performs the following operations:

- a Query Message is sent, over an existing TCP connections, to all the neighbors of the node that makes the request. With this query message the node asks to all its neighbors if they have the desired file.
- Each node that receives the query message, performs the following operations:
  - this node (peer) forwards the query message to its neighbors exploiting the exiting TCP connections.

- If this node (peer) has the desired file, then it sends over the reverse path (which connects this node to the requesting node) a message called “Query Hit” which contains the its IP address. In this way the node X can download the file directly from the node that holds the desired file.

The scalability of this protocol is limited to the scope of the flooding.

1.2.6.2 Gnutella: Peer Joining

So that a peer X joins to the Gnutella peer-to-peer network, this peer has to find other peers (nodes) that belong to the same peer-to-peer network. To do this, the node X performs the following steps:

- the node X starts with a list of candidate peers.
- Then X sequentially attempts TCP connections with each peer that is on this list until a connection is setted up with a node Y (of the same list).

At this point the node X sends a Ping message to the node Y:

- the node Y forwards the Ping message to all its neighbors.
- All the nodes that receives the Ping message, respond with a Pong message that is forwarded to the node X.

In this way the node X receives many Pong messages and therefore the node X can set up additional TCP connections with the nodes that sent the Pong messages to the node X.

1.2.6.3 Gnutella: Advantages and Disadvantages

The advantages of Gnutella are the following:

- we have a fully decentralized system.

- The search cost (for a file) is distributed among the peers involved in the query flooding.
- The distributed processing permits powerful semantics search.

The disadvantages of Gnutella are the following:

- the search scope of the file may be quite large.
- Therefore the time, necessary to find the nodes which have the desired file, may be quite long, since the request has to be propagated in all the Overlay network.
- There is an high content of overhead in the messages and the nodes come and go often from the network.

1.2.7 KaZaA

The protocol used by KaZaA [34] is called "Smart Query Flooding" and it provides the following operations:

- Join; in this operation, at the beginning, the client contacts a super-node and later this client may become a super-node.
- Publish; in this operation, the client sends the list of its files to its super-node and in this way the client publishes the list of its files.
- Search; in this operation, the requesting node sends the query message to the super-node which sends this query to the other super-nodes. These super-nodes flood this query among themselves to find the nodes which have the desired file.
- Fetch; in this operation, the requesting node gets the file directly from the peer or the peers that hold the desired file. In this way the requesting peer gets the desired file from more than one peer at a time.

1.2.7.1 KaZaA: Exploiting Heterogeneity

The KaZaA peer-to-peer network exploits the existing heterogeneity among the nodes of the network, because there are two different kinds of nodes. These nodes are called the super-node and the ordinary node. These two kinds of nodes perform different tasks. Each peer of the network is either a group leader (super-node) or it is assigned to a group leader (ordinary node). Therefore, in this situation, the following connections are created:

- a TCP connection between each peer and its group leader.
- one or more TCP connections between some pairs of group leaders.

Thus, each group leader (super-node) tracks the contents in all its children nodes (namely, ordinary nodes). An example of KaZaA peer-to-peer network is depicted in figure 1.2.

1.2.7.2 Query Consolidation

The query consolidation is performed to optimize the search operation. Moreover the query consolidation is performed because many connected nodes may have only a few files and therefore the propagation of a query to a sub-node (ordinary node) may take more time than for the super-node to answer itself.

Thus, since in the peer-to-peer network there are particular nodes called super-nodes, we have an improvement of the query flooding protocol because the system does not propagate the query message to the ordinary nodes but the system propagates only the query message to the super-nodes. In this way the system reduces the impact of the flooding of queries on the network.

1.2.7.3 Stability

In this peer-to-peer system, the super-node selection favors the nodes with high up-time. In this way the system selects as new super-node only the nodes with the highest up-time

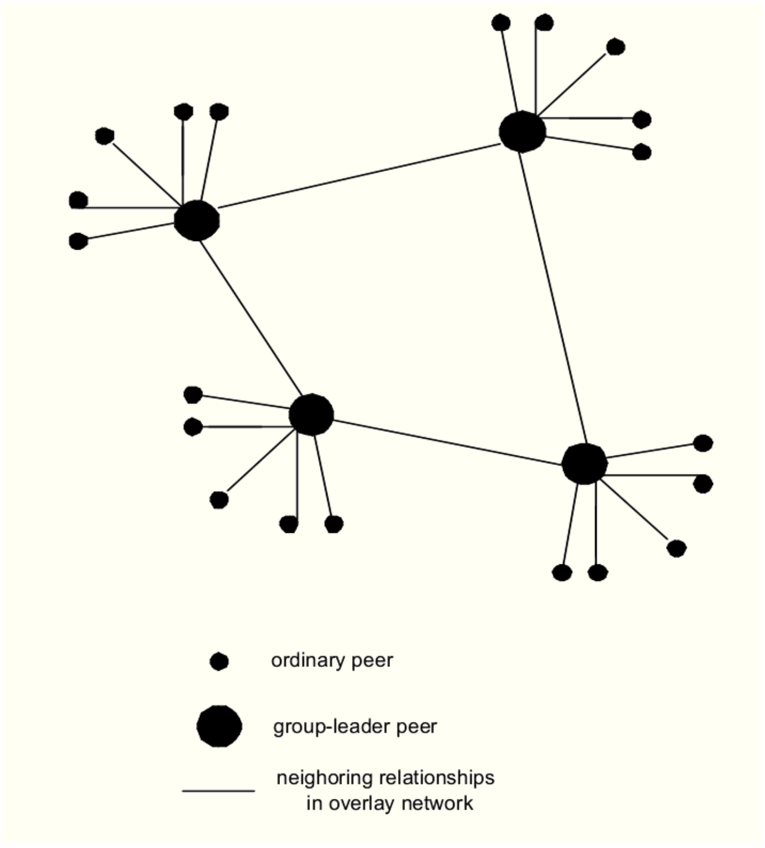

Figure 1.2: KaZaA peer-to-peer network example

because how long the node has been on represents a good predictor of how long the same node will be around in the future.

1.2.8 BitTorrent

The BitTorrent client is a software which manages the downloading and the uploading operations of torrents using the BitTorrent protocol [53].

The first client, known as BitTorrent [53], was created by Bram Cohen during the summer of 2002. Many following clients are partially based on the first BitTorrent client. At the beginning not all the clients were BitTorrent protocol complying. Only later these clients got the support for the BitTorrent protocol.

1.2.8.1 BitTorrent aims

The aim of BitTorrent is to allow an efficient download of the desired content. Therefore the purpose of the BitTorrent is not to improve the research of contents (files). In this way we want to optimize the distribution of files to many peers, whereas there is a single editor (of the file) and many downloaders. Another aim of BitTorrent is to prevent the problems of free-loading and free-ridering.

1.2.8.2 BitTorrent: Simultaneous Downloading

In order for the peers to download the desired large file simultaneously from other peers, the following operations are required:

- the large file is divided into many pieces:
  - the different pieces are replicated on different peers.
  - A peer with a complete piece can trade with other peers in order to get other pieces.
  - In this way each peer can hopefully assemble the entire file.
- It is allowed the simultaneous downloading of different pieces:
  - each peer can retrieve different parts (pieces) of the desired file from different peers at the same time.
  - Each peer can upload parts (pieces) of the same file to other requesting peers.
  - This management of the portions of the files is very important for very large files.

1.2.8.3 BitTorrent: Tracker

The tracker is an architecture of nodes which keeps track of the peers participating in the torrent. The peers register with the tracker through the following procedure: when a new peer arrives in the torrent this peer registers with the tracker and this peer periodically informs the tracker if it is still there. When a new peer joins the torrent, the tracker selects the set of peers from which the new peer can download the file of the considered torrent. Therefore the tracker returns a random set of peers (including their IP addresses) to the new peer from which the new peer can download the desired torrent. In this way the new peer is aware of who to contact for the downloading of data.

1.2.8.4 BitTorrent: Chunk

In the BitTorrent protocol the large files are divided into smaller pieces called chunks. The chunks are pieces of a file with fixed size. Typically, the size of each chunk is 256 KByte.

The use of the chunks allows the simultaneous transfer of the various portions of the file namely; each peer can download different chunks from its neighbor peers and it can upload chunks to other neighbors.

Each peer has to be able to learn what chunks its neighbors have namely: periodically, the considered peer asks its neighbors for the list of the chunks (related to the desired file) that they have. The file download is completed when all the chunks of the file are downloaded.

1.2.8.5 BitTorrent: Overall Architecture

In the overall architecture of BitTorrent, we consider a peer X which wants to download a desired torrent. To do this peer X accesses the web server which contains the web page in which there is the link of the desired torrent file, and thus peer X downloads this torrent file. In the torrent file (downloaded by peer X above) the list of the trackers which manage the peers that have the desired file is found inside of this file. Therefore peer X performs the file request to each tracker (specified in the torrent file). With this operation the peer X registers itself in each tracker and the peer X gets the list of the peers which have the desired. At this point the peer X performs the chunks request to the peers specified in the list received from the trackers. Then the peer X comes to an agreement with the peers specified in the list received from each tracker in order to establish from which peer the peer X can gets each chunks of the desired file (hand shaking operation). The peers, that receive the download request from the peer X, replay to the peer X by sending to the peer X the planned chucks of the file. Moreover when the peer X gets at least one chunk, this peer offers the obtained chunks to the other peers which do not have the entire file. The peers that do not have the entire file are called leechers while the peers that have the entire file are called seeders. Periodically the peer X sends announcements to the trackers in order to communicate to the tracker that it is still alive and active and in order to get the updated list of the peers that hold the desired torrent.

1.2.8.6 BitTorrent: Chunk Request Order

The issue that we are presenting now is the order in which the requested chunks are provided to the requesting peer. A possible solution is the downloading of the chunks in the same order that they appear in the desired file like a HTTP client does it. But if we use this solution, the following problem occurs: if the peer downloads the chunks in the same order that they appear in the file then many peers have only the early chunks of the file. In this way peers have little to share with each other since all the peers have only the the early chunks of the file. Moreover this solution limits the scalability of the system. A further problem is that nobody has the rare chunks, for example it may occur that the most needed chunks are the chunks of the end of the file, and in this way it is limited the ability to complete the download of the desired file.

Therefore the best solution is to use a random selection of the chunks and to provide the rarest chunks first. In this way we increase the probability of completion of the desired file.

1.2.8.7 BitTorrent: Rarest Chunks First

In this subsection we investigate which chunks it is necessary to request first to avoid the problems presented above. The best solution for the chunks selection is to provide the chunk with the fewest available copies, namely for example the rarest chunks first.

With this solution we have benefits to each peer of the peer-to-peer network because it avoided starvation when some peers depart and when some chunks are not anymore available on some peers, since each peer of the network requests the chunk with the fewest available copies first.

Moreover this solution provides benefits to the entire system because it avoids the starvation across all the peers that are wanting a file and it balances the load by equalizing the number of copies of the chunks.

1.2.8.8 Free-Riding Problem in the Peer-to-Peer Networks: how to avoid this problem

The vast majority of the users are free-riders, namely the vast majority of the users do not share files and do not answer to the queries of the other users. Other peers are considered free-riders because of other limits such as a limited number of possible connections or a limited upload speed.

In some cases a few "peers" essentially act as servers and therefore there are only a few individuals contributing to the public good, making them hubs that basically act as a server.

Thus BitTorrent prevent the free-riding phenomenon allowing the fastest peer to download the desired content from the peers which act as servers and occasionally the system let some free-riders download the desired content.

1.2.8.9 BitTorrent: Free-Riding Prevention

We have to consider that some peers have a limited upload bandwidth. In this case if a peer has a limited upload bandwidth this peer must share it bandwidth among multiple peers. Therefore the techniques used to prevent the free-riding phenomenon are the following:

- we give an higher priority to the upload bandwidth namely, in the peer-to-peer network, we favor the neighbors that are uploading at highest rate.
- Moreover the system rewards the top four neighbors, namely:
  - it is measured the download bit rates from each neighbor.
  - The considered peer reciprocates by sending, some parts of the content, to the top four peers.
  - The re-computation of the best four neighbors the and reallocation of the best four neighbors are done every 10 seconds.
- Optimistic unchoking:
  - each peer tries randomly a new neighbor every 30 seconds to see if it will be one of the top four neighbors.
  - In this way a new neighbor has a chance to be a better partner in the communication.

1.2.8.10 BitTorrent Today Issues

The biggest problem of the BitTorrent is the incomplete downloads: usually the peers leave the system when they complete the download of the desired content. In this way some peers will not be able to complete the download of the desired content. Especially this problem is present for less popular contents.

Another issue of the BitTorrent system is that a significant fraction of Internet traffic (estimated around 30%) is used by the peer-to-peer file sharing.

Finally, there are still many outstanding legal issues regarding the exchange of files protected by copyright.

1.3 Live Streaming Peer-to-Peer Architecture: NAPA-WINE

A typical Peer-to-Peer TV system is designed and optimized following a pure layered approach, without considering the effects of the application layer on the underlying transport network layer. This kind of approach can be dangerous for the possibility to have a traffic congestion of the network due to the Peer-to-Peer TV traffic because this traffic is not generated in accordance with a cross-layer approach which involves both layers, application layer and transport layer. In the NAPA-WINE project [31, 32] is introduced an innovative architecture in which a cooperative Peer-to-Peer TV application is developed. In this approach there is a cooperation between the application layer and the transport network layer.

The NAPA-WINE project focuses on improving the performance of Peer-to-Peer live streaming distribution and protecting the network resources from excessive usage. In a peer-to-peer network, two network layers can be identified: the overlay layer in which peers exchange content with neighbor peers, and the IP layer in which routers and links transfer the packets among the network. The main goal of NAPA-WINE is the study of the novel Peer-to-Peer architectures and systems suitable for High Quality TV live streaming over the Internet. In the typical Peer-to-Peer system the overlay layer and the transport network layer are completely independent and times antagonists. For these reasons the lack of co-ordination between these layers can produce a congestion of the network and therefore it can deteriorate the quality of service of the live streaming TV over a peer-to-peer network because of dropping packets due to the congestion of the entire network. Therefore in this approach, where the overlay and the IP network ignore each other, while is acceptable for elastic application of low-bandwidth streaming applications, may cause intolerable performance degradation in the case of high-quality real-time application such as Peer-to-Peer TV. Moreover if the network becomes heavily congested, the application will never be able to meet its minimum requirements.

For these reasons in NAPA-WINE there is a cooperative paradigm in which the application and the network layers interoperate and cooperate to optimize the service offered to the end user. Moreover the only successful paradigm for a large scale Peer-to-Peer TV architecture is a cooperative paradigm, where the network and the application layers join forces to meet the quality of service required and to reach the largest possible population of users.

This cooperative approach give us also an additional incentive of advantage. In a traditional Peer-to-Peer system there are only two main actors which are the network providers and the application users and they interests are often in contrast. In the case of Peer-to-Peer TV there is a new actor which comes into play: the content producer. The interests of this actor are mainly in distributing its content through the Internet in the fastest possible way, thus avoiding every kind of problem, either technical or legal. This new actor represents the major shift with respect to the current Peer-to-Peer applications, where users provide content, a shift that offers further incentives for the network operators to cooperate with the content providers and the users to successfully deliver the audio-video contents.

The NAPA-WINE project proposes an innovative network cooperative Peer-to-Peer architecture which explicitly targets the optimization of the quality perceived by the users and minimization of the impact on the underlying transport network layer. In this study is designed and developed a Peer-to-Peer HQTV system, in which the peers set up a generic mesh based overlay topology, over which the video chunks are exchanged according to a swarming-like approach. In this case a source peer produces the video stream, divided in chunks, and sends them into the overlay network where the peers cooperate to distribute them. An important element of this Peer-to-Peer HQTV system is given by the built-in distributed monitoring tool which allows the application to continuously collect real time information both on the network condition (to prevent the congestion of the network) and on users' perceived performance. The information collected by the monitoring tool can be used to trigger reconfiguration algorithms that act at the level of the chunk distribution mechanism (called scheduler) and at the level of the overlay topology reconfiguration. Moreover the proposed architecture allows the network layer to expose useful information to the application layer and the network provider has the capability to guide the peer-to-peer application, for example by explicitly publishing information about the status of its network, such as link congestion or Autonomous Systems routing preferences.

Therefore the NAPA-WINE project [31, 32] is designed withe goal of efficiency and co-operation between the application and both the network operator and the content provider, aiming at optimizing resource usage and minimizing the Peer-to-Peer TV application impacts on the transport network. the main features of this approach are the following: the algorithms used for topology management and information scheduling use the network measures to minimize the usage of the network and preserve the application performance. There are shared repository used to exchange information between the network layer and the application layer so that the peers can decide the connectivity (with other peers), topology, signaling and information according to the appropriate knowledge- base represented by the shared repository.

For the application and the network layer the jay features are the overlay topology management and the chunk scheduling of information. The overlay topology management makes it possible to build an efficient and rational overlay topology which is correctly mapped on top of the transport network structure. The chunk scheduling guarantees that the network capacity is exploited without waste (for example, by minimizing retransmissions and pursuing an efficient distribution of chunks). These key features have to be supported by measurements that in the most case are dynamic and can only be implemented in the application overlay later. For these reasons the development of efficient algorithms and methods for measurement are fundamental for the overall system to allow an efficient topology management and an efficient scheduling of the chunks.

1.4 Organization of the thesis

The thesis is organized as follows: In the second chapter we explore the Peer-to-Peer Network environment. We present and analyse the proxies network which is completely the same analysis that we can do for the query flooding protocol. Moreover we provide an analytical formulation for the proxies networks which is used to establish the gain between a network without proxy layer and the same network with a proxy layer for each node of the network. Finally we present the results in different scenarios and we establish which proxies network topology achieve the minimum average access time in the various scenarios.

Than in the third chapter we focus on the Peer-to-Peer Network used for the distribution of multimedia content (from a source node retrieved through the query flooding protocol) on asymmetric channel and on two different kind of radio channel. In this chapter we present the state of the art and five new algorithms used to build (in a incremental way) the Peer-to-Peer networks used for the distribution of multimedia contents. For the asymmetric channel we analyse the behaviour of the network, produced by the new algorithm, in terms of the average maximum number of peer accepted by the peer-to-peer network and the average maximum delay of the peer-to-peer network. We present a theoretical optimum that maximize the average maximum number of peers accepted by the peer-to-peer network and minimize the average maximum delay of the peer-to-peer network. Through the simulation results we establish the algorithms which produce peer-to-peer networks closer to the theoretical optimum. For the radio channel we analyse the behaviour of the peer-to-peer networks produced by the six algorithms, presented above, with respect to the percentage of correctly received bits. We present two different radio channel models and a peer-to-peer network simulator. Through the simulation results we establish the algorithms which produce peer-to-peer networks with the highest percentage of correctly received bits.

In the fourth chapter we present an application of the peer-to-peer networks which is a new mechanism (based on peer-to-peer network) used to make reliable an unreliable multicast network. We use a peer-to-peer network of the destination nodes of the multicast network to make reliable the multicast network. Then we compare the results (in term of the average access time) produced by a reliable multicast network (based on an ARQ mechanism) and the results produced by the unreliable multicast with peer-to-peer network used to make the network reliable.

In the fifth chapter we present a new protocol based on the Massey Omura cryptography protocol. We use this protocol to make secure the communication in a group of peers in a peer-to-peer network on a geographic network. The aim of the new protocol is to generalize the Massey Omura protocol so that we can use this generalization to make possible the many to many communications used in the key distribution or in multiple communications used in a peer-to-peer network. We analyse some security problems about the Massey Omura protocol and the Multiple User Massey Omura protocol and we propose how to solve these issues. Moreover we present a protocol used for the generation and the distribution of the decryption shared key used by the proposed protocol.

In the last chapter of the thesis, the conclusions are presented.

Chapter 2

Cooperative Proxies Network And Query Flooding Analytical Model for a Peer-to-Peer Network

2.1 Introduction

A very important topic in the ICT field is based on information retrieval. The nature of internet is based on layer 3 nodes where the source gets the information from the server through the routing of the datagram packet from the server to the client. In this case the information delivery is not very efficient because the requested packet has to travel from the server to the client through the intermediate routers of the path that interconnects the server to the client. A way to increase the performance of the network, in term of average delivery time, is achieved if each node of the network includes proxy functionality (namely with a cache integrated in this node) over the layer 3 (routing) because the request and the desired information don't have to travel between the client and the server but they have to travel between the client and the intermediate node which holds the content. Caching can be performed at different levels in a network of computer [54], among them the browser, the proxy and the web-server levels. Proxy with cache has been developed mainly to reduce the Internet bandwidth consummation, the server load, the latency and to increase the reliability [55]. Caching has been the subject of intensive studies, such as Thomas et al. [56][3], Mokherjee and Tan [57], ElAarag and Romano [58]. But there are relatively few studies, in the literature, that deals with proxy cache networks. Kumar [59] proposed a recent study about the algorithms for performance evaluation of the network based on Tawarmalani's [60] theoretical approach. Moreover the distributed proxies networks has been the subject of many studies such as : [61, 62, 63, 64, 65, 66, 67], these articles address the determination of the performance analysis of a distributed proxy network in various ways but these analyses don't make a comparison between the distributed network with layer 3 and 7 nodes, and the same network with only layer 3 nodes. There isn't an analytical model or an analytical analysis about the query flooding protocol in the current state of the art, for example the articles [1, 3, 2, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14] present the query flooding protocol or an evolution of this protocol (in terms of optimization and reduction of the flooding) without a specific mathematical model for it. Therefore the proxy network analysis presented in this chapter can be used to analyse the performance of the peer-to-peer network during the query flooding operation ([35, 36, 37], Kademlia [38] and KaZaA [34]) by simply replacing the proxies of the proxies network with the peers of the peer-to-peer. We address the analysis of a generic distributed proxies network with a comparison between the network with layer 3 and 7 nodes and the same network with only layer 3 nodes. In layer 3, the nodes perform only routing, store and forward operations. In nodes with layer 7 the nodes include routing, store and forward for layer 3 and proxy caching for layer 7. We compare the behaviour of a generic network of layer 3 and layer 7 nodes with respect to the behaviour of the same network with only layer 3 nodes. First of all we present the analysis about the average time to retrieve the information in a chain of nodes then we address the problem of the generalization of this formulation in a generic network of proxies. We present the gain between the network with layer 3 and 7 nodes and the same network with only layer 3 nodes, in different conditions about the hit probability of the proxies in the network. This chapter is organized as follows, the aim and the main hypotheses of the chapter are explained in section 2.2. Section 2.3 describes the performance analysis of a chain of proxies. In section 2.4 the performance analysis of a generic network of proxies is described. In section 2.5 the results in three different types of scenarios are presented.

2.2 Aim and Hypotheses

The aim of this chapter is to show how the proxies, in a network, can increase the performance of the information retrieval. We use a simplified network and not an actual network of nodes in a real environment, because in this way we derive an analytical performance formulation that could be otherwise impossible to obtain in a real network infrastructure for the many parameters present. The simplified network doesn't model all the characteristics of a real network, it models only the most important characteristics such that it shows if the usage of the layer 7 (proxy) can improve the performance of the network. The mathematical formulas, presented in the sections 3 and 4 , build a framework used to evaluate the performance of a network reproduced in a simulated environment.

The main hypotheses of the framework presented in this paper are the following:

- $C_{k,i}$ is the bandwidth of the link between the k-th node and the i-th generic node.
- $C_{k,i} = C_{i,k}$.
- $Q_{k,i}$ is the dimension of the packet used to query the desired content in the link between the k-th node and the i-th node.
- $V_{k,i}$ is the dimension of the packet that contains the desired content in the link between the i-th node and the k-th node.

- $\tau_{k,i}$ is the propagation delay between the k-th node and the i-th generic node.
- $\tau_{k,i} = \tau_{i,k}$.
- P_{h_i} is the hit probability of the i-th node
- The request of a content from the k-th node to the i-th node is transmitted in an interval of time equal to $t_{req_{k,i}} = Q_{k,i}/C_{k,i}$.
- The desired content is sent from the i-th node to the k-th node in an interval of time equal to $t_{resp_{k,i}} = V_{k,i}/C_{k,i}$.
- k equal to 0 indicates the Client.
- i equal to n+1 indicates the Server.
- All the parameters are constant during the time.

2.3 Chain of Proxies Performance Analysis

In this section we explain the performance analysis of a chain of proxies with: one level of proxy; two levels of proxies; and in a generic chain with n levels of proxies.

First of all we analyze the chain of one proxy as shown in figure 2.1. This case is partially analysed in [63] formula (1.2).

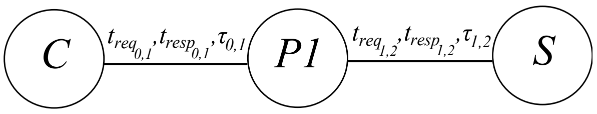
C
$t_{req_{0,1}}, t_{resp_{0,1}}, \tau_{0,1}$
P1
$t_{req_{1,2}}, t_{resp_{1,2}}, \tau_{1,2}$
S

Figure 2.1: One level of proxy

The client C gets the required content sending the request to the proxy P1. P1 receives the request after an interval of time equal to $t_{req_{0,1}} + \tau_{0,1}$. When the proxy P1 receives the request (from the client C) there are two possible events:

- in the first event the proxy P1 has the content with hit probability P_{h_1}, so the proxy sends the content to client C. Client C then receives the content after an average interval of time equal to $P_{h_1} \cdot \left(t_{resp_{0,1}} + \tau_{0,1}\right)$.
- In the second event the proxy P1 doesn't have the required content (this event happens with a probability equal to $1 - P_{h_1}$), so the proxy sends the request to the server S. The server S receives the request after an interval of time equal to $t_{req_{1,2}} + \tau_{1,2}$. The server sends back the content to the proxy P1. P1 receives the content after an interval of time equal to $t_{req_{1,2}} + t_{resp_{1,2}} + 2\tau_{1,2}$ and it transmits the content to the client C. In this way the client receives the content in an average time equal to $(1-P_{h_1}) \cdot \left(t_{req_{1,2}} + t_{resp_{1,2}} + t_{resp_{0,1}} + 2\tau_{1,2} + \tau_{0,1}\right)$.

With only one level of proxy (figure 2.1), the client gets the content in an average access time as follows:

$$\overline{T} = t_{req_{0,1}} + \tau_{0,1} + P_{h_1} \cdot \left(t_{resp_{0,1}} + \tau_{0,1}\right) + (1-P_{h_1}) \cdot \left(t_{req_{1,2}} + t_{resp_{1,2}} + t_{resp_{0,1}} + 2\tau_{1,2} + \tau_{0,1}\right) \tag{2.1}$$

After some steps we obtain the final formula, for one level of proxy:

$$\overline{T} = t_{req_{0,1}} + t_{resp_{0,1}} + 2\tau_{0,1} + (1-P_{h_1}) \cdot \left(t_{req_{1,2}} + t_{resp_{1,2}} + 2\tau_{1,2}\right) \tag{2.2}$$

where P_{h_1} is the hit probability of the proxy P1.

Now we analyze the chain of two proxies as shown in figure 2.2. This case is partially analysed in [63] formula (1.3).

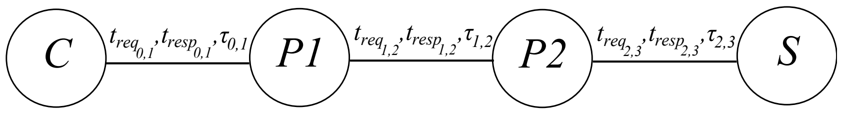
C
$t_{req_{0,1}}, t_{resp_{0,1}}, \tau_{0,1}$
P1
$t_{req_{1,2}}, t_{resp_{1,2}}, \tau_{1,2}$
P2
$t_{req_{2,3}}, t_{resp_{2,3}}, \tau_{2,3}$
S

Figure 2.2: Two level of proxy

The client C gets the required content sending the request to the proxy P1. P1 receives the request after an interval of time equal to $t_{req_{0,1}} + \tau_{0,1}$. When the proxy P1 receives the request (from the client C) there are three possible events:

- in the first event the proxy P1 has the content with hit probability P_{h_1}, so the proxy sends the content to client C. Client C receives the content after an average interval of time equal to $P_{h_1} \cdot \left(t_{resp_{0,1}} + \tau_{0,1}\right)$.
- In the second event the proxy P1 doesn't have the required content but the proxy P2 holds the content (this event happens with a probability equal to $(1-P_{h_1}) \cdot P_{h_2}$), therefore the proxy sends the request to P2. The proxy P2 receives the request after an interval of time equal to $t_{req_{1,2}} + \tau_{1,2}$. The proxy P2 sends back the content to the proxy P1. P1 receives the content after an interval of time equal to $t_{req_{1,2}} + t_{resp_{1,2}} + 2\tau_{1,2}$ and it transmits the content to the client C. In this way the client receives the content in an average time equal to $(1-P_{h_1}) \cdot P_{h_2} \cdot \left(t_{req_{1,2}} + t_{resp_{1,2}} + t_{resp_{0,1}} + 2\tau_{1,2} + \tau_{0,1}\right)$.
- In the third event the proxies P1 and P2 don't have the required content (this event happens with a probability equal to $(1-P_{h_1}) \cdot (1-P_{h_2})$) In this case the proxy P1 sends the request to the proxy P2. P2 receives the request after an interval of time equal to $t_{req_{1,2}} + \tau_{1,2}$. The proxy P2 sends the request to server S. Server S receives the content after an interval of time equal to $t_{req_{2,3}} + \tau_{2,3}$. It then transmits back the content to proxy P2. The proxy P2 receives the content after an interval of time equal to $t_{req_{2,3}} + t_{resp_{2,3}} + \tau_{2,3}$ and it sends back the content to the proxy P1. Proxy P1 receives the content after an interval of time equal to $t_{req_{1,2}} + t_{req_{2,3}} + t_{resp_{2,3}} + t_{resp_{1,2}} + 2\tau_{2,3} + 2\tau_{1,2}$ and it transmits the content to client C. In this way the client receives the content in an average time equal to $(1-P_{h_1}) \cdot (1-P_{h_2}) \cdot (t_{req_{1,2}} + t_{req_{2,3}} + t_{resp_{2,3}} + t_{resp_{1,2}} + t_{resp_{0,1}} + 2\tau_{2,3} + 2\tau_{1,2} + \tau_{0,1})$.

With two levels of proxies (figure 2.2), the client gets the content in an average access time as follows:

$$\overline{T} = t_{req_{0,1}} + \tau_{0,1} + P_{h_1} \cdot (t_{resp_{0,1}} + \tau_{0,1}) + (1 - P_{h_1}) \cdot P_{h_2} \cdot (t_{req_{1,2}} + t_{resp_{1,2}} + t_{resp_{0,1}} + 2\tau_{1,2} + \tau_{0,1}) + \\ + (1 - P_{h_1}) \cdot (1 - P_{h_2}) \cdot (t_{req_{1,2}} + t_{req_{2,3}} + t_{resp_{2,3}} + t_{resp_{1,2}} + t_{resp_{0,1}} + 2\tau_{2,3} + 2\tau_{1,2} + \tau_{0,1}) \quad (2.3)$$

After some steps we obtain the final formula, for two levels of proxies:

$$\overline{T} = t_{req_{0,1}} + t_{resp_{0,1}} + 2\tau_{0,1} + (1 - P_{h_1}) \cdot (t_{req_{1,2}} + t_{resp_{1,2}} + 2\tau_{1,2}) + \\ + (1 - P_{h_1}) \cdot (1 - P_{h_2}) \cdot (t_{req_{2,3}} + t_{resp_{2,3}} + 2\tau_{2,3}) \quad (2.4)$$

Now we analyze the chain of n proxies as shown in figure 2.3. With n levels of proxies the client gets the content in an average access time as follows:

$$\overline{T} = \sum_{i=1}^{n} \left[(t_{req_{i,i+1}} + t_{resp_{i,i+1}} + 2\tau_{i,i+1}) \cdot \prod_{j=0}^{i} (1 - P_{h_j}) \right] \quad (2.5)$$

In the hypothesis, we assume that: $P_{h_0} = 0$ and P_{h_i} is the hit probability of i-th proxy. This formulation is obtain by a generalization of the formula (2.4).

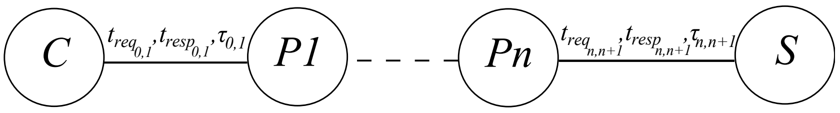
C
$t_{req_{0,1}}, t_{resp_{0,1}}, \tau_{0,1}$
P1
Pn
$t_{req_{n,n+1}}, t_{resp_{n,n+1}}, \tau_{n,n+1}$
S

Figure 2.3: n levels of proxy

2.4 Multiline Proxies and Query Flooding

This section describes the behaviour of multiline proxies (a network of proxies with a generic topology) and it is possible to use the same analytical model of the multiline proxies to estimate the average access time of the query flooding process, in a peer-to-peer network.

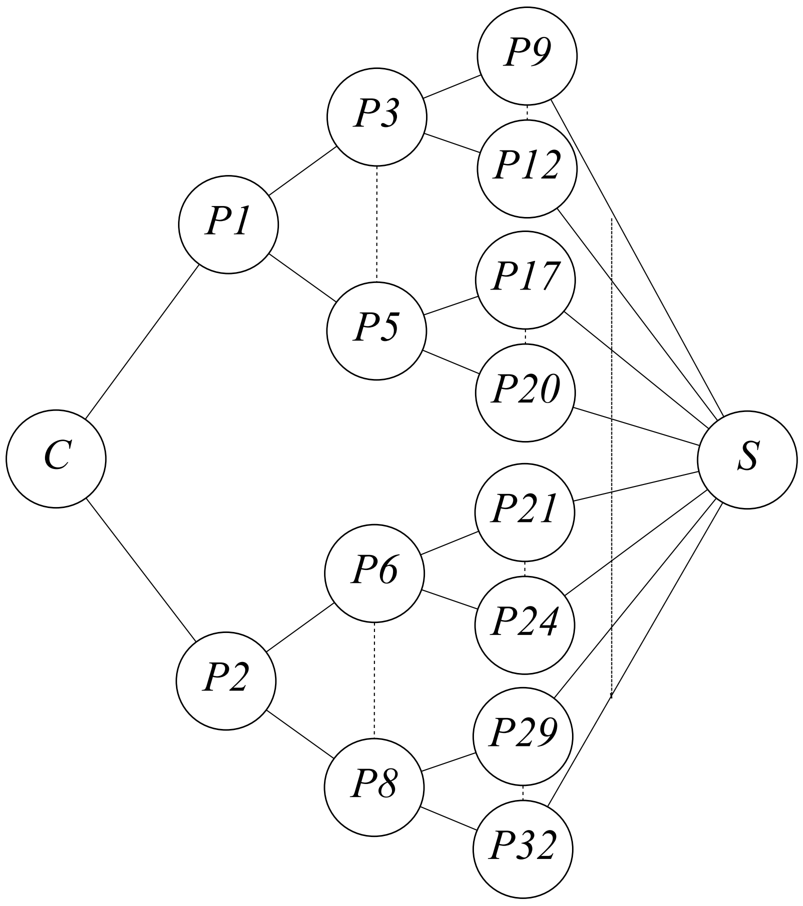

Figure 2.4: Example of multiline proxies

An example of multiline proxies has the topology depicted in figure 2.4, where C is the client, S is the server and *Pi* is the i-th proxy ($i = 1...32$) respectively. For the bandwidth occupation we suppose that: when a proxy of the network has a hit, it stops the propagation of the request in the other proxies of the network. In this way the system reduces the usage of the bandwidth in the proxies network. The mathematical formulation for the average time is explained in the next subsection. This formulation can be used also in the analysis of peer-to-peer network (discovery content Gnutella[35, 36, 37], Kademlia [38] and KaZaA [34]). The chain of proxies is a particular case of the multiline of proxies.

2.4.1 Formulation with Layer 3 and Layer 7

We achieve the recursive formulation for a network with layer 3 and 7 (in terms of the average time for the i-th node) through a generalization of the formulation (2.5), in which the sum $\sum$ is replaced by a recursive sum (through the term $\overline{T}_{L37_k}$) and a recursive mathematical product (through the term $P_{not\,hit_k}$) replaces the product $\prod$.

In this way the formulation achieved is the following:

$$\overline{T}_{L37_i} = Min_k \left\{ \overline{T}_{L37_k} + \left[\frac{Q_{k,i}}{C_{k,i}} + \frac{V_{k,i}}{C_{k,i}} + 2\tau_{k,i} \right] \cdot P_{not\,hit_k} \cdot (1 - P_{h_i}) \right\} \tag{2.6}$$

Where $\overline{T}_{L37_k}$ is the average time delay generated by the k-th node that is in the input of the generic i-th node. P_{h_i} is the hit probability of the i-th node. $P_{not\,hit_k} = \prod_{j \in Path\,from\,C\,to\,K}(1 - P_{h_j})$. $P_{h_C} = 1 - P_{not\,hit_C} = 0$ and $P_{not\,hit_i} = Min_k\{P_{not\,hit_k}\} \cdot (1 - P_{h_i})$.

For each i-th node directly connected to the client $\overline{T}_{L37_k} = 0$.

For the server S the formulation becomes the following (because there is no hit probability in the server):

$$\overline{T}_{L37_S} = Min_k \left\{ \overline{T}_{L37_k} + \left[\frac{Q_{k,S}}{C_{k,S}} + \frac{V_{k,S}}{C_{k,S}} + 2\tau_{k,S} \right] \cdot P_{not\,hit_k} \right\} \tag{2.7}$$

Where each k-th proxy is the closest to the server. The figure 2.5 depicts an example that can be described by formulas (2.6) and (2.7).

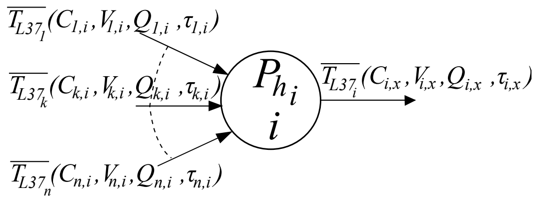
$\overline{T_{L37_1}}(C_{1,i}, V_{1,i}, Q_{1,i}, \tau_{1,i})$
$\overline{T_{L37_k}}(C_{k,i}, V_{k,i}, Q_{k,i}, \tau_{k,i})$
$\overline{T_{L37_n}}(C_{n,i}, V_{n,i}, Q_{n,i}, \tau_{n,i})$
P_{h_i}
i
$\overline{T_{L37_i}}(C_{i,x}, V_{i,x}, Q_{i,x}, \tau_{i,x})$

Figure 2.5: Generic node in a proxy distributed network with layers 3-7

2.4.2 Formulation with the Layer 3 only

The recursive formulation for the network with only layer 3 (example of network in figure 2.6) is the following:

$$T_{L3_i} = Min_k \left\{ T_{L3_k} + \frac{Q_{k,i}}{C_{k,i}} + \frac{V_{k,i}}{C_{k,i}} + 2\tau_{k,i} \right\} \tag{2.8}$$

This formula is obtained by the formula (2.6) without layer 7 ($P_{h_i} = 0 \, \forall \, node\, i$). T_{L3_k} is the time delay generated by the k-th node that is in the input of the generic i-th node. For each i-th node directly connected to the client $T_{L3_k} = 0$.

The figure 2.6 depicts the situation described by formula (2.8).

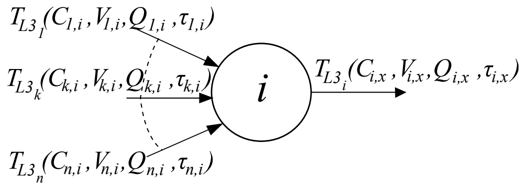
$T_{L3_1}(C_{1,i}, V_{1,i}, Q_{1,i}, \tau_{1,i})$
$T_{L3_k}(C_{k,i}, V_{k,i}, Q_{k,i}, \tau_{k,i})$
$T_{L3_n}(C_{n,i}, V_{n,i}, Q_{n,i}, \tau_{n,i})$
i
$T_{L3_i}(C_{i,x}, V_{i,x}, Q_{i,x}, \tau_{i,x})$

Figure 2.6: Generic node in a proxy distributed network with layer 3 only

2.4.3 Gain Between Layer 3 and Layers 3-7

In this section we achieve the gain (in the server S) between layer 3 and layers 3-7 through the previous formulations.

We achieve the gain (in the server S) between layer 3 and layer 3-7 through the formulas (2.7) and (2.8), in the following way:

$$Gain(S) = \frac{T_{L3_S}}{\overline{T}_{L37_S}} = \frac{Min_k \left\{ T_{L3_k} + \frac{Q_{k,S}}{C_{k,S}} + \frac{V_{k,S}}{C_{k,S}} + 2\tau_{k,S} \right\}}{Min_k \left\{ \overline{T}_{L37_k} + \left[\frac{Q_{k,S}}{C_{k,S}} + \frac{V_{k,S}}{C_{k,S}} + 2\tau_{k,S} \right] \cdot P_{not\,hit_k} \right\}} \tag{2.9}$$

This formulation has the followings limits:

- the limit of the gain between layer 3 and layers 3-7 for $P_{h_i} \rightarrow 1$ and for each value of i is the following:

$$\lim_{P_{h_i} \rightarrow 1, \forall i} Gain(S) > 1 \tag{2.10}$$

- The limit of the gain between layer 3 and layers 3-7 for $P_{h_i} \rightarrow 0$, $P_{not\,hit_k} \rightarrow 1$ and $\forall\,(i,k)$ is the following:

$$\lim_{\substack{P_{h_i} \rightarrow 0 \\ P_{not\,hit_k} \rightarrow 1 \\ \forall (i,k)}} Gain(S) = 1 \tag{2.11}$$

Where, in this case, $\overline{T}_{L37_k} = T_{L3_k}, \forall\,k$, because all the proxies of the network have hit probability equal to 0.

2.5 Scenario and Results

In this section we present the scenario and the results for the comparison between a proxies network (which is equivalent to a peer-to-peer network during a query flooding process) and a router network having the same topology of the proxies network. Therefore we present the result achieved through the framework described in the previous section with different hit probability models.

The experimental framework has the following parameters: $Q_{k,j}$, $C_{k,j}$, $V_{k,j}$, $\tau_{k,j}$, the number of nodes N and the model of the hit probability. The setting of the parameters are the following: $Q_{k,j} = V_{k,j} = 512\,KB$, $\tau_{k,j} = \tau = 50\,msec$, $C_{k,j} = C = 1\,Mbit/sec\;\forall(k,j)$. The proxies network has the topology shown in figure 2.4. We consider two proxies networks: the first one contains 32 proxies and it has the topology depicted in figure 2.4. The second one contains 8 proxies and it has the topology depicted in figure 2.4 but it contains only the first eight proxies of the network. We consider three kind of hit probability models described as follows.

We call scenario 1 the system with an uniform hit distribution $P_{h_i} = P_{hit}$, $P_{hit} \in (0,1)$, for $i = 1, ..., N$, $P_{h_{client}} = 0$, where N is the number of the proxies of the network. In this case the formulation of the Gain (formula (2.9)) has the following limits. The first limit is:

$$\lim_{P_{hit} \to 0} Gain(S) = 1 \tag{2.12}$$

This happens because the proxies network has the same behaviour of the routers network when the hit probability $P_{hit} \to 0$.

The second limit is:

$$\lim_{P_{hit} \to 1} Gain(S) = d \quad . \tag{2.13}$$

Where d is the maximum depth of the proxies network.

We call scenario 2 the system with a hit probability that exponentially decays with the distance in terms of hops from the server. The model assumes that the proxies have higher probability to have in cache the requested information, if they are closer to the server. For this scenario the hit probability model is the following: $P_{h_i} = 1 - \alpha^{depth(i)}$ where $\alpha \in (0,1)$. In this case the formulation of the Gain (formula (2.9)) has the following limits. The first limit is:

$$\lim_{\alpha \to 1} Gain(S) = 1 \tag{2.14}$$

This happens because the proxies network has the same behaviour of the routers network when the hit probability $P_{h_i} \to 0$, $\forall i$, when $\alpha \to 1$.

The second limit is:

$$\lim_{\alpha \to 0} \quad Gain(S) = d \tag{2.15}$$

Where d is the maximum depth of the proxies network.

We call scenario 3 the system with a hit probability that decays linearly with the distance in terms of hops from the client. As in opposition with respect to scenario 2, such a model assumes that the proxies have higher probability to have in cache the requested information, if they are closer to the client. For this scenario the hit probability model is: $P_{h_i} = 1 - \beta \cdot depth(i)/N_{d_i}$, where $\beta \in (0,1)$ and N_{d_i} is the number of hops present in the path that contains the i-th node. In this case the formulation of the Gain (formula (2.9)) has the following limit:

$$\lim_{\alpha \to 0} \quad Gain(S) = d \tag{2.16}$$

Where d is the maximum depth of the proxies network.

In figure 2.7 the performance gain (result obtained through the simulation of the system with uniform hit probability, exponentially decaying hit probability and linearly decaying hit probability) of scenario 1, 2 and 3 are reported. The benefit of using layer 7 strategy with the uniform hit probability P_{hit} profile increases both with the number of node N and P_{hit}. Note that the benefit of using layer 7 strategy with linearly decaying hit probability P_{hit} profile (and with the exponential hit probability P_{hit} profile) increase with N, decrease with β (and α).

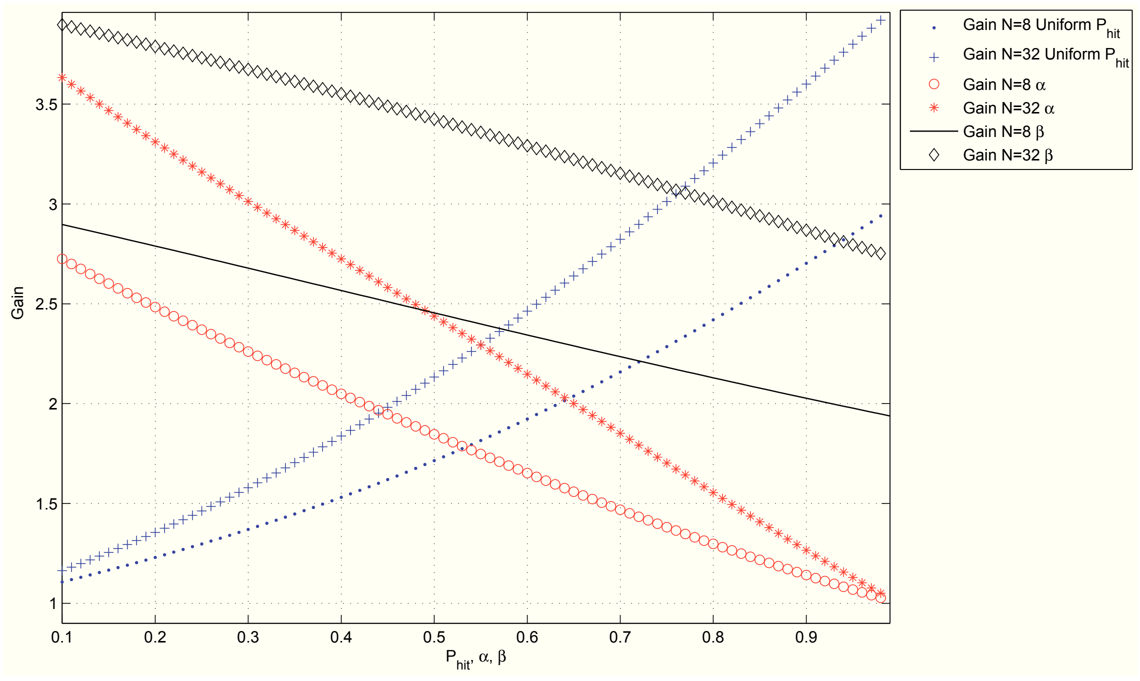

Figure 2.7: Gain in function of P_{hit}, α and β in a multiline proxies network where the number of proxies is 8 and 32

Now we compare the performance of a multiline network with 32 proxies (figure 2.4) with respect to the average access time of a chain of 32 proxies. To compare these two kinds of networks the only meaningful parameter is the average access time. For multiline proxies network the average access time is explained in formulas (2.6) and (2.7). For the chain of proxies the average access time is explained in formula (2.5). With these formulations ((2.5), (2.6) and (2.7)), we compare the multiline network and the chain of proxies in the previous three scenarios. The framework produced the results depicted in figure 2.8 and 2.9 using the network of the figure 2.4 and the chain of 32 proxies for the experiments. In figure 2.8 the average access time of scenario 1 and scenario 2 are reported. In scenario 1 and 2 the performance of the multiline network is better than the one of the chain of proxies. In figure 2.9 the average access time of scenario 3 (hit probability that decays linearly with the distance in terms of hops from the client) is reported. The chain network, only in this case, has the best performance (in term of access time) because for this network the hit probability (in this model) of each node is greater than the hit probability of each node of the multiline proxies network. This situation happens because the maximum depth N_{d_i} of the chain of 32 proxies is bigger than the maximum depth of the multiline network of 32 proxies, the first one has a depth equal to 33 the second one has a depth equal to 4 (see the model of hit probability that decays linearly with the distance in terms of hops from the client, scenario 3).

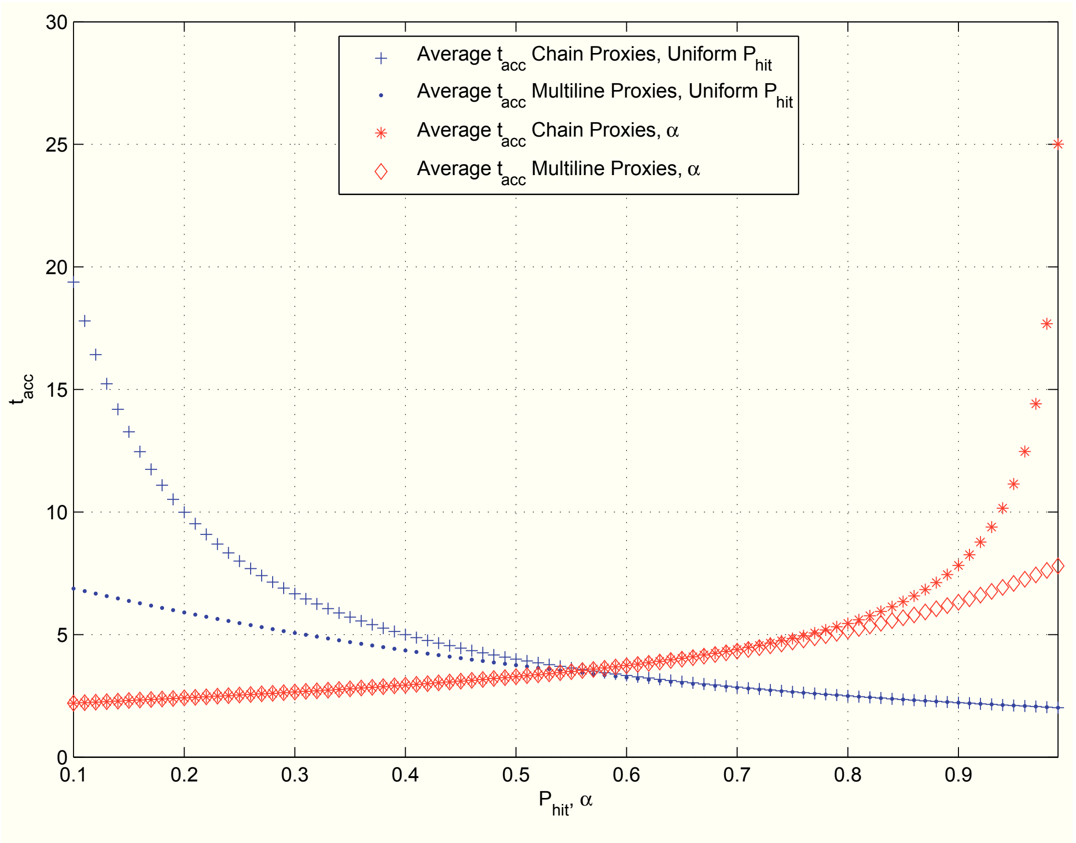

Figure 2.8: Average access time in a chain of 32 proxies and in a multiline network of 32 proxies depending on P_{hit} and α

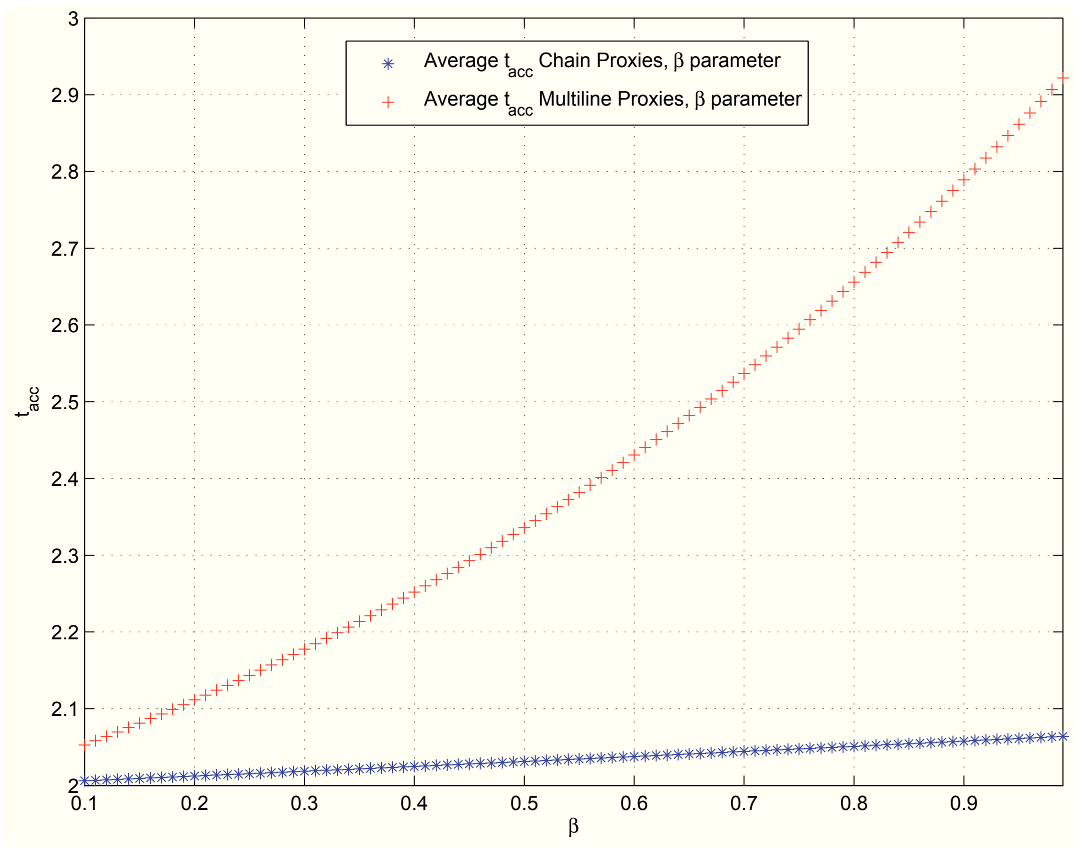

Figure 2.9: Average access time in a chain of 32 proxies and in a multiline network of 32 proxies depending on β.

Chapter 3

Peer-to-Peer Multimedia Distribution on Radio Channel and Asymmetric Channel

3.1 Introduction

In the Internet and Communication Technology (ICT) field, sharing and distribution of information is very important. Various mechanisms and techniques are used to manage information; one of these is based on peer-to-peer networks.

In today's world and in the near future, the exchange and distribution of information will be a very important aspect in the workplace and in daily life. Consequently, mobile devices, devices for home entertainment, personal computers and office terminals must have the mechanisms to achieve the above functionality. Thus the peer-to-peer networks can be used to achieve [68] the following: Video conferences or phone calls [69], in which more users can communicate together simultaneously.

The distribution of multimedia contents provided by a single source node, for example:

streaming distribution of TV contents or radio broadcasting contents [70, 31, 15, 71, 72]. An example of a real-time algorithm used to create a simple distribution peer-to-peer network on asymmetric channels is given in article [15] and the issues of performance of peer-to-peer file sharing over asymmetric and wireless networks is addressed in article [16].

Information sharing, for example in a company, the peer-to-peer network system can be used by employees to allow them to work in a shared manner. In daily life, the peer-to-peer network system can be used for sharing personal information such as audio and video contents, documents and others. The more significant peer-to-peer applications used for this purpose are: Gnutella [35, 36, 37], Kademlia [38], KaZaA [34], Bit-Torrent [53], massively multiplayer online game (MMOG) [73, 74]. The scenario discussed in this chapter is the distribution of multimedia contents provided by a single source node with an appropriate peer-to-peer network on asymmetric channels and on wireless channels.

This chapter is organized as follows, the scenario and the main hypotheses of the chapter are explained in section 3.2. Section 3.3 describes the peer-to-peer algorithms used to build the peer-to-peer distribution networks. In section 3.4 we present how is estimated the maximum delay of a peer-to-peer distribution network. In this section we present the theoretical optimum in which it is maximized the average maximum number of peers and it is minimized the average maximum delay of the peer-to-peer distribution network. Moreover the simulation results for the asymmetric channel are reported in the last part of this section. In section 3.5 we analyse the behaviour of the peer-to-peer algorithms in a simple radio channel. In this section we present:

- the radio channel characterizations.
- The model used to establish the bit error probability of each peer of a peer-to-peer distribution network.

- The peer-to-peer network simulator used to simulate the behaviour of the radio channel in the peer-to-peer distribution network.
- The validation of the model of the peer-to-peer network in an unreliable environment (radio channel) through the simulation results.
- The results used to establish which peer-to-peer algorithm builds the best peer-to-peer distribution network in an unreliable environment.

3.2 Scenario and Hypotheses

The scenarios discussed in this chapter refer to the distribution of multimedia contents transmitted by a single source node with an appropriate peer-to-peer network in an asymmetric channel and in a wireless environment. In this chapter we present two different classes of algorithms. The first class is based on the Tier based algorithm presented in the article [15]. In this class we have a central entity (server) that manages the insertion of the new peers and the construction of the network. The second class of algorithms, is based on a peer list. In this class we have a distributed system in which a new node gets from a server, the list of the nodes of the peer-to-peer network and then the new node periodically performs a query flooding to keep the list updated (such as Kademlia [38] is a distributed hash table for decentralized peer-to-peer computer networks). In this study we are not interested in how the network is managed (centralized or distributed). Instead, by using new algorithms we aim to: maximize the average maximum number of peers that can access the multimedia content and minimize the average maximum delay of the network, in the case of the asymmetric channel, and minimize the bit error probability of each node of the network, in the case of the wireless channel.

In our aim, the source node can be a home-user that streams multimedia content (i.e. audio/video) with a limited output bandwidth ($B < 2$) or a server with a higher output bandwidth ($B \geq 2$) which can supply the content to more than two users, where B is the output bandwidth of the source node.

In the case of the asymmetric channel the building of the network is done in real-time thus the algorithm we use creates a peer-to-peer network for streaming applications in which a source continuously provides content that must be played by a large and unknown number of home-users [15]. For hypothesis each home-user (peer) is characterized by an asymmetric channel such as ADSL and each peer has a uniform distributed output bandwidth. An ADSL system with a cooperative bit-loading approach for each peer of the peer-to-peer network [75] is used to ensure this hypothesis.

In case of the wireless system, we assume that each peer is an access point and that the network infrastructure is produced by the algorithm in non real-time and the algorithms we use in this chapter suppose that the peer-to-peer network is created before the initializing of the stream; moreover it is supposed that the placement of the various access points (peers) is done so that all wireless links have the same signal to noise ratio.

In both cases, the source node transmits the content while the receiving nodes are able to accept partial streams, from more than one node, through their inbound link and to redistribute it to one or more further peers through their outbound links. In this way the source node supplies the multimedia content to a limited number of requesting peers. The peers, that directly receive the streaming from the source node, provide the multimedia content to the other requesting nodes through a peer-to-peer network. The structure of this network depends on the algorithm used for incremental construction of the peer-to-peer network itself.

The base algorithm considers the source bandwidth as a constraint and minimizes the maximum delay in terms of intermediate links [15] without considering the number of nodes that the network is able to accept in accordance with bandwidth constraints.

Below is a list of hypothesis used in the next algorithms:

- the nodes of a peer-to-peer network are characterized by asymmetric channels.

- All peers are always available during the streaming.
- The source node of the network has a finite output bandwidth B.
- The inbound bandwidth of each node is adequate to accept the content.
- All bandwidths are normalized with respect to the bandwidth required to acquire the multimedia content. In this way the bandwidth required to acquire the multimedia content is normalized to 1.
- With respect to the bandwidth referred to above, the output bandwidth of each i-th peer is $0 < \Omega_i < 2$ and Ω_i can be different from Ω_j for each i-th and j-th peer of the network with $i \neq j$.
- Instead in the Mazzini-Rovatti Algorithm (Mazzini & Rovatti, 2008) all the peers have the same output bandwidth value $\Omega \in (0,1)$.
- The delay of each link is normalized to 1.
- In the case of the wireless channel all the links between couple of peers feature an identical signal to noise ratio.

3.3 Algorithms used to build the Peer-to-Peer Distribution Network

In this section we give a brief description of all the algorithms used in this chapter. There are two classes of algorithms that we are going to consider.

3.3.1 Tiers based algorithms

First of all we present the tier based algorithms with a simple analytical formulation for the average maximum number of peers accepted in the network and the maximum delay of the peer-to-peer network.

The first group of algorithms we will consider are classified under the Tier based algorithm (based on Mazzini-Rovatti Algorithm [15]). The first new algorithm we introduce is the Tier based algorithm (T). This algorithm is formulated by making a generalization of Mazzini-Rovatti Algorithm [15] with an output bandwidth of each peer distributed between 0 and 2. The second new algorithm we introduce is the Tier based algorithm with network Reconstruction (TR). The TR algorithm is formulated from the T algorithm we introduce above and its aim is to maximize the number of the peers accepted in the network. In this algorithm the output bandwidths of the peers of each tier are greater than the output bandwidths of the peers found in the next tier. Moreover when the network produced by the T and TR Algorithms don't accept a new peer for the first time, they don't accept more peers. The third algorithm we introduce is the TR Algorithm without Input Blockage. In this algorithm, if a new peer is not accepted in the network, this peer is inserted into a waiting queue. When a new node able to increase the residual output bandwidth of the network is inserted, the algorithm takes the peers from the waiting list and tries to re-insert them. A simple analytical formula for the maximum number of nodes accepted in a T network is:

$$n_T = \lfloor B \rfloor + \sum_{i=1}^{T} \lfloor B_i^* \rfloor \,,\ B_i^* = \sum_{k=1}^{\lfloor B_{i-1}^* \rfloor} \Omega_{k,i} \tag{3.1}$$

where B_i^* is the output bandwidth of the i-th tier of the network, i = 1...T and T is the maximum number of tiers. B_{i-1}^* is the output bandwidth of the previous tier (available output bandwidth of the previous tier), $B_0^* = B$ and $\lfloor B_i^* \rfloor$ is the maximum number of peers of (i+1)-th tier. $\Omega_{k,i}$ is the output bandwidth of the k-th peer contained in the i-th tier.

3.3.1.1 The State of the art

We present the state of the art for the algorithm used to build a peer-to-peer distribution network.

The state of the art is based on Mazzini-Rovatti's algorithm [15]. In this algorithm and in the first three new algorithms, the distribution network is organized in "tiers" numbered from 1 onwards. Peers in the first tier are fed by the source. Peers in the j-th tier receive the content only from peers in the (j−1)-th tier. The number of tiers in the distribution network is indicated by T, that also indicates the maximum delay in terms of intermediate links. In Mazzini-Rovatti's algorithm [15] a new peer is inserted into the tier closest to the source node. We indicate with p_j the number of peers in the j-th tier. The overall bandwidth required to distribute the content to the j-th tier is p_j, while the overall bandwidth made available by the j-th tier is $\Omega \cdot p_j$. We assume a finite total output bandwidth B offered to the first tier by the source node. The elementary step of Mazzini-Rovatti's algorithm is "add a peer to the j-th tier if possible". We indicate this step with A(j). A pseudo-code for A(j) is the following:

Algorithm 3.1 Elementary step of Mazzini-Rovatti's algorithm

```
case $j = 1$
        if $B > p_1$ then $p_1 + +$ else failed
case $j = 2, ..., T$
        if $p_{j-1} \geq p_j + 1$ then $p_j + +$ else failed
case $j = T + 1$
        if $p_T \geq 1$ then $p_{T+1} = 1,\ T + +$ else failed
```

where a peer is added to the j-th tier if and only if the bandwidth emitted by the previous tier (i.e. the source if j = 1) is enough to accommodate it, namely if this bandwidth is ≥ 1. The new peer insertion algorithm has the following pseudo-code:

Algorithm 3.2 Insertion Algorithm

```
for $j = 1$ to $T + 1$
        if $A(j)$ not failed then stop
next
failed
```

This algorithm tries to add the new peer to the smallest-delay tier within bandwidth constraints and fails if no more peers can be fed by peers in the same tier.

The new algorithms aim to maximize the number of accepted nodes (to increase the number of users that can have access to the content), minimize the reconstruction delay and minimize the maximum delay of the network (to provide a better service).

The next subsections describe and introduce new algorithms, adopted for the distribution of multimedia contents.

3.3.1.2 Tier based Algorithm (T)

The generalization of Mazzini-Rovatti's algorithm (Mazzini & Rovatti, 2008), with the new hypothesis that is "$0 < \Omega_i < 2$ for each i-th peer", is provided by the Tiers based algorithm (T). A pseudo-code for this algorithm is the following:

Algorithm 3.3 Elementary step of T Algorithm

```
case $j = 1$
        if $B \geq p_1 + 1$ then $p_1{+}{+}$ else failed
case $j = 2, ..., T$
        if $\sum_{i=1}^{p_{j-1}} \Omega_{i,j-1} \geq p_j + 1$ then $p_j{+}{+}$ else failed
case $j = T + 1$
        if $\sum_{i=1}^{p_T} \Omega_{i,T} \geq 1$ then $p_{T+1} = 1,\ T{+}{+}$ else failed
```

where p_j is the number of peers contained in the j-th tier (with $j = 1 \ldots T$) and $\Omega_{i,j-1}$ is the output bandwidth of the i-th peer contained in the (j−1)-th tier of the network and a new peer is added to the j-th tier if and only if the bandwidth emitted by the previous tier (i.e. the source if $j = 1$) is able to accommodate it. The new peer insertion algorithm has the following pseudo-code:

Algorithm 3.4 Insertion Algorithm

```
for $j = 1$ to $T + 1$
        if $A(j)$ not failed then stop
next
failed
```

This algorithm tries to add the new peer to the smallest delay tier within bandwidth constraints and fails if no more peers can be fed by peers in the same tier.

In the next subsection we describe an algorithm that increases the maximum number of peers accepted in the network, by adopting an insertion algorithm with the reconstruction of the network itself.

3.3.1.3 Tier based Algorithm with network Reconstruction (TR)

We present an evolution of the Tier based Algorithm which improve the maximum number of peers accepted in the network and minimize the maximum delay of the peer-to-peer network.

In this algorithm the tiers nearest to the source node must hold the nodes characterized by the greatest output bandwidth values. To guarantee this aim, the insertion of each new node can trigger a possible reconstruction of the distribution network. The elementary step of the TR algorithm is indicated with the recursive function $A(j, \Omega_N)$, where Ω_N is the output bandwidth of the new peer N. A pseudo-code for $A(j, \Omega_N)$ is the following:

Algorithm 3.5 Elementary step of TR Algorithm

```
case $j = 1$
        if $B \geq p_1 + 1$ then $p_1 + +$
        else $< remove\, N_{m_j} >, < add\, N >, A(j+1, B_{m_j})$
case $j = 2, ..., T$
        if $\sum_{i=1}^{p_{j-1}} \Omega_{i,j-1} \geq p_j + 1$ then $p_j + +$
        else $< remove\, N_{m_j} >, < add\, N >, A(j+1, B_{m_j})$
case $j = T + 1$
        if $\sum_{i=1}^{p_T} \Omega_{i,T} \geq 1$ then $p_{T+1} = 1,\ T + +$ else failed
```

where p_j is the number of peers contained in the j-th tier $(with j = 1 \dots T)$ and $\Omega_{i.j-1}$ is the output bandwidth of the i-th peer contained in the (j−1)-th tier of the network. Ω_N is the output bandwidth of the new node N. N_{m_j} is the peer with the minimum output bandwidth of the j-th tier because the hypothesis of this algorithm is that each j-th tier (with $j = 1 \dots T - 1$) holds all the peers characterized by output bandwidths greater than the output bandwidths of the peers held in the (j+1)-th tier. B_{m_j} is the output bandwidth of the peer N_{m_j}. In this way the algorithm tries to add the new peer in the j-th tier if $\Omega_N > B_{m_j}$.

The new peer insertion algorithm has the following pseudo-code:

Algorithm 3.6 Insertion Algorithm for the TR Algorithm

```
for $j = 1$ to $T + 1$
        if $\left(min\left(PeersOutputBandwidth\left(tier_j\right)\right) < \Omega_N\right) and\, (j < T + 1)$ then break
next
if $A(j, \Omega_N)$ not failed then stop else failed
```

This insertion algorithm tries to insert the new peer (N) in the j-th tier where the output bandwidth of the peer, characterized by the minimum output bandwidth, is less than the output bandwidth (Ω_N) of the new peer (N).

3.3.1.4 TR Algorithm without Input Blockage (TRwIB)

In this part we present an evolution of the TR algorithm that increase the maximum number of peers accepted in the network.

The TRwIB algorithm derives from the TR algorithm. The TRwIB is the TR algorithm without Input Blockage. The engine of this algorithm is the following:

- if the new peer can be inserted into the network (with network reconstruction if it is necessary) then the algorithm performs the insertion operation as the TR algorithm.
- Otherwise the new peer is inserted in a waiting queue; this queue contains the peers that are waiting a new node able to increase the residual output bandwidth of the network. When this event happens the algorithm wakes up the waiting peers and it tries to insert them. The peers, that are not inserted, are maintained in the waiting queue and they are waked up (by the insertion algorithm) if and only if a new peer, is able to increase the residual bandwidth of the network with its insertion.

The disadvantage of this algorithm is represented by the network reconstruction that introduces an additional delay to the network.

3.3.2 Peer List based algorithms

In this subsection we present a new class of algorithms used to build a peer-to-peer distribution network in which we want to achieve the maximum number of peers accepted in the network. To do this we cannot achieve the minimization of the maximum delay of the network. Moreover we present a simple analytical model for the average maximum number of peers accepted in the network.

The second group of algorithms we will consider are classified under the peer list algorithm. The first new algorithm we introduce is the Peer List based Algorithm (PL). In this algorithm, the peer-to-peer distribution network is represented by a peers list, in which each peer is characterized by an id, its available output bandwidth and an id list of children nodes. At the beginning of this algorithm, the peers list contains only the source node. When a new node N wants access to the network, this peer requests, to the peers of the network, an amount of bandwidth equal to the bandwidth required to acquire the multimedia content. If N obtains the required bandwidth, from the network, then the new node is added to the network; otherwise the network isn't able to accept more peers. The second algorithm we consider is the PL algorithm with Reconstruction (PLR). This algorithm is formulated from the PL algorithm. The PLR algorithm inserts the new node (N) in the network if and only if N is able to increase the residual bandwidth of the network. In this case the PLR algorithm extracts the peer (of the network) with minimum output bandwidth and replaces it with the new node. The PLR algorithm exploits the increase of the residual bandwidth (brought by N) by re-inserting the extracted node. If N isn't able to increase the network residual bandwidth then the PLR algorithm doesn't insert N into the network and the network accepts no more peers. The third algorithm we consider is the PLR Algorithm without input blockage. In this algorithm, if the new peer is not accepted in the network, this peer is inserted into a waiting queue. When a new node able to increase the residual output bandwidth of the network is inserted, this algorithm wakes up and tries to insert the waiting peers. A simple analytical formula for the maximum number of nodes accepted in a Peer List based network is:

$$n_{PL} = \left\lfloor B + \sum_{i=1}^{n_{PL}} \Omega_i \right\rfloor \tag{3.2}$$

where Ω_i is the output bandwidth of the i-th peer of the network. In this way from the formulas (3.1) and (3.2) it is immediately proof that $n_{PL} \geq n_T$.

In a Tiers based network the maximum depth is T. If we collect the unused bandwidth B_u of the tiers and if $B_u \geq 1$ we can supply one or more new peers. In this way, in a Tiers based network , if we supply one or more new peers using the unused bandwidth then the Tiers based network degenerates into a Peer List based network and in this case the maximum depth is $\geq T+1$. Thus $depth_{PL} \geq delpth_T$. Therefore the Peer List based networks are optimal with respect to the maximum number of nodes accepted by the network but they don't have the minimum delay in terms of the maximum depth.

In the TR, TR without input blockage, PLR and PLR without input blockage algorithms the insertion of a new peer can trigger a reconstruction of the network required in order to maintain order in the structure of the network. The reconstruction makes a delay. Thus the maximum delay of the network, in terms of the maximum depth of the network, has to be increased by the reconstruction delays.

3.3.2.1 Peer List based Algorithm (PL)

In the PL algorithm the peer-to-peer distribution network is represented by a peers list, where each peer is characterized by an id, its available output bandwidth and the id list of children nodes.

At the beginning, the peers list contains only the source node. When a new node N wants access to the network, this peer requests, to the peers of the network, an amount of bandwidth equal to the bandwidth required to acquire the multimedia content. If N obtains the required bandwidth, from the network, then the new node is added to the network; otherwise the network is not able to accept more peers.

The next algorithm is an improvement of this algorithm and it allows to increase the maximum number of peers accepted in the network.

3.3.2.2 Peer List based Algorithm with network Reconstruction (PLR)

The PLR algorithm is an evolution of the PL algorithm that that increase the maximum number of peers accepted in the network.

This algorithm has the same behaviour as the PL algorithm, when the network is able to accept a new peer otherwise it tries to insert this peer with a reconstruction of the network. The algorithm inserts the new node (N) in the network if and only if N is able to increase the residual bandwidth $(B+(\sum_{i=0}^{n}\Omega_i)-n$ where n is the number of peers of the network) of the network. In this case the algorithm extracts the peer (of the network) with minimum output bandwidth and replaces it with the new node. The algorithm exploits the increase of the residual bandwidth (brought by N) to re-insert the extracted node. If N is not able to increase residual bandwidth of the network then the algorithm does not insert N into the network and the network accepts no more peers. The PLR as well as TR algorithm may have network reconstruction.

The PLR algorithm has the analytical formulation (3.2) where the output bandwidths of the peers are $\Omega_1 \geq \Omega_2 \geq ... \geq \Omega_n > 0$ and n is the number of peers accepted by the network.

3.3.2.3 PLR algorithm without Input Blockage (PLWwIB)

The PLRwIB algorithm is an evolution of the PLR algorithm that that increase the maximum number of peers accepted in the network.

The PLRwIB algorithm derives the PLR algorithm. The PLRwIB is the PLR algorithm without input blockage. The engine of this algorithm is the following:

- if the new peer can be inserted into the network (with network reconstruction if it is necessary) then the algorithm performs the insertion operation as the PLR algorithm.

- Otherwise the new peer is inserted in a waiting queue; this queue contains the peers that are waiting a new node able to increase the residual output bandwidth of the network. When this event happens the algorithm wakes up the waiting peers and it tries to insert them. The peers, that are not inserted, are maintained in the waiting queue and they are waked up (by the insertion algorithm) if and only if a new peer, is able to increase the residual bandwidth of the network with its insertion.

The disadvantage of this algorithm is represented by the network reconstruction that introduces an additional delay to the network.

3.4 Asymmetric Channel

For the analysis of the peer-to-peer algorithms we introduced in section 3.3, we formulate a theoretical optimum in which the maximization of the average maximum number of the peers and the minimization of the average maximum delay of the network is achieved. We compare the results, in terms of the average maximum number of peers and the average maximum delay of the network, of the algorithms presented above, with respect to the theoretical optimum.

3.4.1 Maximum delay of a peer-to-peer distribution network

In this subsection we analyse the maximum delay of a peer-to-peer distribution network. To estimate the maximum delay of a peer-to-peer distribution network, we suppose to have two different cases:

1. in the first case, the network is generated by an algorithm (such as the T algorithm or the PL algorithm) that does not use a reconstruction of the network. In this case the maximum delay of the peer-to-peer distribution network is defined as the maximum depth of this network.

2. In the second case, the network is generated by an algorithm (such as the TR, TR-wIB, PLR and PLRwIB algorithms) that uses a reconstruction of the network. In this scenario the maximum delay of the peer-to-peer distribution network is defined as the maximum depth of this network plus the amount of the delays generated by each reconstruction of the network. For the insertion of a new peer N, we have a reconstruction of the network when it is necessary to extract a peer of the peer-to-peer network, replace it with the new peer N and the insertion algorithm exploits the increase of the residual bandwidth (brought by N) to re-insert the extracted node. The reconstruction delay for the insertion of a new node is the amount of replacement

delays. The delay produced by each k-th substitution of two peers (p_1 and p_2) is:

$$d_{s_k} = \sum_{j=1}^{M} \frac{S_{cp}}{B(p_1, j)} + \sum_{h=1}^{W} \frac{S_{cp}}{B(p_1, h)} \tag{3.3}$$

where $B(p_1, j)$ is the bandwidth between the extracted peer $p_1$and the j-th parent peer of p_1, M is the number of the parent peers of the peer p_1, $B(p_1, h)$ is the bandwidth between the extracted peer p_1 and the h-th child peer of p_1, W is the number of the child peers of the peer p_1 and the peer p_1 sends to its parent nodes and its child nodes a control packet (with size S_{cp}) used to perform the node replacement. Thus the reconstruction delay for the insertion of the i-th node is: $d_i = \sum_k d_{s_k}$ and $d_i = 0$ if the are no node replacements for the insertion of the i-th peer. Therefore the total reconstruction delay of a peer-to-peer network is:

$$d = \sum_{i=1}^{n} d_i \tag{3.4}$$

where n is the maximum number of peers accepted by the network.

3.4.2 Theoretical Optimum

In this subsection we formulate the theoretical optimum in which we maximize the average maximum number of peers accepted in the network and minimize the average maximum delay of the same peer-to-peer network.

The theoretical optimum is achieved when the ratio between the average maximum number of peers (n) and average minimum possible maximum delay (d) of the network is maximized. We indicate with, $\overline{n_1}$, the mean maximum number of peers accepted in the network with an output bandwidth between 1 and 2 ($1 \leq \Omega_i < 2$). We indicate with, $\overline{n_2}$, the mean maximum number of peers accepted in the network with an output bandwidth between 0 and 1 ($0 < \Omega_i < 1$). We have the average maximum number of peers in the network with the minimum possible value of T if and only if the peers (that access the network) are ordered with respect to their output bandwidth namely $\Omega_1 \geq \Omega_2 \geq \Omega_3 \geq ... \geq \Omega_n$, where Ω_i(for $i = 1...n$) is the output bandwidth of the i-th peer that has access the network. In this way there are no reconstructions of the peer-to-peer network and there are no reconstruction delays. With this conditions we can partially apply the theoretical formulation of the article [15] to achieve the optimum value of the ratio between n and T.

In this way the network is divided in two parts. In the first part there are all the peers with output bandwidth $1 \leq \Omega_i < 2$ (with average output bandwidth $\overline{\Omega^1}$) and they receive the multimedia content from the source ($\overline{n_1}$ number of peers and $\overline{T_1}$ the average minimum possible maximum delay of the first part of network). In the second part there are all the peers with output bandwidth $0 < \Omega_i < 1$ (with average output bandwidth $\overline{\Omega^2}$) and they receive the multimedia content from the leaf peers of the first part of the network ($\overline{n_2}$ number of peers and $\overline{T_2}$ the average minimum possible maximum delay of the first part of network).

The average maximum number of peers accepted in the network is $\overline{n} = \overline{n_1} + \overline{n*}$. When $\left(\sum_{i=1}^{\overline{n}} \Omega_i\right) / \overline{n} < 1$ the system reaches its maximum number of peers.

We suppose to have n_T number of peers wants to access the peer-to-peer network. Thus the number of peers of the first part of the network is:

$$\overline{n_1} = \lfloor n_T \cdot (1 - p) \rfloor \tag{3.5}$$

Where p is the probability that each i-th node has output bandwidth $0 < \Omega_i < 1$ and $1 - p$ is the probability that each i-th node has output bandwidth $1 \leq \Omega_i < 2$.

The average minimum possible maximum delay (formulation of the article [15] section III) in a $\overline{n_1}$- nodes (non necessarily tiered) peer-to-peer network fed by a source with bandwidth B is:

$$\overline{T_1} = \left\lceil log_{\overline{\Omega^1}} \left(\frac{\overline{n_1}}{B} \cdot \left(\overline{\Omega^1} - 1 \right) + 1 \right) \right\rceil \tag{3.6}$$

The formula (3.6) give us the average minimum maximum delay of the first part of the network, because it is achieved through the average maximum number of peers $\overline{n_1}$.

The output bandwidth provided by the first part of the network to the second part of the network is:

$$\overline{B_{N_1}} = B + \overline{\Omega^1} \cdot \overline{n_1} - \overline{n_1} = B + \overline{n_1} \cdot \left(\overline{\Omega^1} - 1 \right) \tag{3.7}$$

where B is the output bandwidth of the source.

The total residual bandwidth of the second part of the network is:

$$\overline{B_{N_1}} = \overline{B_{N_1}} + \overline{\Omega^2} \cdot \overline{n^*} - \overline{n^*} = \overline{B_{N_1}} + \overline{n^*} \cdot \left(\overline{\Omega^2} - 1 \right) \tag{3.8}$$

where $\overline{n^*}$ is the mean maximum number of peers accepted in the second part of the peer-to-peer network.

When $\overline{B_{N_2}} = 1$ the second part of the network reaches the average maximum number of peers – 1. Thus from the formula (3.8) the average maximum number of peers of the second part of the network is:

$$\overline{n^*} = \left\lfloor \frac{\overline{B_{N_1}} - 1}{1 - \overline{\Omega^2}} + 1 \right\rfloor \tag{3.9}$$

Therefore the average minimum possible maximum delay of the second part of the network (formula presented in section III of the article [15]) is:

$$\overline{T^*} = \left\lceil log_{\overline{\Omega^2}} \left(\frac{\overline{n^*}}{\overline{B_{N_1}}} \cdot \left(\overline{\Omega^2} - 1 \right) + 1 \right) \right\rceil \tag{3.10}$$

The formula (3.10) give us the average minimum maximum delay of the second part of the network, because it is achieved through the average maximum number of peers $\overline{n^*}$.

With the formula (3.9) the formula (3.10) becomes:

$$\overline{T^*} = \left\lceil log_{\overline{\Omega^2}}\left(\frac{1}{\overline{B_{N_1}}}\right) + 1 \right\rceil \tag{3.11}$$

In this way if we define:

$$\overline{n_2} = \overline{n^*} - 1 = \left\lfloor \frac{\overline{B_{N_1}} - 1}{1 - \overline{\Omega^2}} \right\rfloor \tag{3.12}$$

then the average minimum possible maximum delay (formulation of the article [15] section III) in a $\overline{n_2}$-nodes (non necessarily tiered) peer-to-peer network fed by an equivalent source with bandwidth $\overline{B_{N_1}}$ is:

$$\overline{T_2} = \begin{cases} 0 & if\, \overline{B_{N_1}} < 1 \\ 1 & if\, \overline{B_{N_1}} = 1 \\ \left\lceil log_{\overline{\Omega^2}}\left(\frac{1}{\overline{B_{N_1}}}\right) \right\rceil & if\, \overline{B_{N_1}} > 1 \end{cases} \tag{3.13}$$

The formula (3.13) give us the average minimum maximum delay of the second part of the network, because it is achieved through the average number of peers $\overline{n_2}$.

Thus $\overline{T_2} = \overline{T^*} - 1$ and $\overline{n_2} = \overline{n^*} - 1$. Using the formulas (3.9), (3.11), (3.12) and (3.13), we can simply show that: $(\overline{n_1} + \overline{n_2})/(\overline{T_1} + \overline{T_2}) \geq (\overline{n_1} + \overline{n^*})/(\overline{T_1} + \overline{T^*})$. In conclusion, if $\overline{B_{N_1}} > 1$, the optimum for the ratio between the average maximum number of peers and the average minimum possible maximum delay of the network is:

$$\frac{\overline{n_1} + \overline{n_2}}{\overline{T_1} + \overline{T_2}} = \frac{\lfloor n_T \cdot (1-p) \rfloor + \left\lfloor \frac{\overline{B_{N_1}} - 1}{1 - \overline{\Omega^2}} \right\rfloor}{\left\lceil log_{\overline{\Omega^1}}\left(\frac{\overline{n_1}}{B} \cdot \left(\overline{\Omega^1} - 1\right) + 1\right) \right\rceil + \left\lceil log_{\overline{\Omega^2}}\left(\frac{1}{\overline{B_{N_1}}}\right) \right\rceil} \tag{3.14}$$

3.4.3 Results

In this subsection we present the results about the algorithm which builds the peer-to-peer network characterized by the maximization of the average maximum number of peers accepted in the network and the minimization of the average maximum delay of the network.

The comparison of the algorithms is performed by using the ratio between the average maximum number of peers (n) and the average maximum delay (d) of the network over 1000 samples of peers. The simulator uses 1000 different samples (random generated) and each sample contains 1000 peers. We now briefly describe the network parameters followed when making the comparison of the performance of the algorithms we discussed about in section 3. The value of the output bandwidth of the source node is $B \in [1,10]$. p is the probability that each i-th node has output bandwidth $0 < \Omega_i < 1$ and $1-p$ is the probability that each i-th node has output bandwidth $1 \leq \Omega_i < 2$. We are supposed to have an uniform distribution for p, where $p \in [0,1]$. For each value of p between 0 and 1; with step of 0.01 in the simulation environment; the simulator uses 1000 different samples and each sample contains 1000 peers. We suppose that the size of the control packet used to replace a peer with a new peer is equal to 642 bits (where 192 bits are for the TCP header [76], 192 bits are for the IPv4 header [77], 96 bits of data make up of 32 bits for the IP address of the extracted peer, 16 bits for the port of the extracted peer, 32 bits for the IP address of the new peer, 16 bits for the port of the new peer and 162 bits for the lower layers). The simulation results give a map of the best algorithms with respect to the ratio between the average maximum number of peers and the average maximum delay of the network as functions in term of p and B. The space B, p; with $1 \leq B \leq 10$ and $0 \leq p \leq 1$; is divided in three areas.

The first area has $1 \leq B \leq 2$ and $0 \leq p \leq 1$. In this area the PLR algorithm without input blockage is closest to the optimum because it produces random trees (with $n/d > 1$) while all the Tier based algorithms produce networks that are chains of peers (with $n/d = 1$).

The second area has $2 \leq B \leq 10$ and $0.46 \leq p \leq 1$. In this area the best algorithm is the TR algorithm without Input Blockage.

The third area has $2 \leq B \leq 10$ and $0 \leq p < 0.46$. In this area the best algorithm is the PLR algorithm without Input Blockage.

The summary of the best algorithm for each area is presented in table 3.1.

The confidence intervals of n/d (with p respect to B and) have been evaluated for

	$1 \leq B \leq 2$	$2 \leq B \leq 10$
$0 \leq p < 0.46$	PLRwIB Algorithm	PLRwIB Algorithm
$0.46 \leq p \leq 1$	PLRwIB Algorithm	TRwIB Algorithm

Table 3.1: Summary of the best algorithm for each Area

each algorithm and they have a maximum size of $4.6 \cdot 10^{-3}$, thus they are negligible in this approach.

In the figures 3.1 and 3.2 we depict the simulation results described above.

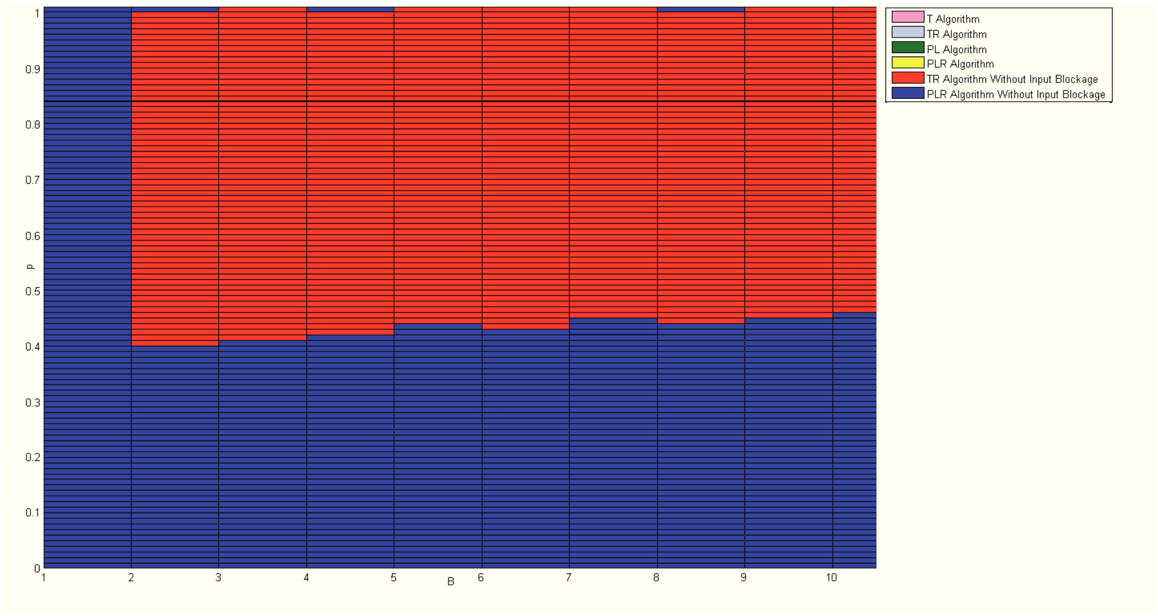

Figure 3.1: Simulation results for the asymmetric channel. Ratio between the average maximum number of peers (n) and the average maximum delay (d).

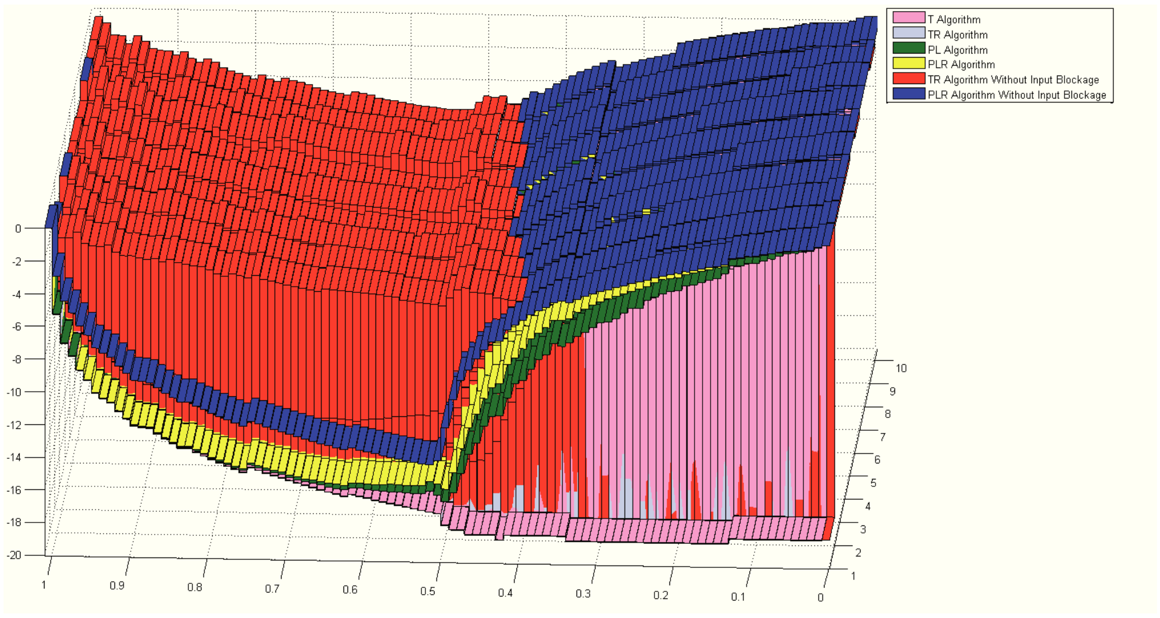

Figure 3.2: Simulation results for the asymmetric channel. 3D view of the results.

3.5 Radio Channel

We now briefly analyse the behaviour of the algorithms described above in a simple radio channel characterization; moreover the algorithm with the maximum percentage of bits correctly received is established.

3.5.1 First radio channel characterization : "Dependent error probabilities over the received bits"

This subsection describes a simple radio channel characterization. Each wireless link between nodes is represented as an ideal wireless link with the following characteristics: the error probabilities over received bits are not independent (in the previous article [78] the error probabilities over received bits were independent). The average bit error probability with respect to small scale fading effects and coherent four phase PSK modulation is given as (Pages 785-486 formulas 14-4-36 and 14-4-38 of the book [79]):

$$P_b = \frac{1}{2}\left[1 - \frac{\mu}{\sqrt{2-\mu^2}}\sum_{k=0}^{L-1}\begin{pmatrix} 2k \\ k \end{pmatrix}\left(\frac{1-\mu^2}{4-2\mu^2}\right)^k\right] \tag{3.15}$$

Where L is the order of diversity (for our channel L=4) and μ is the cross-correlation coefficient with perfect estimation given as (page 786 formula 14-4-36 or table C-1 page 894 of [79]):

$$\mu = \sqrt{\frac{\overline{\gamma_c}}{1+\overline{\gamma_c}}} \tag{3.16}$$

Where $\overline{\gamma_c}$ is the average Signal to Noise Ratio with respect to small scale fading effects.

3.5.2 Second radio channel characterization: "Independent error probabilities over the received bits"

The first model presented is characterized by an independent error probabilities over the received bits.

This subsection describes a simple radio channel characterization. Each wireless link between nodes is represented as an ideal wireless link with the following characteristics: the average bit error probability with respect to small scale fading effects and BPSK modulation is given in (pages 461 and 466 of [79]):

$$P_b \cong \frac{2 \cdot t + 1}{n} \cdot P_{cw} \tag{3.17}$$

where: n represents the number of bit of each codeword; t represents the maximum number of bit that the receiver is able to detect and correct, and P_{cw} represents the average code word error probability with respect to small scale fading effects (page 465 of [79]):

$$P_{cw} = 1 - \sum_{i=0}^{t} \left(\begin{array}{c} n \\ i \end{array} \right) \cdot P_{b_{gross}}^{i} \cdot \left(1 - P_{b_{gross}}\right)^{n-i} \tag{3.18}$$

where $P_{b_{gross}}$ represents the average bit error probability with respect to small scale fading effects in the uncoded system (page 774 of [79]):

$$P_{b_{gross}} = \frac{1}{2} \cdot \left(1 - \sqrt{\frac{\overline{\gamma_c}}{1+\overline{\gamma_c}}}\right) \tag{3.19}$$

$\overline{\gamma_c}$ is the average Signal to Noise Ratio with respect to small scale fading effects.

3.5.3 Analytical Model

In this subsection we present an analytical model for the bit error probability in a peer-to-peer network on a generic radio channel.

In this subsection we present an analytical formulation [78] used to establish the bit error probability of each node of the network, produced by the previous algorithms. The main hypothesis used for this analytical model (and used in the simulator presented in the next section) are as follows:

- stationary network: during the simulation the system does not insert new nodes in the network because the aim is to estimate the network behaviour with an unreliable radio channel.
- Each segment sent by a peer i to another peer j has a constant and fixed dimension $d_{i,j}$.
- Each peer has one or more parent nodes from which it obtains the content; the content (namely the packet) is distributed among the parent peers with a static allocation, for example each peer receives the first segment of each packet from the first parent node, ..., each peer receives the n-th segment of each packet from the n-th parent node and so on.

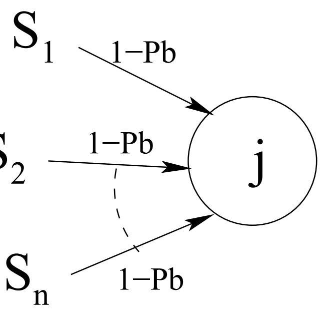

Figure 3.3: j-th node of a peer-to-peer network

- Each peer is identified by a unique peer ID; the peer ID of the source node is 0 and the network peers have incremental peer ID value starting from 1.
- The source node has each packet and transmits it to the peers directly connected to source node.
- The analytical formulation and the simulator considers only the uncoded communication between peers and the probability P_b is the average (with respect to small scale fading effects) bit error probability on decoded word. In this way if there is an error on a bit in the considered decoded segment then the entire segment is lost.

Consider the j-th node of the network:

Suppose that the packet is divided in n segments and these are obtained from different parent nodes. So the j-th node receives the segments $S_1 \dots S_n$ of the packet from n different nodes. Each segment S_i (where $i = 1 \dots n$) has g_i bits and suppose that these bits have different bit error probability namely, the first bit (b_1) of the segment S_i has a bit error probability equal P_{b_1} ... the g_i-th bit (b_{g_i}) of the segment S_i has a bit error probability equal$P_{b_{g_i}}$. In this way for each bit, the correct bit probability is: for the first bit is $P_1 = 1 - P_{b_1}$ $\dots$ for the g_i-th bit is $P_{g_i} = 1 - P_{b_{g_i}}$. Now we have to establish the probability that the segment S_i is received correctly. A segment is correct if all the bits of this segment are received without errors. So the desired probability has the following expression:

$$P(S_i) = P_1 \cdot \ldots \cdot P_{g_i} \cdot (1 - P_b)^{g_i} \tag{3.20}$$

where P_b is the bit error probability of the radio channel and $i = 1 \ldots n$. Therefore the average correct bit probability for the j-th node is:

$$\overline{P}(j) = \frac{\sum_{i=1}^{n} P(S_i) \cdot g_i}{\sum_{i=1}^{n} g_i} \tag{3.21}$$

This formula give us the wireless link model for each node of the network. Moreover, for the nodes directly connected to the source, the probabilities $P_1 \ldots P_{g_i}$ for the segment S_i (where $i = 1 \ldots n$) have the following value:

$$P_1 = P_2 = \ldots = P_{g_i} = 1 \tag{3.22}$$

The bit error probability P_b of the radio channel, used in this subsection, is obtained through the formulas (3.15) and (3.16) with $\overline{\gamma_c}$ equal to the desired SNR.

3.5.4 Peer-to-Peer Network Simulator on an Unreliable Channel

In this subsection we present the simulator of the peer-to-peer network on an unreliable channel.

Each peer-to-peer network is simulated in the following way [78]: for each packet transmitted by the source node S, the simulator analyses the peers in the order defined by their peer ID; for each i-th peer (where $i = 1 \ldots n$), the simulator performs the following operation: it searches the parent nodes of the i-th peer (we indicate this node with N). For each parent node N_f (N_f is the source node if N receives the packet from S), the simulator determines if N_f has the segment of the packet expected by N:

- if N_f has the segment then the simulator determines if N receives it without errors; this is done, whilst simulating the behaviour of the channel for each segment bit sent from N_f to N: the system generates a random number v uniformly distributed in [0,1); with this number the simulator establishes if the bit is lost or is correctly received. The bit is lost if $0 \leq v \leq P_b$. The bit is correctly received if $v > P_b$; where the parameter P_b is obtained through the formulas (3.15) and (3.16) with $\overline{\gamma_c}$ equal to the desired SNR. If the number of lost bits of the segment is greater than 0 then the entire segment is lost and therefore the simulator adds the number of bits of the segment to the number of bits lost by N. Otherwise the segment is correctly received and therefore the simulator adds the number of bits of the segment to the number of bits correctly received by N.

- If N_f does not have the segment, then the simulator adds the number of bits of the segment to the number of bits lost by N.

At the end of the simulation for each peer the system produces the number of the bits correctly received and the number of the bits lost.

3.5.5 Model Validation through the simulator

Here we present the validation of the model through the simulation results using the autocorrelation test.

In order to validate the model of the network in an unreliable environment (radio channel) we use the autocorrelation test (pages 423-426 of the Book [80]). We define the residuals $\varepsilon(t)$ as:

$$\varepsilon(t) = y(t) - \hat{y}(t) \tag{3.23}$$

where $y(t)$ are the simulated results about the average percentage of correctly received bits for each depth t of the network and $\hat{y}(t)$ are the results produced by the model.

If the model is accurately describing the observed data $y(t)$, then the residuals $\varepsilon(t)$ should be white. A way to validate the model is thus to test the hypotheses:

- H_0 : $\varepsilon(t)$ is a white sequence;
- H_1 : $\varepsilon(t)$ is not a white sequence.

The autocovariance of the residuals $\varepsilon(t)$ is estimated as:

$$\hat{r}_\varepsilon(\tau) = \frac{1}{N} \cdot \sum_{t=1}^{N-\tau} \varepsilon(t+\tau) \cdot \varepsilon(t) \tag{3.24}$$

where N is the maximum depth of the peer-to-peer distribution network.

If H_0 holds, then the square covariance estimates is asymptotically χ^2 distributed namely:

$$\frac{N}{\hat{r}^2_\varepsilon(0)} \cdot \sum_{i=1}^{m} \hat{r}^2_\varepsilon(i) \longrightarrow \chi^2(m) \tag{3.25}$$

where m is the number of degrees of freedom and it is equal to the maximum depth of the peer-to-peer distribution network.

Let x denote a random variable which is χ^2 distributed with m degrees of freedom. Furthermore, we define $\chi^2_\alpha(m)$ by:

$$\alpha = P(x > \chi^2_\alpha(m)) \tag{3.26}$$

For $\alpha = 0.01$ we have:

- if $\frac{N}{\hat{r}^2_\varepsilon(0)} \cdot \sum_{i=1}^{m} \hat{r}^2_\varepsilon(i) > \chi^2_\alpha(m)$ then we reject H_0.
- if $\frac{N}{\hat{r}^2_\varepsilon(0)} \cdot \sum_{i=1}^{m} \hat{r}^2_\varepsilon(i) \le \chi^2_\alpha(m)$ then we accept H_0.

We can see this property through the normalized covariance $\hat{r}_\varepsilon(\tau)/\hat{r}_\varepsilon(0)$. In this case $x_\tau = \hat{r}_\varepsilon(\tau)/\hat{r}_\varepsilon(0)$ and we plot (for each peer-to-peer algorithm, in the worst case, SNR = 4 dB) x_τ versus τ and a 99% confidence interval for x_τ.

Since $x_\tau \longrightarrow \mathbb{N}(0, 1/N)$ the lines in the diagram are drawn at $x = \pm 2.5758/\sqrt{N}$:

- it can be seen from the figures 3.4 - 3.9 (for all the peer-to-peer algorithms and in the radio channel characterized by dependent error probabilities over the received bits) that x_τ lies in this interval.

- It can be seen from the figures 3.10 - 3.15 (for all the peer-to-peer algorithms and in the radio channel characterized by independent error probabilities over the received bits) that x_τ lies in this interval.

One can hence expect $\varepsilon(t)$ is a white process for all the peer-to-peer algorithms.

The result of the hypotheses test for each peer-to-peer algorithm in the radio channel characterized by dependent error probabilities over the received bits is:

- T algorithm: the test quantity (3.25) is 17.5213 and $\chi^2_\alpha(m)$ is 24.7250 thus the variable $\varepsilon(t)$ is, under the null hypothesis H_0, approximately $\chi^2_\alpha(m)$.

- TR algorithm: the test quantity (3.25) is 14.0130 and $\chi^2_\alpha(m)$ is 16.8119 thus the variable $\varepsilon(t)$ is, under the null hypothesis H_0, approximately $\chi^2_\alpha(m)$.

- TRwIB algorithm: the test quantity (3.25) is 16.7519 and $\chi^2_\alpha(m)$ is 21.6660 thus the variable $\varepsilon(t)$ is, under the null hypothesis H_0, approximately $\chi^2_\alpha(m)$.

- PL algorithm: the test quantity (3.25) is 27.8567 and $\chi^2_\alpha(m)$ is 29.1412 thus the variable $\varepsilon(t)$ is, under the null hypothesis H_0, approximately $\chi^2_\alpha(m)$.

- PLR algorithm: the test quantity (3.25) is 57.1550 and $\chi^2_\alpha(m)$ is 63.6907 thus the variable $\varepsilon(t)$ is, under the null hypothesis H_0, approximately $\chi^2_\alpha(m)$.

- PLRwIB algorithm: the test quantity (3.25) is 154.0808 and $\chi^2_\alpha(m)$ is 180.7009 thus the variable $\varepsilon(t)$ is, under the null hypothesis H_0, approximately $\chi^2_\alpha(m)$.

The result of the hypotheses test for each peer-to-peer algorithm in the radio channel characterized by independent error probabilities over the received bits is:

- T algorithm: the test quantity (3.25) is 18.6184 and $\chi^2_\alpha(m)$ is 24.7250 thus the variable $\varepsilon(t)$ is, under the null hypothesis H_0, approximately $\chi^2_\alpha(m)$.
- TR algorithm: the test quantity (3.25) is 14.3621 and $\chi^2_\alpha(m)$ is 16.8119 thus the variable $\varepsilon(t)$ is, under the null hypothesis H_0, approximately $\chi^2_\alpha(m)$.
- TRwIB algorithm: the test quantity (3.25) is 15.1688 and $\chi^2_\alpha(m)$ is 16.8119 thus the variable $\varepsilon(t)$ is, under the null hypothesis H_0, approximately $\chi^2_\alpha(m)$.
- PL algorithm: the test quantity (3.25) is 11.6634 and $\chi^2_\alpha(m)$ is 18.4753 thus the variable $\varepsilon(t)$ is, under the null hypothesis H_0, approximately $\chi^2_\alpha(m)$.
- PLR algorithm: the test quantity (3.25) is 14.6460 and $\chi^2_\alpha(m)$ is 16.8119 thus the variable $\varepsilon(t)$ is, under the null hypothesis H_0, approximately $\chi^2_\alpha(m)$.
- PLRwIB algorithm: the test quantity (3.25) is 171.5154 and $\chi^2_\alpha(m)$ is 180.7009 thus the variable $\varepsilon(t)$ is, under the null hypothesis H_0, approximately $\chi^2_\alpha(m)$.

In this case for all the peer-to-peer algorithms described above we observe that the prediction error $\varepsilon(t)$ is white with a level of significance $\alpha = 0.01$ thus the model is validated for all the algorithms and for both the radio channel characterizations.

3.5.6 Results

In this subsection we present the results. We establish the algorithms that have the highest correctly received bits percentage.

The fundamental parameter used to analyze and compare the behaviour of the six types of peer-to-peer networks is represented by average percentage of correctly received bits as a function of depth level of the network. Through the simulation results, figures 3.16 - 3.21 (for the radio channel characterized by dependent error probabilities over the received bits)

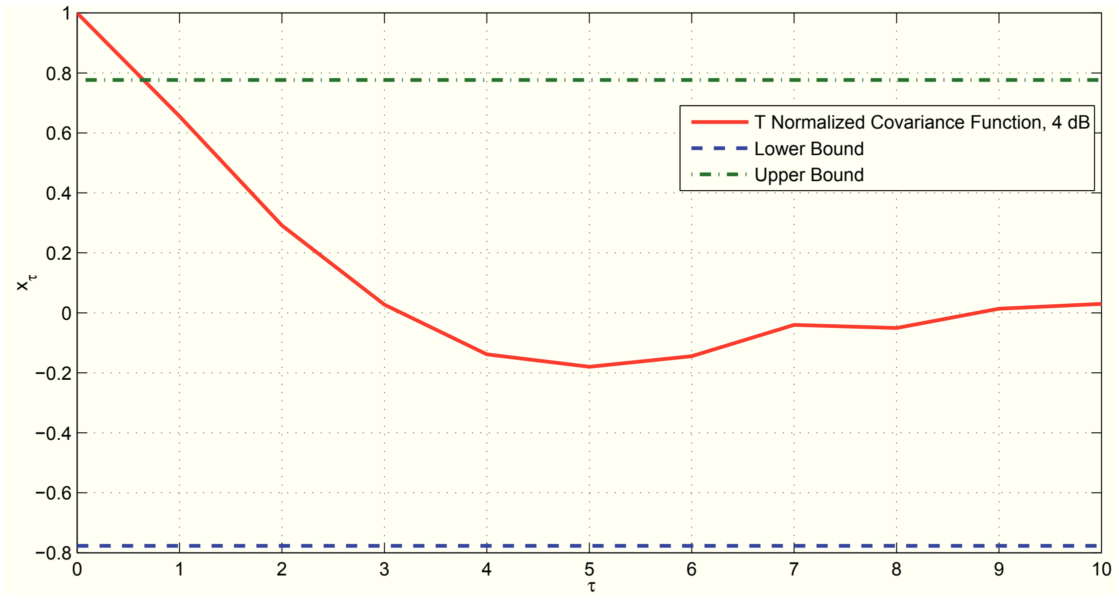

Figure 3.4: Normalized covariance function of $\varepsilon(t)$ for the T algorithm in the worst condition (SNR = 4 dB), in the radio channel characterized by dependent error probabilities over the received bits

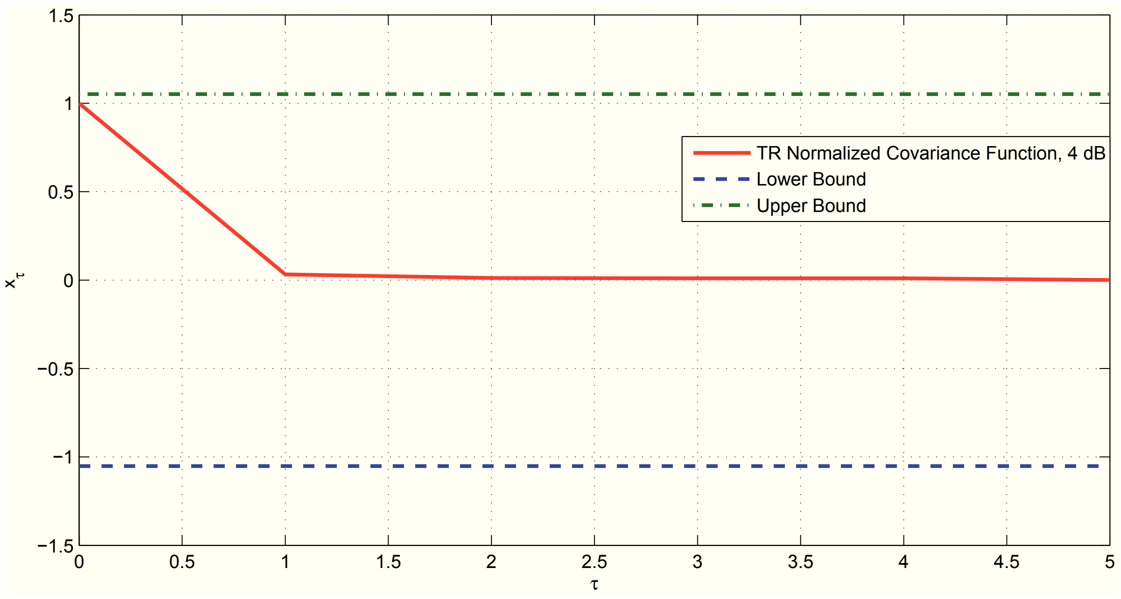

Figure 3.5: Normalized covariance function of $\varepsilon(t)$ for the TR algorithm in the worst condition (SNR = 4 dB), in the radio channel characterized by dependent error probabilities over the received bits

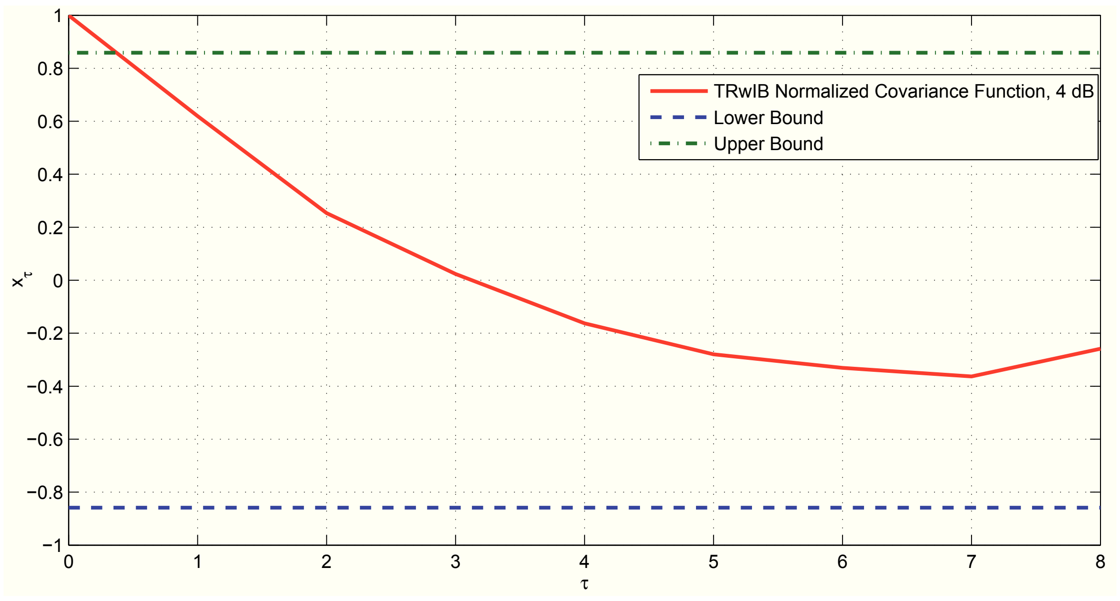

Figure 3.6: Normalized covariance function of $\varepsilon(t)$ for the TRwIB algorithm in the worst condition (SNR = 4 dB), in the radio channel characterized by dependent error probabilities over the received bits

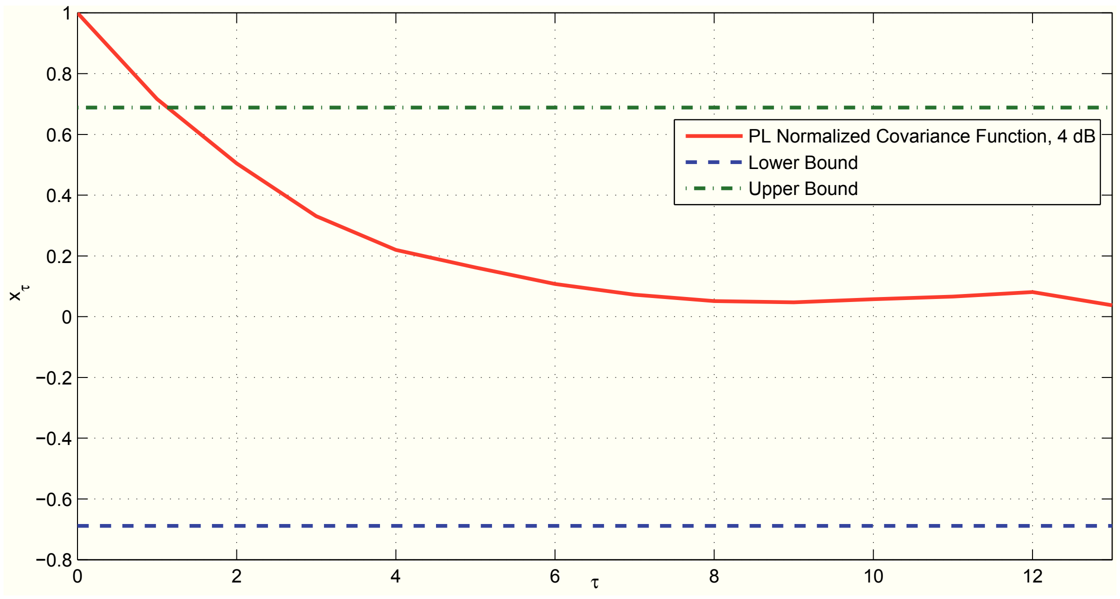

Figure 3.7: Normalized covariance function of $\varepsilon(t)$ for the PL algorithm in the worst condition (SNR = 4 dB), in the radio channel characterized by dependent error probabilities over the received bits

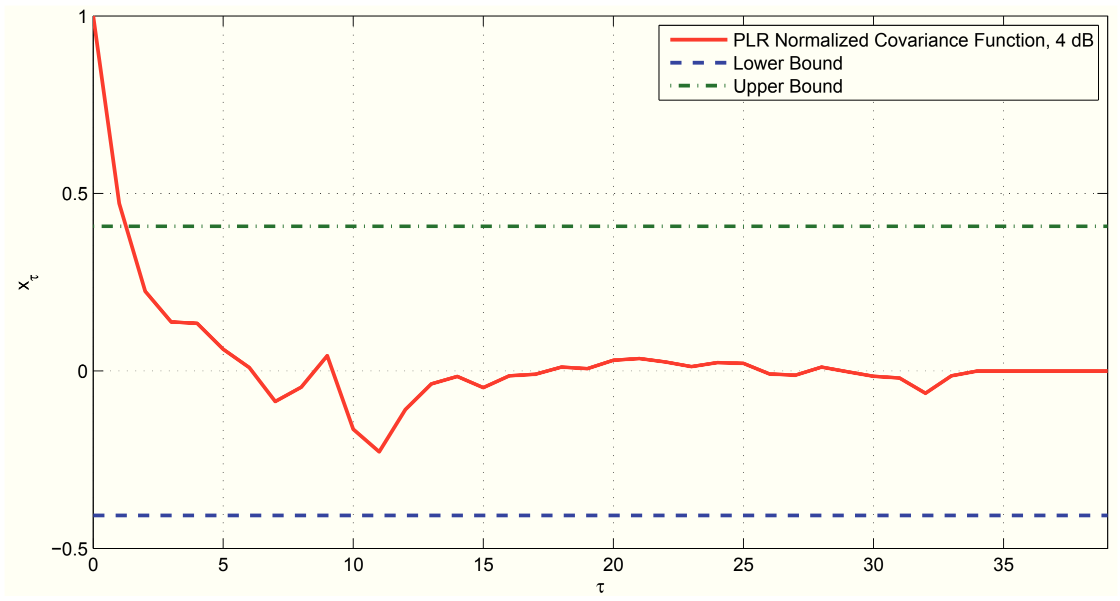

Figure 3.8: Normalized covariance function of $\varepsilon(t)$ for the PLR algorithm in the worst condition (SNR = 4 dB), in the radio channel characterized by dependent error probabilities over the received bits

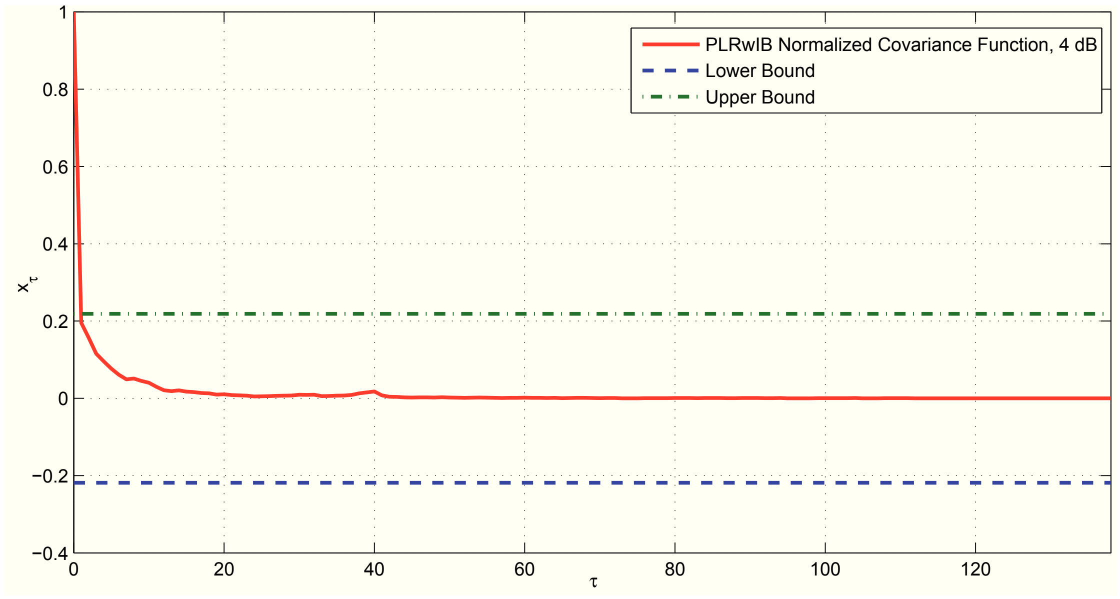

Figure 3.9: Normalized covariance function of $\varepsilon(t)$ for the PLRwIB algorithm in the worst condition (SNR = 4 dB), in the radio channel characterized by dependent error probabilities over the received bits

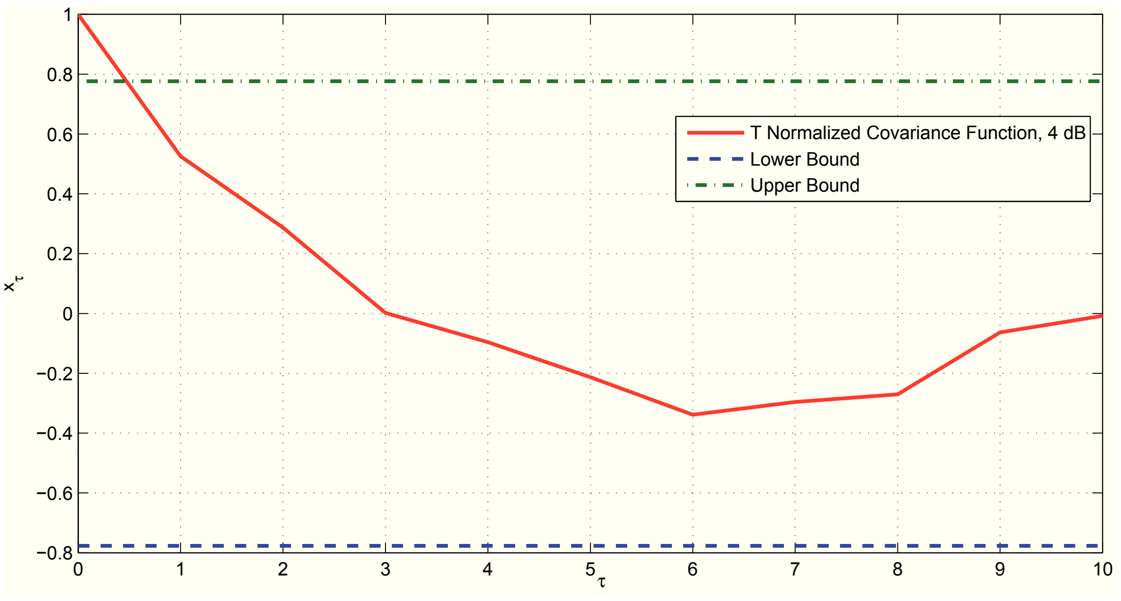

Figure 3.10: Normalized covariance function of $\varepsilon(t)$ for the T algorithm in the worst condition (SNR = 4 dB), in the radio channel characterized by independent error probabilities over the received bits

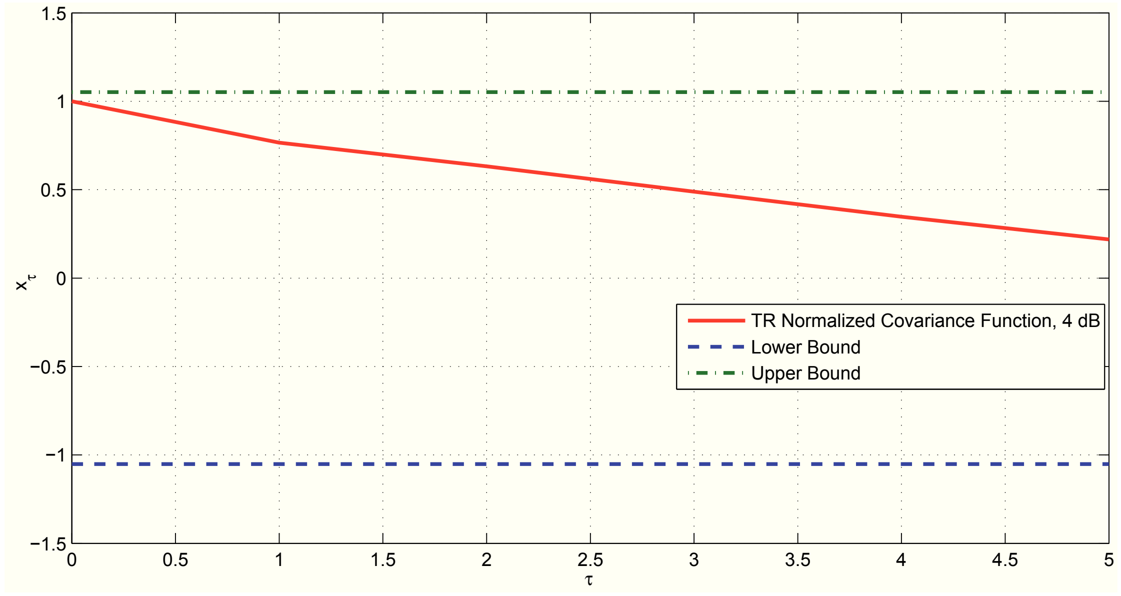

Figure 3.11: Normalized covariance function of $\varepsilon(t)$ for the TR algorithm in the worst condition (SNR = 4 dB), in the radio channel characterized by independent error probabilities over the received bits

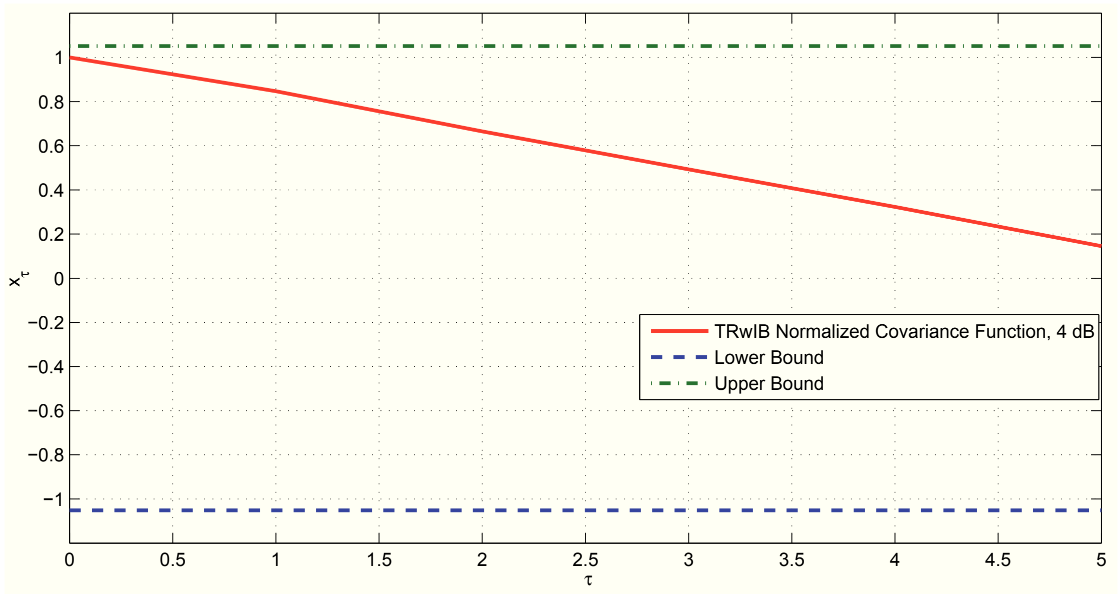

Figure 3.12: Normalized covariance function of $\varepsilon(t)$ for the TRwIB algorithm in the worst condition (SNR = 4 dB), in the radio channel characterized by independent error probabilities over the received bits

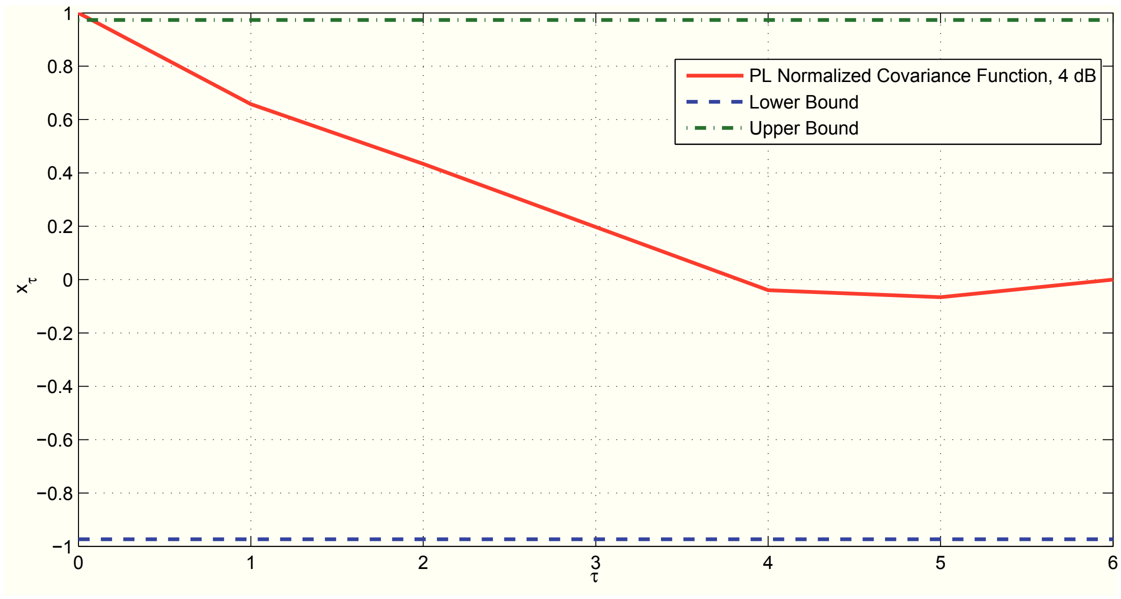

Figure 3.13: Normalized covariance function of $\varepsilon(t)$ for the PL algorithm in the worst condition (SNR = 4 dB), in the radio channel characterized by independent error probabilities over the received bits

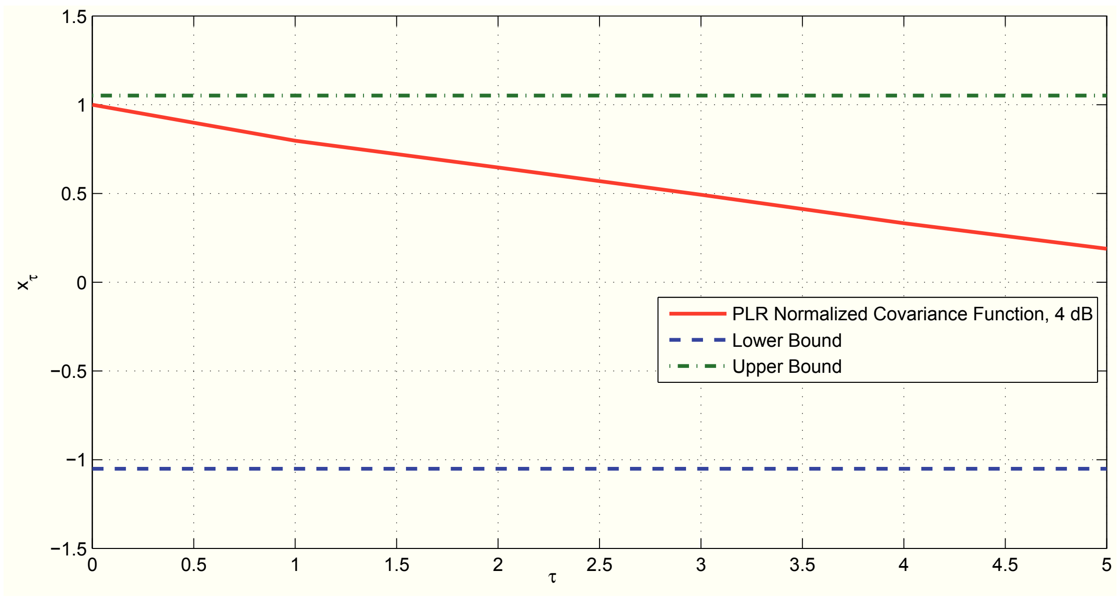

Figure 3.14: Normalized covariance function of $\varepsilon(t)$ for the PLR algorithm in the worst condition (SNR = 4 dB), in the radio channel characterized by independent error probabilities over the received bits

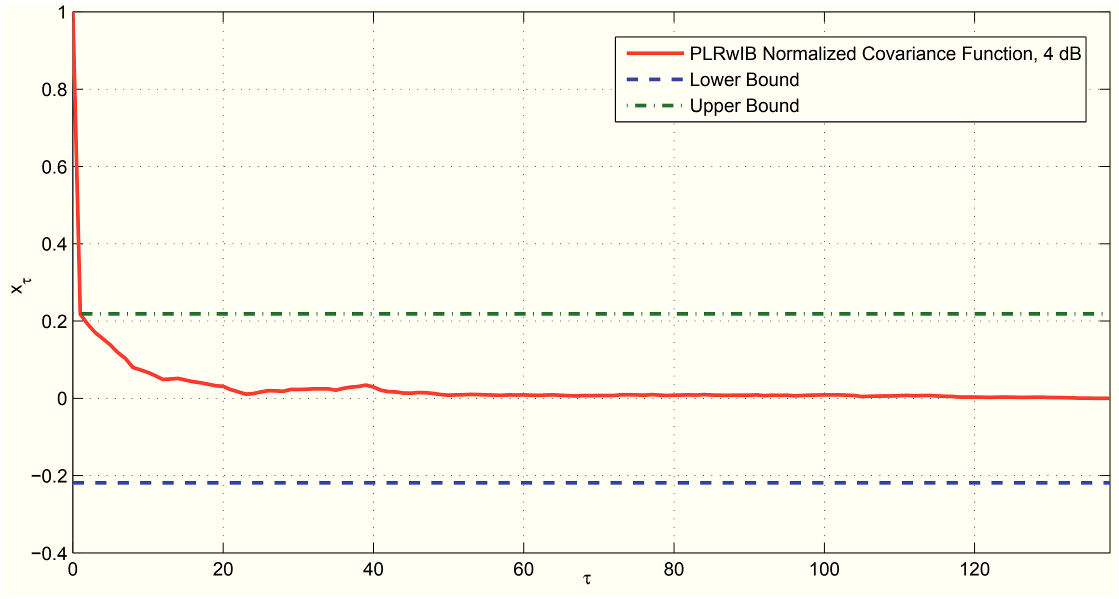

Figure 3.15: Normalized covariance function of $\varepsilon(t)$ for the PLRwIB algorithm in the worst condition (SNR = 4 dB), in the radio channel characterized by independent error probabilities over the received bits

Radio Channel	$d = 1$	$d = 2$	$d = 3$	$d \geq 3$
Independent	PL Network	PL Network	PLR and PLRwIB Network	TR Network
Dependent	All the Network	T, PL, PLR and PLRwIB Network	TR, PL, PLR and PLRwIB Network	TR Network

Table 3.2: Summary of the best behaviour with respect to the average percentage of correctly received bits, where d is the depth level of the peer-to-peer network

and figures 3.22 - 3.27 (for the radio channel characterized by independent error probabilities over the received bits), we observe that by increasing the parameter of SNR (Signal to Noise Ratio) this produces an increase of the percentage of bits correctly received by each node of the network. Figures 3.28, 3.29 and 3.30 depict the comparisons of peer-to-peer networks under the six different types of algorithms we considered in section 3, with respect to the percentage of bits correctly received by each node with SNR = 4 dB, 7 dB and 10 dB in the radio channel characterized by dependent error probabilities over the received bits. In this case the comparison parameter is the average percentage of correctly received bits as a function of depth level of the network. The best behaviour with respect to the average percentage of correctly received bits is obtained in the network generated by:

- the TR algorithm when the depth level is greater or equal to 4.
- The PLR algorithm and PLR algorithm without Input Blockage when the depth level is equal to 3.
- The PL algorithm when the depth level is less or equal to 2.

The summary of the best behaviour, in a radio channel characterized by dependent error probabilities over the received bits, with respect to the average percentage of correctly received bits is presented in the table 3.2.

Figures 3.31, 3.32 and 3.33 depict the comparisons of peer-to-peer networks under the six different types of algorithms we considered in section 3, with respect to the percentage of bits correctly received by each node with SNR = 4 dB, 7 dB and 10 dB in the radio channel characterized by independent error probabilities over the received bits. In this case the comparison parameter is the average percentage of correctly received bits as a function of depth level of the network. The best behaviour with respect to the average percentage of correctly received bits is obtained in the network generated by:

- the TR algorithm when the depth level is greater or equal to 4.
- The TR, PL, PLR and PLRwIB algorithms when the depth level is equal to 3.
- The T, PL, PLR and PLRwIB algorithms when the depth level is less or equal to 2.
- All the algorithms when the depth level is equal to 1.

The summary of the best behaviour, in a radio channel characterized by independent error probabilities over the received bits, with respect to the average percentage of correctly received bits is presented in the table 3.2.

All the results, presented in this section, have been obtained by the following configuration parameters: number of bits supplied by the source node equal to 2048 Kbits divided in packets characterized by a length equal to 128 bits; we use the same sequences of peers, that require access to the network; dimension of each codeword is 16 bits and the number of bits that the receiver is able to detect and correct is 4 bits.

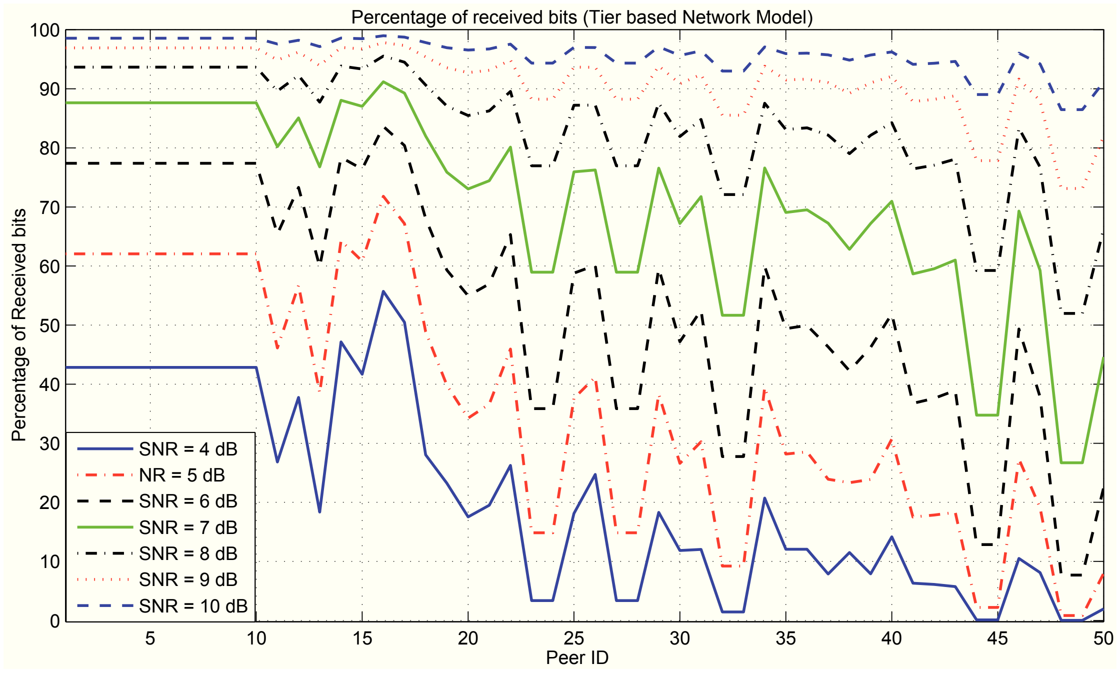

Figure 3.16: T algorithm results in a radio channel characterized by dependent error probabilities over the received bits

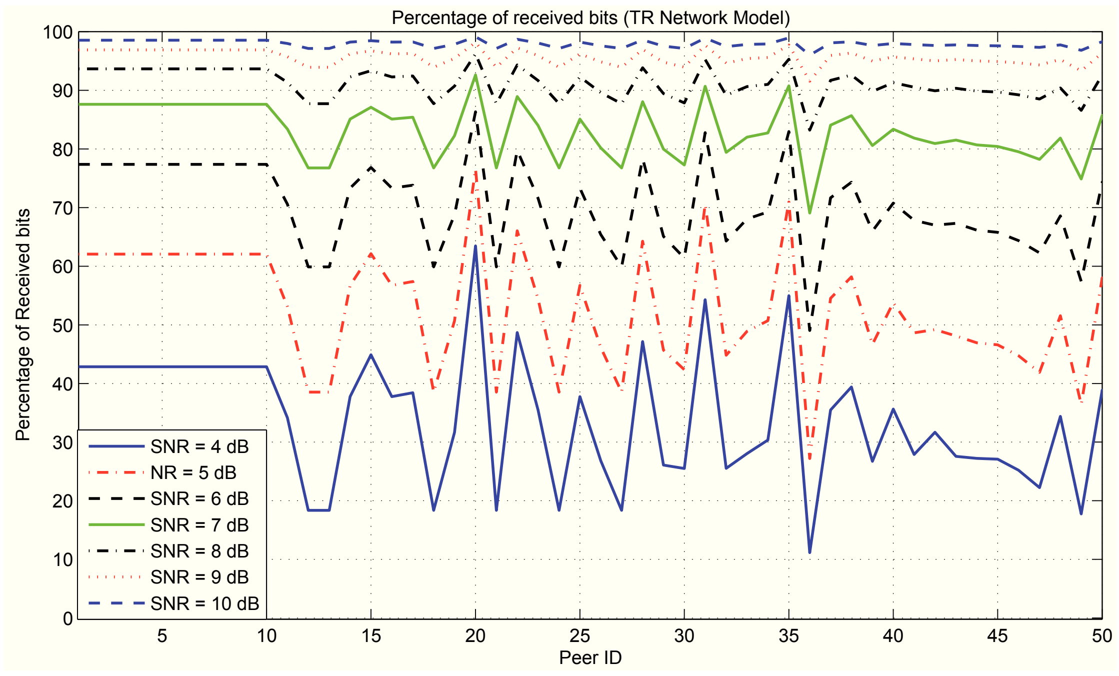

Figure 3.17: TR algorithm results in a radio channel characterized by dependent error probabilities over the received bits

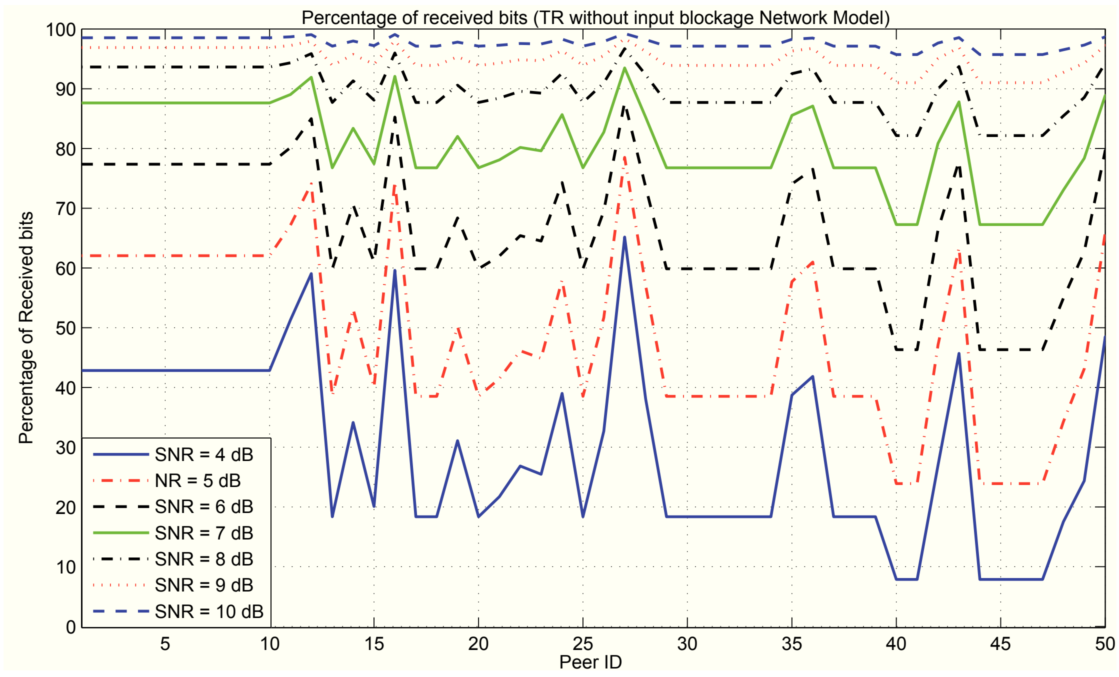

Figure 3.18: TRwIB algorithm results in a radio channel characterized by dependent error probabilities over the received bits

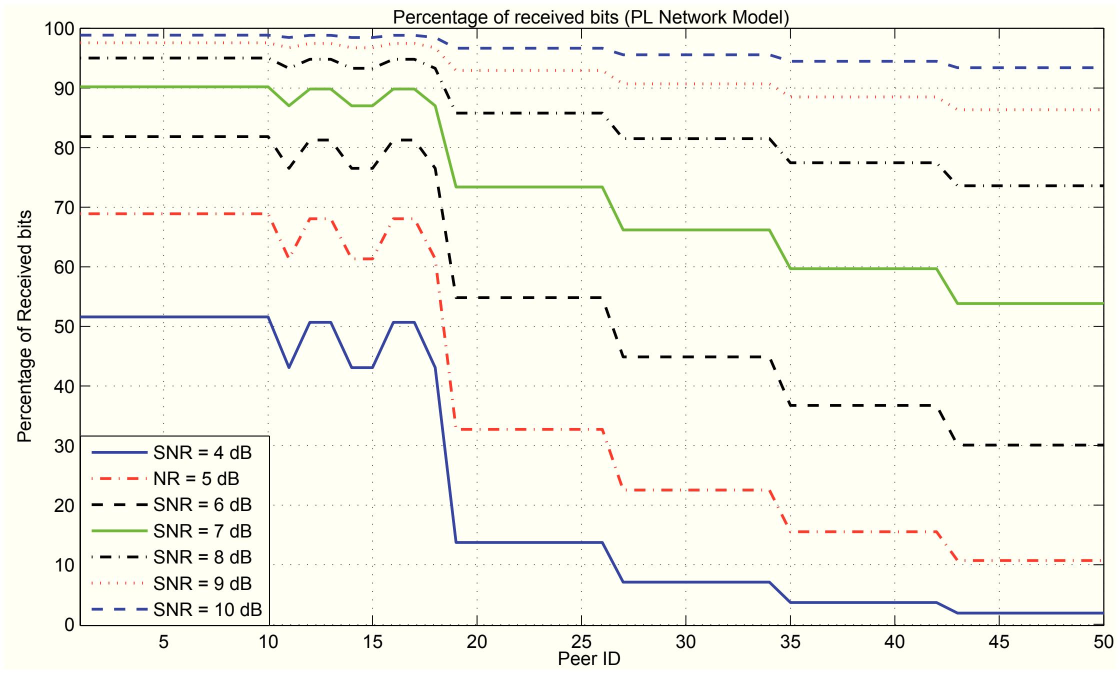

Figure 3.19: PL algorithm results in a radio channel characterized by dependent error probabilities over the received bits

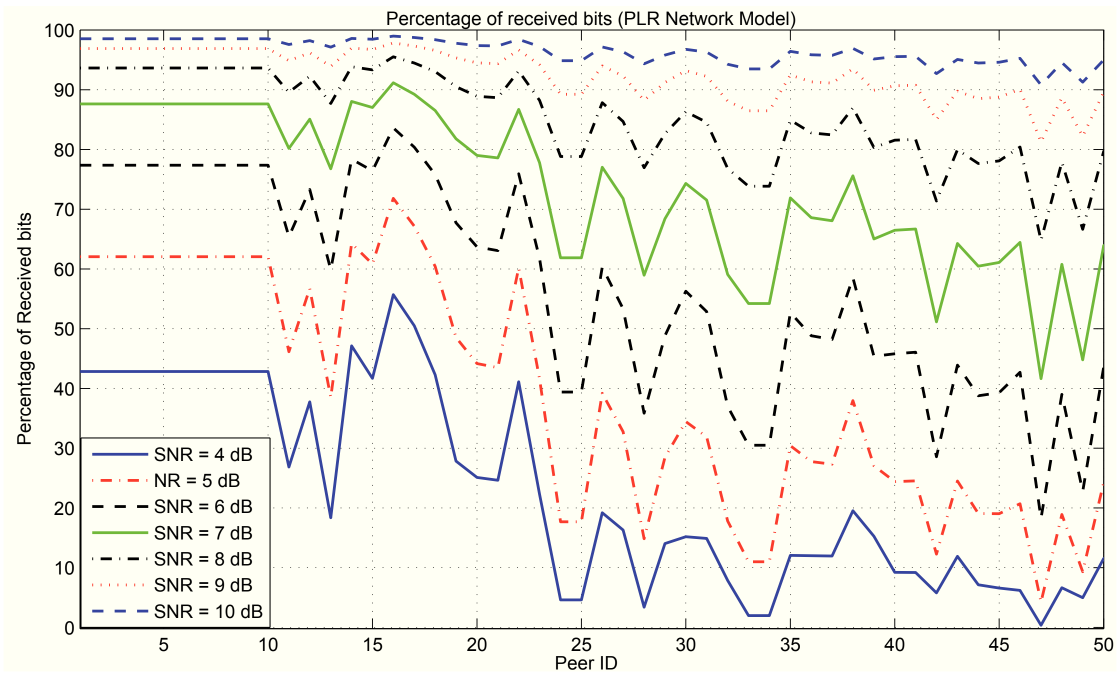

Figure 3.20: PLR algorithm results in a radio channel characterized by dependent error probabilities over the received bits

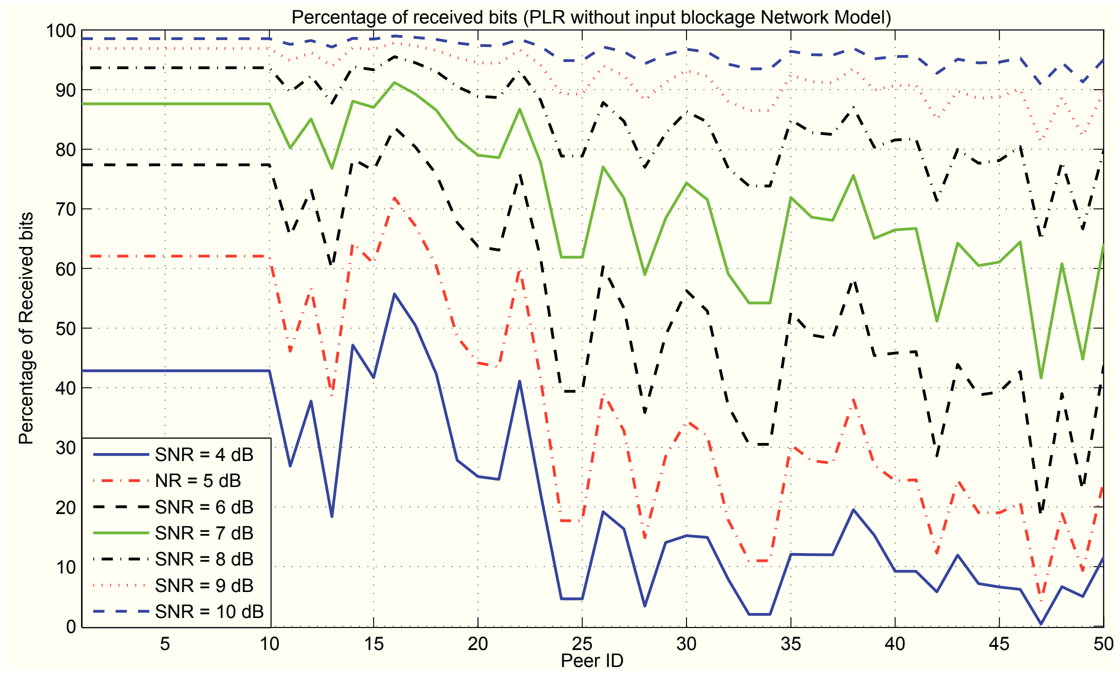

Figure 3.21: PLRwIB algorithm results in a radio channel characterized by dependent error probabilities over the received bits

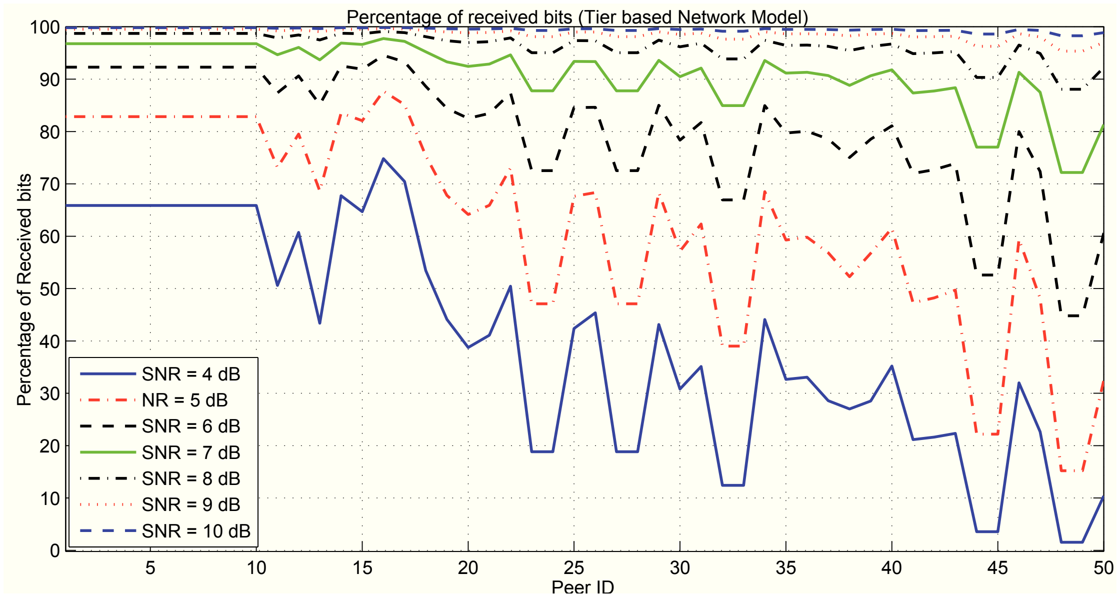

Figure 3.22: T algorithm results in a radio channel characterized by independent error probabilities over the received bits

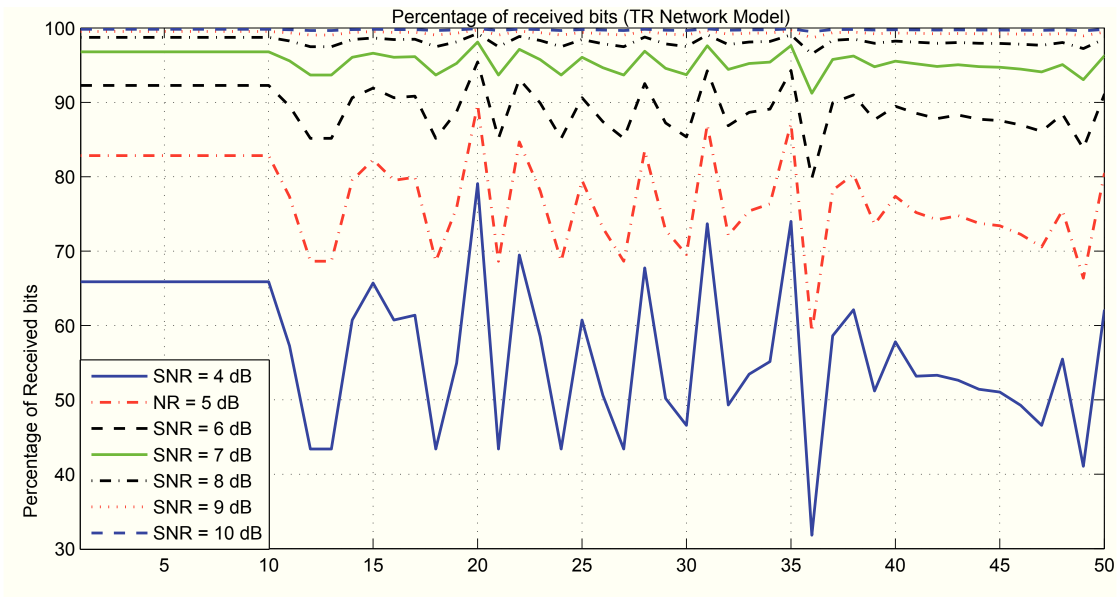

Figure 3.23: TR algorithm results in a radio channel characterized by independent error probabilities over the received bits

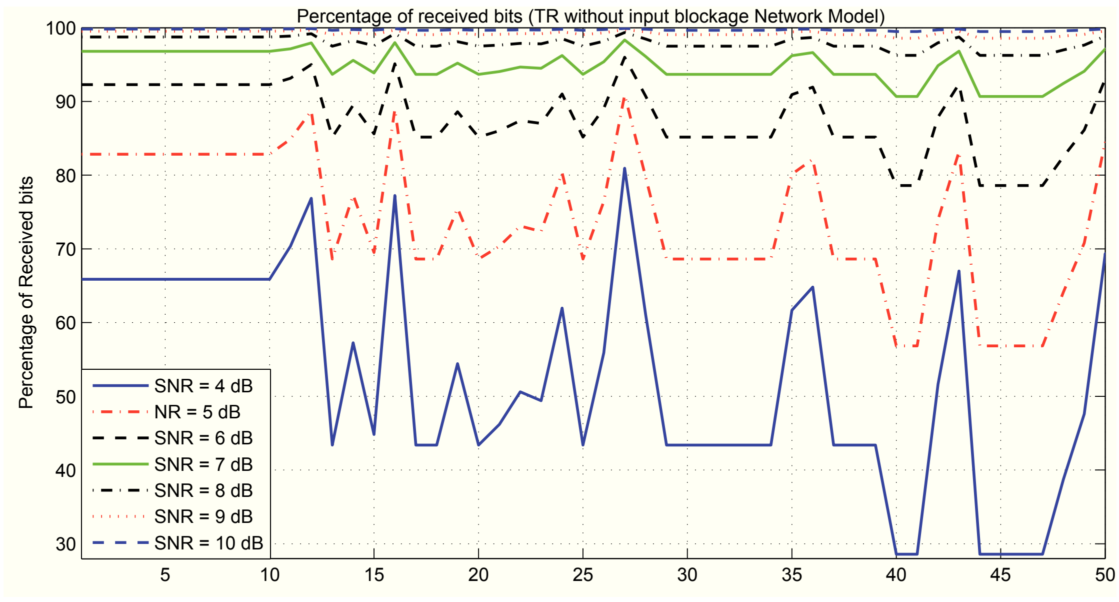

Figure 3.24: TRwIB algorithm results in a radio channel characterized by independent error probabilities over the received bits

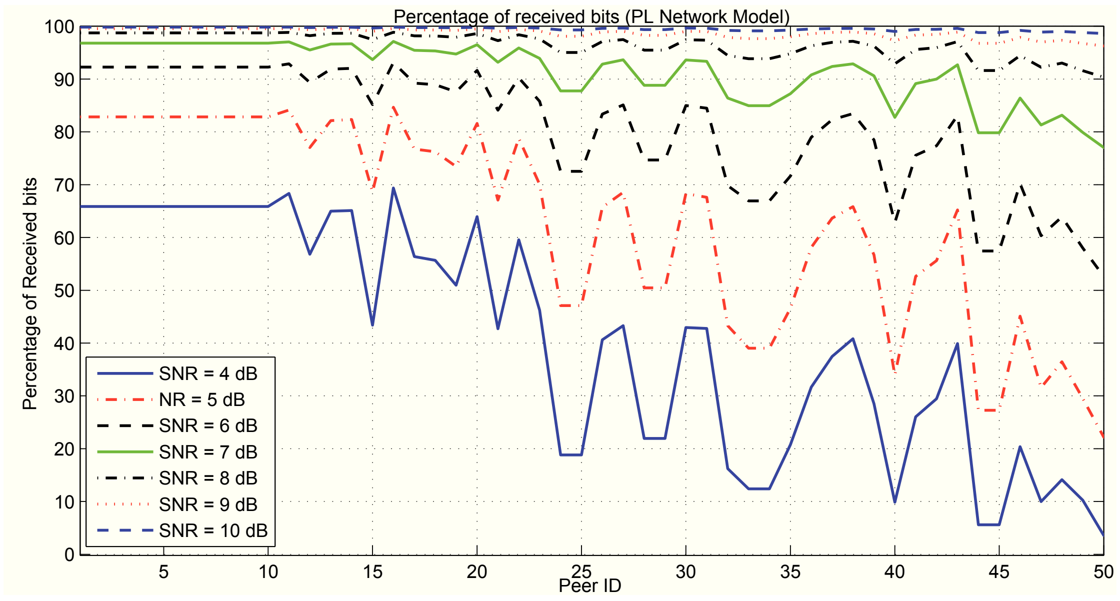

Figure 3.25: PL algorithm results in a radio channel characterized by independent error probabilities over the received bits

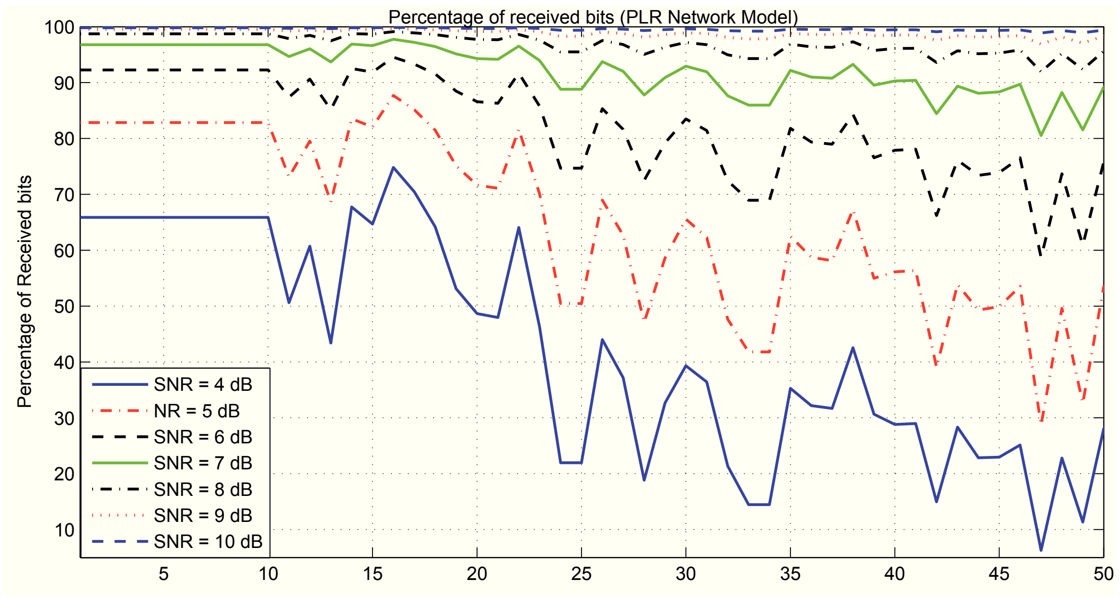

Figure 3.26: PLR algorithm results in a radio channel characterized by independent error probabilities over the received bits

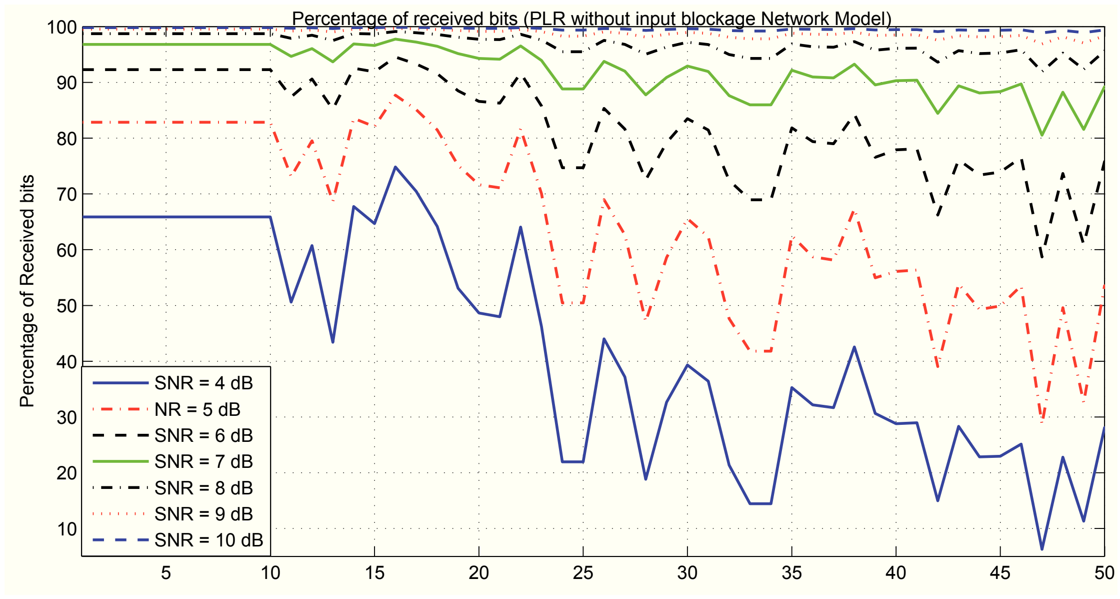

Figure 3.27: PLRwIB algorithm results in a radio channel characterized by dependent error probabilities over the received bits

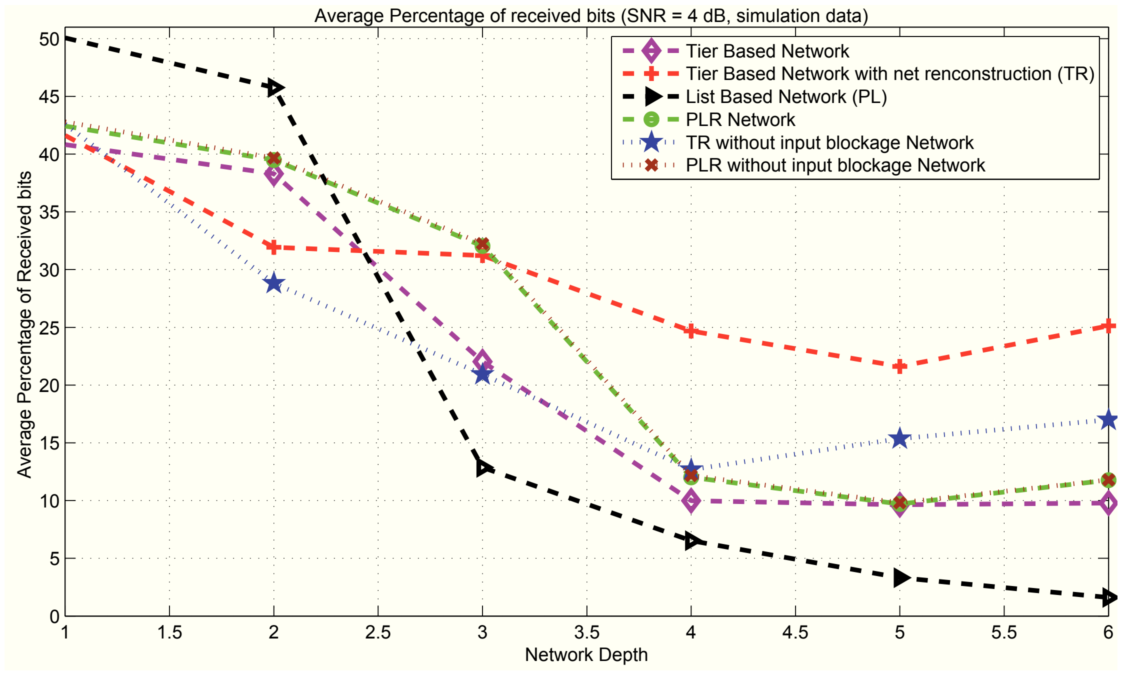

Figure 3.28: Comparison, SNR = 4 dB and radio channel characterized by dependent error probabilities over the received bits

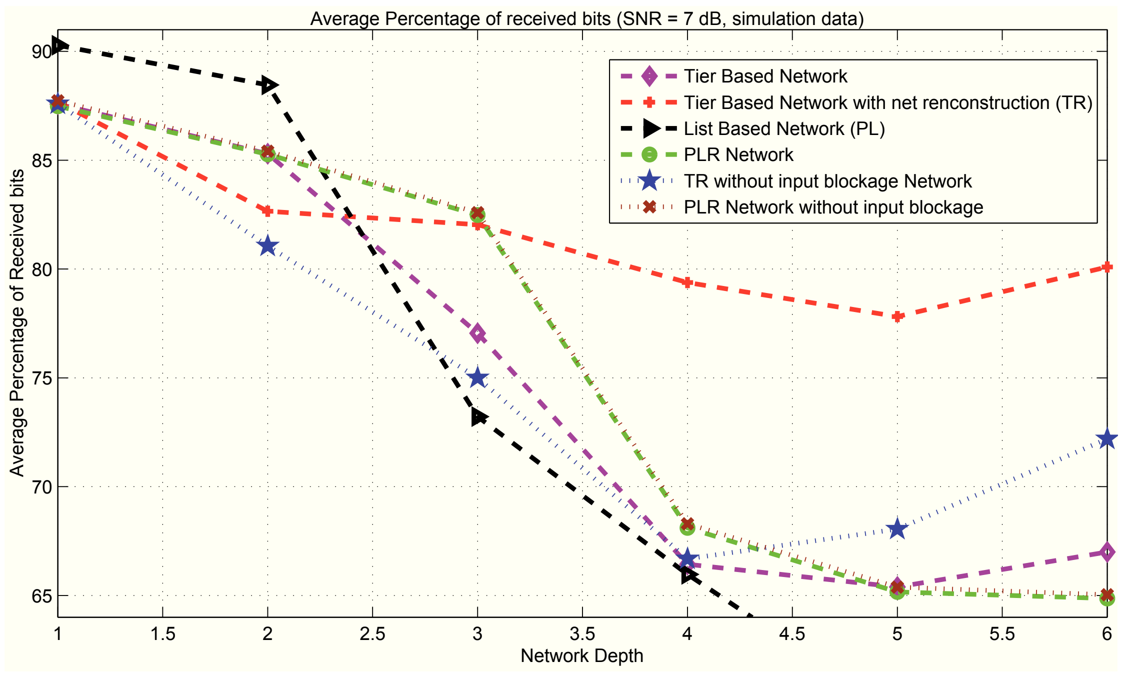

Figure 3.29: Comparison, SNR = 7 dB and and radio channel characterized by dependent error probabilities over the received bits

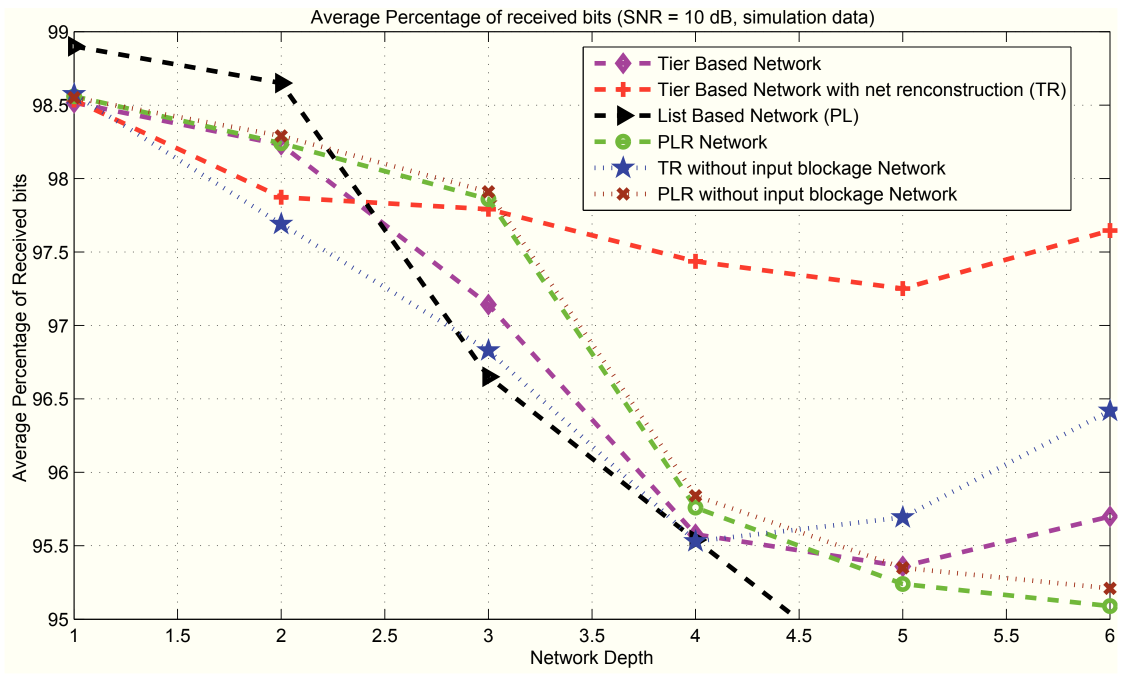

Figure 3.30: Comparison, SNR = 10 dB and and radio channel characterized by dependent error probabilities over the received bits

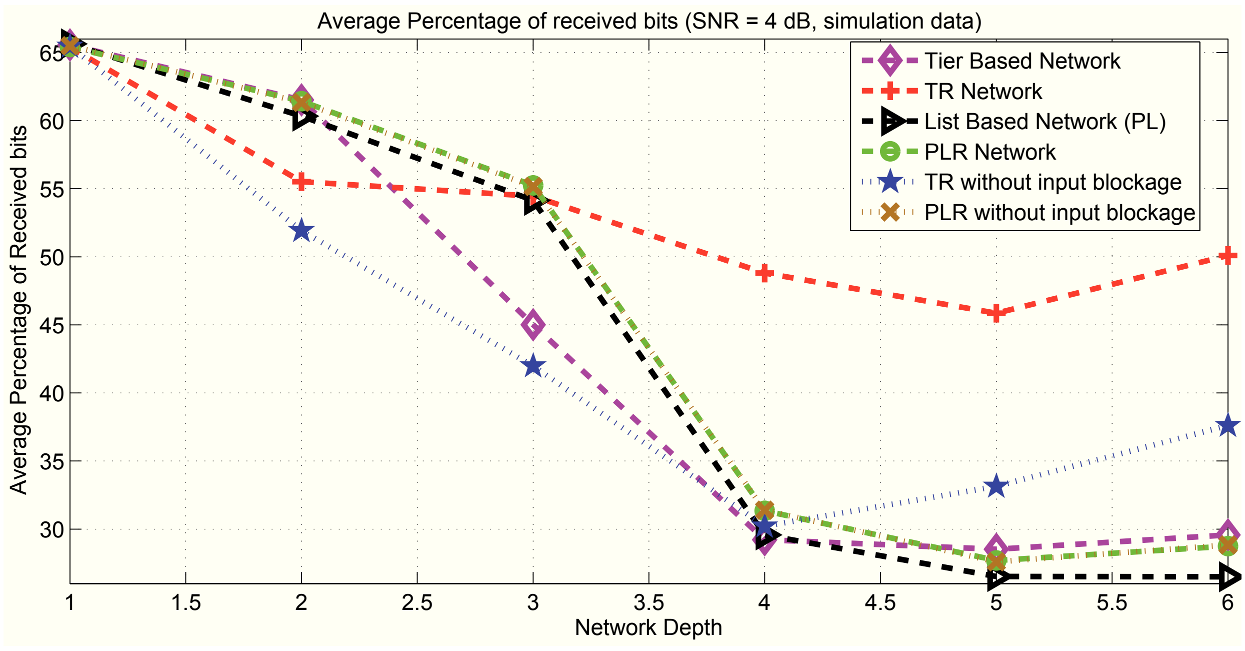

Figure 3.31: Comparison, SNR = 4 dB and radio channel characterized by independent error probabilities over the received bits

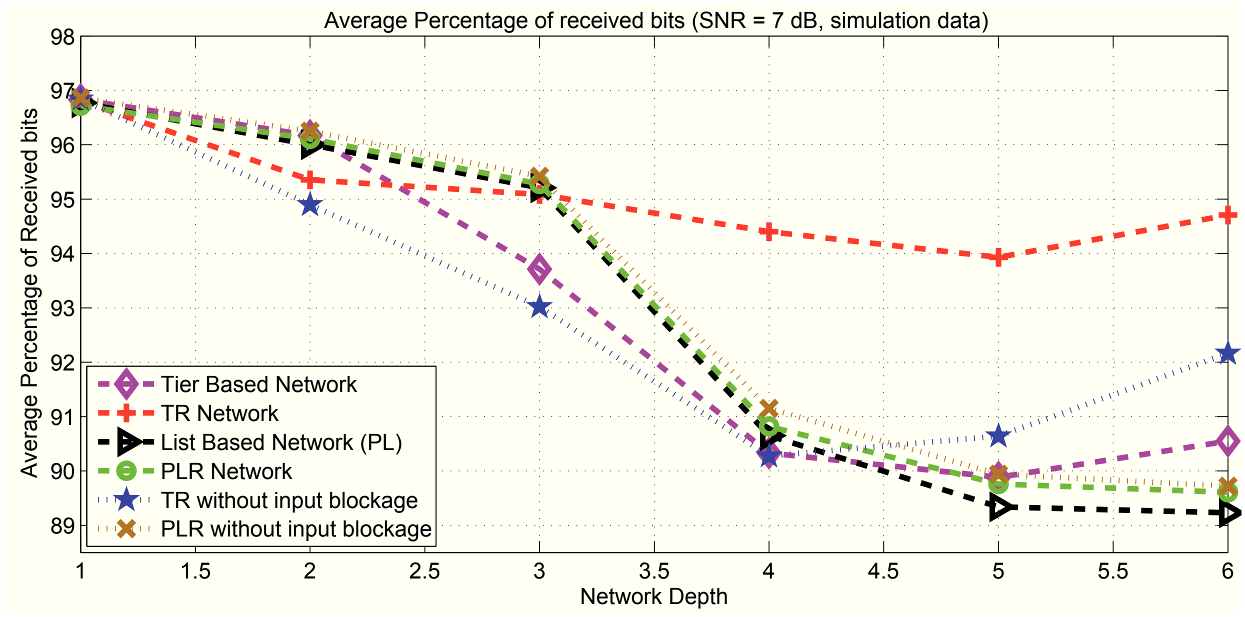

Figure 3.32: Comparison, SNR = 7 dB and and radio channel characterized by independent error probabilities over the received bits

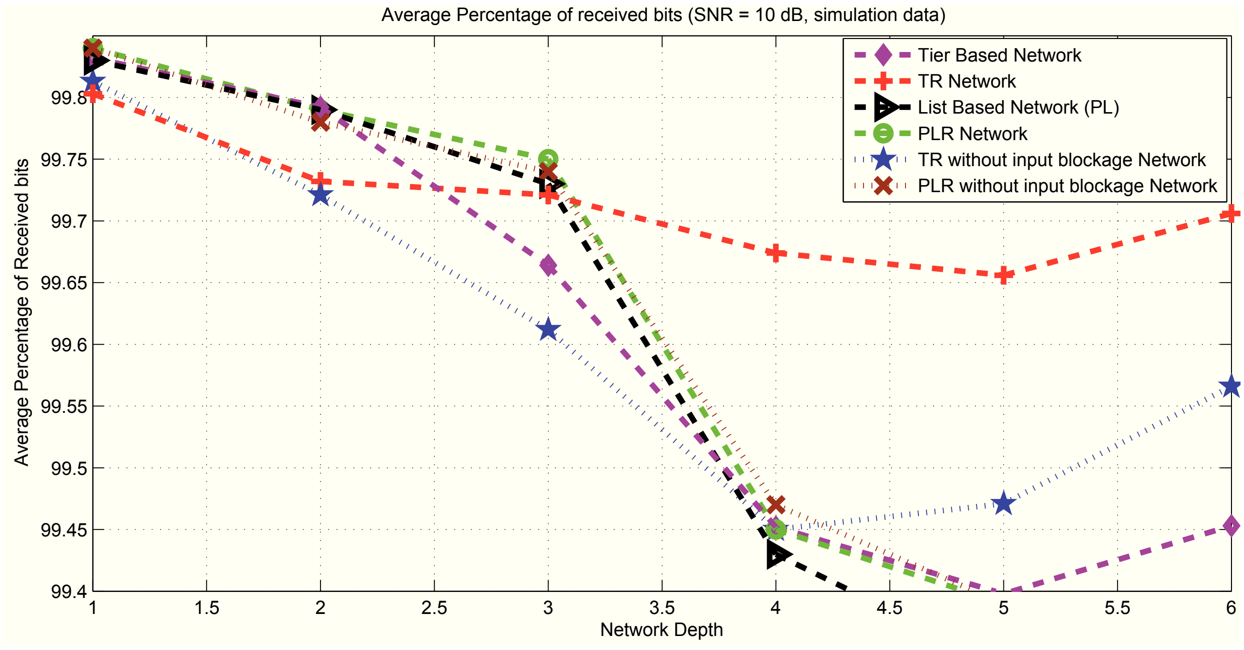

Figure 3.33: Comparison, SNR = 10 dB and and radio channel characterized by independent error probabilities over the received bits

Chapter 4

Peer-to-Peer for Unreliable Multicast

4.1 Introduction

In this chapter we analyse an application of the peer-to-peer networks used to make reliable an unreliable multicast network. The Reliable Multicast is a very investigated field. In fact there are different mechanism used to have a reliable multicast network such as: peer-to-peer (P2P) tree based reliable multicast protocol [17], peer-to-peer epidemic algorithms for reliable multicasting [18], reliable peer-to-peer end system multicasting through replication in which it is presented an effective dynamic passive replication scheme designed to provide reliable multicast service in peer-cast [8], and many others[19, 20]. In this chapter we propose a new simple mechanism used to make reliable an unreliable multicast network that exploits a peer-to-peer network in which the peers are the destination nodes of the multicast tree. This chapter is organized as follows, the scenario and the main hypotheses of the chapter are explained in section 4.2. Section 4.3 describes an ARQ multicast repairs mechanism over IP to make reliable the IP multicast network. Moreover in this section we describe an analytical model for this system. In section 4.4 we present another way to make the IP multicast reliable in a congested multicast unreliable network, with the support of a particular peer-to-peer network. Even in this case we present a simple analytical model for this system. In section 4.5 we present the Waxman [81, 82] model used to estimate the distance (in term of the number of the intermediate links) between each couple of peers in the peer-to-peer network. The results comparison between the ARQ multicast repairs mechanism and the multicast with peer-to-peer support are presented in the last section.

4.2 Scenario and Hypotheses

The aim of this chapter is to show the average access time (with respect to the destination node) in two different approach of reliable multicast. In the first approach we have a multicast over IP network in which each router integrates a simple ARQ system. Thus an infrastructure of routers with ARQ support used to create a reliable multicast network, it is necessary. This approach is based on the article [83]. The second approach is based on an unreliable multicast network over IP where all the destination nodes and the source of the multicast communication make a peer-to-peer network used by each destination node to request and get each lost packet. In this case this system does not need a particular infrastructure because it is a distributed system among the destination nodes.

We suppose that the only cause of packets loss, in the multicast network, is the event of congestion.

In this way we formulate an analytical model for the first and the second approach and we compare them with respect to the average access time.

We suppose that the multicast tree is a M-ary tree with height H and the interval of time used to send or receive a packet from one level of the multicast tree to the next level of the tree is constant and equal to t_f. Thus, for the symmetry of the multicast network, it is not necessary to evaluate the average access time of each destination node in this way we evaluate only the average access time of one destination node. We are not interested in how is incrementally built the multicast tree and the multicast tree does not have to change during the time.

The main hypotheses of the unreliable multicast with peer-to-peer network are the following:

- $C_{k,i}$ is the bandwidth of the virtual (or overlay) link between the k-th node and the i-th generic node.
- $C_{k,i} = C_{i,k}$.
- $Q_{k,i}$ is the dimension of the packet used to query the desired packet in the virtual link between the k-th node and the i-th node.
- $V_{k,i}$ is the dimension of the packet that contains the desired packet in the virtual link between the i-th node and the k-th node.
- $\tau_{k,i}$ is the propagation delay between the k-th node and the i-th generic node.
- $\tau_{k,i} = \tau_{i,k}$.
- P_{h_i} is the hit probability of the i-th node and this probability depend on the packet loss probability of the unreliable multicast network.
- The request of a packet from the k-th node to the i-th node is transmitted in an interval of time equal to $t_{req_{k,i}} = Q_{k,i}/C_{k,i}$.
- The desired packet is sent from the i-th node to the k-th node in an interval of time equal to $t_{send_{k,i}} = V_{k,i}/C_{k,i}$.
- $k = 0$ which indicates the peer who makes the packet request.
- $i = N + 1$ which indicates the source node of the unreliable multicast network.
- all the parameters are constant during the time.

In the multicast network all the links between routers are physical while all the links between peers, in the peer-to-peer network, are virtual (or overlay) links. Moreover we suppose that the interval of time needed to transmit a packet on each link between routers, in the multicast network, is t_f thus the interval of time used to send a request of a retransmission of the packet is $t_{req_{i,i-1}} = t_f$, $\forall i = 1, ..., H$. The average access time and the time intervals t_{send}, t_{req} are all normalized with respect to the parameter t_f.

4.3 ARQ Multicast Repairs Mechanism

We present the mechanism used by the reliable multicast protocol that we consider. In this section we formulate an approach used to make reliable the multicast over IP, based on the article [83].

Reliable multicasting is used to provide QoS in group communications for real time media application such as video conference, phone calls. To provide reliability, in a multicast network, we suppose to use the ARQ (Automatic Repeat Request) strategy, which combines error detection and retransmission of the data. Another error control strategy is the FEC (Forward Error Correction). In ARQ strategy, when an error is detected at the receiver or the receiver does not receive the packet before the time out, the receiver transmits a request to the sender to repeat the incorrect message and this continues until the message is received correctly. There are three types of ARQ strategies: Stop and Wait, Go-Back-N and Selective-Repeat. In this chapter we use the Selective-Repeat strategy. Using the reliable multicast with an ARQ strategy, there is a scalability problem because the increasing of the receiver (and the increasing of the router employed in the multicast tree) could easily result in a feedback implosion problem. This problem can be solved allowing the receivers to send the retransmission request (to the sender) only in the case of error of lost data. There are two method to send the requested packet from the sender to the receiver or a group of receiver: Multicast repairs, in this case the sender receives one or more retransmission request for the same packet and the sender multicasts again the same packet to all the receivers. Unicast Repair: in this case if the sender receives a retransmission request from one or more receivers, it resends the requested packet to only the receivers who did not receive the packet correctly using a unicast communication. The multicast repairs method is simpler than the unicast repairs method and uses less overhead. On the other hand the multicast repairs method consumes much more bandwidth with respect to the second method. In this chapter we use the unicast repairs method to guarantee high performance and less bandwidth consummation. In the following subsection we present the fair share policy presented in the paper [83] and the packet loss probability. We use the packet loss probability to formulate a model of the average time for the transmission of a packet from one node (or router) to another node (or router).

4.3.1 Packet Loss Probability: Analytical Fair Share Policy (FSP)

In FSP mechanism the packets are discarded in case of congestion differently at each queue according to the source priority and the maximum dimension of the buffer of each queue. In this way we consider that each router of the multicast tree contains three priority queue: the first one is the queue with higher priority, and this queue accepts packet with high priority such as multicast data packet or real-time packet. The second queue and the third queue are the medium priority queue and the lower priority queue respectively. The source with high priority will experience less packet discarding than the source (queue) with lower priorities. We are only interested on the congestion of the multicast network so we consider only the packet loss probability for the higher priority queue. Now we report the packet loss model presented in the article [83]. The analytical model underlying the Fair Share Policy (FSP) has the following assumption:

- we assume a Bernoulli Arrival for all sources; in order to have short and discrete interarrivals.

- FSP uses a non pre-emptive priority queuing.
- α_1, α_2, α_3 are the arrival probabilities for each source respectively. α represents the probability of receiving a packet while one packet is served on the channel.
- max_1, max_2, max_3 are the maximum buffer size, in term of maximum number of packets, for each source respectively.
- The total buffer size is: $B = max_1 + max_2 + max_3$, where max_p, $p = 1,2,3$ is calculated as: $max_p = \frac{Pr_p}{\sum_p Pr_p} \cdot B$, where Pr_p is the source p priority.
- All IP routers on the subject Internet are homogeneous in providing resources and traffic conditions.
- All the packet are of the same length.
- We suppose that the higher priority buffer (that manages the multicast packets) can hold only one packet. This condition can make possible the packet loss when the network is congested.

We define the parameters β_1, β_2 and β_3 that represent the service probability of each queue. $\beta_1 = 1$ for the first priority queue. $\beta_2 = P_0^1$ this means that the packet from the source 2 will be served only when the queue corresponding to source 1 (which has higher priority) is empty (with probability P_0^1). $\beta_3 = P_0^1 \cdot P_0^2$ is the service probability of the lowest priority queue and it means that the source 3 will be served only when the highest priority queue and the medium priority queue are empty (with probability P_0^1 and P_0^2 respectively). In our study we are only interested on the highest priority queue, because this queue manages the multicast packets. Thus we analyse the packet loss probability $P_{LOSS} = P_{max_1}$ for the highest priority queue. Moreover the parameter Pc_p represents the probability of successful delivery, of a packet, to the next router for priority traffic p (where $p = 1,2,3$) and $Pc = Pc_1$.

We define α_1^1 the intrinsic arrival probability for the data packets and α_1^2 the extra arrival probability due the IP-control overhead used to establish the IP multicast tree. Thus the source arrival probability for the first queue is:

$$\alpha_1 = \tau \cdot \alpha_1^1 + \alpha_1^2 \quad where \quad \tau = \frac{\Delta_1 + \Delta_2}{\Delta_1} \tag{4.1}$$

where Δ_1 is the processing time at lower layer (for example MAC layer), Δ_2 is the processing time at IP layer and τ is the IP processing time factor. For the highest priority queue, we indicate the interarrivals probability with $\alpha = \alpha_1$. From the article [83] we have the following parameters:

$$\begin{array}{cc} \lambda = (1-Pc\beta)\alpha & \mu = (1-\alpha)Pc\beta \\ \sigma = \alpha Pc\beta + (1-\alpha)(1-Pc\beta) & \\ m = \frac{1-\sigma}{\mu} & q = \frac{\lambda}{\mu} \\ a = -m/2 & b = q \end{array} \tag{4.2}$$

The probability to have n packets in the highest priority queue is:

$$P_n = Ar_1^n + Br_2^n \tag{4.3}$$

where r_1 and r_2 are the distinct roots of the equation (1.4) and A, B are constants:

$$r^2 - mr + q = 0 \tag{4.4}$$

which has the solution:

$$r_1 = \frac{m + \sqrt{m^2 - 4q}}{2}, \quad r_2 = \frac{m - \sqrt{m^2 - 4q}}{2} \tag{4.5}$$

The parameters A and B are the following:

$$B = \left(\frac{\omega - kr_1}{r_2^2 - r_1 r_2}\right) P_0, \quad A = \left(\frac{\omega P_0 - Br_2^2}{r_1^2}\right) \tag{4.6}$$

where:

$$\omega = \left(\frac{(1-\sigma)\alpha}{\mu^2} - \frac{\alpha}{\mu} \right) \textit{ and } k = \frac{\alpha}{\mu} \tag{4.7}$$

The probability to have n packets in the highest priority queue is:

$$\begin{aligned} P_n = \left(\tfrac{\omega P_0 - B r_2^2}{r_1^2}\right) \cdot \left(\tfrac{m+\sqrt{m^2-4q}}{2}\right)^n + \\ + \left(\tfrac{\omega - k r_1}{r_2^2 - r_1 r_2}\right) P_0 \cdot \left(\tfrac{m-\sqrt{m^2-4q}}{2}\right)^n \end{aligned} \tag{4.8}$$

Now we have to achieve the probability P_0. We define:

$$X = \left(\frac{\omega - k r_1}{r_2^2 - r_1 r_2} \right), \quad Y = \left(\frac{\omega - X r_2^2}{r_1^2} \right) \tag{4.9}$$

where $A = X \cdot P_0$ and $B = Y \cdot P_0$. We apply the formulation (4.9) to the formula (4.8) and we achieve P_0 as follows:

$$P_0 = \frac{1}{1 + k + \omega + Y \cdot \sum_{n=3}^{max_1} r_1^n + X \cdot \sum_{n=3}^{max_1} r_2^n} \tag{4.10}$$

In conclusion the packet loss probability is:

$$\begin{aligned} P_{Loss} = P_{n=max_1} = \left(\tfrac{\omega P_0 - B r_2^2}{r_1^2}\right) \cdot \left(\tfrac{m+\sqrt{m^2-4q}}{2}\right)^{max_1} + \\ + \left(\tfrac{\omega - k r_1}{r_2^2 - r_1 r_2}\right) P_0 \cdot \left(\tfrac{m-\sqrt{m^2-4q}}{2}\right)^{max_1} \end{aligned} \tag{4.11}$$

If the highest priority queue can hold only one packet then the parameter that generates congestion in the multicast network is the interarrivals probability α. Thus the multicast network is fully congested if $\alpha = 1$. While the multicast network is not congested when $\alpha = 0$. In this way the increase of the value of α generates an increase of congestion in the multicast network (see figure 1 and 2).

4.3.2 Average access time for the transmission and the retransmission of a packet

In this subsection present the analytical formulation for the average access time in the reliable multicast system based on the ARQ Mechanism presented above.

We suppose to have a M-ary multicast tree. In this case we can formulate a simple model for the average time necessary to send correctly a packet from one node (of the (i-1)-th level of the multicast tree) to another node (of the i-th level of the multicast three). This average time has the following formulation:

$$\begin{gathered}\overline{t(i)} = t_{send_{i-1,i}} + \tau_{i-1,i} + P_{Loss_i} \cdot (t_{req_{i,i-1}} + t_{send_{i-1,i}} + 2\tau_{i-1,i} + \\ + P_{Loss_i} \cdot (t_{req_{i,i-1}} + t_{send_{i-1,i}} + 2\tau_{i-1,i} + \ldots))\end{gathered} \tag{4.12}$$

Thus the average access time at i-th level of the M-ary multicast tree is:

$$\overline{t(i)} = t_{send_{i-1,i}} + \tau_{i-1,i} + (t_{req_{i,i-1}} + t_{send_{i-1,i}} + 2\tau_{i-1,i}) \cdot \left(\frac{P_{Loss_i}}{1 - P_{Loss_i}} \right) \tag{4.13}$$

where $P_{Loss_i}/(1 - P_{Loss_i}) = \eta$ is the number of retransmission. When $P_{Loss_i} \to 1$ then the number of retransmission $\eta \to +\infty$ and $\overline{t(i)} \to +\infty$.

In this way the average access time in a destination node (of the multicast tree) is:

$$\overline{t(H)} = \sum_{i=1}^{H} \overline{t(i)} \tag{4.14}$$

where H is the depth (or maximum level) of the multicast tree and $P_{Loss_H} = 0$ because we suppose that the destination node has no congestion problems so the packet loss probability is equal to 0. Moreover if $\exists i \in \{1,2,...,H\} : P_{Loss_i} \to 1$ then $\overline{t(H)} \to +\infty$.

4.4 Unreliable Multicast with Peer-to-Peer Network Support

In this section we present a new mechanism used to make reliable an unreliable multicast network. In this case the routers of the multicast network don't include special feature such as the ARQ mechanism used for the reliability of the multicast network. In this way we suppose to have a reliable peer-to-peer network (random generated) where the peers are the destination nodes of the unreliable multicast network. In this peer-to-peer network a limited number of peers are connected to the source node of the multicast network. Thus when a destination node does not receive a packet from the multicast network this node contacts the other destination nodes, through the peer-to-peer network, to get the lost packet. If the nodes of the peer-to-peer network don't have the requested packet then:

- one peer of this network contacts the source node of the multicast, and gets it from the source node the requested packet.
- This peer sends to the other peers the requested packet through the peer-to-peer network.

4.4.1 Average Access Time for a Peer-to-Peer Network

In this subsection we present a simple model of the average access time of the peer-to-peer network used to make reliable an unreliable multicast network. This model is based on the formulation presented in the second chapter section 2.4.1 formulas (2.6) and (2.7). The same formulation is presented in our article [84]. In this formulation is explained the average access time of a proxies network. In case of a proxies network each link is a physical link between routers, between a client and a router or between a router and a server. In case of a peer-to-peer network we have an overlay network thus the link between each couple of peers is a virtual link (not physical link) that is made-up of one or more physical links. In this situation we estimate the number of these links through the Waxman model [81, 82], presented in the next subsection. From the formulation (2.6), the average access time of the peer-to-peer network, used to make reliable an unreliable multicast network, has the following recursive formulation:

$$\overline{t(i)} = Min_k \left\{ \overline{t(k)} + \left[\frac{Q_{k,i}}{C_{k,i}} + \frac{V_{k,i}}{C_{k,i}} + 2\tau_{k,i} \right] \cdot P_{not\,hit_k} \cdot (1 - P_{h_i}) \right\} \tag{4.15}$$

where: $\overline{t(k)}$ is the average access time generated by the k-th peer that is in the input of the generic i-th peer. P_{h_i} is the hit probability of the i-th peer and $P_{h_i} = \prod_{i=1}^{H} 1 - P_{Loss_i}$, where H is height (or maximum depth) of the multicast tree and $P_{Loss_H} = 0$ for hypothesis. $P_{not\,hit_k} = \prod_{j \in Path\,from C\,to\,k} \left(1 - P_{h_j}\right)$ where C is the source peer of the request. $P_{h_C} = 1 - P_{not\,hit\,C}$ and $P_{not\,hit_i} = Min_k \left\{P_{not\,hit_k}\right\} \cdot (1 - P_{h_i})$. For each i-th peer directly connected to the peer that requests the packet we have $\overline{t(k)} = 0$.

For the last peer S of the peer-to-peer network (in our case the last peer is the source node of the multicast network) the formulation becomes the following:

$$\overline{t(S)} = Min_k \left\{ \overline{t(k)} + \left[\frac{Q_{k,S}}{C_{k,S}} + \frac{V_{k,S}}{C_{k,S}} + 2\tau_{k,S} \right] \cdot P_{not\,hit_k} \right\} \tag{4.16}$$

where each k-th peer is the closest to the source node. In this way the formula (4.16) give us the final average access time for the peer-to-peer network.

4.4.2 Nodes distance Model

In this subsection we present briefly the model of the distance (in term of the number of the intermediate links) between nodes in a peer-to-peer network. This model is the Waxman model [81, 82] and the probability that two nodes (or peers) are connected with a number d of intermediate links is:

$$f(d) = \delta \cdot e^{-\frac{d}{\gamma \cdot L}} \tag{4.17}$$

where L is the maximum distance between the two nodes, γ and δ are parameters in the range . Larger values of δ result in a peer-to-peer network with higher edge (link) densities, while small values of γ increases the density of short edges (links) relative to longer ones.

This model allow us to estimate the average distance between each couple of peers as follows:

$$E[d] = \overline{d} = \int_0^L d \cdot f(d) dd = (\gamma L)^2 \delta \cdot \left(1 - e^{-\frac{1}{\gamma}}\right) - \gamma L \delta \cdot e^{-\frac{1}{\gamma}} \tag{4.18}$$

If we suppose to know L and we can achieve whereas:

$$\int_0^L f(t) dt = 1 \tag{4.19}$$

Solving this integral we achieve the value of δ:

$$\delta = \frac{1}{\gamma L \cdot \left(1 - e^{-\frac{1}{\gamma}}\right)} \tag{4.20}$$

In this way when we fix the parameter γ, we achieve δ.

In our formulation the maximum distance L is the maximum number of intermediate links (that there are in the multicast network) between the peers considered. We suppose that the interval of time needed to transmit a packet on a physical link is constant. Hence the interval of time needed to transmit a packet on the virtual link between the k-th peer and the i-th peer is:

$$t_{req_{k,i}} = \frac{Q_{k,i}}{C_{k,i}} = t_f \cdot d_{k,i} \tag{4.21}$$

where t_f is the interval of time needed to transmit a packet on a physical link and $d_{k,i}$ is the average distance between the k-th and the i-th node. In the same way the interval of time needed to transmit a packet on the virtual link between the i-th peer and the k-th peer is:

$$t_{send_{k,i}} = \frac{V_{k,i}}{C_{k,i}} = t_f \cdot d_{i,k} \tag{4.22}$$

where t_f is interval of time needed to transmit a packet on a physical link and $d_{i,k}$ is the average distance between the i-th and the k-th node. Moreover we suppose that the interval of time needed to transmit a packet on each link between routers, in the multicast network, is t_f.

4.5 Simulation Results

This section shows the simulation results in the case of reliable multicast with the ARQ and the unreliable multicast with peer-to-peer support.

The experimental framework for both systems (ARQ and peer-to-peer) has the following parameters: t_f, M, H, α, γ, $\tau_{i,j}$ and the highest priority queue size. The setting of the parameters are the following: $t_f = 4ms$, $M = 2$, $H = 8$ (thus the number of nodes of the multicast tree is $N = 2^{H+1} - 1 = 511$ and the number of destination nodes is $N_D = 2^k = 256$), $\alpha \in]0,1[$ with step of 0.001, $\gamma \in]0,1[$ with step of 0.1, $\Delta_2 = 0$, $\tau_{i,j} = Constant = 1ms$, $\forall(i,j)$ and highest priority queue size equal to 1 packet size (to allow a fast congestion of the network).

In figure 4.1 and 4.2 the average access time of the ARQ reliable multicast and the unreliable multicast with peer-to-peer network used to avoid the packet loss are reported. The results address that: for low level of congestion (low values of the interarrivals probability α) the best mechanism is the ARQ reliable multicast with respect to the average access time. For high level of congestion (high values of the interarrivals probability α) the best mechanism is the unreliable multicast with peer-to-peer network used to avoid the packet loss because each peer (of the peer-to-peer network) can get the lost packet from the source node through the peer-to-peer network itself. Moreover the behaviour of the peer-to-peer based mechanism depends on the density of short links through the parameter γ because

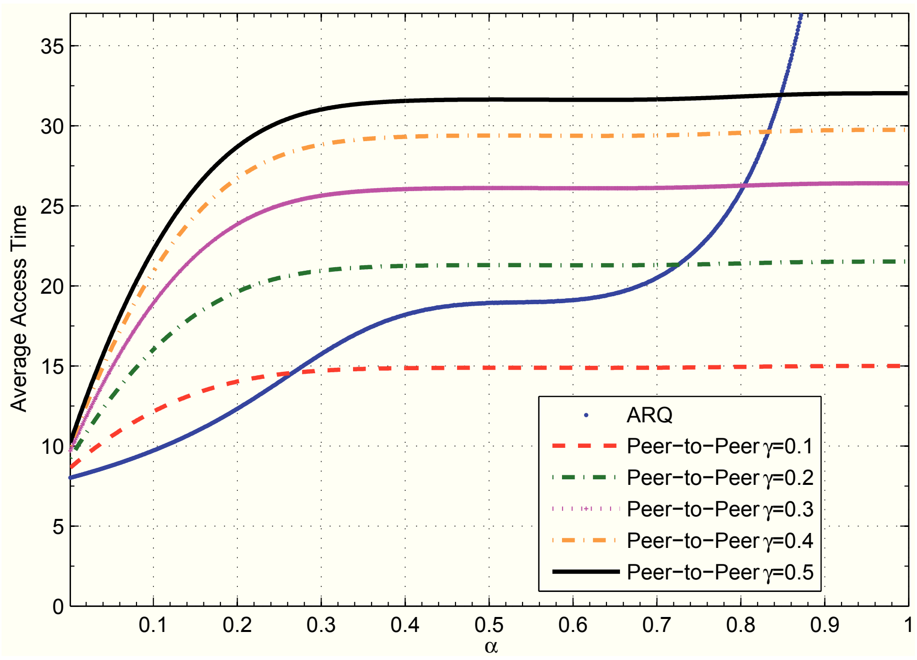

Figure 4.1: Average access time in an ARQ multicast repairs mechanism over IP and in an unreliable multicast with peer-to-peer network used to avoid the packet loss, with respect to α and γ between 0.1 and 0.5.

the increase of γ produces an increase of the average access time. In this way, with low values of γ, the peer-to-peer based mechanism has a better behaviour with respect to the ARQ mechanism even with low level of congestion ($\alpha > 0.28$). With high values of γ, the peer-to-peer based mechanism has a better behaviour with respect to the ARQ mechanism only with high level of congestion ($\alpha > 0.88$).

In this way we can minimize the average access time using an hybrid system in which: when the multicast network has a low level of congestion with respect to the parameters α and γ, the system uses the ARQ reliable multicast. When the multicast network has an high level of congestion the system uses an unreliable multicast network with a peer-to-peer network used to avoid the packet loss.

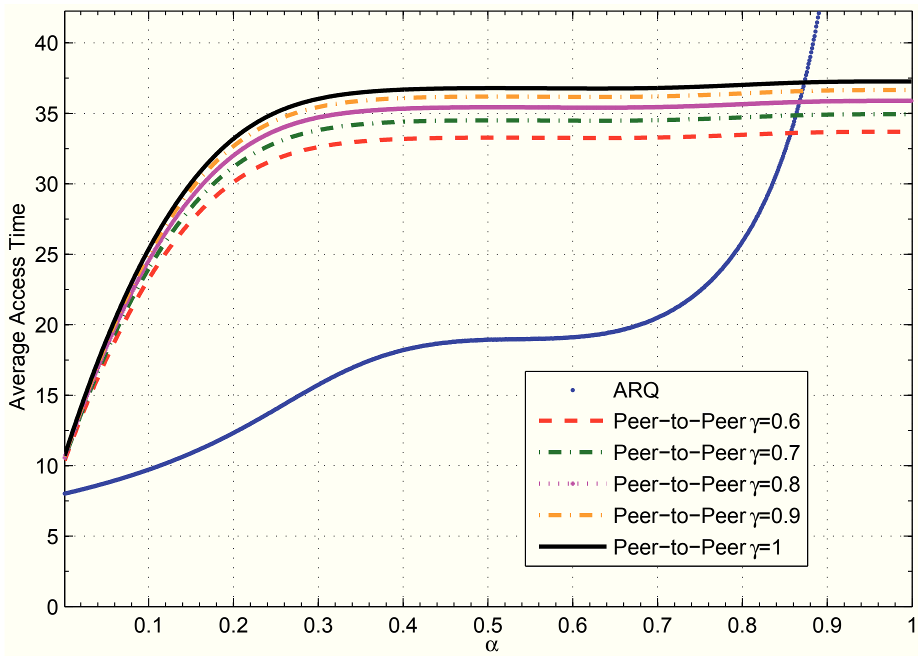

Figure 4.2: Average access time in an ARQ multicast repairs mechanism over IP and in an unreliable multicast with peer-to-peer network used to avoid the packet loss, with respect to α and γ between 0.6 and 1.

Chapter 5

Massey Omura Multiple Users Key Distribution

In this chapter we present a generalization of the Massey Omura protocol to allow secure communication among a group of users or peers. Furthermore, this generalization of the Massey Omura Protocol can be used for the keys distribution in a group of users or peers. Moreover we present the security problems of this Massey Omura Multiple Users Protocol and how it is possible to avoid these issues through a specific encryption function and a specific decryption function by changing the encryption and decryption keys of each peer when the source peer changes. Finally we presented a new cryptography protocol (based on Elgamal protocol) which we used to share the decryption shared key which is used in the Massey Omura Multiple Users Protocol.

5.1 Introduction

Internet was born as a computer network in which all the communications and protocols transmit plain text over a dedicated insecure physical channel, therefore it is difficult to keep privacy and confidentiality of the information transported, due to its public nature. But with globalization, the increase of the number of private users, the creation of particular services like home banking systems, on-line shopping (SSL/TSL [85]), privacy of private information shared among two or more users, it is necessary to use a secure channel to allow these features. Therefore in this scenario the cryptographic techniques and protocols assume an important role to protect private data transmitted over a public insecure network.

The cryptosystems are classified in two kinds: the symmetric cryptosystem and the asymmetric cryptosystem. The symmetric cryptosystems use a single secret key shared among a group of users that want to communicate. On the other hand in the asymmetric cryptosystems there is a public key and a private key for each user. The public key is used to cipher the message and the private key is used to decrypt the message. In this case the users don't have to share a private secret key but the computational complexity of the asymmetric cryptosystems is much more higher than the symmetric cryptosystem as shown in the article "Symmetric and Asymmetric Encryption" [21].

There is another group of cryptography protocols in which the users don't have to share a secret key. The number of studies about these protocols are very less and they only allow the communication between users for each execution of the protocol. Therefore these protocols don't allow the communications with one user who sends information to three or more users in a secure way. Examples of these protocols are Shamir's 3-pass protocol [22][23], Khayat's protocol[24], Massey Omura[25] and other 3-pass protocols like [26]. The cryptosystem presented in this article is a generalization of the Massey Omura protocol that allow the secure communication among a group of users.

There is some architecture of secure multicast that needs an infrastructure, for example the paper "Secure Multicast Using Proxy Encryption"[86] which describes a system that needs an encryption system integrated into each proxy of the network (proxies with encryption support in which it is necessary). Our System doesn't need an infrastructure because the encryption and the decryption are handled by each user of the group. Moreover the encryption and decryption are done at application level (layer 7 stack OSI).

Another important aspect of cryptography is the key distribution systems because the cryptographic keys are the basis of secure communication channels. The key distribution systems that are used to share or distribute keys in the symmetric cryptography[22]. In this scenario there are two possible key distribution systems. The first one is a centralized system (KDC, “Key Distribution Center”, such as Kerberos [22, 23]) in which one entity (such as a server) generates and distributes the shared secret key to the other entities (users) through a secure channel. The second one is the distributed key distribution system in which all the participating entities (users) contribute information which is used to derive the shared secret key.

The main problem of KDC (such as Kerberos) [22, 23] is the single point of failure. In our proposed protocol the problem of the single point of failure is not present because the system is distributed among the users of the group.

The key distribution systems are used in different scenarios for example: Security and key distribution on Distributed Services described in the books [87, 88]. Distributed services running on Unix servers such as Kerberos, OpenLDAP, OpenAFS, Certificate Authority in which the key distribution is a relevant topic, presented in the book “Distributed Services with OpenAFS: for Enterprise and Education” [89]. Sensor and Ad Hoc network key distribution explained in the book [90]. Thus the key distribution systems are very important in public insecure network to keep privacy and confidentiality of the information transported.

In this chapter we address a simple protocol that can be used to distribute, a shared key or an information, to three or more users over an insecure channel. This protocol is a generalization of the Massey Omura protocol in which the secure communication is not between users but the secure communications are from one user to a group of users. In this case the users of the group use reliable multicast communications to reduce the bandwidth occupation of the network and increase the communications performance [91] with respect to the situation in which we use the Massey Omura protocol for each couple of users over an unicast TCP. The performance results of the reliable multicast presented in the article [91] shows that the reliable multicast using IRMA is as fast as the unicast TCP connection from the sender to the slowest receiver in the multicast group at any time. Thus our Massey Omura generalization provides better performance with respect to the classic Massey Omura protocol in which the secure communication is only established for each couple of users.

We use the Elgamal protocol to set up the system. Moreover, we analyse some problems about the decryption shared key of the Massey Omura generalized protocol and the man-in-the-middle.

This chapter is organized as follows: the second section describes the original Massey Omura Protocol. In the third Section the proposed protocol, with a generic (not specified) encryption and decryption functions, is presented. The fourth section presents the problem of the man-in-the-middle and the decryption, encryption functions that solve this problem. In the fifth section we analyse a security problem of our protocol with a solution. In the sixth section we present a protocol used for the generation and the distribution of the shared key. The summary of the chapter is presented in the last section.

5.2 Massey Omura Protocol

In this section we briefly describe the Massey Omura Protocol [25]. The Massey Omura protocol is a three-pass protocol that allow two users to exchange confidential information over a non-secure channel without sharing any secret data. We suppose to have two users, Alice and Bob. If Alice wants to send a message M to Bob by using the Massey Omura protocol, she encrypts the message with her secret encryption key (e_A) and sends the encrypted message to Bob. Bob encrypts with his encryption key (e_B) the message received from Alice and he sends the new message back to Alice; she decrypts Bob's response and gets the message encrypted by Bob with her description key (d_A) and sends the result back to Bob. Bob decrypts the message received from Alice with his decryption key (d_B) and in this way he finally gets the original message M. Alice doesn't have to know and doesn't need to know, the Bob's keys to communicate with him and vice-versa. Instead in the proposed protocol (section 3) all the encryption keys (of all the users involved in the multicast communication) are private and there is a shared decryption key which is shared among all the users involved in the communication.

The level of security of Massey Omura protocol depends on the solution of the discrete logarithm problem because if a third user Carl wants to achieve the plain text message M from the encrypted message C, without knowing the encryption and decryption keys of Alice and Bob, Carl has to solve the discrete logarithm problem. There is no known sub-exponential type algorithm to solve the discrete logarithm problem (as shown in the articles [92, 93, 94]) for the Massey Omura protocol. This guarantees a high level of security in the Massey Omura protocol against the attacks of a third user.

The Massey Omura protocol is susceptible to the man-in-the-middle attack [95] because there isn't any kind of mechanism used to authenticate the users (Alice and Bob) each other. In this way a third user can impersonate one of the two communicating users (Alice or Bob) sending to the other user (Alice or Bob) fake messages. Therefore, in the proposed protocol, a particular encryption function and a particular decryption function are used to solve the problem of the man-in-the-middle.

5.3 The Proposed Protocol

In this section we first present the hypothesis of the proposed protocol and then the description of the protocol step by step.

5.3.1 Hypothesis

The hypothesis of the generalized Massey Omura protocol are:

- There are N+1 users in the network and they want to exchange messages with a particular encryption system.

- All the keys have to be private.
- Scenario: we suppose that one user sends a message to all the other users of the group with the proposed protocol.
- The user 0 wants to send the plain text M_0 to the other users (where the subscript 0 indicates the user 0).
- Each i-th user (with $i = 0...N$) has:
  - a private encryption key $e_{0,i}$ used to cipher the plain text M_0: $C_{0,i} = E(M_0, e_{0,i})$;
  - a private decryption key $d_{0,i}$ used to decipher the message ciphered with the encryption key $e_{0,i}$: $D(C_{0,i}, d_{0,i}) = D(E(M_0, e_{0,i}), d_{0,i}) = M_0$;
  - a shared decryption key $d_{s,0}$ (all the users of the group know this key); it is used to decipher the encrypted message $C = E(M_0, e_{0,1} \cdot e_{0,2} \cdot ... \cdot e_{0,N})$. In this way the plain text is achieved with the following operation: $M_0 = D(C, d_{s,0})$;
  - property of the encryption function: $E(M_0, e_{0,1}) \cdot E(M_0, e_{0,2}) \cdot ... \cdot E(M_0, e_{0,N}) = E(M_0, e_{0,1} \cdot e_{0,2} \cdot ... \cdot e_{0,N})$.
- There is an Encryption Function E that produces an encrypted message $C_{0,i}$ from the plain text M_0 and the encryption key $e_{0,i}$ of the i-th user: $C_{0,i} = E(M_0, e_{0,i})$.
- There is a Decryption Function D used to remove the encryption in the message $C_{0,i}$: $D(C_{0,i}, d_{0,i}) = D(E(M_0, e_{0,i}), d_{0,i}) = M_0$.

5.3.2 Protocol

In this section we present the proposed protocol. We describe the protocol step by step as follows:

- The user 0 sends to the other users (in multicast) the message $C_{0,0}$ ciphered with $e_{0,0}$ (encryption key of the user 0 with the source node the node 0):

$$C_{0,0} = E(M_0, e_{0,0}) \tag{5.1}$$

- each i-th user (with $i = 1, ..., N$), that receives this encrypted message, ciphers the message with the encryption key $e_{0,i}$ and the user sends back this encrypted message to user 0:

$$C_{0,i} = E(C_{0,0}, e_{0,i}) = E(E(M_0, e_{0,0}), e_{0,i}); \tag{5.2}$$

- when the user 0 receives all the encrypted messages (from the other users) this user removes the encryption key $e_{0,0}$ in each received message:

$$D(C_{0,i}, d_{0,0}) = D(E(E(M_0, e_{0,0}), e_{0,i}), d_{0,0}) = E(M_0, e_{0,i}) \tag{5.3}$$

  for $i = 1, ..., N$. In this way the user 0 has the following cipher texts: $E(M_0, e_{0,1})$, $E(M_0, e_{0,2})$, ..., $E(M_0, e_{0,N})$.

- the user 0 produces a new message (as follows) and sends it in multicast to all the other users:

$$E(M_0, e_{0,1}) \cdot E(M_0, e_{0,2}) \cdot ... \cdot E(M_0, e_{0,N}) = E(M, e_{0,1} \cdot e_{0,2} \cdot ... \cdot e_{0,N}) = C \tag{5.4}$$

- each i-th user (with $i = 1, ..., N$) decrypts the message received from the user 0 using the shared decryption key $d_{s,0}$ and this user achieves the plain text M_0 as follows:

$$D(C, d_{s,0}) = D(E(M, e_{0,1} \cdot e_{0,2} \cdot ... \cdot e_{0,N}), d_{s,0}) = M_0 \tag{5.5}$$

where $d_{s,0} = d_{0,1} \cdot d_{0,2} \cdot \ldots \cdot d_{0,N}$ is the multiplicative inverse of the encryption key $e = e_{0,1} \cdot e_{0,2} \cdot \ldots \cdot e_{0,N}$.

5.4 Man-in-the-middle, Encryption and Decryption Functions

The problem of the man-in-the-middle is common in Massey Omura protocol. It is solved in the article [95]. The encryption and decryption functions, presented in this section, follow the hypotheses of section 3 subsection A. We address the parameters and the use of the encryption-decryption functions in a communication between users. This situation can be simply extended in a communication one to many. Let $G_0 = \langle P \rangle$ (used for the hash function), $G_1 = \langle Q \rangle$ and $G_T = \langle g \rangle$ (with G_0 and G_1 not necessarily distinct) be groups of order q, for some large prime number q, and $f : G_0 \times G_1 \to G_T$ a bilinear pairing [95].

User 1 wants to send a message M to user 2 over a non-secure channel. The key pair of the user 1 is: $e_1 \in \mathbb{Z}_n^*$, $V_1 = e_1 \cdot Q$, where e_1 is the private key and V_1 is the public key used only to verify the paternity of the message.

In the same way the key pair of the user 2 is: $e_2 \in \mathbb{Z}_n^*$, $V_2 = e_2 \cdot Q$, where e_2 is the private key and V_2 is the public key used only to verify the paternity of the message.

Assume that both e_1 and e_2 are generated randomly only once. Thus $V_1$and V_2 are generated only once because we suppose to use Elgamal Cryptosystem [96] to exchange these public keys among all the users. It is important that the public keys don't change oftentimes to reduce the usage of Elgamal protocol. The steps of the modified Massey Omura protocol are:

- Step 1: user 1 computes $C_1 = e_1 M$. User 1 computes $C_1^\delta = e_1 h(C)$, where $h : G_0 \to G_0$ is a public hash function that is known by user 1 and user 2. Then, user 1 sends C_1

and C_1^δ to user 2.

- Step 2: user 2 receives C_1 and C_1^δ. We suppose to use pairing key so user 2 checks if $f(C_1^\delta, Q) = f(h(C_1), V_1)$ (a). If the relation (a) is not satisfied the user 2 interrupts the protocol because someone tried to impersonate the user 1. Otherwise user 2 computes $C_2 = e_2 C_1$ and sends this result back to user 1.
- Step 3: user 1 receives C_2. User 1 check if $f(C_2, Q) = f(C_1, V_2)$ (b). If the relation (b) is not satisfied the user 1 interrupts the protocol because someone tried to impersonate the user 2. Otherwise user 1 computes: $C_3 = e_1^{-1} C_2 = e_1^{-1} e_2 e_1 M = e_1^{-1} e_1 e_2 M = e_2 M$ and sends it back to user 2.
- Final Step: user 2 receives C_3. User 2 checks if $f(C_1, Q) = f(M, V_1)$ (c). If the relation (c) is not satisfied the user 2 interrupts the protocol, refusing the message, because someone tried to impersonate the user 1. Otherwise user 2 computes $e_2^{-1} C_3 = e_2^{-1} e_2 M = M$. In this way the user 2 obtains, from user 1, the message in a secure way.

The relations (a), (b) and (c) that the user 1 and 2 check arise from the bilinear pairing properties (article [95], [97]):

- Relation (a) check: $f(C_1^\delta, Q) = f(e_1 h(C_1), Q) = f(h(C_1), e_1 Q) = f(h(C_1), V_1)$.
- Relation (b) check: $f(C_2, Q) = f(e_2 C_1, Q) = f(C_1, e_2 Q) = f(C_1, V_2)$.
- Relation (c) check: $f(C_1, Q) = f(e_1 M, Q) = f(M, e_1 Q) = f(M, V_1)$.

These three check points avoid the problem of the man-in-the-middle.

Bilinear pairing definition and properties

The bilinear pairings are mappings over elliptic curves namely they map a pair of points (from one or two elliptic curves) to an element that belongs to a multiplicative group in a finite field. This mapping is characterized by some proprieties. A bilinear pairing is defined as a map $f : G_0 \times G_1 \to G_T$, where G_0, G_1 and G_T are groups of order q (with q large prime number) that satisfies the following properties [97]:

1. Bilinearity: $f(aP,bQ) = f(P,Q)^{ab}$ for each $P \in G_0$, $Q \in G_1$ and $a,b \in \mathbb{Z}$.

2. Non-degeneracy: for each $P \in G_0$ there is $Q \in G_1$ so that $f(P,Q) \neq 1$. Observe that, if $G_0 = \langle P \rangle$ (that is, G_0 is generated by P) and $G_1 = \langle Q \rangle$, then $G_T = \langle g \rangle$ with $g = f(P,Q)$.

3. Computability: there is an efficiency algorithm to compute $f(P,Q)$ for each $P \in G_0$, $Q \in G_1$.

Corollary: if $f : G_0 \times G_1 \to G_T$ is a bilinear pairing, then $f(cP,Q) = f(P,cQ)$ for each $P \in G_0$, $Q \in G_1$ and $c \in \mathbb{Z}$.

This Corollary derives from the Bilinearity propriety: $f(cP,Q) = f(P,Q)^c$ but also $f(P,cQ) = f(P,Q)^c$ thus $f(cP,Q) = f(P,cQ)$.

Encryption and Decryption Functions

We suppose that the user 0 wants to send the plain text M_0 to the other users. In this scenario each i-th user has a key pair $e_{0,i} \in \mathbb{Z}_n^*$ and $V_{0,i} = e_{0,i} \cdot Q$, where $e_{0,i}$ is the private encryption key and $V_{0,i}$ is the public key used to verify the paternity of the message. The encryption function is: $E(M_o, e_{0,i}) = e_{0,i} \cdot M_o = C_o$. The decryption function is: $D(C_0, e_{0,i}^{-1}) = C_0 \cdot e_{0,i}^{-1} = M_0$. The encryption and decryption functions satisfy the hypothesis presented in section 3, subsection A.

5.5 Security and Many to Many communications

If the encryption key and the description key of each user don't change when the source user changes (from the user 0 to the user 1, the user 2 and so on...) there is a security problem. This problem happens because each i-th user (where $i = 0, ..., N$) knows the following parameters:

- the user i knows its encryption key $e_{0,i}$ and its decryption key $d_{0,i}$;
- the user i knows the values of the following shared keys:
  - $d_{s,0} = d_{0,1} \cdot d_{0,2} \cdot ... \cdot d_{0,N}$;
  - $d_{s,1} = d_{0,0} \cdot d_{0,2} \cdot ... \cdot d_{0,N}$;
  - ...
  - $d_{s,i} = d_{0,0} \cdot d_{0,1} \cdot d_{0,2} \cdot ... \cdot d_{0,i-1} \cdot d_{0,i+1} \cdot ... \cdot d_{0,N}$;
  - ...
  - $d_{s,N} = d_{0,0} \cdot d_{0,2} \cdot ... \cdot d_{0,N+1}$.
- in this way the i-th user can achieve the decryption keys (and the encryption keys) of the other users by solving a system of equations (with N+1 equations and N+1 variables, where the variables are $d_{0,0}, d_{0,1}, ..., d_{0,N}$):

$$\begin{cases} d_{s,0} = d_{0,1} \cdot d_{0,2} \cdot ... \cdot d_{0,N} \\ d_{s,1} = d_{0,0} \cdot d_{0,2} \cdot ... \cdot d_{0,N} \\ ... \\ d_{s,i} = d_{0,0} \cdot d_{0,1} \cdot d_{0,2} \cdot ... \cdot d_{0,i-1} \cdot d_{0,i+1} \cdot ... \cdot d_{0,N}) \\ ... \\ d_{s,N} = d_{0,0} \cdot d_{0,2} \cdot ... \cdot d_{0,N+1} \end{cases}$$

Thus there is a security problem because it is supposed that each user has to know only its encryption key, its decryption key and the shared decryption key. We can solve this problem by changing the encryption and decryption key of each user when the source user (that wants to send a message to the others users) changes, namely:

- when the source user is the $user_0$ each i-th user (where $i = 0, ..., N$) has the following keys:
  - encryption key: $e_{0,i}$;
  - decryption key: $d_{0,i}$;
  - shared decryption key (for all the user): $d_{s,0}$.
- ....
- when the source user is the $user_N$ each i-th user (where $i = 0, ..., N$) has the following keys:
  - encryption key: $e_{N,i}$;
  - decryption key: $d_{N,i}$;
  - shared decryption key (for all the user): $d_{s,N}$.

Where for each i-th user (for $i = 0, ..., N$) there are the following constraints:

- $e_{0,i} \neq e_{1,i} \neq e_{2,i} \neq ... \neq e_{N,i}$ (encryption keys)
- $d_{0,i} \neq d_{1,i} \neq d_{2,i} \neq ... \neq d_{N,i}$ (decryption keys)
- $d_{s,0} \neq d_{s,1} \neq d_{s,2} \neq ... \neq d_{s,N}$ (shared keys)

In this way it is not possible to have a system of equations (with N+1 equations and N+1 variables) used to achieve the decryption and encryption keys of the other users.

5.6 Decryption shared key generation and distribution

The generic decryption shared key is: $d_s = d_1 \cdot d_2 \cdot \ldots \cdot d_N$. Each j-th user of the network has to know the shared key d_s but the user doesn't have to know the private decryption keys of the other users. We achieve this aim using Elgamal Encryption System [96]. We suppose to have $N+1$ users and each i-th user wants to know d_s without knowing the private keys d_j of each j-th user (with $j = 1...N$ and $j \neq i$). We suppose that the user 0 wants to send the message M to all the other users. In this case the users 1,2,3 ... N have to share the decryption key d_s and the user 0 doesn't have to know this key. For hypothesis each i-th user (with i=1...N) generates a encryption key e_i and a decryption key d_i both private and all the other users don't have to know e_i and d_i. In this way we use the Elgamal Encryption System to generate and share the decryption shared key d_s (from d_1, d_2, ... , d_N) among the users 1,2, ... N.

Hypothesis about each i-th user (with $i = 1...N$) (Elgamal Protocol [96]):

- let p be a large prime.
- Let g be a prime element mod p, both known (all the users 0,1,...,N know them).
- The i-th user chooses a number a_i between 0 and $p-1$ where $0 \leq a_i < p-1$.
- The i-th user computes $z_i \equiv g^{a_i} \, mod \, p$.
- In this way the public key of the i-th user is (p, g, z_i).
- The private key of the i-th user is a_i.

Protocol. Now suppose that the (i-1)-th user wants to send to the i-th user a message M, where $0 \leq M \leq p-1$:

- i-th user public key (p, g, z_i).
- i-th user private key a_i.

The (i-1)-th user chooses a number k_{i-1} uniform between 1 and $p-1$ ($1 \leq k_{i-1} < p-1$) and computes:

- $C_1 = g^{k_{i-1}} \, mod \, p$.
- $M_C = \left(M \cdot z_i^{k_{i-1}}\right) mod \, p$.
- In this way the encrypted message is: $C = (C_1, M_C)$.

The (i-1)-th user sends C to the i-th user. The i-th user decrypts M from C as follows:

$C_1^{-a_i} M_C \, mod \, p \equiv \left(g^{k_{i-1}}\right)^{-a_i} M \cdot z_i^{k_{i-1}} \, mod \, p \equiv g^{-a_i k_{i-1}} \cdot M \cdot g^{a_i k_{i-1}} \, mod \, p \equiv M.$

Decryption shared key generation and distribution. In the Massey Omura protocol proposed in this chapter, we suppose to have a private encryption key e_i and a private decryption key d_i for each i-th user (for $i = 1, ..., N$). We use the Elgamal Formulation to make a system used to share the decryption key $d_s = d_1 \cdot d_2 \cdot ... \cdot d_N$ among the users 1,2,3, ... N such that each i-th user (with $i = 1, ..., N$) has to know d_s and doesn't have to know d_j for each $j = 1, ..., N$ and $j \neq i$. This system is described as follows.

1. The first user chooses a secret (and private) number $\beta_1 \in [1, p-1)$. He/she computes $d(u_1) = d_1 z_1^{\beta_1} \, mod \, p$ and sends it to the second user using the Elgamal Protocol.
2. The second user decrypts the message received from the first user and he/she achieves $d(u_1)$. The 2-nd user chooses a secret (and private) number $\beta_2 \in [1, p-1)$. He/she computes $d(u_2) = d_2 z_2^{\beta_2} d(u_1) \, mod \, p$ and sends it to the third user using the Elgamal Protocol.
3. ... And so on ...
4. The last user (N-th user) decrypts the message received from the (N-1)-th user and he/she achieves $d(u_{N-1})$. The N-th user chooses a secret (and private) number $\beta_N \in [1, p-1)$. He/she computes $d(u_N) = d_N z_N^{\beta_N} d(u_{N-1}) \, mod \, p \equiv d_1 d_2 \cdot ... \cdot d_N z_1^{\beta_1} z_2^{\beta_2} ... z_N^{\beta_N} \, mod \, p \equiv d_s z_1^{\beta_1} z_2^{\beta_2} ... z_N^{\beta_N} \, mod \, p$ and sends it to the first user with the Elgamal protocol.

Now all the users remove their private keys from the message as follows.

1. The first user decrypts the message received from the N-th user and he/she achieves $d(u_N) = d_s z_1^{\beta_1} z_2^{\beta_2} ... z_N^{\beta_N} \, mod \, p$. The first user computes $f_s(u_1) = z_1^{-\beta_1} d(u_N) \, mod \, p \equiv d_s z_2^{\beta_2} ... z_N^{\beta_N} \, mod \, p$ and sends it to the second user with the Elgamal protocol.

2. The second user decrypts the message received from the first user and he/she achieves $f_s(u_1)$. The 2-nd user computes $f_s(u_2) = z_2^{-\beta_2} f_s(u_1) \, mod \, p \equiv d_s z_3^{\beta_3} ... z_N^{\beta_N} \, mod \, p$ and sends it to the third user using the Elgamal Protocol.

3. ... And so on ...

4. The last user (N-th user) decrypts the message received from the (N-1)-th user and he/she achieves $f_s(u_{N-1})$. The N-th user computes $f_s(u_N) = z_N^{-\beta_N} f_s(u_{N-1}) \, mod \, p \equiv d_s z_N^{-\beta_N} z_N^{\beta_N} \, mod \, p \equiv d_s$. In this way the N-th user achieves the shared decryption key without knowing the decryption keys of the other users.

Finally the N-th user sends the shared decryption key d_s to each i-th user (with $i = 1, ..., N - 1$) using the Elgamal Protocol.

When the source node changes (in the Massey Omura Multiple User System) all the encryption and decryption keys have to change for the security problem described in section 5. For this reason it is necessary to use again the Mechanism described above used to share the new shared decryption key.

When a user leaves the secure network, he/she sends, through Massey Omura generalized protocol, his/her private encryption key to the other users so that all the other users of the group can remove this key from the shared decryption key.

5.7 Summary of the Chapter

In this chapter we have presented a simple generalization of the Massey Omura protocol which allows a user to send, a message or a shared key, to all the users of the same multicast group of the sender. The proposed protocol doesn't need an encryption network infrastructure because the encryption and the decryption are handled by each user of the group. Moreover, the proposed protocol provides a key distribution system distributed among the users of the multicast group thus the problem of the single point of failure is not present (instead in the case of the "Key Distribution Systems" such as Kerberos [22, 23], the single point of failure represents the main problem of this system).

In this chapter, the man-in-the-middle (main security problem of the Massey Omura protocol) and a particular security problem of the proposed protocol are addressed. We solved the problem of the man-in-the-middle using a bilinear pairing keys over elliptic curves. We used the bilinear pairing keys such that each user is able to check the paternity of the received message. The security problem of the proposed protocol is related to the many to many communications security. This problem is solved by changing the encryption and the decryption keys of each user of the multicast group when the source node (of the multicast communication) changes. Moreover the the reliable multicast performance results, presented in the article [91], shows that the reliable multicast using IRMA is as fast as the unicast TCP connection from the sender to the slowest receiver in the multicast group at any time. Thus our Massey Omura generalization provides better performance and lower occupation bandwidth with respect to the classic Massey Omura protocol in which the secure communication is only established for each couple of users (with an unicast TCP).

In the last part of this chapter we have presented a protocol (based on Elgamal protocol) used to share and distribute the shared decryption key of the proposed protocol.

Chapter 6

Conclusions

The multimedia contents distribution on a geographic network is a rarefied and huge field. First of all in this thesis we had classified the main parts necessary in the multimedia distribution on a geographic network. The main aspects of a geographic network that we had investigated are:

- the mechanism used to retrieve the sources of the multimedia content.
- The kind of overlay network (peer-to-peer) used to distribute the multimedia content.
- The usage of this overlay network in different field of the telecommunication network.
- The security of the overlay network over a geographic network.

Thus in a multimedia content distribution peer-to-peer network on geographic network there are many aspect to consider. First of all there is a huge number of works about the protocol used to retrieve the sources of a content in a peer-to-peer network (query flooding protocol), therefore, it would simply not be practical to cite the "most relevant" contributions to a "field" so difficult to define. Anyway all the works, about the query flooding protocol, provide only mechanisms that can be used to optimize the query flooding protocol and reduce the effects of the flooding on the network without specifying a mathematical model (in terms of average access time) for the query flooding protocol. Some of this works are presented in the articles [1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14]. Thus it is important to investigate this topic so that it is possible to achieve an analytical model of the query flooding protocol that can be used in any kind of query operation. We had achieved an analytical model for the query flooding protocol through a complex analysis of proxies network. In this analysis we can see how the proxies permit an improvement in the performance with respect to the routing operations in a generic network of routers. Moreover we addressed a simple formulation and framework about the performance of the network with and without layer 7 (proxy) and we applied it in three different types of scenarios to show the gain achieved with the usage of proxies instead of router. The multiline proxies network has better performance than a chain of proxies for the first two formulation about the hit probability. For this reason a multiline of proxies give us an improvement with respect to a chain of proxies. With actual devices this kind of approach (multiline proxies) is not expensive and produces big benefits in terms of information retrieval performance.

Through the query flooding operation, each peer of the peer-to-peer network can achieve the list of the peers that hold the desired multimedia content. In a multimedia content distribution system, after the previous step in which the list of the peers that hold the desired multimedia content is retrieved, it is necessary to establish the kind of peer-to-peer network used to distribute this multimedia content to the peers that require it. Thus the second aspect of an overlay network (peer-to-peer network) on geographic network is how is built the peer-to-peer network so that it is possible to provide the multimedia content to the vast majority of peers (that require this content) with the minimum delay. The construction of the peer-to-peer networks used for the distribution of the multimedia contents is not a very investigated field. In fact the most important real-time algorithm used to create a simple distribution peer-to-peer network on asymmetric channels is given in article [15] and the issues of performance of peer-to-peer file sharing over asymmetric and wireless networks is addressed in article [16]. Therefore it is very important to analyse the state of the art [15] and generate new algorithms used to build multimedia content distribution peer-to- peer network on asymmetric channel and radio channel. For this purpose we create new algorithms used to build peer-to-peer network in an incremental way and we had established which of these algorithms produce the peer-to-peer network characterized by the maximum number of peers accepted and the minimization of the maximum delay of the network. In this scenario we can conclude that the maximization of the average maximum number of peers that can access the multimedia content and the minimization of the average maximum delay of the network is achieved, in the case of the asymmetric channel; when the source node is a home-user (where $1 \leq B \leq 2$) by using the PLR algorithm without Input Blockage, as in section 3.4 subsection 3 we showed that the PLR algorithm without Input Blockage is closest to optimum when $1 \leq B \leq 2$ and $0 \leq p \leq 1$. When the source node is a server (where $B \geq 2$) the best algorithm is:

- the TR algorithm without Input Blockage when $0.46 \leq p \leq 1$.
- The PLR algorithm without Input Blockage when $0 \leq p \leq 0.46$.

We had evaluated the performance of these peer-to-peer network also on two different kind of radio channel models. In the first radio channel scenario we have a dependent error probability over the received bits. In the second radio channel scenario we have an independent error probability over the received bits. The evaluation of the performance of the peer-to-peer networks produced by the algorithms presented is done through an analytical model and a peer-to-peer Network Simulator on an Unreliable Channel. We had validated the analytical model through the simulation results using the autocorrelation test of hypotheses in both radio channel models. In this case we had achieved which peer-to-peer algorithm produces the best behaviour with respect to the percentage of correctly received bits. In the case of the radio channel with dependent error probabilities over the received bits, we can also conclude that the TR and PLR algorithms without Input Blockage are a big improvement in comparison to Mazzini-Rovatti's algorithm [15] provided that new network conditions are followed, because they are suboptimal with respect to the theoretical optimum. In the case of the radio channel characterized by dependent error probabilities over the received bits, the best behaviour with respect to the percentage of correctly received bits is obtained in the network generated by:

- the TR algorithm when the depth level is greater or equal to 4.
- The PLR algorithm and PLR algorithm without Input Blockage when the depth level is equal to 3.
- The PL algorithm when the depth level is less or equal to 2.

In the case of the radio channel characterized by independent error probabilities over the received bits, the best behaviour with respect to the average percentage of correctly received bits is obtained in the network generated by:

- the TR algorithm when the depth level is greater or equal to 4.
- The TR, PL, PLR and PLRwIB algorithms when the depth level is equal to 3.
- The T, PL, PLR and PLRwIB algorithms when the depth level is less or equal to 2.
- All the algorithms when the depth level is equal to 1.

There are other types of networks which can be used to distribute the multimedia content. The most popular of these kinds of networks is the multicast network. The most important problem of the multicast network is the reliability of the communications and this issue is solved using reliable multicast protocols. The reliable multicast is a very investigated field. In fact there are different mechanism used to have a reliable multicast network such as: peer-to-peer (P2P) tree based reliable multicast protocol [17], peer-to-peer epidemic algorithms for reliable multicasting [18], reliable peer-to-peer end system multicasting through replication in which it is presented an effective dynamic passive replication scheme designed to provide reliable multicast service in peer-cast [8], and many others [19, 20]. Therefore, it is interesting to see if there is an innovative mechanism that exploits the peer-to-peer network to make reliable a standard unreliable multicast network. The new mechanism that we introduce in this thesis is based on an unreliable multicast network that is made reliable using a particular peer-to-peer network which is build using the destination node of the multicast network and the source node of the multicast as peers of the peer-to-peer network. We compare the average access time of this type of system with the average access time in a reliable mutlicast network based on a particular ARQ mechanism. From the result presented in chapter 4, when there is high congestion of the multicast network the best approach is the unreliable network made reliable by the peer-to-peer network. When the congestion level of the multicast network is low the best approach is the reliable peer-to-peer network based on the ARQ mechanism presented in chapter 4. In this way we can minimize the average access time using an hybrid system in which: when the multicast network has a low level of congestion the system uses the ARQ reliable multicast. When the multicast network has an high level of congestion the system uses an unreliable multicast network with a peer-to-peer network used to avoid the packet loss.

Finally the last aspect of the geographic network is the security of the communications among a group of peers. Thus it is important to investigate the different class of cryptosystems. The cryptosystems are classified in two kinds: the symmetric cryptosystems and the asymmetric cryptosystems. Symmetric cryptosystems use a single secret key shared among a group of users that want to communicate. On the other hand in the asymmetric cryptosystems there is a public key and a private key for each user. The public key is used to cipher the message and the private key is used to decrypt the message. In this case the users do not have to share a private secret key but the computational complexity of the asymmetric cryptosystems is much higher than the symmetric cryptosystem as shown in the article "Symmetric and Asymmetric Encryption" [21]. There is another group of cryptography protocols in which the users do not have to share a secret key. In this way the level of security is much higher than the previous cryptography protocols. The number of studies about these protocols are very few and they only allow the communication between users for each execution of the protocol. Therefore these protocols do not allow the communications with one user who sends information to three or more users in a secure way. Examples of these protocols are Shamir's 3-pass protocol [22, 23], Khayat's protocol [24], Massey Omura [25] and other 3-pass protocols such as the one presented in the article [26]. Thus to ensure the maximum level of security with secure communications among a group of three or more peers it is necessary to create a new protocol, based for example on the Massey Omura protocol, which can allow the communications among the peers of a peer-to-peer network in a secure way. Therefore in the fifth chapter of the thesis we showed a generalization of the Massey Omura protocol to allow secure communication among a group of users or peers. Moreover we presented the security problems of this Massey Omura Multiple Users Protocol and how it is possible to avoid these issues through a specific encryption function and a specific decryption function by changing the encryption and decryption keys of each peer when the source peer changes. Finally we presented a new cryptography protocol which we used to share the decryption shared key which is used in the Massey Omura Multiple Users Protocol.

Some parts of this thesis contain information from my past publications such as [98, 99, 100, 78, 84] which are shown in the bibliography below.

Bibliography

[1] J. Guo and B. Li, “Mitigating information asymmetries to achieve efficient peer-to-peer queries,” *Parallel Processing, 2004. ICPP 2004. International Conference on*, vol. 1, pp. 91–98, August 2004.

[2] S. Jiang and X. Zhang, “Floodtrail: an efficient file search technique in unstructured peer-to-peer systems,” *Global Telecommunications Conference, 2003. GLOBECOM ’03. IEEE*, vol. 5, pp. 2891–2895, December 2003.

[3] S. Jiang, L. Guo, X. Zhang, and H. Wang, “Lightflood: Minimizing redundant messages and maximizing scope of peer-to-peer search,” *Parallel and Distributed Systems, IEEE Transactions on*, vol. 19, no. 5, pp. 601–614, May 2008.

[4] C. Tian, H. Jiang, X. Liu, and W. Liu, “Revisiting dynamic query protocols in unstructured peer-to-peer networks,” *Parallel and Distributed Systems, IEEE Transactions on*, vol. 23, no. 1, pp. 160–167, January 2012.

[5] R. Gaeta and M. Sereno, “Generalized probabilistic flooding in unstructured peer-to-peer networks,” *Parallel and Distributed Systems, IEEE Transactions on*, vol. 22, no. 12, pp. 2055–2062, December 2011.

[6] X. Liu, Y. Liu, and L. Xiao, “Improving query response delivery quality in peer-to-peer systems,” *Parallel and Distributed Systems, IEEE Transactions on*, vol. 17, no. 11, pp. 1335–1347, November 2006.

[7] K. Jayanthi and M. Karnan, "Implementation of an enhanced file search efficiency in p2p network," *Computational Intelligence and Computing Research (ICCIC), 2010 IEEE International Conference on*, pp. 1–4, December 2010.

[8] J. Zhang, L. Liu, C. Pu, and M. Ammar, "Reliable peer-to-peer end system multicasting through replication," *Peer-to-Peer Computing, 2004. Proceedings. Proceedings. Fourth International Conference on*, pp. 235 – 242, August 2004.

[9] W. Ai, L. Xinsong, and L. Kejian, "Efficient flooding in peer-to-peer networks," *Computer-Aided Industrial Design and Conceptual Design, 2006. CAIDCD '06. 7th International Conference on*, pp. 1–6, November 2006.

[10] H. Matsunam, T. Terada, and S. Nishio, "A query processing mechanism for top-k query in p2p networks," *Data Engineering Workshops, 2005. 21st International Conference on*, p. 1240, April 2005.

[11] N. Xiong, Y. Liu, S. Wu, L. Yang, and K. Xu, "An efficient search algorithm without memory for peer-to-peer cloud computing networks," *Parallel and Distributed Processing Workshops and Phd Forum (IPDPSW), 2011 IEEE International Symposium on*, pp. 1452–1457, May 2011.

[12] S. Tewari and L. Kleinrock, "Optimal search performance in unstructured peer-to-peer networks with clustered demands," *Communications, 2006. ICC '06. IEEE International Conference on*, pp. 36–41, June 2006.

[13] H. Kim, Y. Kim, S. Kang, and K. Kim, "Restricted path flooding scheme in distributed p2p overlay networks," *Information Science and Security, 2008. ICISS. International Conference on*, pp. 58–61, January 2008.

[14] S. Margariti and V. Dimakopoulos, “A novel probabilistic flooding strategy for unstructured peer-to-peer networks,” *Informatics (PCI), 2011 15th Panhellenic Conference on*, pp. 149–153, October 2011.

[15] G. Mazzini and R. Rovatti, “Peer-to-peer distribution on asymmetric channels,” *IEEE Communication Letters*, vol. 12, no. 9, pp. 699–701, September 2008.

[16] Y.-N. Lien, “Performance issues of peer-to-peer file sharing over asymmetric and wireless networks,” in *ICDCSW ’ 05: Proceedings of the First International Workshop on Mobility in Peer-to-Peer Systems*, Ohio, USA, June 2005, pp. 850–855.

[17] M. Yang and Y. Yang, “Mmc04-5: A peer-to-peer tree based reliable multicast protocol,” in *Global Telecommunications Conference, 2006. GLOBECOM ’06. IEEE*, San Francisco, California, USA, November-December 2006, pp. 1–5.

[18] z. . Genç, Z., “Peer-to-peer epidemic algorithms for reliable multicasting in ad hoc networks,” in *ICIT (International Conference on Information Technology)*, Istanbul, Turkey, December 2004.

[19] G. Benincasa, A. Rossi, N. Suri, M. Tortonesi, and C. Stefanelli, “An experimental evaluation of peer-to-peer reliable multicast protocols,” *Military Communications Conference, 2011 - MILCOM 2011*, pp. 1015–1022, November 2011.

[20] S. Doria and M. Spohn, “A multicast approach for peer-to-peer content distribution in mobile ad hoc networks,” *Wireless Communications and Networking Conference, 2009. WCNC 2009. IEEE*, pp. 1–6, April 2009.

[21] G. J. Simmons, “Symmetric and asymmetric encryption,” *ACM Comput. Surv.*, vol. 11, no. 4, pp. 305–330, December 1979.

[22] R. Oppliger, *Contemporary Cryptography.* Massachusetts: Artech House, Inc., 2005.

[23] A. J. Menezes, P. V. Oorschot, and S. Vanstone, Eds., *Handbook of Applied Cryptography*. Boca Raton: CRC Press, 1997.

[24] S. H. Khayat, "Using commutative encryption to share a secret," *available from http://eprint.iacr.org/2008/356.pdf*, August 18, 2008.

[25] J. K. Omura and J. L. Massey, "Computational method and apparatus for finite field arithmetic," United States Patent Patent 4,587,627, May 6, 1986.

[26] U. A. G. Degraf, P. M. Eduardo, S. A. Olivo, and M. C. Alberto, "A three-pass protocol for cryptography based on padding for wireless networks," in *Consumer Communications and Networking Conference, 2007. CCNC 2007. 4th IEEE*, Jan. 2007, pp. 287–291.

[27] IEEE, "Ieee standards for local area networks: Carrier sense multiple access with collision detection (csma/cd) access method and physical layer specifications," *ANSI/IEEE Std 802.3-1985*, 1985.

[28] ——, "Ieee standards for local area networks: Token-passing bus access method and physical layer specification," *ANSI/IEEE Std 802.4-1985*, 1985.

[29] ——, "Ieee standards for local area networks: Token ring access method and physical layer specifications," *IEEE Std 802.5-1989*, 1989.

[30] ——, "Ieee standard for information technology- telecommunications and information exchange between systems-local and metropolitan area networks-specific requirements-part 11: Wireless lan medium access control (mac) and physical layer (phy) specifications," *IEEE Std 802.11-1997*, pp. i–445, 1997.

[31] M. M. H. A. M. L. N. S. Leonardi, E. and D. Rossi, "Building a cooperative peer-to-peer-tv application over a wise network: The approach of the european fp-7 strep napa-wine," *IEEE Communications Magazine*, vol. 46, no. 6, pp. 20–22, April 2008.

[32] R. Birke, E. Leonardi, M. Mellia, A. Bakay, T. Szemethy, C. Kiraly, R. Cigno, F. Mathieu, L. Muscariello, S. Niccolini, J. Seedorf, and G. Tropea, “Architecture of a network-aware p2p-tv application: the napa-wine approach,” *Communications Magazine, IEEE*, vol. 49, no. 6, pp. 154 –163, June 2011.

[33] D. A. Riehl, “Peer-to-peer distribution systems: Will napster, gnutella, and freenet create a copyright nirvana or gehenna?” *William Mitchell Law Review*, vol. 27, pp. 1761–1781, 2001.

[34] KaZaA. [Online]. Available: http://www.kazaa.com

[35] “The gnutella protocol specification v0.4.” [Online]. Available: http://www.stanford.edu/class/cs244b/gnutella_protocol_0.4.pdf

[36] I. A. Matei, R. and P. Foster, “Mapping the gnutella network,” *Internet Computing*, vol. 6, no. 1, pp. 50–57, January - February 2002.

[37] Y. X. Wang, Y. and Y. Li, “Analyzing the characteristics of gnutella overlays,” in *Information Technology, 2007. ITNG ’07. Fourth International Conference on*, Las Vegas, Nevada, April 2007, pp. 1095–1100.

[38] P. Maymounkov and D. Mazieres, “Kademlia: A peer-to-peer information system based on the xor metric,” in *International Workshop on Peer-to-Peer Systems*. Cambridge, MA, USA: Springer-Verlag, March 2002, pp. 53–65.

[39] Ipoque, “Internet study 2008/2009,” 2009. [Online]. Available: http://www.ipoque.com/sites/default/files/mediafiles/documents/internet-study-2008-2009.pdf

[40] D. BrookShier, *JXTA: Java P2P Programming*. SAMS Publishing, 2002.

[41] B. Wilson, *JXTA*. New Riders, 2002.

[42] I. Clarke, S. Miller, T. Hong, O. Sandberg, and B. Wiley, “Protecting free expression online with freenet,” *Internet Computing, IEEE*, vol. 6, no. 1, pp. 40–49, January/February 2002.

[43] R. Dingledine, M. J. Freedman, and D. Molnar, *Free Haven. In Peer-to-peer*. O’Reilly, November 2000.

[44] ——, “The free haven project: distributed anonymous storage service,” *International workshop on Designing privacy enhancing technologies: design issues in anonymity and unobservability*, pp. 67–95, 2001. [Online]. Available: http://freehaven.net/doc/berk/freehaven-berk.ps

[45] A. D. R. Marc Waldman and L. F. Cranor, “Publius: A robust, tamper-evident, censorship-resistant, web publishing system,” *Proc. 9th USENIX Security Symposium*, pp. 59–72, August 2000.

[46] E. Korpela, D. Werthimer, D. Anderson, J. Cobb, and M. Leboisky, “Seti@home-massively distributed computing for seti,” *Computing in Science Engineering*, vol. 3, no. 1, pp. 78–83, January/February 2001.

[47] W. D. C. J. Bowyer, S. and M. Lampton, “Twenty years of serendip, the berkeley seti effort: Past results and future plans,” *Astronomical and Biochemical Origins and the Search for Life in the Universe*, p. 667, 1996.

[48] D. e. a. Anderson, “Internet computing for set,” *Bioastronomy 99: A New Era in Bioastronomy*, p. 511, 2000.

[49] RFC6120, “Request for comments: 6120,” in *Internet Engineering Task Force (IETF)*, P. Saint-Andre, March 2011.

[50] RFC6121, “Request for comments: 6121,” in *Internet Engineering Task Force (IETF)*, P. Saint-Andre, March 2011.

[51] RFC6122, "Request for comments: 6122," in *Internet Engineering Task Force (IETF)*, P. Saint-Andre, March 2011.

[52] Skype. [Online]. Available: www.skype.com

[53] BitTorrent. [Online]. Available: http://www.bittorrent.com

[54] A. Feldmann, R. Cáceres, F. Douglis, G. Glass, and M. Rabinovich, "Performance of web proxy caching in heterogeneous bandwidth environments," in *IN PROCEEDINGS OF IEEE INFOCOM '99*, March 1999, pp. 107–116.

[55] B. D. Davison, "A survey of proxy cache evaluation techniques," in *In Proceedings of the Fourth International Web Caching Workshop (WCW99)*, San Diego, California, March-April 1999, pp. 67–77.

[56] V. D. D. A. D. K. Thomas, H., "World wide wait: a study of internet scalability and cache-based approaches to alleviate it," *Management Science*, vol. 49, no. 10, pp. 1425–1444, October 2003.

[57] V. S. Mookerjee and Y. Tan, "Analysis of a least recently used cache management policy for web browsers," *Operations Research*, vol. 50, no. 2, pp. 345–357, March 2002.

[58] H. ElAarag and S. Romano, "Comparison of function based web proxy cache replacement strategies," in *Performance Evaluation of Computer Telecommunication Systems, 2009. SPECTS 2009. International Symposium on*, vol. 41, Istanbul, Turkey, July 2009, pp. 252–259.

[59] C. Kumar, "Performance evaluation for implementations of a network of proxy caches," *Decis. Support Syst.*, vol. 46, pp. 492–500, 2009.

[60] P. D. M. Tawarmalani and K. Karthik, "Allocating objects in a network of caches: centralized and decentralized analyses," *Management Science*, vol. 55, no. 1, pp. 132–147, January 2009.

[61] A. Wolman, M. Voelker, N. Sharma, N. Cardwell, A. Karlin, and H. M. Levy, "On the scale and performance of cooperative web proxy caching," *SIGOPS Oper. Syst. Rev.*, vol. 33, no. 5, pp. 16–31, December 1999.

[62] S. Dykes and K. Robbins, "A viability analysis of cooperative proxy caching," in *INFOCOM 2001. Twentieth Annual Joint Conference of the IEEE Computer and Communications Societies. Proceedings. IEEE*, vol. 3, Anchorage, Alaska, April 2001, pp. 1205–1214.

[63] ——, "Limitations and benefits of cooperative proxy caching," *Selected Areas in Communications, IEEE Journal on*, vol. 20, no. 7, pp. 1290–1304, September 2002.

[64] M. Abrams, C. R. Standridge, G. Abdulla, S. Williams, and E. A. Fox, "Caching proxies: Limitations and potentials," Virginia Polytechnic Institute & State University, Blacksburg, VA, USA, Tech. Rep., 1995.

[65] M. Rabinovich, J. Chase, and S. Gadde, "Not all hits are created equal: Cooperative proxy caching over a wide-area network," in *Computer Networks and ISDN Systems*, 1998, pp. 22–23.

[66] P. Rodriguez, C. Spanner, and E. Biersack, "Analysis of web caching architectures: hierarchical and distributed caching," *Networking, IEEE/ACM Transactions on*, vol. 9, no. 4, pp. 404–418, August 2001.

[67] M. Piatek, A. J. Jackson, and P. Juola, "Distributed web proxy caching in a local network environment," 2004.

[68] K. Tomoya and Y. Shigeki, "Application of peer-to-peer technology to marketing," in *Proceedings of the 2003 International Conference on Cyberworlds (CW' 03)*, Singapore, December 2003, pp. 372–279.

[69] F. L. Bakos, B. and J. K. Nurminen, "Peer-to-peer applications on smart phones using cellular communications," in *WCNC 2006 proceedings*, Las Vegas, Nevada, April 2006, pp. 2222–2228.

[70] M. M. M. M. Ciullo, D. and E. Leonardi, "Understanding peer-to-peer-tv systems through real measurements," in *IEEE GLOBECOM 2008 proceedings*, New Orleans, Louisiana, November - December 2008, pp. 1–6.

[71] J. D. Boever, "Value networks of peer-to-peer tv: An analysis of actors and their roles," in *The Third International Conference on Internet and Web Applications and Services*, Athens, Greece, June 2008, pp. 686–695.

[72] M. Y. Ma, M. and J. Wang, "The implementation and application of iptv supported on pull mode of peer-to-peer," in *International Symposium on Knowledge Acquisition and Modeling*, Wuhan, China, December 2008, pp. 371–374.

[73] R. A. M. M. Carter, C. and A. Bendiab, "Hybrid client-server, peer-to-peer framework for mmog," in *Multimedia and Expo (ICME), 2010 IEEE International Conference on*, Suntec City, Singapore, July 2010, pp. 1558–1563.

[74] V. Tay, "Massively multiplayer online game (mmog) - a proposed approach for military application," in *Cyberworlds, 2005. International Conference on Cyberworlds. 23-25 Nov. 2005. IEEE Computer Society*, Singapore, November 2005, pp. 396–400.

[75] N. Papandreou and T. Antonakopoulos, “Cooperative bit-loading and fairness bandwidth allocation in adsl systems,” in *Circuits and Systems, 2003. ISCAS ’ 03. Proceedings of the 2003 International Symposium on*, vol. 2, Bangkok, May 2003, pp. 352–255.

[76] RFC793, in *Transmission Control Protocol DARPA, Internet Program Protocol Specification*, Information Sciences Institute University of Southern California, 4676 Admiralty Way Marina del Rey, California 90291, September 1981.

[77] RFC791, in *Internet Protocol DARPA, Program Protocol Specification*, Information Sciences Institute University of Southern California, 4676 Admiralty Way Marina del Rey, California 90291, September 1981.

[78] D. Merlanti and G. Mazzini, “Peer-to-peer distribution on radio channel,” in *Proceedings of the 17th international conference on Software, Telecommunications and Computer Networks*, Hvar, September 2009, pp. 106–110.

[79] J. Proakis, *Digital Communications 4-th Edition*. New York: McGraw-Hill, August 2000.

[80] T. Soderstrom and P. Stoica, *System Identification*. Prentice Hall, July 1989.

[81] B. Waxman, “Routing of multipoint connections,” in *Broadband switching*, no. 6. Los Alamitos, CA, USA: IEEE Computer Society Press, 1991, pp. 347–352.

[82] A. Lakhina, J. Byers, M. Crovella, and I. Matta, “On the geographic location of internet resources,” *Selected Areas in Communications, IEEE Journal on*, vol. 21, no. 6, pp. 934–948, August 2003.

[83] A. AlWehaibi, M. Kadoch, A. Agarwal, and A. ElHakeem, “Packet loss probability for diffserv over ip and mpls reliable homogeneous multicast networks,” *Information Processing Letters*, vol. 90, pp. 73–80, April 2004.

[84] D. Merlanti and G. Mazzini, "Cooperative proxies network performance analysis," in *Software, Telecommunications and Computer Networks (SoftCOM), 2010 International Conference on*, September 2010, pp. 336–340.

[85] T. Dierks and E. Rescorla, "The transport layer security (tls) protocol version 1.2, rfc: 5246," IETF - Network Working Group, Tech. Rep., August 2008.

[86] S. Qing, W. Mao, J. Lopez, and G. Wang, "Secure multicast using proxy encryption," in *Information and Communications Security, 7th International Conference, ICICS 2005, Beijing, China, December 10-13, 2005, Proceedings*, vol. 3783, pp. 280–290.

[87] B. K. P., M. Friedemann, and S. Andre, Eds., *Theory and Practice in Distributed Systems, International Workshop, Dagstuhl Castle, Germany*, vol. 938. Springer, 1994.

[88] E. Wolfgang and T. Stefan, Eds., *Engineering Distributed Objects Second International Workshop, EDO 2000 Davis, CA, USA,*, vol. VIII. Springer, 2001.

[89] F. Milicchio and W. A. Gehrke, *Distributed Services with OpenAFS: for Enterprise and Education*. Springer, 2007.

[90] W. Dorothea and W. Roger, Eds., *Algorithms for Sensor and Ad Hoc Networks*. Springer, 2007, vol. 4621.

[91] K.-W. Lee, S. Ha, and B. V., "Irma: A reliable multicast architecture for the internet," in *INFOCOM '99. Eighteenth Annual Joint Conference of the IEEE Computer and Communications Societies. Proceedings. IEEE*, vol. 3. INFOCOM '99. Eighteenth Annual Joint Conference of the IEEE Computer and Communications Societies. Proceedings. IEEE, March 1999, pp. 1274 – 1281.

[92] N. Koblitz, "Elliptic curve cryptosystems," *Mathematics Of Computation*, vol. 48, no. 177, pp. 203–209, January 1987.

[93] D. B. Johnson and A. J. Menezes, "Elliptic curve dsa (ecsda): an enhanced dsa," in *Proceedings of the 7th conference on USENIX Security Symposium - Volume 7*. Berkeley, CA, USA: USENIX Association, 1998.

[94] V. S. Miller, "Use of elliptic curves in cryptography," in *Lecture notes in computer sciences; 218 on Advances in cryptology—CRYPTO 85*. New York, NY, USA: Springer-Verlag New York, Inc., 1986, pp. 417–426.

[95] A. M. Deusajute and P. S. L. M. Barreto, "The sip security enhanced by using pairing-assisted massey-omura signcryption," *X Reunión Española sobre Criptología y Seguridad de la Información – RECSI'2008, Salamanca, España, 2008. Lecture Notes in Computer Science. Heidelberg : Springer.*, vol. 5023, pp. 371–388, 2008.

[96] T. Elgamal, "A public key cryptosystem and a signature scheme based on discrete logarithms," *Information Theory, IEEE Transactions on*, vol. 31, no.4, pp. 469–472, Jul 1985.

[97] D. Boneh and M. Franklin, "Identity-based encryption from the weil pairing," *in Advances in Cryptology - CRYPTO 2001, ser. Lecture Notes in Computer Science*, vol. 2139, pp. 213–229, Springer-Verlag, 2001.

[98] D. Merlanti and G. Mazzini, "Massey omura multiple users key distribution," in *The 19th International Conference on Software, Telecommunications and Computer Networks (SoftCOM 2011)*, Split and Adriatic Islands, Croatia, September 2011, pp. 322–326.

[99] ——, "Peer-to-peer for unreliable multicast," in *The 19th International Conference on Software, Telecommunications and Computer Networks (SoftCOM 2011)*, Split and Adriatic Islands, Croatia, September 2011, pp. 446–450.

[100] ——, "Peer-to-peer multimedia distribution on radio channel and asymmetric channel," *INTECH, Open Book - Multimedia - A Multi-disciplinary Approach to Complex Issues*, pp. 1–26, April 2012.

List of Publications

Merlanti Danilo, Mazzini Gianluca. "Peer-to-Peer distribution on radio channel", *The 17th International Conference on Software, Telecommunications and Computer Networks (SoftCOM 2009)*, pp.106-110, Split - Hvar - Korcula, Croatia, 24-26 September 2009.

Merlanti Danilo, Mazzini Gianluca. "Cooperative proxies network performance analysis", *The 18th International Conference on Software, Telecommunications and Computer Networks (SoftCOM 2010)*, pp.336-340, Split and Adriatic Islands, Croatia, 23-25 September 2010.

Merlanti Danilo, Mazzini Gianluca. "Massey Omura Multiple Users Key Distribution", *The 19th International Conference on Software, Telecommunications and Computer Networks (SoftCOM 2011)*, pp. 322-326, Split and Adriatic Islands, Croatia, 15-17 September 2011.

Merlanti Danilo, Mazzini Gianluca. "Peer-to-Peer for Unreliable Multicast", *The 19th International Conference on Software, Telecommunications and Computer Networks (SoftCOM 2011)*, pp. 446-450, Split and Adriatic Islands, Croatia, 15-17 September 2011.

Merlanti Danilo, Mazzini Gianluca. "Peer-to-Peer Multimedia Distribution on Radio Channel and Asymmetric Channel", INTECH, *Open Book - Multimedia - A Multi--disciplinary Approach to Complex Issues*, ISBN:979-953-307-866-2.