Multipath routing protocol for mobile ad hoc networks

Jiazi Yi

UNIVERSITE DE NANTES

ÉCOLE DOCTORALE

« SCIENCES ET TECHNOLOGIES DE L'INFORMATION ET DE MATHEMATIQUES »

Année : 2010

Thèse de Doctorat de l'Université de Nantes

Spécialité : AUTOMATIQUE ET INFORMATIQUE APPLIQUEE

Présentée et soutenue publiquement par

Jiazi YI

le 19 novembre 2010

à l' École Polytechnique de l'Université de Nantes

Protocole de routage à chemins multiples pour des réseaux ad hoc

Jury

Président	:	M. Philippe Jacquet	*Professeur, École Polytechnique, Palaiseau*
Rapporteurs	:	M. Rodolphe Vauzelle	*Professeur, Université de Poitiers*
		M. Djamal Zeghlache	*Professeur, Telecom SudParis*
Examinateurs	:	M. Jeanpierre Guédon	*Professeur, Université de Nantes*
		M. Benoît Parrein	*Maître de Conférences, Université de Nantes*
Membre Invité:			
		M. Laurent Reynaud	*Ingénieur, Orange Labs R&D, Lannion*

Directeur de Thèse : Jeanpierre Guédon
Laboratoire : IRCCyN
Co-encadrant : Benoît Parrein
Laboratoire : IRCCyN
Composante de rattachement du directeur de thèse : École Polytechnique de l'Université de Nantes

N° ED 503 –xxx

Acknowledgements

The preparation of this thesis has benefited from the guidance of many people, but no more than Benoît Parrein, my advisor. During three years of my PhD study in France, they have spend uncountable time helping me on the research work, from choosing the research topic to formulating the problems, from test design to carrying out the experiments, and from paper writing to the redaction of the thesis. My director Jeanpierre Guédon also offered a lot of help during the writing of the thesis. I am truly grateful for their efforts.

I would like to thank Salima Hamma, my advisor of the master thesis in Polytech'Nantes, who gives me a lot of help during my first half year in the IVC team. Without her, I will not start my research in this domain. During my internship, Eddy Cizeron also helped me a lot in the study of network simulations.

I am also very fortune to be able to work with Sylvain David, Xavier Lecourtier, Pascal Lesage and Asmaa Adnane in the SEREADMO project. They helped me a lot in the implementation of the testbed, the experimentation, and the research of the compatibility. I would also like to thank Wissam Hamdach, who worked with me for his master thesis on the queue length metrics.

During the redaction of the thesis, I also got a lot of help from Dr. Thomas Clausen, Ulrich Herberg (Hipercom Team) and Henning Rogge (OLSRd team). They gave a lot of valuable suggestions on the OLSR and its implementation.

I would like to thank all the members of IRCCyN-IVC team and my friends during my stay in France. I had a lot of pleasure with them during the three years of the PhD study.

Finally, I would like to thank my parents, Xiaowen Yi and Ping Liang, for everything that they have given to me. They taught me the joy of love, the importance of family, and they have always been supportive for everything I have done. Without their support, encouragement and love, I cannot image that I could accomplish this work.

II

Contents

List of Figures

List of Tables

Introduction

Wireless communication may be one of the most exciting technologies in the 20th century. In 1901, the Italian inventor Guglielmo Marconi tested a transatlantic radio telegraph system by using Morse Code. It can be regarded as the beginning of modern wireless communication. Nowadays, wireless techniques have been widely used in our lives: from TV telecontrol to bluetooth mouse/earphone, from digital TV to Global Position System (GPS), and from 3G cell phone to Wireless Local Area Networks (WLAN). It is believed that wireless is the wave of the future, and more and more devices (laptop, desktop, monitor, sensor ...) will throw their communication cable away.

In the 21st century, the wireless technology is still developing rapidly and trying to be "Faster, Higher, and Stronger": faster data rate, higher bandwidth and stronger connectivity. Today, we can control our computers through bluetooth telecommand, make video calls and upload pictures anytime, anywhere to share with our friends. With the third generation of cellular wireless standards being widely used in the world, the fourth generation (4G) is also on its way. In December 2009, the Long Term Evolution (LTE) service was publicly available in Stockholm and Oslo. In the following years, more and more countries will join the "4G club". Another widely used wireless technology is IEEE 802.11-based WLAN (Wi-Fi). It is mostly been used for providing wireless data connectivity in local areas. The speed has also increased significantly from 1 Mbit/s (802.11b) to 150 Mbit/s (latest 802.11n).

With the fast development of the telecommunications, a decentralized wireless network called *mobile ad hoc network* (MANET) was proposed in the 1970s. Since the 1990s, it has drawn more and more attention. Compared to the WLAN and cellular technologies, MANET does not rely on any preexisting infrastructure, such as routers or access points. Each node plays as a router by forwarding data for other nodes. This kind of networks is suitable for the applications where central infrastructure is undesirable. The possible applications can be wireless mesh networks, sensor networks, vehicular ad hoc networks, mobile networks for military and disaster recovery, etc.

By providing great mobility and flexibility, MANETs also introduced a lot of technical challenges and difficulties, such as spectrum allocation, media access, energy efficiency, security, etc. One of the most difficult issues in MANET is the routing protocol, because the topology of the network maybe changing all the time, which makes the fixed routing protocol unavailable in this circumstance.

In this PhD thesis, we try to find a reliable routing protocol for MANET. We propose to use multipath algorithm to get different routes (node-disjoint or link-disjoint) from the source to the destination.

We will first introduce related technologies in the literature in Part I. Three aspects of related works are presented. Chapter 1 is about the mobile networks, especially ad hoc networks. After a

short introduction to the network reference models and general characteristic of ad hoc networks, the rest of the chapter deals with the routing protocols for ad hoc networks. There are two major categories of routing protocols: proactive (table driven) routing and reactive (on-demand) routing. Some hybrid routing and multipath routing protocols which based on those two prototypes are also introduced. In the literature, some of the protocols have been implemented in real testbeds, which donate valuable experience and results, so the implementation and testbeds are also summarized in the end of the chapter. Chapter 2 introduces one of the most interesting applications over networks: the video services and transportation. We begin with an overview of the video compression standard, and then present related techniques for video transmission over IP networks. In Chapter 3, we present the Forward Error Correction coding in the literature. Two radon discrete transforms: Mojette and Finite Radon Transform (FRT) are introduced, which can be used to reduce packet loss during data transmission.

In Part II, we exposed our main contribution by presenting Multipath Optimized Link State Routing Protocol (MP-OLSR). The SEREADMO project is introduced in Chapter 4 as the background of the research work of this thesis. Chapter 5 is the about the specification of MP-OLSR. MP-OLSR is a multipath extension of well-known OLSR protocol. It inherits the topology sensing mechanism of OLSR, and we propose the Multipath Dijkstra Algorithm to get the multiple paths from the source to the destination. The source routing technique is used for packet forwarding, combined with *route recovery* and *loop detection* to reduce route failure and transient loops in the network. The queue length link metric and the compatiblity between single path and multipath protocols are also discussed. In Chapter 6, the protocols are simulated in NS2 and Qualnet simulators. All the ideas and propositions mentioned in Chapter 5 are implemented in the simulator to validate the availability. We demonstrated our deployment of testbed in Chapter 7, and the test was taken in the campus of Polytech'Nantes. All the simulation and testbed results are discussed in detail to demonstrate the advantages of multipath routing.

Part III focuses on the H.264/SVC video application over the MP-OLSR protocol. In Chapter 8, we introduced the transportation interface of H.264/SVC, especially the scalability characteristic. The priority of different scalable layers is studied, which is important for data protection during the network transmission. Chapter 9 is about the multipath transportation of H.264/SVC. An evaluation framework, called *SVCEval* is built to simulate the video application over various kinds of networks. We proposed to use FRT for priority FEC coding, and integrated the FEC simulation module *FECSim* in *SVCEval* framework. Based on the framework and PFEC mechanisms proposed, the simulations of video transmission are also done.

We conclude the thesis in the last chapter with main conclusions and future perspectives. The related implementations and tools are introduced in the Appendix. The material and code are available on the site of IRCCyN or my personal site:

http://www.irccyn.ec-nantes.fr/

http://www.jiaziyi.com/

Part I

State of the Art

Chapter 1

Mobile ad hoc Networks

In Mobile Ad hoc NETwork (MANET), all nodes are routers and forward packets without any infrastructure or centralized management. This kind of network is spontaneous, self-organized and self-maintained. Given the great flexibility provided by MANETs, we are also facing a lot of challenges in spectrum allocation, media access, energy efficiency, packet routing, etc. In all these issues, one of the most difficult problems is how to effectively route the packets, due to the unreliability of wireless medium and the dynamic topology.

In this chapter, different aspects of MANETs are introduced. We begin with a general introduction of wireless networks, the reference models and related wireless technologies used in physical layer and data link layer. Then we present an overview of MANETs, their challenges, difficulties and applications. We focus on the routing protocol for MANETs, which is also the main issue to be further discussed in this thesis. The two main categories of routing protocols for MANETs are presented with their extensions: multipath, QoS, security, etc. In the end of this chapter, we introduce some testbeds built for MANETs, which is also a crucial aspect of the research in MANETs because they bring the MANETs into real world.

1.1 Wireless Mobile Networks

In this section, two important network models are firstly introduced: the OSI reference model and TCP/IP reference model. Then, some widely used wireless technologies are presented.

1.1.1 Network Reference Model

1.1.1.1 The OSI Reference Model

The OSI model is based on a proposal developed by the International Standards Organization (ISO) as a first step toward international standardization of protocols used in the various layers and it was revised in 1995. The model is called the ISO OSI (Open Systems Interconnection) Reference Model because it deals with connecting open systems—that is, systems that are open for communication with other systems. Figure 1.1 shows the architecture of OSI model.

There are seven layers included in the OSI model, from the Physical Layer at bottom, to the Application Layer at the top.

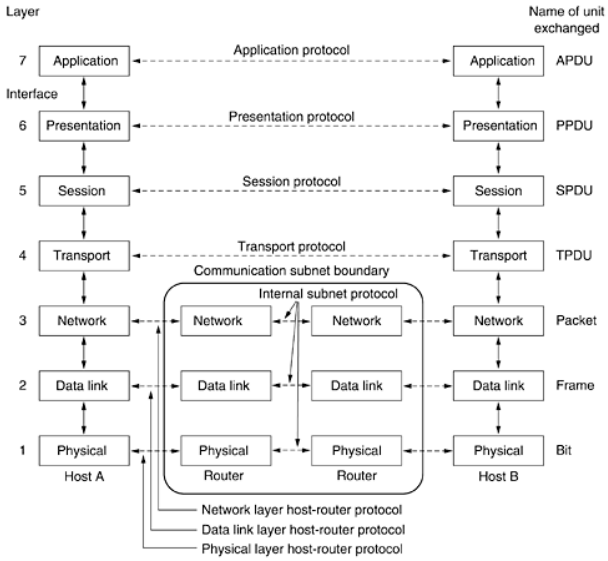


Figure 1.1: The OSI reference model (extracted from (Tanenbaum, 2003))

The Physical Layer The Physical Layer is the first and lowest layer in the seven-layer OSI model of computer networking. It is concerned with transmitting raw bits over a communication channel, and consists of the basic hardware transmission technologies of a network. It is the fundamental layer of the network which underlies the logical data structures of the higher level functions.

The Data Link Layer The Data Link Layer is the Layer 2 of the OSI model, which transfers data between adjacent network nodes in a wide area network or between nodes on the same local area network segment. It accomplishes this task by having the sender break up the input data into data frames (typically a few hundred or a few thousand bytes) and transmitting the frames sequentially.

Another issue related to the data link layer is traffic regulation: how to manage the traffic between a faster sender and a slow receiver. Certain flow regulation is often applied to manage the buffer and handle the errors.

For broadcast and wireless networks, the data link layer also has to deal with the problem of accessing control of the shared channel. So the Medium Access Control (MAC) protocol is also in this layer.

The Network Layer The Network Layer is layer 3 of the OSI model. It controls the operation of the subnet, and is responsible for end-to-end (source to destination) packet delivery including routing through intermediate hosts. A key design issue is determining how packets are routed from source to destination. To achieve the goals of the network layer, the following functions are included:

- Connection model: The network layer protocol can be either connection-oriented, or connectionless.
- Host addressing: Every host in the network needs to have an unique identifier, i.e. address to determine who it is. The most widely used is IPv4 address, and IPv6 is the next generation.

- Message forwarding and routing: the networks use gateways or routers to forward the packets in the networks. The routing protocol makes choice of the route, and specifies how routers communicate with each other, disseminating information that enables them to select routes between any two nodes on a computer network.

The Transport Layer The Transport Layer is layer 4 of the OSI model. The basic function of the transport layer is to accept data from above, split it up into smaller units if need be, pass these to the network layer, and ensure that the pieces all arrive correctly at the other end. Furthermore, all this must be done efficiently and in a way that isolates the upper layers from the inevitable changes in the hardware technology.

The Session Layer The Session Layer is layer 5 of the OSI model. It provides the mechanism for opening, closing and managing a session between end-user application processes. Sessions offer various services, including dialog control (keeping track of whose turn it is to transmit), token management (preventing two parties from attempting the same critical operation at the same time), and synchronization (checkpointing long transmissions to allow them to continue from where they were after a crash) (Tanenbaum, 2003).

The Presentation Layer The Presentation Layer is layer 6 of the OSI model. It is concerned with the syntax and semantics of the information transmitted. In order to make it possible for computers with different data representations to communicate, the data structures to be exchanged can be defined in an abstract way, along with a standard encoding to be used "on the wire". The presentation layer manages these abstract data structures and allows higher-level data structures (e.g., banking records), to be defined and exchanged.

The Application Layer The Application Layer is layer 7, and also the top layer of the OSI model. It provides services for application programs to enable effective communication with other application programs. It deals with the issues like network transparency, and is concerned with the user's view of network. Some widely-used services, like HTTP (Hyper Text Transfer Protocol), DNS (Dynamic Name System), FTP (File Transfer Protocol), IMAP (Internet Message Access Protocol), etc.

1.1.1.2 The TCP/IP Reference Model

Although OSI model is the standard proposed by ISO, a more widely used model is TCP/IP. It is created in the 1970s by DARPA, an agency of the United States Department of Defense. It was evolved from ARPANET, which was the world's first wide area network and a predecessor of the Internet.

TCP/IP is generally described as having four abstraction layers. The three top layers in the OSI model—the Application Layer, the Presentation Layer and the Session Layer—are not distinguished separately in the TCP/IP model where it is just the Application Layer.

The layer below the application layer in the TCP/IP model is now usually called the transport layer. It is designed to allow peer entities on the source and destination hosts to carry on a conversation, just as in the OSI transport layer. On this layer, two transport protocols are defined. Transmission Control Protocol (TCP) is a connection-oriented protocol which tries to provide reliable services. The User Datagram Protocol (UDP) is a connectionless protocol without sequencing

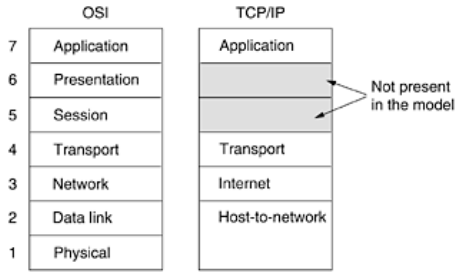


Figure 1.2: TCP/IP and OSI reference model (extracted from (Tanenbaum, 2003))

and flow control. It is widely used for one-shot, client-server-type request-reply queries and applications in which prompt delivery is more important than accurate delivery, such as transmitting speech or video.

The Internet Layer is to permit hosts to inject packets into any network and have them travel independently to the destination (potentially on a different network). It defines an official packet format and protocol called IP (Internet Protocol). The job of the internet layer is to deliver IP packets where they are supposed to go. Packet routing is clearly the major issue here, as is avoiding congestion. For these reasons, it is reasonable to say that the TCP/IP internet layer is similar in functionality to the OSI network layer. A comparison of OSI and TCP/IP reference model is presented in Figure 1.2.

1.1.2 Wireless Technologies

1.1.2.1 Characteristics of Wireless Network

Wireless network refers to any type of computer network that is wireless, and is commonly associated with a telecommunications network whose interconnections between nodes is implemented without the use of wires (Goldsmith, 2005). The vision of wireless communications supporting information exchange between people or devices is the communications frontier of the next few decades, and much of it already exists in some form. This vision will allow multimedia communication from anywhere in the world using a small handheld device or laptop.

To enable the wireless applications, a lot of technical challenges must be addressed. The wireless devices must be small enough to be portable, and incorporate multiple modes of operation to support different services and data. The power consuming is also an important issue for mobile devices, because the consumers do not want heavy batteries and recharge them every hour. So transmission and signal processing in the portable terminal must consume the minimal power.

The design of wireless networks has fundamental difference from the design of wired network due to the nature of the wireless channel. This channel is an unpredictable and difficult communications medium. The radio spectrum is a scarce resource that must be allocated to many different applications and systems. For this reason, spectrum is controlled by regulatory bodies both regionally and globally. A regional or global system operating in a given frequency band must obey the restrictions for that band set forth by the corresponding regulatory body. Spectrum can also be very expensive since in many countries spectral licenses are often auctioned to the highest bidder.

When a wireless signal is propagating through a wireless channel, it experiences random fluctuations and attenuation. This problem is even more serious when the sender, receiver or surrounding

objects are moving. Thus, the characteristics of the wireless channel appear to change randomly with time, which makes it difficult to design reliable systems with guaranteed performances.

Given the wireless transmission a broadcast channel most of the time, security is another problem that is much more difficult than the wired one. The analog cellular systems have almost no security that one can easily listen the conversations by scanning the analog cellular frequency band. In the digital cellular systems, some level of encryption is implemented. However, with enough knowledge, time and determination most of these encryption methods can be cracked and, indeed, several have been compromised. To support applications like electronic commerce and credit card transactions, the wireless network must be secure against such listeners.

The networking through wireless channel is another challenge. The network must be able to locate a given node in millions of distributed mobile hosts and route the messages to the defined destinations which might moving in a free way with high speed. The finite resources of the network must be allocated in a fair and efficient manner relative to changing user demands and locations. Moreover, there currently exists a tremendous infrastructure of wired networks: the telephone system, the Internet, and fiber optic cable, which should be used to connect wireless systems together into a global network. However, wireless systems with mobile users will never be able to compete with wired systems in terms of data rates and reliability. Interfacing between wireless and wired networks with vastly different performance capabilities is a difficult problem.

The overall design of the wireless network system might be the most significant challenge. As stated in Section 1.1.1, in wired networks, a layered approach is used. Protocols associated with different layers of the system operation are seperately designed, with baseline mechanisms to interface between layers. The wireless network model inherits from the wired model. This layering model can reduces the complexity and facilitates modularity and standardization, but because of the lack of global optimization, it can also leads to inefficiency and low performance of the system. The dynamic nature and poor performance of the underlying wireless communication channel indicates that high-performance networks must be optimized for this channel and must be robust and adaptive to its variations, as well as to network dynamics. Thus, these networks require integrated and adaptive protocols at all layers, from the link layer to the application layer.

1.1.2.2 Physical Layer for Wireless Networks

For the modern wireless transmission, the electromagnetic waved are employed. These waves were predicted by the British physicist James Clerk Maxwell in 1865 and first observed by the German physicist Heinrich Hertz in 1887. The number of occurrences of a repeating wave per unit time is called *frequency,* f, and is measured in Hz. The distance between two consecutive maxima (or minima) is called the wavelength, which is designated by λ.

The electromagnetic spectrum with different frequencies is shown in Figure 1.3.

As shown in the figure, the radio waves have long wave length and are easy to generate and can travel long distances. They are widely used in both indoor and outdoor radio communications. The microwave (above 100 MHz) travels in nearly straight lines and can therefore be narrowly focused. It is widely used for long-distance telephone communication, mobile phones, television distribution, and other uses that a severe shortage of spectrum has developed. Unguided infrared and millimeter waves are widely used for short-range communication. For example, the remote controls used on televisions, VCRs, and stereos all use infrared communication.

As presented in those examples, the properties of the electromagnetic waves are frequency

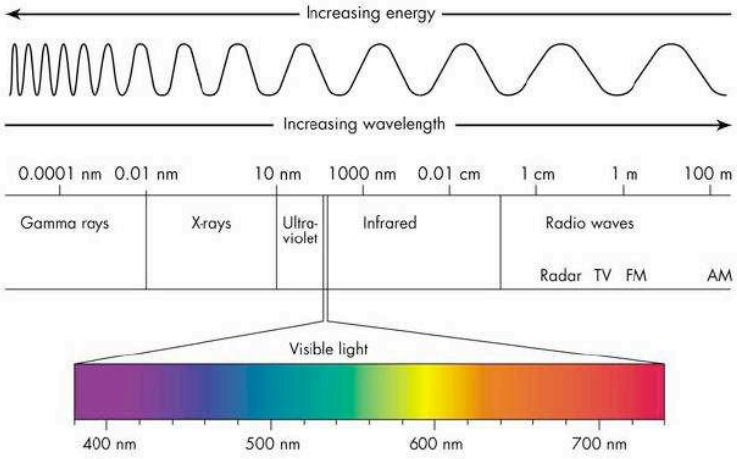


Figure 1.3: The electromagnetic spectrum

dependent. It is important to choose the appropriate physical layer for the specific wireless transmission.

1.1.2.3 Data Link Layer for Wireless Networks

The data link layer tries to provide a well-defined service interface to the network layer, deal with transmission errors and regulate the flow. For wireless network, the issue how to determine who gets to use the channel when there is competition for it. There is a sublayer called MAC (medium access control) to deal with this problem. One of the most popular protocols is 802.11 MAC (Gast, 2002).

802.11 is a member of the IEEE 802 family, which is a series of specifications for local area network (LAN) technologies. IEEE 802 specifications are focused on the two lowest layers of the OSI model because they incorporate both physical and data link components. All 802 networks have both a MAC and a Physical component. The MAC is a set of rules to determine how to access the medium and send data, but the details of transmission and reception are left to the physical layer. The structure of the 802 family is shown in Figure 1.4.

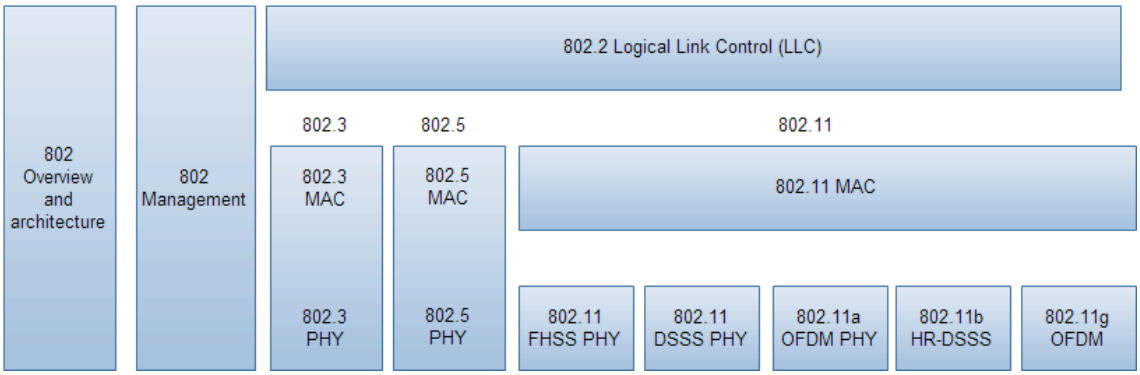


Figure 1.4: The IEEE 802 family

The base 802.11 specification includes the 802.11 MAC and several physical layers: a frequency hopping spread-spectrum (FHSS) physical layer and a direct-sequence spread-spectrum (DSSS)

link layer. Later revisions to 802.11 added additional physical layers. 802.11b specifies a high-rate direct-sequence layer (HR/DSSS); products based on 802.11b hit the marketplace in 1999 and make up the bulk of the installed base. 802.11a describes a physical layer based on orthogonal frequency division multiplexing (OFDM), which is the first of the high-speed wireless LANs. An enhanced version of 802.11b, 802.11g, was approved by IEEE in November 2001 after much politicking about whose patented technology it would use. It uses the OFDM modulation method of 802.11a but operates in the narrow 2.4-GHz ISM band along with 802.11b. It can operate at up to 54MBps. In the meantime, the HiperLAN (High Performance Radio LAN) proposed by ETSI (European Telecommunications Standards Institute) will never see the market.

For the 802.11 MAC, two modes are supported. The first one is called DCF (Distributed Coordination Function) which does not need any kind of central control. And the other one is PCF (Point Coordination Function), which uses the base station to control all activities in its cell.

1.2 Overview of Wireless ad hoc Networks

1.2.1 Characteristics of Wireless ad hoc Networks

1.2.1.1 History

The term "ad hoc" is a Latin expression that means *for this* or more specifically *for this purpose only*. This term has been applied especially to wireless networks comprised of communicating entities that belong to a network only during a communication session and are within radio range with other entities (Boukerche, 2006). In an ad hoc network, connections among entities are usually temporary since nodes may be added or removed either logically or physically. In general, ad hoc networks operate in a standalone way, but may be connected to another network such as the Internet. Without an inherent infrastructure, the mobiles handle the necessary control and networking tasks by themselves, generally through the use of distributed control algorithms. Multihop routing, whereby intermediate nodes relay packets towards their final destination, can improve the throughput and power efficiency of the network.

The idea of having an infrastructureless network was proposed in the 1970s. A series of research in ad hoc networking was supported by the DARPA (Defense Advanced Research Projects Agency). The first generation is called PRNET (Packet Radio Networks), which several wireless terminals could communicate with one another on a battlefield. It is in conjunction with ALOHA (Areal Locations of Hazardous Atmospheres) and CSMA (Carrier Sense Medium Access), approaches for medium access control and distance-vector routing.

In the 1980s, the ad hoc network is further improved and implemented as a part of Survivable Adaptive Radio Networks (SURAN). Its goal is to build a packet-switch network for the battlefield without any infrastructure.

In the 1990s, the ad hoc technology is applied in commercial applications. The notebook and other portable equipment employed wireless interfaces. The IEEE 802.11 subcommittee adopted the term "ad hoc model". In the meantime, a lot of MAC layer protocol and routing protocols are proposed for multihop data transmission.

The characteristics of ad hoc networks include (Mueller et al., 2004):

- Unreliability of wireless medium. Compared to the wired network, the communication through the wireless medium is unreliable and subject to errors. Also, due to varying envi-

ronmental conditions such as high levels of electro-magnetic interference (EMI) or inclement weather, the quality of the wireless link may be unpredictable.

- Unpredictability of environment. Ad hoc networks may be deployed in unknown terrains, hazardous conditions, and even hostile environments where tampering or the actual destruction of a node may be imminent. Depending on the environment, node failures may occur frequently.

- Resource-constrained nodes. Nodes in a MANET are typically battery powered as well as limited in storage and processing capabilities. Moreover, they may be situated in areas where it is not possible to re-charge and thus have limited lifetimes. Because of these limitations, they must have algorithms which are energy-efficient as well as operating with limited processing and memory resources. The available bandwidth of the wireless medium may also be limited because nodes may not be able to sacrifice the energy consumed by operating at full link speed.

- Dynamic topology. The topology in an ad hoc network may change constantly due to the mobility of nodes. As nodes move in and out of range of each other, some links break while new links between nodes are created.

1.2.1.2 Challenges and Difficulties

The most fundamental aspect of the wireless ad hoc network is that it is without any infrastructure. This makes it have big differences with the existed wireless networks, including cellular system and WLAN. Given the features of the ad hoc network, it is prone to different faults, which include transmission errors, node failures, link failures, route breakages, and congested node & links, etc. So the following issues should be carefully considered (Toh, 2002):

- Spectrum Allocation and Purchase. The radio spectrum is also an important resource. Regulations regarding the use of radio spectrum are under the control of the FCC. To prevent interference, ad hoc networks must operate over some form of allowed or specified spectrum range.

- Media Access. Because there is no centralized control and global synchronization in ad hoc networks, the media access becomes more difficult. Since the same media are shared by multiple nodes, access to the common channel must be made in a distributed fashion. So the MAC protocol must contend for access to the channel while at the same time avoiding possible collisions with the neighboring nodes.

- Energy Efficiency. For the static hosts and routers, the power consumption is not a big issue. For the nodes in ad hoc networks, most of them are driven by batteries. So the mobile devices in the networks should be energy efficient because the nodes are not only endpoints, but also intermediate nodes.

- TCP Performance. The TCP is a connection-oriented transport protocol that tries to provide reliable services with flow and congestion control. However, the TCP is unable to distinguish the presence of mobility and network congestion. The frequent link failure will result in packet loss and retransmission. How to guarantee the performance of the TCP protocol over ad hoc network networks still need to be further studied.

- Multicasting. The multicast backbone comprises an interconnection of multicast routers that are capable of tunneling multicast packets through non-multicast routers. Some multicast protocols use a broadcast-and prune approach to build a multicast tree rooted at the source. However, it requires that the routers are static, and cannot be used in ad hoc networks.
- Security & Privacy. The nodes in the ad hoc networks are fragile, especially in the battlefields or the applications in disaster recovery. So the security and privacy have to be considered.
- Routing. The mobility of the network will make the links between the nodes unstable. The existing distance-vector and link-state based routing protocols are unable to be used in ad hoc warless networks, because they cannot adapt to frequent link failures. This is the most challenging issue in the network layer.

1.2.2 Application of Wireless ad hoc Networks

The wireless ad hoc networks and their variations can be applied in different applications, including mesh networks, opportunistic networks, vehicular networks and wireless sensor networks. In this subsection, different cases of ad hoc networks are presented.

1.2.2.1 Mesh Networks

Mesh networks are built upon a mix of fixed and mobile nodes interconnected via wireless links to form a multihop ad hoc network. Unlike pure MANETs, a mesh network introduces a hierarchy in the network architecture by adding dedicated nodes (called mesh routers) that communicate wirelessly to construct a wireless backbone (Conti & Giordano, 2007).

The wireless mesh networks are the ideal solutions to provide both indoor and outdoor broadband wireless connectivity in urban, suburban, and rural environments without the need for extremely costly wired network infrastructure.

Nortel Wireless Mesh Network At National Taiwan University, a public WLAN trial site is build in 2008 to provide internet access and related services:

- Public WLAN: Providing campus-wide indoor and outdoor coverage to NTU's professors, students, and staff as benchmark, leading-edge wireless mesh deployment in Taiwan;
- Partner with Taiwan local companies for network build-out;
- Includes a comprehensive network solution: Up to 17 Wireless AP 7220s, 1 Passport 1424 (NAP-R), 1 Optivity NMS Software, Wireless Gateway 7250, 2220/2221 WLAN Mobile Adapters.

Figure 1.5 presents the coverage of the network in the campus and Figure 1.6 shows the architecture of the mesh network.

Intelligent transportation systems Portsmouth Real-Time Travel Information System (PORTAL) [1] as shown in Figure 1.7 is aimed at providing real-time travel information to bus passengers in the city of Portsmouth. This system is realized by equipping more than 300 buses with mesh

[1] Portsmouth Real-Time Travel Information System, http://findarticles.com/p/articles/mi_m0EIN/is_2004_July_27/ai_n6125447/

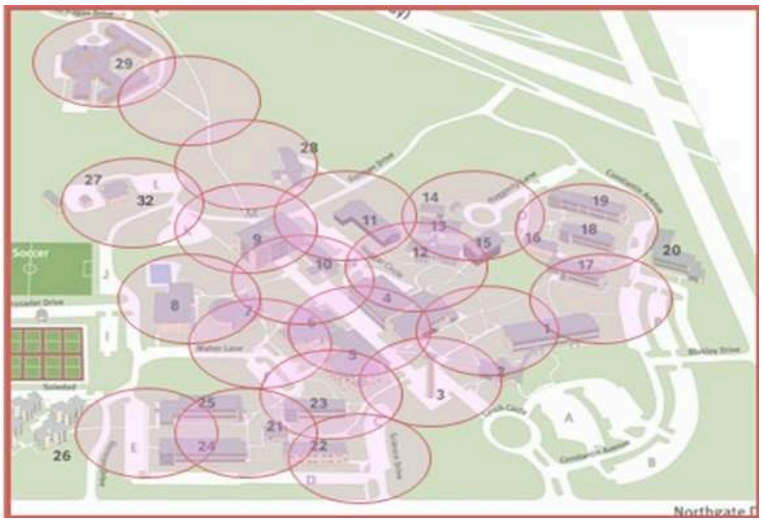


Figure 1.5: Wireless Mesh Network's Coverage at Taiwan Universtity

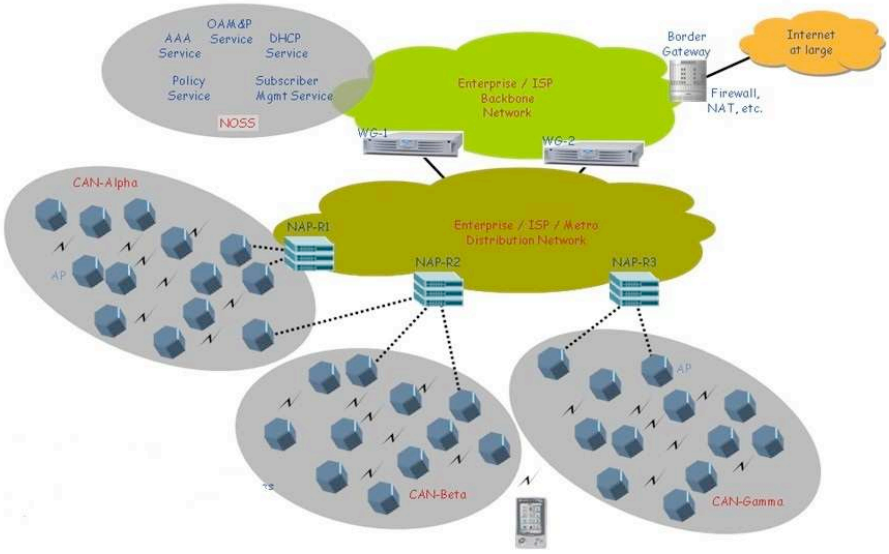


Figure 1.6: Nortel Wireless Mesh Network Architecture

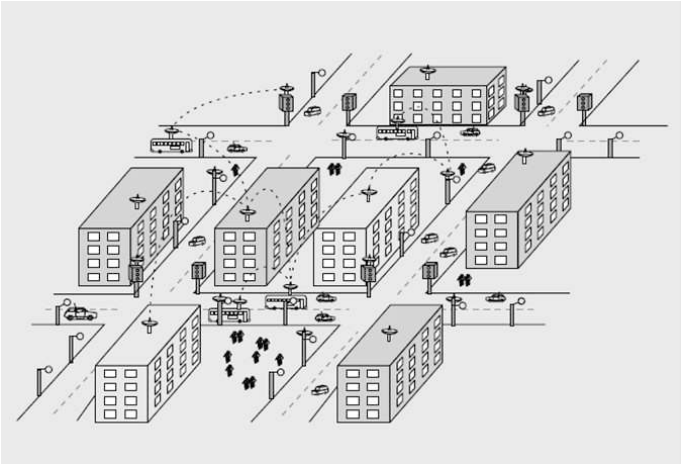


Figure 1.7: Intelligent Transportation System

technology provided by MeshNetworks Inc. The wireless mesh network allows anybody to display, at more than 40 locations throughout the city, real-time information on transportation services, such as where his/her bus is, its ultimate destination, and when it is scheduled to arrive. The same system is also expected to be used to address and alleviate transportation congestion problems, control pollution, and improve transportation safety and security.

1.2.2.2 Opportunistic Networks

Opportunistic networks are one of the most interesting evolutions of MANETs. In opportunistic networks, mobile nodes are enabled to communicate with each other even if a route connecting them never exists. Furthermore, nodes are not supposed to possess or acquire any knowledge about the network topology, which is necessary in traditional MANET routing protocols (Pelusi et al., 2006). In the network, any possible node can be opportunistically be used as next hop to bring the message closer to the destination.

Pocket Switched Networks in the Haggle Project The Haggle Project [2] is a 4-year project, started in January 2006, funded by the European Commission in the framework of the FET-SAC initiative [3]. It targets solutions for communication in autonomic/opportunistic networks. In this framework, researchers are studying the properties of Pocket Switched Networks (PSNs), i.e., opportunistic networks that can exploit any possible encountered device (e.g., cell phones and PDAs that users carry in their pockets) to forward messages.

Wildlife monitoring Opportunistic networks is very interesting for wildlife monitoring. It can investigate the behavior of wildlife by tracking the species. Researchers use opportunistic networks as a reliable, cost-effective, and not intrusive means to monitor large populations roaming in vast areas. Systems for wildlife monitoring generally include special tags with sensing capacity to be carried by the animals under study, and one or more base stations to collect the data from the tags and send them to the destination processing centre. A network protocol is also used to transmit the data from the tags to the base station(s). Base stations can be fixed or mobile, however, in

[2] The Haggle Project, http://www.haggleproject.org
[3] http://cordis.europa.eu/ist/fet/comms-sy.htm

both cases data collection from all the deployed tags is quite challenging. The realistic projects include ZebraNet [4] at Princeton University which is used for tracking zebras wearing special collars and SWIM (Shared Wireless Infostation Model) which is used to monitor whales.

Opportunistic networks for developing areas Opportunistic networks can provide intermittent Internet connectivity to rural and developing areas where they typically represent the only affordable way to help bridging the digital divide. One such example is the DakNet Project aimed at realizing a very low-cost asynchronous ICT infrastructure to provide connectivity to rural villages in India, where it is not cost-effective to deploy standard Internet access. Another interesting opportunistic application scenario has been investigated in the framework of the Saami Network Connectivity (SNC) project [5] aimed at providing network connectivity to the nomadic Saami population of the reindeer herders. In its initial stage, the SNC project has only focused on providing email, file transfer, and cached web services to the Saami people. Reindeer herd telemetry is also going to be provided to support the herding activity itself.

The opportunistic networks do not need the existence of a complete path between two nodes, i.e. the source and destination might nenver connected to the same network at the same time. However, there are a lot of applications that could tolerate this kind of delay, as presented before. The main focus of the networks is still on routing and forwading packets effectively.

1.2.2.3 Vehicular ad hoc Networks

VANETs use ad hoc communications for performing efficient driver assistance and car safety. The communications include data from the roadside and from other cars. VANET research aims to supply drivers with information regarding obstacles on the road and emergency events, mainly due to line-of-sight limitations and large processing delays. VANET can be used to communicate premonitions, notification of emergencies, and warnings about traffic conditions. It can be used for distributing information about road conditions and maintenance, weather forecasts, or other relevant data distribution requirements between vehicles. VANET enable the use of advanced driver assistance systems (ADAS) and vehicular-to-vehicular (V2V) communications, also called inter-vehicular communications (IVC), as well as communication with roadside infrastructure. VANET have an advantage compared to traditional MANET. They rarely have constraints related to the capacities of the devices.

California PATH California PATH [6] seeks to learn how traffic information can positively impact the environment, traffic safety and traffic congestion. It will synthesize data and research in the areas of traffic data collection, emissions- and fuel-consumption-based navigation and "smart engine" controls to turn an Audi vehicle into a working prototype of the ultimate traffic and fuel-smart car. The project incorporates data on traffic signals, road conditions, vehicle velocity, terrain grade and traffic congestion conditions, creating a composite of information from which smart engine controls can choose the safest, most fuel-efficient speeds and routes.

[4] The ZebraNet Wildlife Tracker http://www.princeton.edu/~mrm/zebranet.html
[5] Community Wireless Solutions. http://www.cuwireless.net/
[6] California PATH, http://www.path.berkeley.edu/

European Project CarTALK 2000 The European Project CarTALK 2000 [7] focuses on new driver assistance systems which are based upon inter-vehicle communication. The main objectives are the development of co-operative driver assistance systems and the development of a self-organizing ad-hoc radio network as a communication basis with the aim of preparing a future standard.

To achieve a suitable communication system, algorithms for radio ad-hoc networks with extremely high dynamic network topologies are developed and prototypes are tested in probe vehicles. Apart from the technological goals, CarTALK 2000 actively addresses market introduction strategies including cost/ benefit analyses and legal aspects, and eventually aims at the standardization to bring these systems to the European market.

LaRA, France LaRA [8] is to assist the drivers in order to improve safety, comfort and efficiency of road transport. The ultimate goal is to remove completely the driver from the control loop, at least in particular situations such as dedicated freeways and at low speed in urban situations.

1.2.2.4 Wireless Sensor Networks

WSN benefits from the advances in computing technology, which led to the production of small, wireless, battery powered, smart sensor nodes. These nodes are active devices with computing and communication capabilities that not only sample real world phenomena but also can filter, share, combine, and operate on the data they sense (Conti & Giordano, 2007).

Habitat and Environmental Monitoring for Scientific Applications The periodic information retrieval required by most of the habitat and environmental applications can be performed, in most of the cases, only by means of WSN. WSN enables regular observation of the environment without invading the environment of plants and animals and make possible a 24-hour monitoring.

- The Great Duck Island Habitat Monitoring project is a pilot application for monitoring migratory seabirds (Leach's Storm Petrel) on Great Duck Island, Maine. The WSN was used to monitor the microclimates in and around nesting burrows. Eventually, data is transferred via satellite to the database at the University of California at Berkeley. Intel and the University at Berkeley proposed WSN for creating a macroscope (sensor nodes strapped at different elevations on a redwood tree) to study the microclimate around redwoods.
- The Envisense GlacsWeb project uses WSN for monitoring the glacial environment to study sub glacial bed deformations by collecting measurement samples via pressure, temperature, and orientation sensors and delivering them to a base station located on the glacier surface, from which they are delivered to the sink.

Monitoring for Civilian Applications Forest fire detection, flood detection, and precision agriculture. Alarms, propagated by multihop through the WSN, enable a quick reaction before the fire becomes uncontrollable.

- Health monitoring. WSN can be used as part of a health monitoring system that can be worn by the patient. CodeBlue system developed at Harvard University exploits a WSN to

[7] Europe Cartalk Website, http://www.cartalk2000.net/
[8] LaRa sûr La Route Automatisee, http://www.lara.prd.fr

raise an alert when vital signs fall outside of the normal parameters. The system monitors heart rate, oxygen saturation, and EKG data and relays the data over a short-range wireless network to a set of devices, including ambulance-based terminals.

- Tracking applications. Instead of sensing environmental data, sensor nodes are deployed to sense the presence of persons and objects. In the simplest case, objects can be tracked by tagging them with a small sensor node. The sensor node is tracked as it moves through a field of sensor nodes that are deployed in the environment at known locations. The sensor nodes can be used as active tags that announce the presence of a device.

- Intelligent home environment. The smart home can communicate with the environment and people through the use of sensors and can act upon the environment through the use of actuators.

The pure ad hoc networks can be used for military applications and disaster recoveries, where the fixed infrastructures are undesirable. In section 1.4, some implementations and testbed based on ad hoc routing protocols will be further introduced.

1.3 Routing Protocol for ad hoc networks

Routing Protocols are used to discover and maintain routes between the source and destination nodes. For MANET, there are two main kinds of routing protocol: on-demand protocols (also called reactive protocols) and table-based protocols (also called proactive protocols).

For reactive protocols, nodes only compute routes when they are needed. Usually, caches are used to reduce the effort of route discovery.

For proactive protocols, each node maintains a routing table containing routes to all nodes in the network. Nodes must periodically exchange messages with routing information to keep routing tables up-to-date.

Furthermore, some hybrid protocols are proposed. This is because both proactive and reactive routing have specific advantages and disadvantages that make them suitable for certain kinds of scenarios. The hybrid methods try to take the advantages of those two and achieve better performance.

In the rest part of this section, some typical reactive and proactive protocols are firstly introduced. Then the protocols that try to fulfill the demands of QoS or security requirements are presented.

1.3.1 Reactive Routing Protocols

Because reactive routing only tries to find a route when necessary, it is believed that it is more scalable to dynamic, large networks. When a node needs a route to another node, it initiates a route discovery process to find a route. Generally, it consists of the following two main phases:

Route discovery It is the process of finding a route between two nodes, whether directly reachable within wireless transmission range or reachable through one or more intermediate network hops through other hosts.

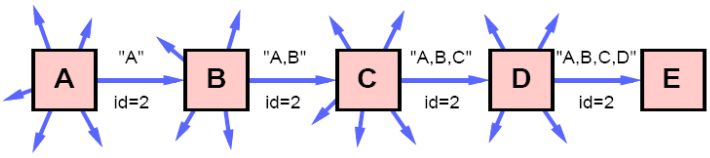


Figure 1.8: Route Discovery example: from Node A to Node E

Route maintenance It is the process of repairing a broken route or finding a new route in the presence of a route failure. Route maintenance procedure monitors the operation of the route and informs the sender of any routing errors.

Two of the most widely used reactive routing protocols: DSR and ADOV are introduced.

1.3.1.1 Dynamic Source Routing

Source routing is a routing technique in which the sender of a packet determines the complete sequence of nodes through which to forward the packet; the sender of a packet determines the complete sequence of nodes through which to forward the packet; the sender explicitly lists this route in the packet's header, identifying each forwarding "hop" by the address of the next node to the destination host.

The DSR (Johnson et al., 2007) is designed for use in the wireless environment of an ad hoc network. There is no periodic router advertisement in the protocol. Instead, when a host needs a route to another host, it dynamically determines one based on cached information and on the results of a route discovery protocol.

1.3.1.1.1 Route Discovery When node S originates a new packet destined to some other node D, it places in the header of the packet a source route giving the sequence of hops that the packet should follow on its way to D. Normally, S will obtain a suitable source route by searching its *Route Cache* of routes previously learned. But if no route is found in its cache, it will initiate the *Route Discovery* protocol to dynamically find a new route to D. In this case, we call S the initiator and D the target of *Route Discovery*.

For instance in Figure 1.8, node A is attempting to find a route to node E. A transmits a *ROUTE REQUEST* message as a single local broadcast packet, which contains a unique request id. When another node receives a *ROUTE REQUEST*, if it is the target of the *Route Discovery*, it returns a *ROUTE REPLY* message to the initiator. If not, this node appends its own address to the route record in the *ROUTE REQUEST* message and propagates it by transmitting it as a local broadcast packet, until the packet reaches its destination(node E in this example). When the packet gets to the destination, the node will returns a *ROUTE REPLY* message. The way the *ROUTE REPLY* message is returned can be starting a new *Route Discovery* or just using the reversed sequence of the nodes depends on the networks situation (unidirectional or bidirectional).

When initiating a *Route Discovery*, the sending node saves a copy of the original packet in a local buffer called the *Send Buffer*. The *Send Buffer* contains a copy of each packet that cannot be transmitted by this node because it does not yet have a source route to the packet's destination. While a packet remains in the *Send Buffer*, the node should occasionally initiate a new *Route Discovery*.

There are some additional features introduced in (Johnson et al., 2001) to improve the performance of Route Discovery:

- Caching overheard routing information;
- Replying to *ROUTE REQUEST*s using *Cached Route*s;
- Preventing *ROUTE REPLY* Storms;
- *ROUTE REQUEST* hop limits.

1.3.1.1.2 Route Maintenance When originating or forwarding a packet using a source route, each node transmitting the packet is responsible for confirming that the packet has been received by the next hop along the source route; the packet is retransmitted (up to a maximum number of attempts) until this confirmation of receipt is received.

If the packet is retransmitted by some hop the maximum number of times and no receipt confirmation is received, this node returns a *ROUTE ERROR* message to the original sender of the packet, identifying the link over which the packet could not be forwarded. The source node then remove this broken link from its cache, and use another route in the *Route Cache* (if has one), or starts a new *Route Discovery*.

There are also some additional features introduced in (Johnson et al., 2001) to improve the performance of Route Maintenance:

- Packet salvaging;
- Automatic route shortening;
- Increased spreading of *ROUTE ERROR* messages;
- Caching negative information.

For DSR, there are several advantages: First, unlike conventional routing protocols, it uses no periodic routing advertisement messages, thereby reducing network bandwidth overhead, particularly during periods when little or no significant host movement is taking place. Second, it does not require transmissions between hosts to work bidirectionally. And furthermore, it is able to adapt quickly to changes such as host movement, yet requires no routing protocol overhead during periods in which such changes do not occur.

However, The Route Maintenance protocol does not locally repair a broken link. The broken link is only communicated to the initiator. The DSR protocol is only efficient in MANETs with less than 200 nodes. Problems appear by fast moving of more hosts, so that the nodes can only move around in this case with a moderate speed. Flooding the network can cause collisions between the packets. Also there is always a small time delay at the beginning of a new connection because the initiator must first find the route to the target.

1.3.1.2 Ad hoc On-Demand Distance Vector Routing

The Ad Hoc On-Demand Distance Vector Routing Protocol is a reactive routing protocol based on DSDV (Perkins & Royer, 1999). AODV uses a broadcast route discovery mechanism, as is also used in the DSR algorithm. Instead of source routing, however, AODV uses hop-by-hop routing by maintaining routing table entries at intermediate nodes.

1.3.1.2.1 Route Discovery As in DSR, the route discovery process is initiated when a source needs a route to a destination and it does not have a route in its routing table. The source node floods the network with a RREQ packet specifying the destination for which the route is requested. When the destination receives the RREQ packet, the node generates a RREP packet, which is sent back to the source along the reverse path. Each node along the reverse path sets up a forward pointer to the node it received the RREP from. This sets up a forward path from the source to the destination.

1.3.1.2.2 Route Maintenance When a node detects a broken link while attempting to forward a packet to the next hop, it generates a RERR packet that is sent to all sources using the broken link. The RERR packet erases all routes using the link along the way. If a source receives a RERR packet and a route to the destination is still required, it initiates a new route discovery process.

The advantages of AODV are: loop free routing, optional multicast and reduced control overhead. But in the same time, delay is caused by route discovery process and bidirectional connections are needed.

1.3.2 Proactive Routing Protocols

In Proactive Routing, also called table-driven Routing, routes are calculated before one is needed. The protocol tries to keep routing information to all nodes every time up-to-date. The update of the tables can be periodically or driven by events. Two of the most popular proactive routing protocols, DSDV and OLSR are introduced in this subsection.

1.3.2.1 Dynamic Destination Sequenced Distance Vector Routing

Dynamic Destination Sequenced Distance Vector Routing (DSDV (Perkins & Bhagwat, 2001)) tries to keep the simplicity of Distributed Bellman-Ford routing and avoid the looping problem by tagging each routing table entry with a sequence number.

In DSDV, packets are transmitted between the nodes of the network using route tables stored at each node. Each route table, at each of the nodes, lists all available destinations and the number of hops to each. Each routing table entry is tagged with a sequence number that is originated by the destination node. To maintain the consistency of route tables in a dynamically varying topology, each node periodically transmits updates, doing so immediately when significant new information is available. These packets indicate which nodes are accessible from each node and the number of hops necessary to reach them, following traditional distance-vector routing algorithms.

Routing information is advertised by broadcasting or multicasting the packets that are transmitted periodically and incrementally as topological changes are detected—for instance, when nodes move within the network. Data is also kept about the length of time between the arrival of the first and the arrival of the best route for each particular destination. On the basis of this data, a decision may be made to delay advertising routes that are about to change, thus damping fluctuations of the route tables. The advertisement of possibly unstable routes is delayed to reduce the number of rebroadcasts of possible route entries that normally arrive with the same sequence number. If new routing information is received, a route with a more recent sequence number is used. If the new route has equal sequence number but better metric, this route is chosen.

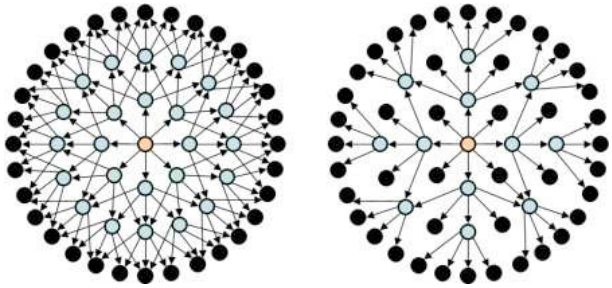


Figure 1.9: Flooding without MPR (left) and with MPR (right)

1.3.2.2 Optimized Link State Routing

In (Clausen & Jacquet, 2003), a proactive routing protocol, called Optimized Link State Routing for MANET is proposed. The protocol inherits the stability of the link state algorithm. Due to its proactive nature, it has an advantage of having the routes immediately available when needed. OLSR is an optimization of a pure link state protocol for MANET. First, it reduces the size of control packets: instead of all links, it declares only a subset of links with its neighbors who are its multipoint relay selectors. Secondly, it minimizes flooding of this control traffic by using only the selected nodes, called multipoint relays (MPR), to diffuse its message in the network. This technique significantly reduces the number of retransmissions in a flooding or broadcast procedure, as shown in Figure 1.9.

Only the MPRs of a node retransmit the packet from the node. For this purpose, each node maintains a set of its neighbors which are called MPR Selectors of the node.

The protocol functioning includes:

- Neighbor sensing. Each node periodically broadcast its HELLO messages about its neighbor and their link status. These control messages are only one-hop and permit each node to learn the knowledge of its neighbors up to two hops.
- Multipoint Relay selection. The MPR set is calculated in a manner to contain a subset of one hop neighbors which covers all the two hop neighbors.
- MPR information declaration. In order to build the intra-forwarding database needed for routing packets, each broadcasts specific control messages called Topology Control (TC) messages. A TC message is sent periodically by each node in the network to declare its MPR Selector set. Each node of the network maintains a topology table, in which it records the information about the topology of the network obtained from the TC messages.
- Routing table calculation. Each node maintains a routing table which allows it to route the packets for other destinations in the network. Because the routing table is based on the information contained in the neighbor table and the topology table, if any these tables is changed, the routing table is re-calculated to update the route information. The shortest-path algorithm is employed to get the route from source to the destination.

OLSRv2 OLSR version 2 (OLSRv2) has the same algorithm and ideas as OLSRv1. Being modular by design, OLSRv2 is made up from a number of generalized building blocks, standardized independently and applicable also for other MANET protocols. It is simpler and more efficient than OLSRv1, and has a more modular and extensible architecture:

- RFC 5148 - Jitter Considerations in Mobile Ad Hoc Networks (Clausen et al., 2008),
- RFC 5444 - Generalized MANET Packet / Message Format (Clausen et al., 2009c) and
- RFC 5497 - Representing Multi-Value Time in Mobile Ad Hoc Networks (MANETs) (Clausen & Dearlove, 2009)

are published as RFCs, with the remaining constituent parts,

- MANET Neighborhood Discovery Protocol (Clausen et al., 2009a) ,
- OLSRv2 (Clausen et al., 2009b))

being in the final phases of standardization.The multipath and its compatibility that we propose can also exist as additional modules in the OLSRv2 framework.

1.3.2.3 Reactive Routing vs. Proactive Routing

At the moment, there are two protocols mainly discussed on the IETF standard track:

- Dynamic MANET On-demand (DYMO) (Chakeres & Perkins, 2010) routing is successor to AODV. So it shares the reactive feature of its ancestor. Compared to AODV, DYMO has TLVs to allow for extensibility and allows a route to be improved by changing in response to superior routing information.
- OLSRv2, as already introduced, is the successor to OLSRv1.

They represents the reactive and proactive protocols respectively in the standardization of MANET routing protocols.

The performance evaluation in (Hamma et al., 2006) shows that the traffic load, the mobile node mobility and the network density all have impact on the performance of the routing protocol. The proactive protocol offers better performances for CBR (Constant Bit Rate) sources given that it guaranties lowest delay and jitter. But it consumes more bandwidth. And when the mobility is low, the reactive protocol performances low delay and overhead.

1.3.3 Hybrid Routing Protocols

As explained above, both a purely pro-active and purely reactive approaches to implement a routing protocol for a MANET have their disadvantages. The hybrid routing scheme, tries to take advantage of both of them. Normally, it use pro-active discovery within a node's local neighborhood, and using a reactive protocol for communication between these neighborhoods.

1.3.3.1 Zone Routing Protocol

In (Haas & Pearlman, 1997), a hybrid routing, called Zone Routing Protocol is proposed. ZRP divides the topology into zones and seek to utilize different routing protocols within and between the zones based on the weaknesses and strengths of these protocols. ZRP is totally modular, meaning that any routing protocol can be used within and between zones.

ZRP refers to the locally proactive routing component as the IntrA-zone Routing Protocol (IARP). The globally reactive routing component is named IntEr-zone Routing Protocol (IERP). IERP and IARP are not specific routing protocols.

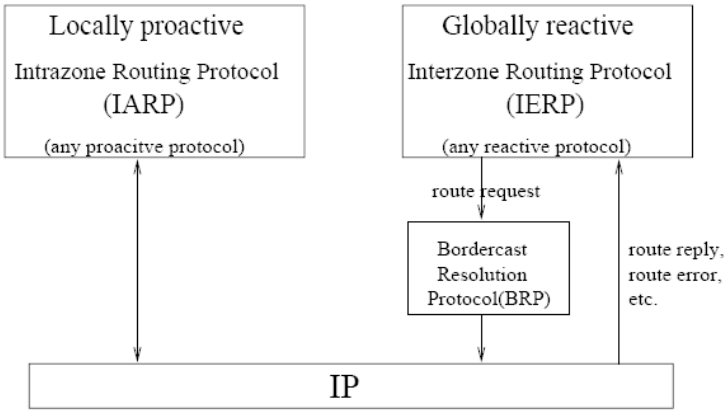


Figure 1.10: The different components of the ZRP

The fact that the topology of the local zone of each node is known can be used to reduce traffic when global route discovery is needed. Instead of broadcasting packets, ZRP uses a concept called bordercasting. Bordercasting utilizes the topology information provided by IARP to direct query request to the border of the zone. The bordercast packet delivery service is provided by the Bordercast Resolution Protocol (BRP).

The hybrid ZRP is actually more of a framework than a routing protocol, and it relies on well defined and robust routing protocols to be utilized in and between the zones. The latest ZRP Internet draft expired January 2003, but work is still said to be done by the authors and others. The need for solutions like ZRP might arise when the basic protocols are well tested and their limitations have been proven.

1.3.4 Multipath Routing Protocols

The routing protocols introduced above are uni-path routing. Based on those protocols, a lot of multi-path routing protocols are proposed. These protocols consist of finding multiple routes between a source and destination node by exploiting the density of the network. These multiple paths between source and destination node pairs can be used to compensate for the dynamic and unpredictable nature of ad hoc networks.

The multi-path routing could offer several benefits: load balancing, fault-tolerance, higher aggregate bandwidth, lower end-to-end delay, etc. The rest of this section is going to introduce several multi-path routing protocols that have been proposed.

1.3.4.1 Alternative Path Routing

In (Pearlman et al., 2002), Alternate Path Routing (APR) is proposed. APR had its origins in the traditional circuit-switched telephone networks, where it reduced call blocking by providing multiple network routes for the initial call-setup messaging.

APR's ability to provide a load balancing and enhanced survivability makes it an attractive technique for bandwidth-limited MANETs that are designed as packet-radio extensions to the wired Internet. However, the APR performance gains achieved on the wired Internet do not necessarily carry over to MANETs. In particular, the overlapping radio-coverage of neighboring nodes can

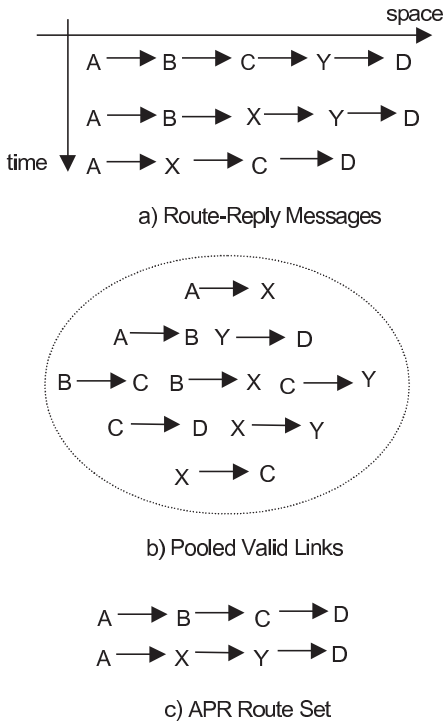


Figure 1.11: APR decomposes route replies into links states in order to reconstruct shorter, more diverse APR routes

result in strong interdependence between alternate routes which limits APR's benefits to particular MANET topologies and channel access techniques.

APR is based on the ZRP. A link state version of the ZRP's IntrA-zone Routing Protocol (IARP) can provide each node with up-to-date connectivity within its routing zone. The IntEr-zone Routing Protocol (IERP) provides routes, as needed, for destinations that lie outside of the routing zone. If a route exists, there will be a reply explicitly defining a unique path to the destination. The ZRP Route-Reply messages effectively provide a partial snapshot of the network topology. From the perspective of APR route-set construction, the most value can be obtained from the route-replies by decomposing the reported routes into a collection of links. The links returned from the route-query can be pooled with valid links from other route-queries, and the proactively-tracked links form within the source node's local routing zone, in order to maximize the effective routing information, as shown in Figure 1.11.

The potential benefits of APR make it appear to be an ideal candidate for the bandwidth limited and dynamic mobile ad-hoc networks. The investigation of APR in the MANET environment has revealed that APR can, in some circumstances, provide notable improvements to end-to-end capacity. Quite often, however, the network topology and channel characteristics limit what APR is able to achieve.

1.3.4.2 Ad hoc On-demand Multipath Distance Vector Routing

AOMDV (Marina & Das, 2006) is an on-demand, multipath distance vector routing protocol for mobile ad hoc networks. It extends the AODV protocol to compute multiple disjoint loop-free paths in a route discovery.

AOMDV shares several characteristics with AODV. It is based on the distance vector concept and uses hop-by-hop routing approach. Moreover, AOMDV also finds routes on demand using a route discovery procedure. The main difference lies in the number of routes found in each route discovery. In AOMDV, *RREQ* propagation from the source towards the destination establishes multiple reverse paths both at intermediate nodes as well as the destination. Multiple *RREP*s traverse these reverse paths back to form multiple forward paths to the destination at the source and intermediate nodes. AOMDV also provides intermediate nodes with alternate paths as they are found to be useful in reducing route discovery frequency.

The core of the AOMDV protocol lies in ensuring that multiple paths discovered are loop-free and disjoint, and in efficiently finding such paths using a flood-based route discovery. AOMDV route update rules, applied locally at each node, play a key role in maintaining loop-freedom and disjointness properties.

AOMDV relies as much as possible on the routing information already available in the underlying AODV protocol, thereby limiting the overhead incurred in discovering multiple paths. In particular, it does not employ any special control packets. In fact, extra RREPs and RERRs for multipath discovery and maintenance along with a few extra fields in routing control packets (i.e., *RREQ*s, *RREP*s, and *RERR*s) constitute the only additional overhead in AOMDV relative to AODV.

1.3.4.3 Split Multipath Routing

In (Lee & Gerla, 2002), the author proposes an on-demand routing scheme called Split Multipath Routing (SMR) that establishes and utilizes multiple routes of maximally disjoint paths. Providing multiple routes helps minimizing route recovery process and control message overhead. The protocol uses a per-packet allocation scheme to distribute data packets into multiple paths of active sessions. This traffic distribution efficiently utilizes available network resources and prevents nodes of the route from being congested in heavily loaded traffic situations.

Route Discovery The main goal of SMR is to build maximally disjoint multiple paths. It is necessary to construct maximally disjoint routes to prevent certain nodes from being congested, and to utilize the available network resources efficiently. The destination must know the entire path of all available routes so that it can select the routes. The source routing approach is used. In the normal source routing approach, the duplicate RREQ packets are discarded. This may cause the multi-paths are mostly overlapped.

In order to avoid this overlapped route problem, the author introduces a different packet forwarding approach. Instead of dropping every duplicate RREQs, intermediate nodes forward the duplicate packets that traversed through a different incoming link than the link from which the first RREQ is received, and whose hop count is not larger than that of the first received RREQ.

For Route Selection, the destination selects two routes that are maximally disjoint. One of the two routes is the shortest delay route; the path taken by the first RREQ the destination receives. After this process, the destination waits certain duration of time to receive more RREQs and learn all possible routes. It then selects the route that is maximally disjoint to the route that is already replied. If there is more than one route that are maximally disjoint with the first route, the one with the shortest hop distance is chosen. If there still remain multiple routes that meet the condition, the path that delivered the RREQ to the destination the quickest between them is selected. The destination then sends another RREP to the source via the second route selected.

Route Maintenance In SMR, when a node fails to deliver the packet to the next hop, a RERR message is send back to the source, and the source removes the correspond entry from its routing table. When the source is informed of a route disconnection and the session is still active, it may use one of the two policies in rediscovering routes:

- initiates the route recovery process when any route of the session is broken, or
- initiates the route recovery process only when both routes of the session are broken.

The first scheme reconstructs the routes more often and produces more control overhead than the second scheme, but the former provides multiple routes most of the time and is robust to route breaks. SMR outperforms DSR because multiple routes provide robustness to mobility.

1.3.4.4 OLSR-based multipath routing

The APR and AOMDV are mainly based on reactive topology discoveries. In the literature, there are also some routing mechanisms proposed based on proactive approach. In (Viennot & Jacquet, 2007), the authors proved that for OLSR, the broadcasting mechanism based on MPR is necessary and sufficient to provided multiple connectivities between the source and destination. So several protocols based on OLSR are introduced.

QOLSR In (Badis & Agha, 2004), the authors proposed a multipath QOLSR based on the measurement of the bandwidth and the delay. It is an enhancement of the OLSR routing protocol to support multiple-metric routing criteria. Each node calculates the delay and bandwidth information for each of its neighbors. Two approaches can be applied:

- The first approach uses one single metric in route decisions such as hop count or delay or available bandwidth.
- The second approach treats each metric individually.

The first approach is easy to implement by using Dijkstra algorithm. The second is not feasible due to the algorithm complexity. The problem of finding a path with a additive and m multiplicative metrics is NP-complete if $n + m \geqslant 2$ (Kuipers et al., 2002). The best path with all parameters at their optimal values may not exist if multiple metrics are included. Queuing delay is more dynamic, thus bandwidth is considered as more important. The strategy is to find a path with maximum bandwidth (a widest path), and when there is more than one widest path, the algorithm choose the one with the shortest delay, so called the *shortest-widest path algorithm.*

In the network, each node MPR generates a TC message including its MPR selectors' bandwidth and delay. Based on this information, the routing table is calculated. When the best path calculated by the *shortest-widest path algorithm* does not satisfy bandwidth and delay requirements, the multiple paths are calculated.

The proposed algorithm computes multiple loop-free and node-disjoint paths with a small correlation factor based on the delay and bandwidth metrics. The correlation factor is defined as the number of links connecting two disjointed paths. It is calculated to minimize interference between the multiple paths in order to achieve better QoS guarantees to applications and improve network resource utilization. However, the authors did not prove that the correlation factor can be correctly calculated. Indeed, OLSR nodes cannot have a global view of the network, but only links advertised in *HELLO* and *TC* messages (by default, the advertised link set in TC of the node is

limited to the MPR selector set (Clausen & Jacquet, 2003)). Moreover, this approach assumes a freshness of the measures (bandwidth, delay) which is difficult to obtain and maintain in practice.

Source Routing OLSR Kun & al. (Kun et al., 2005) propose another version of multipath OLSR using IP source routing. Based on Dijkstra algorithm, the node calculates multiple paths to the destination. The calculated paths are strictly node-disjoint. The path is inserted in the IP header of the packet before sending. Based on these multiple paths, the paper introduces an algorithm of load balancing to transmit data through the paths based on the congestion information of all the intermediate nodes on each path. The congestion information of one path is measured as the maximal size of the queue of the intermediate nodes (the queue size of a node is encapsulated in HELLO packets and advertised in TC messages). The algorithm of load balancing selects two paths to transmit data according to their congestion information, and balances the load on the selected paths.

In (Zhou et al., 2005), the authors also propose a similar algorithm to calculate node-disjoint multiple paths by removing used nodes from the topology information base. However, the pure source routing is problematic especially with topology changes which will be analyzed in the following section.

In both studies, strict node disjoint routes are not always necessary and the suppression of nodes in multiple calls of Dijkstra algorithm could not work for sparse networks. The node-disjoint multiple paths are not suitable for partition or fusion group of nodes that can temporarily imply a single link for connection. Furthermore, the backward compatibility is not considered, which might be very important for the deployment of a new protocol.

1.3.5 Quality of Service

Because the physical characteristic and the dynamic topology of the MANETs, it is a difficult task to offer the guaranteed quality of service (QoS), such as delay, jitter, bandwidth and delivery ratio.

The routing protocols introduced above mainly focused on finding feasible routes from the source to the destination, and do not take QoS into consideration. To support QoS, the essential problem is to find a route with sufficient available resources to meet the QoS constraints and possibly to incorporate optimizations, such as finding the lowest cost or most stable of the routes that meet the QoS constraints. Given these goals, the following are the basic design considerations for a QoS-aware routing protocol (Chen & Heinzelman, 2007).

- Resource estimation. The main goal is to obtain information about the available resources from lower layers which helps in performing call admission and QoS adaptation.
- Route discovery. Except for the reactive discovery and proactive discovery, the protocols that support QoS need to determine the combination of reduced latency and reduced overhead.
- Resource reservation. Because the bandwidth is shared by neighboring hosts, the resource reservation scheme needs to be applied to guarantee the QoS.
- Route maintenance. After the building of the routes, those routes still need to be well maintained because of the change of the topology.
- Route selection. The more reliable routes need to be selected to prevent the frequent route failures in the network.

- Choosing routes with the largest available bandwidth or minimum delay.
- Providing a call admission feature to deny route requests if insufficient bandwidth is available to support the request.
- Providing feedback to the application about available bandwidth resources or route delay estimation.

In the following, two QoS-aware routing protocols are introduced.

1.3.5.1 Ad hoc QoS On-demand Routing

Ad hoc QoS On-demand Routing (AQOR) (Xue & Ganz, 2003) is a resource reservation-based routing and signaling algorithm that provides end-to-end quality of service (QoS) support, in terms of bandwidth and end-to-end delay. The following features are integrated for QoS:

- on-demand route discovery between the source and destination
- signaling functions for resource reservation and maintenance
- hop-by-hop routing

When a route discovery is needed, the source hosts initiates a route request with bandwidth and delay requirements. The intermediate hosts check their available bandwidth and perform bandwidth admission hop by hop. If the bandwidth at the intermediate host is sufficient to support the request, an entry will be created in the routing table with an expiration time. When the data flow passes the nodes along the routes, bandwidth reservation is made.

And in AQOR, the following design decisions are made to reduce the connection maintenance overhead:

- AQOR facilitates QoS violation detection at the destination of the connection who can detect the flow's actual QoS, without the need of additional signaling;
- The routing adjustment overhead due to QoS violations, is reduced by employing destination-initiated recovery;
- The requirement for connection tear-down process, along the old path before route adjustment, is eliminated by the temporary reservation mechanism.

1.3.5.2 OLSR-based QoS Routing

The OLSR-based approach (Ge et al., 2003) does not modify the routing scheme of OLSR, but it chooses different criteria that incorporate bandwidth into consideration to select the multipoint relay (MPR) set so as to find a larger bandwidth route.

Bandwidth estimation is performed by taking advantage of the carrier-sense capability in the IEEE 802.11 MAC protocol and measuring the percentage of busy time to get the available bandwidth information. Because the decision of how each node selects its MPRs is essential to determining the optimal bandwidth route in the network, the MPR selection criteria is changed by taking the "good bandwidth" link into consideration. As many nodes as possible that have high bandwidth links connecting to the MPR selector must be included into the MPR sets.

The Maximum Bandwidth Spanning Tree Algorithm is proposed to calculate the routing table. By building the maximum bandwidth spanning tree, the node can find the optimal path in its known partial network topology.

Route maintenance and resource reservation are not considered in this protocol.

1.3.6 Security

The security is an important issue for network transmission. This subsection tries to give a summary of the related works of wireless network security, especially ad hoc network security.

The security in wireless networks has great difference from the wired networks because of the nature of the physical medium. The transmitting signal travels through the air in a wireless network, so any nodes that are in the transmission range of the transmitter can receive the signal. If the node is aware of the frequency and other parameters (modulation, coding, key, etc), it can potentially decode the signal without permission.

Except for the transmission medium, the mobility of the wireless network is another big challenge. Existing technologies often rely on the availability of traffic checkpoints (which most traffic goes through). With the check points, the security devices can inspect traffic of suspicious behavior and implement security policies and respond as needed. However, this is unachievable in ad hoc networks because such checkpoints do not exist with the mobility of the nodes. The traditional security solutions also depend on a few centrally located devices for security of the network, such as key distribution, etc. Such solutions are also not applicable for ad hoc networks. Normally a threshold scheme is used for key managment in cryptopgrahic systems (Shamir, 1979).

So different security mechanisms have to be used for ad hoc networks. One approach to designing security mechanisms for systems is to look at the threats that the system faces and the attacks possible given the vulnerabilities. The designed security mechanisms should then ensure that the system is secure in the light of these threats, attacks, and vulnerabilities (Anjum & Mouchtaris, 2007).

Threat is the means through which the ability or intent of an agent to adversely affect an automated system, facility or operation can be manifested. All methods or things used to exploit a weakness in a system, operation, or facility constitute threat agents.

Vulnerability is any hardware, firmware, or software flaw that leaves an information system open for potential exploitation. The exploitation can be of various types, such as gaining unauthorized access to information or disrupting critical processing. The vulnerabilities of ad hoc network can be in the routing, wireless links or other mechanisms such as auto-configuration.

An *attack* is an attempt to bypass the security controls on a computer. The attack may alter, release, or deny data. The success of an attack depends on the vulnerability of the system and the effectiveness of existing countermeasures.

In the rest of this subsection, two secure routing protocols are presented.

1.3.6.1 Secure Reactive Routing

SAODV (Zapata, 2005) is a secure version of AODV. It provides integrity, authentication and non-repudiation for routing data mainly by using new extension messages. In these extension messages there is a signature created by digesting the AODV packet using the private key of the original sender of the Routing message. The extension format is shown in Figure 1.12.

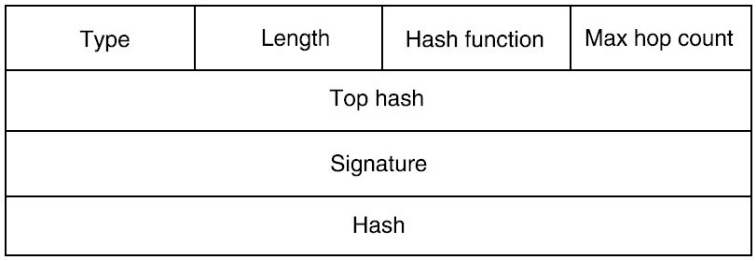


Figure 1.12: The SAODV message extension

The type field specifies the type of message. For example, a secure RREQ message may set the type field to 64, indicating an RREQ message, and 65 is for RREP message. The length field defines length of the type-specific data, not including the Type and Length fields of the extension in bytes. The hash function field specifies the hash function selected. The max hop count field is set by the originator of the RREQ to the TTL value. The top hash value is calculated by applying the hash function to a randomly selected number, called the seed, max hop count times. The signature field is the signature of the all the fields in the AODV packet that are before this field but the Hop Count field. This field has variable length, but it must be 32-bits aligned.

In SAODV, originators of routing messages (e.g. RREQ, RREP) digitally sign each message (excluding the hop-count field in the AODV message and the hash field in the SAODV extension), which ensures that nodes do not impersonate other nodes. Thus, digital signatures are used to protect the integrity of the nonmutable data in RREQ and RREP messages. A node that receives either of these messages verifies that the signature on the message is correct before taking any other action.

SAODV can also protect the mutable information such as the hop-count field in the RREQ and RREP messages by making use of the concept of hash chains (created by applying a hash function repeatedly to a seed number). This assures nodes receiving AODV messages that the hop-count values provided are accurate and have not been decremented by an adversary on the path.

In (Toubiana et al., 2008), the authors proposed a security framework called Adaptive Secured Multipath for Ad hoc Networks (ASMA) to protect the network from packet dropping attack. It combines multipath routing with the trust management, and evaluates trustworthy relationship based on localized trust model. The simulations were taken in NS2 with a subway scenario. The results revealed that compared to DSR, the ASMA can have better routing efficiency countering attacks.

1.3.6.2 Secure Link-State Routing Protocol

In (Adjih et al., 2003), the author proposed an extension to OLSR to make it secure against attacks. The main idea they propose is to use digital signatures for authenticating the OLSR routing messages. Such authentication may be done on a hop-by-hop basis or on an end-to-end basis.

To prevent malicious nodes from injecting incorrect information into the network, a signature is generated by the originator of each OLSR control message and transmitted with the control message. In addition, a timestamp is associated with each signature, in order to estimate a message freshness. Thus, upon receiving the control message, a node can determine if the message originates from a trusted node, and if the message integrity is preserved.

Signatures and timestamps are, inherently, separate entities from OLSR control traffic: while OLSR control traffic serves to acquire and distribute topological information, signatures serve to validate information origins and integrity. Thus, we introduce signatures as a separate type of OLSR control messages, encapsulated and transmitted.

Signatures are used by a receiving node to authenticate the corresponding OLSR control message: every control message without a matching corresponding signature is rejected. Depending on the properties of the signature method, different levels of authentication and resilience to attacks can be provided. For instance, the highest level of authentication may be provided by using individual asymmetric keys, as the messages advertised as generated from every non-compromised node are uniquely accepted when they indeed originate from this node. Weaker (but less complex or less computationally intensive) systems can be imagined, e.g. employing a shared secret-key system among trusted nodes.

In PhD thesis (Adnane, 2008), the author proposed a trust management strategy for OLSR. The trust-based reasoning allows each node to evaluate the behavior of other nodes, to detect misbehavior nodes, and determine the trust on other nodes. The solution requires only minor modifications on the OLSR and can be extended depending on the attack type and user needs.

1.4 Implementation and Testbed

In the literature, a high number of network protocols are only assessed through simulators due to implementation difficulties. This might induce two problems: firstly, with current simulation technology, it is not easy to simulate the exact real world scenario, especially the behavior in the physical layer model. There are already some works trying to have a better channel model. In (Pereira et al., 2006), the authors presents a study on the effect of indoor environment modeling precision on MIMO (Multiple Input Multiple Output) channel characterization. The environment modeling effects on MIMO channel characterization was investigated. In (Stepanov & Rothermel, 2008), the authors use intelligent ray tracing model to simulate a more realistic physical layer with a very high cost (3 days on a 50-node PC cluster to produce 120 GB of output data for just one scenario). The results show that there are differences between the commonly used physical layer model (free space or two ray ground) and the more realistic physical layer.

Secondly, some of the techniques and network parameters are easy to achieve in a simulator, but not in practice. For example, some of the protocols use extra information, such as delay and bandwidth as link metrics (Badis & Agha, 2004) to improve the performance of the network without mentioning how to obtain and maintain this kind of real-time information. This information might be easy to get in the simulator, but it is not very practical for a general usage.

Consequently, it is important not only to test the protocol with the simulator, but also to validate it in a real testbed to ensure that it is practical and feasible with current technology. In this section, some real testbed for ad hoc networks are introduced.

1.4.1 OLSRv2 Testbed

The team from Niigata University implemented OLSRv2 as nOLSR [9] and setup several testbeds to study the performance of MANET. It includes the research on the routing protocol, rate-switch

[9] nOLSRv2, Niigata OLSRv2 Implementation, Niigata University, Japan

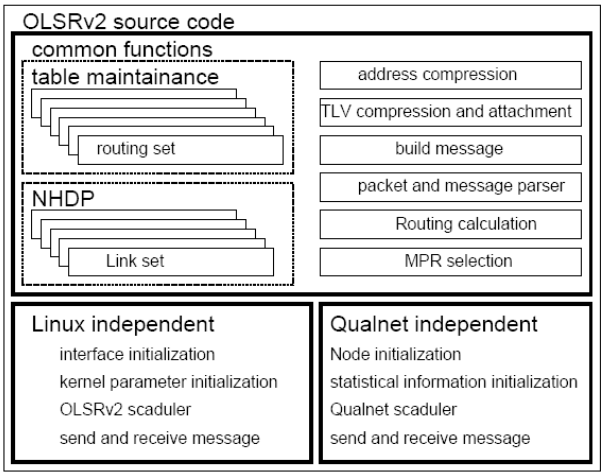


Figure 1.13: nOLSRv2 Implementation

control, address compression and link layer feedback (Owada et al., 2007).

The implemented OLSRv2 routing protocol uses C, and supports most functions included in the OLSRv2 specification, which includes attached network handling, multiple message aggregation to one packet, hop-by-hop TTL control mechanism, multiple interface handling, IPv4 & IPv6, address compression, etc. nOLSRv2 also has corresponding implementation in the QualNet network simulator [10]. In the implementation, the OLSRv2 functions are divided into three parts: the Linux socket I/O part, the Qualnet I/O part and the common OLSRv2 part (Figure 1.13). Dividing platform into dependent functions such as Linux and Qualnet specific functions from the OLSRv2 functionalities, OLSRv2 common functions can be used in both the simulator and Linux in addition to other platforms. This future is very interesting, because if we use the OLSRv2 implementation in Qualnet to do the simulation in the future, we can transplant it to Linux platform easily.

Several testbeds have also been set to evaluate the performance of the protocol. In November, 2004, a large-scale WMN testbed was built in Niigata University campus (1000×600m, Figure 1.14). The testbed has 55 fixed nodes. Forty nodes are placed inside the buildings and connected via coaxial cable to collinear omni-directional antennas on the roof or side of the buildings respectively. A node is composed of a 81(W)×38(H)×114.5(D) cm compact computer (IBM PowerPC 405 GPr 266 MHz, 8Mbyte FROM, 128 MByte SDRAM, SSD Linux 0,3 Kernel 2.4.26, Ethernet, PC card and CF card interfaces) and IEEE 802.11 wireless LAN card, inserted in the PC card interface. 1 GByte CF memory is also inserted in the CF card interface.

Another testbed is also set up for rural hill-village area in Yamakoshi (Takahashi et al., 2007). It is a small village where no commercial broadband service is available at present because of its geographical conditions. The aim of the project is to build a WMN testbed, called Yamakoshinet, and to perform experiments to study networking technologies for deploying economical and disaster-tolerant communication networks for the rural hill-village areas.

Overview of the Yamakoshinet is shown in Figure 1.15. Two settlements, Takezawa and Mushigame of the Yamakoshi Village are selected and covered by the WiFi-based layer 3 WMN using 8 and 12 pole top antennas and nodes, respectively. The two settlements are each located

[10] Qualnet simulator, Scalable Network Technologies, http://www.scalable-networks.com/.

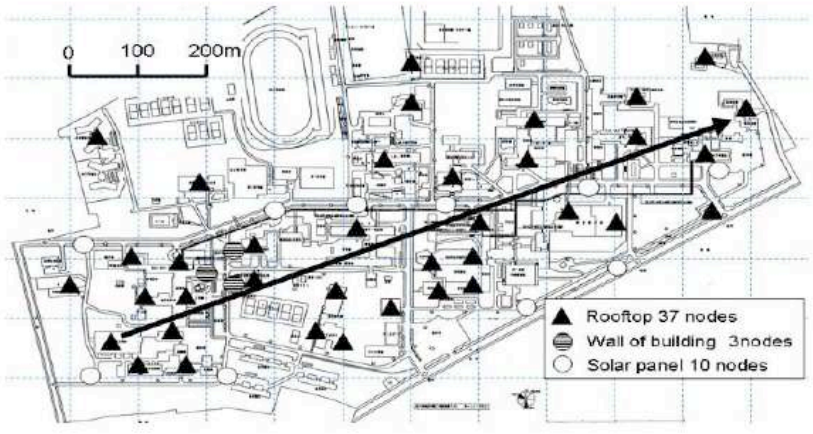


Figure 1.14: nOLSRv2 Campus Testbed

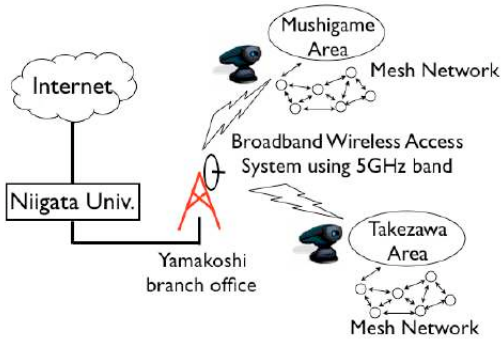


Figure 1.15: The Yamakoshinet in rural mountain area

about 1 mile from the branch office, and connected to the office with the broadband wireless access systems using 5 GHz band, respectively. The branch office is directly connected to Niigata University with the 100 Mbps commercial wide-area Ethernet service. In order to provide the wireless LAN spot service in the whole area, the newly developed mesh access points are installed on the power poles in each settlement.

Based on these testbeds, a set of experiments are performed, which include comparing OLSRv1 with OLSRv2, the effect of link layer feedback, rate-switch control, routing control, etc. From the experimental performance evaluation results, the authors conclude that when the LLN_THRESHOLD parameter is set appropriately, LLN can improve OLSRv2 performance.

1.4.2 Testbed based on OLSR daemon

For OLSR, one of the most sophisticated implementations is OLSR daemon (olsrd [11]) which is highly portable and scalable.

FunkFeuer [12] is a free, experimental network in Vienna. The goal is to build an unregulated network which has the potential to bridge the digital valley between the social layers and deliver the infrastructure and the knowledge for it. In Vienna, there are abound 300 members and 30

[11] Olsrd, an adhoc wireless mesh routing daemon, http://www.olsr.org/
[12] FunkFeuer.at, http://map.funkfeuer.at

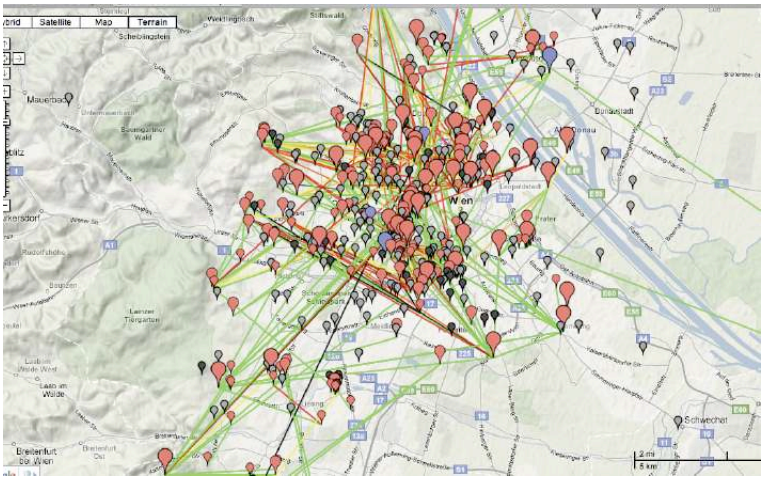


Figure 1.16: Funkfeuer mesh network in Vienna

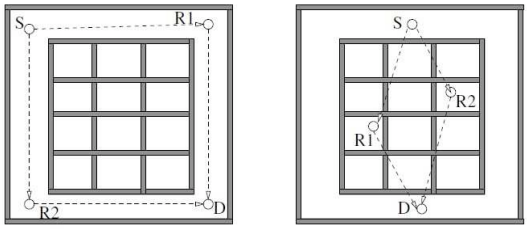


Figure 1.17: Experiment scenarios for video stream testbed with 4 nodes in the library building with topology line-of-sight (left) and behind the walls (right)

core members. It uses mixture of 5GHz bridges and 2.4 GHz mesh nodes. It covers about 30 km diameter with distributed ownership and use single uplink (1 GBit).

The corresponding project in Berlin, Germany is named Freifunk, which includes around 800 nodes in Berlin. Now the team is working on the CPU optimization, ETX metrics, and plan to find cheap multiradio solutions, better routing metric, better logging and metric aware algorithm.

1.4.3 Multipath Testbed

Compared with the single path routing, the multipath routing testbed is rare in the literature.

In (Mao et al., 2003), the author proposed a multipath testbed for video streaming. It consists of four IBM Thinkpad notebooks equipped with 802.11b cards. IBM High Rate Wireless LAN cards are used working in the DCF mode with a channel bandwidth of 11Mbps. The system is built on Microsoft Windows 2000. In the testbed, the static routing is used to build simple scenarios as shown in Figure 1.17.

In (Zhai et al., 2005), a testbed based on IBM laptops and Cisco wireless cards 1.18 is introduced.

The MSR routing protocol is implemented. It sends packets over multiple paths collected in Route Discovery phase. In routes selection, the disjoint paths are preferred in MSR because a more independent path can provide more resources between two nodes. Delay is used as the metric to distribute packets over multiple routes and a periodic probing mechanism is deployed to obtain the dynamic delay information for each path in use. A test scenario is shown in Figure 1.19.

Figure 1.18: Cisco Aironet 350 Series PC card

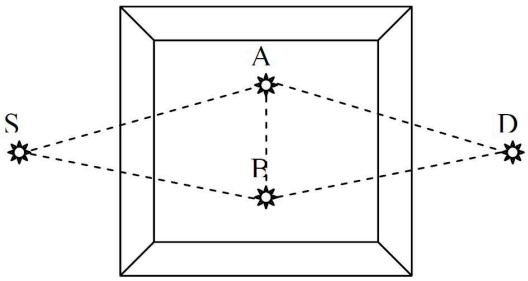


Figure 1.19: MSR Testbed

The OMF (cOntrol and Management Framework) [13] is a Testbed Control, Measurement and Management Framework. Based on this framework, a multi-path extension of OLSR is implemented based on Greedy Dominating Set algorithm (Tsai & Moors, 2008) for gateway placements in mesh networks.

The results obtained from these few testbeds show that the multipath routing protocol can provide shorter average route recovery time and higher throughput.

1.5 Conclusion

In this chapter, insights of wireless and ad hoc networks are provided. Because of the mobile and de-centralized nature of the ad hoc networks, they have great flexibility and applications. In the meantime, it also brings numerous challenges to the design of the network, such as spectrum allocation, media access, routing, etc.

Routing the data in ad hoc networks is one of the most important issues. At the moment, a lot of routing protocols have been proposed. They can be divided mainly into two categories: proactive routing and reactive routing. Based on those two types, some hybrid routing protocols and multipath routing protocols were also introduced. Those protocols can be better adapted to the ad hoc networks than the traditional routing protocols.

However, the routing mechanism still needs to be improved to fulfill the requirement for data integrity and delay restriction. Many challenging technical issues still need to be further studied

[13] OMF, the testbed control and management framework, http://omf.mytestbed.net

and demand our attention and investigation. In this thesis, we are trying to address the problem of providing reliable transmission by using multipath routing. It is an attractive technology for ad hoc networks, for both data transmission and multimedia applications. In the next chapter, we will first give an introduction on video service: its standard and interface with networks.

Chapter 2

Video Services and Transportation

Video transmission is nowadays widely used in different areas in our daily life, science and industry. The digital transmission of television signal via satellites is very common, and we can also chat with our colleagues face-to-face through video phone in our office or through mobile phone.

Video compression is applied to reduce the quantity of data used to represent digital video images. Normally, it exploits spatial and temporal redundancy contained in a sequence of images. Video compression is absolutely required by the video services because of the limited disk space, bandwidth, CPU power, etc. It is being used wherever digital video communications, storage, processing, acquisition and reproduction occur.

Video compression has two important benefits. First, it makes it possible to use digital video in transmission and storage environments that would not support uncompressed ('raw') video. For example, current Internet throughput rates are insufficient to handle uncompressed video in real time (even at low frame rates and/or small frame size). A Digital Versatile Disk (DVD) can only store a few seconds of raw video at television-quality resolution and frame rate and so DVD-Video storage would not be practical without video and audio compression. Second, video compression enables more efficient use of transmission and storage resources. If a high bitrate transmission channel is available, then it is a more attractive proposition to send high-resolution compressed video or multiple compressed video channels than to send a single, low-resolution, uncompressed stream. Even with constant advances in storage and transmission capacity, compression is likely to be an essential component of multimedia services for many years to come (Richardson, 2003).

The video compression considered here involves the bitrate reduction of a digital video signal carrying visual information. Traditional video-based compression focuses on eliminating the redundant elements of the signal like other information compression techniques. The redundancy removed can be the one located in temporal, spatial or frequency domains.

Figure 2.1 illustrates the spatial and temporal redundancy in two consecutive video frames. In the left side of the image, there is less variation in the grass field. So that area exists significant spatial redundancy. The frame rate of this video sequence is 30 frames/second (NTSC), so the interval between these two frames is 1/30 second. Most of these frame remains the same, which means there is also large temporal redundancy.

The compression system normally includes an encoder and a decoder. The encoder converts the source data into a compressed form (with a reduced number of bits) prior to transmission or storage and the decoder converts the compressed form back into a representation of the original video data. The encoder/decoder pair is often described as a CODEC (enCOder/ DECoder, Figure

(a) frame #92 (b) frame #93

Figure 2.1: Two consecutive frames from football sequence with large homogeneous region

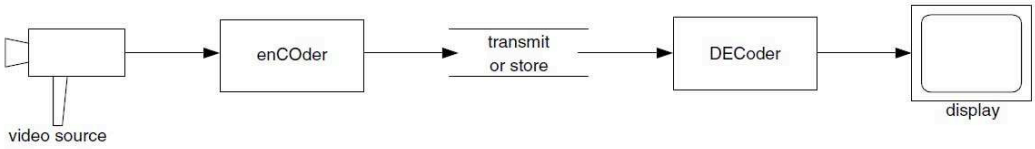


Figure 2.2: Video encoder and decoder

2.2) .

In this chapter, the video coding is firstly introduced. We have a short review on the previous coding standard, and then focus on the latest H.264 standard. Then we present the current video transmission interfaces in the literature, which is equally important for the application for video services.

2.1 Video Coding

2.1.1 Basic Concepts

2.1.1.1 Capture

A video sequence is spatially and temporally continuous. To represent a visual scene in digital form, it is necessary to sample the real scene (on a rectangular grid of the image) and temporally (as a series of still frame at regular intervals in time). Each spatio-temporal sample (picture element or pixel) (figure 2.3) is represented as a number or set of numbers that describes the brightness and color of the sample (Richardson, 2003).

The output of a CCD array is an analogue video signal in which a varying electrical signal that represents a video image. Sampling the signal at a specific time produces a sampled image or frame that has defined values at a set of sampling points. The common format of a sampled image is a rectangular grid.

For spatial sampling, it occurs at each of the intersection points on the grid and the sampled image may be reconstructed by representing each sample as a square picture element. The quality

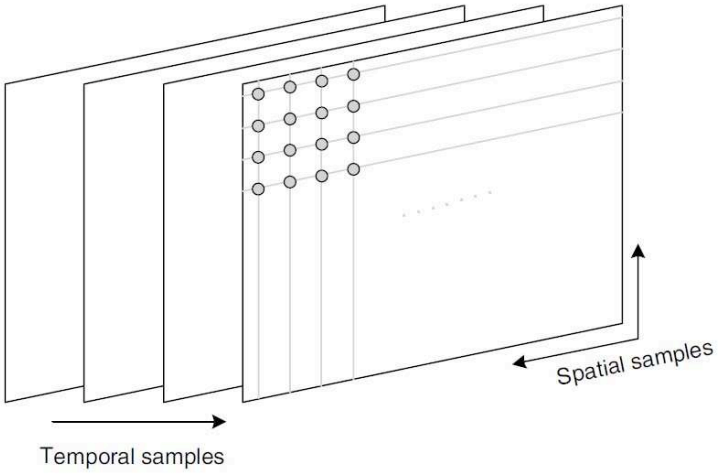


Figure 2.3: Sampling of a video sequence

(a) Image with low number of sampling points (b) Image with high number of sampling points

Figure 2.4: Image with different number of sampling points (a. 176×144, b. 352×288)

of the image is decided by the number of sampling points. Higher resolution can be achieved by increasing the sampling points (Figure 2.4).

Temporal sampling is related to a moving video. It is like taking a rectangular screenshot of the signal at periodic time intervals. Then these series of frame are played back to produce the appearance of motion. It is obvious that a higher temporal resolution (sampling rate) can provide smoother motion in the video scene with the cost of more samples to be captured and stored in the single time unit. The standard for television is 25 or 30 complete frames per second; very low bit-rate video communication maybe below 10 frames per second; and the smooth motion can be up to 60 frames per second.

2.1.1.2 Color Spaces

The digital video applications rely on the display of color video and so need a mechanism to capture and represent color information. A monochrome image requires just one number to indicate the brightness or luminance of each spatial sample. Color images, on the other hand, require at

Format	Luminance resolution
Sub-QCIF	128×96
Quarter CIF	176×144
CIF	352×288
4CIF	704×576
HDTV	1280×720 or more

Table 2.1: Video frame formats

least three numbers per pixel position to represent color accurately. A color space is an abstract mathematical model describing the way colors can be represented as tuples of numbers, typically as three or four values or color components.

The RGB (red-white-blue) is the most well-known color space. A color image sample is represented with three numbers that indicate the relative proportions of Red, Green and Blue (the three additive primary colors of light) in the RGB color space. Any color can be created by combining red, green and blue in varying proportions. The RGB color space is very suitable to capture and display of color images.

However, in the RGB color space the three colors are equally important and so all those three compnents need to be saved at the same resolution. But it is possible to represent a color image more efficiently by separating the luminance from the color information and representing luma with a higher resolution than color. That is why YCbCr color space is proposed. It is more efficient and used as a part of the color image in video and digital photography systems. Y is the luminance (luma) component and can be calculated as a weighted average of R, G and B:

$$Y = k_r R + k_g G + k_b B \tag{2.1}$$

where k are weighting factors.

The color information can be represented as color difference (chrominance or chroma) components, where each chrominance component is the difference between R, G or B and the luminance

$$\begin{aligned} C_b &= B - Y \\ C_r &= R - Y \\ C_g &= G - Y \end{aligned} \tag{2.2}$$

The complete description of a color image is given by Y (the luminance component) and two color differences C_b and C_r that represent the difference between the color intensity and the mean luminance of each image sample.

2.1.1.3 Video Formats

To facilitate the compression and transmission, it is common to capture or convert to a set of defined size. The Common Intermediate Format (CIF) is the basis for a popular set of formats listed in Table 2.1.

The choice of frame resolution depends on the application and available storage or transmission capacity. For example, 4CIF is appropriate for standard-definition television and DVD-video; CIF and QCIF are popular for videoconferencing applications; QCIF or SQCIF are appropriate for

mobile multimedia applications where the display resolution and the bitrate are limited.

2.1.2 Video Compression Standard

Two organizations dominate the video compression standardization. One is ITU-T VCEG (Video Coding Experts Group) and another is called ISO/IEC MPEG (Moving Picture Experts Group). In the last decades, there have been a lot of video compression standards been proposed. H.120 (ITU, 1994a) is the first digital video encoding standard. It was published by the CCITT (International Telegraph and Telephone Consultative Committee) in 1984, with a revision in 1988 that included contributions proposed by other organizations. As the first digital video encoding standard, H.120 video was not of good enough quality for practical use — it had very good spatial resolution, but very poor temporal quality. However, after this standard, the researchers began to be aware that it was necessary to encode using an average of less than 1 bit for each pixel in order to improve the video quality without introducing large bitstream. To achieve this, it is required to group a set of pixels coded together. This led to the following block-based codecs.

H.261 (ITU, 1994b) is regarded as the first practical video encoding standard. It is ratified in November 1988 and originally designed for transmission over ISDN (Integrated Services Digital Network) lines on which data rates are multiples of 64 kbit/s. Two different frame sizes are supported: CIF (352×288 luminance with 176×144 chrominance) and QCIF (176×144 with 88×72 chrominance) using a 4:2:0 sampling scheme. The coding algorithm was designed to be able to operate at different video bit rates (between 40 kbit/s and 2 Mbit/s). It also has a backward-compatible trick for sending still picture graphics with 704×576 luma resolution and 352×288 chroma resolution (which was added in a later revision in 1993). A lot of products support this standard. In fact, it has great influence that most of the subsequent international video coding standards are based closely on the H.261 design. Some typical structures which are still dominating today includes:

- 16×16 macroblock motion compensation,
- 8×8 DCT,
- scalar quantization,
- zig-zag scan,
- run-length,
- variable-length coding.

The Moving Picture Experts Group (MPEG) was established in 1988 to address the need for standard video and audio formats, and build on H.261 to get better quality by using more advanced coding methods. Nowadays, MPEG-1 has become the most widely used lossy audio/video format in the world. Its technology is used in huge number of products. Part 2 of the MPEG-1 standard is about the video and is defined in ISO/IEC-11172-2 (ISO, 1993). The design was heavily influenced by H.261. However, the MPEG-1 still has some weakness. For example, no standard support for interlaced video with poor compression when used for interlaced video, and only one standardized "profile" (Constrained Parameters Bitstream) which was not suitable for higher resolution video. In addition, it only supports one color space, 4:2:0 sub-sampling pattern.

MPEG-2, also known as H.262 (ITU, 2000), is the successor of MPEG-1. Part 1 and part 2 of MPEG-2 were developed in a joint collaborative team with ITU-T. It is now widely used as the format of digital television signals. It is also popular as the movie format and other programs that are distributed on DVD and similar discs. Part 2 of MPEG-2, shares the main feature with the previous MPEG-1 standard. But it also provides support for interlaced video which is used by analog broadcast TV systems. MPEG-2 video is not optimized for low bit-rates, especially less than 1 Mbit/s at standard definition resolutions. All standards-compliant MPEG-2 Video decoders are fully capable of playing back MPEG-1 Video streams conforming to the Constrained Parameters Bitstream syntax. It is also used in some HDTV transmission systems.

H.263 (ITU, 2005) was developed as an improvement based on H.261, MPEG-1 and MPEG-2. It is originally designed as a low-bitrate compressed format for videoconferencing. Its first version was completed in 1995 and provided a suitable replacement for H.261 at all bitrates. The version 2 of H.263 (also known as H.263+) has the entire technical content of the original version of the standard, but enhanced H.263 capabilities by adding several annexes which can substantially improve encoding efficiency and provide other capabilities.

MPEG 4 - part 2 (or MPEG - Visual)(ISO, 2004c) belongs to the MPEG-4 ISO/IEC standards. It is a discrete cosine transform compression standard, similar to previous standards such as MPEG-1 and MPEG-2. It has approximately 21 profiles, including profiles called *Simple*, *Advanced Simple*, *Main*, *Core*, *Advanced Coding Efficiency*, etc. The most commonly deployed profiles are *Advanced Simple* and *Simple*, which is a subset of *Advanced Simple*.

2.1.3 H.264/AVC and H.264/SVC

H.264/AVC (ITU, 2003) as the latest video coding standard of the ITU-T Video Coding Experts Group (VCEG) and the ISO/IEC MPEG, its main goal is to enhance compression performance and provide a "network-friendly" video representation addressing "conversational" (video telephony) and "non-conversational" (storage, broadcast or streaming) applications (Wiegand et al., 2003).

The new standard is designed for technical solutions in different application areas:

- Serial storage on optical and magnetic devices, Blueray, DVD, etc.
- Conversational services over Internet and intranet.
- Video-On-Demand (VOD) and multimedia streaming services over Internet and intranet.
- Broadcast over cable, satellite, cable modem, DSL, terrestrial, etc.
- Multimedia Messaging Services (MMS) over ISDN, DSL, Ethernet, LAN, wireless and mobile networks, etc.
- High-Definition Television (HDTV).

The new applications may be also deployed over future networks. To make the standard be able to handle different applications and networks, the H.264/AVC design covers a Video Coding Layer (VCL), which is designed to efficiently represent the video content, and a Network Abstraction Layer (NAL), which formats the VCL representation of the video and provides header information in a manner appropriate for conveyance by a variety of transport layers or storage media (Figure 2.5).

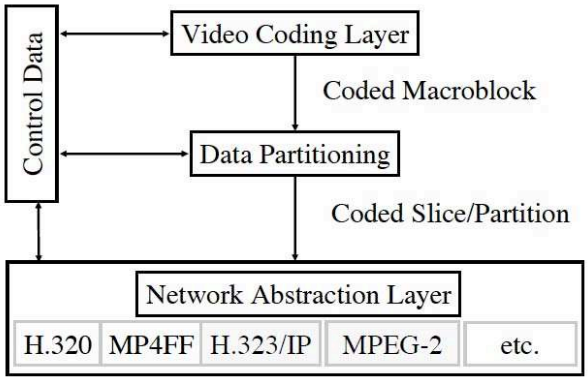


Figure 2.5: Structure of H.264/AVC video encoder (extracted from (Wiegand et al., 2003))

2.1.3.0.1 Network Abstraction Layer The NAL is designed in order to provide "network friendliness" to enable simple and effective customization of the use of the VCL for a broad variety of systems. The NAL facilitates the ability to map H.264/AVC VCL data to transport layers such as.

- RTP/IP for any kind of real-time wire-line and wireless Internet services.
- File formats, such as ISO MP4.
- H.32X for wired and wireless conversational services.
- MPEG-2 systems for broadcasting services.

One of the most important concepts for NAL is NAL unit. So the coded video data is organized into NAL units, each of which is effectively a packet that contains an integer number of bytes. The first byte of each NAL unit is a header byte that contains an indication of the type of data in the NAL unit, and the remaining bytes contain payload data of the type indicated by the header.

The payload data in the NAL unit is interleaved as necessary with emulation prevention bytes, which are bytes inserted with a specific value to prevent a particular pattern of data called a start code prefix from being accidentally generated inside the payload.

The NAL unit structure definition specifies a generic format for use in both packet-oriented and bitstream-oriented transport systems, and a series of NAL units generated by an encoder is referred to as a NAL unit stream.

2.1.3.0.2 Video Coding Layer As in all prior ITU-T and ISO/IEC JTC1 video standards since H.261, the VCL design follows the so-called block-based hybrid video coding approach. Each coded picture is represented in blockshaped units of associated luma and chroma samples called macroblocks. The basic source-coding algorithm is a hybrid of inter-picture prediction to exploit temporal statistical dependencies and transform coding of the prediction residual to exploit spatial statistical dependencies. There is no single coding element in the VCL that provides the majority of the significant improvement in compression efficiency in relation to prior video coding standards. It is rather a plurality of smaller improvements that add up to the significant gain.

2.1.3.1 H.264 Scalable Video Coding

H.264/SVC (Schwarz et al., 2007) is an extension of the H.264/AVC which is standardized by the Joint Video Team of the ITU-T VCEG and the ISO/IEC MPEG. It is a highly attractive solution to the problems set by the characteristics of modern video transmission systems. The term "scalability" refers to the removal of parts of the video bit stream in order to adapt it to the various needs or preferences of end users as well as to varying terminal capabilities or network conditions. SVC enables the transmission and decoding of partial bit streams to provide video services with lower temporal or spatial resolutions or reduced fidelity while retaining a reconstruction quality that is high relative to the rate of the partial bit streams. Hence, SVC provides functionalities such as graceful degradation in lossy transmission environments as well as bit rate, format, and power adaptation. These functionalities provide enhancements to transmission and storage applications.

The usual modes of scalability are temporal, spatial, and quality scalability. Spatial scalability and temporal scalability describe cases in which subsets of the bit stream represent the source content with a reduced picture size (spatial resolution) or frame rate (temporal resolution), respectively. With quality scalability, the substream provides the same spatio–temporal resolution as the complete bit stream, but with a lower fidelity — where fidelity is often informally referred to as signal-to-noise ratio (SNR). Quality scalability is also commonly referred to as fidelity or SNR scalability.

As an extension of H.264/AVC, the SVC tries to provide following essential requirements.

- Similar coding efficiency compared to single-layer coding—for each subset of the scalable bit stream.
- Little increase in decoding complexity compared to single layer decoding that scales with the decoded spatio–temporal resolution and bit rate.
- Support of temporal, spatial, and quality scalability.
- Support of a backward compatible base layer (H.264/AVC in this case).
- Support of simple bit stream adaptations after encoding.

As H.264/AVC, SVC also includes VCL and NAL layer. But there are also very important differences, especially in the VCL:

- The possibility to employ hierarchical prediction structures for providing temporal scalability with several layers while improving the coding efficiency and increasing the effectiveness of quality and spatial scalable coding.
- New methods for inter-layer prediction of motion and residual improving the coding efficiency of spatial scalable and quality scalable coding.
- The concept of key pictures for efficiently controlling the drift for packet-based quality scalable coding with hierarchical prediction structures.
- Single motion compensation loop decoding for spatial and quality scalable coding providing a decoder complexity close to that of single-layer coding.
- The support of a modified decoding process that allows a lossless and low-complexity rewriting of a quality scalable bit stream into a bit stream that conforms to a nonscalable H.264/AVC profile.

The characteristics of different kinds of scalability will be further studies in Chapter 8.

2.2 Video Transmission over Networks

A video codec normally needs to work with communication networks. Indeed, it can be regarded as a part of a communication system that involves coding video, audio and related information, combining the coded data and storing and/or transmitting the combined stream. This subsection mainly presents the current status of standardization of the video transmission over IP based networks, especially the H.264/SVC video transmission.

2.2.1 Video Transport Interface

The H.264/SVC is backward compatible with H.264/AVC. So SVC retains H.264/AVC's Network Abstraction Layer (NAL) concept and key properties. NAL units form the basic structure of a SVC bit stream. The parameter set concept is still used to convey most important information that pertains to more than one NAL unit.

A number of requirements for the system layer integration of SVC are listed below (Wang et al., 2007).

- In order to resolve which scalable layers are to be transmitted, there must be a data structure indicating which scalable layers the server can provide.
- To extract the desired layers from the entire scalable stream, basic data units need to be defined, and those data units need to include data-unit-to-layer mapping information. This information is also used by the receiver and in broadcasting, where some layers that are not needed are filtered out, e.g., when it is desirable to decode only the base layer that is H.264/AVC compatible. The desire for an H.264/AVC backward compatible base layer makes this a particularly difficult requirement, as the base layer syntax and semantics have been standardized in 2003 and must not be changed in a nonbackward-compatible way.
- To allow for seeking and fast forward/backward operations, random access points are needed. While the definition of random access points for a nonscalable bit stream is trivial, for scalable streams that is not the case.
- Layer switching points are needed for efficient scalable stream adaptation.
- As the scalable stream, when pre-encoded, is typically stored in a file container, The SVC design should allow simple operation for composing an SVC file from an SVC bit stream.

2.2.2 Video File Format and Packet

Within the ISO/IEC MPEG-4 standard, there are several parts that define file formats for the storage of time-based media (such as audio or video). Except from Part 12 itself, they are all based on, and derived from, the ISO Base Media File Format (ISO/IEC 14496-12) (ISO, 2004a), which is a structural, media-independent definition and which is also published as part of the JPEG2000 family of standards (as ISO/IEC 15444-12).

The file structure is object-oriented; a file can be decomposed into its constituent objects very simply, and the structure of the objects can be inferred directly from their type and position. The

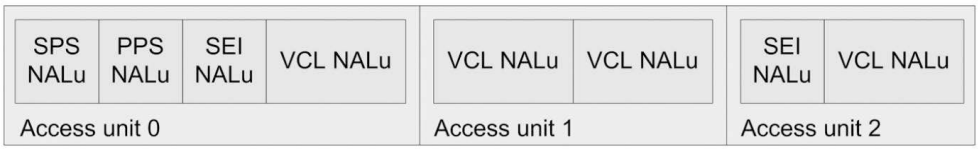


Figure 2.6: H.264/AVC elementary stream

types are 32-bit values and usually chosen to be four printable characters, for ease of inspection and editing.

The ISO Base Media File Format is designed to contain timed media information for a presentation in a flexible, extensible format, which facilitates interchange, management, editing, and presentation of the media. This presentation may be "local" to the system containing the presentation, or may be accessed via a network or other stream delivery mechanism.

The MP4 File Format (ISO/IEC 14496-14 (ISO, 2003)) is based on the ISO Base Media File Format. MP4 files are generally used to contain MPEG-4 media, including not only MPEG-4 audio and/or video, but also MPEG-4 presentations. When a complete or partial presentation is stored in an MP4 file, there are specific structures for that presentation.

MPEG-4 presentations are scenes, described by a scene language such as MPEG-4 BIFS (Binary Format for Scenes). Within those scenes, media objects can be placed; these media objects might be audio, video, or entire sub-scenes. Each object is described by an object descriptor. Within the object descriptor, the streams that make up that object are described. The entire scene is described by an initial object descriptor (IOD). This is stored in a special box within the movie box in MP4 files. The scene and the object descriptors it uses are stored in tracks—a scene track and an object descriptor track; for files that comprise a full MPEG-4 presentation, this IOD and these two tracks are required (Amon et al., 2007).

The AVC File Format (ISO/IEC 14496-15 (ISO, 2004b)) is based on the ISO Base Media File Format. Not truly a file format in its own, it describes how to store H.264/AVC streams in any file format based on the ISO Base Media File Format, including MP4, 3GPP, etc. An H.264/AVC stream is a sequence of access units, each divided into a number of NAL units. There are different NAL unit types defined, e.g., video coding layer (VCL) NAL units, Supplemental Enhancement Information (SEI) NAL units (carrying additional information (e.g., on the bitrate) not needed for the decoding process) or parameter set NAL units (Figure 2.6). In an AVC file, all NAL units to be processed at one instant in time form a file format sample. The size of each NAL unit (this length indication can be configured as 1, 2, or 4 bytes) is stored within the elementary stream in front of each NAL unit. The size of the entire sample is given in the sample size box.

The SVC File Format is a further specialization of the AVC File Format, and compatible with it. Like the AVC File Format, it defines how SVC streams are stored within any file format based on the ISO Base Media File Format (Amon et al., 2007).

Since the SVC base layer is compatible with H.264/AVC, the SVC File Format can also be used in an H.264/AVC-compatible fashion. However, full exercise of the scalability features of SVC encouraged the development of some SVC-specific structures to enable scalable operation. These extensions fall into three categories, differing in the level of detail they cover (and therefore also in the complexity of using them).

1. If there are some expected, normal subsets of the scalable stream that will often be extracted, it is possible to define tracks that contain simple instructions on how to form those streams.

By following the instructions, a file reader can construct a stream for a particular operating point (i.e., subset) of the scalable stream with very little parsing or structural understanding of the scalable stream itself. These are called extractor tracks.

2. The data in the stream can be grouped into tiers, which contain one or more scalability layers of the scalable stream. Each tier has a description, and all the data in the stream can be mapped to a specific tier. If decisions about scalability can be made on the basis of the tier descriptions, then these structures can be used to select the tiers of interest, and discover rapidly the data that is associated with those tiers. The descriptive data in this case is not timed; only the mapping from the coding data to the descriptions is timed. This technique uses sample groups.

3. Finally, the data in the scalable stream can have time-parallel data associated with it, providing exact information about the associated video coding data. In this case, the descriptive data itself is timed, and can vary on a time-basis. This technique uses a time-parallel metadata track.

Finally, of course, scalable operations can be performed, if needed, by parsing the SVC coding data itself.

2.2.3 Realtime Transport Protocol

RTP (Real-time Transport Protocol (Schulzrinne et al., 2003)) provides end-to-end network transport functions suitable for applications transmitting real-time data, such as audio, video or simulation data, over multicast or unicast network services. RTP does not address resource reservation and does not guarantee quality-of-service for real-time services. The data transport is augmented by a control protocol (RTCP) to allow monitoring of the data delivery in a manner scalable to large multicast networks, and to provide minimal control and identification functionality. RTP and RTCP are designed to be independent of the underlying transport and network layers. The protocol supports the use of RTP-level translators and mixers.

RTP itself does not provide any mechanism to ensure timely delivery or provide other quality-of-service guarantees, but relies on lower-layer services to do so. It does not guarantee delivery or prevent out-of-order delivery, nor does it assume that the underlying network is reliable and delivers packets in sequence. The sequence numbers included in RTP allow the receiver to reconstruct the sender's packet sequence, but sequence numbers might also be used to determine the proper location of a packet, for example in video decoding, without necessarily decoding packets in sequence.

While RTP is primarily designed to satisfy the needs of multi-participant multimedia conferences, it is not limited to that particular application. Storage of continuous data, interactive distributed simulation, active badge, and control and measurement applications may also find RTP applicable (Schulzrinne et al., 2003).

RFC 3984 (Wenger et al., 2005) describes an RTP Payload format for the ITU-T Recommendation H.264 video codec and the technically identical ISO/IEC International Standard 14496-10 video codec. The RTP payload format allows for packetization of one or more Network Abstraction Layer Units (NALUs), produced by an H.264 video encoder, in each RTP payload. The payload format has wide applicability, as it supports applications from simple low bit-rate conversational usage, to Internet video streaming with interleaved transmission, to high bit- rate video-on-demand.

The benefits of having aggregation packets and out-of-decoding- order transmission of NAL units are as follows. Firstly, aggregation of multiple coded pictures into the same RTP packet can reduce packet header overhead. A bit rate saving of 5 to 10 percent is typical when the video bit rate is not larger than 64 kbps. Secondly, when temporal scalability is supported, sending lower temporal layers earlier than other data can avoid rebuffering in mobile streaming, since after a handover, instead of rebuffering data, the player can play a lower frame rate . Thirdly, improved error resilience can be achieved by sending more important data earlier such that there is more time for the retransmission. This would also allow for simple cross-layer synchronization of NAL units in different SVC layers transmitted in different RTP sessions (Wenger et al., 2007).

2.3 Conclusion

Video is an important service provided by networks. Beginning from 1980s, there have been a lot of coding standard proposed. The latest standard is H.264/AVC, which is a block-oriented motion-compensation-based codec standard developed by the ITU-T VCEG together with the ISO/IEC MPEG. SVC is an extension of the H.264/AVC video compression standard, which aims to enable the encoding of a high-quality video bitstream that contains one or more subset bitstreams that can themselves be decoded with a complexity and reconstruction quality similar to that achieved using the existing H.264/MPEG-4 AVC design with the same quantity of data as in the subset bitstream.

Other than the video codec, another important issue is the transmission interface for networks. So the video transmission over networks is also introduced briefly in this section. We are specially interested in the transport interface for H.264/SVC video. The file format of H.264/SVC is compatible with H.264/AVC. The RTP payload format for H.264/SVC allows for packetization of one or more Network Abstraction Layer Units (NALUs) produced by an H.264 video encoder.

How to guarantee the quality of the video services over the network transmission is an interesting topic, especially for MANETs, in which the packet loss is frequent. So some Forward Error Correction (FEC) technics are proposed to improve the stability of the transmission. In the next chapter, the FEC coding will be presented for the error-prone environment.

Chapter 3

Forward Error Correction Coding Based on Discrete Radon Transform

Coding theory is used widely in computer science, mathematics, electrical engineering, etc. The basic goal is efficient and reliable communication in an uncooperative environment. There are two essential aspects of coding theory: removal of the redundancy from the original data; correction of errors in the transmitted data. Those two aspects correspond to *source coding* and *channel coding* respectively:

- *Source Coding* attempts to compress the data from a source in order to transmit the data more efficiently. The video coding mentioned in section 2.1.3 is also a kind of source coding.

- *Channel Coding* can be also called error correction/detection. It is used to protect data sent over the loss channel during the transmission.

In this chapter, we mainly focus on the channel coding, and specially erasure channel coding.

In the communication, information travels from a source to a destination through a channel. We can choose how the information is constructed at the source and handled at the destination. But the behavior of the channel is normally out of our control. The uncertainties of the channel might damage or distort the information that travels through. The channel coding usually protect data by adding redundancy to the original source. This conflicts with the efficiency of the transmission. So it is necessary to balance the two. The coding scheme should communicate a decent amount of information but also recover from errors effectively.

Figure 3.1 illustrates the communication channel model (Shannon, 1948). The original message has k tuples and is coded into codeword with n tuples by channel encoder.

The redundancy can be added by different methods, such as replication or partitioning. And there is an erasure code, which encode a message of k symbols into a coded word with n symbols ($n > k$) such that the original k symbols can be recovered from a subset of the n symbols. If with $m = n - k$ symbols, the decoder can recover the codeword when any m symbols are erased in n, then the code is said to be Maximum Distance Separable (MDS).

In the literature, Forward Error Correction (FEC) code are commonly used to produce redudant fragments. Reed-Solomon codes are classic block codes for erasure correction. They use a systematic way of building codes that could detect and correct multiple random symbol errors.

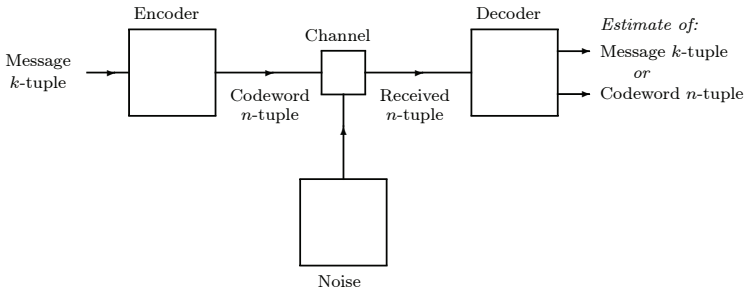


Figure 3.1: Communication channel model

They have ideal property that if k of the n transmitted symbols are received then the original k source symbols can be recovered.

However, the Reed-Solomon codes are practical only for small and rare k, n. The standard implementations of encoding and decoding have a cost of order $k(n-k)\log_2 n$ packet operations. In our study, we are more interested in geometrical Radon transforms, which provide linear complexity.

In the following of this chapter, we will introduce two kinds of discrete Radon Transform as Forward Error Correction code: Mojette Transform and Finite Radon Transform.

3.1 Mojette Transform

The *Mojette Transform* was proposed in 1995 by Jeanpierre Guédon (Guédon & Normand, 2005). The word "Mojette" is an old word in French, which means the class of beans. In old times, those white beans used are the standard tool for children to start computing additions and subtractions. Mojette is the name of the transform to remember that when only adds are invoked, the computations can be easily made.

This transform has many applications today in computer science and communication theory, namely image analysis, threshold cryptography and erasure channel coding. In the following, projections of an image (and pixels) even Mojette transform can be applied to any set of data.

3.1.1 Direct and Inverse Mojette Transform

3.1.1.1 Direct Transform

The Mojette transform is an exact and discrete Radon transform defined for specific "rational" projection angles. It easily describes an image by means of a finite set of 1-Demension projections. The rational projection angles, θ_i, are defined by a set of discrete vectors (p_i, q_i) as $\theta_i = tan(q_i/p_i)$. These vector must respect the condition that p_i and q_i are coprime and since tan is π-periodic, q_i is restricted to be positive except for the case $(p_i, q_i) = (1, 0)$. The transform domain of an image is a set of projections where each element (called a "bin" as in tomography) corresponds to the sum of the pixels centered on the line of projection. This is a linear transform defined for each projection angle by the operator (Guédon, 2009):

$$[M_\delta f](b, p, q) = proj_{p_i,q_i}(b) = \sum_{k=-\infty}^{+\infty} \sum_{l=-\infty}^{+\infty} f(k, l)\Delta(b + kq_i - lp_i) \tag{3.1}$$

Algorithm 3.1 Algorihtm Dirac-Mojette of image $f(k,l)$ with rectangular support $P \times Q$

INPUT: image $f(k,l)$, size $P \times Q$ of f support, set of projection angles $S_I = (p_i, q_i), 1 \leq i \leq I$
OUTPUT: projection values $proj(b, p_i, q_i)$
Set the $(0,0)$ position of $f(k,l)$ at the bottom of left corner;
for all projection index i of S_i **do**
  $B(i) \leftarrow (Q-1)|p_i| + (P-1)q_i + 1$;
  initialize vector proj of length $B(i)$;
  for all $pixel(k,l)$ **do**
    $proj(-q_i k + p_i l) \leftarrow proj(-q_i k + p_i l) + f(k,l)$;
  end for
end for

where (k,l) defines the location of an image pixel, b is the index of a bin and $\Delta(n)$ is the Kronecker delta function equal to 1 when $n = 0$ and zero otherwise. The equation $b = -kq + pl$ presents the line of projection, i.e. the set of projected pixels. Invertible projections can be obtained not only with addition but using any linear discrete operation; other practically useful operations include modulo 256 addition and bitwise XOR. The Mojette transform, $M_I f$, corresponds to the set of I projections as $M_I f = \{M_{p_i,q_i}, 1 \leq i \leq I\}$. The pixel is used for image processing. For more general purpose, it can be replaced by *ixel*, which can be any information element.

The main difference from the classical Radon transform is the sampling rate on each projection, which is no longer constant but depends on the chosen angle as $1/\sqrt{p_i^2 + q_i^2}$. The number of bins, $B(i)$, for each projection depends on the chosen direction vector (p_i, q_i), and for a $P \times Q$ image is found as

$$B(i) = (Q-1)|p_i| + (P-1)q_i + 1 \tag{3.2}$$

The algorithm for the computation of the 2D Mojette transform is shown in Algorithm 3.1.

The order of complexity is $O(PQ)$ for each projection computation. So for a set of I projections, the total Mojette complexity is $O(IPQ)$. An example of Mojette grid with four projections is shown in Figure 3.2.

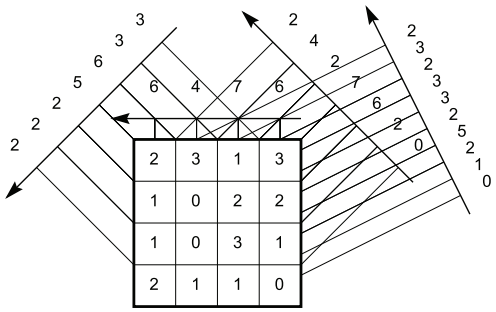


Figure 3.2: An example of Mojette grid with 4 projections for angles $(p,q) \in \{(0,1),(1,1),(-1,1),(2,1)\}$. (extracted from (Guédon, 2009))

Because the Mojette transform is also a discrete Radon transform, it inherits the properties from the Radon transform:

- Linearity, central slice theorem. The Mojette transform shares the linearity and CST as Radon transform. This means that the visual representation of the transform can also be

summarized into a matrix writing in the form of projection. In this case, image is a 1D vector of N pixels and projection is the vector of bins.

- Autocorrelation. It can be expressed by: "The Mojette projection of the autocorrelation of a 2D discrete image equals the autocorrelation of the Mojette projection of the image." The main interest is to perform the autocorrelation of the projection instead of the image when image analysis is required.

- Angular sampling. The Mojette transform can be simply viewed as a convenient simple tool to get a single projection (for instance for image analysis) in a given direction.

3.1.1.2 Inverse Transform

The inverse transform of Mojette can be regarded as the reconstruction of the original information. For the rectangular regions, the reconstructability can be described as Katz' criterion:

Given a set of pixels on a rectangular array ($P \times Q$) and a set S_I of I projection directions given by $S_I = \{(p_i, q_i), 1 \leq i \leq I\}$ restricting ($|q_i| > 0$), define P_I as $P_I = \sum_{i=0}^{I-1} |p_i|$ and Q_I as $Q_I = \sum_{i=0}^{I-1} q_i$, then ($P_I \geq P$) or ($Q_I \geq Q$) is equivalent to a unique image defined on the ($P \times Q$) array is reconstructible by the set of projections in the directions S_I.

Inverting the Mojette projections is equivalent to solving the linear system $\mathcal{A}X = B$ where B are the projection data, $\mathcal{A}$ is the projection matrix, and X is the pixel values we are trying to reconstruct. However, the binary matrix X is quite large (number of pixels $\times$ number of bins) and is rectangular and very spare due to the frame nature of the Mojette.

The reconstruction algorithm uses the transform geometry and the fact that the number of pixels that contribute to each bin is not constant. In some cases, a bin might correspond to a unique pixel in the image; this bin is then said to be reconstructible. The reconstruction solves for one pixel at a time and subtracts this value from the bins that include this pixel in each of the I projections. After this bin update, the set of projections is the exact transform of the pixels that are left to be reconstructed. The reconstruction propagates from the image corners (where there is only one pixel value in the bins) to the center. The following processes are performed iteratively:

1. finds a reconstructible bin, i.e. a bin projected from a single pixel,

2. "back projects" its value onto the original pixel,

3. updates the projections.

Two sub-problems remain unsolved with this algorithm. First, locate the bins in a projection which can be back projected (i.e. those bins for which only one pixel value remains unknown for its corresponding line of projection). Second, determine which one of the pixels, (k, l), in the line of projection, $b = q_i k - p_i l$, is yet to be reconstructed. To overcome these problems, two accounting images are projected with the same set of projection angles and reconstructed simultaneously with the unknown image. Further information about the Mojette direct/inverse transform can be found at (Guédon, 2009) in Chapter 3 and Chapter 4.

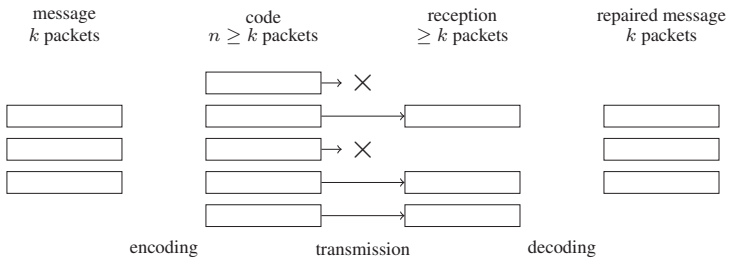


Figure 3.3: An error correcting code for erasure channels (extracted from (Guédon, 2009))

3.1.2 Redundancy of Mojette Transform

To measure the redundancy of the Mojette Transform, the following equation is defined:

$$\text{Red} = \frac{\text{nb}_{bins}}{\text{nb}_{pixels}} - 1 \tag{3.3}$$

When the redundancy is negative, the corresponding set is obviously not large enough to allow for reconstruction. A redundancy of zero occurs when the number of bins equals the number of pixels which is possible only for degenerated projections (with one bin per pixel) of the type $(1, P)$ or $(Q, 1)$ is computed from a rectangular shape $(P \times Q)$. The redundancy is positive when a reconstructible set of non degenerated projections is chosen. However, even a positive redundancy does not necessarily guarantee the reconstruction is possible. But it is obvious that we can produce as many projections as the level of protection requires (with $gcd(p, q) = 1$ and $q > 0$).

With the redundancy introduced, the Mojette transform can compensate possible losses of projection, i.e. over erasure channel. It is safe to assume that if a projection is received, then its elements contain no errors (otherwise we can use inter projection error detection/correction). Moreover, a systematic construction is possible which means in error correcting codes that the first k symbols are the exact copy of symbols. The $l = n-k$ remaining packets are produced by the code from message symbols. Mojette transform has $(1 + \epsilon)$ MDS property where ϵ represents the decoding overhead for a sufficient set of projections. The parameter ϵ is expressed as the ratio of the number of information elements I:

$$\epsilon = \frac{\#sufficientBinsNumber}{I} - 1 \tag{3.4}$$

Figure 3.3 illustrates the case of a message that was partitioned into three packets ($k = 3$); the erasure code provides a representation made of five packets ($n = 5$) and the reconstruction is still possible if up to two of the five packets are lost at the time of transmission (or storage).

3.1.3 Applications of Mojette Transform

Discrete Tomography

The Mojette operator M and the backprojector (its adjoint M^*) can be used fro reconstructing an image from an incomplete set of noisy projections (Serviers et al., 2005).

The discrete Radon operators obtained can be implemented in Mojette space. Different pixel interpolation models are available in that space. They all share the same discrete features that are translated into morphological mathematicsm, discrete geometry and matrix characteristics. The

presented exact discrete reconstruction needs a large amount of initial data but approximations (via angular and radial interpolations) are possible while maintaining the structure of the null space subject to discrete constraints. Classical algorithms include (Guédon, 2009):

- Filtered Back Projection (FBP) algorithm, which takes into account a finite (even if huge) number of projections belonging to a Farey series.
- Conjugate Gradient algorithm (CG), which consists in using the previous discrete version into an iterative algorithm with a reduction of the matrix size (by reducing the number of angles).
- Discrete exact Mojette reconstruction algorithm

Mojette Based Security

The Mojette based security was originally designed for watermarking purpose (Autrusseau, 2002). The digital watermarking techniques using the Mojette transform can be used to embed and detect watermarks. In this work, a digital watermarking technique was proposed for a medical imaging framework. Mojette phantoms, which were invisible within the projections, were embedded in Less Significant Bit-planes of medical images, by simple XOR operations. The proposed watermarking technique was blind (the original image was not needed during the watermark retrieval process). The key in this technique is shared between the physician and the patient, they each have a part of the directions used to generate the phantoms as well as their position in the image.

The Mojette transform can also be used to implement a shared secret application (Evenou et al., 2006) (distributed secret). For instance, it is expected that only two users over three to be able to reconstruct the data. A pseudo-random noise generator is used in order to create a noise having the same size as the original file, then, an Exclusive OR (XOR: $\otimes$) is applied on this noise and the original file, providing a noisy file ($F \otimes N$). Finally, the Mojette transform is applied on both the noisy file and the noise itself (N).

The Mojette transform has interesting properties in image encryption context. It has been demonstrated that applying the inverse Mojette transform on erroneous bins leads to a quick propagation of errors, providing encrypted images (Autrusseau et al., 2003). Such instability of the inverse Mojette transform may be positively exploited for encryption purpose. Furthermore, an encryption framework based on the Mojette transform may be significantly improved by taking benefit of all others properties of the Mojette transform for communication or storage application (lossless compression capabilities, Forward Error Correction,...).

Network Communication

In communication, Mojette Transform can be used as a FEC code or multiple description operator. In (Parrein et al., 2007), the authors propose to use Mojette transform for Unequal Error Protection (UEP) to provide an adapted protection to scalable data with an intrinsic hierarchy. This scheme is adapted to scalable sources (JPEG2000 image) and is based on the concept of "gray" packets, i.e. all data transport units that convey the source information have an equal weight at the decoding process. To achieve this, the Mojette transform is used to project a 2D image into 1D projections resulting in transport units.

The selected angles are of the from $(p_i, 1)$ for $i = 1, 2, ..., N$. In this case, M_s projections out of N projections are necessary and sufficient to reconstruct the stream s. Knowing M_s and the

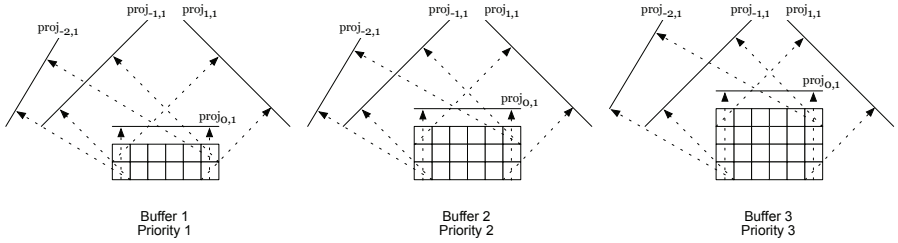


Figure 3.4: An example of a UEP scheme where buffer i is dedicated to priority i packets (for $i = 1, 2, 3$). Two, three and four projections (out of four) are necessary to reconstruct buffer 1, buffer 2 and buffer 3, respectively (extracted from (Parrein et al., 2007)).

sub-streams sizes, the 2D blocks can then be reconstructed for each stream s. These blocks can be seen as geometrical buffers whose capacities are function of the priority level of each substream. That is, for greater buffers heights, less protection is provided.

Figure 3.4 shows three geometrical buffers dedicated to three different packets priorities. The following set of four projections is transmitted: $\{proj_{-2,1}, proj_{-1,1}, proj_{0,1}, proj_{1,1}\}$ out of which a variable number of projections is necessary to reconstruct the different parts of the source data. Namely, two, three and four projections (out of four) are necessary to reconstruct buffer 1, buffer 2 and buffer 3, respectively.

Compared to Equal Error Protection (EEP), the experiment results show that UEP scheme based on Mojette transform can provide better image transmission quality with linear complexity.

3.2 Finite Radon Transform

The FRT discussed here is an adaptation of the Finite Radon Transform as presented by Matúš and Flusser in 1993. In 2001, Svalbe and van der Spek (Svalbe & van der Spek, 2001) showed it was possible to use a fully discrete technique to reconstruct high quality images from real x-ray sinograms. The real projection data was mapped into Mojette form and then the FRT was used to perform the digital image reconstruction. FRT has similar properties as Mojette Transform. It can be used in discrete tomographic reconstruction, image compression, and be extended for FEC code for data transmission. In this section, we will have a short introduction of FRT.

3.2.1 Direct and Inverse FRT

The FRT samples the image values at points corresponding to the Dirac image model (Kingston & Svalbe, 2006), for which the transform can be written as

$$R_m(t) = \sum_{y=0}^{p-1} I(x,y)\delta(t + my - x)_p \quad \text{for } 0 \leq m < p$$
$$R_p(t) = R_0^{\perp}(t) = \sum_{x=0}^{p-1} I(x,y)\delta(t - y)_p, \tag{3.5}$$

where $\delta(\eta)_p$ is 1 when $\eta \equiv 0 (\text{mod } p)$ and equals zero otherwise. Exactly how one should extend the point based image model is a fundamental difference between several discrete projection

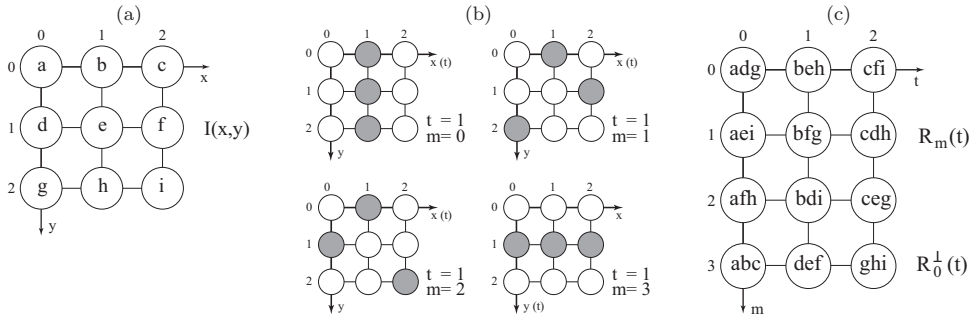


Figure 3.5: (a) 3×3 image, $I(x, y)$. (b) Examples of each of the four distinct directions for discrete lines of projection in the FRT over 3×3, $x \equiv my + t(\text{mod } p)$ for $0 \leq m < p$ and $y \equiv t(\text{mod } p)$ for $m = p$ (c) The resulting FRT of $I(x, y)$, $R_m(t)$. The *adg* in (0,0) stands for $a + d + g$ (extracted from (Kingston, 2005)).

methods; the Dirac model is the simplest representation to which the pixels based on more complex image models should be able to revert.

Figure 3.5(a) shows an example of a 3×3 discrete data set, with arbitrary pixel values labeled from a to i, as indicated. In the FRT, the set of discrete projections $R_m(t)$ of an image $I(x, y)$ is generated by repeated displacement of a vector $(m, 1)$, for $0 \leq m < p$, and the vector $(1, 0)$ for $R_p(t)$ (or $R_0^{\perp}(t)$). The origin of the vector displacements occurs at translate t, where $0 \leq t < p$. Figure 3.5(b) depicts the pixels summed by the projection vectors (shaded grey) for $t = 1$ over all $m = 0$, 1, 2 and 3. We take x (and t) to increase in unit pixel steps from left to right across the top row of I and take y to increase in unit steps from top to bottom. Choosing the data array size to be square and furthermore restricting the length of the array sides to be a prime number (p) of pixels makes each projection unique, because of the way it samples each and every element of the $p \times p$ image array once and only once. The resulting set of projection values is shown in Figure 3.5(c) (where t increases along the rows and m increases down each column). The perpendicular projection R_p (also denoted $R_0^{\perp}$) is taken as sums along the data rows, with translate t here increasing with y.

The FRT is exactly invertible. The values from each projection bin, Rm(t), can be back-projected using the same discrete directions $\tan^{-1}(1/m)$ and translate t as for the projection, i.e. $I(x, y) = \sum_{m=0}^{p-1} R_m(x - my)_p + R_p(y)$. The back-projected value at each image pixel is then Isum plus p times the original pixel value (as shown for the example in Figure 3.6), so all of the original image values (be they binary, integer or real) are exactly recoverable through

$$I(x, y) = \frac{1}{p}\left(\sum_{m=0}^{p-1} R_m(x - my)_p + R_p(y) - I_{sum}\right) \tag{3.6}$$

The back-projection in the FRT does not require any pre-filtering of the projection data, because each pixel in $I(x, y)$ is summed by the FRT with unit weight in each projection.

The main differences between Mojette and FRT are:

1. FRT requires a $p \times p$ prime size for the original data.
2. FRT produces an exact k-to-k mapping from original domain to coding domain.
3. FRT is a toric construction of the projection that delivers equal size coded unit (p or $p - 1$ size).

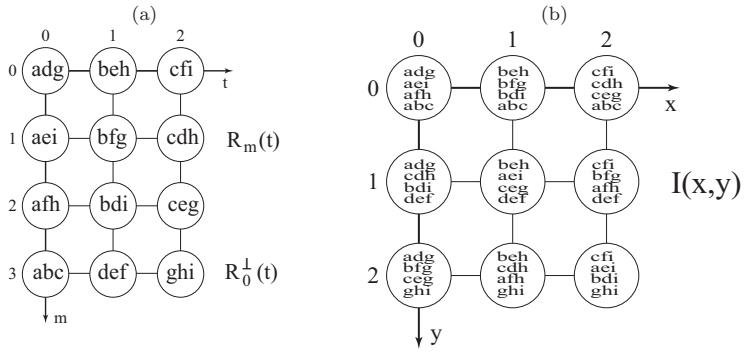


Figure 3.6: (a) The FRT projections of Figure 3.5a. (b) Back-projection of (a), is the first step to recover the original data. Subtracting $I_{sum} = abcdefghi$ from each pixel and dividing $p = 3$ completes the reconstruction (extracted from (Kingston, 2005)).

3.2.2 Applications of FRT

Like Mojette transform, the FRT has applications in many areas such as tomographic reconstruction, image compression, image representation, image convolution, watermarking, and erasure coding.

3.2.2.1 Discrete Tomographic Reconstruction

Reconstruction from digital projections requires no filtering or data interpolation. Mapping from digital FRT projections to recover the image is an exact process. Interpolation is, however, still required to obtain these discrete projections from real projection data. Periodic gaps occur in discrete lines, complicating the use of the FRT in reconstruction. For larger gaps, the approximation of continuous lines by discrete lines becomes poorer and interpolation becomes more difficult. In (Kingston & Svalbe, 2006), some methods are investigated to perform the interpolation in a search to find a reconstruction technique which makes minimal assumptions and uses little or no filtration or interpolation to produce the best reconstructed image.

3.2.2.2 Image Compression

In (Matús & Flusser, 1993), the authors propose to use FRT for image compression. Summing across the image produces relatively smooth projection data, hence FRT data is generally more compact than the image data. When an image is mapped to FRT projections, and the discrete line sum values are scaled to be the average value over the p pixels of the line, the majority of projections have very little variance about the mean value $\bar{I} = I_{sum}/p^2$.

Storing the projection values as the average value in the pixels of each discrete line can cause some small errors in reconstruction, (up to a few grey levels), if these stored values are rounded to the nearest integer grey level. The majority of these projections have very little variance, so lossless compression can be achieved since the (low variance) projection data can be stored as the deviation from $\bar{I}$ using fewer bits.

3.2.2.3 2D Image convolution

In (Matús & Flusser, 1993), the authors also showed the FRT can be used to reduce the dimensionality of many image processing operations. For example, 2-D image convolutions are reduced to

a series of 1-D convolutions over FRT projections. This is particularly useful when only applying the convolution to a subset of the projections.

3.2.2.4 Erasure Coding

In (Kingston & Svalbe, 2006), the authors showed that the FRT recover lost projections or missing data using *row solving algorithms* (Chandra et al., 2008). These algorithms make use of the rows of data that lies inside the redundant image area to recover the missing projections by back-projecting the data and then shifting, subtracting and integrating the rows of data.

In (Normand et al., 2010), our team proposed to use the cyclic prime Vandermonde structure to mathematically formalize the FRT projection and inversion operators and then to provide explicit expressions for the de-ghosting algorithms. It is used to recovery the data based on data structures called anti-images (or ghosts). Ghost images (matrix) contain signed pixel (ixel) values that, when projected, sum to zero for one or more projection directions.

3.3 Conclusion

In this chapter, the Forward Error Correction Coding is introduced. We focused on two kinds of Discrete Radon Transform: Mojette Transform and Finite Radon Transform. Compared to classical erasure codes (such as Reed-Solomon codes), the geometrical based Radon Transforms have lower complexity. This is a very attractive feature for mobile transmission because the power and resources of the mobile nodes are limited, and also very interesting for real time multimedia services because of the time restriction.

The Mojette transform and FRT are both discrete data projection methods that are exactly invertible and are computed using simple addition operations with almost MDS property. Adding defined level of redundancy into data and projection spaces enables the use of forward error correction to recover the original data packets when some of the projections are lost during the data transmission. This property makes it attractive for ad hoc networks, which are commonly error-prone networks. In the following of our study, it will be used for data protection during multipath video transmission.

Part II

Multipath Optimized Link State Routing

Introduction

In the previous part, we presented the bibliography concerning the ad hoc networks, especially the routing protocols for this kind of networks. In this part of the thesis, we expose our main contribution in the routing protocol: Multipath Optimized Link State Routing (MP-OLSR). It is a multipath extension of OLSR, which can be regarded as an hybrid routing scheme because it combines the proactive nature of topology sensing and reactive nature of route computation. We probe the multipath routing protocol from design to simulation, and finally the real implementation to validate the functions of the protocol.

The rest of this part is organized as follows: In Chapter 4, the SEREADMO project is introduced. It is a French national project which supported this research in multipath routing. The goal is to find a secure and reliable routing protocol for wireless ad hoc networks. We had a brief overview of the project: the participants, sub-projects and achievements. It gives the background of this thesis and the sub-projects related to the thesis which helps having an overall understanding of the research work.

Then we present the functionality of MP-OLSR in Chapter 5. The detailed specifications for the multipath routing are defined. MP-OLSR inherits the topology sensing mechanism from OLSR, which helps the nodes in the network to explore the network topology. The *Multipath Dijkstra Algorithm* is proposed to obtain multiple paths from the source to the destination. Source routing is employed to forward the packets. To avoid route failure and possible transient loops in the network, *Route Recovery* and *Loop Detection* are introduced to improve the performance of the network. The link metric based on queue length information is discussed as a possible replacement of the hop count metric. And in the end of the chapter, the problem of compatibility with OLSR is also illustrated.

The related simulation and performance analysis are presented in Chapter 6 to demonstrate the availability of different functions. The simulations based on well-known simulators such as NS2 and Qualnet are launched in different scenarios to study the behavior of the protocols with different configurations in various network environments. All the functions presented in Chapter 5 are implemented in the simulators to prove the validity of the proposed mechanisms.

The network simulators are powerful tools for network research, but they still have difference compared to the real world, from the modeling of physical layer to the implementation of routing protocols. To test the protocol in the real world, the implementation and testbed are presented in Chapter 7. We implemented MP-OLSR based on OLSR by using netbooks as mobile nodes. The testbed includes several nodes to form some simple scenarios for comparison between single path and multiple paths routing. In addition, a semi-realistic testbed, which is a mix of simulation tool and testbed, is also built as a compromise of the limited number of nodes in the real testbed.

Chapter 4

The SEREADMO Project

SEREADMO (Securité des Réseaux Ad Hoc & Mojette) is a French national project supported by RNRT (National Network of Research in Telecommunications). It lasted for 3 years from 2006 to 2009. Its goal is to find a secure and reliable routing protocol for wireless ad hoc networks. The three fundamental ideas of the project are:

- The routing protocol makes use of the Mojette Transform. The data distribution and the validation of data integrity are defined by Mojette Transform, which divides the information into redundant projections.
- The MIMO technology is used.
- The protocol guarantees the authentication and the confidentiality of the packets transmitted in the intermediate nodes and the destination.

Four famous French universities and companies, including

- University of Nantes and IRCCyN lab (UMR CNRS 6597)
- THALES Communications
- Keosys
- University of Poitiers and SIC lab (UMR CNRS 6172)

participated in the project.

The Image, Video and Communication (IVC) team of L'Institut de Recherche en Communications et Cybernétique de Nantes (IRCCyN) have 10 years' experience in source-channel coding of the Mojette Transform. As presented in Chapter 3, the Mojette Transform was firstly proposed by JP Guédon and N. Normand to make use of it for the secure transport on ATM network in 1996. From 1999, the team participated in the Etat Région contract on the Multimedia Mobile Communication (CER C2M) for the multimedia codec on the problem of shadow zone and the non-optimized reception in mobile environment. In this circumstance, a thesis supported by Région Pays de la Loire on ad hoc networks began in 2005. Also, the international conference IEEE Packet Video also took place at IRCCyN, Polytech Nantes in 2003. It was organized by JP Guédon and IVC team. 20% of the papers were on mobile video. The development of the Mojette Transform allows offering the coherent primitives for the watermarking, encryption and integrity of the data for all kinds of packets.

No.	Sub-project	Description	Participants
1	Routing and Mojette	Study and Specification of routing protocol and Mojette transform	IRCCyN, SIC, Thales
2	Routing Security	Study and specification of the procedure of security of routing protocol	IRCCyN, Thales
3	Adaption of MIMO for routing security	Study and specification of MIMO technology for routing protocol	IRCCyN, SIC, Thales
4	Security architecture equipment	Study and specifcation of the nomadic structure associated with the secure routing	IRCCyN, Thales
5	Usage and services for routing protocol	Study of the usage and services adapted by the capacity of routing protocol	IRCCyN, SIC, Thales, Keosys
6	Protocol experimentation	Experiment the data service in a urbain environment based on the testbed	IRCCyN, Thales, Keosys

Table 4.1: The sub-projects of SEREADMO and their participants

The THALES Communications is the premier French specialist in the domain of information security. The Sécurité des Systèmes d'Information (SSI) of THALES Communication has great competence and sophisticated products which are adapted for heterogeneous information system. The SSI has more than ten years' experience in wired networks (RNIS, IP, X25...) and mobile networks (GSM, Bluetooth, ...). In the recent years, the major realizations have been leading the new communication technologies.

The lab SIC focuses on the physic characteristic of the channel to optimize robustness of the wireless system in a real environment between 1 to 60 GHz frequency. The research activities of SIC Department come within the department of Sciences et Technologies de l'Information et de la Communication, and especially within images and wireless communications. They form a complete chain from fundamental researches to the development of industrial applications, in computer graphics and geometry, image processing and analysis of color and/or textured image sequences and wireless communication systems.

Keosys is leader in imaging and communication systems for nuclear medicine (NM, SPECT/CT & PET/CT) & radiology (CR, CT, MG & MRI). All Keosys equipments are multimodality and can therefore be installed at whatever nuclear medicine center. Behind the modality, Keosys provides state-of-the-art nuclear medicine workstations, Dicom servers and CD/DVD writers. The skills in telecommunications allow interconnecting several nuclear medical and radiology centers through a highly secured telemedicine network.

Table 4.1 presents the different sub-projects and their participants (RNRT, 2005).

My contribution is mainly in sub-project 1 and sub-project 6, which includes:

- Theory design of the multipath routing protocol;
- Implementation of the protocol in Qualnet and NS2 network simulator;
- Implementation of the forward error correction coding in simulator;
- Performance analysis and evaluation;
- Testbed setup and experiments.

The whole project was finished in July 2009. In accordance with the objects of the project, the following results were obtained:

- A multipath routing protocol. The MP-OLSR has been validated with the metrics of quality of service (delay, packet delivery ratio, jitter, etc.) in simulations and testbed. This will be presented in detail in this part.
- A data securing mechanism based on redundancy coding. The Mojette Transform is used to improve the packet delivery ratio in the mobile networks which have unreliable links.
- Exploitation of a realistic physic layer model and the evaluation with the routing protocol. On one hand, it uses a model to simulate the behavior of the channel in ad hoc networks. It is based on the rayon's theory and the homogeneous zone concept which consider the channel behavior in static and low mobility scenarios. On the other hand, it makes uses of the quality of the radio links to determine the route.

The related publications in international journals and conference are as follows:

International Journals

- J. Yi, A. Adnane, S. David, B. Parrein, Multipath Optimized Link State Routing for Mobile ad hoc Networks, Ad hoc Networks Journal, Elsevier, Volume 9, Issue 1, 28-47, Jan. 2011.
- C. Pereira, Y. Pousset, R. Vauzelle, P. Combeau. Sensitivity of the MIMO Channel Characterization to the Modeling of the Environment. IEEE Transactions on Antennas and Propagation, Volume 57, n°4, pages 1218-1227 - April 2009.

International Conferences

- S. Hamma, E. Cizeron, H. Issaka et J P Guédon, Performance evaluation of reactive and proactive routing protocol in IEEE 802.11 ad hoc network, SPIE ITCOM- next-generation and sensor networks, Boston, Oct. 2006.
- E. Cizeron, S. Hamma, A Multiple Description Coding strategy for Multi-Path in Mobile Ad hoc Networks, ICLAN 2007, Paris, 5-6-7 Decmber. 2007
- Jiazi Yi, Sylvain David, Hassiba Asmaa Adnane, Benoît Parrein, Xavier Lecourtier, Multipath OLSR: Simulation and Testbed. 5th OLSR Interop/Workshop, Vienna : Austria 2009.
- J. Yi, E. Cizeron, S. Hamma, B. Parrein, Simulation and Performance Analysis of MP-OLSR for Mobile Ad hoc Networks, IEEE WCNC, Las Vegas, April 2008.
- B. Parrein, F. Boulos, P. Le Callet, J P Guedon, Priority Image and Video Encoding Transmission Based on a Discrete Radon Transform, IEEE Packet Video 2007, Lausanne, novembre 2007.
- J. Yi, E. Cizeron, S. Hamma, B. Parrein, P. Lesage, Implementation of multipath and multiple description coding in OLSR, Proceedings of 4th OLSR Interop/workshop, Ottawa, Canada 2008.
- F. Boulos, W. Chei, B. Parrein, P. Le Callet, A new H.264/AVC error resilience model based on regions of interest, IEEE Packet Video 2009, Seattle, May 2009.

- Delahaye R., Pousset Y., Poussard A.-M., Vauzelle R., « Influence of the propagation model and the radio link physical layer quality criterion on ad hoc networks routing results", 4 pages, IEEE PIMRC'07 – Greece, october 2007
- Delahaye R., Poussard A.-M., Pousset Y., Vauzelle R., « Propagation Models and Physical Layer Quality Criteria Influence on Ad hoc Networks Routing », 5 pages, IEEE 7th ITST, Sophia Antipolis, France - June 2007
- Delahaye R., Pousset Y., Poussard A.M., Chatellier C. and Vauzelle R., "A realistic physical layer modeling of 802.11g ad hoc networks in outdoor environments with a computation time optimization",4 pages, WMSCI'07, Orlando, July 2007
- A.M. Poussard, W. Hamidouche, R. Vauzelle, Y.annis Pousset, B. Parrein, "Realistic SISO and MIMO Physical Layer implemented in two Routing Protocols for Vehicular Ad hoc Network", 5 pages, ITST 2009, Lille, October 2009
- Hamidouche W., Vauzelle R., Olivier C., Pousset Y., Perrine C., « Impact of realistic MIMO physical player on video Transmission over mobile ad hoc Network », 5 pages, 20th IEEE-PIRMMC, Tokyo (Japan) - September 2009

Patents

- French national patent (07 08837): Method for exchanging keys by indexation in a multi-path network (Procédé pour échanger des clés par indexation dans un réseau multi-chemin), Thales, France.
- French national patent (07 07180): Device and method for directing exchange flows for public or non sensitive values for creating common secret keys between several areas (Dispositif et procédé pour aiguiller des flux d'échange de valeurs publiques (ou non sensibles) permettant de créer des clés secrètes communes entre plusieurs zones), Thales, France.
- French national patent (07 06126): Method for distributing cryptograhic keys in a communication network. (Procédé de distribution de clés cryptographiques dans un réseau de communication), Thales, France.

Chapter 5

Specification of MP-OLSR

The MultiPath Optimized Link State Routing (MP-OLSR) can be regarded as a hybrid multipath routing protocol. It sends out HELLO messages and TC messages periodically to be aware of the network topology, just like OLSR. The difference is that MP-OLSR does not always keep a routing table to all the possible destinations. It only calculates the routes when there are data packets need to be sent out.

The core functioning of MP-OLSR has two main parts: *topology sensing* and *route computation*. The *topology sensing* makes the nodes get to the topology information of the network, which includes *link sensing*, *neighbor detection* and *topology discovery*. This part gets benefit from MPRs as well as OLSR. By sending the routing control messages proactively, the node could be aware of the topology of the network: its neighbors, 2-hop neighbors and other links. The routing computation uses the *Multipath Dijkstra Algorithm* to populate the multiple paths based on the information get from the topology sensing. The source route (the hops from the source to the destination) will be saved in the header of the data packets. The medium hops just read the packet head and forward the packet to the next hop.

The *topology sensing* and *route computation* make it possible to find multiple paths from source to destination. In the specification of the algorithm, the paths will be available and loop-free. However, in practice, the situation will be much more complicated due to the change of the topology and the instability of the wireless medium. So *route recovery* and *loop detection* are also proposed as auxiliary functionalities to improve the performance of the protocol. The *route recovery* can effectively reduce the packet loss, and the *loop detection* can be used to avoid potential loops in the network.

As presented in the state of the art, most of the multipath routing protocols proposed are based on single path version of an existing routing protocol: AODV and AOMDV, DSR and SMR, ASMA. However, the backward compatibility of those protocols with its single path version is not considered. In fact, given the heterogeneous nature of the wireless networks, it is important to make the new protocols backward-compatible. This can make the deployment of the new protocol much easier. So this part also discusses the compatibility between MP-OLSR and standardized OLSR by using IP source routing.

The specification of MP-OLSR has been discussed in the 4th OLSR workshop (2008, Ottawa, Canada), 5th OLSR workshop (2009, Vienna, Austria) and recently 78th IETF meeting (2010, Maastricht, Netherland) with the designers and developers of OLSR. They offered a lot of valuable propositions, from both theoritical and practical points of view. The proposed ideas have been

examined in the thesis.

In Section 5.1, 5.2 and 5.3, the core functionalities, including *topology sensing, route computation* and *packet forwarding* are first introduced. Then we present *route recovery* and *loop detection* in Section 5.4 and 5.5 which can be used to improve the packet delivery ratio and delay of the network. The compatibility between OLSR and MP-OLSR is studied in Section 5.7. The related simulation and performance evaluation are presented in Chapter 6 on page 93.

5.1 Topology Sensing

To get the topology information of the network, the nodes use the *topology sensing* which includes link sensing, neighbor detection and topology discovery, just like OLSR.

Link sensing populates the local link information base (*Link Set*). It is exclusively concerned with OLSR interface addresses and the ability to exchange packets between such OLSR interfaces. *Neighbor detection* populates the neighborhood information base (*Neighbor Set* and *2-hop Neighbor Set*) and concerns itself with nodes and their main addresses. Both link sensing and neighbor detection are based on the periodic exchange of HELLO messages. *Topology Discovery* generates the information base which concerns the nodes which are more than two hops away (*Topology Set*). It is based on the flooding of the TC messages (optimized by selecting the *MPR set*).

Through topology sensing, each node in the network can get sufficient information of the topology to enable routing. The link state protocol tries to keep the link information of the whole network as mentioned above. By default, the path quality is measured by the number of hops.

For the purpose of making the thesis self-contained, this part summarized the *Topology Sensing* functionality, which has the same idea as OLSRv1 and OLSRv2.

5.1.1 Information Repositories

5.1.1.1 Routing Control Message: HELLO and TC

The format of the routing control message in MP-OLSR inherits from OLSRv2. It is specified by the Generalized MANET Packet/Message Format (*packetbb*) (Clausen et al., 2009c), in which specifies the syntax of a packet format designed for carrying multiple routing protocol messages for information exchange between MANET (Mobile Ad hoc NETwork) routers.

The *packetbb* mainly specifies:

- A packet format and related packet header format, allowing zero or more messages to be contained within a single transmission.
- A message format containing attributes associated with the message or the originator of the message.
- A generalized type-length-value (TLV) format representing attributes. Each TLV can be associated with a packet, a message, or one or more addresses or address prefixes in a single address block.

Figure 5.1 summarized the generalized MANET packet format to be used in MP-OLSR. Based on this format, the packets in MANET can be easily extended to include extra information as TLV units. In section 5.6, we also use the TLV to carry queue length information in the packets. For detailed definition of each field, please refer to (Clausen et al., 2009c).

Figure 5.1: Generalized MANET Packet Format

HELLO Message The HELLO messages are generated by a router in MANET. It can be generated proactively at a regular interval or as a response to a change in the router itself. The purpose of the HELLO message is to broadcast the local node topology information to the one-hop neighbors to serve the task such as *link sensing* and neighbor detection.

The format of HELLO message follows the standard (Clausen et al., 2009c) and is defined in (Clausen et al., 2009a). A HELLO message must contain the following information:

- All of the network addresses in the *Local Interface Set* of the router on which the HELLO message is being generated.
- All of the relevant information in the corresponding *Link Set* and the *Neighbor Information Base*.
- Related REFRESH_INTERVAL used to define the lifetime of the information.

With the broadcast of HELLO messages to the node's one-hop neighbor, the node is able to get the topology information in two hops.

Topology Control Message TC messages are broadcasted by each node to the whole network to build the intra-forwarding database needed for routing packets. The format of TC message allows the standard (Clausen et al., 2009c) and is defined in (Clausen et al., 2009b).

A TC message is sent by a node in the network to declare a set of links, which must include at least the links to all nodes of its MPR Selector set, i.e., the neighbors which have selected the sender node as a MPR. TC messages are flooded to all nodes in the network and take advantage of MPRs. MPRs enable a better scalability in the distribution of topology information.

With the broadcast of TC messages to the whole network, the node is able to get the topology information that is more than two hops away.

5.1.1.2 Routing Information Base

Each node (router) in the MANET maintains an information base. The information base is constructed by the broadcast of HELLO and TC messages, and is used for routing decision. It mainly includes: *link set*, *2-hop set,* and *router topology set.*

Link Set *Link set* records information about current and recently lost links between this MANET interface and MANET interfaces of 1-hop neighbors. The Link Set consists of Link Tuples, each of which contains information about a single link. Link quality information, when used, is recorded in Link Tuples.

2-Hop set *2-Hop set* records the existence of symmetric links between symmetric 1-hop neighbors of this MANET interface and other routers (symmetric 2-hop neighbors). The *2-Hop Set* consists of 2-Hop Tuples, each of which records a network address of a symmetric 2-hop neighbor, and all network addresses of the corresponding symmetric 1-hop neighbor.

Router Topology Set The router's Topology Set records topology information about the links between routers in the MANET. It is obtained by the broadcast of TC messages, and mainly maintains the topology information that are more than 2 hops away.

Figure 5.2 shows the relationship between the network topology graph and all the related information bases. Based on these repositories, the operations of topology sensing can be perfomed.

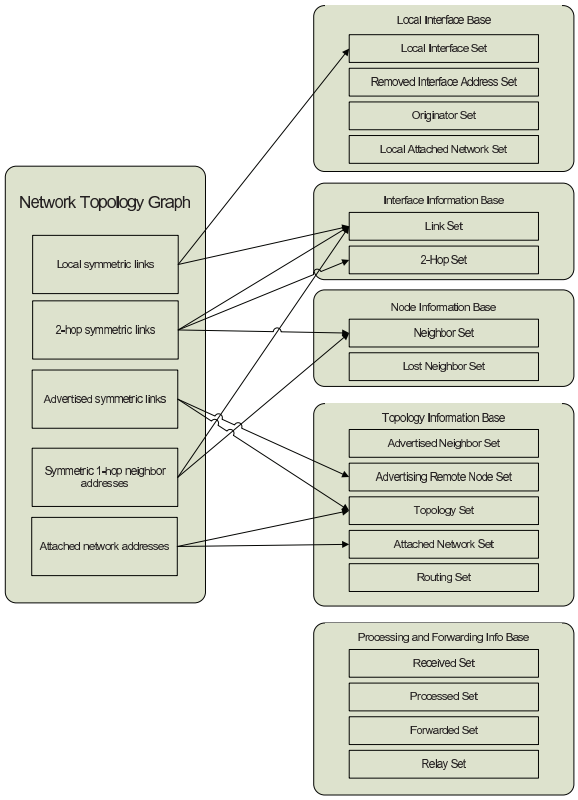


Figure 5.2: The MP-OLSR network topology graph

5.1.2 Link Sensing and Neighbor Detection

The *link sensing* and *neighbor detection* are based on the transmission of HELLO messages. Based on the received messages, the procedures called *link sensing* and *neighbor detection* are performed to build the *link set* and *2-hop set*.

On receiving a packet, the node examines the packet header and each of the message headers. If the message type is known to the node, the message is processed locally according to the specification for that message type. The message is also independently evaluated for forwarding. If parsing fails at any point the relevant entity (packet or message) must be silently discarded. Figure 5.3 shows the flowchart of packet processing and forwarding. So the incoming packets (TC or HELLO) can be appropriately distributed for next step of process.

When receiving a HELLO message, a router must update its *Link set* for the MANET interface on which the HELLO message is received, and update its *Neighbor Set*. The algorithm will first find all Neighbor Tuples (henceforth matching Neighbor Tuples) where the message's *N_ neighbor_ addr_ list* contains any network address which overlaps with any address in the node's Neighbor Address List. If there are no matching neighbor tuples, a new neighbor tuple will be created. If there are one or more matches, then related information has to be updated. Figure 5.4 shows the procedure to be performed for *Neighbor Set* updating. The 1-hop neighbors are then maintained properly based on the exchange of HELLO messages.

In addition to the *Neighbor Set*, the *2-Hop Set* also need to be updated. Figure 5.5 shows the procedure to be performed for *2-Hop Set* updating. So the information of the 2-hop nodes is saved in the *2-Hop Set*.

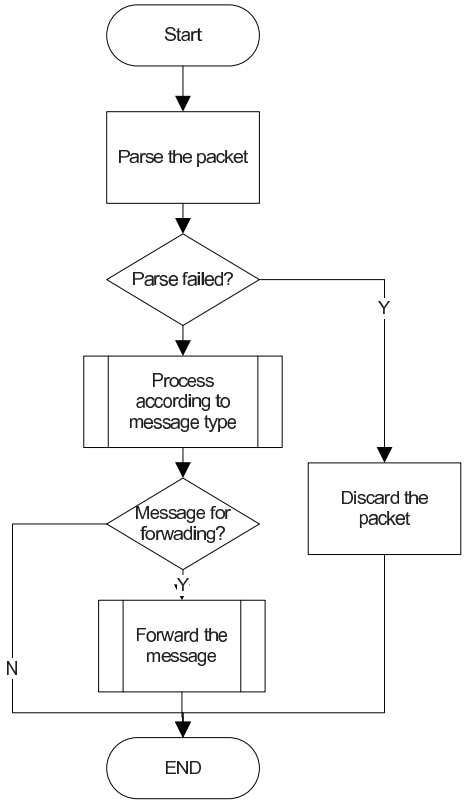


Figure 5.3: Packet processing and forwarding

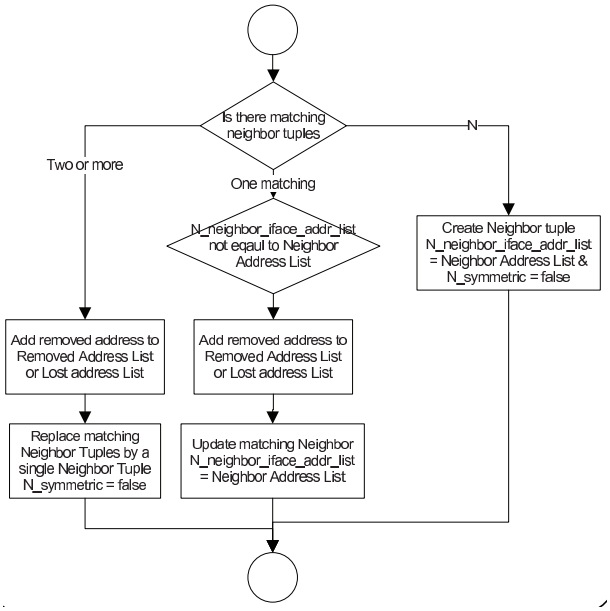


Figure 5.4: Update the *Neighbor Set*

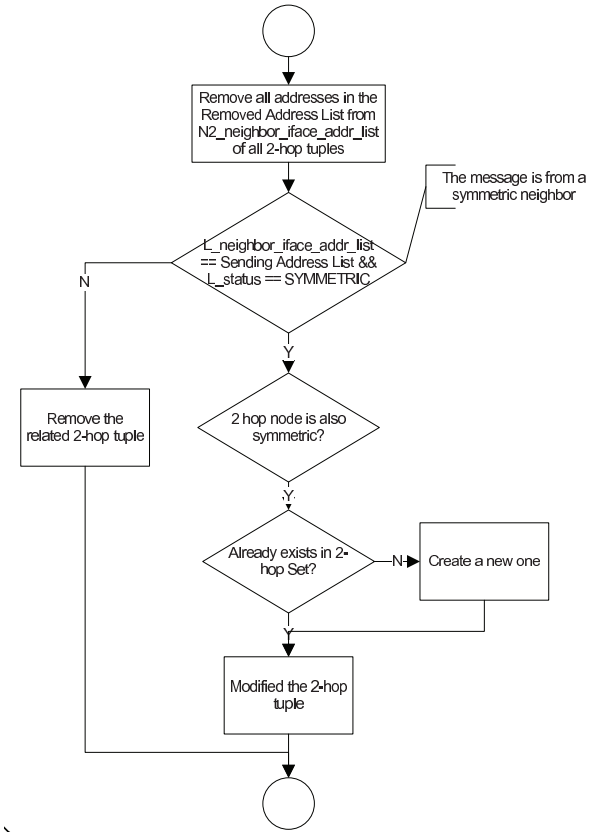


Figure 5.5: Update the *2-Hop Neighbor Set*

Based on the topology information of 1-hop and 2-hop neighbor, each node must select, from among its willing symmetric 1-hop neighbors, a subset of nodes as MPRs. MPRs are used to flood control messages from a node into the network, while reducing the number of retransmissions that will occur in a region. The use of MPR optimizes the classical flooding mechanism and reduces the shared topology information in the network.

5.1.3 Topology Discovery

Link Sensing and *Neighbor Detection* make the node be aware of its 1-Hop neighbors and 2-Hop neighbors by sending HELLO messages. To get the topology information located more than 2 hops away, *Topology Discovery* is needed. It is based on the broadcast of TC messages.

A node with one or more OLSRv2 interfaces, and with a non-empty neighbor set must generate TC messages. A node with an empty neighbor set should also generate "empty" TC messages for a period "hold" time after it last generated a non- empty TC message. Complete TC messages are generated and transmitted periodically on all OLSRv2 interfaces, with a default interval between two consecutive TC transmissions. In addition to the periodic broadcasting, it can be generated in response to a change of contents. Only MPR can forward the TC messages to the next hop.

When receiving a TC message, it is processed according to its type. The node first checks the message is from itself or unavailable. If so, the message must be discarded. Otherwise, the node will populate the related information base set (*Advertising Remote Node Set*, *Topology Set*, etc.) based on the received message. The procedure is shown in Figure 5.6. So based on the

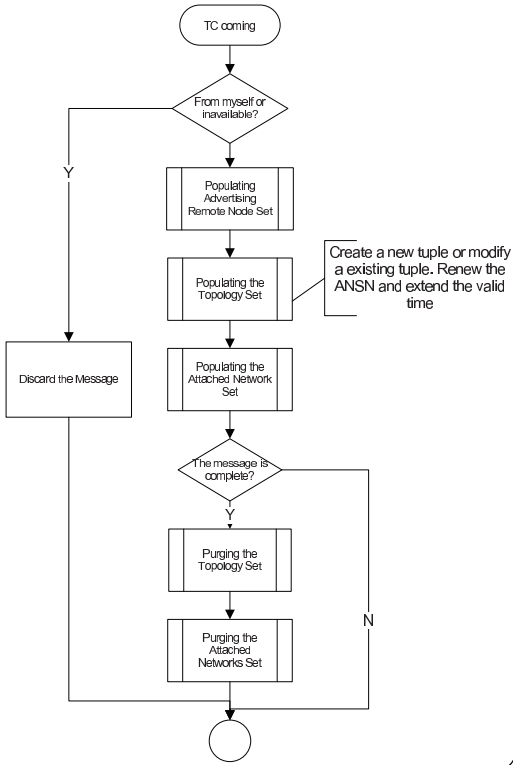


Figure 5.6: TC message processing

broadcasting and processing of TC messages, the topology information that more than two hops aways can be saved in the *Topology Set*.

This section summarized the *topology sensing* of MP-OLSR. Based on *topology sensing,* each connected node in the network is able to have the global view of the network topology. Because it shares the same idea as OLSRv2, only the main procedures are introduced here. For further information, please refer to (Clausen et al., 2009b,a,c).

5.2 Route Computation

In OLSR, routes are determined by nodes each time they receive a new topology control messages (TC or HELLO). The routes to all the possible destinations are saved in the routing table. For MP-OLSR, an on demand scheme is used to avoid the heavy computation of multiple routes for every possible destination. In this section, the hypotheses will be first introduced and followed with the algorithm that we proposed for multipath computation.

5.2.1 Hypotheses

The aim of the multipath algorithm is to build a set $\mathcal{K}$ of N paths, with no loops, joining a source node (noted s) and a destination node (noted d).

An ad hoc network can be represented by a graph $\mathcal{G} = (\mathcal{V}, \mathcal{E}, c)$ where $\mathcal{V}$ is the set of vertices, $\mathcal{E} \subset \mathcal{V} \times \mathcal{V}$ the set of arcs and $c : \mathcal{V} \rightarrow \mathcal{R}^{*+}$ a strictly positive cost function. We assume the graph is loopless, i.e. no arcs join a node to itself and that no pair of vertices can be connected by more

Algorithm 5.1 Calculate N routes in $\mathcal{G}$ from s to d

```
MultiPathDijkstra(s, d, G, N)
c_1 ← c
G_1 ← G
for i ← 1 to N do
  SourceTree_i ← Dijkstra(G_i, s)
  P_i ← GetPath(SourceTree_i, d)
  for all arcs e in E do
    if e is in P_i OR Reverse(e) is in P_i then
      c_{i+1}(e) ← f_p(c_i(e))
    else if the vertex Head(e) is in P_i then
      c_{i+1}(e) ← f_e(c_i(e))
    else
      c_{i+1}(e) ← c_i(e)
    end if
  end for
  G_{i+1} ← (V, E, c_{i+1})
end for
return (P_1, P_2, ..., P_N)
```

than one arc. Given an ordered pair of distinct vertices (s, d) we can define a path between s and d as a sequence of vertices $(v_1, v_2, ..., v_m)$ so that $(v_q, v_{q+1}) \in \mathcal{E}$, $v_1 = s$ and $v_m = d$.

The above representation implies that it is necessary to define what the cost function c refers to in an ad hoc context. Generally, the cost of a link is a quantitative measurement of its quality. The cost is incremental; and the smaller the link cost, the better. Different metrics can be applied. By default, the hop count metric is used.

5.2.2 Multipath Dijkstra Algorithm

For a source node s in the network, MP-OLSR will keep an *updated Flag* for every possible node in the network to identify the validity of the routes to the corresponding node. Initially, for every node i, the $updatedFlag_i$ is set to $false$, which means the route to the corresponding destination does not exist or needs to be renewed. When there is a route request to a certain node i, the source node will first check the $updatedFlag_i$:

- If the $updatedFlag_i$ equals $false$, the node will perform Algorithm 5.1 to get the multiple paths to node i, save it into the *multipath routing table*, and renew the corresponding $updatedFlag_i$ to *true*.
- If the $udpdatedFlag_i$ equals *true*, the node will find a valid route to node i in the *multipath routing table*.

The *multipath routing table* is a multimap that maintains the multiple paths from the current node to a defined destination. Every time the node receives a new TC or HELLO message and results in the changes in the topology information base, all the $updatedFlag$s will be set to $false$.

The algorithm to obtain the N paths from s to d is detailed in Algorithm 5.1.

The proposed algorithm is applied to a graph $\mathcal{G} = (\mathcal{V}, \mathcal{E}, c)$, two vertices $(s, d) \in \mathcal{E}^2$ and a strictly positive integer N. It provides a N-tuple $(P_1, P_2, ..., P_N)$ of (s, d)-paths extracted from $\mathcal{G}$. $Dijkstra(\mathcal{G}, n)$ (Algorithm 5.2) is the standard Dijkstra's algorithm which provides the source tree of the shortest paths from vertex n in graph $\mathcal{G}$; $GetPath(SourceTree, n)$ is the function that

Algorithm 5.2 The Dijkstra Algorithm

```
Dijkstra(s, d, G)
for all V ∈ V do
  dist[V] ← +∞
  previous[V] ← null
end for
dist[S] ← 0
Q ← copy(G)
while Q is not empty do
  u ← extract_min(Q)
  for all neighbor v of u do
    alt = dist[u] + length(u, v)
    if alt < dist[v] then
      dist[v] ← alt
      previous[v] ← u
    end if
  end for
end while
```

extracts the shortest-path to n from the source tree *SourceTree*; *Reverse(e)* gives the opposite edge of e; $Head(e)$ provides the vertex edge to which e points.

For the algorithm 5.2, $u \leftarrow extract\_min(\mathcal{Q})$ searches for the vertex u in the vertex set $\mathcal{Q}$ that has the least $dist[u]$ value. That vertex is removed from the set $\mathcal{Q}$ and returned to the user. $length(u, v)$ calculates the length between the two neighbor-nodes u and v. alt is the length of the path from the root node to the neighbor node v if it were to go through u. If this path is shorter than the current shortest path recorded for v, that current path is replaced with this alt path. And then we can read the shortest path by iteration.

The incremental functions $f_p : \mathcal{R}^{*+} \rightarrow \mathcal{R}^{*+}$ and $f_e : \mathcal{R}^{*+} \rightarrow \mathcal{R}^{*+}$ in Algorithm 5.1 are used at each step to get disjoint path between s and d. f_p is used to increase costs of the arcs that belong to the previous path P_i (or the opposite arcs belonging to it). This will make future paths tend to use different arcs. f_e is used to increase the costs of the arcs who lead to vertices of the previous path P_i. So there are three possible settings:

- if $id = f_e < f_p$, paths tend to be arc-disjoint;
- if $id < f_e = f_p$, paths tend to be vertex-disjoint;
- if $id < f_e < f_p$, paths also tend to be vertex-disjoint, but when not possible they tend to be arc-disjoint.

where id is the identity function. By using the cost functions, we can expect that there is diversity in the N paths regarding the network topology.

Figure 5.7 gives an example of multipath Dijkstra algorithm. As the topology shown in the figure, node S is trying to get multiple paths to node D. The cost function used are $f_p(c) = 3c$ and $f_e(c) = 2c$.

The hop count metric is used so that the cost of the links is initialized to one. When the Dijkstra algorithm is firstly applied, the first path obtained is: S→E→F→D. Then the cost functions are used to increase the cost of related links. The cost of S→E, E→F, F→D are changed from 1 to 3 by using $f_p(c)$, and S→A, C→D are changed from 1 to 2 by using $f_e(c)$. Then the Dijkstra algorithm is used again to get the second path S→A→B→C→D.

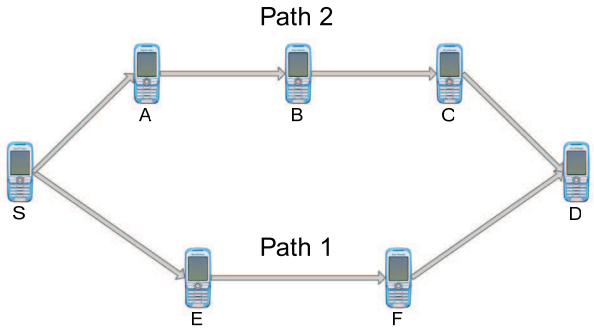


Figure 5.7: An example of multipath Dijkstra algorithm that tries to find two paths

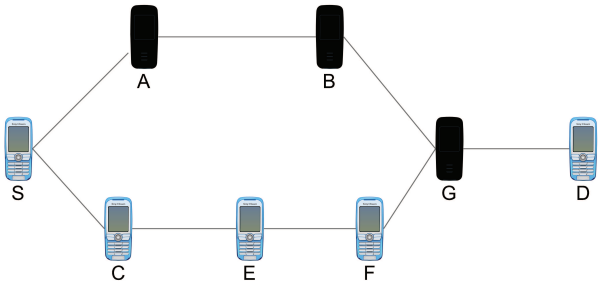


Figure 5.8: Multipath Dijkstra Algorithm in sparse case. The node disjoint path is non-desirable after node A, B, and G are removed.

But contrary to providing strict node-disjoint paths, the multiple paths generated by our algorithm do not need to be completely disjoint. The reason for this choice is that the number of disjoint paths is limited to the (s, d) minimal cut (defined as the size of the smallest subset of edges one cannot avoid in order to connect s and d). This minimal cut is often determined by the source and destination neighborhoods. For example, if s only has 3 distinct neighbors, one cannot generate more than 3 disjoint paths from s to d. As a consequence, this limitation of diversity may be local, the rest of the network being wide enough to provide far more than 3 disjoint paths. Another drawback of completely disjoint paths algorithms is that it may generate very long paths since every local "cutoff" can only be used once.

In (Zhou et al., 2005), the authors propose another algorithm to find multiple paths by deleting the vertices which were used. This can ensure that the paths obtained by the algorithm are strictly node-disjoint paths. However, in certain circumstances, this kind of strict node-disjoint scheme might cause some problems as mentioned above.

For example, in Figure 5.8, node S is trying to get multiple paths to node D. For MultiPath Dijkstra Algorithm, we use the number of hops as link cost metric and set $f_p(c) = 3c$ and $f_e(c) = 2c$ (more penalty to the used links). Initially, the cost for all the links is set to 1. For the first step, the shortest path S→A→B→G→D will be found. Then the cost functions will be used to increase the cost of the related arcs:

- S→A, A→B, B→G and G→D will be changed from 1 to 3 by using f_p.
- S→C and F→G will be changed from 1 to 2 by using f_e.

Then for the next step, the second shortest path S→C→E→F→G→D will be found. If we use the algorithm proposed in (Zhou et al., 2005), and we delete the intermediate node A, B and G

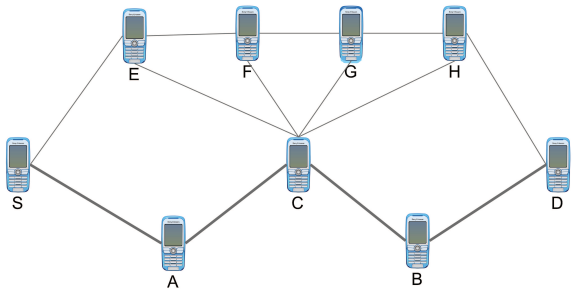


Figure 5.9: Obtaining different path sets by using different cost functions. Path S→A→C→B→D will first be chosen and second one might be different depending on the choice of cost functions

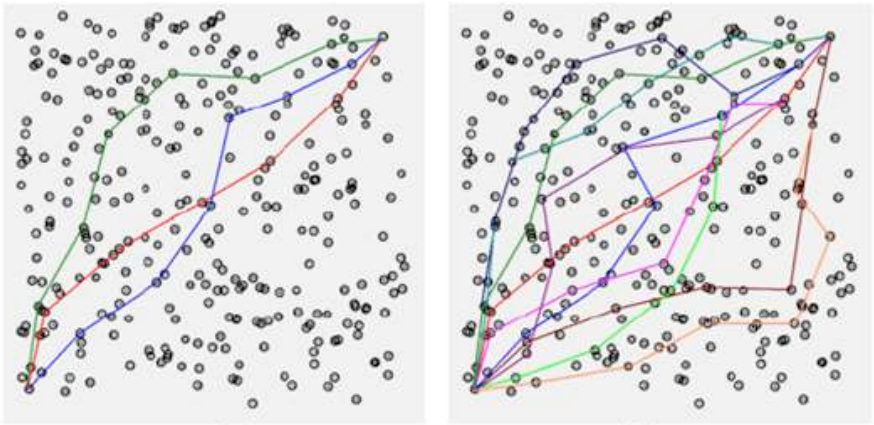


Figure 5.10: Finding different number of paths by Multipath Dijkstra Algorithm

after the first step, it is impossible to obtain the second path.

As illustrated above, another benefit of using cost functions is that we can get different multiple path set (node-disjoint or link-disjoint) by choosing different cost functions according to our preference and the network requirement. The network topology in Figure 5.9 is presented as an example.

If we choose $f_e(c) = c$ and $f_p(c) = 3c$, the paths we obtain are two link-disjoint paths: S→A→C→B→D and S→E→C→H→D.

If we choose $f_e(c) = 2c$ and $f_p(c) = 3c$, then the algorithm tends to search for node-disjoint paths. Then S→A→C→B→D and S→E→F→G→H→D will be found.

Figure 5.10 presents the effect of the algorithm in a more complex scenario with 300 nodes. In the left figure, the algorithm tries to find three paths, and three node-disjoint paths are found. In the right figure, the algorithm tries to search ten paths. Because there is not enough nodes to build ten totally disjoint paths, the algorithm obtained ten paths with certain diversity according to the setting of the parameters.

5.3 Packet Forwarding

In OLSR, the hop-by-hop routing is used. When a node receives a new routing message, it will update its routing table to all the possible destinations in the network. Each routing table entry

describes the collection of best paths to a particular destination. When forwarding an IP data packet, the routing table entry providing the best match for the packet's IP destination is located. The matching routing table entry then provides the next hop towards the packet's destination. Figure 5.11 shows the structure of the data packet transmitted in the network (by using UDP protocol). The IP header includes the source address and destination address of the packet. So when a packet reaches an intermediate node, the node just read the destination address from the IP header and finds a match from its own routing table. In another word, the routing decision is made at local node.

Figure 5.11: The data packet in OLSR

For MP-OLSR, the routing decision is made at the source node. To guarantee the packets travel through the multiple paths, the source routing is employed. Thus, the data packets need to carry the source routing information, i.e. the path information from the source to the destination. Figure 5.12 presents the packet used in MP-OLSR. The MP-OLSR header, which defines the nodes need to travel for the packet, is inserted between the IP header and UDP header.

Figure 5.12: The data packet in MP-OLSR

Figure 5.13 shows the structure of the MP-OLSR header. It contains a 8-byte common header and the source addresses. The common header includes the following fields: *Version, Protocol, Source routing length* and *Flow*. Then it is followed by the address list of the source route. The length of the header depends on the number of nodes in the source node.

From the aspect of traffic allocation granularity, there are two possible choices: *per-connection allocation* and *per-packet allocation*. For per-connection allocation, all the packets of a connection are constrained to follow the same path. However, with this configuration, it is difficult to ensure that traffic is distributed evenly over multiple paths since connections can vary widely in their

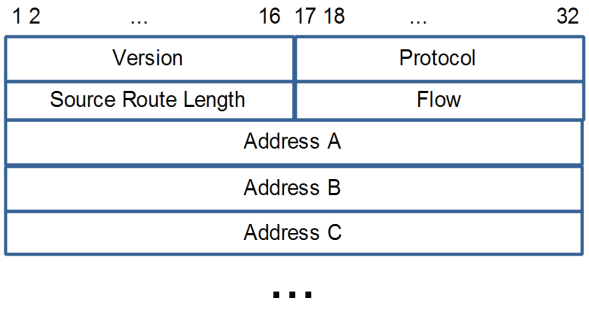


Figure 5.13: MP-OLSR header

rates of flow. Moreover, the efficiency with which the network resources are utilized in such an allocation depends on the relative duration of the connections.

In the literature, it is believed that a smaller allocation granularity would permit a finer control to be exerted over the performance of the network resources (Krishnan & Silvester, 1993). A per-packet allocation allows a more graceful degradation of service in case of component failures along a path. It responds to a component failure by aborting the connection, and restarts the connection after a new path is determined. On the other hand, the per-packet allocation merely adjusts the fraction of traffic routed over the multiple paths such that the unreliable path is bypassed completely. Thus spreading the traffic across similar paths is more robust, since less traffic is likely to be affected by a component failure.

Thus, in MP-OLSR, the per-packet allocation scheme is employed. The packet will be distributed into different paths by using *round-robin* algorithm. In fact, the data packets can also be distributed by using the lower layer information. This will be further discussed in section 5.6 on page 87.

5.4 Route Recovery

5.4.1 Route Failure in MP-OLSR

By using the scheme of the *Topology Sensing*, we can obtain the topology information of the network with the exchange of HELLO and TC messages. All this information is saved in the topology information base of the local node: link set, neighbor set or topology set. Ideally, the topology information base can be consistent with the real topology of the network. However, in reality, it is hard to achieve, mainly because of the mobility of the ad hoc network.

Firstly, for the HELLO and TC messages, there are certain intervals during each message generation (2s for HELLO and 5s for TC by default (Clausen & Jacquet, 2003)). During this period, the topology might change because of the movement of the nodes. Secondly, when the control messages (especially the TC messages) are being transmitted in the network, delay or collision might happen. This will result in the control message being outdated or even lost.

Both of the two reasons mentioned above will result in the inconsistency between the real network topology and the node's topology information base. This means that when a node is computing the multiple paths based on the information base, it might use links that do not exist anymore, and cause the route failure.

Furthermore, even if the topology information is correct when the route is being constructed at the source node, the topology might change while the packets are being forwarded in the network. And because of the source routing scheme MP-OLSR uses, the source route cannot be adapted to this kind of changes.

For the OLSR, the problem is less serious because it uses hop-by-hop routing. Unlike the source routing, whose routes are decided completely at the source, the nodes in OLSR just forward the packets to the next hop. So there is more chance for a node in OLSR to forward a packet to the next available link.

5.4.2 Route Recovery Algorithm

Several techniques already exist in the literature to deal with the route failures in source routing. DSR handles route errors using route maintenance, mainly by sending RERR messages, which will

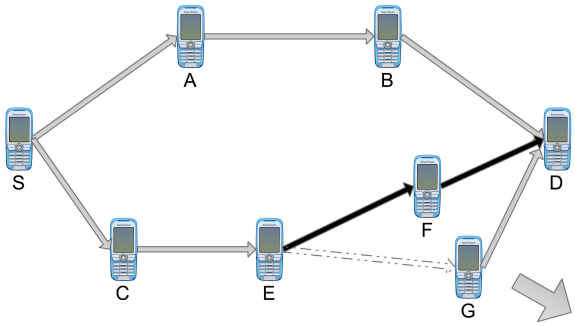


Figure 5.14: An example of route recovery. S is the source and D is the destination. The movment of G makes the link from E to G unavailable.

increase the end-to-end delay significantly. In (Tsai & Moors, 2009), the authors propose another method to avoid the effect of short term link deterioration by using opportunistic paths in mesh networks.

To overcome the disadvantage of the source routing, we propose *Route Recovery* for MP-OLSR. The principal is very simple: before an intermediate node tries to forward a packet to the next hop according to the source route, the node first checks if the next hop in the source route is one of its neighbors (by checking the neighbor set). If yes, the packet is forwarded normally. If no, then it is possible that the "next hop" is not available anymore. Then the node will recompute the route and forward the packet by using the new route.

In Figure 5.14 we present an example of route recovery. Node S is trying to send packets to D. The original multiple paths we got are S→A→B→D and S→C→E→G→D. However, node G moves out of the transmission range of node E and makes the second path unavailable. The source node S is not able to detect the link failure immediately (because of the delay and long interval of TC messages) and keeps sending the packets along the path, and all these packets are dropped during this period if only the source routing is used. With Route Recovery, when the packet arrives, node E will first check if node G is still one of its neighbors, before forwarding the packet according to the source route. If not, node E will recompute the route to node D, and get E→F→D. Then the following packets will be sent through the new path.

Because the *Route Recovery* just checks the topology information saved in the local node, it will not introduce much extra delay. And most importantly, it will effectively improve the packet delivery ratio of the network. This will be proved by the simulation in section 6.2.4.1 on page 107.

5.5 Loop Detection

5.5.1 Loops in OLSR and MP-OLSR

It is important to mention the LLN (Link Layer Notification) before coming to the problem of the loops of the protocol. LLN is an extended functionality defined in (Clausen & Jacquet, 2003), and implemented in different OLSR or MPOLSR simulations and implementations[1]. If link layer information describing connectivity to neighbor nodes is available (i.e. loss of connectivity though absence of a link layer acknowledgement), this information can be used in addition to the information from the HELLO-message to maintain the neighbor information base and the MPR selector

[1] nOLSRv2, Niigata OLSRv2 Implementation, Niigata University, Japan

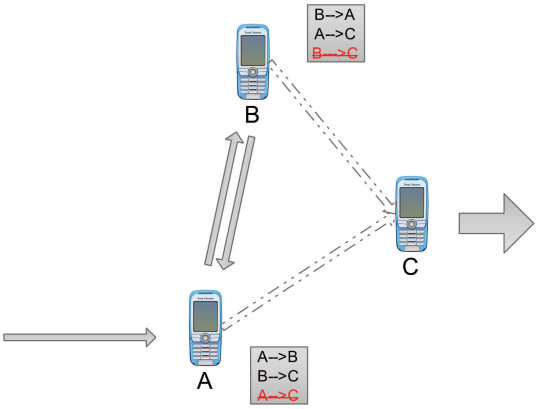


Figure 5.15: An example of loop in the network. The movement of node C results in inconsistency of the information bases in node A and B. One transient loop is formed between A and B.

set. The routing protocol can act on the acknowledgement from LLN (mainly the loss of links), and remove the corresponding links from its information base. The results of the real OLSRv2 testbed (Owada et al., 2007) and our work based on NS2 simulation in section 6.1.3 can show that LLN is very important and effectively improves the packet delivery ratio of the OLSR and MP-OLSR protocol.

In theory, the paths generated by the Dijkstra algorithm in MP-OLSR are loop-free. However, in reality, the LLN and *Route Recovery* which are used to adapt to the topology changes make the loops possible in the network. With LLN, when a node tries to send a packet over a link but fail in the end, the link layer will give a feedback to the routing protocol to notify the link loss. This kind of abrupt interruption will result in additional operations on the topology information base rather than just regular HELLO and TC messages. This means that other nodes cannot be aware of these changes immediately. So LLN might cause some inconsistency of the topology information in different nodes. And with *Route Recovery*, which might change the path in intermediate nodes, loops can occur temporarily in the network.

In Figure 5.15 we give an example of how a loop is generated in the network. Node A is an intermediate node of a path. The packets with source route A→C arrive at node A and need to be forwarded to node C. Then node C moves out of the transmission range of node A and node B, and makes the links A→C, B→C not available anymore.

When the new packets arrive at node A, the transmission to node C will fail. Then in node A, the routing protocol will be acknowledged by LLN, and it will remove the link A→C from node A's link set. For node A, although it can detect the link failure of A→C by LLN, it is hard to know the failure of B→C immediately. This is because link B→C can only be removed when the NEIGHB HOLD TIME (6 seconds by default (Clausen & Jacquet, 2003)) expires. In the meantime, *Route Recovery* will be awaken. A new path A→B→C will be established and the following packets will be forwarded along the new path. Then the packets will be redirected to node B. The same operation will be performed in node B: LLN of the failure of B→C, and Route Recovery. Unfortunately, because node B cannot detect the link failure of A→C immediately, and the new path obtained by *Route Recovery* is B→A→C. Thus the packet will be returned to node A, and node A to B again, creating a loop. This is not a permanent loop, but a transient loop which will exist for several seconds and will disappear when the related link expires. However, this

kind of temporary loops will block the links in the loop and congest the related transmission area.

In (Speakman et al., 2008), the authors also address the looping issues in OLSRv2, and LLN will significantly increase the number of loops.

5.5.2 Source Route Loop Detection

In (Speakman et al., 2008), the authors introduce two types of loop detection techniques: LD-Mid (Mid-Loop Detection) and LD-Post (Post-Loop Detection). LD-Mid just compares the address of the next hop against the address of the previous hop, so it is only able to detect "two-way" loops between 2 nodes. LD-Post records all incoming packets that need to be forwarded and compares against each new incoming packet to see if the same packet has traversed this node before. So it can detect loops that are farther away, by taking more memory. When a loop is detected, the Packet Discard strategy is used to drop the packets that are unlikely to reach the destination but only increase the load of the network.

For MP-OLSR, we propose a simple method based on source routing that can effectively detect loops without causing extra cost of memory: after the Route Recovery is performed, we can get a new set of multiple paths from the current node to the destination. The node will compare the first new path with the ancient source route in the packet. We can verify if the new path includes the nodes that the packet had crossed before. If the answer is no, it means that there will be no loop in the future, and we will make use of the new path. If the new path includes the node that the packet have passed before, there is high probability that a loop will happen (a very rare case is that the failed link is recovered in this short period, in several milliseconds, then the loop is released). MP-OLSR will switch to the next path of the multiple paths set, until all the paths have been verified. If there is no suitable path, the packet will be discarded.

For the example in Figure 5.15, node A will get a path A→B→C by Route Recovery. Then when the packet arrives at node B, a new path B→A→C will be generated because of link breakage of B→C. Node B will compare the new one with the ancient source route A→B→C in the packet. We will find that the packet has already crossed node A, and so there might be a loop. Then we will try to find if there is any other possible path, or else the packet will be discarded.

Compared with LD-Post, which needs to keep a record of all the incoming packets, our loop detection mechanism could effectively detect the possible loops in the network without consuming extra memory space. By reducing the loops in the network, the network congestion can be reduced. So the performance of the network can be improved, especially the end-to-end delay. In Section 6.2.4.1 on page 107, we show the effect of the loop detection mechanism by simulating the network.

5.6 The Queue Length Link Metric

The single path and multipath algorithms discussed in the previous of this chapter are based on the hop count metric. The costs of the links between two nodes are treated equally, i.e. the path with the least hop counts is considered as the best path. However, this measurement is not always correct because it does not consider the quality of the links. In fact, in a network with high data rate, some of the nodes are congested easily because of unstable links.

The link metric to be used in MANET is a very large topic. The first related Internet Draft based on ETX metric came out recently in March 2010 (Rogge et al., 2010). However, although

the ETX metric is used widely in mesh networks, there is also article presenting that ETX is flawed when estimating optimal routes in real mesh networks and worse than the OLSR RFC standard (Johnson & Hancke, 2009). Within SEREADMO project, we has also tried related work in (Poussard et al., 2009) by using BER. It can offer better performance in certain scenarios, but the benefit is not obvious in various situations.

In this section, we propose to use the queue length as link metric for computation of routing paths: firstly, the queue length can reflect the quality of the links faithfully. The unstable links will result in packet retransmission in the MAC layer, which will increase the queue length significantly. With our observation during the simulation, the average queue length has important effect on the delay. Secondly, the FIFO queue works in the same level as the routing protocol. Employing the queue length as metric does not need the cross-layer interaction, which facilitates the protocol implementation. This part of work related to queue length metric is extracted from the Wissam Hamdash's master thesis (Hamdach, 2010).

5.6.1 Queue Monitor

To monitor the queue, we set a timer event with period T to sample the length. However, the queue length is an instant value which changes dramatically in very short period of time. To avoid the jitter for route computation, we model the queue length with monitor function $f(t)$:

$$f(t) = \begin{cases} 0.8f(t-1) + 0.2x(t), & x(t) < f(t-1) \\ 0.5x(t) + 0.5f(t-1), & x(t) \geq f(t-1) \end{cases} \tag{5.1}$$

where $f(t)$ is the current modeled queue length, $f(t-1)$ is the modeled queue length at the previous sampling point (T *seconds ago*). $x(t)$ is the instant real queue length at the current sampling point.

Figure 5.16 illustrates the behavior of the monitor function with sampling interval $T = 0.5$ second. The function is sensitive to the increase of the queue. However, when the length becomes to decrease, the function drops slowly and tries to "memorize" the older queue length. This is inspired by the concept of *locality* (Denning, 2005) in computer science: an access to a memory location indicates that the nearby locations will very likely to be accessed again soon. In the network, for the congested nodes, there is also much higher probability than normal nodes that the congestion will happen in a near future. The function can also reduce the jitter when performing route computation.

5.6.2 Message Encapsulation and Propagation

In OLSR and MP-OLSR, the topology information is exploited by transmission of HELLO and TC messages. To make other nodes in the network be aware of the queue length information in the local node, it is necessary to attach the information to the HELLO and TC messages. Thanks to the TLV mechanism of OLSRv2 (Clausen et al., 2009c), it is easy to add an additional TLV for the queue length information.

Thus, a *TLV_ Queue_ Length* is defined. It is composed of 3-byte TLV header and 1-byte for queue length, as illustrated in Figure 5.17. The length is normalized into a integer between $[0, 255]$ to prevent the overflow of 1-byte space. The TLV can then be carried by TC and HELLO messages

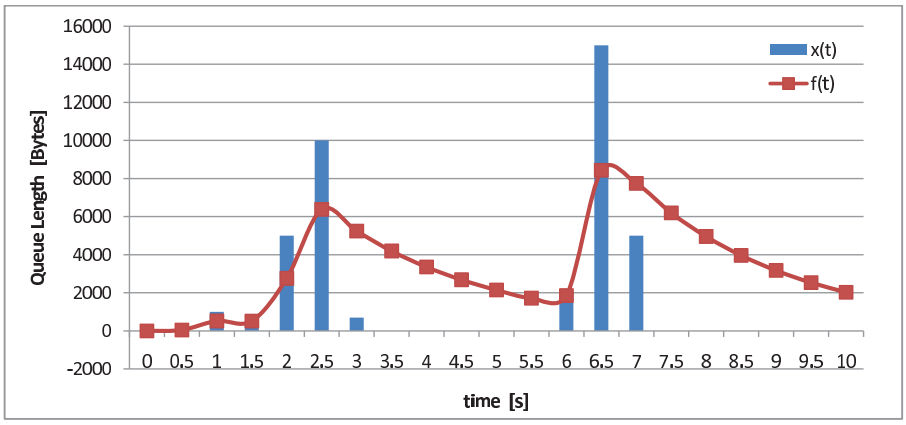


Figure 5.16: The monitor function for queue length. $x(t)$ is the instant real queue length, $f(t)$ is the modeled queue length.

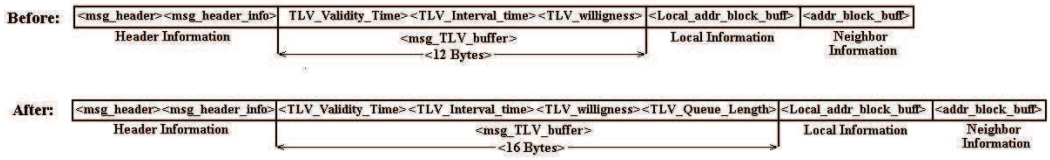


Figure 5.17: *TLV_ Queue_ Length* added to the HELLO message

and broadcasted to the whole network.

5.6.3 Message Processing

Upon the reception of HELLO or TC messages, the queue length is extracted from *TLV_ Queue_ Length*. A linear weight function is employed to define the link cost c between two nodes:

$$\begin{cases} c = \frac{\alpha Q_{av}}{Q_{max}} + 1 \\ Q_{av} = \frac{(Q_1 + Q_2)}{2} \end{cases} \tag{5.2}$$

where $Q_1$is the normalized queue length in local node, Q_2 is the normalized queue length of the neighbor node. α is the slope coefficient which determines the effect of queue length to the link cost.

We proposed two different slope coefficients, $\alpha_1 = 315$ and $\alpha_2 = 3200$ (Figure 5.18) to test the effect different punishment of the queue length. Those parameters are chosen to test the behavior of the protocol with low punishment and high punishment in the link cost respectively. Then the Multipath Dijkstra Algorithm can make use of the cost based on queue length as the initial cost of the links. The simulation results are presented in section 6.2.4.5 on page 120.

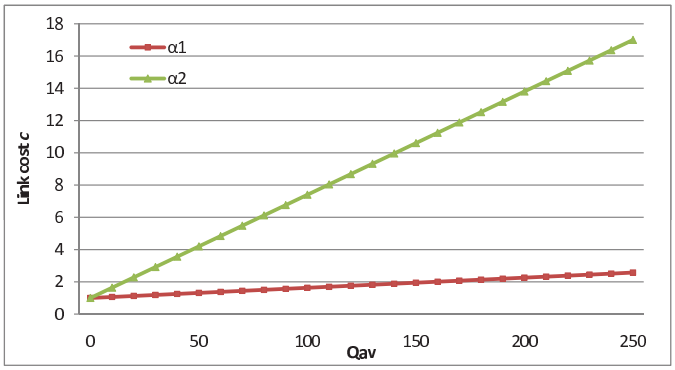


Figure 5.18: Weight function with different slope coifficients $\alpha_1 = 315$ and $\alpha_2 = 3200$

5.7 Compatibility of MP-OLSR and OLSR

5.7.1 Challenges

As presented in the related work, most of the multipath routing protocols proposed are based on single path version of an existing routing protocol: AODV and AOMDV, DSR and SMR. However, the backward compatibility of those protocols with its single path version is not considered. In fact, because the reactive based multipath routing requires extra operation during the route discovery to gather enough information for the multipath construction, the compatibility is not easy to achieve.

Given OLSR, the standardized protocol, and the distributed/ heterogeneous property of ad hoc networks, it will be interesting to study the compatibility between MP-OLSR and OLSR, which will facilitate the application and deployment of the multipath routing. Furthermore, in some scenarios that multipath routing has no much performance enhancement (e.g. sparse scenarios where no multipath can be found or static scenarios), we can keep the possibility of using single path routing to reduce the computation complexity by providing good flexibility.

As presented in the introduction, MP-OLSR is based on the OLSR protocol. In fact, the two protocols follow the same steps for the detection of neighborhood and network topology, but they are different in routing the data packets. In OLSR, the source node calculates the shortest path to the destination and sends the packets to the next hop. The intermediate OLSR nodes forward the packets according to their routing table. In MP-OLSR, the source node calculates different paths, and specifies one path in each packet (source routing) before sending it to the next hop. And the intermediate MP-OLSR nodes will forward the packet according to the source routing initialized by the source node. In Figure 5.12 on page 83, we have proposed a MP-OLSR header for source routing. It works well if the copatibility with OLSR is not considerded. However, it is impossible for other nodes to parse the packet with this tructure.

The study of backward compatibility makes the deployment of the new protocol much easier because it can make use of the network that already exists. Moreover, it allows returning to the single-path version if necessary (basically with no mobility and low traffic). It is important to point out that we are studying the mutual compatibility between OLSR and MP-OLSR. In the following, we show that each protocol can use the nodes of either protocol to perform routing, with respect to QoS parameters.

5.7.2 IP Source Routing and MP-OLSR

To ensure the compatibility between the two protocols OLSR and MP-OLSR, we propose an implementation of MP-OLSR protocol based on IP-source routing (IETF, 1981). The IP is the standard intercommunication protocol used in Internet, so it can simplify the protocol and provide better efficiency than adding an extra source routing head as discussed in Section 5.3 on page 82.

The IP source routing allows partially or completely specifying the path in the data packets. This option is mostly used by network administrators to test the routes. The IP protocol supports two forms of source routing:

- Strict source routing: the exact route of the packet is specified by the sender.
- Loose source routing: the sender gives one or more hops that the packet must go through. This means that before reaching its final destination, the packet should go through the following IP addresses as intermediate destinations. These intermediate destinations are responsible for forwarding it to the next destination.

The IP source routing option implicitly ensures the compatibility between OLSR and MP-OLSR. Indeed, when OLSR nodes receive MP-OLSR packet (with source route), they will forward it directly according to the IP source routing. On the other hand, when an MP-OLSR node receives a packet generated by an OLSR node (without source route), it will recompute the path as the packet is from its application layer and attaches the source route to the packet. So this packet can also take advantage of the loop detection in all the following MP-OLSR intermediate nodes.

However, the IP source routing option accepts only a maximum of 9 addresses (in total, 11 nodes including source and destination nodes = 10 hops) because of the limitation of the length of the IP head. Therefore, when the route contains more than 10 hops (large network), other solutions must be proposed. To solve this problem, we propose two possible solutions:

- The first solution still makes use of strict source routing. If the path found by MP-OLSR is more than 10 hops, it will just simply forward packets to the next hop, without using the source routing. The next hop will decide the rest of the route, no matter whether it is an MP-OLSR node or an OLSR node. This solution is easy to implement but does not guarantee the multiple paths as defined by the source node.
- The second is by using the loose source routing: the source node just specifies 10 "key" hops that the packet needs to travel to get to the destination and allows each intermediate node to choose a route to the next hop. This solution can guarantee the source routing as defined by the source node. But it requires the loose source routing support from the IP layer and the MP-OLSR protocol to maintain a routing table just like OLSR.

Based on IP source routing, the MP-OLSR nodes can cooperate with the OLSR nodes in the networks. This feature is important for the protocol in practice. In Section 6.2.4.6, the simulation is done to validate the method.

5.8 Conclusion

In this chapter, the specifications of MP-OLSR are introduced. The *topology sensing* (based on OLSR) and *route computation* are basic procedures for multipath routing. To improve the

performance of the protocol, the *route recovery* and *loop detection* are proposed to avoid route failure and reduce transient loops in the network. We proposed a queue length metric to evaluate the link quality. The queue length information is saved in TLVs and propagated to the whole network by HELLO and TC messages. The compatibility between MP-OLSR and OLSR are also studied to make the single path routing and multipath routing be able to cooperate with each other.

In the following of this part, different experimentations are performed to test the multipath routing protocol in various scenarios.

Chapter 6

Simulation and Performance Analysis

To evaluate the performance of the protocol, the simulations are performed. The network simulator is a software or hardware that predicts the behavior of a network, without an actual network.

The networks simulation can be defined as the process of designing a model of a real network system and conducting experiments with this model for the purpose of understanding the behavior of the system and/or evaluating various strategies for its control. To make this process useful, the behavior of the model is expected to faithfully mimic the response of the system under study (Caro, 2003).

The *discrete-event simulation* is the main tool used to study the characteristics of the communication networks. The term *discrete* implies that the operation of a system is represented as a chronological sequence of events. Each event occurs at an instant in time and marks a change of state in the system.

In *discrete-event simulation*, a system is modeled as it evolves over time by a representation in which the system state changes instantaneously when an event occurs, where an *event* is defined as an instantaneous occurrence that causes the system to change its state or to perform a specific *action*. The events can be the periodic packet transmission of the routing protocol, the arrival of a packet, etc. The actions can be updating of the router information, router computation, etc.

During the simulation, an *event queue* is maintained by the simulator. In the queue, each event is associated with an *event time,* i.e. the time at which the event is set to occur. The events in the event queue are sorted by the event time. The simulator also maintains a simulation clock which is used to simulate time. The simulation clock is advanced in discrete steps. The simulator operates by continually repeating the following series of steps until the end of simulation:

- The simulator removes the first event from the event queue, i.e., the event scheduled for the earliest time.
- The simulator sets the simulation clock to the event time of the event. This may result in advancing the simulation clock.
- The simulator handles the event, i.e., it executes the actions associated with the event. This may result in changing the system state, scheduling other events, or both. If other events are scheduled, they may be scheduled to occur at the current time or in the future.

The network simulation is an extremely powerful and flexible tool. It allows studying a great number of possible network scenarios with different protocols and configurations included in the

simulation model. Compared to the real testbed, it is much easier to implement and less expensive. With the network simulation, the researchers are expected to have meaningful conclusions of practical interest.

Nowadays, the popular network simulators include *Qualnet* [1], *Network Simulator 2 (NS2)* [2] and *OPNET* [3]. In our study, NS2 and Qualnet are used to simulate the wireless network and evaluate the multipath routing protocol.

In this chapter, we firsty introduced the implementation and simulation based on NS2. Then we change our simulation platform to Qualnet because some of the flaws in NS2. All the functions proposed in Chapter 5 are simulated, verified and discussed in detail. The source code mentioned in this chapter is available online[4].

6.1 Simulation Based on NS2

6.1.1 Introduction of *Network Simulator 2*

The NS network simulator is from U.C. Berkeley/LBNL, is an object-oriented discrete event simulator targeted at networking research and available as public domain. Its first version (NS- 1) began in 1989 as a variant of the REAL network simulator and was developed by the Network Research Group at the Lawrence Berkeley National Laboratory (LBNL), USA. Its development was then part of the Virtual InterNetwork testbed (VINT) project, supported by Defense Advanced Research Projects Agency (DARPA), at Lawrence Berkeley National Laboratory (LBNL), Xerox PARC, and UCB. The NS1 inherits the work of REAL simulator, including several types of TCP (SACK, Tahoe and Reno) and routing protocol. The description language used by NS1 is an extension of Tcl. Each simulation scenario is described by a simulation scenario. By using the command provided by NS, the researcher can define the network topology, configure the network traffic, collect the statistic information, run the simulation, etc. By building this kind of generic script language, NS can describe the network configuration efficiently.

The aim of the VINT was not to design a new network simulator, but to unify the effort of all people working in the research field of network simulation. The result is that NS2 is widely used in the networking research community and has found large acceptance as a tool to experiment new ideas, protocols and distributed algorithms. Compared to NS1, there are three main improvements:

- Redefinition of the object structure.
- Make use of the Object-Tcl to replace the Tcl for simulation configuration.
- The code of the Otcl interpreter is separated for the main simulator. Almost all the functions of NS1 are integrated into NS2.

Currently NS2 development is still supported through DARPA. NS has always included substantial contributions from other researchers, including wireless code for both mobile ad hoc networks and wireless LANs from the UCB Daedelus, CMU Monarch projects and Sun Microsystems.

The NS2 is well-suited for packets switched networks and wireless ad hoc networks. It is widely used for small scale simulations of queuing and routing algorithms, transport protocols, congestion

[1] Qualnet, http://www.scalable-networks.com/
[2] NS2, http://www.isi.edu/nsnam/ns/
[3] OPNET, http://www.opnet.com/
[4] MP-OLSR source code, http://www.jiaziyi.com/MP-OLSR

control, and some multicast related work. It provides substantial support for simulation of TCP, routing, and multicast protocols over wired and wireless networks.

Because NS2 is an effective simulation tool and also an open source software, it is the mostly used simulator for studies on mobile ad hoc networks. So it plays an important role in the research community of mobile ad hoc networks.

For the first year of my PhD research, the NS2 simulator was employed for network simulation.

6.1.1.1 Structure of *NS2*

The NS2 architecture closely follows the IP reference model. The code of NS2 is split between for its core engine and OTcl for configuration and simulation scripts, as shown in Figure 6.1.

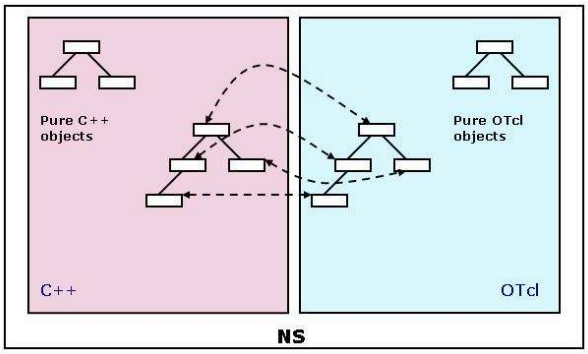


Figure 6.1: The NS2 split object model

The reason to use two kinds of language is that for a simulator, there are two issues to handle. Firstly, to implement a detailed simulation of the protocol, a high-performance language is needed to process the packet and packet header very quickly. Those algorithms need to run numerous times so that the speed is very important. Secondly, a lot of the research is based on the configuration and parameters of the networks. In another words, the simulation scenario scripts need to be modified frequently. In the meantime, those configuration scripts just need to run once for each scenario at the beginning of the simulation, so the speed is less important.

The combination of the two languages offers a kind of compromise between performance and ease of use. The package provides a compiled class hierarchy of objects written in C++ and an interpreted class hierarchy of objects written in OTcl related to the compiled ones. The user creates new objects through the OTcl interpreter. New objects are closely mirrored by a corresponding object in the compiled hierarchy. Tcl procedures are used to provide flexible and powerful control over the simulation (start and stop events, network failure, statistic gathering and network configuration). The OTcl interpreter provides commands to create the networks topology of links and nodes and the agents associated with nodes.

The basic element in the network is *node*. The function of a *node* is to receive a packet, to examine it and map it to the relevant outgoing interfaces. A node is composed of simpler *classifier* objects. Each *classifier* in a node performs a particular function, looking at a specific portion of the packet and forwarding it to the next *classifier*. The Node itself is a standalone class in OTcl. However, most of the components of the node are themselves TclObjects. The typical structure of a (unicast) node is as shown in Figure 6.2 (Fall & Varadhan, 2010).

The *Agent* is another important concept in NS2. It represents endpoints where network-layer

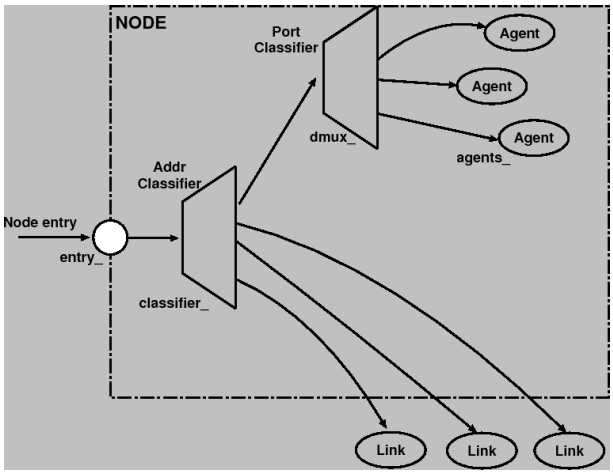


Figure 6.2: The structure of a unicast node in NS2

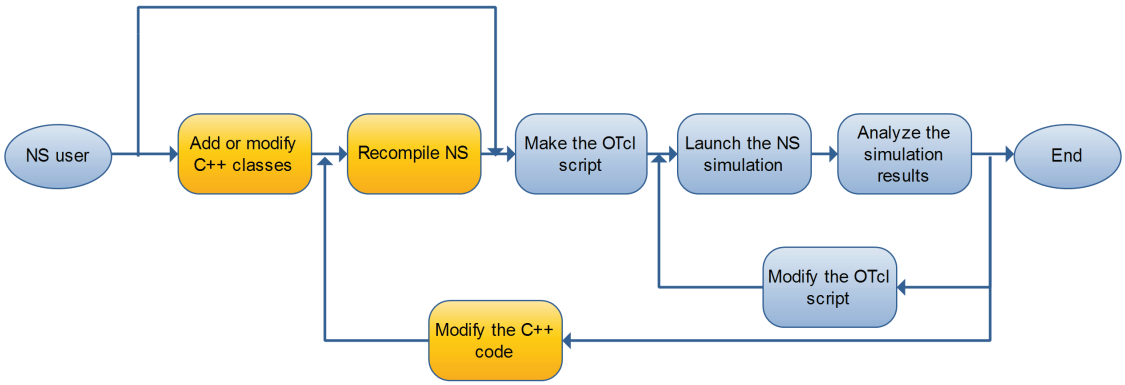


Figure 6.3: The procedure to take for NS2 simulation

packets are constructed or consumed, and are used in the implementation of protocols at various layers. Users create new sources or sinks from the class Agent.

Link is another aspect of the network topology. In NS2, *Links* are modeled either as simplex- or duplex-links with a predefined capacity, delay, and queuing discipline. In addition, links can be turn down or restored at any point in time during the simulation, simulating link failures.

6.1.1.2 Simulation procedure of *NS2*

Before the simulation by using NS2, the researchers have to define what kind of network elements will be used. If all the components (routing protocol, application, radio channel, etc.) needed are already provided by NS2, then there is no necessary to modify the NS itself. Only writing the OTcl script is enough. If the component is not included in NS2, then the researchers have to extend NS2 by adding new C++ class and OTcl class by themselves. And then write the OTcl script for network simulations. Figure 6.3 shows the procedure to be taken in NS2 simulation.

1. To add a new component in NS2, the researchers have to add new C++ classes and then recompile NS2. Those components can be new routing protocols, MAC protocols, application protocols, etc. After the extension, the new network component is ready to be used in a simulation.

Algorithm 6.1 The *recv()* method for MP-OLSR in NS2

```
void MPOLSR::recv(Packet * p, Handler * h)
Parse the header of packet p
if The packet is a MPOLSR routing packet then
  Process the TC & HELLO routing packet
else if The packet is originated by the local node then
  Add the IP header to the packet
  if updatedFlag_i = false then
    Call the Multipath Dijkstra Algorithm
  end if
  Find a route from the multipath routing table
  if There is no route found then
    Throw the NO_ROUTE error
    return
  end if
  Add the source routing heade to the packet
  Forward the packet to the next hop
else
  Forward the packet according to the source route
end if
```

2. Write the OTcl script. The first thing is to configure the network topology and basic link characteristic such as bandwidth, packet loss strategy, etc.

3. Build the agents and bind the related protocols to the end points.

4. Configure the application traffic over the network.

5. Set the *trace* object. The *trace* object is used to record the specified events into a trace file. By recording this log file, the whole procedure of the simulation is saved. It is used to analyze the network performance after the simulation.

6. Set other parameters for the simulation, such as simulation time, node mobility, etc.

7. Use the *ns* command to run the OTcl script.

8. Analyze the trace file generated to evaluate the network performance.

9. Adjust the protocol and the OTcl script according to the results obtained and relaunch the simulation until get needed results.

6.1.2 Environment and Assumption

Implementation of MP-OLSR in NS2

To simulate the MP-OLSR protocol, the multipath routing is implemented in NS2 based on *um-olsr* [5] implementation. The MP-OLSR module exists as a *routing agent* in NS2. When MP-OLSR receives a packet from upper layers (normally UDP agent), the *recv(Packet *p, Handler *h)* method is called. The *recv()* method is the main entry point for an *Agent* which receives packets, and is invoked by upstream nodes when sending a packet. In most cases, Agents make no use of the second argument (the handler defined by upstream nodes). It is done as Algorithm 6.1.

[5] UM-OLSR, http://masimum.dif.um.es/?Software:UM-OLSR

Simulation Parameters

In the simulation, the channel capacity of mobile hosts was set to 11 Mbps. A two-ray ground reflection model, which considers both the direct and a ground reflection path, was used as radio propagation model. In this model, the received power at distance d is predicted by

$$P_r(d) = \frac{P_t G_t G_r h_t^2 h_r^2}{d^4 L} \tag{6.1}$$

where P_t is the transmitted signal power, G_t and G_r are the antenna gains of the transmitter and the receiver respectively. $L(L \geq 1)$ is the system loss, h_t and h_r are the heights of the transmit and receive antennas respectively. We use the DCF (Distributed Coordination Function) of IEEE 802.11 for wireless LAN as the MAC layer protocol. It has functionality to notify the network layer about link breakage.

The mobility model used is the Random Waypoint Mobility Model (Camp et al., 2002). The Random Waypoint Mobility Model includes pause times between changes in direction or speed. A node begins by staying in one location for a certain period of time. Once this time expires, the node chooses a random destination in the simulation area and a speed that is uniformly distributed between the min speed and max speed. The node then travels toward the newly chosen destination at the selected speed. Upon arrival, the node pauses for a specified time period before starting the process again. One of the problems of the Random Waypoint is that its initial random distribution of nodes is not representative of the manner in which nodes distribute themselves when moving. So in our simulation, the data transformation starts after 20 seconds of the begin of the simulation.

Other detailed parameters for the simulation is shown in Table 6.1.

6.1.2.1 Performance Metrics

The following metrics are used to evaluate the performance of the protocols:

- Packet delivery ratio: the ratio of the data packets delivered to the destination.
- Routing load: gives the number of routing packets over the number of received data packets. Each routing packet sent or forwarded by a mobile is counted.
- Average end-to-end delay: The end-to-end delay is averaged over all surviving data packets form the sources to the destinations. It includes queuing delay and propagation delay.

6.1.3 Simulation Results Based on NS2

6.1.3.1 Link layer notification

The concept of *Link Layer Notification (LLN)* has been introduced in Section 5.5.1 and implemented in *um-olsr*. In NS2, when a node is trying to forward a packet to the next hop that has been out of its transmission range, its link layer will drop the packet because of *MAC_RET* (REtry Timeout). In the mean time, the link layer will give a feedback to the routing layer and inform the lost of the link.

To implement to LLN in NS2, a function called *mpolsr_mac_failed_callback* is defined. It simply calls the *mac_failed()* function of the MP-OLSR class. When the *mac_failed* function is called, it will drop the packet and remove the related link from the topology information base.

In this way, *mpolsr_mac_failed_callback* will be called in two different situations :

Parameter	Values
Scenario Setting	
Number of nodes	50
Simulation area	1000 ×1000
Simulation time	200 (s)
Wireless Channel	
Channel Type	WirelessChannel
Antenna Model	OmniAntenna
Radio-propagation model	TwoRayGround
Network Interface Type	WirelessPhy
Receiving Threshold	3.65262e-10
Radio Transmission Range	250 meters
Bandwidth	11e6
Link Layer	
MAC Type	802_11
SlotTime_	0.000050 (s)
SIFS_	0.000028 (s)
PreambleLength_	0 (no preamble)
PLCPHeaderLenght_	128 (bits)
DataRate_	11.0e6 (bps)
Outgoing Queue (IFQ)	CMUPriQueue
Maximum Packets in IFQ	50
Routing Protocol	
Number of paths	3
Cost Function	$f_p(c) = f_e(c) = 2c$
Application Layer	
Type of Services	CBR
Transmission Time of CBR	20 s
Number of CBR	30
Packet Size	512 Bytes
Packet Rate	10 packet/s (40Kbps)

Table 6.1: Parameters for NS2 simulation

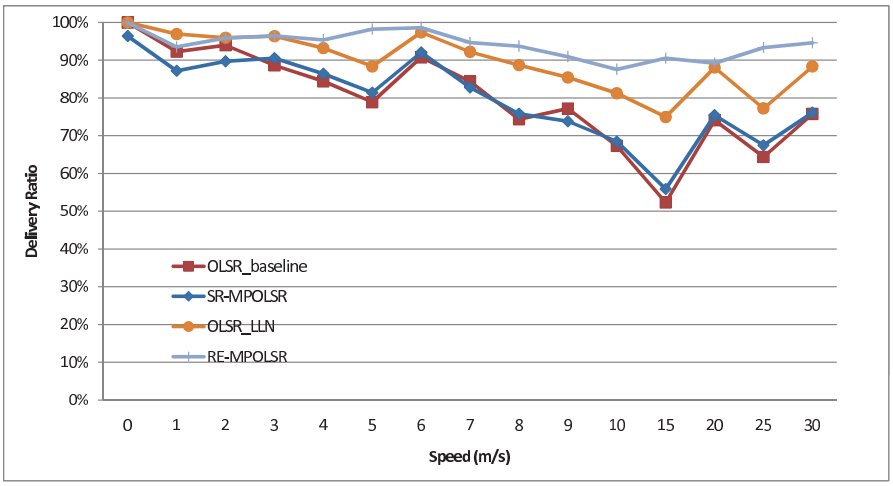


Figure 6.4: Delivery ratio of the NS2 simulations

- In ARP protocol, when a node wants to resolve a destination address failed.
- In MAC protocol, when a RTS is sent but maximum number of retries is exceeded or when a data packet was transmitted but never acknowledged.

The LLN is implemented in OLSR and MP-OLSR to compare the performance of the mechanism.

6.1.3.2 Route recovery

Route recovery is also implemented as described in Section 5.4.2 as the first auxiliary function of MP-OLSR. Before forwarding the packets to the next hop, the neighborhood is checked to validate the link to be used.

Thus, combined with LLN, there are four routing protocols studied with NS2:

- The baseline OLSR (without LLN).
- OLSR with LLN.
- SR-MPOLSR: MP-OLSR with LLN, use source routing only.
- RE-MPOLSR: MP-OLSR with route recovery and LLN.

Figure 6.4 shows the delivery ratio of the simulations. As we can see from the figure, the OLSR with feedback has better delivery ratio than the original OLSR. This is because the protocol with feedback could detect a link failure as soon as the first packet is dropped and recomputes the routing table. In contrast, the original OLSR tends to continue to send the packets through a failed link until it finds out that it is not able to receive a HELLO message from the original neighbor. Both with the link layer feedback, the SR-MPOLSR's delivery ratio is about ten percent lower than the OLSR. This is because of the drawbacks of source routing that have been mentioned in the previous section and compared with OLSR, which always sends the packets through the best route, the SR-MPOLSR sends the packets in multiple routes, some of which might be more unreliable. However, the RE-MPOLSR gives better delivery ratio thanks to the route recovery.

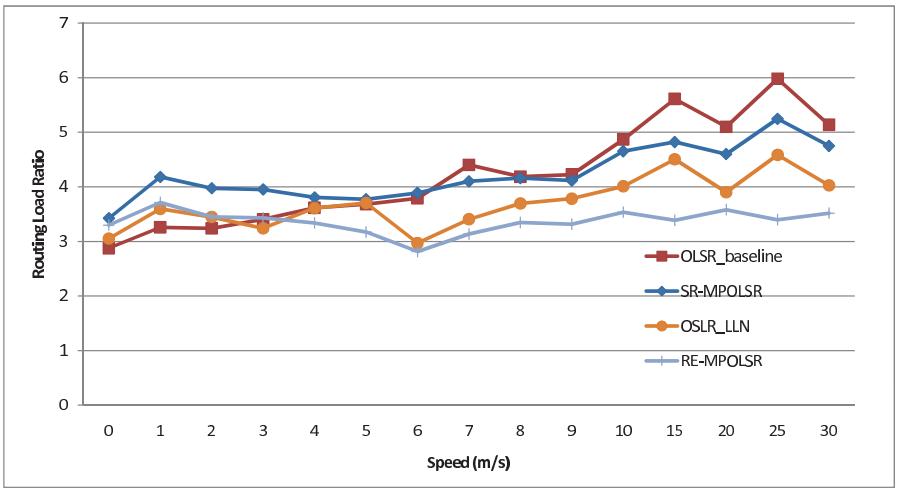


Figure 6.5: Routing load of the NS2 simulations

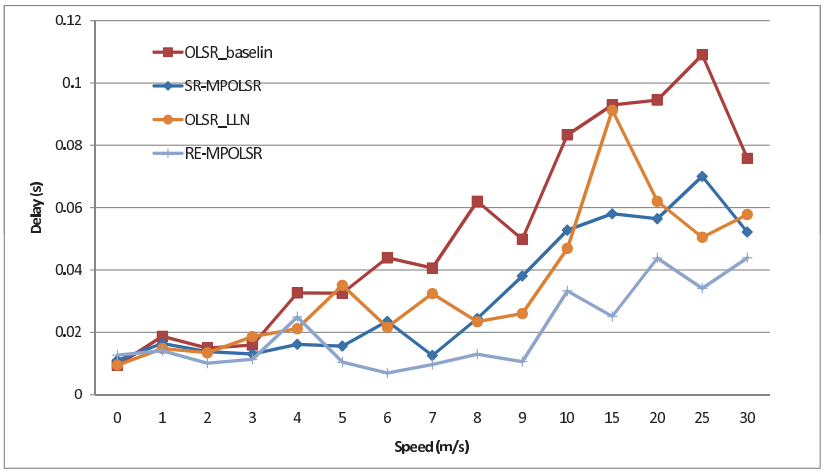


Figure 6.6: Average end-to-end delay of the NS2 simulations

Figure 6.5 gives the simulation results of routing load. Because in all the scenarios, the number of the generated control messages is the same, so the protocol with higher delivery ratio tends to have lower routing load.

Figure 6.6 shows the average end-to-end delay. It includes the queue delay in every node and the propagation delay from the source to the destination. The multipath routing could reduce the queue delay because the traffic is distributed in different routes. On the other hand, it might increase the propagation delay because some of the packets are sent through the sub-optimal route. As we can see from the figure, compared with the single path protocols, although the multipath routing might result in more propagation delay, it could effectively reduce the queue delay, and tends to have lower end-to-end delay. The figure also shows that although RE-MPOLSR might spend more time in recomputing and recovering the path, it still has the lowest delay because the recovery mechanism could avoid sending data packets through a failed route, which will result in more retry time and the network congestion. The multipath protocols also provide more stable end-to-end delay.

In this section, the simulation based on NS2 is performed. The results reveal that at low data rate, MP-OLSR has better performance than OLSR. LLN and route recovery are combined to improve the delivery ratio and delay of the protocol. The simulations with more nodes and higher data rate are also taken. However, we found that the scalability of NS2 was not satisfying because the simulation time is too long and it suffered strange queue behavior at high data rate (discussed in Appendix A).

To further study the routing protocol, another more advanced simulator, called *Qualnet* is employed. In the next section, the simulations related to *Qualnet* simulator will be presented.

6.2 Simulation Based on Qualnet

6.2.1 From NS2 to Qualnet

As discussed in Section 6, NS2 is one of the most popular simulators with various equipped models, protocols. And it is free and easy to obtain. So for the first year of our study, the NS2 simulator is used and some valuable results were obtained. However, as the research goes further, there are some problems found in NS2:

- The NS2 is a result of rather long process of development, in which numerous participators are included. So the software design is not very good with respect to current standards. A even more serious problem is that with all those different components, the documentation is quite limited and outdated. Most problems can only be solved by consulting the newsgroups and browsing the source code. So it is very hard to add new components and modify existing ones.

- The NS2 has a rather complex structure. It is not easy to add new components and modify existing ones concerning several software modules (C++ and OTcl) are included. Different layers in NS2 are not clearly defined, and the concept of *agent* doesn't exist in real networks. So it is not easy to study different scenarios at different levels of detail.

- NS2 lacks graphical tools to visualize the simulation and analyze simulation trace files.

- NS2 lacks scalability given hundreds of nodes and high data rate traffic. Some strange behavior happens such as the queuing problem stated in Annex A.

- On the aspect of routing protocols, only OLSRv1 is implemented in NS2. The implementation of the latest OLSRv2 is not found in NS2.

So beginning from the second year of the research, we change the simulation platform from NS2 to Qualnet. Compared to the NS2 simulator, Qualnet has well-defined layer structure and extensive set of pre-built models in different layers. It has rather good and highly modular software design and excellent documents. An advanced graphical and mathematical tool for experiment building, monitoring and post-processing are included. It also provides good scalability and a realistic 3D model of the environment.

In this section, the implementation and simulation related to Qualnet is presented.

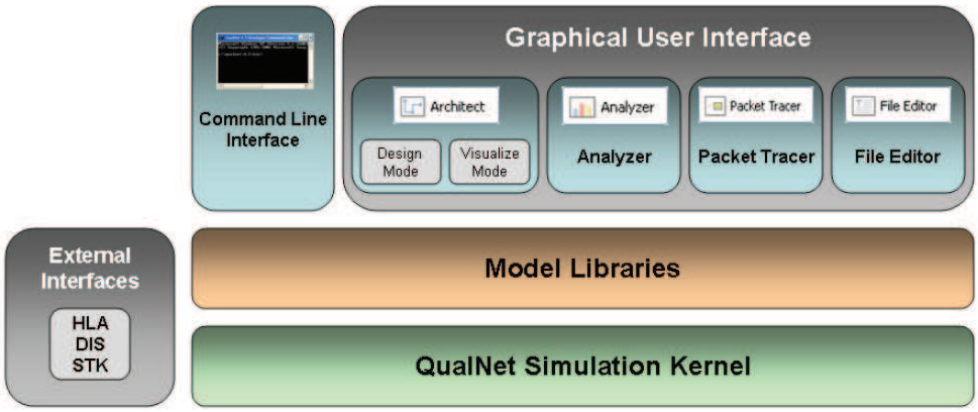


Figure 6.7: Qualnet Architecture

6.2.2 Introduction of Qualnet simulator

The Qualnet simulator is the commercial version of GloMoSim developed by Scalable Network Technologies. However, it alleviates most of the GloMoSim's flaws, such as poor documentation, lack of tools to monitor the system behavior, analyze the results, etc.

QualNet comes with an extensive suite of faithful implementations of models and protocols for both wired and wireless networks (local, ad hoc, satellite and cellular), as well as extensive documentation and technical support. A lot of libraries are already implemented in Qualnet, including *Multimedia and enterprise, Network security, Satellite, Sensor networks, UMTS,* etc (SNT, 2009a).

It uses simulation and emulation to predict the behavior and performance of networks to improve their design, operation and management. With the tools and libraries provided, it enables users to:

- Design new protocol models.
- Optimize new and existing models.
- Design large wired and wireless networks using pre-configured or user-designed models.
- Analyze the performance of networks and perform what-if analysis to optimize them.

6.2.2.1 Structure of *Qualnet*

Figure 6.7 illustrates the structure of Qualnet (SNT, 2009c). It includes a simulation kernel, model libraries, graphical & command line interfaces, and some external interfaces.

- Simulation Kernel. It is a parallel discrete-event scheduler which provides the scalability and portability to run hundreds of nodes on a variety of platforms. The source code is not opened, and the users do not directly interact with the kernel, but use the Qualnet API to develop protocol models.
- Model Libraries. QualNet includes support for a number of model libraries that enable you to design networks using SNT developed protocol models. The source code for this part is usually available and users can modify the related protocols and add new libraries for Qualnet.

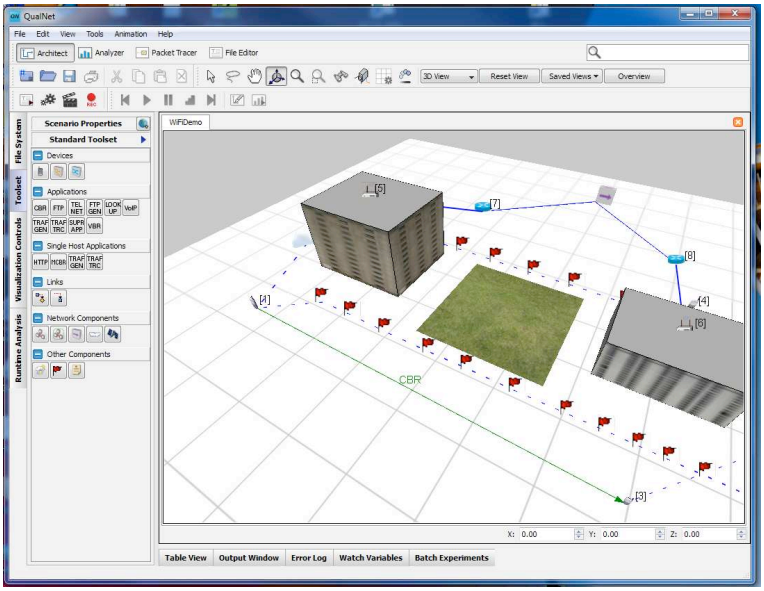


Figure 6.8: The Qualnet GUI

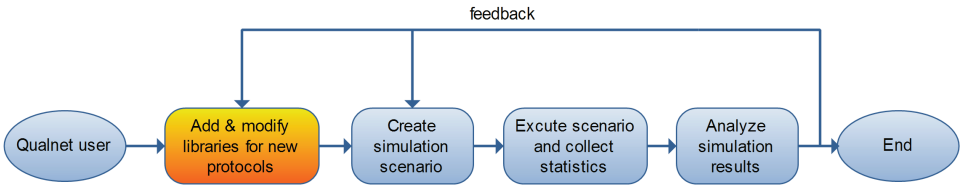


Figure 6.9: Qualnet simulation procedure

- Graphical User Interface (GUI) includes different components (Figure 6.8). *Architect* is a network design and visualization tool; *Analyzer* is a statistical graphing tool that displays the metrics collected during the simulation of a network scenario in a graphical format; *Packet Tracer* provides a visual representation of packet trace files generated during the simulation of a network scenario; *File Editor* is a text editing tool that displays the contents of the selected file in text format and allows the user to edit files.

- Command Line Interface enables a user to run QualNet from a command window. When QualNet is run from the command line, input to QualNet is in the form of text files which can be created and modified using any text editor.

- External Interfaces are also provided to interact with external tools in real-time.

6.2.2.2 Simulation procedure of *Qualnet*

Generally, the simulation study based on Qualnet follows the procedure shown in Figure 6.9.

- The first step is to add or modify the libraries for new protocols. It can be the protocol in any layer of the IP reference model: physical layer, data link layer, network layer, etc. The new protocol needs to be integrated into Qualnet and the program has to be recompiled.

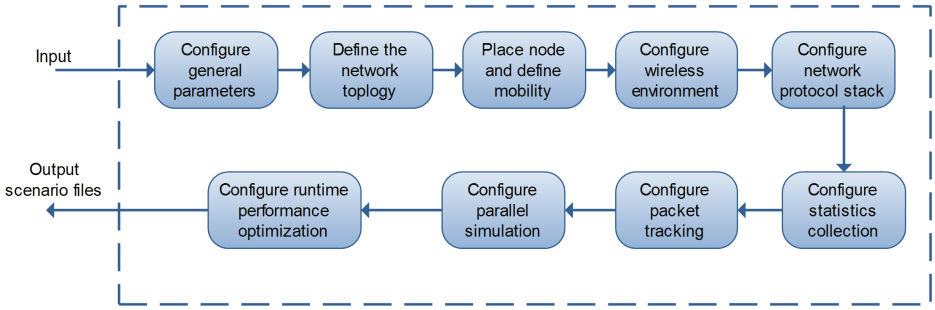


Figure 6.10: Qualnet scenario creation

- The second step is to create and prepare the simulation scenario based on the system description and metrics of interest. Figure 6.10 illustrates the detailed step for creating a simulation scenario.

- The next step is to execute, visualize, and analyze the created scenario and collect simulation results. Simulation results can include scenario animations, runtime statistics, final statistics, and output traces.

- The last phase is to analyze the simulation results. The researchers can make use of the Qualnet Analyzer or make their own program to obtain the statistics not collected by Qualnet. Based on the conclusion of the results, the researchers can adjust the protocol or simulation parameters and relaunch the simulation.

6.2.3 Environment and Assumption

6.2.3.1 Implementation of MP-OLSR in *Qualnet*

Qualnet uses a layered architecture similar to that of the TCP/IP network protocol stack (SNT, 2009b). Within that architecture, data moves between adjacent layers. Qualnet's protocol stack consists of, from top to bottom, the Application, Transport, Network, Link (MAC) and Physical Layers, as shown in Figure 6.11.

The functions of each layers are as defined in Section 1.1.1.2. But some of the routing protocols in Qualnet reside at the application layer because they need to use UDP or TCP services.

The OLSR protocol is an application-layer routing protocol. In Qualnet, OLSR uses UDP to transmit the TC and HELLO messages, and uses an IP kernel function provided by Qualnet to update the IP forwarding table. So for the protocol itself, it does not receive data packet to forward, but IP protocol handles the packet forwarding.

MP-OLSR is also an application-layer routing protocol based on OLSRv2, but it needs to receive the data packet to read/modify the source routing packet header. To enable the application layer routing protocol MP-OLSR can handle the data packet, an *MPOLSRRouterFunction* is defined as shown in Algorithm 6.2. And *NetworkIpSetRouterFunction* is used to register *MPOLSRRouterFunction*. This enables IP to directly call *MPOLSRRouterFunction* to determine the route for a packet if MP-OLSR is running at that interface.

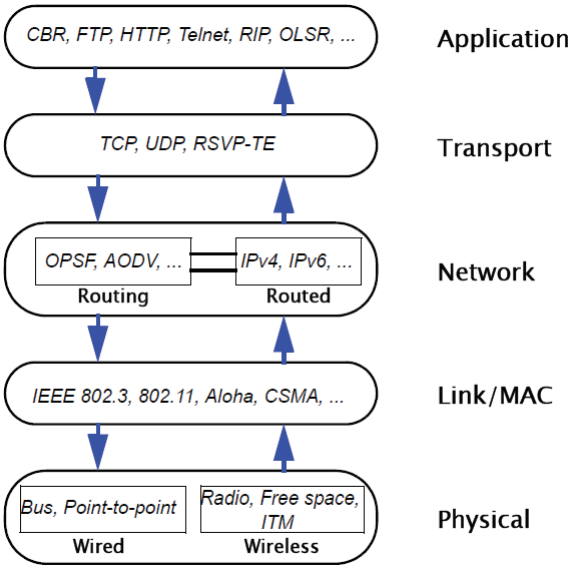


Figure 6.11: Qualnet Protocol Stack

Algorithm 6.2 Handle data packets at the application layer for MP-OLSR

```
void MPOLSRRouterFunction(
        Node* node,
        Message* msg,
        NodeAddress destAddr,
        NodeAddress previousHopAddress,
        BOOL* packetWasRouted){

        ...
// PURPOSE:  Determine routing action to take for a given data packet
//         1. for new data packets:
//                      calculate the source route and add into
//                      the head of the packet.
//         2. for data packets to be forwarded:
//                      read the next hop in the source route
//                      and set the nexthop route recovery
//         3. for data got destination: forward to the transport layer
}

void RoutingMPOLSRv2_Niigata_Init(Node *node,
                const NodeInput* nodeInput,
        int interfaceIndex,
        NetworkType networkType){
        ...
          // Set network router function
        NetworkIpSetRouterFunction(node,
                &MPOLSRRouterFunction,
                interfaceIndex);
        ...
}
```

6.2.3.2 Simulation Parameters

The simulation is taken an area of 1500m by 1500m. The nodes will move at certain speed according to the *Random Way Point Model* (RWP) to simulate pedestrian and cycle applications.

For each simulation, the total simulation time is 100 seconds. A simple CBR (Constant Bit Rate) UDP application runs at several nodes to measure the performance of data transmission. Each CBR application corresponds to a source-destination flow which generates UDP packets of 512 bytes at the source, at different rates. The flow starts 15s after the simulation began, to allow enough exchange of the routing messages. The data transmission lasts for 80s to have an average behavior of each flow.

The 802.11b radio is used and the data rate is set to 11Mbps. We use the two-ray ground pathloss model, constant shadowing model with shadowing mean of 4.0 dB, and the transmission power is set to 15dBm. With these settings, the transmission distance is about 270 meters in our simulations. We repeat each simulation 100 times and give the average results. Different random seeds are used to have different scenarios. Different seeds will generate different pseudo-random sequences used in simulation. This will effect the mobility according to the mobility model, back off timers, the interference pattern, etc. The detailed parameters are listed in Table 6.2 for the purpose of repeatability, which is widely used in WiFi devices and simulation study.

The parameters for OLSRv2 and MP-OLSR are presented in Table 6.3. For the multipath routing, the Round- Robin packet-distribution scheme is used.

6.2.3.3 Performance Metrics

In addition to the metrics used in section 6.1.2.1, the following metrics are considered:

- Average Time in FIFO Queue: average time spent by packets in the queue.
- Average Jitter: the mean deviation (smoothed absolute value) of the difference D in packet spacing at the receiver compared to the sender. If S_i is the RTP timestamp from packet i, and R_i is the time of arrival in RTP timestamp units for packet i, then for two packets i and j, D may be expressed as

$$D(i,j) = (R_j - S_j) - (R_i - S_i) \tag{6.2}$$

  The interarrival jitter is calculated continuously as each data packet i is received from the source, using this difference D for that packet and the previous packet $i-1$ in order of arrival (not necessarily in sequence), according to the formula

$$J(i) = J(i-1) + (|D(i-1,i)| - J(i-1))/16 \tag{6.3}$$

  This algorithm is the optimal first-order estimator and the gain parameter $1/16$ gives a good noise reduction ratio while maintaining a reasonable rate of convergence (Schulzrinne et al., 2003).

6.2.4 Simulation Results Based on Qualnet

6.2.4.1 Route Recovery and Loop Detection

In this part, the auxiliary functionalities are simulated to see the effects of the *Route Recovery* and *Loop Detection* on the performance of the network. There are 4 CBR sources in the 81 nodes

Parameter	Values
Simulator	Qualnet 5.0
Routing Protocol	OLSRv2 and MP-OLSR
Simulation area	1500m × 1500m
Mobility	RWP, max speed 0-10m/s
Simulation Time	100 seconds
Applications	CBR
Application Packet size	512 bytes
Transmission Interval	0.1 s
CBR start-end	15s - 95s
Transport Protocol	UDP
Network Protocol	IPv4
IP Fragmentation Unit	2048
Priority Input Queue Size	50000
MAC Protocol	IEEE 802.11
MAC Propagation delay	1uS
Short Packet Transmit Limit	7
Long Packet Transmit Limit	4
Rtx Threshold	0
Physical Layer Model	PHY 802.11b
Wireless Channel Frequency	2.4 GHz
Propagation Limit	-111.0 dBm
Pathloss Model	Two Ray Ground
Shadowing Model	Constant
Shadowing Mean	4.0 dB
Transmission Range	270m
Temperature	290K
Noise Factor	10.0
Receive Sensitivity	-83.0
Transmission Power	15.0dBm
Data Rate	11Mbps

Table 6.2: Qualnet Simulator Parameter Set

Parameter	Values
TC Interval	5s
HELLO Interval	2s
Refresh Timeout Interval	2s
Neighbor Hold Time	6s
Topology Hold Time	15s
Duplicate Hold Time	30s
Link Layer Notification	Yes
No. of path in MP-OLSR	3
MP-OLSR f_e	$f_e(c) = 2c$
MP-OLSR f_p	$f_p(c) = 3c$

Table 6.3: OLSR and MP-OLSR Parameters

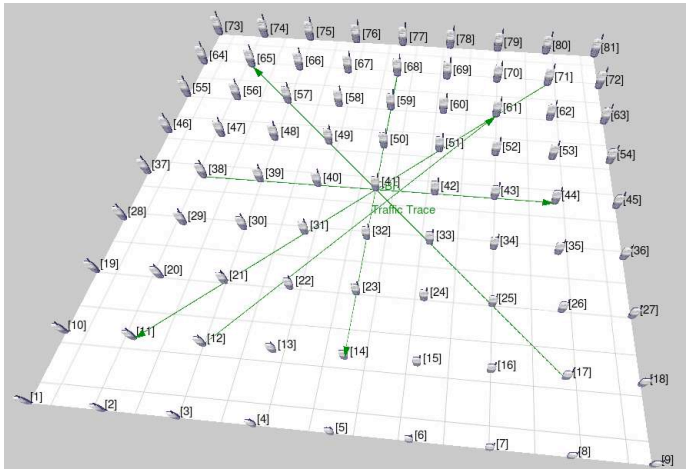


Figure 6.12: The simulation topology with 81 nodes

network. The initial position of the nodes is as shown Figure 6.12.

Here, three different MP-OLSR protocols are compared:

- MP-OLSR without Route Recovery and Loop Detection,
- MP-OLSR with Route Recovery but without Loop Detection,
- MP-OLSR with Route Recovery and Loop Detection.

Figure 6.13 shows the comparison of the delivery ratio. The delivery ratio of the protocol without route recovery is very poor and very sensitive to the mobility of the nodes. This is because of the drawbacks of the pure source routing based on OLSR that we have mentioned in section 3.3: because of the delay of the transmission of the routing control packets (especially the TC messages), the source node find it very hard to get the most updated topology information when it is constructing the source route. Therefore a node transmits a packet through a link that does not exist anymore. This phenomenon is more serious when the speed of the nodes increases. In fact, the SR-MPOLSR (Zhou et al., 2005) also has very low delivery ratio like MP-OLSR without route recovery in our settings. This means that the pure source routing based on OLSR is not adapted.

From Figure 6.13 we can also find that the MP-OLSR protocol with loop detection has a slightly better delivery ratio than the one without loop detection. This is because with the mechanism of the loop detection, there are more possibilities to avoid the loop by switching to another path. When the loop detection is not applied, more following packets will be involved in the loop and be dropped because of the TTL (time-to-live) count to zero. Figure 6.14 presents the number of packets dropped because the TTL comes to 0 (TTL is set to 64 in our case). Given the size of the simulation area (1500×1500) and the transmission range of the node (270m), the number of hops is lower than 10 in most cases. So when a packet is dropped because TTL counts to 0, it can be assumed that a loop exists in our simulation.

In fact, compared to the effect on the delivery ratio, the loop detection has more influence on the average end-to-end delay. Figure 6.15 shows the delay of the different protocols. It is essential to mention that only the packets that successfully reached to the destination are taken into account. So with or without route recovery does not affect the delay very much and gives similar results. However, the protocol without loop detection has much longer delay than the others because loops can easily congest the network.

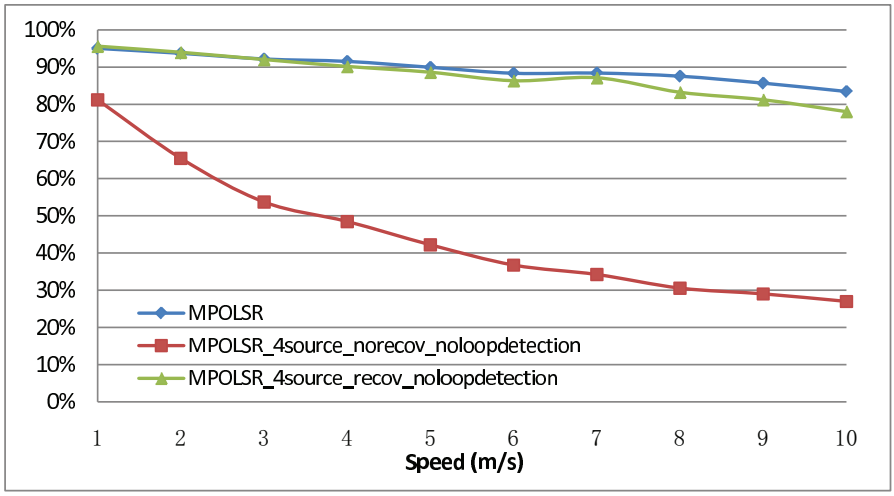


Figure 6.13: Delivery Ratio of MP-OLSR implementations with or without route recovery and loop detection

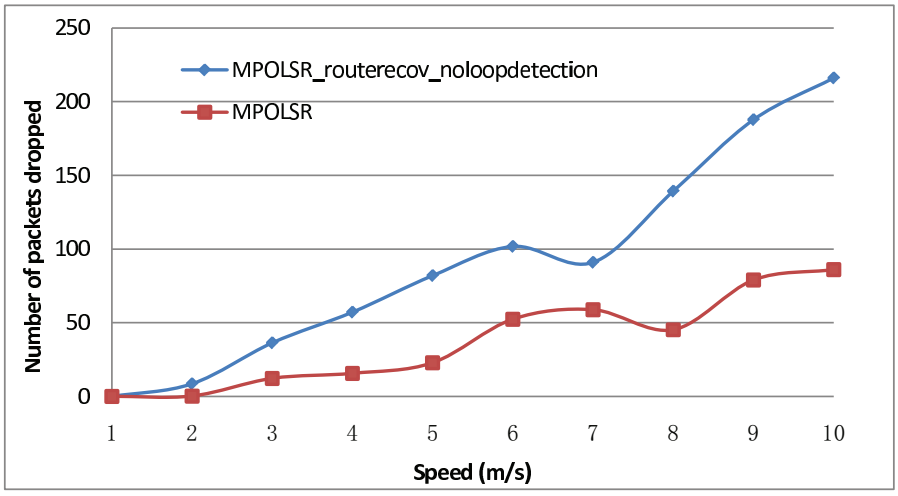


Figure 6.14: Number of packets dropped because the TTL comes to 0

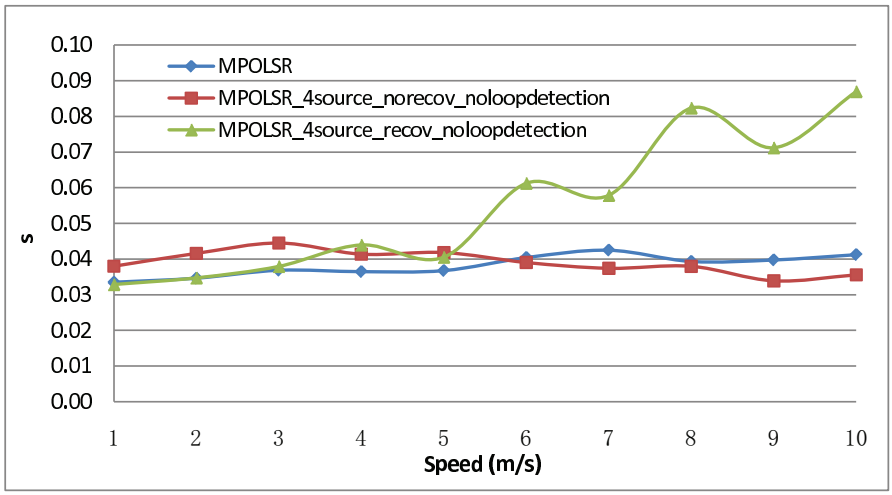


Figure 6.15: Average end-to-end delay of MP-OLSR implementations with or without route recovery and loop detection

As we can see from this subsection, because of the change of the topology in ad hoc networks, only the multipath pure source routing based on OLSR cannot provide better performances because of the mobility of the network. So the *route recovery* and *loop detection* are proposed as additional functionalities. The study illustrates the importance of the two functionalities for MP-OLSR. The simulation results show that they can effectively improve the data delivery ratio and the end-to-end delay respectively. The trend is clearer with higher mobility. We can expect more gain with the auxiliary functionalities with even higher speed.

6.2.4.2 Two-path scenario

A simplified static 2-path scenario is set up to study the multipath mechanism. The topology is as shown in Figure 6.16. In this scenario, node 1 is the source and node 2 is the destination. There are two possible paths from node 1 to node 2: 1→3→4→5→6→7→2, which has 6 hops and 1→8→9→10→11→12→13→2, which has 7 hops. Because MP-OLSR and OLSR protocols tend to have similar performances under low network load, we saturate the network by using high data rate to see the difference between the protocols. In this scenario, we begin with 100 packets/s (400kbps), and up to 200 packets/s (800kbps).

Figure 6.17 is the packet delivery ratio with different data rate. This figure shows that at low network load (100 packets/s), both of the protocols have almost 100% packet delivery ratio. When the network is saturated by high data rate (more than 120 packets/s), the packet loss increases, but MP-OLSR still outperforms OLSR (double of the delivery ratio at high rate). From Figure 6.18, we can conclude that the delay increases very quickly when the network is congested (from 140 packets/s). Because only the packets that successfully reached the destination are accounted for when we are computing the end-to-end delay, it is insignificant to conclude that OLSR has better delay than MP-OLSR given the delivery ratio of OLSR is much lower (in the test of more than 140 packets/s). The performance of the delay will be compared in the rest of the section.

To study the difference in the delivery ratio, we choose the scenario of 120 packets/s (when the network begins to saturate) and analyzed the queuing information in every node. Figure 6.19 presents the number of total packets dequeued in every node. It is the number of packets forwarded

Figure 6.16: Topology of the 2 path scenario, node 1 is the source and node 2 is the destination. Two paths are possible: 1→3→4→5→6→7→2 (6 hops) and 1→8→9→10→11→12→13→2 (7 hops)

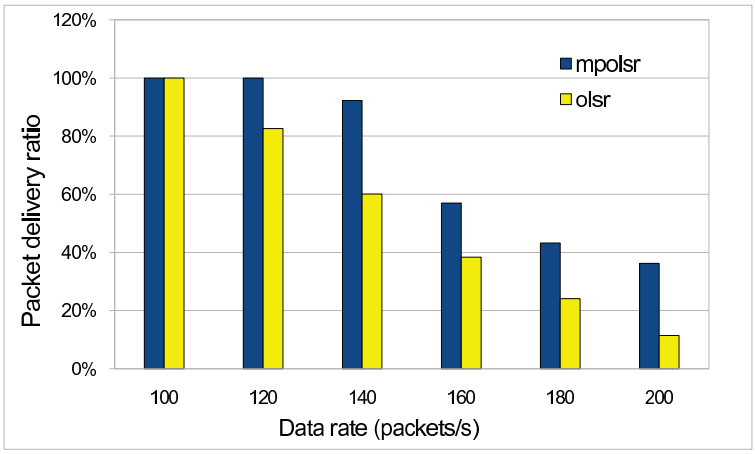


Figure 6.17: Delivery ratio of the 2 path scenario with different data rates

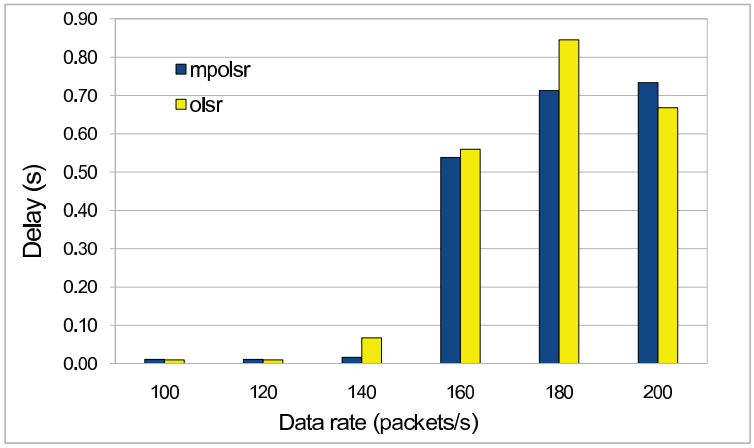


Figure 6.18: Delay of the 2 path scenario with different data rates

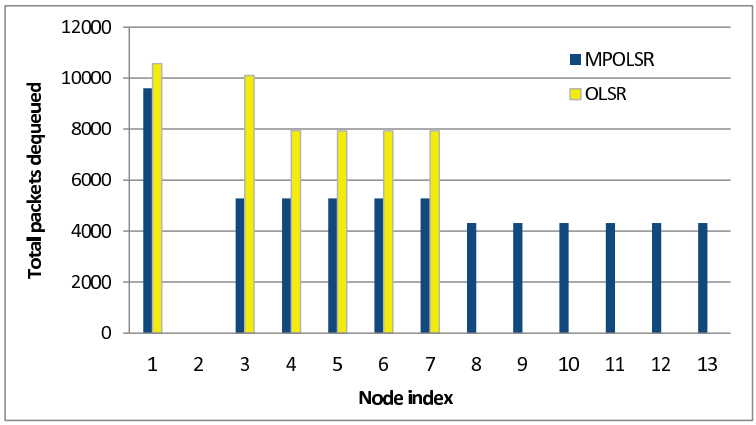


Figure 6.19: Total packets dequeued in each node in a 2 path scenario with 120 packets/s

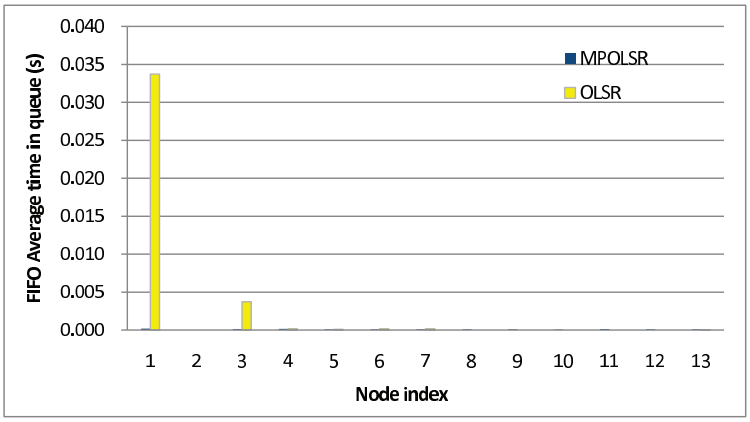


Figure 6.20: Average time in queue in each node in a 2 path scenario with 120 packets/s

by the nodes. As we can see, the OLSR sent all the packets along path 1→3→4→5→6→7→2. And for MP-OLSR, the packets were distributed along the two paths.

The average time in queue is given in Figure 6.20. We can conclude from the figure that there are congestion in node 1 and node 3 because they have much longer average time in queue. The time in queue for MP-OLSR is much lower than OLSR (close to 0 for MP-OLSR and 0.03s for OLSR in node 1).

The average jitter of the two protocols is given in Figure 6.21. The MP-OLSR has a little higher average jitter in this static scenario because it sends packets through different paths. So it is more difficult to make sure the packets arrive in order. The jitter in the mobile scenarios will be compared in the rest of the section.

From this simple scenario we can conclude that the network congestion can be released, and the packet delivery ratio can be improved by using multipath routing, especially in the scenarios of high network load. In fact, the routing protocol can take more benefits from the multipath algorithm, which will be discussed in the rest of this section.

6.2.4.3 81 nodes, 4 sources scenario

This scenario retains the same configuration as the one in Section 6.2.4.1, which has 81 nodes and 4 sources.

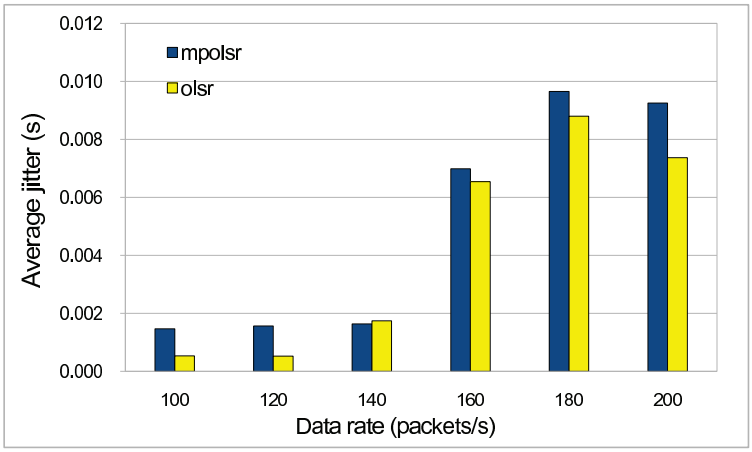


Figure 6.21: Average jitter of the 2 path scenario with different data rates

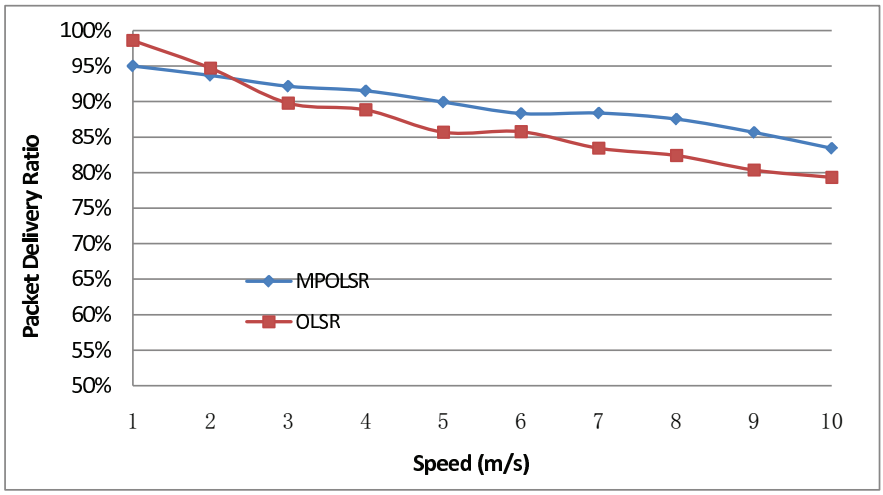


Figure 6.22: Delivery ratio of MP-OLSR and OLSR in a scenario of 81 nodes and 4 CBR sources

In Figure 6.22, the data delivery ratio of the two protocols is given. Only at the speed of 1m/s (3.6km/s), OLSR has a slightly better delivery ratio (about 3%) than the MP-OLSR. This is because given more paths transmitting packets at the same time, there is also a higher possibility of a collision at the MAC layer. This inter path interference can be eliminated by using multichannel techniques, which guarantees a different frequency band for each path (Gharavi, 2008). In our case, there is only one channel used, so MP-OLSR has more packets dropped due to the collision at the MAC layer. However, as the speed of the nodes increases, the links become more unstable, and there are also more loops in the network. The delivery ratio of OLSR then decreases quickly and MP-OLSR outperforms OLSR.

Compared to the slight gain in the delivery ratio (about 5% at high speed), the multipath protocol performs much better on average end-to-end delay than the single path, as shown in Figure 6.23. The delay of OLSR is about 4 times more than MP-OLSR beginning from 4m/s (14.4km/h). The end-to-end delay includes the propagation delay from the source to the destination and the queue delay in every relay nodes. The multipath protocol might have a longer propagation delay because some of the packets are forwarded through longer paths. However, the most important is

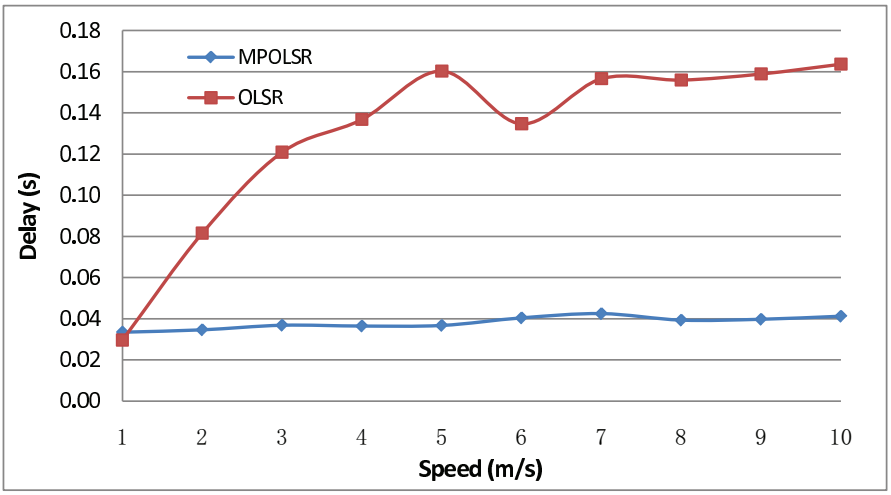


Figure 6.23: Average end-to-end delay of MP-OLSR and OLSR in a scenario fo 81 nodes and 4 CBR sources

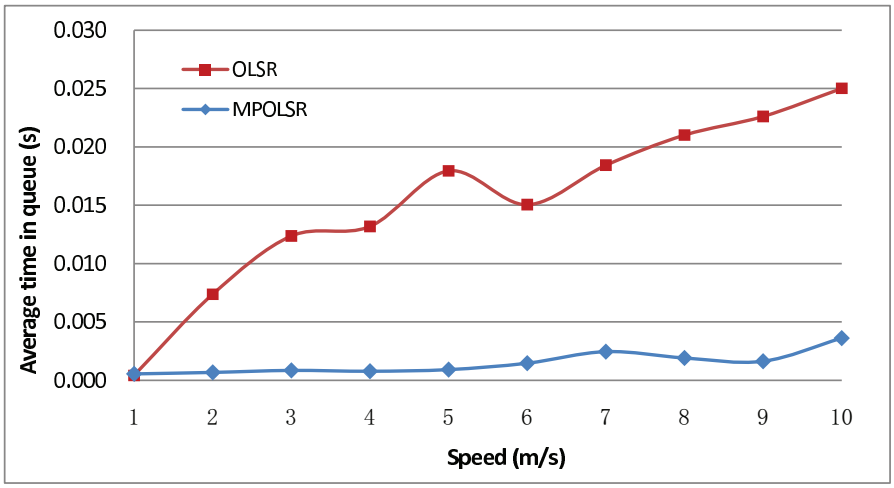


Figure 6.24: Average time in queue of MP-OLSR and OLSR in a scenario of 81 nodes, 4 CBR sources

that it can effectively reduce the queue delay by distributing the packets to different paths rather than to a single one. In addition, the loop detection mechanism can also reduce the unnecessary transmissions by avoiding the loops. As shown in Figure 6.24, the MP-OLSR has much less average time in the queue compared with OLSR. In our simulations, the MP-OLSR also offers more stable delay in different simulations. The standard deviation of the delay of OLSR is at least 10 times more than MP-OLSR, sometimes up to 100 times more in the mobile scenarios. For the purpose of visibility, the error line of standard deviation is not plotted in the figure.

The average jitter is also measured. It shares the same trend as the delay in Figure 6.23. In fact, we are not only interested in the average jitter (which is an accumulation of difference *D(i, i−1)* in packet spacing), but also the instant feature of $D(i, i-1)$. So we choose a scenario with medium mobility of 0-5 m/s and monitor the $D(i, i-1)$ of a CBR flow during the simulation as shown in Figure 6.25. $D(i, i-1)$ equals 0 means there is no difference in packet spacing (ideal case) and the greater absolute value indicates longer jitter during the transmission. For both OLSR and

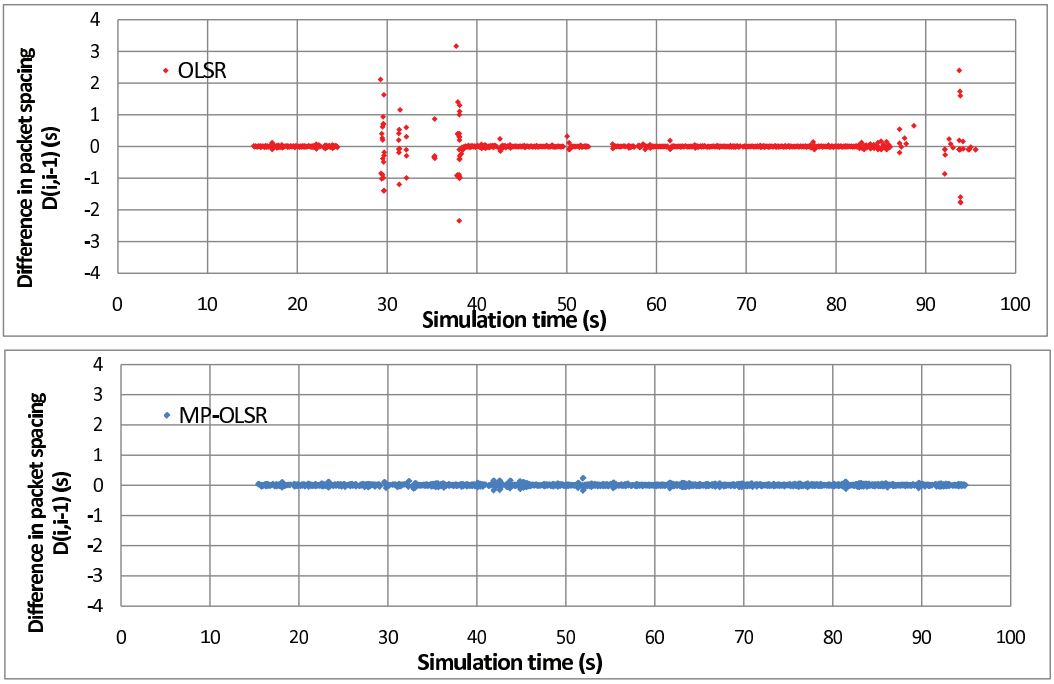


Figure 6.25: Packet spacing difference $D(i, i-1)$ of a CBR flow of MPOLSR and OLSR in a scenario of 81 nodes, 4 CBR sources

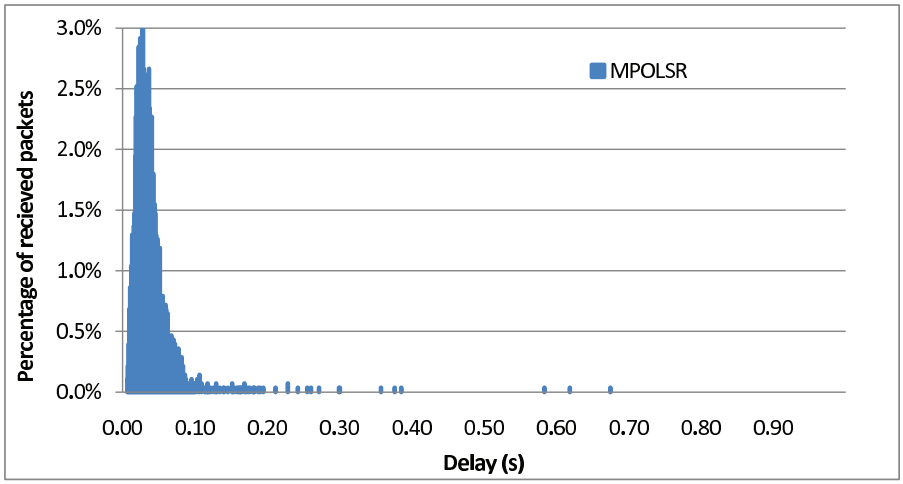
MP-OLSR, there is no data transmission at the first 15s for network initialization. The OLSR flow is interrupted from 25s to 40s and from 85s to 95s and suffers from big jitter. This is because the failure of the only path in OLSR has a serious affect on the stability of the transmission. Compared to OLSR, the MP-OLSR protocol has much shorter jitter during all the transmission.

Figures 6.26 show the distribution of end-to-end delay of all received packets in a scenario with medium mobility (0-5m/s). The distribution of the delay of OLSR spreads more widely compared to MP-OLSR. In this case, in the 2731 packets received by using OLSR, 1967 packets (82.96%) are received with delay less than 0.1s. For MP-OLSR, in the 2776 packets received, 2712 packets (97.69%) reach the destination in 0.1s. In fact, the standard deviation of the delay of OLSR is at least 10 times more than that of MP-OLSR, and even sometimes up to 100 times more in the high mobile scenarios.

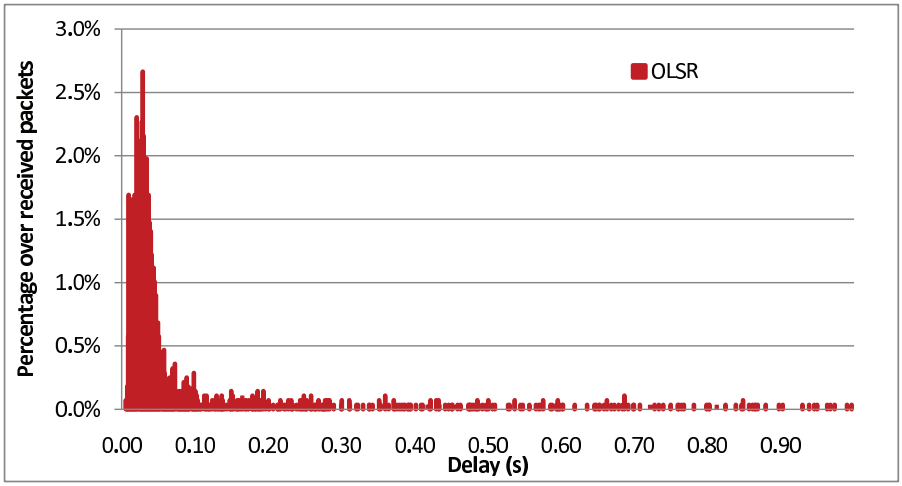
For the routing control message, because the MPOLSR does not change the topology sensing mechanism of the OLSR protocol, the two protocols tend to have the same number of routing control messages generated. In our simulation scenarios, the number of HELLO messages and TC messages generated is almost the same, as verified in figure 6.27. So MP-OLSR will not introduce much extra routing traffic compared to OLSR.

Because MP-OLSR is a hybrid routing protocol and uses source routing, we are also interested in the difference between proactive routing and reactive routing. The performance of DSR and AODV is also measured.

Figures 6.28 and 6.29 present the delivery ratio and delay of DSR and AODV respectively (compared to OLSR and MP-OLSR). The AODV protocol has much lower delivery ratio than others in this 81-node scenario. The DSR has almost the same performance as MP-OLSR at low



(a) MP-OLSR



(b) OLSR

Figure 6.26: Distribution of delay of received packets in a scenario of 81 nodes, 4 CBR sources

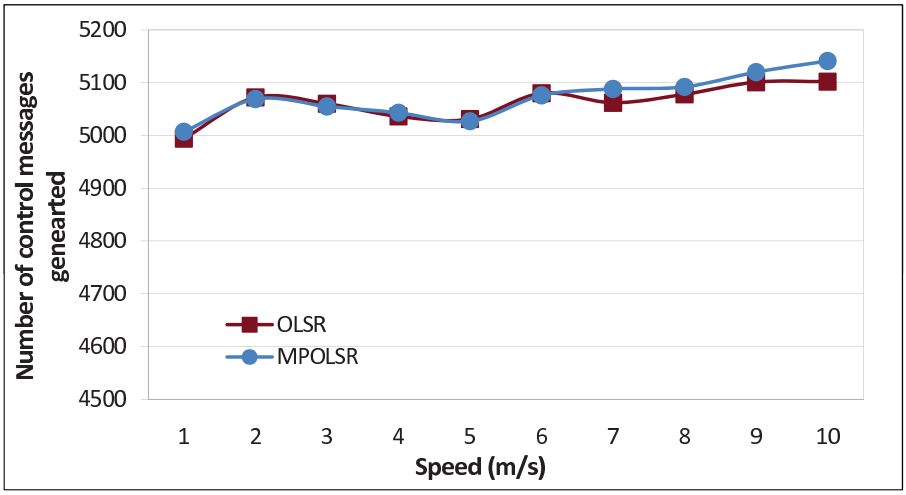


Figure 6.27: Total control messages (HELLO and TC) generated by MP-OLSR and OLSR in a scenario of 81 nodes, 4 CBR sources

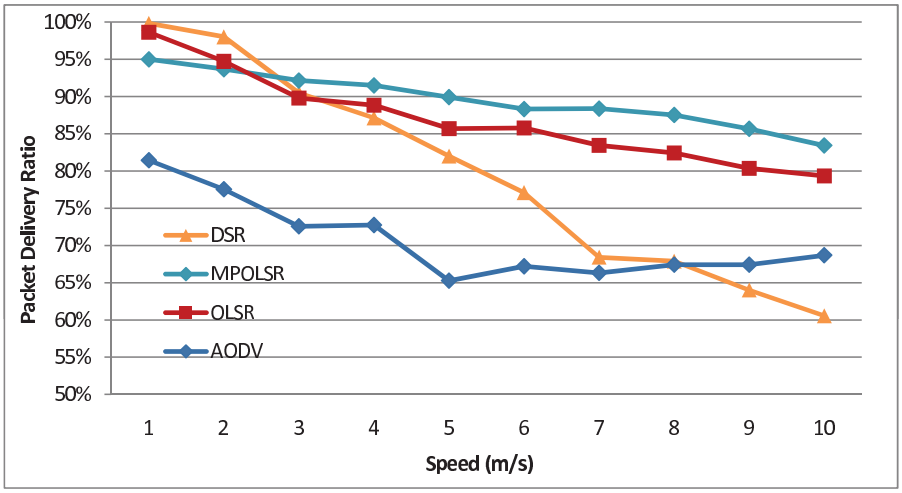


Figure 6.28: Delivery Ratio of AODV and DSR in a scenario of 81 nodes and 4 CBR sources

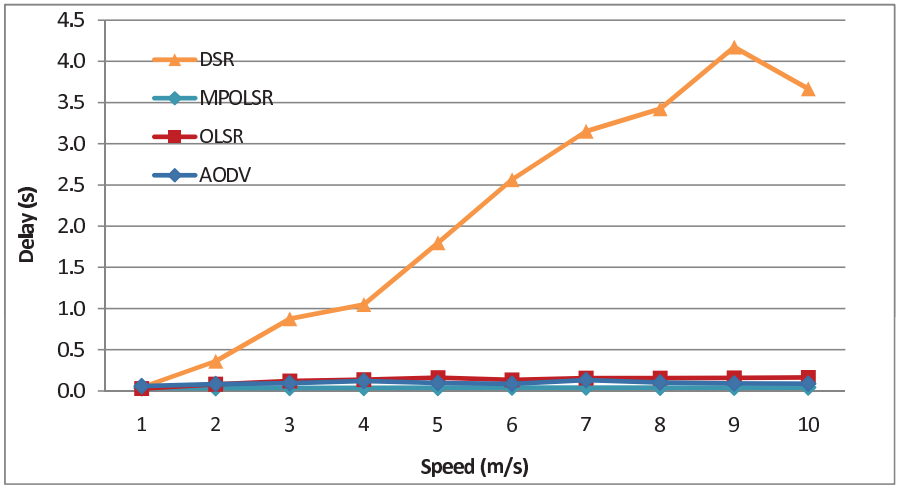


Figure 6.29: Average end-to-end elay of AODV and DSR in a scenario of 81 nodes and 4 CBR sources

node speed (1m/s and 2m/s). However, when the mobility increases, the packet loss and delay of DSR increase significantly. In fact, the DSR uses source routing like MP-OLSR and also has a corresponding route recovery mechanism. But the reactive nature of DSR cannot adapt to frequent topology changes. Its route recovery mechanism is based on transmission of explicit RERR (route error) messages, which will increase rapidly as the node speed rises. On the other hand, the route recovery of MP-OLSR, which is based on link layer feedback and local network topology information base, does not need extra packet transmission in the network.

There have been several multipath routing protocol proposed based on DSR, like SMR (Lee & Gerla, 2002), which relies on the same reactive mechanism. The reactive feature makes those protocols hard to compare with the proactive ones in the scenarios with frequent topology changes. According to the simulation result of SMR, even if it can reduce the delay to 1/5 of DSR, it is still much higher than the proactive protocols.

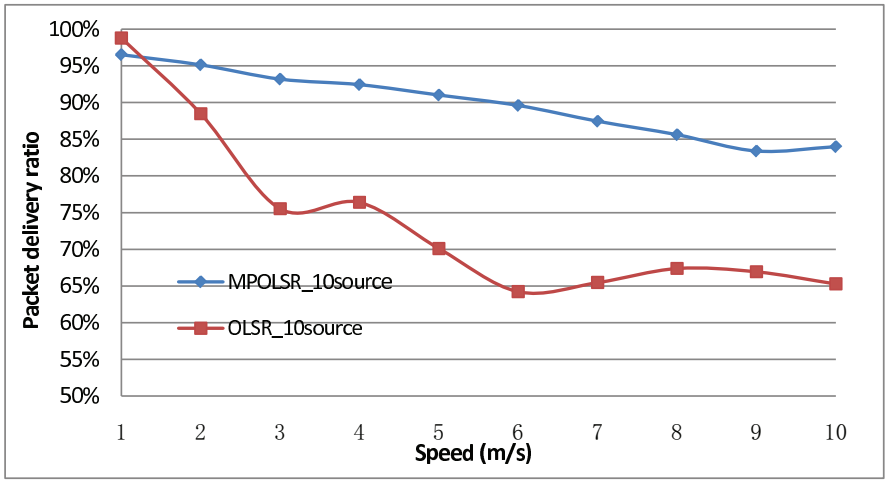


Figure 6.30: Delivery ratio of MP-OLSR and OLSR in a scenario of 81 nodes and 10 CBR sources

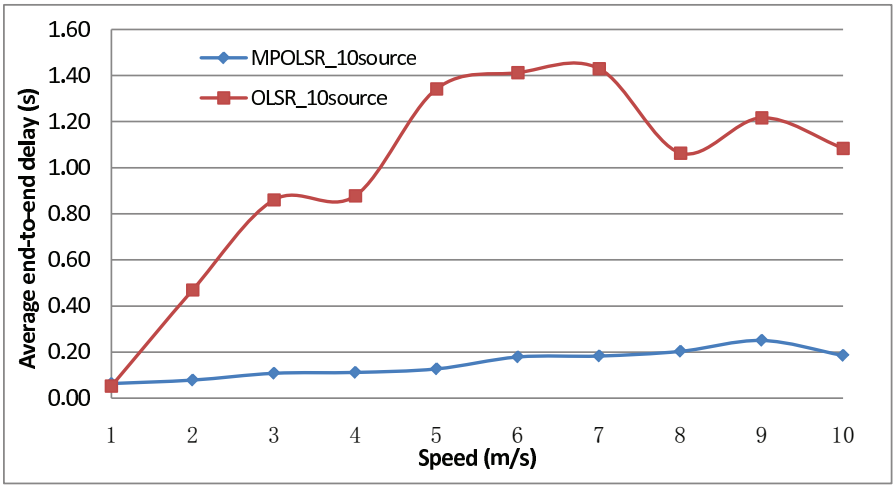


Figure 6.31: Average end-to-end delay of MP-OLSR and OLSR in a scenario of 81 nodes and 10 CBR sources

6.2.4.4 81 nodes, 10 sources scenario

In this scenario, a higher load is applied to the network. There are 10 CBR sources transmitting the data packets to the destination, instead of 4 in the previous scenario.

In Figure 10, we present the delivery ratio of the two protocols. Compared to Figure 5 of the 4-source scenario, whose delivery ratio is always more than 80%, the OLSR protocol is unstable with the high load. With a speed superior to 6m/s (21.6km/h), it drops to about 65%. The MP-OLSR is more robust with high load, and stays at about 85% even with high mobility.

Figure 6.31 shows the end-to-end delay. As we can see from the figure, the increase of the delay is much more significant than in Figure 6.23, which is more than 1 second at higher speed.

In addition to the scenarios presented in this section, we performed other scenarios like *duplex scenario* (two nodes send and receive packets from each other simultaneously, like VoIP application), *short-time scenario* (we have many more source-destination pairs, but very short transmission time, for an application like instant message sending). The results share the same trends and

features. The multipath protocol is more adapted to the mobile scenarios.

6.2.4.5 Queue Length Metric Simulation

In this section, the test for the queue length is performed. The MP-OLSR is simulated with different configurations:

- MP-OLSR with hop-count metric (Orig_MPOLSR);
- MP-OLSR with queue length metric, slope coefficient $\alpha = 315$ (MPOLSR_315), i.e. the link cost increases slowly with the queue length;
- MP-OLSR with queue length metric, slope coefficient $\alpha = 3200$ (MPOLSR_3200), i.e. the link cost increases quickly with the queue length and more punishment on the queue length;

6.2.4.5.1 Low Traffic Scenario Figure 6.32 and 6.33 compare the packet delivery ratio and delay in the low traffic scenarios (10 packets/s in the source). As we can see, when the traffic load is very low, there is no much difference in those three configurations. This is reasonable because the queue length monitored is 0 most of the time in both configurations. It also proves that the overhead of the *TLV_ Queue_ Length* will not affect the performance of the network.

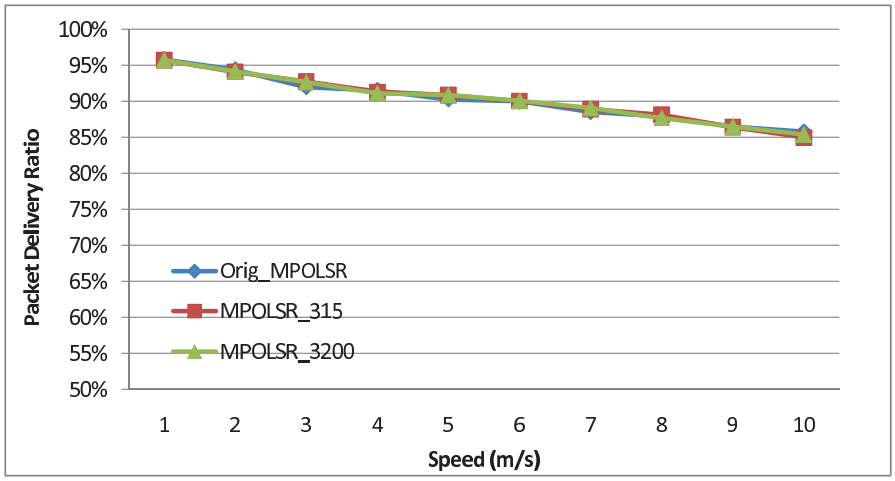


Figure 6.32: Packet delivery ratio of MP-OLSR with queue length metric in low traffic scenarios

6.2.4.5.2 High Traffic Scenario As shown in the previous simulations, when there is just normal low traffic applied to the network, there is no difference when queue length metric is used. So we try to overload the network by injecting higher data rate traffic at the source node. So the CBR traffics are increased from 10 packets/s (40kbit/s) to 30 packets/s (120kbit/s).

Figure 6.34 shows the packet delivery ratio. With low mobility (1m/s - 5m/s), the protocol using queue length metric can significantly improve the delivery ratio (5% by *MP-OLSR_ 315* and 10% by *MPOLSR_ 3200*). *MPOLSR_ 3200* has better performance than *MPOLSR_*315 because it applies more punishment on the queue length, i.e. the algorithm has greater possibility to avoid congested links. When the node speed increases, the improvement is less obvious (about 3%) because it is more difficult to get the updated queue length with high topology changes, given the HELLO message interval with 2 seconds and TC message interval with 5 seconds. However,

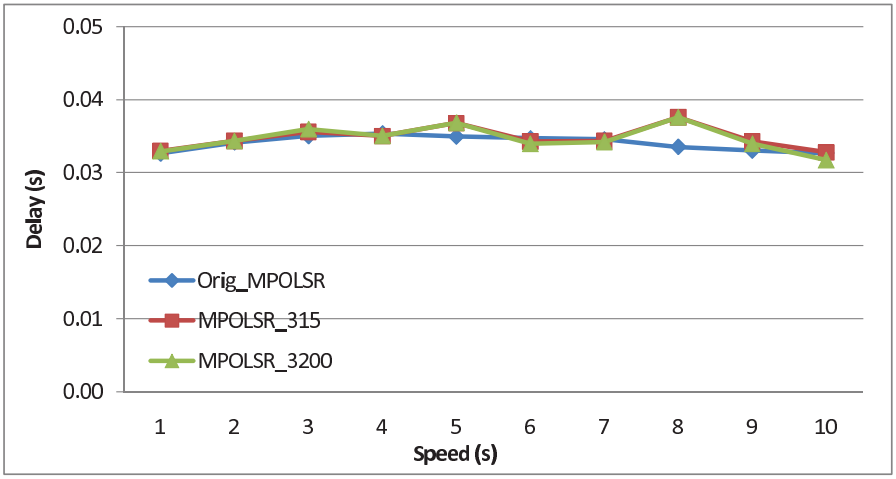


Figure 6.33: Average end-to-end delay of MP-OLSR with queue length metric in low traffic scenarios

the MP-OLSR using queue length metric can still provide better performance than the one using hop-count metric.

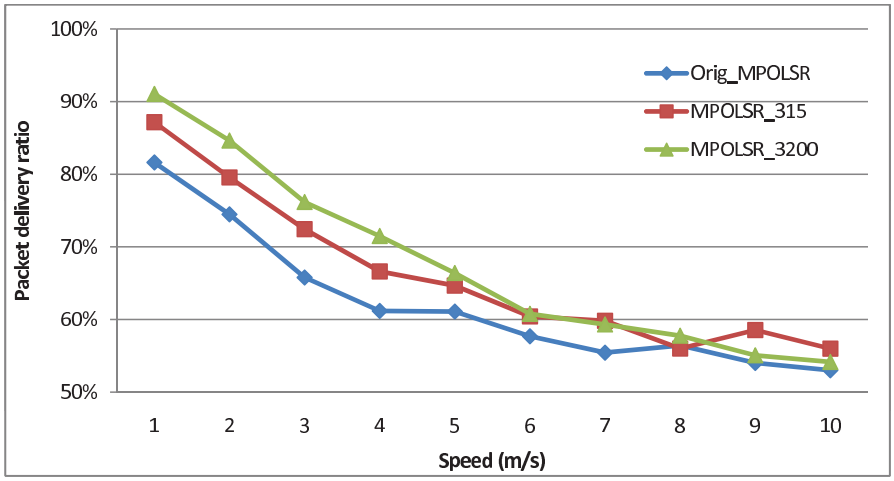


Figure 6.34: Packet delivery ratio of MP-OLSR with queue length metric in high traffic scenario

Figure 6.35 shows the average end-to-end delay. The configurations with queue length metric have slightly better results than the hop-count metric in speed range from 1 m/s to 4 m /s (about 6% less than *Orig_MPOLSR*). However, at high mobility (more than 5 m/s), the configurations with queue length metric has longer delay than the original MP-OLSR using hop count metric. This is because: firstly, when the queue length metric is applied, the algorithm might get paths with more hop counts (this will not be a critical issue if the measurement is correct); secondly, as mentioned in the previous paragraph, the queue length information might not be accurate enough to make the route decision because of the high mobility.

The overhead of the routing messages is also measured. Figure 6.36 shows the total bytes of routing messages in the network. The queue length metric will not increase the number of TC and HELLO message packets, but the *TLV_Queue_Length* adds an additional TLV with 4 bytes. The size of HELLO and TC messages is normally 40 bytes, so the overhead of bytes in routing messages

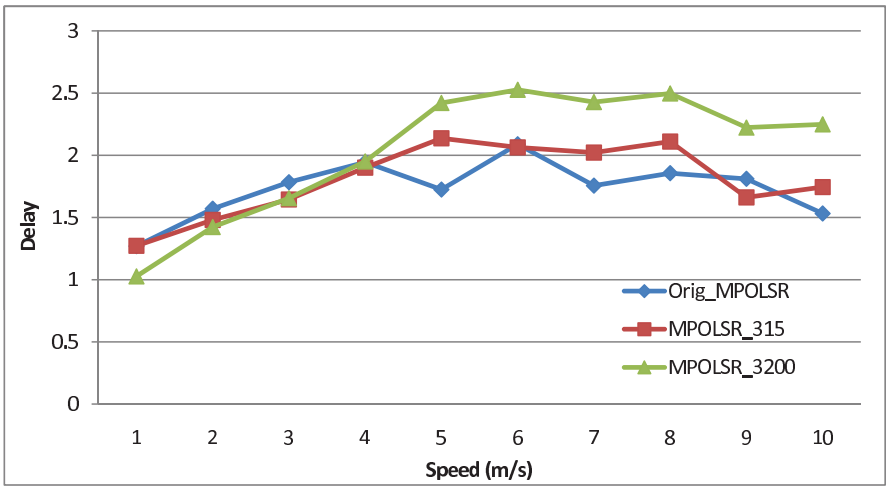


Figure 6.35: Average end-to-end delay of MP-OLSR with queue lenght metric in high traffic scenario

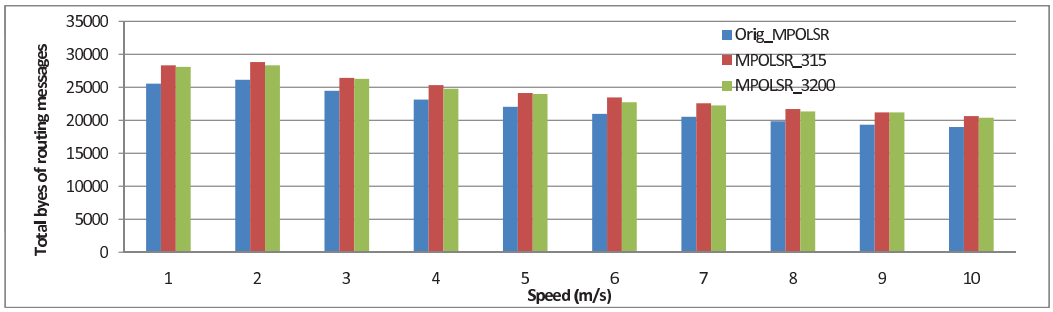


Figure 6.36: Total bytes of routing message

is about 9% more than the hop-count MP-OLSR (without increasing the packet number), which is an acceptable cost. The reason that number of bytes dropped as the node speed increases is mainly because in the scenarios with more topology changes, the connectivity among the nodes is also limited, so less routing message forwarding is needed.

We can conclude from the simulation results that the queue length metric can provide better performance in the low mobility scenarios with the accurate measurement of the network link quality. However, the improvement is limited as the mobility increases, which makes it more difficult to propagate the queue length information correctly.

6.2.4.6 Scenario for compatibility test

In this subsection, we present routing performances when OLSR and MP-OLSR protocols cooperate in the same network in order to check the backward compatibility. The following results are obtained with Qualnet simulator, and scenarios of 81 mobile nodes using a strict IP source routing. Nodes were uniformly placed initially on a square grid of 1500m×1500m. Table 6.2 summarizes the simulation settings.

6.2.4.6.1 Scenario 1: network with MP-OLSR source nodes In the first scenario, the simulation results are obtained for a network of 4 MP-OLSR source nodes. We change the density

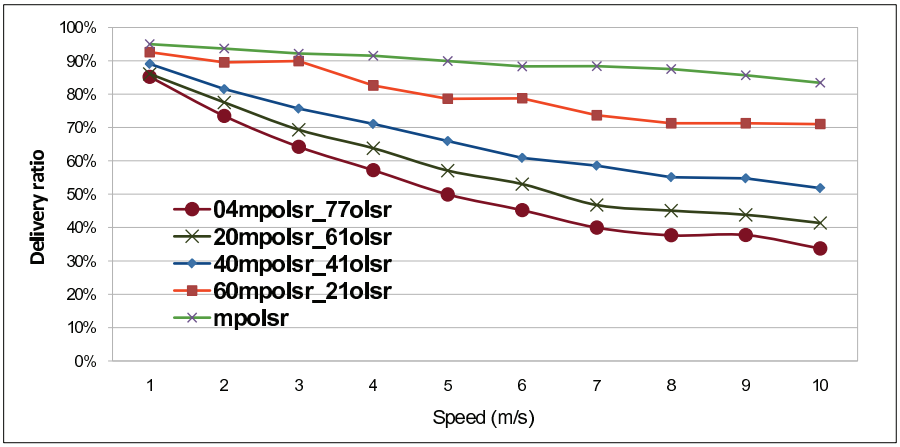


Figure 6.37: Ratio of delivered packets for a network of 81 OLSR and MP-OLSR mobile nodes, with MP-OLSR source nodes

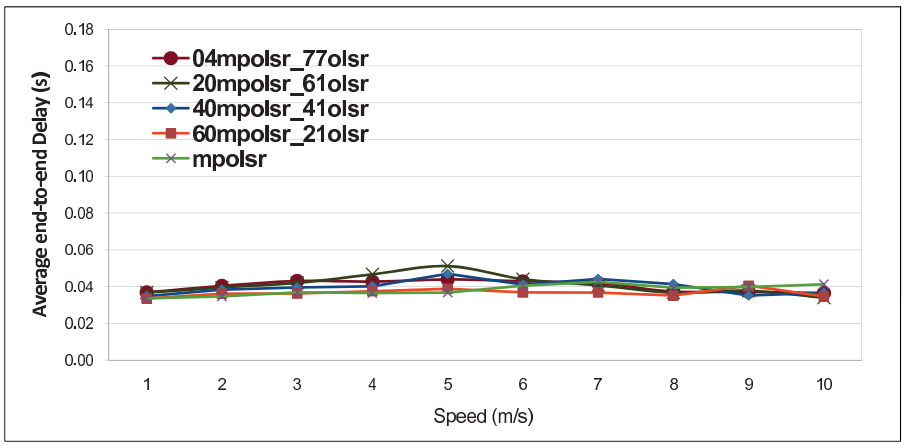


Figure 6.38: Average end-to-end delay for a network of 81 OLSR and MP-OLSR mobile nodes, with MP-OLSR source nodes

of the OLSR nodes. We start by studying a network of 4 MP-OLSR sources and all the rest 77 hosts are OLSR nodes (denoted *4mpolsr_77olsr*), then we replace OLSR nodes by MPOLSR nodes, and for each scenario we note the number of MP-OLSR and OLSR nodes in the network (*20mpolsr_61olsr*, *40mpolsr_41olsr*, *60mpolsr_21olsr*). Finally, we give the results for a network of only MP-OLSR nodes.

Figure 6.37 shows the delivery ratio when the OLSR nodes are involved in the routing by carrying the packet generated by the MP-OLSR source nodes. In general, the OLSR nodes have no problem in forwarding the source routing packets generated by MP-OLSR nodes. However, the OLSR intermediate nodes cannot have the same performance with MP-OLSR nodes. This is mainly because OLSR cannot perform route recovery and loop detection for the packets. So, for the scenarios where the density of OLSR nodes is too high, it is better for MP-OLSR nodes to just forward the packet to the next hop without appending the source route. From this figure, we can conclude that the compatibility can be achieved and the high density MP-OLSR nodes can provide better performance. On the other hand, OLSR nodes do not significantly affect the average end-to-end delay of the MP-OLSR protocol (Figure 6.38).

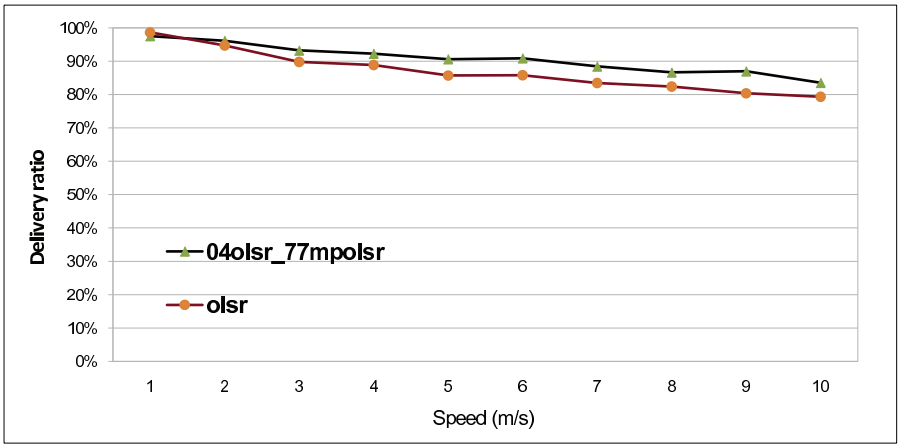


Figure 6.39: Ratio of delivered packets for a network of 81 OLSR and MP-OLSR mobile nodes, with OLSR source nodes

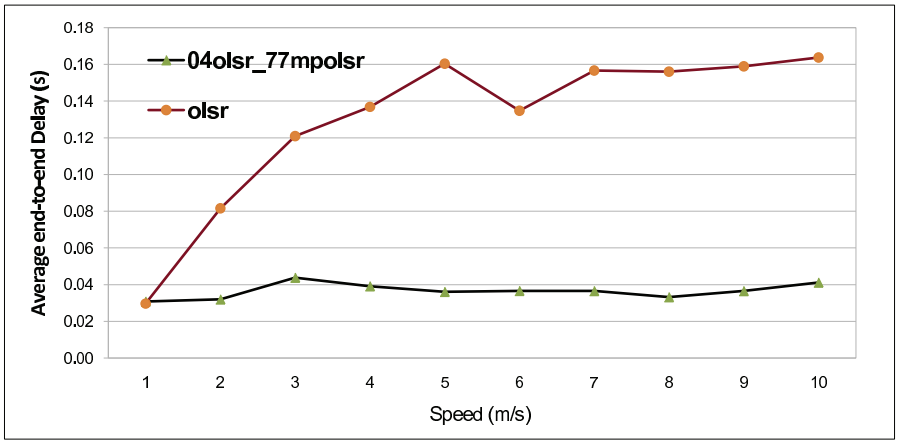


Figure 6.40: Average end-to-end delay for a network of 81 OLSR and MP-OLSR mobile nodes, with OLSR source nodes

6.2.4.6.2 Scenario 2: network with OLSR source nodes In the second scenario, we simulate the case in which the sources are OLSR nodes. Here, we have 4 OLSR source nodes with 77 MP-OLSR nodes, and we compare this scenario with that of all OLSR nodes. In Figures 6.39 and 6.40, we present the delivery ratio and average end-to-end delay respectively. As we can see in these figures, OLSR nodes have no problems in sending packets to MP-OLSR nodes. Furthermore, with the help of the loop detection and the multipath feature of MP-OLSR, we can increase the packet delivery ratio and reduce the end-to-end delay of the network.

The obtained results reveal that the MP-OLSR and OLSR can cooperate within the same network. This feature makes the deployment of the MP-OLSR protocol much easier because it can make use of the existing OLSR network. Because MP-OLSR nodes can perform multipath routing and loop detection, it can improve the performance of the network. For OLSR nodes, the route failure might increase when using source routing, because they do not have route recovery. Therefore, when the density of OLSR nodes is very high, it is better for MP-OLSR to forward the packets without source route to get hop-by-hop behavior.

6.3 Conclusion

In this chapter, the simulations of MP-OLSR were performed in NS2 and Qualnet simulators. A brief introduction of network simulation was first given, especially the *discrete-event simulation*.

For the first year of our research, the NS2 simulator was used for network simulation. The MP-OLSR is implemented based on OLSRv1. The simulation results reveal that the *link layer notification* and *route recovery* are very important for data delivery ratio by detecting the link broken in the networks. The MP-OLSR could offer better delivery ratio and delay than OLSR, especially in the scenarios with high mobility.

The simulations based on NS2 provide valuable results. However, during the simulations, we also found that the scalability of NS2 was limited. So the Qualnet simulator is employed for the rest of our research. Compared to NS2, it follows the same trend in simulation results, but provides better scalability, user interface, support for OLSRv2 and detailed documents.

More simulation scenarios are taken in Qualnet. The simulation results reveal that MP-OLSR can provide better quality of service than other protocols, especially in an error-prone network. The queue length link metric has almost the same performance as hop count metric in low data rate scenarios but can effectively improve the delivery ratio in the networks with high loads and low mobility. The compatibility simulations show that the MP-OLSR and OLSR could cooperate with each other in the same network.

However, the simulation results are still limited because it is hard for the simulator to model the realistic physical layer. So a testbed is implemented in the next chapter to validate the protocol in a real scenario.

Chapter 7

Implementation and Testbed

As already stated in Section 1.4, the simulation still has some limitations. For example, the two way ground radio channel model is much more simplified than the real world, which includes obstacles, reflections and influence from the environment (interference, temperature, etc). From the implementation point of view, there is also much difference between the testbed and simulation, such as information gathering, processing efficiency, etc.

The goal of the implementation of the platform is to validate the functions of MP-OLSR and show that the protocol works well not only in the simulators, but also in the real world. In this chapter, the testbed for MP-OLSR is firstly introduced. The multipath protocol is implemented on the netbook platform, based on Linux system. An UDP-based file transport protocol is used as application over the network and *Wireshark* and a suitable plugin is used for packet tracing.

Then the experiments are taken in the campus of Polytech'Nantes, France to evaluate the performance of MP-OLSR and compare the single path and multipath routing protocols.

The source code of the implementation is available online [1] [2].

7.1 Testbed for MP-OLSR

7.1.1 Hardware and Software Configuration

7.1.1.1 Hardware Platform

In the recent years, with the development of the wireless technology, there are a lot of choices of mobile equipments with different capacities of processing and storage. Those equipments include laptops, smart phones, PDAs, netbooks, wireless routers, etc.

For a testbed aimed for experimentation and validation, it is required that the hardware platform is easy to access and make modifications. Because the tests need to be taken outdoors, the mobile nodes should be portable. The devices should also have different interfaces such as 802.11 a/b/g wireless interface, Ethernet card and USB for both wireless transmission and result collection. Furthermore, the equipment should be relatively less expensive because several nodes are going to be included in the testbed.

The smart phone and PDA are highly portable with long battery life. But it is not easy to develop and debug the software on them. The wireless routers such as Linksys WRT54GL are

[1] http://www.irccyn.ec-nantes.fr/spip.php?article527
[2] http://www.jiaziyi.com/MP-OLSR.php

Parameter	Value
CPU	Intel Atom N270 1.6 GHz
Memory	DDR2 1024MB
Hard Disk	20 GB
Chipset	INTEL 945GMS + ICH7-M
Graphics	integrated GMA 950
Screen	8.9 inche, 1024×600
USB	3×USB 2.0
Card reader	SD/MMC
LAN	10/100(Attansic L2)
WiFi	802.11 b/g mini PCI-E Card

Table 7.1: Parameters of eeePC 901

also examined. It has a Boardcom BCM 5352 CPU with 200MHz, equiped with a 16 MB RAM and 4 MB flash memory. It is possible to hack the router and replace it with OpenWrt [3], which is a Linux distribution for embedded devices. However, the memory is too limited to install the MP-OLSR module and related libraries, and store the logs generated during the test.

Considering all the issues stated before, the ASUS eeePC 901 netbook with parameters shown in Table 7.1 is chosen as the hardware platform of the testbed. It has enough disk space and CPU speed to perform the experiments. The widely used operating systems and programming tools can be used in the netbook, which greatly facilitates the development and tests.

Except for the mobile nodes of the ad hoc network, an *analyze machine* and *sniffers* are also included in the testbed as shown in Figure 7.1. The analyze machine is a fixed computer with the capacity to collect and analyze the information of routing packets, data packets and QoS metrics, which come from the nodes of the ad hoc network. The sniffers are also mobile nodes, with the capacity of wireless communication which can receive and inject packets to the network. They are used to validate the security of the network and can generate arbitrary data to disturb the network.

7.1.1.2 Operating System

To develop the new routing protocol, it is required that there are enough APIs provided to implement the related functions and the developer be able to easily access the system kernel. In fact, because the MP-OLSR does not use regular table-based IP routing, but the source routing, it is necessary to modify the IP layer of the system.

The *Backtrack*[4] Linux distribution is used because there are lot of network tools included. It is a Linux-based penetration testing arsenal that aids security professionals in the ability to perform assessments in a purely native environment dedicated to hacking. The tools provided by *Backtrack* include information gathering, network mapping, vulnerability identification, etc.

The system is installed in a 2 GB SD card instead of the hard disk to facilitate the modification of the program in several eeePC machines.

[3] OpenWrt, http://openwrt.org/
[4] *Back Track Linux Distribution, http://www.backtrack-linux.org/*

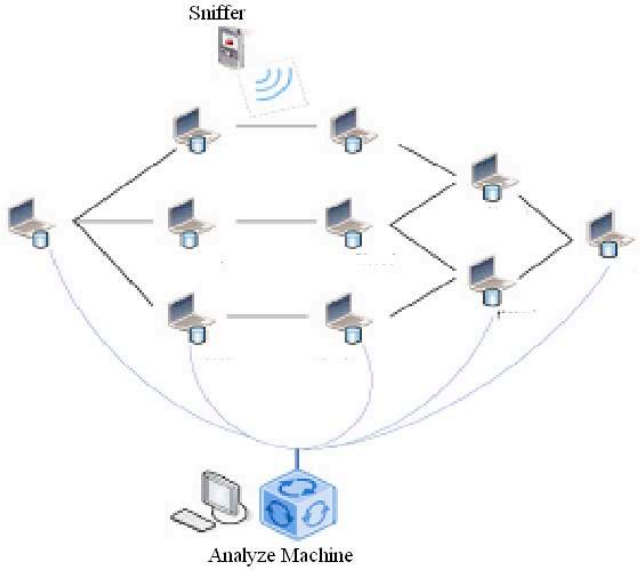


Figure 7.1: The testbed with mobile nodes, analyze machine and sniffer

Figure 7.2: Some ASUS eeePCs 901 before deployment at the headquarters of the testbed

Name	Lastest version	Language	Supported OS
Olsrd	v0.6.0 05/2010	C	Linux, Windows, Iphone, Mac OSX...
PyOLSR	07/2003	Python	Linux, Windows
OOlsr	11/2004	C++	Linux, Windows
NRL-OLSR	08/2007	C++	Windows, Linux, Zaurus...
CRC-OLSR	2004	C	Linux
nOLSRv2	2009	C	Linux

Table 7.2: Implementation of OLSR protocol

7.1.1.3 OLSR prototype

Because the MP-OLSR and OLSR share the same topology sensing mechanism, it is better to extend the existed OLSR implementation to the multipath routing. At the moment, there are several OLSR implementations available online as shown in Table 7.2. Because it is necessary to modify the source code of OLSR to implement multipath routing, only the open-source implementations are considered.

In all those implementations, the *olsrd* is chosen as the prototype of MP-OLSR. The *olsrd* has always been well maintained since 2004. Its advantages include:

- High portability. It can run on different platforms including Windows, Linux, OS X, Android, OpenWrt, etc.
- Little CPU occupancy. It is helpful for saving valuable battery on the portable devices.
- High scalability. It has been deployed on community wireless mesh networks with up to 2000 nodes.
- Open source. It is released under a BSD license. So it is easy to integrate other works with *olsrd* implementation.

7.1.1.4 UFTP Application

The service to be applied in the testbed is another issue that needs to be considered in the experiments. Given the fact that the ad hoc network is an error-prone environment, it is not suitable for a TCP based application because the acknowledgement in the TCP transmission will even congest the fragile wireless links. So in our testbed, the UFTP[5] application is used.

UFTP is an UDP based encrypted multicast file transfer program, designed to securely, reliably, and efficiently transfer files to multiple receivers simultaneously. A UFTP session consists of 3 main phases: The Announce/Register phase, the File Transfer phase, and the Completion/Confirmation phase. The File Transfer phase additionally consists of the File Info phase and the Data Transfer phase for each file sent.

Figure 7.3 illustrates the main procedures of a UFTP transmission.

The server sends out an announcement over a public multicast address which the clients are expected to be listening on. On receiving the announcement message, the client sends a registration to respond to the announcement. The server will then send a confirmation message for file transmission.

[5]UDP based FTP with multicast, http://www.tcnj.edu/~bush/uftp.html

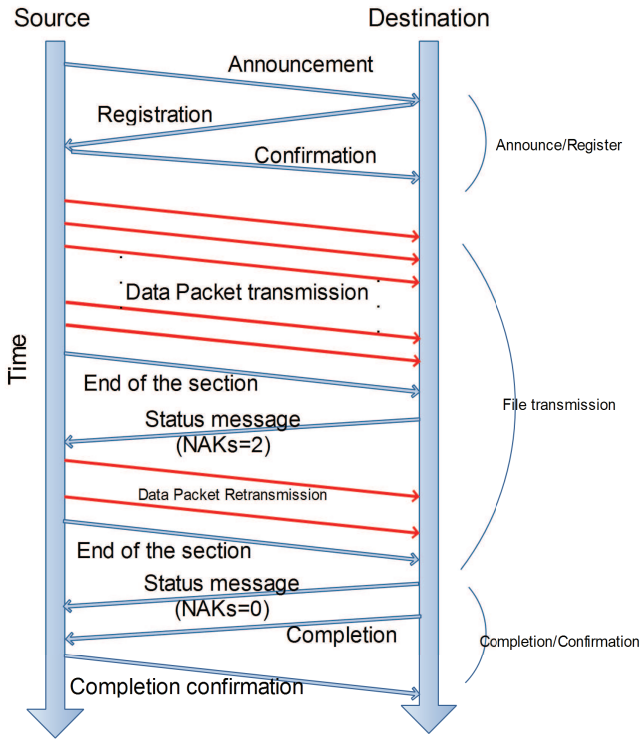


Figure 7.3: A file transmission by using UFTP

Then it is followed by the file transmission phase. The server sends a message describing the file in question. Besides the name and size of the file, this message describes how the file will be broken down. A file is divided into a number of blocks, and these blocks are grouped into sections. A block is a piece of the file that is sent in a single packet. A section is a grouping of blocks that can be sent together before the server needs to request feedback from the clients. The total number of blocks and sections is included in this message. Because UDP does not guarantee that packets will arrive in order, each block is numbered so the client can properly reassemble the file. When the server finishes a section, it sends a message to the clients requesting status. The clients then send back a status message containing the list of NAKs (negative acknowledgments) for the blocks in that section. Once all sections have been sent, if the server has received a non zero number of NAKs from any client, the server will begin a second pass of the data, this time only sending the packets that were NAKed. The server will continue with subsequent passes of the data until all clients have either received the file or have timed out while the server was waited for a status message. When a client has received the entire file, it sends a completion message in response to the next status request. For each section, the file transmission phase is repeated.

The Completion/Confirmation phase shuts down the session between the server and clients. It starts with a message from the server indicating the end of the session. The clients then respond with a completion message, and the server responds to each completion with a confirmation message.

7.1.1.5 Results and Log

The following log files are used to analyze the results of each scenario:

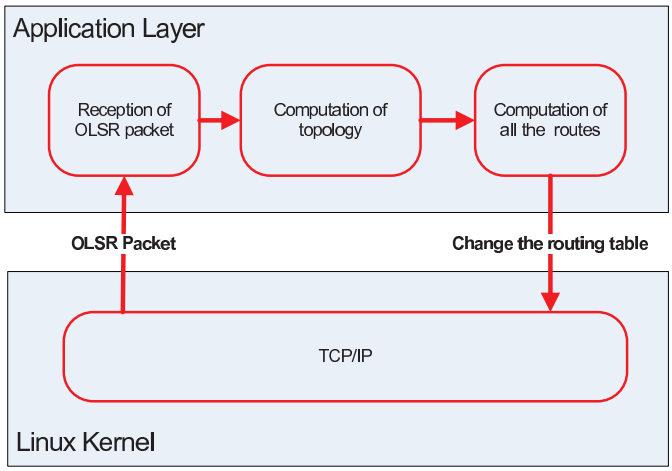


Figure 7.4: Implementation of OLSRd in Linux

- MP-OLSR log: log of routing operations of MPOLSR (link detection, route calculation with Multipath Dijkstra algorithm).
- UFTP log: log of the process of the UFTP application.
- Packet log: log of packets captured by network interface in the network. We make use of the Wireshark [6] to monitor the packet transmission in the network. It is a free and open-source packet analyzer for network troubleshooting, analysis and protocol development. To parse the source routing header of the MP-OLSR, a plug-in for Wireshark is also developed.

Relying on the log files, we measured the following parameters for each test:

- Test duration.
- Transfer duration: the duration of UFTP transmission, from the first to the last packet sent.
- Requested rate: the rate initialized by the user.
- Average rate: the number of data bytes actually delivered during the transfer.

7.1.2 Implementation of MP-OLSR

The MP-OLSR implementation for the testbed was developed by Keosys and University of Nantes in the context of ANR SEREADMO project. It is based on *olsrd*. The *olsrd* works in the application layer in the system. It makes use of the UDP protocol to send out HELLO and TC messages. On receiving a OLSR routing packet (TC or HELLO) from the IP layer of the Linux kernel, the routes to all the possible destinations in the network are computed. Then the next hops of the destinations are saved in the IP routing table by using *olsr_update_kernel_routes*. The *olsrd* does not process the data packets directly. The IP routing forwards the packets to the next hop automatically. This procedure is illustrated in Figure 7.4.

In MP-OLSR, because the source routing is used, the MP-OLSR module needs to compute the multiple path and appends the source routing head for the data packets. So four tasks need to be done by the multipath module:

[6] Wireshark, http://www.wireshark.org/

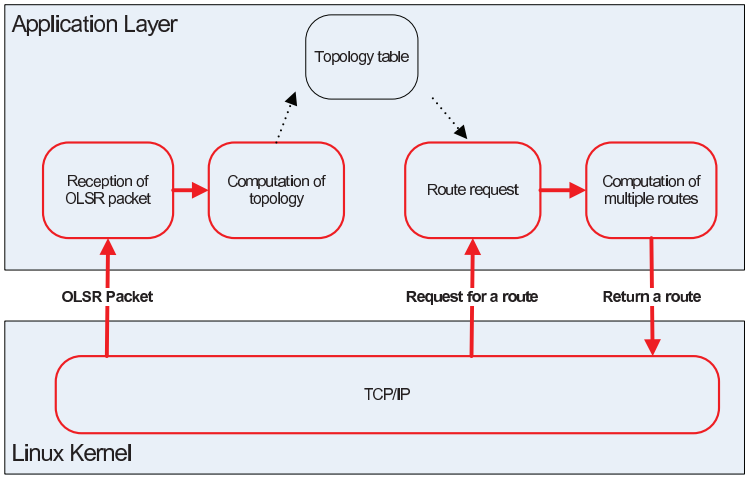


Figure 7.5: Implementation of MP-OLSR in Linux

1. It needs to send and process the TC and HELLO messages and maintains the information topology base, just like OLSR.
2. When there are route requests of the data packets, the Multipath Dijkstra Algorithm are called to get multiple paths to the destination.
3. After the path is obtained, the module appends the source route to the packet and sends it to the next hop.
4. For intermediate nodes, they read the source routing information and forward the packet to the next hop.

The packet processing procedure is show in Figure 7.5.

To accomplish the listed functions, there are two modules in the implementation of MP-OLSR. The first is an extension of *olsrd*, the second is a netfilter module.

7.1.2.1 Multipath Extension of OLSRd Module

The multipath extension of *olsrd* is used to detect the topology changes in *olsrd* (i.e. topology information base) and transmit the changes to the *netfilter module*.

It is a dynamic library in Linux and exists as a plug-in in *olsrd* (Tonnesen, 2004). The extensibility of *olsrd* allows developers to add custom packages or functionality without changing any code in *olsrd* itself. With the help of function *register_pcf*, after the module is loaded, there will be a callback of *olsrd_pcf_event* when there are modifications in the topology. The topology generated by *olsrd* is then saved in the specified data structure of multipath module. Whenever there is a change of the topology, the database of the module is updated. The updated information will also be transmitted to *netfilter module*. A data structure with fixed size (*dev_tc_token*) is used to transmit the topology modifications.

More details about the implementation of the multipath plugin can be found at Appendix B.1.

7.1.2.2 Netfilter Module

The *netfilter module* is used for data packet processing and multipath computation. It receives topology information changes from the *olsrd* multipath plugin, and maintains a topology infor-

mation base. When a data packet arrives, instead of routing the packet by IP routing table, the *netfilter module* will process the packet by source routing:

- If the local node is the original of the packet, a source route will be computed and appended on the packet;
- If the local node is an intermediate node, the packet will be forwarded according to the source route;
- If the local node is the destination, the source routing head will be removed and the packet will be passed to the upper layer.

More details about the implementation of the multipath plugin can be found at Appendix B.2.

7.2 Experiment and Performance Analysis

The experiments of the testbed were taken in June, 2009 in the Chantrerie Campus of Polytech'Nantes, France. Researchers from our team (IRCCyN-IVC) and our partners (Thales Communication, Keosys and SIC) participated in the experimentation.

7.2.1 Preparation of the Platform for Experimentation

7.2.1.1 Configuration Scripts

When the *Backtrack* 3 starts, the *cfg_ser.lzm* is loaded. This file just includes one script (*rc.sereadmo*) and called the *config.csh* in the SD card. The initialization of different components of the MP-OLSR makes use of the scripts below:

config.csh It is called directly by the OS at the starting of the system. It will mount the initialization directory and call the *config_std.csh* script.

config_std.csh* and *common.csh The following operations are performed by *config_std.csh*:

1. Network initialization. The address for LAN and WLAN are assigned to different machines.
2. *olsrd module* initialization.
3. *Netfilter module* initialization.
4. *Wireshark plugin* initialization. The plugin is for packet tracing during the test.
5. SSH initialization. The SSH is initialized for remote control during the test.
6. NTP initialization. The NTP is initialized for time synchronization.

7.2.2 Realistic Testbed Results

Because the number of the machine is limited (15 netbooks), it is necessary to place the nodes carefully to make sure that there are wireless links from the source to the destination. So each link was previously measured by using the *ping* command. Different scenarios are designed to test the multiple path and multi-hop performance. During the experiments, the host node uses the SSH to control other machines and walky-talkies are used for communication between different members of the team. Figure 7.6 shows some pictures of the test day.

Figure 7.6: Test day of the SEREADMO

7.2.2.1 Scenario 1, OLSR and MP-OLSR on 4 paths

The first scenario presented includes 6 nodes. The location of the nodes is shown in Figure 7.7. The object is to test the multipath algorithm, so we try to find as many paths as possible in this simple scenario. The number of paths for MP-OLSR is set to 4. A node with an IP address 10.0.0.100 is chosen as source, and another node 10.0.0.98 as destination. The distance from the source node to the destination node is about 60 meters. There are different kinds of obstacles in this scenario: trees, buildings, and cars moving between the nodes, which will block certain links for a random period of time. Iptable rules are employed to block the direct transmission between source and destination to construct a multi-hop scenario.

In the following tests, the data rate is set to 62 KBytes/s to transfer a file with 17.8 MBytes. Results are presented in Table 7.3. For MP-OLSR, the transmission was finished in 6 minutes 12 seconds. Nodes with IP addresses 10.0.0.90, 10.0.0.95, 10.0.0.99 and 10.0.0.105 are chosen as intermediate nodes to relay the packets. During the transmission, 9.9% of the packets were lost. For OLSR, only 10.0.0.90 and 10.0.0.95 are used to forward the data packets. The connection is lost after 5 minutes 17 seconds of transmission. For the packets sent out, 37.53% were lost.

To compare the network performance of these two protocols, we also analyzed the log file from Wireshark. Figures 7.8 show the number of packets sent out to different nodes from the source (10.0.0.100) to the next hops in each tick (1 second per tick) for MP-OLSR and OLSR respectively.

As we can see from Figure 7.8a, for a fixed source rate (10.0.0.100), the traffic load is distributed over 4 paths. The transmission is almost continuous even if some of the links are unavailable. When certain links break, the traffic is assigned to other nodes (for example, from 290s to 300s and 380s

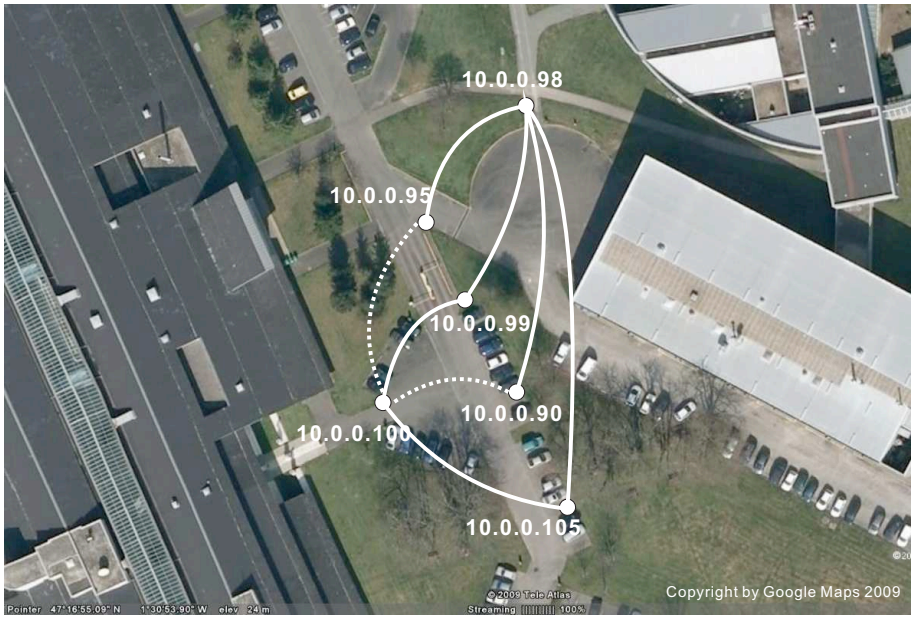
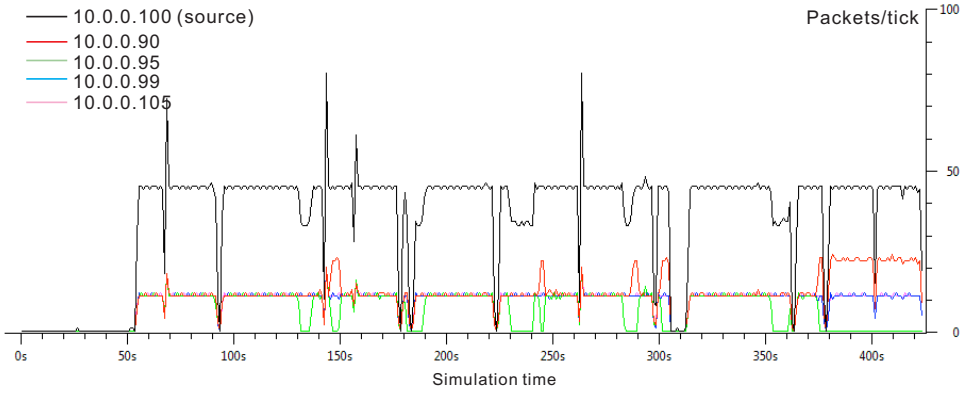


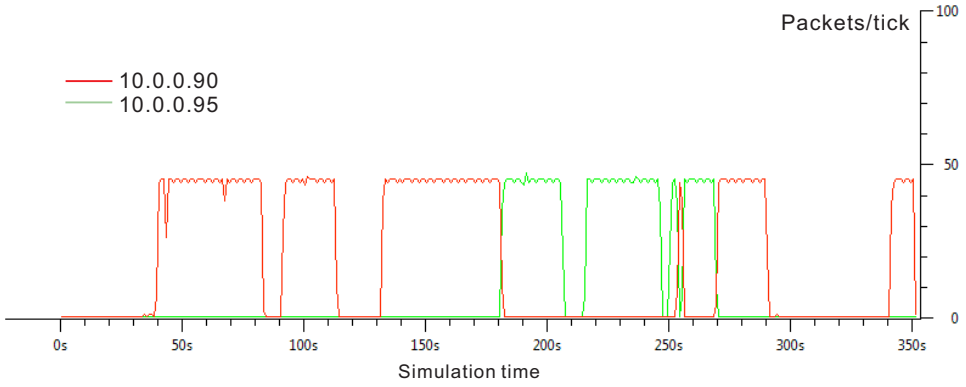
Figure 7.7: Network topology of the scenario 1: OLSR and MP-OLSR with 4 paths. 10.0.0.100 is the source and 10.0.0.98 is the destination

Protocol	MP-OLSR	OLSR
Duration of the transmission	6m 12s	5m17s (Connection lost)
Test duration	7m50s	6m26s
Size of the sent file	17.8 MB	17.8 MB
Requested rate	62KB/s	62KB/s
Average rate	48.99 KB/s	38.73 KB/s
Packets sent by 10.0.0.100	15002	9784
Packets sent by 10.0.0.90	4503	5303
Packets sent by 10.0.0.99	3715	0 (route not used)
Packets sent by 10.0.0.95	2084	2909
Packets sent by 10.0.0.105	3726	0 (route not used)
Packets received by 10.0.0.98	13516	6112
Rate of lost packets	9.90%	37.53% (Connection lost)

Table 7.3: Results of scenario 1, OLSR and MP-OLSR with 4 paths, data rate = 62 KB/s



(a) MP-OLSR



(b) OLSR

Figure 7.8: Wireshark trace of scenario 1 with rate=62KBytes/s

to 420s).

Compared to MP-OLSR, the transmission with OLSR (Figure 7.8b) is just through one path (so, the source rate is the same with the unique path and is not plotted here). The transmission is interrupted for a short period because the only path is unavailable. In this case, the node will try to find another route to the destination. But the data transmission will be stopped during this period (for example, 80s-90s, 115s-130s in figure 7.8b). And if this kind of route switch takes too much time, and the host or destination failed to receive acknowledgment for the other side (Figure 7.3), the connection will be lost and result in the failure of the file transmission in the end.

7.2.2.2 Scenario 2: OLSR and MP-OLSR routing on 3 paths

In the second scenario, we compared OLSR and MP-OLSR with 3 paths which are three or four hops away. The goal is to test the protocol with longer paths in a complex scenario. The allocation of the nodes is shown in Figure 7.9. The distance from the source to the destination is about 200 m. There is a large building between them. To reach the destination, the packets have to travel around the building or go through the hall inside the building.

Several links are unstable, mainly those in the parking-lot and inside the building (Figure 7.10). MP-OLSR only uses one or two paths most of the time. However, despite these frequent changes in the network topology, the transmission is successful with MP-OLSR.

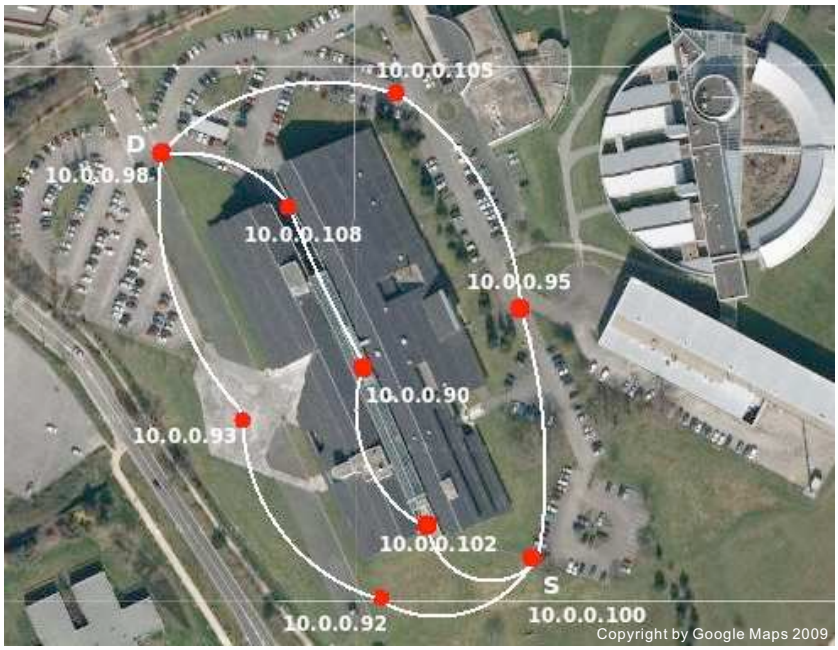


Figure 7.9: Network topology of scenario 2: OLSR and MP-OLSR with 3 paths, 10.0.0.100 is the source and 10.0.0.98 is the destination

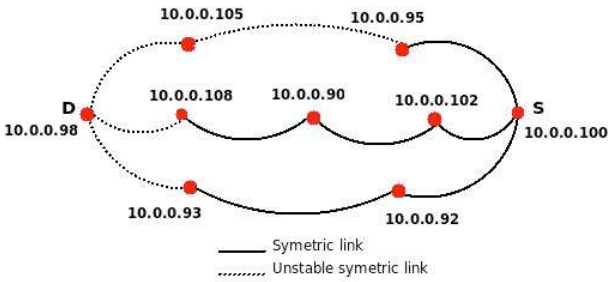


Figure 7.10: Quality of links in scenario 2: OLSR and MP-OLSR with 3 paths

Protocol	MP-OLSR	OLSR
Duration of the transmission	9m40s	8m43s (Connection lost)
Test duration	14m6s	9m6s
Size of the sent file	17.8 MB	17.8 MB
Requested rate	62KB/s	62KB/s
Average rate	31.42 KB/s	34.85KB/s
Packets sent by 10.0.0.100	14528	12548
Packets received by 10.0.0.98	9145	8544
Rate of lost packets	37.05%	31.90% (Connection lost)

Table 7.4: Results of scenario 2, OLSR and MP-OLSR routing with 3 paths

With OLSR, the UFTP connection stopped after sending several packets beofre the transmission completed. This is because compared to multiple paths, the unstable paths have more negative effects on the single path routing protocol. So the file transmission failed in the end because of time out. Test results are presented in Table 7.4.

In this subsection, experimentations are performed to show the efficiency and validity of the MP-OLSR routing protocol in real scenarios. UFTP is taken as application example. Mobility is not considered in the presented scenario, but the network topology still changes due to the failure of links.

7.2.3 Semi-realistic Testbed Results

If we compare the results from the simulator and the real testbed, we will find that the network performance in real testbed is not as good as that in the simulator, but with the same trend. This is reasonable because in the network simulation, we simulate the physical layer in a free space by using the ideal mathematical model (two-ray ground and shadowing). And in the real scenario, there are many more factors that will affect the final results: obstacles, radio reflection (much more complex than two-ray ground model), texture in the environment, radio interference (from other Wi-Fi devices, etc.), or even humidity and temperature. However, although it is hard to simulate the exact real scenario with current simulation technology, we can still have a relative evaluation between the protocols through the simulation results.

Because of the limitation of resources, it is very hard for us to perform the tests in the real scenario with a large number of nodes and mobility like in the simulation (for example, tens of nodes moving in an area of 1 km2). As a compromise, to test the performance of the protocols with the real UFTP application in a more complex scenario, we set a semi-realistic testbed based on IP Network Emulation (IPNE)[7] interface, as shown in Figure 7.11. The IPNE implements a packet sniffer/injector. It sniffs the packets from the physical layer network, sends them through the Qualnet simulation, and injects them back to the physical network. The virtual Qualnet Network is transparent to the UFTP application.

We launch the UFTP transmission at 100kbps, with the same under layer settings as in Table 6.2. There are also 81 nodes in the network with medium mobility (maximum 5m/s). The Wireshark traces are shown in Figures 7.12 for OLSR and MP-OLSR respectively. The same thing happens with the tests in the real scenario, the file transmission using OLSR failed after a short period of time (170s). And for MP-OLSR, although it also suffers from the unstable paths during

[7]Scalable Network Technologies, IP Network Emulation Interface

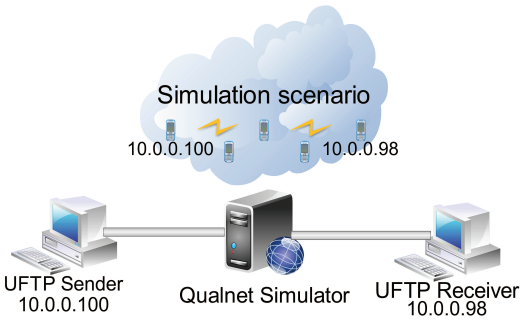
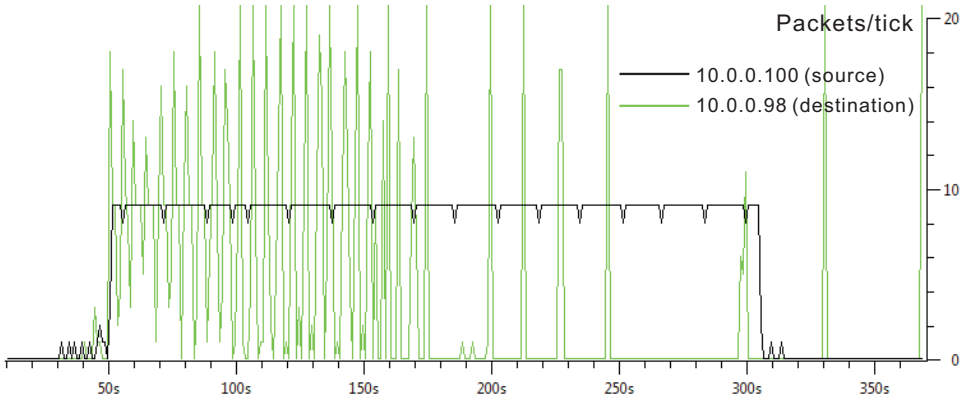
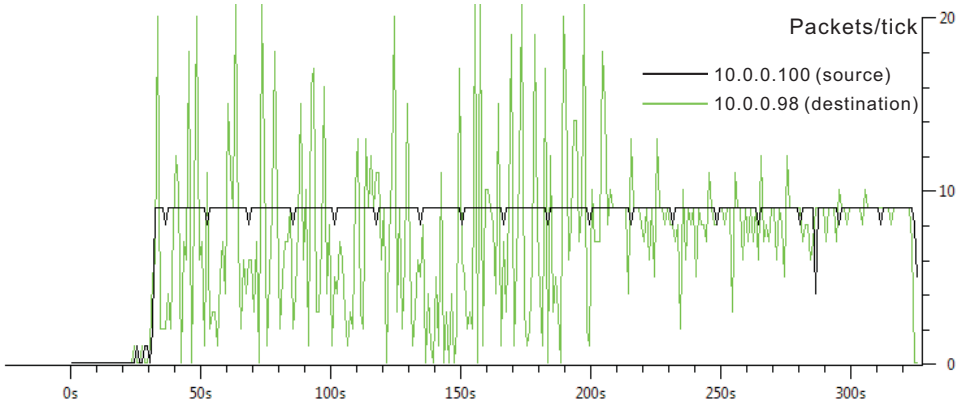


Figure 7.11: Semi-realistic IPNE testbed, with UFTP application

the same period, the file successfully reached the destination in the end. This result is consistent with the real testbed.



(a) OLSR



(b) MP-OLSR

Figure 7.12: Wireshark trace of UFTP source and destination nodes, 81 nodes OLSR ad hoc network

7.3 Conclusion

In this chapter, a testbed for MP-OLSR was implemented based on *olsrd*. The extension mainly includes two modules: *multipath plugin for olsrd* and *netfilter module*. Several experimentations are performed on the testbed. The nodes were placed in real scenarios and the UFTP was used for file transmission. From the *Wireshark* packet trace we can learn that the single path routing protocol had transient packet loss during the transmission which led to the transmission failure because of timeout. For multipath routing, although there were also route failures in the multiple routes, there was less chance that all the routes broke at the same time. So the file transmission could continue till the end. The semi-realistic testbed also confirmed our conclusion.

Based on the results obtained from the simulator and testbed, we can conclude that MP-OLSR could offer better performances than OLSR, especially in the harsh scenarios (high mobility in simulation and frequent link breakage in the testbed). This is mainly because the proactive-based multipath Dijkstra algorithm could find appropriate multiple paths and distribute the traffic into different paths by source routing. And the *route recovery* and *loop detection* could effectively avoid the disadvantage of source routing and the possible loops in the network.

Part III

Scalable Video Services over ad hoc Networks

Introduction

In the Part II, the multipath routing protocol has been studied in detail. In the simulation and experiments, the Constant Bit Rate (CBR) application is mainly used to measure the performance of the network through different metrics: packet delivery ratio, delay, jitter, etc.

In this part, we will focus on the application of video services over the multipath routing protocol. Compared to the custom data transmission, the quality of service required by multimedia application is different:

- High bandwidth. Even with the video compression technology, the bandwidth needed by video transmission is much more significant than E-mail or regular web-access. The higher the video resolution and frame rate, the higher bandwidth is required.

- Low delay. The file transmission and on-demand video applications are not delay sensitive. If the delay is unformly 0.2s or 2 seconds, there is no much difference. However, for the real-time video applications, such as videoconferencing, the delay requirements are very strict. The images delayed by 2 seconds are unacceptable.

- Low jitter. The arriving order of the packets is not important for regular file transmissions. However for the video and audio services, if the transmission time varies randomly between 200ms and 2 seconds, the result will be a big problem for the decoder.

- Reliability. In the file transmissions, all the packets are required to be delivered correctly. Normally the lost packets will be retransmitted (such as UFTP). In the video transmission, errors are tolerated. But the quality of the video will be degraded because of the packet loss.

- Quality of Experience (QoE). The goal of the video transmission is to provide high quality video services for end users. The quality of the video is one of our main concerns.

We are especially interested in the H.264/SVC, which is the scalable extension of the latest H.264/AVC standard and is approved as Amendment 3 of AVC.

In Chapter 8, the H.264/SVC and its transport interface are introduced. We begin with the three types of scalability that H.264/SVC provided: *temporal*, *spatial* and *quality* scalability, and the data structure in NAL units to keep this scalable information. Then we further studied the properties of H.264/SVC in the error-prone networks. We presented four different error concealment algorithms in the literature briefly, then we discussed the priority of the packets from different scalable layers by examining the effect of packet loss to the video quality. This can help us to define the importance of different packets, which is crucial for the priority coding.

In Chapter 9, we studied the H.264/SVC video transmission over ad hoc networks. A video evaluation framework, *SVCEval* is build for simulating H.264/SVC transmission over different kinds

of networks. Compared to other simulation methods, *SVCEval* can provide good balance between the reality of the simulation and the cost of the experiments. Then the framework of Priority Forward Correction (PFEC) coding is introduced for multipath scalable video transmission. The PFEC is also integrated into the *SVCEval* evaluation framework to test the performance of the coding scheme.

Chapter 8

H.264 Scalable Video Coding and Transport Interface

In the State of the Art, we have mentioned H.264/SVC as an extension of H.264/AVC. In this chapter, we will go further on the scalability properties of H.264/SVC. In section 8.1, the system model of H.264/SVC is reviewed. It includes the scalable extension (*temporal, spatial,* and *quality*) of SVC and the transport and signaling of SVC. For wireless communication, we are interested in the error concealment of the video codec. So in section 8.2, we first introduced some well-known error-concealment algorithm in H.264/SVC. Then the packet-loss simulation is performed to study the packet priority in H.264/SVC bitstream. This is important for the further study of Unequal Error Protection in the Chapter 9.

8.1 System Models

8.1.1 The Scalable Extension of SVC

Because H.264/SVC is developed as an extension of H.264/AVC, it inherits the core coding tools of AVC. One of the design principles of SVC is that new tools should only be added if necessary for efficiently supporting the required types of scalability: *temporal scalability, spatial scalability* and *quality scalability* (Schwarz et al., 2007). In the following of this subsection, those three kinds of scalability in the literature will be introduced briefly.

Temporal Scalability A bit stream provides temporal scalability when the set of corresponding access units can be partitioned into a temporal base layer and one or more temporal enhancement layers with the following property. The temporal layers can be identified by a temporal layer identifier T, which starts from 0 for the base layer and is increased by 1 from one temporal layer to the next. Then for each natural number k, the bit stream that is obtained by removing all access units of all temporal layers with a temporal layer identifier T greater than forms another valid bit stream for the given decoder.

For hybrid video codecs, temporal scalability can generally be enabled by restricting motion-compensated prediction to reference pictures with a temporal layer identifier that is less than or equal to the temporal layer identifier of the picture to be predicted. In H.264/SVC, temporal

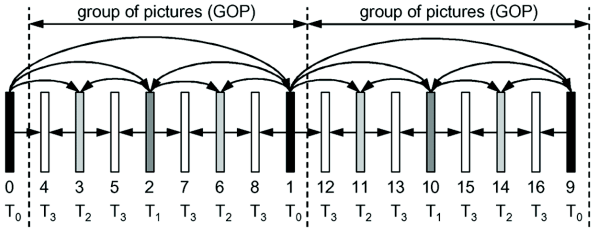


Figure 8.1: Prediction structures for temporal scalability, coding with hierarchial B-pictures ($T_i > 0$), (extracted from (Schwarz et al., 2007))

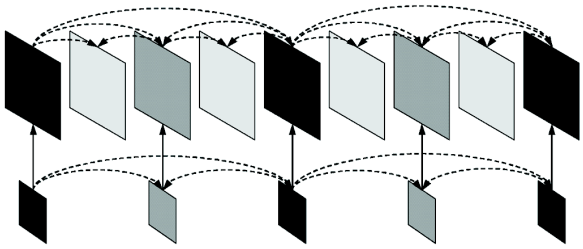


Figure 8.2: Multilayer structure with additional inter-layer prediction for enabling spatial scalable coding (extracted from (Schwarz et al., 2007)).

scalability with dyadic temporal enhancement layers can be efficiently provided with the concept of hierarchical B-pictures.

An example is as shown in Figure 8.1. The set of picture between two successive pictures of the temporal base layer together with the succeeding base layer picture is referred to as a *group of pictures* (GOP). The enhancement layer pictures are coded as B-pictures. Each set of temporal layers $\{T_0, ..., T_k\}$ can be decoded independently of all layers with a temporal layer identifier $T > k$. The arrows indicate the dependency of different layers.

Spatial Scalability For supporting spatial scalable coding, SVC follows the conventional approach of multilayer coding, which is also used in some previous coding standards (H.262, H.263, etc). Each layer corresponds to a supported spatial resolution and is referred to by a spatial layer or *dependency identifier D*. The dependency identifier D for the base layer is equal to 0, and it is increased by 1 from one spatial layer to the next. In each spatial layer, motion-compensated prediction and intra-prediction are employed as for single-layer coding. But in order to improve coding efficiency, additional so-called inter-layer prediction mechanisms are incorporated. An example of spatial scalable coding is shown in Figure 8.2.

Quality Scalability Quality scalability can be considered as a special case of spatial scalability with identical picture sizes for base and enhancement layer. This case is supported by the general concept for spatial scalable coding and it is also referred to as coarse-grain quality scalable coding (CGS).

8.1.2 NAL Unit and RTP Payload for SVC

As H.264/AVC, the bitstream of SVC is transmitted in NAL units. The NAL unit syntax of SVC is an extension to the one byte NAL unit structure of H.264/AVC (Amon et al., 2007). Figure 8.3

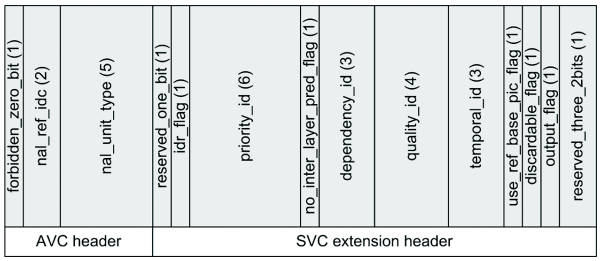


Figure 8.3: SVC NAL unit structure (extracted from (Amon et al., 2007))

shows the structure of SVC NAL unit.

The first byte of the header extension mainly contains the syntax element *priority_id* and also indicates whether the NAL unit belongs to a so-called IDR (instantaneous decoding refresh) access unit (*idr_flag*). The second and third byte provide information on the scalability dimensions represented by the syntax elements *dependency_id*, *temporal_id* and *quality_id*. In addition, the second and third byte of the extension NAL unit header provide information e.g., about the possibility to discard NAL units from the decoding of layers with higher *dependency_id* (*discardable_flag*), whether a NAL unit is coded without using inter-layer prediction (*no_inter_layer_pred_flag*) or if a decoded base picture (i.e., *quality_id* equal 0) is used for inter prediction (*use_ref_base_prediction_flag*).

The RTP payload for H.264/AVC has been defined in RFC 3984 (Wenger et al., 2005) by IETF Network Working Group. The standardization of RTP payload format for SVC video is still in progress (Wenger et al., 2010). The design criteria of the payload includes (Wenger et al., 2006):

1. Backward compatibility with RFC 3984. As SVC is a backward compatible extension of H.264/AVC, the same should be the case for its packetization. In particular, it should be possible—or even required—to transport the base layer utilizing RFC 3984. Only if this is achieved, RFC 3984-aware legacy devices are still capable of utilizing an SVC base layer in an RTP environment.

2. Re-use of RFC 3984's mechanisms wherever possible. Many person-years of standardization and implementation work have been allocated to the design of RFC 3984. Re-using these results has positive commercial implications, as experience with the packetization and its technical and commercial environment is already available.

3. Focus on the use case of Internet transport. Intelligence in the network (such as congestion control and security) is not considered a desirable feature; whenever possible, intelligence should be implemented at the endpoints.

8.2 H.264/SVC in an Error-Prone Network

8.2.1 Error Concealment in H.264/SVC

In the previous section, the scalability properties of SVC and packetization of the bitstream are introduced. When the packets are being transmitted in the unreliable networks, especially in the wireless environment, it is inevitable that packet loss happens. Some of the packet loss has severe

impact on the playback quality of the compress video. So some error concealment techniques are created to guarantee the robustness of the SVC video transmission.

The algorithms can be divided into two categories: the *Inter Layer Error Concealment* takes lower layer information, and the *Intra Layer Error Concealment* just considers the information in the same spatial layer. In (Chen et al., 2006), the authors proposed four error concealment algorithms:

- Frame Copy (FC). This is an intra layer error concealment method. In the "frame copy" (FC) algorithm, each pixel value of the concealed frame is copied from the corresponding pixel of the first frame in Reference Picture List 0 (RefPicList0). This algorithm can be invoked for both base layer and enhancement layer.

- Temporal Direct (TD). This is an intra layer error concealment method. In the "temporal direct motion vector generation" (TD) algorithm, the MVs and reference indices of each subblocks of the missing frame are calculated as if they were coded using the "temporal direct mode". This algorithm can be invoked for both base layer and spatial enhancement layer.

- Motion and Residual Upsampling (BLSkip). This is an inter layer error concealment method. In the "motion and residual upsampling" (BLSkip) algorithm, SVC tools are used and the BLSkip mode is set in the enhancement layer. Residual upsampling is also used to upsample the residual of the base layer for enhancement layer. However, the motion compensation is done at the enhancement layer using the upsampled motion fields. This algorithm can directly be used for the enhancement layer if there is no packet loss in the base layer. If base layer is also lost, it needs to generate the MVs using TD method before BLSkip can use motion field upsampling.

- Reconstruction Base Layer Upsampling (RU). This is an inter layer error concealment method. In the "reconstruction base layer upsampling" (RU) algorithm, the base layer picture is reconstructed and upsampled using the H.264/AVC 6-tap filter for the lost enhancement layer picture. If base layer packet is also lost, the FC method is used for the enhancement layer, rather than using an upsampled concealed base layer picture.

In the SVC reference software JSVM (Joint Scalable Video Model), BLSkip, Frame Copy and temporal direct mode are supported.

8.2.2 Packet Priority in H.264/SVC

The Priority Forward Error Correction (PFEC) was first proposed by (Albanese et al., 1994). Each packet is assigned a priority which determines the minimal number of codeword projection that is required to recover that packet. To make use of PFEC, it is important to know the priority of the packets in the video bitstream. For H.264/SVC bitstream, the scalability structure is defined by three syntax elements as mentioned before: *dependency_ id*, *quality_ id*, and *temporal_ id*.

However, although those three variables can provide scalable information of the bitstream, no assumption on a relation between the priority of the packets and the values of *dependency_ id* (D), *quality_ id* (Q), or *temporal_ id* (T) is explicitly made in the SVC draft. Therefore, the *priority_ id* syntax element has been introduced to indicate the priority of the NAL units. In (Amonou et al.,

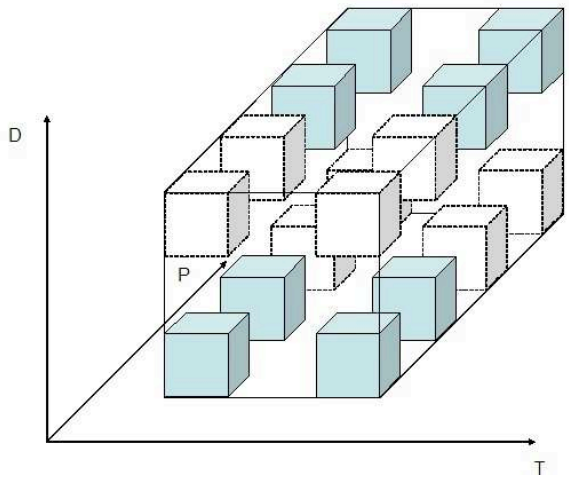


Figure 8.4: (D,T,P) graphical representation of an SVC stream. White cubes represent empty data sets (extracted from (Amonou et al., 2006)).

Scalable level	# Packets	# Packets percentage	# Bytes	# Bytes percentage
L0	452	63%	119380	42%
L1	266	37%	278221	58%
T0	288	32%	216917	45%
T1	47	7%	30159	6%
T2	77	11%	49363	10%
T3	119	17%	64821	14%
T4	167	23%	69633	15%
T5	80	11%	46708	10%

Table 8.1: Packet statistic of *football* video sequence

2006), the authors presented some considerations about the use of the *priority_id* syntax element. A 3D graphical representation (D,T,Q) of an SVC stream is used, as shown in Figure 8.4.

The constraint proposed is: the *priority_ids* must be assigned in such a way that any subset of that stream which includes all NALUs with *priority_id* $\leq$ P, *dependency_id* $\leq$ D, *temporal_level* $\leq$ T, where (P,D,T) are integer values, must also be a valid stream.

Analysis of the bitstream sample To further study the property of the scalable priority of the SVC bitstream, we use the *football* video sequence as a case study to simulate the effect of packet loss in different layers. This will give us a more straight view of the priority of the SVC stream.

To analyze the scalable information of the SVC video, the *.yuv* raw file is coded into H.264/SVC bitstream by using JSVM (more details about the video codec configuration can be found in section 9.3 on page 161).

Then we use the *BitstreamExtractor* to generate the packet trace. The number of packets and bytes in each scalable level is shown in Table 8.1. From the table we can see how the bitstream is composed by different scalable layers.

It is necessary to point out that in the original packet trace, it includes type-14 NAL units with length in just 9 bytes. In H.264/AVC, NAL unit types 14, 15 and 20 are reserved for future extensions. In H.264/SVC, according to (Wenger et al., 2010), NAL unit type 14 is used for *prefix*

Scalable level	# Packets	# Packets percentage	# Bytes	# Bytes percentage
L0	229	47%	197828	42%
L1	263	53%	276801	58%
T0	198	40%	217389	45%
T1	25	5%	29966	6%
T2	45	9%	48986	10%
T3	60	12%	64420	14%
T4	84	17%	67447	15%
T5	80	16%	46421	10%

Table 8.2: Packet statistic of *football* video sequence without type 14 NAL units

NAL unit. It is a NAL unit with *nal_ unit_ type* equal to 14 that immediately precedes in decoding order a NAL unit with *nal_ unit_ type* equal to 1, 5, or 12. The NAL unit that immediately succeeds in decoding order the prefix NAL unit is referred to as the associated NAL unit. In the internet draft (Wenger et al., 2010), it mentioned that "*Although the prefix NAL unit is ignored by an H.264/AVC decoder, it is necessary in the SVC decoding process. Given the small size of the prefix NAL unit, it is best if it is transported in the same RTP packet as its associated NAL unit*".

However, in our cases, removing those type 14 NAL units has no effect on the quality of video. So those NAL units (packets) are removed from the packet trace. Table 8.2 shows the statistic of the packets from different scalable level after removing the type-14 NAL units. From the spatial point of view, the coded bitstream has two spatial layers, each of them has about 50% over the total number of packets. From them temporal point of view, there are five temporal layers. The *T0* layer packets has 40%, which are all marked as *non-discardable* packets, i.e. the packet loss from *T0* will make the decoding impossible.

Packet loss simulation To confirm the priority of different scalable layers, a packet-loss simulation is designed. A packet-loss simulator called *ErrorPatternGenerator* is made so that we can define the packet loss from a specified scalable layer (temporal, spatial or quality).

An example of *ErrorPatternGenerator* is shown as following:

ErrorPatternGenerator p 2 *packet_ trace_ filename.trace*, t1, 20

The command means that 20 packets from scalable level t1 are dropped. The random seed 2 is used. For more details about the usage of the *ErrorpatternGenerator,* please refer to Appendix C on page 181.

The PSNR is measured to compare the packet loss from which layer has more impact on the video quality, which corresponds to higher priority.

The results from our simulation reveal that the *temporal_ id* could best classify the impact of packet loss to the video quality by giving the better hierarchy. A layer with a smaller *temporal_ id* has a lower frame rate than the one with a larger *temporal_ id value*. A given temporal layer depends on the lower temporal layers, but does not depend on any higher temporal layers.

Figure 8.5 presents the impact of packet loss from different temporal layer to the quality of the video (*t1* stands for the packet loss from layer with *temporal_ id* equals 1, etc.). As shown in the figure, with the same percentage of packet loss (over total packets) from different temporal layers, the packet loss from *t1* has the most impact on the video quality, and then is the *t2*, etc. The packet loss from t4 and t5 has the least impact.

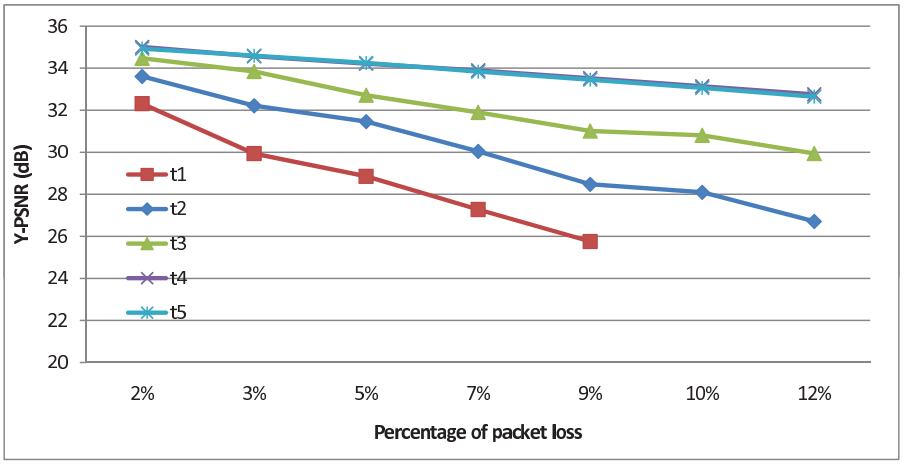


Figure 8.5: The impact of packet loss from different temporal layers to the quality of the *football* video

The results indicate that with our current configuration of JSVM codec, the *t1* packets have the highest priority, and the *t4* and *t5* packets have the lowest priority. It is necessary to point out that this simulation does not intend to give a general conclusion of the packets priority in H.264/SVC, but to give a clue of the priority in the current scenarios. This hierarchy will be used in the following chapter for priority FEC coding.

8.3 Conclusion

In this chapter, we introduced the transport interface of the H.264/SVC. It extended H.264/AVC to support different kinds of scalability: *temporal, spatial* and *quality* scalability. For the transmission of the H.264/SVC over networks, the RTP payload for SVC is also briefly introduced. The structure of the NAL unit is extended to 3 bytes to include scalable information.

Then the error concealment algorithms in H.264/SVC are presented. Currently, there are BLSkip, *frame copy* and *direct mode* supported by JSVM reference software. Based on the error concealment, we made an *ErrorPatternGenerator* to launch the packet-loss simulation. The simulation results show that with the current configuration, the *temporal_id* can be used to define the priority of the packets: the packets with lower *temporal_id* have higher priority, i.e. the loss of those packets has more negative effect on the quality of the video.

Chapter 9

Multipath Transportation for H.264/SVC

In Chapter 6, we have simulated MP-OLSR with regular CBR applications. The results show that the MP-OLSR can provide higher packet delivery ratio and lower delay in most of the scenarios. In this chapter, we are interested in the H.264/SVC video services over the ad hoc routing protocols.

Firstly, we build an evaluation framework called SVCEval to simulate the video traffic with Qualnet simulator. It can be adapted to different kinds of networks. Then we propose to use FRT for PFEC to protect the data with higher priority. A PFEC codec simulator is also integrated into SVCEval framework. Finally, the related simulations are presented.

9.1 Video Evaluation Framework

9.1.1 Introduction of Video Evaluation over Networks

To evaluate the quality of the video transmission over a specified network, the most straightforward method, of course, is to build such kind of network and evaluate the video transmission over it. However, this will be costly and time consuming. So most of the time, network simulations are employed for the evaluation.

From the network point of view, a lot of network simulators are widely used, such as NS2 and Qualnet. And for the video, there are generally three ways to characterize an encoded video for network simulation: *video bitstream*, *video traffic mode*l and *video trace*.

- Video bitstream contains the complete video information, and is generated by using the encoder. This can be used in both network experiments and the network simulations. However, the bitstream is generally very large in size. Another limitation is that most of the video are protected by copyright, which limits the researcher to access those resources.

- Video traffic models are developed based on the statistical properties of samples of the real traffic. They have received a lot of attention in the research (Heyman & Lakshamn, 1996). Normally the model requires only a small number of parameters to describe the video traffic in an accurate and computationally efficient way. If the model is accurate enough, it can be used for network simulations to generate virtual video traces. However, this kind of video

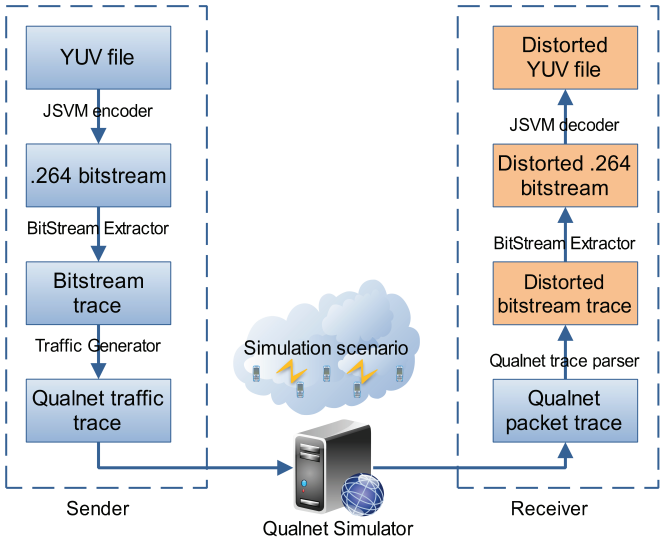


Figure 9.1: The evaluation framework *SVCEval* for H.264/SVC

traffic model for H.264/SVC is still an open topic, and the virtual video trace is not easy to be decoded for video quality evaluation.

- Video traces are an alternative to the two methods mentioned above. Instead of containing the actual bits that carries all the video information like the bitstream, the video traces contain just the most important information as discrete events (i.e. packet size and transmission time) for network simulation and encoding. Most of the time, it includes the size of the packet that need to be transmitted and the corresponding quality information (e.g. the scalable layer information in the H.264/SVC). The video trace for network simulation is very attractive because it can accurately present the video traffic and there is no copyright issues. In (Seeling et al., 2004) the authors present a library of video traces. And in (Klaue et al., 2003) an evaluation framework for MPEG and H.264/AVC is presented to evaluate the video quality over NS2 and tcpdump traces.

Currently, most of the study in H.264/SVC video quality and error concealment such as in (Guo et al., 2009) and (Ma et al., 2009) is based on error patterns (Wenger, 2002; Guo et al., 2006). This scheme can define a specified loss rate in the bitstream and it is very useful and efficient for the error resilient study. However, it is not sufficient if one wants to simulate the video transmission over a specified network.

9.1.2 *SVCEval* Evaluation Framework

To evaluate the H.264/SVC transmission over different kinds of networks, especially ad hoc networks, we proposed a evaluation framework *SVCEval* as shown in Figure 9.1. It is based on the SVC reference software JSVM and makes use of the Qualnet simulator for the network simulation.

Operation in the video source

At the video sender, the YUV file is encoded by the JSVM encoder, and the H.264 bitstream is generated.

Start-Pos.	Length	LId	TId	QId	Packet-Type	Discardable	Truncatable
0x00000000	262	0	0	0	StreamHeader	No	No
0x00000106	14	0	0	0	ParameterSet	No	No
0x00000114	16	0	0	0	ParameterSet	No	No
0x00000124	8	0	0	0	ParameterSet	No	No
0x0000012c	9	0	0	0	ParameterSet	No	No
0x00000135	9	0	0	0	ParameterSet	No	No
0x0000013e	9	0	0	0	SliceData	No	No
0x00000147	1425	0	0	0	SliceData	No	No
0x000006d8	9	0	0	0	SliceData	No	No
0x000006e1	1470	0	0	0	SliceData	No	No
0x00000c9f	9	0	0	0	SliceData	No	No
0x00000ca8	1497	0	0	0	SliceData	No	No

Figure 9.2: An example of H.264/SVC bitstream trace (packet trace)

```
100.0MS 262
100.0MS 14
100.0MS 16
100.0MS 8
100.0MS 9
100.0MS 9
100.0MS 9
100.0MS 1425
100.0MS 9
100.0MS 1470
100.0MS 9
100.0MS 1497
```

Figure 9.3: An example of Qualnet application trace

Then the *BitStreamExtractor* (provided by JSVM) is used to generate the bitstream trace from the given bitstream with the option *-pt*:

```
BitStreamExtractorStatic -pt video_trace.trace input_video.264
```

The bitstream trace generated is a text file which specifies the parameters of each packet inside the bitstream. Those parameters include the start position of the packet inside the bitstream, the length of the packets, the values of *dependency_id* (LId), *temporal_level* (TId) and *quality_level* (QId) for the packet, the type of the packet and two flag which indicate whether the packet is *discardable* or *truncatable*. An example of the bitstream trace is illustrated in Figure 9.2.

To make the Qualnet be able to read the bitstream trace, we created a *QualnetTrafficGen* to generate the input traffic trace file for Qualnet simulation, which includes mainly the packet rate and size. The simulator will take the traffic trace file and run the simulation according the configuration of scenarios to simulate different kinds of networks. The command is like:

```
QualnetTrafficGen <rate_in_packets/s> video_trace.trace qualnet_traffic_trace.trc
```

The output of *QualnetTrafficGen* is the input for Qualnet simulation. This trace file has two columns: the first for time interval between subsequent data, and the second for each data length, as shown in Figure 9.3.

Operation in the video destination

At the video receiver, a packet trace file is produced by the simulator. The packet trace file records all the operations on each packet in each node and each layer (so normally hundreds of MBytes) in an XML file.

A *QualnetTraceParser* is developed to analyze the trace to detect which packets are lost and which packets are properly received. For real-time transmission, we can set a delay threshold to discard the packets that timed out. The command is like:

```
QualnetTraceParser <source_id> <destination_id> <protocoly_type> <delay_threshold>
                   <input_qualnetTrace> <input_BistreamTrace>
```

More details about the usage of the QualnetTraceParser, please refer to Appendix D on page 183.

The trace parser can generate the distorted bitstream trace, and then we use the BitstreamExtractor and JSVM decoder to have the distorted YUV file after the video transmission:

```
BitStreamExtracotrStatic original_video.264 distorted_video.264 -et distorted_trace.trace
```

Then we can evaluate the quality of video with different metrics such as PSNR or Mean Opinion Score (MOS). The PSNR tool is provided by JSVM:

```
PSNRStatic <width> <height> <original_file> <distorted_file>
```

The *SVCEval* framework can be adapted to different kinds of networks. Figure 9.4 presents an example of the quality of video transmission in various network scenarios: wired network, MP-OLSR static mesh/ad hoc with 250m transmission range, and MP-OLSR static mesh/ad hoc with 200m transmission range. We can conclude from the figure that the peroframcne of wired networks is much bettern than ad hoc networks (up to 5dB). And in the scenarios with mobility and low connectivity (range of 200m), the quality becomes unacceptable (PSNR below 25 dB). So it will be interesting to use FEC coding to protect the data during transmission. In section 9.3, the performance of different protocols in ad hoc networks will be further studied based on *SVCEval* framework.

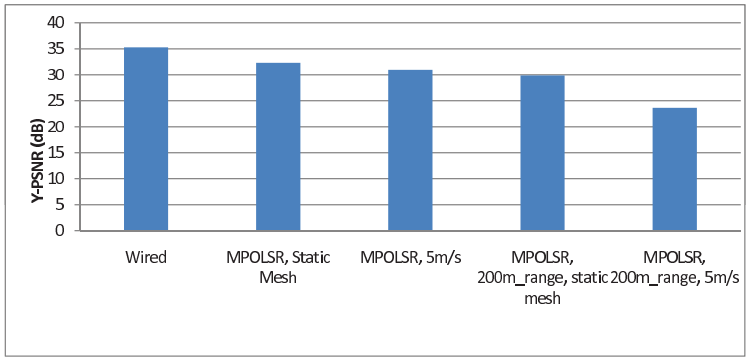


Figure 9.4: *SVCEval* in different networks

9.2 Forward Error Correction for H.264/SVC

9.2.1 FRT for Priority FEC

As mentioned in the previous section, a wide range of scalability (spatio-temporal and quality) can be achieved by using SVC. It allows removal of parts of the bit-stream and still get reasonable

coding efficiency with reduced temporal, spatial or SNR resolution. This feature is very attractive for unstable network transmission because we can focus on the more important scalable layers to improve the final video quality. This can be achieved by using Priority Forward Error Correction Coding (PFEC).

FRT and Redundancy

We make use of FEC code based on Finite Radon Transform (FRT) for PFEC. As introduced in section 3.2 on page 59, it is a discrete data projection methods that are exactly invertible and are computed using simple addition operations. FRT differs significantly from Mojette transform by providing equal size projection (packets).

In (Normand et al., 2010), our team proposed to employ FRT for FEC coding by using the concept of *ghost functions* in image coding. These are functions made of positive and negative values (or zeros for the trivial ghost function) which can exist in the image but will disappear in the projection data as they sum to give zero in that direction; hence the term *ghost functions*.

Incorporation of a known level of redundancy into data and projection spaces enables the use of forward error correction to recover the exact, original data when network packets are lost or corrupted during data transmission. Figure 9.5 shows data packed into a $k \times (p–1)$ (p is always prime) subset of a $p \times (p–1)$ data. The content of the $r \times (p–1)$ redundancy area is designed so that the full image projections sum to zero for r sequential rows of the FRT space. If r or less data rows are missing, the the projected sums in the FRT contain an exact copy of the missing data. The data can be recovered from its projected form. For detailed information, please refer to previous work in (Normand et al., 2010).

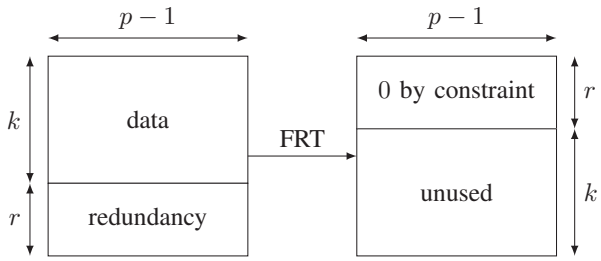


Figure 9.5: Systematic FRT encoding

PFEC over multipath routing

Compared to the equal forward error correction, which applies equal redundancy to all the packets and increases the overhead significantly, the PFEC can give a good balance between the error correction and network load by focusing on the most important packets. For packets of layer L_i, $(i = 1, 2, ...N)$ in a bitstream of N layers, we can apply function $PFEC(b_i, n_i, k_i)$, where b_i is the number of packets to be buffered for encoding procedure, n_i is the total number of coded projection, and k_i is the number of packets needed for the reconstruction of b_i packets.

The priority of the packets P_i in scalable layer L_i can be defined according to the constraint or packet loss simulation introduced in section 8.2.2 on page 150, in which the priority of different layers is studied. P_j has higher priority than P_k if $P_j \leq P_k$, $(j, k = 1, 2, ...N)$.

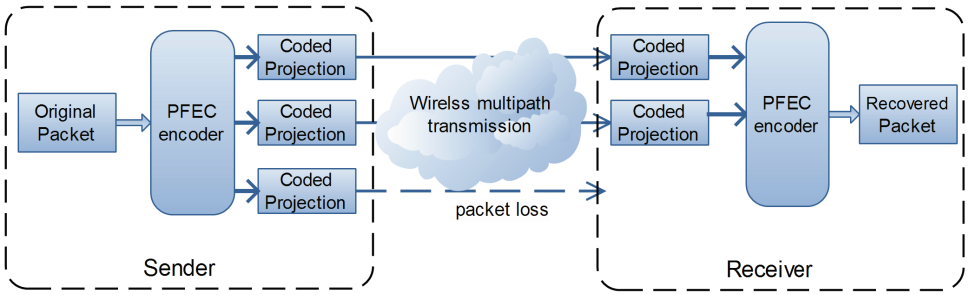


Figure 9.6: Mulitpath transmission with PFEC

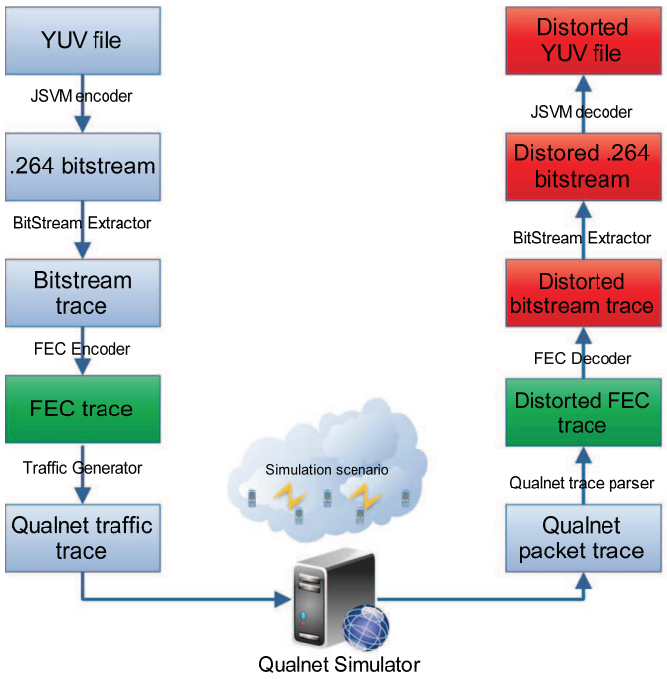


Figure 9.7: The *FECSim* framwork for FEC simulation

With FRT, the packets with higher priority can be assigned with higher redundancy. If $P_1 \leq P_2 \leq ... \leq P_N$, then we should have $\frac{n_1}{k_1} \geq \frac{n_2}{k_2} \geq ... \geq \frac{n_N}{k_N}$. The coded projections will be distributed into disjoint multiple paths. So even when less than $n_i - k_i$ of the projections are lost in L_i because of route failure, it is still possible to recover the original packet, as illustrated in Figure 9.6.

9.2.2 The Priority FEC Evaluation Framework

To evaluate the video transmission over network with Priority FEC coding, an FEC simulator called *FECSim* is integrated into *SVCEval* framework, as shown in Figure 9.7.

It mainly consists of two parts: in the sender, the *QualnetFECTrafficGen* plays as a FEC encoder; in the receiver, the *QualnetFECTraceParser* plays as a FEC decoder.

QualnetFECTrafficGen

The FEC traffic generator simulates the behavior of FEC encoder. It reads the original bitstream packet trace and the configuration of the priority FEC:

t1 2 3 2 s
t2 2 3 2 s

The configuration means that for temporal layer *t1* and *t2*, the encoder will buffer 2 data packets, and generate 3 coded projections with the same size. In the receiver, 2 projections in 3 are needed to recover the original packets. The *s* means that systematic code is used.

The output of the *QualnetFECTraffic Gen* is mainly a traffic file for Qualnet simulation and a FEC traffic trace which will be used for decoding process. For more details about the *QualnetFec-TrafficGen*, please refer to Appendix E on page 185.

QualnetFECTraceParser

The FEC trace parser simulates the behavior of FEC decoder. It reads the Qualnet packet trace and generates the distorted bitstream trace based on the FEC trace (generated by *QualnetFEC-TrafficGen*). For more details about the *QualnetFECTraceParser*, please refer to Appendix E on page 185.

9.3 Performance Analysis

9.3.1 Test Conditions and Network Scenario

To demonstrate the performance of multipath routing of the H.264/SVC video transmission over ad hoc networks, we performed the simulated based on the evaluation framework proposed. The *football* sequence with high and irregular motion is used as a sample. The configuration of the JSVM codec is as follows.

- JSVM 9.8.
- Two layers with based layer QCIF@30Hz and enhancement layer CIF@30Hz.
- Group of picture size is 16.
- SliceMode is set to fixed number of bytes per slice, with *SliceArgument* set to 1000, the date rate is 720 kbps (120KB/s).

The PFEC scheme is applied to the video stream. The layers with *temporal_ id* 1 and 2 are encoded using systematic code. Each time at the sender, the coder will buffer 2 packets, and generate 3 projections. At the receiver, the decoder needs 2 projections to recover the original packets. The rest of the layers are not coded and transmitted in its original form (i.e. not protected). The layer with *temporal_ id* 0 is regarded as *non_ discardable* packets, so we assume those packets are transmitted along reliable channel. The overhead by using this configuration is about 15%.

The simulation is taken in Qualnet with 81 nodes in a 1500×1500 area. The detailed parameters are already presented in section 6.2.3.2. As shown in Figure 9.8, one source-destination pair is chosen as video sender and receiver. Three other CBR sources are defined as background traffic.

Quality

To evaluate the video communication systems, it important to determine the quality of the video images displayed at the endpoints. For communication, the measurement of QoS (Quality of

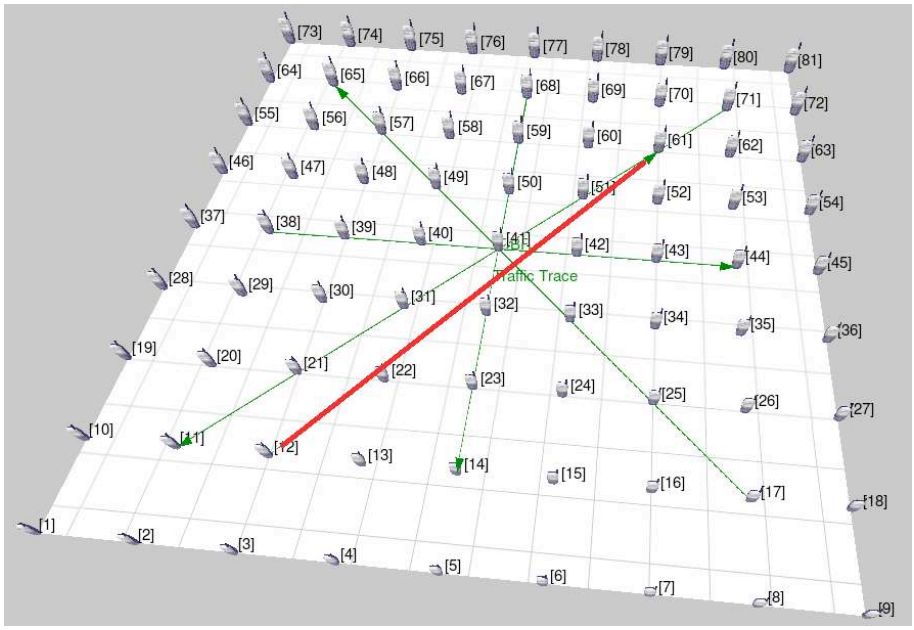


Figure 9.8: Network scenario with video tream (red line) and backgournd traffic (green lines)

Service) is commonly used relied on determining the quality of the underlying network. However, for video evaluation, we are also interested in QoE (Quality of Experience): it is a high level of quality to users, which considers the overall experience the consumer has when accessing and using provided services. Video quality measurement is a difficult task because it can be affected by a lot of factors. Those methods to evaluate the video quality can be divided into objective methods and subjective methods.

Objective video evaluation techniques are mathematical models that approximate results of subjective quality assessment, but are based on criteria and metrics that can be measured objectively and automatically evaluated by a computer program.

Besides the objective methods, many "subjective video quality measurements" are described in ITU-T recommendation BT.500. Their main idea is the same as in Mean Opinion Score for audio: video sequences are shown to the group of viewers and then their opinion is recorded and averaged to evaluate the quality of each video sequence.

In our experiments, one of the most traditional methods is used, which calculats the peak signal-to-noise ratio (PSNR) between the original video signal and signal passed through the network system.

9.3.2 Simulation Results

Figure 9.9 compares the delivery ratio of OLSR and MP-OLSR with or without PFEC coding. The four configurations have almost the same performance at low speed, but the delivery ratio of single path routing (OLSR and OLSR FEC) decreases quickly as the mobility increases. This is because as the links become more unstable, the MP-OLSR could take benefits from the multipath routing.

The PFEC could slightly increase the delivery ratio of MP-OLSR (about 2%), but not significant for OLSR. This is because: firstly, in the network, the packet loss is continuous most of the time because of congestion or route failure. If the single path routing is applied, all the projections from

the coded packets are lost continuously in the same route, and FEC is not helpful in this situation. With multiple paths, the projections are distributed in the disjoint paths and forwarded to the destination independently. The FEC can still work even some of the routes failed as illustrated in Figure 9.6.

Secondly, it is inevitable that the FEC coding will increase the network load even when priority coding strategy is employed because the redundancy is added in the packets to protect the data. This will increase the packet loss and maybe results in worse video quality in the end for single path routing (for example, the 5m/s and 6m/s for OLSR FEC). This problem is less serious for multipath routing because it can provide higher overall bandwidth.

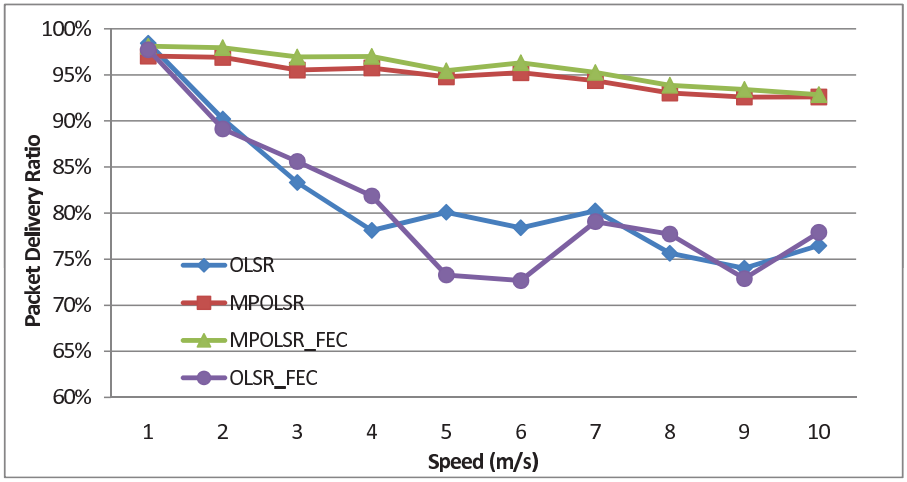


Figure 9.9: The delivery ratio of different protocols (with or with out FEC code)

Figure 9.10 compares the quality of the video transmission by using different kinds of protocols. Compared to OLSR, MP-OLSR has worse quality in very low-mobility scenarios (1m/s and 2m/s). As the node speed increases, the quality of OLSR drops quickly and MP-OLSR outperforms OLSR. This result is consistent with the conclusion from our previous work that the single path routing might have better delivery ratio than MP-OLSR in the network with very less topology changes. However, in these low-mobility scenarios, the MP-OLSR can make use of single path also because the MP-OLSR is compatible with OLSR.

Although the improvement of the MP-OLSR with priority FEC coding (MPOLSR FEC) in delivery ratio is not obvious, the MPOLSR_FEC can effectively improve the video quality by 2 dB on average, which is a significative improvement. Figure 9.11 shows a case study of the packet delivery ratio of different temporal scalable layers. The improvement of the protected layers (T1 and T2) is about 7%, and the improvement of the unprotected layers is not obvious (about 2%). So the packets with high priority (temporal id equals 1 or 2) are better protected with PFEC.

The overhead produced by PFEC is 15% with our configuration. If the Equal Error Protection (EEP) is applied, the overhead will be up to 50%. In fact, the simulations of EEP are also taken, but did not provide significant improvement (even worse in high mobility scenarios). Although the EEP offered more redundancy, it also injected more load to the network, which will bring down the network performance. Our previous work (Parrein et al., 2007) proved that with the same redundancy, PFEC can also provide better user experience than EEP.

Figure 9.12 presents the screenshots of three frames from the scenario with max speed of 4m/s. The MP-OLSR with PFEC provided the best video quality and OLSR suffers from the most packet

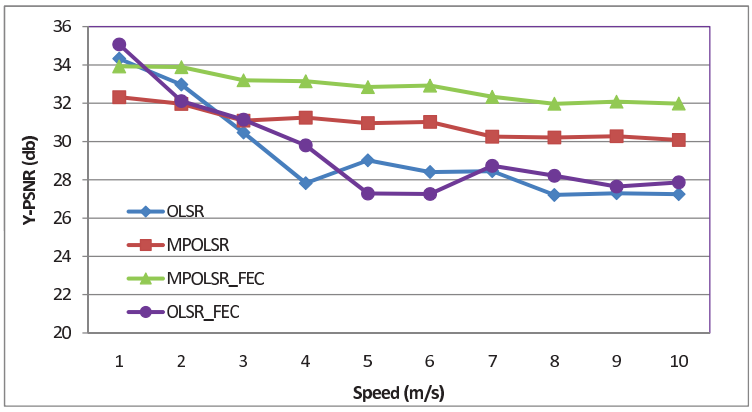


Figure 9.10: The quality of video transmission through the different protocols (with or without FEC code)

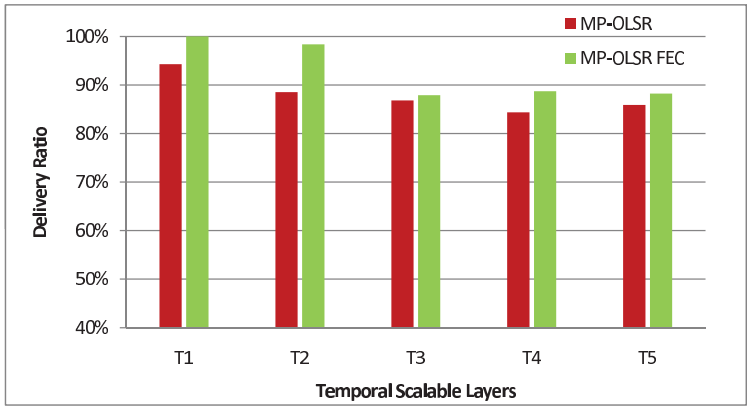


Figure 9.11: Packet delivery ratio of different temporal scalable layers

loss. The frames displayed by OLSR are delayed because the frame copy error concealment method is used. In this way, the previous frames are copied if there is too much packet loss.

More video sequences are also tested. Figure 9.13 and 9.14 show the Y-PSNR and the screenshots of the *foreman* sequence respectively. The quality of the video shares the same trend as the *football* sequence, and with about improvment of 2 dB. But with less movement and more regular changes, we can obtain better PSNR than the *football* sequence.

9.4 Conclusion

In this chapter, the multipath transmission of H.264/SVC over ad hoc networks is studied. We build an evaluation framework called *SVCEval* to simulate the H.264/SVC on Qualnet. It provides great flexibility for evaluating video transmission over various kinds of networks. Then we proposed to use priority FEC coding based on FRT to protect the scalable layers with higher priority. A FEC codec simulator *FECSim* is also integrated into the *SVCEval* framework. The simulation results reveal that multipath routing can effectively improve the quality of video transmission over ad hoc networks. And priority FEC coding can provide better quality (by 2 dB in Y-PSNR in our experiment scenarios) by protecting the layers with high priority with reasonable costs.

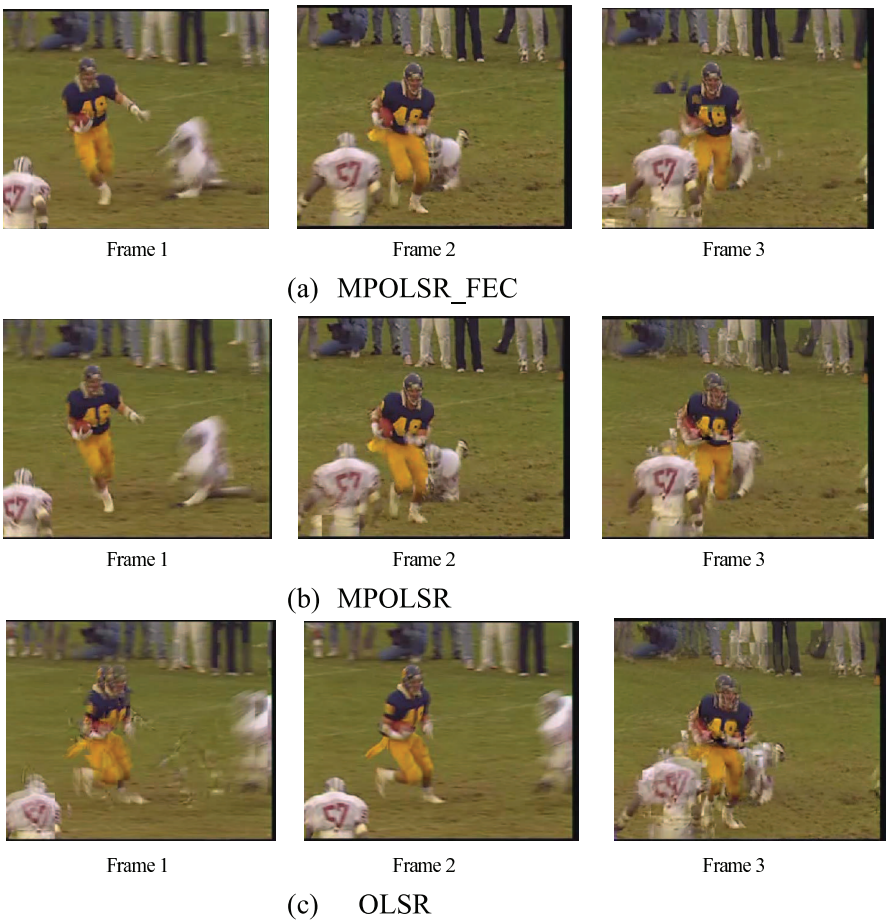


Figure 9.12: Screenshots of the *football* video sequence

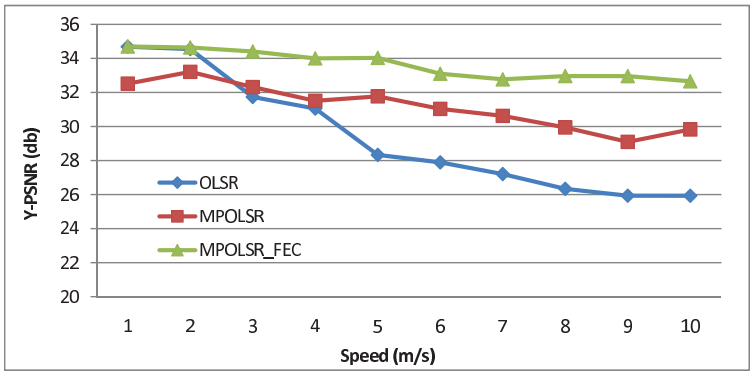


Figure 9.13: The quality of *foreman* video transmission of different protocols

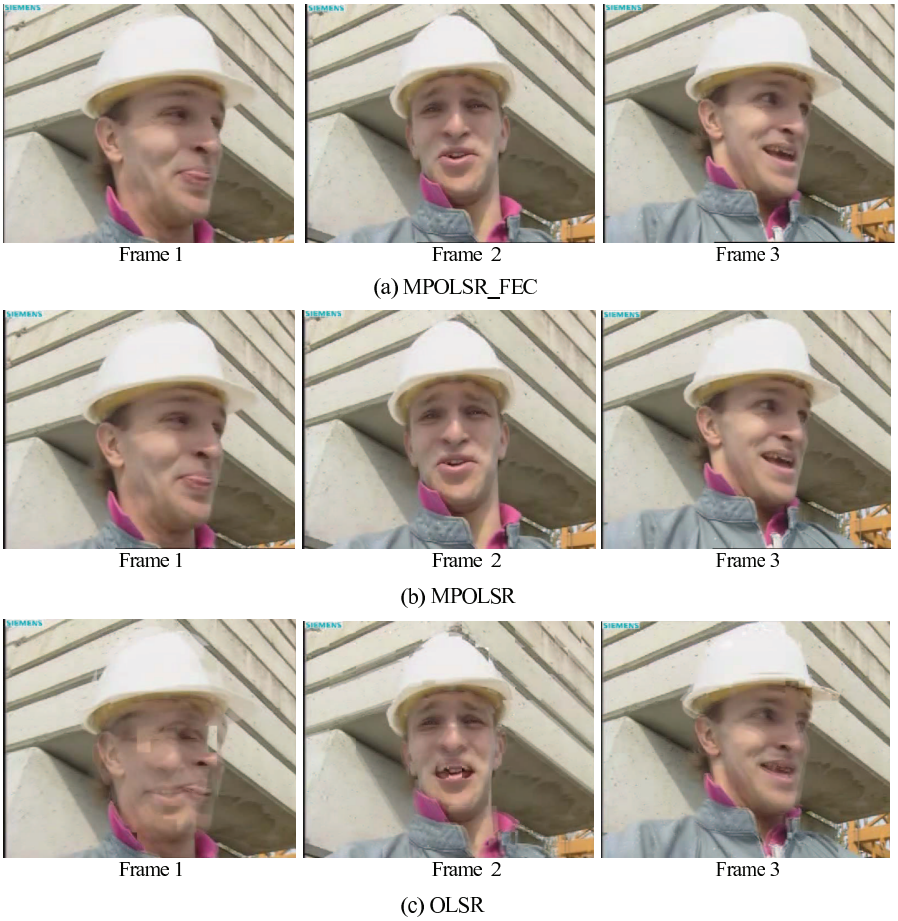


Figure 9.14: Screenshots of the *foreman* video sequence

Conclusion

Summary of Contributions

Mobile ad hoc networks are well known for their dynamic topology, which gives great challenges for routing protocol in the networks. In our work, we try to address the problem of routing data in this kind of networks. Based on OLSR, we proposed Multipath Optimized Link State Routing as a multipath extension.

The core function of MP-OLSR has two parts: *topology sensing* and *route computation*. The *topology sensing* inherits from OLSR, which based on the periodical HELLO and TC messages. Through the broadcasting of HELLO messages to the 1-hop neighbors, the node could get the information of the 1-hop neighbors and 2-hop neighbors. In the meantime, the Multipoint Relays (MPR) and related MPR selectors are chosen such that there is at least one path to each of its 2-hop neighbors via an MPR. The TC messages are flooded to the whole network through MPRs, which effectively reduces the broadcasting traffic in the network. On receiving the TC messages, the nodes can build the topology information base of the links that are more than two hops away. For *route computation*, we proposed the Multipath Dijkstra algorithm to generate multiple paths from the source to the destination. Two cost functions are used to get the node-disjoint paths or link-disjoint paths as necessary. The source route is saved in the header of the data packets and guides the packets to be forwarded from the source to the destination.

In a perfect world, the paths generated by the core function will always be available and loop-free. However, in practice, due to the frequent changes of the topology, the algorithm suffers from route failures and transient loops. So we proposed *route recovery* and *loop detection* to increase the packet delivery ratio and decrease the end-to-end delay. *Route recovery* checks the related link before forwarding the packets to the next hop. If the link is not available, the route will be recomputed. *Loop detection* makes use of the source route to check the possible loops during the transmission and tries to redirect the packets out of the transient loops. Those functions are performed inside the local node with low complexity, so they will not introduce extra network load and delay.

The original definition of OLSR does not consider the link quality, and simply measure the paths by number of hops. We proposed to use queue length as the link metrics for route decision. The queue length in the local node is monitored and broadcasted to the whole network with HELLO and TC messages. Thanks to the extensibility of OLSRv2, the queue length information can be easily integrated as a TLV into the routing messages. On receiving the queue length information, it is normalized into the link cost for Multipath Dijkstra Algorithms.

The backward compatibility of MP-OLSR with OLSR is also considered, which will facilitate the application of deployment of the multipath routing. Two possible solutions: strict IP source

routing and loose IP source routing are proposed to make the MP-OLSR nodes and OLSR nodes be able to cooperate with each other.

To validate the efficiency of the MP-OLSR, the simulations are performed. In the first phase, NS2 is used. The results show that the *link layer notification* and *route recovery* are important for improving the packet delivery ratio of the protocol. More simulations are taken in Qualnet simulator afterwards because of some of the drawbacks of NS2. From the simulation results of Qualnet, we can conclude that:

1. The OLSR and MP-OLSR has almost the same performance in the scenarios with low data rate and mobility. MP-OLSR outperforms OLSR in the error-prone networks with high network load and topology changes.

2. The MP-OLSR can effectively find the multiple paths and distribute the traffic into those paths to release the load on the unique path.

3. The *route recovery* is important for reducing the route failures of source routing. The *loop detection* can avoid possible loops in the network. Both of them can effectively improve the packet delivery ratio and end-to-end delay.

4. The queue length link metric can be helpful to avoid congestion nodes when routing the data packets.

5. The MP-OLSR protocol is compatible with OLSR. The nodes can cooperate with each other in the same network. Higher density of MP-OLSR nodes can have better performance.

Because there are still some limitations of the simulation tools, especially in the modeling of physical channel, a testbed was build to validate the protocol in a real environment. Compared to the simulation, the deployment and experiments in the testbed were more complex. In cooperation with the partners from the industry and academic fields, the tests were taken in the campus of Polytech'Nantes, France. The results prove that the multipath routing protocol can provide more stable data transmission than the single path protocol. The related source code for the simulation and testbed has been released online and receives good feedback.

Other than the regular data transmission, we are also interested in the video services over ad hoc networks. In our study, the H.264/SVC video, which is a scalable extension of the latest H.264/AVC standard, is considered. An evaluation framework, called *SVCEval* is build to simulate the SVC video over various kinds of networks. Compared to the evaluation methods based on bitstream or traffic models, our framework has good balance between the simulation cost and flexibility. To improve the quality of the video, a Priority Forward Error Correction coding scheme based on Finite Radon Transform is proposed to protect the scalable layers with higher priority. A simulator of the codec, *FECSim*, is integrated into the *SVCEval* framework. The simulation results reveal that the MP-OLSR with PFEC can improve the quality of video transmission significantly.

Future Work

In our work, we have proposed an effective multipath routing protocol for ad hoc networks and its possible application for video services. However, there is still much work to be done in the aspect of routing protocol and video services.

The link metric to be used in MANET is still a very large topic and needs to be further studied. We have proposed to use queue length as link metric, but its performance in a real network scenario still need to be validated. Another issue closely related to the link metric is the path selection. The cost functions for Mulitpath Dijkstra algorithm need to be carefully chosen according to the link metric to get the appropriate path set.

The MP-OLSR provided better performance than OLSR in the simulations and testbed. However, due to the fragile physical layer, the overall performance of the ad hoc network is still not satisfying the demand for real application because of the high packet loss ratio and low data rate. To ensure the reliable transmission over multiple hops, a more powerful physical layer and a cross-layer strategy are needed.

We are also planning to push the protocol to the standardization procedure. So more technical details have to be refined, such as the number of the paths, cost function, source routing implementation, etc.

For the video services, it will be important to optimize the traffic distribution according to the upper layer scalability information and the lower layer link quality. How the redundancy of the FEC coding should be allocated to ameliorate the quality of video transmission is also the subject of future research.

With the development of the wireless communication, more people and devices will be connected to the network in the future. The ad hoc network will be an attractive domain to provide this kind of “anytime, anywhere” services. The developing wireless technology and requirement of inter-connection will lead to the next innovations in ad hoc networks.

Appendix A

Study of the queue length in NS2

To study the scalability of NS2, the scenarios with high data rate up to 100 packets/second (400Kbps) are also taken. The results reveal that the MP-OLSR has longer delay than the OLSR. It is abnormal because the MP-OLSR is expected to have better performance by distributing the traffic into different paths. So a research focused on the length of the interface queue (IFQ) is taken to study the end-to-end delay in detail.

A.1 Implementation of IEEE MAC 802.11 and the cross-layer information in NS2

For a single node, the simulation of the ad hoc networks based on IEEE 802.11 with NS2 has several components below:

- Channel: The *WirelessChannel* simulates the physical media (air) for wireless communication. It keeps a list of all nodes on this channel (mostly, all the nodes participate the simulation). The list is sorted based on the X-dimension values of nodes before it can be used.
- Network Interface: In the class *WirelessPhy*, NS2 simulates the wireless physical layer. Besides working for receiving and sending packets as any other physical layer, it deals with the propagation model (free space, two ray ground or shadowing), node sleeping (duty cycle management), energy model management and maybe different antenna models and modulations. It also has the class *MobileNode* to trace the mobility of nodes.
- Media Access Control.
- Outgoing Queue: The only target of the outgoing queue is MAC. It might has many variations such as droptail, priqueue, etc.
- Link Layer: It has a composed component ARP which works as ARP procedure, mapping the protocol address (such as IP address) to hardware address (such as MAC) address.
- Network Layer: Routing agents with many variations such as DSDV, DSR, AODV, etc. The MPOLSR is also located in this layer.
- UDP Agent Layer: Generates the data traffic in different manners, such as CBR or FTP.

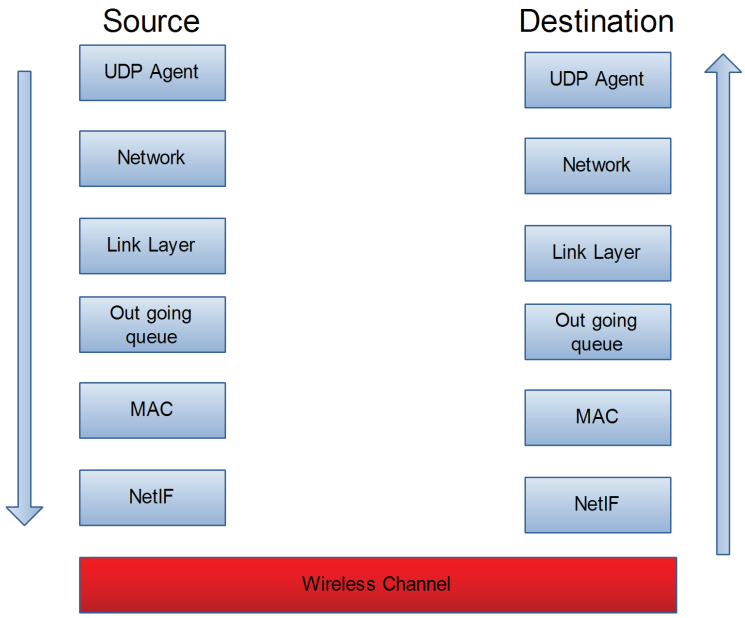


Figure A.1: Different layers in NS2 simulation

As illustrated in Figure A.1, for each packet transmitted from the source to the destination, it has to pass all the layers in NS2.

In all those layers, the Outgoing queue can reflects the congestion in the network and has the most important impact on the delay. To study the queue length in NS2, the first task is to obtain the queue length information at the routing layer. Because the Outgoing queue does not rely in the same layer of the routing protocol (network layer), it is necessary to modify the NS2 following Algorithm A.1 to enable cross layer information interaction in NS2. Then we can get the lower layer method by calling the method of the lower layer objects. For example, to get the length of the queue, just use *ifq->prq.length()*.

A.2 Simulation Results

The simulations mainly follows the parameters used in Table 6.1. The only change is that we chose the static scenario and single data source to make the packets easy to trace. The distance from the source node to the destination node is about 5 hops. The length of the queue at each node is recorded when it is trying to forward a data packet. In this way, the behavior of the length of the queue can be monitored. In this simulation, the max queue length is set to 50 packets, which means that when the new arrival packets will dropped if the queue is full.

Figure A.2 and Figure A.3 show the length of the queue at the source node in MP-OLSR and OLSR. The source node data rate is set to 100 packets/s, and each packet is 512 bytes. As we can see from the figures, given the data rate as high as 100pkts/s, the queue will be saturated during the transmission time (30s to 170s) and result in packet loss and long delay, both for OLSR and MP-OLSR.

Figure A.4 and figure A.5 show the length at the data rate of 40 pkts/s. At the source node, MP-OLSR has much more congestion than OLSR at the source node, which also cause much longer delay (0.11275s for MP-OLSR and 0.01829s for OLSR).

Algorithm A.1 Getting cross layer information in NS2

```
//In MPOLSR.h, declare the Queue and Link Layer object
NsObject *ll; //link layer
CMUPriQueue * ifq; //outgoing queue

////////////////////////////////////////////////////////
//In the command() of MPOLSR.cc, initiate the related object
...
TclObject *obj;
obj = TclObject::lookup(argv[2]==0);
ll = (NsObject*)obj;
obj = TclObject::lookup(argv[3])==0);
ifq = (CMUPriQueue*)obj;

////////////////////////////////////////////////////////
//In the simulation script, add the following code:
set rt($i) [$node_($i) agent 255];
$rt add_ll [$node_ ($i) set ll_(0)] [$node_($i) set ifq_(0)];
```

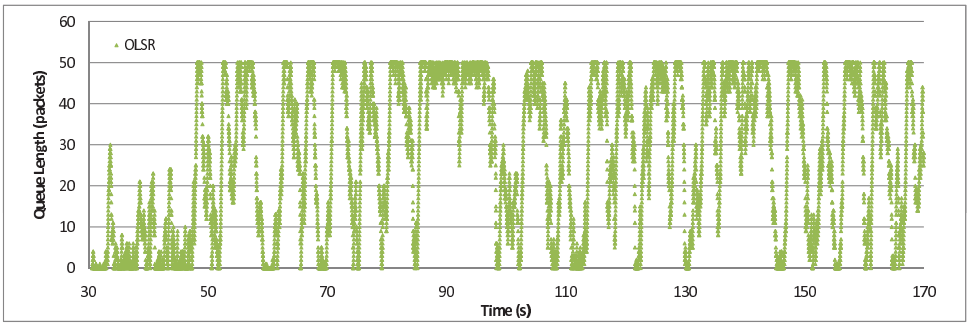


Figure A.2: Queue length in source node at 100 packets/s, OLSR

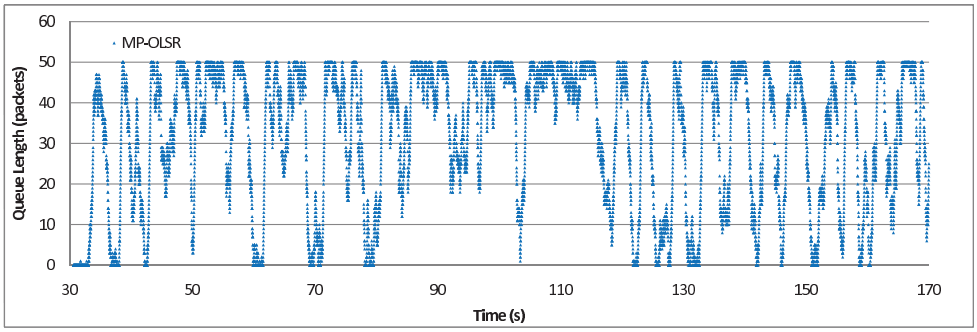


Figure A.3: Queue length in source node at 100 packets/s, MPOLSR

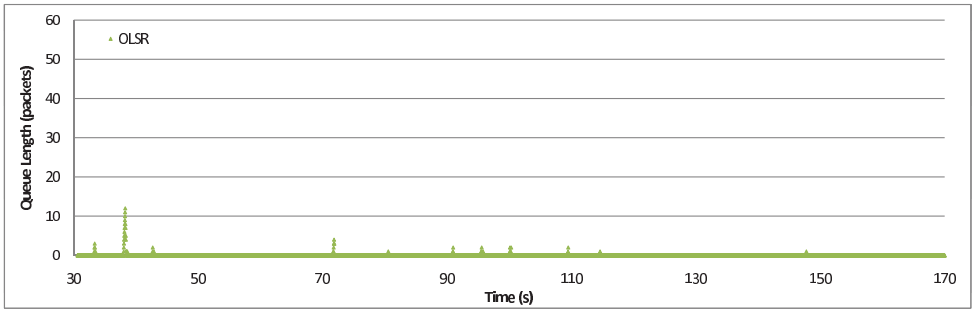


Figure A.4: Queue length in source node at 40 packets/s, OLSR

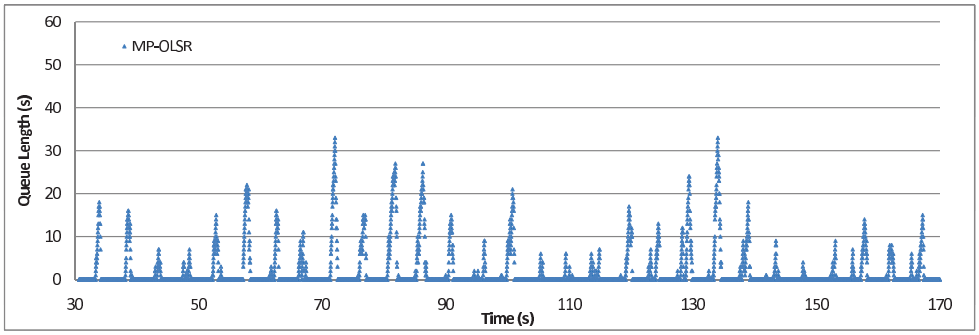


Figure A.5: Queue length in source node at 40 packets/s, MPOLSR

Concerning the difference of the protocol between OLSR and MP-OLSR, there are two possible reasons for the congestion in MP-OLSR.

- Firstly, the multi path transmission on the source node interface;
- Secondly, the multipath route computation introduces longer delay.

To explore the reasons of the congestion, more simulations are performed. Figure A.6 illustrates the queue length in a simulation of single path MP-OLSR, i.e. the basic MP-OLSR mechanisms are still used, but Multipath Dijkstra Algorithm only computes one path, rather than multiple paths. As shown in the figure, the single path MP-OLSR has even more congestion at the source node than MP-OLSR with 3 routes. So the multiple path routing is not the reason of the congestion.

To explore in which layer of the simulation model the delay mainly is located, how the packet travel through different layers are examined based on the trace file of the NS2 simulator. Figure

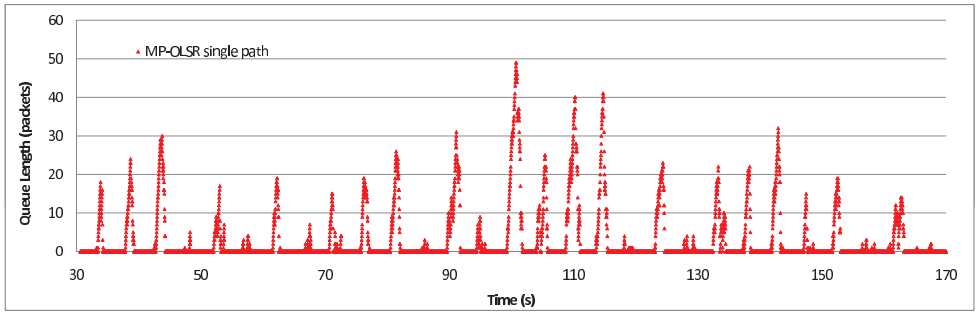


Figure A.6: Queue length in source node at 40 packets/s, single path MPOLSR

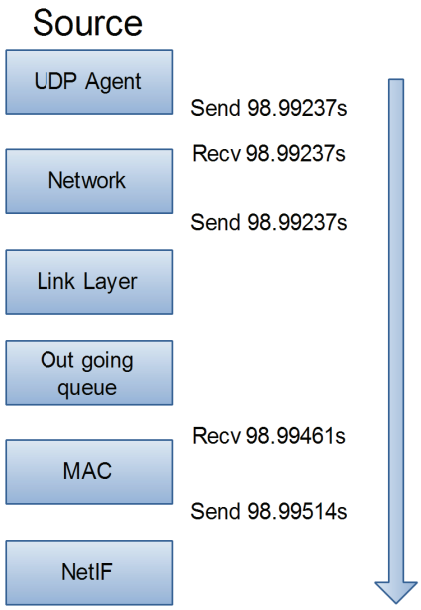


Figure A.7: Packet travels different layers in the source node without congestion

A.7 illustrates how a packet travels different layers (without congestion) in NS2 and indicates the time of each event. In this case, it took 0.00277 second from the packet generation at the UDP agent layer to the sending out at MAC layer.

In Figure A.8, a case of congested packet is studied. From the sending at the network layer to the receiving at the MAC layer, the time spend is 0.62997 second, where the congestion happened. In fact, the multi path computation is performed at the network layer, which almost did not take time. So the time consumed is in the *Link Layer* and *Outgoing queue,* where mainly ARP is performed. This a abnormal because the only difference between OLSR and MP-OLSR is that MP-OLSR use source routing. The possible explanation is that the NS2 simulator does not have very good support for source routing with its outgoing queue mechanism at high data rate.

However, at high data rate in NS2 simulation, the MP-OLSR suffers from congestion in the source node. The length of the IFQ in the node of the ad hoc network has big influence in the performance of the networks, both the data delivery ratio and the end-to-end delay. But based on the study of the outgoing queue in NS2, we found that the reason of the congestion is a flaw in the scalability of NS2 rather than the multipath routing protocol. The problem has been submitted to the NS2 mailing list, but there is no definite answer.

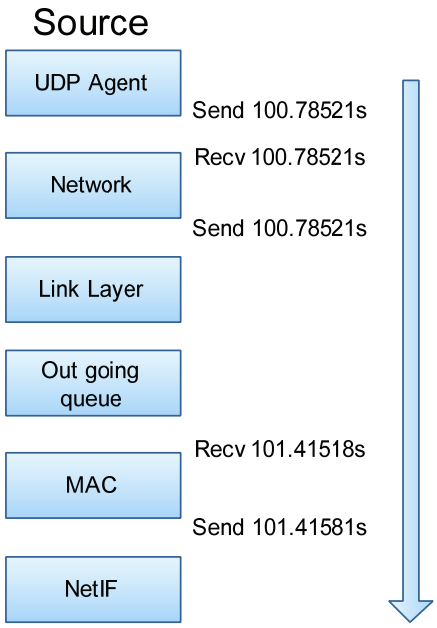


Figure A.8: Packet travels different layers in the source node with congestion

Appendix B

MP-OLSR Testbed Implementation

B.1 Multipath Extension of OLSRd Module

B.1.1 Data Structure

In the multipath plugin for *olsrd*, the following data structures are implemented:

g_cur_ser_tc_tab It is a hash table to save the topology information of the network. Each node in the network is presented by a *ser_tc_entry* structure. The IP addresses are used as the index of the hash table. The *ser_tc_entry* maintains a two-way link to its previous and following nodes as shown in Figure B.1.

Like the node's information, the 1-hop neighbors are also saved in a *dst_tab* hash table, as illustrated in Figure B.2. Each *ser_tc_entry_dst* presents a unidriction link to a 1-hop neighbor.

dev_cmd The *dev_cmd* is a command list that contains one or several commands that need to be forwarded to the *netfilter module*. The command is saved in a *ser_tc_token* structure. Each *ser_tc_token* is a elementary order that presents a modification in the network topology. It has four fields:

- The identification of the order (type)

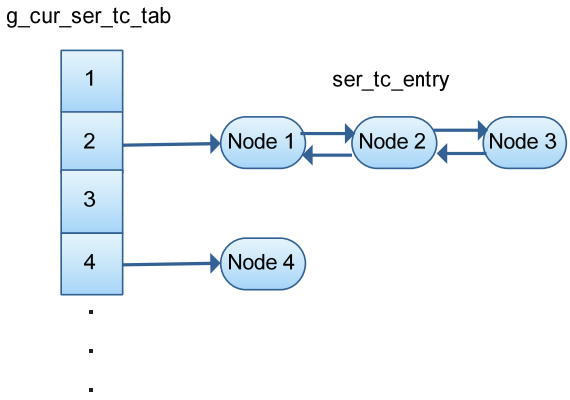


Figure B.1: The *g_cur_ser_tc_tab* for the toplogy in plugin of *olsrd*

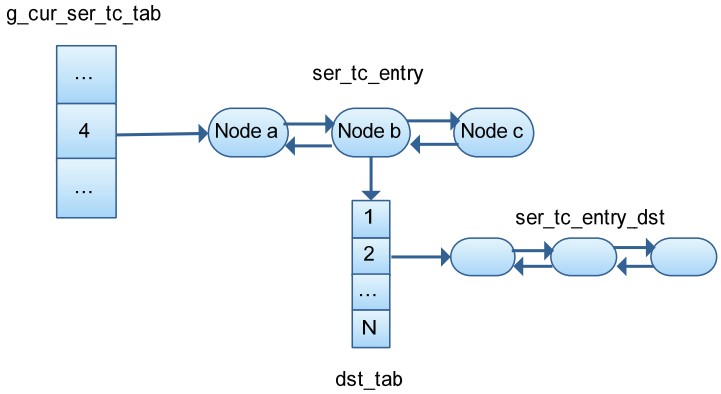


Figure B.2: The 1-hop neighbors in olsrd

Type	ID	addr_1	addr_2	value
Initailisation of the toplogy	1	-	-	-
Add a node	2	IP address of the node	-	-
Remove a node	3	IP address of the node	-	-
Add an unidirection link	4	IP address of the source	IP address of the destination	link cost
Remove a link	5	IP address of the source	IP address of the destination	-
Add an additional IP for a node (alias)	6	Principle IP	Additional IP	-
Remove an additional IP for a node (alias)	7	Principle IP	Additional IP	-
Configure the principle IP address for the local node	8	Principle IP of the local node	Network mask	-
Add an addictional IP for the local node	9	Additional IP of the local node	Network mask	-

Table B.1: The order of *ser_ tc_ token*

- Address 1 (addr_1)
- Address 2 (addr_2)
- Additional data (value)

The identifications of the orders and their usages are shown in Table B.1.

B.1.2 Algorithm

Module Initialization The initialization of the module is realized by calling *my_ init* method when Linux load the dynamic library and the *olsrd_ plugin_ init* method. During the initialization, the following operations are performed:

- Build the connection between the multipath *olsrd module* and *netfilter module.*
- Register the callbacks when there are topology modifications.

Figure B.3: The list of nodes *g_tc_node_lst*

Topology Modification When there is a modification in the topology information base, the following operations are performed:

1. Update the activity flag (*tc_reset_active_flag* method). For each node in the *g_cur_ser_tc_tab*, reset the activity flag (*is_active*) to 0, and the same operation for all the links to its neighbors.

2. Update the *olsrd* topology information base (*tc_check_data* method). The algorithm traverses the topology table of *olsrd*. For each link, find the source and destination node in *g_cur_ser_tc_tab*. If the nodes exist, set the *is_active* flag to 1. If the node does not exist, it is created (*tc_create_node* method) and the node create order is saved to be sent to the *netfilter module*. For the node found or created, the links in the link table (*dst_tab*) are also updated.

3. Remove the useless data (*tc_remove_useless_link* method). The algorithm traverse all the nodes and links. The nodes and links with the *is_active* flag equals 0 are considered to be lost. They are removed from the database and the corresponding removing orders are saved to be sent to the *netfilter module*.

4. Synchronize with the *netfilter module* (*dev_send* method). Send the modification orders saved in the previous steps to the *netfilter module*.

It is also worth to mention that the update of the IP routing table (by using *olsr_update_kernel_routes* method in *process_routes.c*) in the original *olsrd* is disabled.

B.2 Netfilter Module

B.2.1 Data Structure

g_tc_node_lst The *g_tc_node_lst* in the *netfilter module* maintains the topology information of the network (Figure B.3). Each node in the network is presented by an *olsr_node*.

The structure mainly includes:

- The IP address of the node (*olsr_node_addr*).
- The previous and next pointer to the other nodes (*next_node* and *prev_node*).
- The list of link. The *link_src* pointer denotes the local node is the source of the link, and *link_dst* denotes that the local node is the destination of the link.
- The cost of the node used in Dijkstra algorithm (dijkstra_cost).
- The previous node in the shortest path used in Dijkstra Algorithm (dijkstra_path).
- The number of hops between current node and source node in Dijkstra Algorithm.

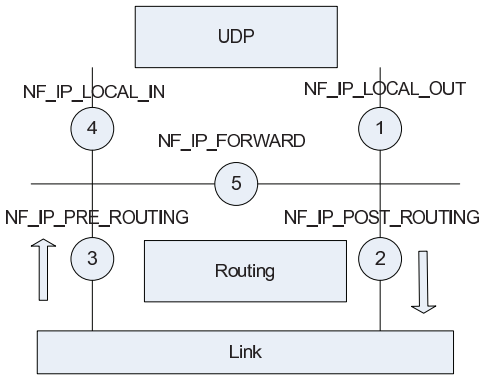


Figure B.4: Netfilter hooks

olsr_link It corresponds to a unidirectional link between two nodes. The structure mainly includes:

- The source node of the link (*src_node*).
- The destination node of the link (*dst_node*).
- The link cost obtained from *olsrd.*
- The link cost used by the Multipath Dijkstra algorithm (*dijkstra_cost*).

B.2.2 Algorithm

Communication with *olsrd* and its plugin The *netfilter/iptable* [1] provided by Linux is used to process the data packets and routing information. It provides 5 hooks on packet travel inside the IP stack. Hook 1 and 4, between TCP/UDP layer and IP layer, Hook 2 and 3, between IP layer and link layer and finally hook 5 for forwarded packets (Figure B.4).

The implemented *netfilter module* extension use the hook 1 (local out) to intercept packet, request a path and add retrieve path to the packet and later use the hook 3 (prerouting) to remove path when packets reach their destination. For each node machine crossed on packet path, the hook 3 (pre-routing) is also used to check path failure and apply route recovery if needed. Like in the hook 1 (local out), a path request is issued to retrieve a new path then packet is updated and forwarded with this new path.

Multipath Dijkstra Algorithm The Multipath Dijkstra Algorithm described in Algorithm 5.1 also relies in the *netfilter module.*

[1] Netfilter, http://www.netfilter.org

Appendix C

ErrorPatternGenerator

The *ErrorPatternGenerator* is created to generate the packet loss pattern for H.264/SVC packet trace. Based on the program, the user can generate a distorted packet trace by dropping packets from defined scalable layers.

Syntax

The command is like

ErrorPatternGenerator	<r | p | b> <random_seed> <video_packet_trace> <scalable_layer 1> <value 1> [scalable_level 2] [value 2]...

- <r | p | b> The error pattern type. *r*: the loss pattern is defined by the ratio of the dropped packets. *p*: the loss pattern is defined by he number of packets. *b*: the loss pattern is defined by the number of bytes.
- <random_seed> The random seed that generates the random numbers.
- <video_packet_trace> The packet trace of the video bitstream. It should be an output of JSVM *BitstreamExtractor*.
- <scalable_layer 1> <value 1> To define the scalable layer of the packet loss. The value is defined by the error pattern type.

Example An example is shown as following:

```
ErrorPatternGenerator p 2 video_sample.trace t1 20
```

It means that error pattern type *p* is used. In scalable layer *t1*, 20 packets will be dropped randomly using random seed 2.

The user can also define multiple scalable layers, but it is not recommended to set different kinds of scalable at the same time (for example, set *temporal* and *spatial* scalabilities in the same command).

Output

The output of the *ErrorPatterGenerator* includes:

```
Input File:   video_sample.trace
Random Seed:  2
Loss Pattern:
  t1  20.0
Output Trace File:  video_sample_seed2_byPacket_t1_20.0.trace
Output Dropped File:  video_sample_seed2_byPacket_t1_20.0.drop
Output Log File:  video_sample_seed2_byPacket_t1_20.0.log

Dropped Packets Statistcis
Type  Drp_No  Drp_Bs  Ttl_Pkt TtlBs B/Pkt Actual_Lost_Ratio  Set_value
t1    20      24175   25      29966 1198  0.8                20.0

Total Bytes:  474629
Dropped Bytes:  24175
Total Lost Rate (by bytes): 0.05093452
Total Packets:  492
Dropped Packets:  20
Total Lost Rate (by packets):    0.040650405

//ErrorPatternGenerator v0.1 for JSVM BitstreamExtractor, By Jiazi Yi,
Polytech'Nantes, 2010
```

Figure C.1: Output log file of *ErrorPatternGenerator*

Output log File. It records the following information:

- The input of the program: input file name, random seed, loss patter, etc.
- The name of related output files.
- The statistics of dropped packets: type, number of packets dropped, number of bytes dropped, total packet, total bytes, average bytes per packet, actual lost ratio, the pre-set value of the packet loss.
- The statistics of the bitstream file: total bytes, dropped bytes, lost ratio, etc.

An example of the output log file is shown in Figure C.1.

Output trace file. The output distorted bitstream trace. It can be used as the input of *BitstreamExtractor* to generate the distorted bitstream.

Output dropped file. The dropped bitstream trace.

Appendix D

QualnetTraceParser

The *QualnetTraceParser* is created to analyse the the Qualnet trace. It is an important part of the *SVCEval* framework, which connects the JSVM video codec and Qualnet simulator.

Syntax

The command is like

QualnetTraceParser	<source_id> <destination_id> <protocol_type> <non_dis> <delay_threshold> <input_qualnetTrace> <input_BitstreamTrace>

- <source_id> The video source node in the network.
- <destination_id> The destination node in the network.
- <protocol_type> The type of application layer protocol in Qualnet. Normally, the TRAFFIC-TRACE application is used, so it should be set to 58.
- <non_dis> If we preserve the non-discardable packets or not (true/false). With current error-concealment algorithms, the loss of the non-discardable packets in the bitstream might results in the error when decoding. So it is recommended that set this option to *true* to make sure the bitstream can be decoded.
- <delay_threshold> The delay threshold in seconds. The packets with delay more than this threshold are regarded as lost. This option is very useful for time-critical traffic. If the value is <=0, then the threshold is not considered.
- <input_qualnetTrace> Qualnet XML packet trace.
- <input_BitstreamTrace> The original bitstream trace.

Example An example is shown as following:

QualnetTraceParser	12 61 58 true 0.2 qualnet_packet_trace.trace bitstream_trace

It means that node 12 is the source of the video transmission, and node 61 is the destination. The *non_ discardable* packets will be reserved. The delay threshold is set to 0.2s, i.e. the packets will delay longer than 0.2 are considered as lost packets. The Qualnet trace file name is *qualnet_ packet_ trace.trace*, and the original bitstream trace is *bitstream_ trace.*

```
*************************************************************************
*Created by Jiazi Yi, Ecole Polytech'Nantes, 2010
*************************************************************************
*The input Qualnet trace file is video_sample.trace
*The input BitStream trace file is
football_y00_v98_160frms_1000_rf1_wo9.trace
*The output sender trace file is video_sample_snd.tr
*The output receiver trace file is video_sample_rcv.tr
*The output BitStream trace file is video_sample_bs.btr
*The output dropped bitstream trace file is video_sample_bs.dtr
*Totally 70 packets are lost
*In which 4 packets are dropped because of delay threshhold 0.2
*ATTENTION! 39 non-discardable packets are reocvered
*That is to say, there are 31 actually lost.
*************************************************************************
```

Figure D.1: An example of the output log file of *QualnetTraceParser*

Output

The output of the *QualnetTraceParser* includes:

An log file. It consists of following information:

- The input file names: Qualnet packet trace file and bitstream file.
- The output file names: sender trace file, receiver trace file, bitstream trace file and dropped bitstream trace file.
- Related statistics.

An example is as shown in Figure D.1.

An sender trace file. It records the packets trace that are sent out at the video source.

An receiver trace file. It records the packets trace that are received at the destination.

An bitstream packet trace file. It records the received distorted bitstream trace.

An dropped bitstream packet trace file.

Appendix E

FECSim: FEC codec simulator

E.1 QualnetFECTrafficGen

QualnetFECTrafficGen is a FEC traffic generator which simulates the behavior of FEC encoder. It reads the original bitstream packet trace and generates the coded packet trace based on the configuration of the priority FEC.

Syntax

The command line of *QualnetFECTrafficGen* is like:

```
QualnetFECTrafficGen  <packet_rate> <input_bitstream_trace> <FEC_config>
```

- <packet_rate> The packet rate per second.
- <input_bitstream> The input bitstream trace generated by *BitstreamExtractor*.
- <FEC_config> The FEC configuration file which specifies the encoding parameters for each scalable layer:

```
<scalable_layer> <#packets_to_be_coded> <#coded projection> <systematic>
```

Example

An example of QualnetFECTrafficGen is shown as following:

```
QualnetFECTrafficGen 10 bitstream.trace fecConfig.cfg
```

The content of *fecConfig.cfg* is:

```
t1 2 3 2 s
t2 2 3 2 s
```

The configuration stands for that for temporal layer *t1* and *t2*, the encoder will buffer 2 data packets, and generate 3 coded projections. In the receiver, 2 projections in 3 are needed to recover the original packets. The *s* means that systematic code is used. For the other scalable layers, the FEC code is not used by default.

Output

The output of *QualnetFECTrafficGen* has two files:

- Coded packet traffic trace. It is the input for Qualnet simulation.
- Coded packet information. It records the FEC information of each projection:

  <sequence_number> <packet_size> <#packets_to_be_coded> <#coded projection> <systematic> <start_pos>

  This file is needed for decoding process. An example is shown in Figure E.1.

```
0   261   1 1 1 false 0x00000000
1   14    1 1 1 false 0x00000105
2   16    1 1 1 false 0x00000113
3   8     1 1 1 false 0x00000123
4   9     1 1 1 false 0x0000012b
...
...
46  1289  2 3 2 true  0x00008f35  0x000092f3
47  1289  2 3 2 true  0x00008f35  0x000092f3
48  1289  2 3 2 true  0x00008f35  0x000092f3
49  1294  2 3 2 true  0x000096c3  0x00009a81
50  1294  2 3 2 true  0x000096c3  0x00009a81
51  1294  2 3 2 true  0x000096c3  0x00009a81
```

Figure E.1: An example of coded FEC trace

E.2 QualnetFECTraceParser

The FEC trace parser simulates the behavior of FEC decoder. It reads the qualnet packet trace and generate the distorted bitstream trace based on the FEC trace (generated by *QualnetFEC-TrafficGen*).

Syntax

The command line of *QualnetFECTraceParse* is like:

QualnetFECTraceParser <source_id> <destination_id> <delay_thershold> <non_dis> <qualnet_trace> <fec_trace> <original_bitstream>

- <source_id> The video source node in the network.
- <destination_id> The destination node in the network.
- <delay_threshold> The delay threshold in seconds. The packets with delay more than this threshold are regarded as lost. This option is very useful for time-critical traffic. If the value is <=0, then the threshold is not considered.
- <non_dis> If we preserve the non-discardable packets or not (true/false). With current error-concealment algorithms, the loss of the non-discardable packets in the bitstream might results in the error when decoding. So it is recommended that set this option to *true* to make sure the bitstream can be decoded.
- <qualnet_trace> Qualnet XML packet trace.
- <fec_trace> The FEC trace generated by *QualnetFECTrafficGen*.
- <original_bitstream> The original bitstream trace.

Example

An example of *QualnetFECTraceParser* is as shown in following:

```
QualnetFECTraceParser 12 61 0.2 true qualnet_test.trace fec_trace.fec bitstream.trace
```

Output

The output of *QualnetFECTraceParser* includes two file:

- A distorted bitstream packet trace. This is the result of the decoding process. It can be used as the input of *BitstreamExtractor* to generate the distorted bitstream.
- A log file, as shown in Figure E.2.

```
*************************************************************************
*            Created by Jiazi YI, Ecole Polytech'Nantes, 2010           *
*************************************************************************
*The sender node is 12, the reciever node is 61
*The delay threshold is 0.20000000298023224,
* the recover non-discardable option is true
*The Qualnet trace input is qualnet_test.trace
*The FEC trace input is fecTest.fec
*The original bitstream input is bitstream.trace
*The distorted bitsteam output is qualnet_test.btr
*The log output is qualnet_test.log
*
* Total num of bitstream record is 654
* Total num recieved bitstream record is 648
* Num of recovered bitstream record because of the non_dis option is 6
* Num of acutal received bitstream record is 642
* Num of bitstream record reconstructed by systematic code: 110
* Num of bitsrream record reconstructed by non-systematic code: 532

* Num of FEC record: 710
* Bytes of FEC record: 574371
* Num of received FEC record: 691
*************************************************************************
```

Figure E.2: An example of *QualnetFECTraceParser* log

Appendix F

Related Publications

1. Jiazi Yi, Asmaa Adnane, Sylvain David, Benoît Parrein. *Multipath optimized link state routing for mobile ad hoc networks*. Elsevier Ad Hoc Networks Journal. Volume 9, Issue 1, 28-47, Jan. 2011.

2. Jiazi Yi, Sylvain David, Hassiba Asmaa Adnane, Benoît Parrein, Xavier Lecourtier. *Multipath OLSR: Simulation and Testbed.* 5th OLSR Interop/Workshop, Vienna : Austria, 2009.

3. Benoît Parrein, Jiazi Yi. *SEREADMO : protocole de routage sécurisé pour réseaux ad hoc mobiles.* Colloque Francophone sur l'Ingénierie des Protocoles 2009, Strasbourg, France, 2009.

4. Jiazi Yi, Eddy Cizeron, Salima Hamma, Benoît Parrein, Pascal Lesage. *Simulation and Performance Analysis of MP-OLSR for Mobile Ad hoc Networks*. 4th OLSR Interop/Work Shop, Ottawa : Canada, 2008.

5. Jiazi Yi, Eddy Cizeron, Salima Hamma, Benoît Parrein. *Implementation of Multipath and Multiple Description Coding in OLSR.* IEEE WCNC 2008, Las Vegas : USA, 2008.

- Jiazi YI, Multipath optimized link state routing. Session of IETF MANET working group, 78th IETF meeting, Maastricht, Netherland, 2010.

Bibliography

Adjih, C., Clausen, T., Jacquet, P., Laouiti, A., Muhlethaler, P., & Raffo, D. (2003). Securing the olsr protocol. In *In 2nd IFIP Annual Mediterranean Ad Hoc Networking Workshop (Med-Hoc-Net 2003), Mahdia*, pages 25–27.

Adnane, H.-A. (2008). *La confiance dans le routage Ad Hoc: étude du protocole OLSR*. Ph.D. thesis, University of Rennes 1.

Albanese, A., Blomer, J., Edmonds, J., Luby, M., & Sudan, M. (1994). Priority encoding transmission. *IEEE Transactions on Information Theory*, **42**(6), 1737–1744.

Amon, P., Rathgen, T., & Singer, D. (2007). File format for scalable video coding. *IEEE Transactions on Circuits and Systems for Video Technology*, **17**(9), 1174–1185.

Amonou, I., Cammas, N., Kervadec, S., & Pateux, S. (2006). Some considerations about restriction on priority_id syntax element. Technical report, ISO/IEC JTC1/SC29/WG11 and ITU-T SG16 Q.6.

Anjum, F. & Mouchtaris, P. (2007). *Security for Wireless ad hoc Networks*. John Wiley & Sons, Inc.

Autrusseau, F. (2002). *Modélisation psychovisuelle pour le tatouage des images*. Ph.D. thesis, University of Nantes.

Autrusseau, F., Guédon, J., & Bizais, Y. (2003). Mojette cryptomarking scheme for medical images. pages 958–965.

Badis, H. & Agha, K. A. (2004). Qolsr multi-path routing for mobile ad hoc networks based on multiple metrics: bandwidth and delay. In *IEEE Vehicular Technology Conference*, pages 2181–2184, Los Angeles, CA, USA.

Boukerche, A. (2006). *Handbook of Algorithms for Wireless Networking and Mobile Computing*. Chapman & Hall/CRC.

Camp, T., Boleng, J., & Davies, V. (2002). A survey of mobility models for ad hoc network research. *Wireless Communications & Mobile Computing (WCMC): Special issue on Mobile Ad Hoc Networking: Research, Trends and Applications*, **2**(5), 483–502.

Caro, G. A. D. (2003). Analysis of simulation environments for mobile ad hoc networks. Technical report, Dalle Molle Institute for Artificial Intelligence, Manno, Switzerland.

Chakeres, I. & Perkins, C. (2010). IETF Internet Draft: Dynamic MANET On-demand (DYMO) Routing, draft-ietf-manet-dymo-21.txt.

Chandra, S., Svalbe, I., & Guédon, J.-P. (2008). An exact, non-iterative Mojette inversion technique utilising ghosts. In *DGCI'08: Proceedings of the 14th IAPR international conference on Discrete geometry for computer imagery*, pages 401–412, Berlin, Heidelberg. Springer-Verlag.

Chen, L. & Heinzelman, W. B. (2007). A survey of routing protocols that support qos in mobile ad hoc networks. *IEEE Network*, pages 30–38.

Chen, Y., Xie, K., Zhang, F., Pandit, P., & Boyce, J. (2006). Frame loss error concealment for svc. *Journal of Zhejiang University - Science A*, **7**(5), 677–683. 10.1631/jzus.2006.A0677.

Clausen, T. & Dearlove, C. (2009). IETF Request for Comments: 5497, Representing Multi-Value Time in Mobile Ad Hoc Networks (MANETs).

Clausen, T. & Jacquet, P. (2003). IETF Request for Comments: 3626, Optimized Link State Routing Protocol OLSR.

Clausen, T., Dearlove, C., & Adamson, B. (2008). IETF Request for Comments: 5148, Jitter Considerations in Mobile Ad Hoc Networks.

Clausen, T., Dearlove, C., & Dean, J. (2009a). IETF Internet Draft, MANET Neighborhood Discovery Protocol (NHDP), draft-ietf-manet-nhdp-10.

Clausen, T., Dearlove, C., & Jacquet, P. (2009b). IETF Internet Draft, The Optimized Link State Routing Protocol version 2, draft-ietf-manet-olsrv2-10.

Clausen, T., Dearlove, C., Dean, J., & Adjih, C. (2009c). IETF Request for Comments: 5444, Generalized Mobile Ad Hoc Network (MANET) Packet/Message Format.

Conti, M. & Giordano, S. (2007). Multihop ad hoc networking: The reality. *IEEE Communications Magazine*, **45**(4), 88–95.

Denning, P. J. (2005). The locality principle. *Commun. ACM*, **48**(7), 19–24.

Evenou, P., Autrusseau, F., & Hamon, T. (2006). Secure distributed storage based on the mojette transform. In *NOTERE*, pages 161–170, Toulouse, France.

Fall, K. & Varadhan, K. (2010). The ns manual. Technical report, The VINT Project.

Gast, M. (2002). *802.11 Wireless Networks - The definitive Guide*. O'Reilly.

Ge, Y., Kunz, T., & Lamont, L. (2003). Quality of service routing in ad-hoc networks using olsr. In *36th Hawaii International Conference on System Sciences*.

Gharavi, H. (2008). Multichannel mobile ad hoc links for multimedia communications. *Proceedings of the IEEE*, **96**(1), 77–96.

Goldsmith, A. (2005). *Wireless Communications*. Cambridge University Press.

Guédon, J. (2009). *The Mojette Transform: Theory and Applications*. Wiley Press.

Guédon, J. & Normand, N. (2005). The mojette transform: The first ten years. *Lecture Notes in Computer Science*, **3429/2005**, 79–91.

Guo, Y., Wang, Y. K., & Li, H. (2006). SVC/AVC loss simulator donation, JVT-Q069, Tech report.

Guo, Y., Chen, Y., Wang, Y.-K., Li, H., Hannuksela, M., & Gabbouj, M. (2009). Error resilient coding and error concealment in scalable video coding. *IEEE Transactions on Circuits and Systems for Video Technology*, **19**(6), 781–795.

Haas, Z. J. & Pearlman, M. R. (1997). The Zone Routing Protocol (ZRP) for Ad Hoc Networks. Internet-draft, IETF MANET Working Group. Expiration: May, 1998.

Hamdach, W. (2010). QoS for Video Transmission over Wireless Ad-hoc Networks, Master Thesis, University of Nantes.

Hamma, S., Cizeron, E., Issaka, H., & Guedon, J.-P. (2006). Performance evaluation of reactive and proactive routing protocol in ieee 802.11 ad hoc network. In *Next Generation Communication and Sensor Network*, Boston, USA.

Heyman, D. P. & Lakshamn, T. V. (1996). Methods for Performance Evaluation of VBR Video Traffic Models. *IEEE/ACM Trans. Net.*, **4**(1), 40–8.

IETF (1981). Request for Comments: 791, IP: Internet Protocol.

ISO (1993). *ISO/IEC 11172. Information technology – Coding of moving pictures and associated audio for digital store*. International Organization for Standardization.

ISO (2003). *Information technology: Coding of audio-visual objects, Part 14: MP4 file format*. International Organization for Standardization.

ISO (2004a). *Information technology – Coding of audio-visual objects – Part 12: ISO base media file format, ISO/IEC 14496-12*. International Organization for Standardization.

ISO (2004b). *Information technology:Coding of audio-visual objects, Part 15: Advanced Video Coding (AVC) file format*. International Organization for Standardization.

ISO (2004c). *ISO/IEC 14496-2:2004 - Information technology – Coding of audio-visual objects – Part 2: Visual*. International Organization for Standardization.

ITU (1994a). *ITU-T Recommendation H.120, Codecs for video conferencing using primary digital group transmission*. Telecommunication Standardization Section of ITU.

ITU (1994b). *ITU-T Recommendation H.261, Video codec for audiovisual services at p× 64 kbits*. Telecommunication Standardization Section of ITU.

ITU (2000). *H.262 : Information technology - Generic coding of moving pictures and associated audio information: Video*. Telecommunication Standardization Section of ITU.

ITU (2003). *H.264, Series H: Audiovisual and Multimedia Systems. Infrastructure of audiovisual services - Coding of moving video. Advanced video coding for generic audioviual services*. Telecommunication Standardization Section of ITU.

ITU (2005). *H.263, Series H: Audiovisual and Multimedia Systems. Infrastructure of audiovisual services - Coding of moving video. Video Coding for low bit rate communication*. Telecommunication Standardization Section of ITU.

Johnson, D. & Hancke, G. (2009). Comparison of two routing metrics in olsr on a grid based mesh network. *Ad Hoc Networks*, **7**(2), 374–387.

Johnson, D. B., Maltz, D. A., & Broch, J. (2001). *Ad hoc Networking*. Addison-Wesley.

Johnson, D. B., Hu, Y., & Maltz, D. A. (2007). IETF Request for Comments: 4728, The Dynamic Source Routing Protocol (DSR) for Mobile Ad Hoc Networks for IPv4.

Kingston, A. & Svalbe, I. (2006). Projective transforms on periodic discrete image arrays. In P. Hawkes, editor, *Advances in Imaging and Electron Physics*, volume 139 of *Advances in Imaging and Electron Physics*, pages 75 – 177. Elsevier.

Kingston, A. M. (2005). *Extension and Application of Finite Discrete Radon Projection Theory*. Ph.D. thesis, School of Physics, Monash University.

Klaue, J., Rathke, B., & Wolisz, A. (2003). Evalvid - a framework for video transmission and quality evaluation. In *13th International Conference on Modelling Techniques and Tools for Computer Performance*, Illinois, USA.

Krishnan, R. & Silvester, J. A. (1993). Choice of allocation granularity in multipath source routing schemes. In *INFOCOM '93. Proceedings.Twelfth Annual Joint Conference of the IEEE Computer and Communications Societies. Networking: Foundation for the Future. IEEE*, pages 322–329 vol.1.

Kuipers, F., Van Mieghem, P., Korkmaz, T., & Krunz, M. (2002). An overview of constraint-based path selection algorithms for QoS routing. *Communications Magazine, IEEE*, **40**(12), 50–55.

Kun, M., Jingdong, Y., & Zhi, R. (2005). The research and simulation of multipath olsr for mobile ad hoc network. In *International Symposium on Communications and Information Technologies (ISCIT)*, pages 540–543.

Lee, S. J. & Gerla, M. (2002). Split multi-path routing with maximally disjoint paths in ad hoc networks. In *Communications, 2001. ICC 2001. IEEE International Conference on*, volume 10, pages 3201–3205.

Ma, Q., Wu, F., Lou, J., & Sun, M.-T. (2009). Frame loss error concealment for spatial scalability using hallucination. In *Packet Video Workshop, 2009. PV 2009. 17th International*, Seattle, WA.

Mao, S., Lin, S., Panwar, S., & Wang, Y. (2003). A multipath video streaming testbed for ad hoc networks. *IEEE Vehicular Technology Conference*, pages 2961–2965.

Marina, M. K. & Das, S. R. (2006). Ad hoc on-demand multipath distance vector routing. *Wireless communications and mobile computing*, **6**, 969–988.

Matús, F. & Flusser, J. (1993). Image representation via a finite radon transform. *IEEE Trans. Pattern Anal. Mach. Intell.*, **15**(10), 996–1006.

Mueller, S., Tsang, R., & Ghosal, D. (2004). Multipath routing in mobile ad hoc networks: Issues and challenges. In *In Performance Tools and Applications to Networked Systems, volume 2965 of LNCS*, pages 209–234. Springer-Verlag.

Normand, N., Parrein, B., Svalbe, I., & Kingston, A. (2010). Erasure coding with the fnite radon transform. In *IEEE Wireless Communications and Networking Conference*, Sydney, Australia.

Owada, Y., Maeno, T., Imai, H., & Mase, K. (2007). Olsrv2 implementation and performance evaluation with link layer feedback. In *Proceedings of the 2007 international conference on Wireless communications and mobile computing*, Honolulu, Hawaii, USA.

Parrein, B., Boulos, F., Callet, P. L., & Guédon, J. P. (2007). Priority image and video encoding transmission based on a discrete radon transform. In *IEEE 16th International Packet Video Workshop*, Lausanne, Switzerland.

Pearlman, M. R., Haas, Z. J., & Sholander, P. (2002). Alernate path routing in mobile ad hoc networks. In *IEEE MILCOM*, Los Angeles, CA.

Pelusi, L., Passarella, A., & Conti, M. (2006). Opportunistic networking: Data forwarding in disconnected mobile ad hoc networks. *IEEE Communications Magazine*.

Pereira, C., Pousset, Y., Vauzelle, R., & Combeau, P. (2006). Sensitivity of the mimo channel characterization to the modeling of the environment. *Antennas and Propagation, IEEE Transactions on*, **57**(4), 1218 – 1227.

Perkins, C. & Royer, E. (1999). Ad-hoc on-demand distance vector routing. In *Second IEEE Workshop on Mobile Computing Systems and Applications*, pages 90–100.

Perkins, C. E. & Bhagwat, P. (2001). Dsdv routing over a multihop wireless network of mobile computers. In C. E. Perkins, editor, *Ad Hoc Networking*, chapter 3, pages 53–74. Addison-Wesley.

Poussard, A.-M., Hamidouche, W., Vauzelle, R., Pousset, Y., & Parrein, B. (2009). Realistic SISO and MIMO Physical Layer implemented in two Routing Protocols for Vehicular Ad hoc Network. In *IEEE ITST*, Lille, France.

Richardson, I. E. (2003). *H.264 and MPEG-4 Video compression*. John Wiley & Sons Inc.

RNRT (2005). Appel à projet RNRT 2005, Numéro de référence THALES/RNRT/EG/27.06.2005.REF.V0. Tech report.

Rogge, H., Baccelli, E., & Kaplan, A. (2010). Ietf internet draft, packet sequence number based etx metric for mobile ad hoc networks.

Schulzrinne, H., Casner, S., Frederick, R., & Jacobson, V. (2003). IETF Request for Comments: 3550, RTP: A Transport Protocol for Real-Time Applications.

Schwarz, H., Marpe, D., & Wiegand, T. (2007). Overview of the scalable video coding extension of the h.264/avc standard. *IEEE Transactions on Circuits and Systems for Video Technology*, **17**(9), 1103–1120.

Seeling, P., Reisslein, M., & Kulapala, B. (2004). Network performance evaluation using frame size and quality traces of single-layer and two-layer video: A tutorial. *IEEE Communications Surveys & Tutorials*, **6**(3), 58–78.

Serviers, M., Normand, N., Guedon, J., & Bizais, Y. (2005). The Mojette transform: Discrete angles for tomography. *Electronic Notes in Discrete Mathematics*, **20**, 587 – 606. Proceedings of the Workshop on Discrete Tomography and its Applications.

Shamir, A. (1979). How to share a secret. *Commun. ACM*, **22**(11), 612–613.

Shannon, C. E. (1948). A mathematical theory of communication. *Bell system technical journal*, **27**.

SNT (2009a). Qualnet 5.0 model library index. Technical report, Scalable Network Technologies, Inc.

SNT (2009b). Qualnet 5.0 programmer's guide. Technical report, Scalable Network Technologies, Inc.

SNT (2009c). Qualnet 5.0 user's guide. Technical report, Scalable Network Technologies, Inc.

Speakman, L., Owada, Y., & Mase, K. (2008). An analysis of loop formation in OLSRv2 in ad-hoc networks and limiting its negative impact. In *IEEE International CQR Workshop*, Naples, Florida, USA.

Stepanov, I. & Rothermel, K. (2008). On the impact of a more realistic physical layer on manet simulations results. *Ad Hoc Networks*, **6**(1), 61 – 78.

Svalbe, I. & van der Spek, D. (2001). Reconstruction of tomographic images using analog projections and the digital radon transform. *Linear Algebra and its Applications*, **339**(1-3), 125 – 145.

Takahashi, Y., Owada, Y., Okada, H., & Mase, K. (2007). A wireless mesh network testbed in rural mountain areas. In *WinTECH '07: Proceedings of the second ACM international workshop on Wireless network testbeds, experimental evaluation and characterization*, pages 91–92, New York, NY, USA. ACM.

Tanenbaum, A. S. (2003). *Computer Networks, 4th Edition*. Prentice Hall.

Toh, C.-K. (2002). *Ad hoc Mobile Wireless Networks: Protocols and Systems*. Prentice Hall PTR.

Tonnesen, A. (2004). Implementing and extending the optimized link state routing protocol.

Toubiana, V., Labiod, H., Reynaud, L., & Gourhant, Y. (2008). An analysis of asma performances against packet dropping attacks in dense networks. In *ISCC*, pages 254–259. IEEE.

Tsai, J. W. & Moors, T. (2008). Minimum interference multipath routing using multiple gateways in mesh networks. In *IEEE Conference on Mobile ad-hoc and Sensor Systems*, Atlanta, Georgia.

Tsai, J. W. & Moors, T. (2009). Opportunistic multipath routing in wireless mesh networks. *Lecture Notes of the Institute for Computer Sciences, Social Informatics and Telecommunications Engineering*, **22**, 71–85.

Viennot, L. & Jacquet, P. (2007). Bi-connexité, k-connexité et multipoints relais. In *9ème Rencontres Francophones sur les Aspects Algorithmiques des Télécommunications*, pages 9–12, Ile d'Oléron France.

Wang, Y.-K., Hannuksela, M. M., & Pateux, S. (2007). System and transport interface of svc. *IEEE Transactions on Circuits and Systems for Video Technology*, **17**(9), 1149–1163.

Wenger, S. (2002). Error Patterns for Internet Experiments, VCEG Q15-I-16r1.

Wenger, S., Hannuksela, M., Stockhammer, T., Westerlund, M., & Singer, D. (2005). Ietf request for comments: 3984, rtp payload format for h.264 video.

Wenger, S., Wang, Y.-k., & Hannuksela, M. (2006). Rtp payload format for h.264/svc scalable video coding. *Journal of Zhejiang University - Science A*, **7**(5), 657–667. 10.1631/jzus.2006.A0657.

Wenger, S., Wang, Y.-K., & Schierl, T. (2007). Transport and signaling of svc in ip networks. *IEEE Transactions on Circuits and Systems for Video Technology*, **17**(9), 1164–1173.

Wenger, S., Wang, Y., Schierl, T., & Eleftheriadis, A. (2010). Ietf internet draft: Rtp payload format for svc video, draft-ietf-avt-rtp-svc-21.txt.

Wiegand, T., Sullivan, G. J., Bjontegaard, G., & Luthra, A. (2003). Overview of h.264/avc video coding standard. *IEEE Transactions on Circuits and Systems for Video Technology*.

Xue, Q. & Ganz, A. (2003). Ad hoc qos on-demand routing (aqor) in mobile ad hoc networks. *Journal of Parallel and Distributed Computing*, **63**(2), 154 – 165. Routing in Mobile and Wireless Ad Hoc Networks.

Zapata, M. G. (2005). Secure ad hoc on-demand distance vector (saodv) routing. Internet-draft, IETF MANET Working Group.

Zhai, Y., Yang, O., Wang, W., & Shu, Y. (2005). Implementing multipath source routing in a wireless ad hoc network testbed. *IEEE Pacific Rim Conference on Communications, Computers and signal Processing*, pages 292– 295.

Zhou, X., Lu, Y., & Xi, B. (2005). A novel routing protocol for ad hoc sensor networks using multiple disjoint paths. In *2nd International Conference on Broadband Networks*, Boston, MA, USA.

Résumé

Les réseaux ad hoc sont constitués d'un ensemble de nœuds mobiles qui échangent des données sans infrastructure de type point d'accès ou artère filaire. Ils sont par définition auto-organisés. Les changements fréquents de topologie des réseaux ad hoc rendent le routage multi-sauts très problématique. Dans cette thèse, nous proposons un protocole de routage à chemins multiples appelé Multipath Optimized Link State Routing (MP-OLSR). C'est une extension d'OLSR à chemins multiples qui peut être considérée comme une méthode de routage hybride. En effet, MP-OLSR combine la caractéristique proactive de la *détection de topologie* et la caractéristique réactive du *calcul de chemins multiples* qui est effectué à la demande. Les fonctions auxiliaires comme la *récupération de routes* ou la *détection de boucles* sont introduites pour améliorer la performance du réseau. L'utilisation de la longueur des files d'attente des nœuds intermédiaires comme critère de qualité de lien est étudiée et la compatibilité entre routage à chemins multiples et chemin unique est discutée pour faciliter le déploiement du protocole. Les simulations basées sur les logiciels NS2 et Qualnet sont effectuées pour tester le routage MP-OLSR dans des scénarios variés. Une mise en œuvre a également été réalisée au cours de cette thèse avec une expérimentation sur le campus de Polytech'Nantes. Les résultats de la simulation et de l'expérimentation révèlent que MP-OLSR est particulièrement adapté pour les réseaux mobiles et denses avec des trafics élevés grâce à sa capacité à distribuer le trafic dans des chemins différents et à des fonctions auxiliaires efficaces. Au niveau application, le service vidéo H.264/SVC est appliqué à des réseaux ad hoc MP-OLSR. En exploitant la hiérarchie naturelle délivrée par le format H.264/SVC, nous proposons d'utiliser un codage à protection inégale (PFEC) basé sur la Transformation de Radon Finie (FRT) pour améliorer la qualité de la vidéo à la réception. Un outil appelé *SVCEval* est développé pour simuler la transmission de vidéo SVC sur différents types de réseaux dans le logiciel Qualnet. Cette deuxième étude témoigne de l'intérêt du codage à protection inégale dans un routage à chemins multiples pour améliorer une qualité d'usage sur des réseaux auto-organisés.

Mots-clés : Réseaux ad hoc mobiles, chemins multiples, OLSR, MP-OLSR, codage canal à effacement, H.264/SVC

Abstract

Ad hoc networks consist of a collection of wireless mobile nodes which dynamically exchange data without reliance on any fixed based station or a wired backbone network. They are by definition self-organized. The frequent topological changes make multi-hops routing a crucial issue for these networks. In this PhD thesis, we propose a multipath routing protocol named Multipath Optimized Link State Routing (MP-OLSR). It is a multipath extension of OLSR, and can be regarded as a hybrid routing scheme because it combines the proactive nature of *topology sensing* and reactive nature of *multipath computation*. The auxiliary functions as *route recovery* and *loop detection* are introduced to improve the performance of the network. The usage of queue length metric for link quality criteria is studied and the compatibility between single path and multipath routing is discussed to facilitate the deployment of the protocol. The simulations based on *NS2* and *Qualnet* are performed in different scenarios. A testbed is also set up in the campus of Polytech'Nantes. The results from the simulator and testbed reveal that MP-OLSR is particularly suitable for mobile, large and dense networks with heavy network load thanks to its ability to distribute the traffic into different paths and effective auxiliary functions. The H.264/SVC video service is applied to ad hoc networks with MP-OLSR. By exploiting the scalable characteristic of H.264/SVC, we propose to use Priority Forward Error Correction coding based on Finite Radon Transform (FRT) to improve the received video quality. An evaluation framework called *SVCEval* is built to simulate the SVC video transmission over different kinds of networks in Qualnet. This second study highlights the interest of multiple path routing to improve quality of experience over self-organized networks.

Keywords: Mobile ad hoc network (MANET), multiple paths, OLSR, MP-OLSR, erasure channel coding, H.264/SVC