Demand-Assigment Mechanisms in GEO Satellite Systems and Resource-Optimized Algorithms for Multicast Applications in Wired and Wireless Networks

Relatore
Chiar.mo Prof. Francesco Delli Priscoli

Candidato
Ing. Dario Pompili

Anno Accademico 2003/2004
Dedico questa tesi a mio padre, mia madre e mia zia, sui quali ho sempre potuto fare affidamento, ed alla mia fidanzata Alessandra che con la sua musica tanto mi ha aiutato nel corso di questi anni di dottorato.
ACKNOWLEDGEMENTS

The author wishes to thank most sincerely Prof. Francesco Delli Priscoli for his continuing guidance in the completion of this work, as well as for his valuable support as advisor during the entire Ph.D. program.

The author is indebted to his colleagues and friends Ing. Marco Vittucci, Ing. Luca Lopez, Ing. Giuseppe Sette, and Ing. Gianfranco Santoro for their valuable work offered during the writing of this manuscript and sound comments that greatly improved the content and readability of this thesis.

To all the academic members of the System Engineering Department of the University of Rome “La Sapienza”, I wish to express my deepest gratitude for excellent advice, constructive criticism, helpful and critical reviews throughout the Ph.D. program.

Prof. Carlo Bruni’s position as a Ph.D. coordinator could not have been filled more adequately.

Last but not least, the author is grateful to the many anonymous reviewers that with their unselfish comments greatly improved the content of the papers from which this thesis has been partly extracted.
# TABLE OF CONTENTS

DEDICATION ................................................................. 1

ACKNOWLEDGEMENTS .................................................... 2

TABLE OF CONTENTS ..................................................... 2

LIST OF TABLES ........................................................... 7

LIST OF FIGURES .......................................................... 8

INTRODUCTION ............................................................. 11

0.1 First Part ............................................................. 11
  0.1.1 Chapter I ......................................................... 11
  0.1.2 Chapter II ....................................................... 12

0.2 Second Part ........................................................ 12
  0.2.1 Chapter III ....................................................... 12

0.3 Third Part ........................................................... 13
  0.3.1 Chapter IV ....................................................... 13
  0.3.2 Chapter V ....................................................... 15

0.4 Simulation Software Tools ......................................... 15

I DEMAND-ASSIGNMENT ALGORITHMS FOR SATELLITE BANDWIDTH ALLOCATION ......................................................... 16

1.1 Introduction .......................................................... 16

1.2 Basic Definitions and QoS Contract ................................ 19

1.3 Satellite Terminal System Architecture ............................ 22

1.4 Capacity Demand-Assignment Procedure .......................... 23

1.5 Markov Modulated Poisson Process based algorithm for Satellite Traffic Prediction .............................................. 27
  1.5.1 Data Generation in a Markov Modulated Poisson Process 28
  1.5.2 The Identification/Tuning Phase: the MMPP matching problem .......................................................... 31
  1.5.3 The Traffic Prediction Phase .................................... 33

1.6 Simulation Results .................................................... 33

1.7 Conclusions ........................................................... 40

II A CONTROL BASED TRAFFIC CONTROLLER PROVIDING QOS GUARANTEES ...................................................... 42

2.1 Introduction ........................................................... 42
III DIFFSERV-INTEGRATED ALGORITHMS FOR RESOURCE OPTIMIZATION IN SOURCE SPECIFIC AND GROUP SHARED MULTICAST APPLICATIONS IN INTERNET

3.1 Introduction

3.2 DIMRO - DiffServ-Integrated Multicast algorithm for Internet Resource Optimization
   3.2.1 A polynomial time algorithm for the exact determination of channel rates in a multirate multicast
   3.2.2 Least cost source rooted trees
   3.2.3 DIMRO in non-QoS-aware networks
   3.2.4 DIMRO in DiffServ aware networks

3.3 DIMRO-GS - DiffServ-Integrated Multicast algorithm for Internet Resource Optimization Group Shared applications
   3.3.1 DIMRO-GS in non-QoS aware networks
   3.3.2 DIMRO-GS in DiffServ aware networks

3.4 Simulation Results
   3.4.1 Random network model
   3.4.2 DIMRO Performance Evaluation
   3.4.3 DIMRO-GS Performance Evaluation
   3.4.4 DiffServ DIMRO and DIMRO-GS Performance Evaluation

3.5 Conclusions

IV MOBILE AD HOC NETWORKS (MANET)

4.1 The Notion of “Ad Hoc Network”

4.2 The Communication Environment and the MANET Model

4.3 A Promising Application: “Sensor Networks”

4.4 Mobility Model Introduction

4.5 Mobility Model Overview

4.6 Cellular Mobility Models
   4.6.1 Random Walk Mobility Model
4.6.2 Fluid-Flow Mobility Model .................................................. 106
4.6.3 Random Gauss-Markov Mobility Model .................................. 106
4.7 Ad Hoc Mobility Models .......................................................... 107
4.7.1 Random Mobility Model ...................................................... 108
4.7.2 Constant Velocity Random Direction Mobility Model .................. 108
4.7.3 Random Waypoint Mobility Model ......................................... 109
4.7.4 Random Direction Mobility Model ......................................... 110
4.7.5 A Probabilistic Version of the Random Mobility Model ............. 111
4.7.6 Smooth Random Mobility Model .......................................... 112
4.7.7 City Area, Area Zone, and Street Unit Mobility Models ............. 113
4.7.8 The Metropolitan, National, and International Mobility Models ... 115
4.7.9 Group Mobility Models ....................................................... 115

V A PROBABILISTIC PREDICTIVE MULTICAST ALGORITHM IN AD HOC NETWORKS (PPMA) .................................................. 129
5.1 Introduction .............................................................................. 129
5.2 Related Work .......................................................................... 131
5.2.1 PAST-DM: Progressively Adapted Sub-Tree in Dynamic Mesh .... 131
5.2.2 ITAMAR: Independent-Tree Ad hoc Multicast Routing ............. 132
5.2.3 AODV: Ad hoc On Demand Distance Vector protocol ............. 133
5.3 Problem Setup ......................................................................... 134
5.3.1 Motivations and Goals ......................................................... 134
5.3.2 Transmission Energy Model .................................................. 135
5.3.3 Probabilistic Link Cost ......................................................... 135
5.4 Probabilistic Link Cost Function Terms ..................................... 137
5.4.1 Energy Term ........................................................................ 137
5.4.2 Distance Term ...................................................................... 139
5.4.3 Lifetime Term ...................................................................... 141
5.5 Centralized PPMA .................................................................... 146
5.5.1 Distance-Based Criterion ...................................................... 152
5.5.2 Link Cost-Based Criterion .................................................... 154
5.6 Distributed PPMA ..................................................................... 156
5.7 Performance Evaluation ............................................................. 161
5.7.1 Network Mobility Model ....................................................... 162
5.7.2 Simulation Results ............................................................... 164
<table>
<thead>
<tr>
<th>Section</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>Conclusions</td>
<td>170</td>
</tr>
<tr>
<td>FINAL CONCLUSIONS AND FUTURE RESEARCH</td>
<td>173</td>
</tr>
<tr>
<td>APPENDIX</td>
<td>176</td>
</tr>
<tr>
<td>LIST OF ACRONYMS</td>
<td>178</td>
</tr>
<tr>
<td>REFERENCES</td>
<td>183</td>
</tr>
<tr>
<td>VITA</td>
<td>194</td>
</tr>
<tr>
<td></td>
<td>Table Title</td>
</tr>
<tr>
<td>---</td>
<td>-------------------------------------------------------</td>
</tr>
<tr>
<td>1</td>
<td>E-mail parameter definitions</td>
</tr>
<tr>
<td>2</td>
<td>File Transfer parameter definitions</td>
</tr>
<tr>
<td>3</td>
<td>Web Browsing parameter definitions</td>
</tr>
<tr>
<td>4</td>
<td>Video Conferencing parameter definitions</td>
</tr>
<tr>
<td>5</td>
<td>Voice over IP parameter definitions</td>
</tr>
<tr>
<td>6</td>
<td>Shortened variable names</td>
</tr>
<tr>
<td>7</td>
<td>IP traffic source statistical parameters</td>
</tr>
<tr>
<td>8</td>
<td>Average and peak rates of traffic sources</td>
</tr>
<tr>
<td>9</td>
<td>IP packet QoS parameters</td>
</tr>
<tr>
<td>10</td>
<td>Network parameters</td>
</tr>
<tr>
<td>11</td>
<td>Network node mobility</td>
</tr>
<tr>
<td>12</td>
<td>Simulation parameters</td>
</tr>
<tr>
<td>13</td>
<td>Multicast group size</td>
</tr>
<tr>
<td>Figure</td>
<td>Description</td>
</tr>
<tr>
<td>--------</td>
<td>-----------------------------------------------------------------------------</td>
</tr>
<tr>
<td>1</td>
<td>Downstream and upstream traffic in a Satellite Access Network</td>
</tr>
<tr>
<td>2</td>
<td>Scenario Overview</td>
</tr>
<tr>
<td>3</td>
<td>Internal structure of the Satellite Terminal (ST) at time $h$</td>
</tr>
<tr>
<td>4</td>
<td>Example of the proposed capacity assignment procedure ($N = 2$)</td>
</tr>
<tr>
<td>5</td>
<td>Markov Modulated Poisson Process (MMPP) State Chain</td>
</tr>
<tr>
<td>6</td>
<td>Flow chart of the MMPP prediction algorithm</td>
</tr>
<tr>
<td>7</td>
<td>OPNET Simulation Scenario</td>
</tr>
<tr>
<td>8</td>
<td>Traffic Source parameters and their statistical behaviour</td>
</tr>
<tr>
<td>9</td>
<td>Queuing delay Cumulative Distribution Functions (CDFs) of Voice (Fig. 9(a)) and Video (Fig. 9(b)) IP datagrams</td>
</tr>
<tr>
<td>10</td>
<td>Voice bit loss percentile comparisons expressed for significant capacity values</td>
</tr>
<tr>
<td>11</td>
<td>Video bit loss percentile comparisons expressed for significant capacity values</td>
</tr>
<tr>
<td>12</td>
<td>TCP average bit rate comparisons expressed for significant capacity values</td>
</tr>
<tr>
<td>13</td>
<td>SATIP6 reference scenario</td>
</tr>
<tr>
<td>14</td>
<td>Internal structure of the HS Controller</td>
</tr>
<tr>
<td>15</td>
<td>Scheme of the control system</td>
</tr>
<tr>
<td>16</td>
<td>PID Controller</td>
</tr>
<tr>
<td>17</td>
<td>Offered Traffic</td>
</tr>
<tr>
<td>18</td>
<td>Link Efficiency vs System Congestion</td>
</tr>
<tr>
<td>19</td>
<td>Determining channel rates: Step 1</td>
</tr>
<tr>
<td>20</td>
<td>Determining channel rates: Step 2</td>
</tr>
<tr>
<td>21</td>
<td>Determining channel rates: Step 3</td>
</tr>
<tr>
<td>22</td>
<td>Determining channel rates: Step 4</td>
</tr>
<tr>
<td>23</td>
<td>Determining channel rates: Step 5</td>
</tr>
<tr>
<td>24</td>
<td>Correct path computation</td>
</tr>
<tr>
<td>25</td>
<td>DIMRO and OSTP Rejection Rate in Network 1</td>
</tr>
<tr>
<td>26</td>
<td>DIMRO and OSTP Network Load in Network 1</td>
</tr>
<tr>
<td>27</td>
<td>DIMRO and OSTP Rejection Rate in Network 2</td>
</tr>
<tr>
<td>28</td>
<td>DIMRO and OSTP Network Load in Network 2</td>
</tr>
<tr>
<td>29</td>
<td>DIMRO-GS and FTM Rejection Rate in Network 1</td>
</tr>
<tr>
<td>30</td>
<td>DIMRO-GS and FTM Network Load in Network 1</td>
</tr>
<tr>
<td>31</td>
<td>DIMRO-GS and FTM Rejection Rate in Network 2</td>
</tr>
</tbody>
</table>
Distributed PPMA: computation of current_cost and new_cost as in Algorithm 3.

Tree lifetime for PPMA and Steiner algorithm for small multicast groups in a medium mobility (Fig. 61(a)) and high mobility environment (Fig. 61(b))

Tree lifetime for PPMA and Steiner algorithm for large multicast groups in a medium mobility (Fig. 62(a)) and high mobility environment (Fig. 62(b))

Number of connected receivers for PPMA and Steiner algorithm for small multicast groups in a medium mobility (Fig. 63(a)) and high mobility environment (Fig. 63(b))

Number of connected receivers for PPMA and Steiner algorithm for large multicast groups in a medium mobility (Fig. 64(a)) and high mobility environment (Fig. 64(b))

Average available node energy for PPMA and Steiner algorithm for small multicast groups in a medium mobility (Fig. 65(a)) and high mobility environment (Fig. 65(b))

Average available node energy for PPMA and Steiner algorithm for large multicast groups in a medium mobility (Fig. 66(a)) and high mobility environment (Fig. 66(b))
INTRODUCTION

The present thesis has been developed in the Laboratory of the System Engineering Department of the University of Rome “La Sapienza”. This thesis presents resource optimized algorithms for efficient bandwidth sharing in telecommunication systems, and is divided into three parts: the first part presents demand-assignment mechanisms in GEO (Geostationary Earth Orbit) satellite systems, while the second and the third part propose resource-optimized algorithms for multicast applications in wired and wireless networks, respectively. In the following each part of the thesis, which is composed of one or two chapters, is separately described, and the main contributions of each chapter is outlined.

0.1 First Part

The first part of this thesis is composed of two chapters, Chapter I and Chapter II, and deals with the problem of the design of control-predictive based demand-assignment mechanisms for a satellite network guaranteeing a target QoS (Quality of Service) to Internet traffic, while efficiently exploiting the air interface. The proposed mechanisms are in charge of dynamically partitioning the satellite uplink capacity among the connections in progress in the considered spot-beam. Such a partitioning is performed aiming, on the one hand, to match the QoS requirements of each connection and, on the other hand, to maximize bandwidth exploitation.

0.1.1 Chapter I

Chapter I describes a Control Theory approach, based on a Markov Modulated Chain Prediction Model for the traffic forecast, to effectively tackle the problem of the delay between the bandwidth request and bandwidth assignment, and the signaling overhead caused by control messages. The presented algorithms efficiently cope with both the satellite propagation delay and the delays inherent in the periodic nature of the bandwidth demand-assignment mechanism. The demand-assignment algorithm and the proposed Markov Modulated Chain traffic prediction model are shown to be easy to implement and to improve the overall satellite network performance.
0.1.2 Chapter II

Chapter II presents an innovative control-based Traffic Controller whose objective is to share a single resource, namely the available bit rate, among a set of IP flows characterized by different QoS requirements. This Traffic Controller has been developed for a satellite environment, in the framework of the SATIP6 (SATEllite broadband multimedia system for IPv6) project [3] financed by the European Commission in the context of the 5th framework programme (IST-Information Society and Technology programme). Nevertheless, the proposed Traffic Controller can be profitably used in New Generation IP-based Satellites [89] or in any wireless or wired network. The main novelty behind the proposed Traffic Controller is that it relies on an original control based approach which exploits a closed-loop architecture and a fuzzy tuner. Simulation results demonstrated that such Traffic Controller outperforms a well known open-loop Traffic Controller (namely the one based on the presence of Dual Leaky Buckets and Earliest Deadline First scheduling algorithm).

0.2 Second Part

The second part of the thesis is composed of Chapter III only.

0.2.1 Chapter III

Chapter III introduces two novel multicast algorithms for wired Internet resource optimization. The first proposed multicast algorithm, DIMRO (DiffServ-Integrated Multicast algorithm for Internet Resource Optimization), builds source rooted multicast trees to allow one node in the multicast group to send data to the other member nodes. This algorithm is suitable for source specific applications such as live video distribution, software and file distribution, replicated database server, web site replication, periodic data delivery. The second proposed multicast algorithm, DIMRO-GS (DiffServ-Integrated Multicast algorithm for Internet Resources Optimization in Group Shared applications), constructs the shared tree connecting each group member with all the other group members for group shared applications such as videoconference, distributed games, file sharing, collaborative groupware, replicated database. The objective is to achieve efficient traffic balancing in the network in order to avoid bandwidth bottlenecks and consequent network partitions, which cause low network performance. Both algorithms can be
integrated with the Differentiated Service (DiffServ) Quality of Service approach. The QoS services requested by the receivers are mapped into the proper DiffServ class, such a way to respect the expected QoS requirements. The low computational complexity of the proposed algorithms effectively leads to time and resource saving.

0.3 Third Part

The third and last part of this thesis, which is made up of two chapters, Chapter IV and Chapter V, proposes efficient multicast algorithms in Ad Hoc Networks.

0.3.1 Chapter IV

Chapter IV describes the main characteristics of Ad Hoc Networks, which are collections of mobile nodes communicating using wireless media without any fixed infrastructure, as well as their features and applications. A Mobile Ad Hoc Network (MANET) is a network architecture that can be rapidly deployed without relying on pre-existing fixed network infrastructure. The nodes in a MANET can dynamically join and leave the network, frequently, often without warning, and possibly without disruption to other ongoing communications. Moreover, the nodes in the network can be highly mobile. Examples of the use of the MANETs are tactical operation, for fast establishment of military communication during the deployment of forces in unknown and hostile terrain; rescue missions, for communication in areas without adequate wireless coverage; national security, for communication in times of national crisis, where the existing communication infrastructure is non-operational due to a natural disaster or a global war; law enforcement, for fast establishment of communication infrastructure during law enforcement operations; commercial use, for setting up communication in exhibitions, conferences, or sales presentations; education, for operation of wall-free (virtual) classrooms; sensor networks, for communication between intelligent sensors mounted on mobile platforms. Nodes in a MANET exhibit nomadic behavior by freely migrating within some area, dynamically creating and tearing down associations with other nodes. Groups of nodes that have a common goal can create formations (clusters) and migrate together, similarly to military units on missions or to guided tours on excursions. Nodes are equipped with portable communication devices. Lightweight batteries may power these devices. Limited battery life can impose restrictions on the transmission range, communication activity (both transmitting and receiving) and computational power of these devices. In addition, all the network nodes have equal capabilities.
This means that all nodes are equipped with identical communication devices and are capable of performing functions from a common set of networking services. However, all nodes do not necessarily perform the same functions at the same time. In particular, nodes may be assigned specific functions in the network, and these roles may change over time.

Chapter IV also presents the most popular mobility models used for simulations in ad hoc networks. The movement pattern of users plays an important role in performance analysis of mobile and wireless networks. There exists a variety of mobility models that find application in different kinds of simulations and analytical studies of wireless systems. Cellular mobility models were developed to test the behavior of cellular protocols and strategies. The most known cellular mobility model is the Random Walk Model. In the Random Walk Model, a host moves from its current location to a new location by randomly choosing a direction and speed in which to travel. In ad hoc wireless mobile networks, the mobility models focus on the individual motion behavior between mobility ‘epochs’, which are the smallest time periods in a simulation in which a mobile host moves in a constant direction at a constant speed. A good mobility model should not change the spatial distribution of the nodes during the simulation. A property often required for mobility models is to keep uniform that distribution over the entire simulation area. In some mobility model we encounter an unrealistic choppy motion with sudden stopping, sharp turning, and completely random wandering. A realistic node motion is another fundamental property a mobility model must have. The most used mobility model in ad hoc network simulations is the Random Waypoint Mobility Model. In this model, each node of the network chooses uniformly at random a destination point, i.e., ‘waypoint’, in a rectangular deployment region. A node moves to this destination with a randomly chosen velocity. When it reaches the destination, it remains static for a predefined pause time and then starts moving again according to the same rule. In many situations it is necessary to model the behavior of mobile nodes that move together. For example, many military scenarios occur where a group of soldiers must collectively search a particular plot of land in order to destroy land mines, capture enemy attackers, or simply work together in a cooperative manner to accomplish a common goal. In order to model such situations, group mobility models exist to account for these new cooperative characteristics.
0.3.2 Chapter V

Chapter V proposes PPMA, a new Probabilistic Predictive Multicast Algorithm in ad hoc networks, which leverages the tree delivery structure for multicasting, solving its drawbacks in terms of lack of robustness and reliability in highly mobile environment. Existing multicast protocols fall short in a harsh ad hoc mobile environment because node mobility causes conventional multicast trees to rapidly become outdated. The amount of bandwidth resources required for building up a multicast tree is commonly less than that required for other delivery structures, since a tree avoids unnecessary duplication of data. However, a tree structure is more subject to disruption due to link/node failure and node mobility than more meshed structures. This chapter explores these contrasting issues, and proposes PPMA to solve most of the existing problems for multicasting in MANETs. By exploiting the non-deterministic nature of ad hoc networks, the proposed algorithm takes into account the estimated network state evolution in terms of node residual energy, link availability, and node mobility forecast, in order to maximize the multicast tree lifetime, and consequently reduce the number of costly tree reconfigurations. PPMA is provided in both its centralized and distributed version.

0.4 Simulation Software Tools

For all the proposed algorithms this thesis presents performance evaluation through extensive simulations run on software tools such as OPNET [1] or C++ based simulators. The presented Integer Linear Problems (ILP), which are used as benchmarks for the performance evaluation of the developed algorithms, are implemented in AMPL [64] and solved with CPLEX [2].
CHAPTER I

DEMAND-ASSIGNMENT ALGORITHMS FOR SATELLITE BANDWIDTH ALLOCATION

1.1 Introduction

High powered direct broadcast television satellites, using the European DVB (Digital Video Broadcast) standard, can be used to broadcast high volumes of data directly to home terminals. At present, these are unidirectional transmission channels that do not allow an interaction between service providers and users. There are several ways to design a return channel for satellite broadcast/multicast services. Many believe terrestrial return channels to be the most cost effective and practical solution. Commonly proposed terrestrial return channels are PSTN, ISDN, xDSL, GSM/GPRS, etc. However, there is a large world-wide interest for a DVB Return Channel via Satellite (DVB-RCS) [4], which could be particularly suitable to support a large amount of non real-time return connections provided that an appropriate bandwidth management mechanism is designed (ASTRA BBI system).

In this chapter we propose a dynamic bandwidth management mechanism for an efficient and flexible partitioning of the uplink capacity available in a given spot-beam among the connections in progress in such spot-beam. This capacity is equal to the sum of the uplink carrier capacities assigned to such spot-beam. Such a partitioning is performed aiming, on the one hand, to match the Quality of Service (QoS) requirements of each connection and, on the other hand, to maximize the bandwidth exploitation. A Control Theory approach, based on a Markov Modulated Chain Prediction Model for the traffic forecast, is adopted to effectively tackle the problem of the delay between the bandwidth request and bandwidth assignment, and the signaling overhead caused by control messages [46][45][134]. The proposed mechanism is fully compliant with the DVB-RCS standard [4]. In the following we will refer to the term downstream and upstream to indicate the traffic flowing from the Internet to a Satellite Terminal (ST) and from a ST to the Internet, respectively, as shown in Fig. 1.

In the considered scenario (see Fig. 2), an on-board packet switch is present in charge of addressing the IP datagrams (hereon referred to as packets) towards a downlink carrier assigned
The considered scenario is outlined in Fig. 2 which shows a 2-layer switch Satellite system including a Hub Station (HS), a certain number of Satellite Terminals (STs) and a Network Control Centre (NCC) in charge of performing traffic control tasks [123]. In the return direction, each ST can manage one or more upstream(s) coming from the User Terminals (UTs). Each upstream is relevant to a different connection involving an UT and entails the transmission of packets from the considered UT, via the associated ST and the satellite, to the HS and the backbone IP network (see Fig. 1(b)). Each connection has its own specific requirements (e.g., in terms of minimum required bandwidth and maximum transfer delay) which are reflected in a specific QoS Contract established at connection set-up [13][38] (see Section 1.2).

Clearly, in order to enhance the exploitation of the valuable satellite capacity, connections which have no very stringent delay requirements are not fixedly assigned uplink bandwidth portions. For such kind of connections, uplink bandwidth has to be managed according to demand-assignment mechanisms in which the STs periodically ask the NCC for the temporary assignment of a certain portion of bandwidth. Then, the NCC, on the grounds of the received bandwidth requests, also considering the actual QoS requirements of the connections asking for bandwidth assignment, decides how the available uplink bandwidth should be optimally

Figure 1: Downstream and upstream traffic in a Satellite Access Network
Figure 2: Scenario Overview

partitioned, and communicate the relevant decisions to the STs.

A key problem of such a kind of mechanisms relies on the fact that, because of the high propagation delays of satellite networks, especially for Geostationary Earth Orbit (GEO) satellite [130] where the one-way propagation delay passing through the satellite equals 250 ms, bandwidth assignments are received fractions of second after bandwidth requests (about half a second in GEO systems).

In addition, in order to keep signaling overhead limited, a certain minimum time interval must elapse between two consecutive bandwidth requests from the same ST. These issues can cause further delays in data transfer (which would increase the unavoidable satellite propagation delay), as well as ST buffer overflows. A possible solution to such problems could be to ask for more bandwidth than that actually necessary at the time of bandwidth request. Nevertheless, by so doing, there is the risk of the so-called over-assignments, i.e., the assignment to a certain ST of capacity not actually necessary, which, possibly, is subtracted to other STs actually needing it.

This chapter copes with the above-mentioned problem, by designing an innovative demand-assignment mechanism and an original traffic prediction model based on a Markov Modulated Poisson Process (MMPP). The primary objective of these algorithms is, on the one hand, to
avoid further delays in data transfer and, on the other hand, to guarantee an efficient exploitation of uplink satellite bandwidth. Although different works have considered the Demand Assigned Multiple Access (DAMA) problem [113][93][67], up to the author’s knowledge a few papers have addressed the demand-assignment problem with QoS guarantees in systems subject to delays [57][54][52][7][8].

The chapter is organized as follows. Section 1.2 introduces the basic concepts utilized in the chapter, as well as the objectives of the designed procedures. Section 1.3 presents the Satellite Terminal (ST) architecture. Section 1.4 describes the proposed demand-assignment procedure. Section 1.5 describes the proposed Markov Modulated Poisson Process algorithm which our demand-assignment procedure is based on for the traffic prediction. Section 1.6 shows numerical results obtained through extensive simulation campaign run on a satellite simulator developed with OPNET. Finally, Section 1.7 concludes the chapter sketching the main achieved results.

1.2 Basic Definitions and QoS Contract

In the following, for the sake of brevity, by “uplink” we always mean the “return uplink”, i.e., the link from the ST to the satellite. By “return” traffic (or “return” packets) we indicate the traffic (or the packets) originated by the UTs and directed to the HS via the STs and the satellite.

Let \( S \) denote the number of different STs that are in the considered spot-beam. Let \( i \) denote a generic ST with at least an in progress connection; so, \( i \) can assume the following values: 1, 2, .., \( S \). Let \( C(i) \) denote the number of different uplink connections simultaneously in progress involving the \( i^{th} \) ST. Let \((i, j)\) denote a generic in progress uplink connection involving the \( i^{th} \) ST; so, \( j \) can assume the following values: 1, 2, .., \( C(i) \). Practically, the computations of the various variables will not be performed at any time \( t \), but only at discrete-time instants \( t_h \) \((h = 1, 2, ..)\) periodically occurring with a proper period \( T_{short} \) (expressed in seconds), i.e., \( t_{h+1} = t_h + T_{short} \). So, in the following, we will just refer to these discrete-time instants and, for the sake of notation simplicity, the discrete-time instant \( t_h \) will be indicated as \( h \). By \( h^{th} \) time interval, we will mean the time interval \([h, h+1]\). Let \( R_{ij}(h) \) denote the bit rate of the return traffic relevant to the connection \((i, j)\) which, at time \( h \), is offered to the \( i^{th} \) ST. Such bit rate is computed during an appropriate monitoring period according to the following relationship,

\[
R_{ij}^{in}(h) = \frac{\sum_{k=h-M}^{h} L_{ij}^{in}(k)}{M \cdot T_{short}}
\]
where:

\( M \) is the duration, expressed in number of discrete-time instants, of the monitoring period (i.e., \( M \cdot T_{short} \) is the monitoring period duration expressed in seconds);

\( L_{ij}^n(k) \) is the sum of the lengths (expressed in bits) of the return packets, relevant to the connection \((i, j)\), which, during the \( k^{th} \) time interval, are incoming into the \( i^{th} \) ST.

**Definition 1** Let \( D_{ij} \) denote the queuing delay (expressed in seconds) which a packet relevant to the connection \((i, j)\) experiences from the time at which it arrives at the ST (coming from an UT) to the time at which it may be forwarded towards the uplink air interface.

**Definition 2** Let \( R_{av}^{ij} \) denote the average throughput of a connection \((i, j)\), defined as the ratio between the number of bits transmitted during the connection lifetime and the connection duration.

The QoS guarantees which have to be granted to a connection are specified in a QoS Contract [13][38], established at connection set-up. The QoS Contract relevant to the connection \((i, j)\) includes the following requirements:

1. A first QoS requirement concerns the definition of the traffic to be anyhow granted to the connection \((i, j)\), i.e., the so-called Static Bit Rate, \( R_{ij}^{static} \). Note that \( R_{ij}^{static} \) can vary as time varies. Nevertheless, since these variations are slow with respect to the demand-assignment procedures dealt with in this chapter, they will not be further considered. An appropriate Connection Admission Control (CAC) procedure assures that the Static Bit Rates of the in progress connections satisfy the following constraint:

\[
\sum_{i=1}^{S} \sum_{j=1}^{C(i)} R_{ij}^{static} \leq R_{tot}^{up} \tag{2}
\]

where \( R_{tot}^{up} \) is the overall uplink capacity available in the considered spot-beam.

2. If the connection \((i, j)\) is a Real Time connection (voice, video-conference, etc.), a second fundamental QoS requirement (hereon referred to as Delay QoS Requirement) concerns the maximum transfer delay, hereafter indicated as \( D_{ij}^{max} \), which can be tolerated by the connection \((i, j)\). This means that, in general, the queuing delay \( D_{ij} \) should not exceed \( D_{ij}^{max} \). As a matter of fact, in case the queuing delay \( D_{ij} \) exceeds \( D_{ij}^{max} \) the packet is no more meaningful and it is discarded by the ST without being transmitted towards the
satellite. In this respect, for Real Time connections a small amount of packet loss can be tolerated, according to the specific real time application, even though such a loss should be minimized.

In light of the above, it should be clear that the QoS Contract relevant to a Real Time connection \((i, j)\) is characterized by the two parameters \(R_{ij}^{static}\) and \(D_{ij}^{max}\), while the QoS Contract relevant to a Non Real Time connection \((i, j)\) is characterized by the only parameter \(R_{ij}^{static}\).

The algorithms proposed in this chapter aim at the minimization of the Real Time traffic to be discarded because has waited more than the maximum tolerated delay (QoS requirement 2) and at the maximization of the average throughput of the Non Real Time traffic. In this work we assume the traffic packet expiration due to the \(D_{ij}^{max}\) overcome as the only possible traffic loss, meaning that no queue overflows can occur. In other words, we make the assumption that each queue is dimensioned such a way to accept all possible coming packets. This way Critical Traffic will not suffer, under heavy traffic conditions, poor QoS due to queue length exceeding, as shown in the simulation results.

The \(i^{th}\) ST avails of a semi-permanently assigned Static Bit Rate equal to the sum of the static bit rates \(R_{ij}^{static}\) relevant to the connections in progress at such ST. Moreover, it avails of a Dynamic Bit Rate which is temporarily granted by the NCC following the ST requests, according to an appropriate demand-assignment mechanism (detailed in Section 1.4). Let \(R_{up}^{dyn}\) denote the available dynamic uplink capacity defined as the uplink capacity relevant to the considered spot-beam which is not statically assigned, i.e., the capacity which can be dynamically assigned. Such a capacity can be easily computed according to the following equation:

\[
R_{up}^{dyn} = R_{up}^{tot} - \sum_{i=1}^{S} \sum_{j=1}^{C(i)} R_{ij}^{static}
\]  

(3)

Let \(R_{ij}^{dyn}[h_1, h_2]\) denote the Dynamic Bit Rate assigned to the connection \((i, j)\) during the time interval \([h_1, h_2]\). Clearly, for any time interval \([h_1, h_2]\), the following uplink capacity constraint must be respected:

\[
\sum_{i=1}^{S} \sum_{j=1}^{C(i)} R_{ij}^{dyn} \leq R_{up}^{dyn} \forall h \in [h_1, h_2]
\]  

(4)
1.3 Satellite Terminal System Architecture

The $i^{th}$ ST is provided with a set of $C(i)$ FIFO Buffers (see Fig. 3): each of these buffers stores the packets (waiting for being transmitted in the uplink channel) of one of the uplink connections the ST in question is involved in. A Classifier, which is fed with the traffic coming from the UTs linked to the $i^{th}$ ST, is in charge of sorting the packets towards the $C(i)$ FIFO Buffers. In the following, for the sake of brevity, the FIFO Buffer storing the packets relevant to the connection $(i, j)$ will be simply referred to as queue $(i, j)$. Let $q_{ij}(h)$ denote the number of bits stored in the queue $(i, j)$ at time $h$. Let $\delta_{ij}^{ass}[h_1, h_2]$ denote the fraction of the available dynamic uplink capacity $R_{up}^{dyn}$ granted by the NCC to the connection $(i, j)$ for being used during the time interval $[h_1, h_2]$. In other words, $\delta_{ij}^{ass}(h) \cdot R_{up}^{dyn}$ represents the Dynamic Bit Rate at which, during the time interval $[h_1, h_2]$, the $i^{th}$ ST is allowed to forward packets, relevant to the connection $(i, j)$, towards the uplink air interface. Thus, the parameter $\delta_{ij}^{ass}[h_1, h_2]$ is always included in the range $[0, 1]$ and, for any time interval $[h_1, h_2]$, the following fundamental uplink capacity constraint (which is equivalent to (4)) must be respected:

$$\sum_{i=1}^{S} \sum_{j=1}^{C(i)} \delta_{ij}^{ass}(h) \leq 1 \forall h$$  (5)
The packets stored in the queue \((i, j)\) can be either forwarded over the uplink air interface or, if they are relevant to Real Time connections, discarded because they are expired, i.e., they have waited more than the maximum tolerated delay \(D_{ij}^{\text{max}}\).

### 1.4 Capacity Demand-Assignment Procedure

Let us introduce the following definitions (see also Fig. 4):

- Let \(L\) denote the round-trip delay expressed in number of time intervals. Such a delay \(L\) is equal to \(2 \cdot \lceil \left( D_{\text{prop}} + T_{\text{comput}} \right) / T_{\text{short}} \rceil\), where \(D_{\text{prop}}\) is the maximum propagation delay in the run from any ST to the NCC, or in the opposite run, and \(T_{\text{comput}}\) is the ST (or NCC) demand-assignment computing time;

- Let \(\eta\) denote the generic discrete-time at which a ST performs a bandwidth demand; these times will be referred to as demand times. In this chapter, for the sake of simplicity, we assume that all STs synchronously carry out their bandwidth demands, i.e., these demands are all performed at the same demand times. Moreover, we assume that bandwidth demands are periodically performed. Nevertheless, the concepts can be straightforwardly extended to the case in which the ST demands are asynchronous and the bandwidth demands are not periodic;

- Let \(T_{\text{inf}}\) denote the period occurring between two consecutive bandwidth requests, expressed in number of time intervals;

- Let \(N\) denote the ratio between \(L\) and \(T_{\text{inf}}\), i.e., \(N = L / T_{\text{inf}}\) (for instance, Fig. 4 is relevant to the case \(N=2\)). The choice of the parameter \(N\) has to be carried out by carefully trading-off the contrasting requirements, on the one hand, of frequently sending the bandwidth requests (thus allowing a tight tracking of the traffic arrived at the STs coming from the UTs) and, on the other hand, of limiting the signaling overhead caused by such bandwidth requests. Clearly, the former and the latter requirements drive towards high and low values for the parameter \(N\), respectively;

- Let \(R^{in}_{ij}[\eta, h]\) denote the predictions, performed by the \(i^{th}\) ST at time \(\eta\), of the average bit rate which will enter the queue \((i, j)\) during the \(h^{th}\) time interval (i.e., \(R^{in}_{ij}[\eta, h] \cdot T_{\text{short}}\) represent the prediction of the number of bits entering the queue \((i, j)\) during the \(h^{th}\) time
interval). In Section 1.5, an ad-hoc Markov Modulated Poisson Process (MMPP) prediction model is presented to perform such predictions. This model shows to outperform other considered prediction models in the simulated scenarios.

In the proposed demand-assignment mechanism the STs do not directly calculate the bandwidth they require. Conversely, they just send to the NCC some key parameters which are used by the NCC itself to perform appropriate bandwidth assignments.

So, whenever at a time $\eta$ a bandwidth demand has to be performed, the $i^{th}$ ST sends to the NCC the following pieces of information:

1. The $C(i)$ predictions of the lengths of the queues $(i, j)$ ($j = 1, 2, ..., C(i)$) at time $\eta + T_{inf}$; at time $\eta$, the $i^{th}$ ST computes these predictions, indicated as $q_{ij}^*(\eta + T_{inf})$, according to the following equation:

$$q_{ij}^*(\eta + T_{inf}) = q_{ij}(\eta) + \sum_{k=\eta}^{\eta+T_{inf}-1} R_{ij}^{in} [\eta, k] \cdot T_{short} - \delta_{ass} [\eta, \eta + T_{inf}] \cdot R_{up}^{dyn} \cdot T_{inf} \quad (6)$$

2. The $C(i)$ predictions of the average bit rates of the traffic which will enter the queue $(i, j)$ during the time interval $[\eta + T_{inf}, \eta + T_{inf} + L]$; at time $\eta$, the $i^{th}$ ST computes these predictions, indicated as $R_{ij}^{in} [\eta + T_{inf}, \eta + T_{inf} + L]$, according to the following equation:

$$R_{ij}^{in} [\eta + T_{inf}, \eta + T_{inf} + L] = \sum_{k=\eta + T_{inf}}^{\eta + T_{inf} + L - 1} \frac{R_{ij}^{in} [\eta, k]}{L} \quad (7)$$

3. The $C(i)$ coefficients $\beta_{ij}$ used to grant an higher weight to the queues $(i, j)$ relevant to Real Time connections which are losing bits because of packet expiries. These coefficients are computed according to the following expression:

$$\beta_{ij} = 1 + K^{opt} \cdot \frac{B_{ij}^{loss}[\eta - T_{inf}, \eta]}{B_{ij}^{out}[\eta - T_{inf}, \eta]} \quad (8)$$

where $B_{ij}^{loss}[\eta - T_{inf}, \eta]$ represents the amount of bits discarded from the queue $(i, j)$ during the time interval $[\eta - T_{inf}, \eta]$ due to packet expiries; $B_{ij}^{out}[\eta - T_{inf}, \eta]$ represents the whole amount of bits that, during the same time interval, is either retrieved from the queue $(i, j)$ for being forwarded towards the air interface, or discarded from the queue $(i, j)$ due to packet expiries. $K^{opt}$ is an appropriate optimized feedback gain whose value determines the aggressiveness of the feedback control law (the optimal value $K^{opt}=0.97$ has been chosen in the simulation experiments reported in Section 1.6).
Basing on the pieces of information received from the STs, at time $\eta + L/2$, the NCC, basing on the information received from each ST, computes the capacity assignments for the time interval $[\eta + L/2; \eta + L + T_{inf}]$ and broadcast them to each ST.

At $h = \eta$, each ST computes the predicted queue lengths $q^*_{ij}(\eta + T_{inf})$, the predicted incoming traffic $R_{in}^{*}$ in the time interval $[\eta; \eta + L]$ and the coefficients $\beta_j$.

These parameters are sent to the NCC.

For the computation of $q^*_{ij}(\eta + T_{inf})$ each ST exploits the capacity assignments, relevant to the time interval $[\eta; \eta + T_{inf}]$, arrived at this ST, coming from the NCC, at $h = \eta$.

At $h = \eta + L$, each ST receives the capacity assignments which the NCC decided at time $\eta + L/2$. In the time interval $[\eta + L; \eta + L + T_{inf}]$ each ST will regulate its transmissions to match these capacity assignments.

**Figure 4:** Example of the proposed capacity assignment procedure ($N = 2$)

Basing on the pieces of information received from the STs, at time $\eta + L/2$, the NCC has to decide the capacity assignments $\delta_{ij}^{ass}[\eta + L, \eta + L + T_{inf}]$ for any $(i, j)$ pair (see Fig. 4). Then, the proposed idea is to select these assignments aiming at emptying the ST queues at time $\eta + L + T_{inf}$ (this is just the last time at which the assignment decided by the NCC at time $\eta + L/2$ will be effective). The expected length of the queue $(i, j)$ at time $\eta + L + T_{inf}$ can be computed according to the following equation:

$$q^*_{ij}(\eta + L + T_{inf}) = q^*_{ij}(\eta + T_{inf}) + R_{in}^{*}[\eta + T_{inf}, \eta + T_{inf} + L] \cdot L +$$

$$- \{ \delta^{ass}_{ij}[\eta + T_{inf}, \eta + L] + \delta^{ass}_{ij}[\eta + L, \eta + L + T_{inf}] \} \cdot R_{up}^{dyn} \cdot L$$

(9)

Note that the term $\delta^{ass}_{ij}[\eta + T_{inf}, \eta + L]$ appearing in (9) is relevant to capacity assignments already performed by the NCC at the time(s) previous to $\eta + L/2$. Eq. (9) assumes $L > T_{inf}$ which is the most common case. Nevertheless, the extension to the opposite case is straightforward. Then, the target capacity assignments, indicated as $\delta^{ass}_{ij}[\eta + L, \eta + L + T_{inf}]$, which the NCC, at time $\eta + L/2$, needs to assign in order to empty the ST queues at time $\eta + L + T_{inf}$, can be obtained by imposing that the right hand side of the previous equation is equal to zero,
meaning that $q_{ij}^*(\eta + L + T_{inf}) = 0$. This yields:

$$
\delta_{ij}^{ass*}[\eta + L, \eta + L + T_{inf}] = \frac{q_{ij}^*(\eta + T_{inf})}{R_{up}^{dyn} \cdot L} + \frac{R_{ij}^{in*}[\eta + T_{inf}, \eta + T_{inf} + L]}{R_{up}^{dyn}} - \delta_{ij}^{ass}[\eta + T_{inf}, \eta + L] \tag{10}
$$

However, the target capacity assignment can be actually granted only if the uplink capacity constraint, expressed in (5), is satisfied. In order to force the respect of this constraint and to take into account the parameters $\beta_{ij}$ in (8), the actual capacity assignments $\delta_{ij}^{ass}[\eta + L, \eta + L + T_{inf}]$ can be computed according to the following expression:

$$
\delta_{ij}^{ass}[\eta + L, \eta + L + T_{inf}] = \frac{\beta_{ij} \cdot \delta_{ij}^{ass*}[\eta + L, \eta + L + T_{inf}]}{\sum_{i=1}^{S} \sum_{j=1}^{C(i)} \beta_{ij} \cdot \delta_{ij}^{ass*}[\eta + L, \eta + L + T_{inf}]} \tag{11}
$$

Note that, by using the above-mentioned expression, the uplink capacity constraint is satisfied with the sign of equality meaning that all the available dynamic uplink capacity is actually assigned.

In conclusion, the proposed demand-assignment procedure takes place according to the following steps (see also Fig. 4 which assumes $N = 2$), hereafter described starting from the generic demand time $\eta$:

1. At the time $\eta$ the STs compute the forecast queue lengths $q_{ij}^*(\eta + T_{inf})$ according to (6), the forecast bit rates $R_{ij}^{in*}[\eta + T_{inf}, \eta + T_{inf} + L]$ according to (7) and the coefficients $\beta_{ij}$ according to (8). All these parameters are sent to the NCC.

2. At the time $\eta + L/2$, the NCC receives the pieces of information mentioned in the previous issue from the STs and, basing on such information, computes, according to (10) and (11), the capacity assignments $\delta_{ij}^{ass}[\eta + L, \eta + L + T_{inf}]$ to be granted to the connections $(i, j)$ during the time interval $[\eta + L, \eta + L + T_{inf}]$. Such assignments are broadcasted to the STs.

3. At the time $\eta + L$, the $i^{th}$ ST receives the capacity assignments $\delta_{ij}^{ass}[\eta + L, \eta + L + T_{inf}]$ granted by the NCC. These assignments determine the amount of packets which the $i^{th}$ ST is authorized to forward from the queues $(i, j)$ towards the uplink air interface during the time interval $[\eta + L, \eta + L + T_{inf}]$. Moreover, the $i^{th}$ ST utilizes these capacity assignments at next demand time(s) for the computation, according to (6), of the forecast queue lengths.
As a final remark, note that the $i^{th}$ ST can rearrange, among the connections $(i, j)$ it is involved in, the capacity granted to it. In this rearrangement the $i^{th}$ ST can take into account updated information concerning the present lengths of the queues $(i, j)$ (this information was not available to the NCC when it computed the capacity assignments), according to appropriate criteria [25][53]. So, at a time $\eta + L$, the $i^{th}$ ST can compute the overall uplink capacity, indicated as $\alpha_i[\eta + L, \eta + L + T_{inf}]$, assigned to it during the time interval $[\eta + L, \eta + L + T_{inf}]$, according to the following equation:

$$\alpha_i[\eta + L, \eta + L + T_{inf}] = \sum_{j=1}^{C(i)} \delta_{ij}^{ass}[\eta + L, \eta + L + T_{inf}]$$ (12)

Therefore, the capacities actually granted to the connections during the time interval $[\eta + L, \eta + L + T_{inf}]$, $\delta_{ij}[\eta + L, \eta + L + T_{inf}]$, can differ from the ones granted by the NCC (i.e., $\delta_{ij}^{ass}[\eta + L, \eta + L + T_{inf}]$). Nevertheless, the following constraint must be respected:

$$\alpha_i[\eta + L, \eta + L + T_{inf}] \geq \sum_{j=1}^{C(i)} \delta_{ij}[\eta + L, \eta + L + T_{inf}]$$ (13)

### 1.5 Markov Modulated Poisson Process based algorithm for Satellite Traffic Prediction

In this section we introduce our Markov Modulated Poisson Process (MMPP) based algorithm for satellite traffic prediction. This algorithm is constituted by two phases: 1) the Identification/Tuning phase, and 2) the Traffic Prediction phase. In the Identification/Tuning phase we find the order of the MMPP chain and the associated parameters according to some aggregated measurements on the behavior of the IP data stream. More precisely, we will assume ergodicity of the traffic stream, and we will exchange Time Averages, which are measured on the traffic stream, for Ensemble Averages, which are then used to configure the MMPP chain. In the Traffic Prediction phase we use the tuned MMPP system chain achieve traffic prediction.

The remain of the section is organized as follows. In Section 1.5.1 we recall the main features of a Modulated Poisson Process chain and present the general problem of data generation in this process. In Section 1.5.2 the Identification/Tuning phase is described and the MMPP matching problem is detailed. Finally, in Section 1.5.3 the Traffic Prediction phase and the entire MMPP flow chart are presented.
1.5.1 Data Generation in a Markov Modulated Poisson Process

A Markov Modulated Poisson Process (MMPP) \([46][45][134]\) of order \(N_M\) consists of a \(N_M\) state Markov chain in which each state \(i, (i = 1, 2, ..., N_M)\), is associated to a Poisson process with rate \((\lambda_M)_i\). A MMPP is a time varying Poisson process, whose rate is changed (modulated) according to the Markov chain. Let us consider in the following a discrete-time Markov chain, as shown in Fig. 5, which is sampled every \(\Delta t\), starting from an initial time \(t_0\). In the following, for sake of simplicity, we will use the index \(k\) to represent the discrete-time instant \(t_k = t_0 + k\Delta t\).

If \(S_k (k = 0, 1, ...)\) is a state moving at discrete-time instants between a finite number of states \(i = 1, 2, ..., N_M\) with transition probabilities that only depend on the previous state, then \(S_k (k = 0, 1, ...)\) is defined as a Markov process. Let the matrix \(P = [p_{ij}] \in \mathbb{R}^{N_M \times N_M}\) be the one-step state transition matrix whose elements are the probabilities to transit from one state into another in one time-step, as shown in Fig. 5. These are formally defined as:

\[
p_{ij} = \text{Prob}\{S_{k+1} = j \mid S_k = i\}, \quad \forall i, j = 1, 2, ..., N_M \tag{14}
\]

This matrix \(P\) can be shown to be a stochastic matrix since it has the property that all its elements are nonnegative \((p_{ij}, \forall i, j = 1, 2, ..., N_M, \text{are probabilities})\) and that the sum of the elements in each row equals 1 (each state has at least one possible transition into another state). These properties can be mathematically expressed as follows:

\[
p_{ij} \in \mathbb{R}^+ \cup \{0\}, \quad \sum_{h=1}^{N_M} p_{ih} = 1 \quad \forall i, j = 1, 2, ..., N_M \tag{15}
\]

Note that \(p_{ij}\) do not depend on time, i.e., on \(k\).

Let \(\Theta_M(k)\), at time \(t_k\), be the column vector with \((\Theta_M)_i(k) = \text{Prob}\{S_k = i\}\) as \(i^{th}\) entry, representing the state distribution of the chain at step \(k\). It holds that:

\[
(\Theta_M)_i(k) \in \mathbb{R}^+ \cup \{0\}, \quad \sum_{i=1}^{N_M} (\Theta_M)_i(k) = 1 \quad \forall k, \forall i = 1, 2, ..., N_M \tag{16}
\]

Under this assumption, the evolution of a MMPP is completely described by the following equation:

\[
\Theta_M(k + 1)^T = \Theta_M(k)^T \cdot P \tag{17}
\]

where \(^T\) is the transpose operator.

Throughout this chapter, we will neglect the transitory state, and we will consider the Markov chain in its steady state, where the state distribution \(\Theta_M\) becomes time-independent,
Markov Modulated Poisson Process (MMPP) State Chain

Figure 5: Markov Modulated Poisson Process (MMPP) State Chain

and does not depend anymore on step \( k \). In the steady state, (17) simplifies to:

\[
\Theta_M^T = \Theta_M^T \cdot \mathcal{P}
\]

It is worth pointing out that the vector \( \Theta_M \) is the left eigenvector of the stochastic matrix \( \mathcal{P} \) corresponding to the eigenvalue 1. This is a nonnegative vector and the sum of its elements equals 1, exactly as in (16), but now without the dependency on \( k \):

\[
(\Theta_M)_i \in \mathbb{R}^+ \cup \{0\}, \quad \sum_{i=1}^{N_M} (\Theta_M)_i = 1 \quad \forall i = 1, 2, \ldots, N_M
\]

Let us introduce the observation or measurement process \( \psi_k \), \( (k = 0, 1, 2, \ldots) \), which accounts for the incoming IP traffic that we want to predict. In the following we will not represent traffic according to the fluid approximation, although fluid models\(^1\) are widely used in addressing a variety of network control problems such as congestion control [114][14], routing [15], and pricing [156][24]. Rather, we will consider a more realistic model where traffic is actually made up of discrete-packets of variable length. In our approach, in order to measure traffic, we logically segment a traffic data stream into fixed-length blocks, whose size \( B \) is a constant predefined number of bits. The block size \( B \) is the finest granularity in the measurement of traffic and can be as small as needed. This approach allows us to deal with traffic composed of variable length IP packets, while keeping the complexity of the predicting algorithm low. This approximation should not been seen as a limiting factor, since we are neither interested in predicting the packet length nor the inter-arrival time between consecutive packets. In fact, as explained in Section 1.4, we need to calculate \( R_{ij}^{(n)}[\eta, h] \), which is the prediction, performed by the \( i^{th} \) ST at time \( \eta \), of the average bit rate which will enter the queue \((i, j)\) during the \( h^{th} \) time interval. Thus, we are interested in predicting the average bit rate during a time interval.

\(^1\)Fluid models replace discrete packets with continuous flows.
If we divide the monitoring interval $T_{\text{mon}} = K \Delta t$ into $K$ sub-intervals each with time-length $\Delta t$, where the maximum number of expected traffic blocks are $L$, then the following relation among $B$, $L$, and $\Delta t$ must hold:

$$\frac{B \cdot L}{\Delta t} \geq R_{\text{MAX}}$$

(20)

where $R_{\text{MAX}}$ is the maximum expected bit rate.

There is a tradeoff between the granularity of the traffic measurement, which depends on $B$, and the complexity of the proposed predicting algorithm, which increases as $L$ increases, as will be clear later on. In particular, the smaller $B$ is, the larger $L$ must be in order to satisfy (20), given a maximum expected bit rate $R_{\text{MAX}}$. As far as the choice of $T_{\text{mon}}$, i.e., $K$, is concerned, the longer the monitoring time is, the more statistically relevant the measures of the traffic are, but the longer it takes to have updated measures. An appropriate value for $T_{\text{mon}}$ should be tuned considering this tradeoff, as well as the expected statistical variability of the traffic.

The observation or measurement process $\psi_k$, $(k = 0, 1, ..)$, accounts for the number of measured blocks arriving in the time interval $[t_k, t_{k+1}]$. It has values in the set $\mathcal{L} : \{0, 1, \ldots, L\}$, and depends probabilistically on $S_k$ via the matrix $C = [c_{l+1j}] \in \mathbb{R}^{(L+1) \times N_M}$, $l = 0, 1, \ldots, L$, $j = 1, 2, \ldots, N_M$, which represents the Poisson distributions:

$$c_{l+1j} = \text{Prob}\{\psi_k = l \mid S_k = j\} = \frac{\left(\lambda_{M}\right)_j^{l} \cdot e^{-(\lambda_{M})_j}}{l!}$$

(21)

where $(\lambda_{M})_j \geq 0$ is the arrival bit rate of the Poisson process associated with the state $j$. If the state parameter $(\lambda_{M})_j$ is 0, then its associated emission bit rate will be zero as well. This deterministic process will be called zero process. The row of $C$ that corresponds to the zero process is equal to $[1 \ 0 \ \cdots \ 0]$. Let $\Xi$ be the vector with $\text{Prob}\{\psi_k = l\}$ as $(l+1)^{th}$ entry, then:

$$\Xi = C \cdot \Theta_M$$

(22)

For sake of compactness, the Poisson parameters $(\lambda_{M})_i$ are arranged in a column vector $\Lambda_M = [(\lambda_{M})_i] \in \mathbb{R}^{N_M}$. At time step $k$, the MMPP, which is described by the tuple $\langle \mathcal{P}; \Lambda_M; N_M \rangle$, will generate $\psi_k$ bit emissions according to the Poisson process with arrival rate $(\lambda_{M})_i$, if the state of the Markov chain is $i$. This implies that if $(\lambda_{M})_i = 0$, no bit emissions will be generated.

In our prediction model we exploit the first order statistics of the bit emission $\psi_k$. The first order statistic is described by the cumulative distribution function, which is defined as $\mathcal{F}(v) = \text{Prob}\{\psi_k \leq v\}$, with $v = 0, 1, \ldots, L$. The MMPP cumulative distribution function $\mathcal{F}_M$
is a weighted average of the cumulative distributions of each Poisson process associated to a state of the chain. Thus:

$$F_{M}(v) = \sum_{i=1}^{N_{M}} (\Theta_{M})_{i} \cdot F_{(\lambda_{M})_{i}}(v)$$  \hspace{1cm} (23)$$

where

$$F_{(\lambda_{M})_{i}}(v) = e^{-(\lambda_{M})_{i} \cdot v} \cdot \sum_{l=0}^{v} \frac{(\lambda_{M})_{i}^{l}}{l!}$$  \hspace{1cm} (24)$$

1.5.2 The Identification/Tuning Phase: the MMPP matching problem

After computing the cumulative distribution function of the arrival process $\psi_{k}$, the vectors of the first order parameters of the MMPP chain, $\Lambda_{M}$, $\Theta_{M}$ and the model order $N_{M}$ are determined by solving a nonnegative least square problem [100]. When all these first order parameters are determined, the identification problem is solved, and traffic prediction can be achieved using the data generation of the determined MMPP chain, as described in Section 1.5.1.

In order to solve the model identification problem, we are concerned with the following problem: Find $N_{M}$, $\Theta_{M}$, and $\Lambda_{M}$, $\forall i = 1, 2, ..., N_{M}$. Given observations on the arrivals $\psi_{k}$ for each time step $k = 0, 1, 2, ..., K - 1$ within the monitoring interval $T_{mon}$, so that they form a MMPP with first order statistics $F_{M}$ matching those of the measured traffic $\psi_{k}$, i.e., $F_{data}$, as accurately as possible. The state parameter vectors $\Theta_{M} = (\Theta_{M})_{i}$ and $\Lambda_{M} = (\Lambda_{M})_{i}$, with $i = 1, 2, ..., N_{M}$, can then be used to predict future traffic according to (23).

The cumulative distribution function of the data sequence is a staircase function. It is computed as follows:

$$F_{data}(v) = \frac{1}{K} \cdot \sum_{j=0}^{v} \sum_{k=0}^{K-1} \delta(\psi_{k}, j), \hspace{0.5cm} v = 0, 1, 2, ..., L$$  \hspace{1cm} (25)$$

where $\delta(\psi_{k}, j)$ is the Kronecker delta and $K$ is the total number of subintervals which the monitoring interval $T_{mon}$ is divided in.

The distribution function of the MMPP is a linear combination of Poisson distributions, as can be inferred from (23). The cumulative distribution function of the data, $F_{data}$, must be approximated by the MMPP cumulative distribution function $F_{M}$. This implies that $F_{data}$ must be approximated by a nonnegative linear combination of cumulative distributions of Poisson processes:

$$F_{d} \simeq D \cdot \Theta_{M}$$  \hspace{1cm} (26)$$
where \( F_d \) is the vector of size \( L + 1 \) with elements \( (F_d)_v = F_{\text{data}}(v) \), with \( v = 0, 1, \ldots, L \), and \( L \) is the maximum number of expected traffic blocks. The columns of the matrix \( D \in \mathbb{R}^{(L+1) \times N_M} \) correspond to the cumulative distribution functions \( F_{(\lambda_M)} \), in (24).

If the states of the MMPP modeling a given data set were given, i.e., the \( (\lambda_M) \) parameters were known, the matrix \( D \) would be determined and \( \Theta_M \) could be easily determined by solving the equations’ system in (26). Without the knowledge of the states or the number of states, which is a more realistic scenario, the same approach yields \( N_M, \Theta_M \) and \( \Lambda_M \) if \( D \) is replaced by an enlarged version \( D' \in \mathbb{R}^{(L+1) \times N'_M} \). This matrix \( D' \) has more columns than \( D \) has \( (N'_M \geq N_M) \), but the same number of rows. Each column \( (D')_j \), with \( j = 1, 2, \ldots, N'_M \), of this enlarged matrix represents a possible state, i.e., a possible Poisson cumulative distributions. In fact, in each subinterval of the monitoring period the domain of possible block arrivals \([0, L]\) is discretized in order to get a broad choice of candidate states. The discretization step is chosen to increase linearly because the variance of a Poisson process is equal to its rate. The computed cumulative distribution vector \( F_d \) must be reconstructed as a nonnegative linear combination of the columns of \( D' \). We can now formulate the optimization problem for the model identification as follows:

\[
\textbf{P} : \text{MMPP Identification Problem}
\]

\begin{align*}
\text{Given} & : \quad \mathcal{L}, N'_M, \psi_k, k = 0, 1, 2, \ldots, K - 1 \\
\text{Find} & : \quad x = (x)_j; \quad j = 1, 2, \ldots, N'_M \\
\text{Min} & : \quad \|F_d - D'x\|_2 \\
\text{Subject to} & : \quad (x)_j \geq 0; \quad j = 1, 2, \ldots, N'_M \\
& \quad (F_d)_v = F_{\text{data}}(v); \quad v = 0, 1, \ldots, L
\end{align*}

We solved this problem using the \textit{Khun Tucker’s algorithm} [126]. The MMPP parameters are completely defined by the solution of this optimization problem, as follows. Let \( x^* \) be the optimal value of \( x \). The number of non-zero components of \( x^* \) gives automatically the MMPP model order \( N_M \leq N'_M \). The indices of the nonzero components of \( x^* \) represent \( (\lambda_M)_j \), whereas the values of the nonzero \( x^* \) components give \( (\Theta_M)_j \). It worth pointing out the model order \( N_M \) is automatically determined.
1.5.3 The Traffic Prediction Phase

Once the identification problem has been solved, and all the parameters have been tuned, which are particular tasks characterizing the Identification/Tuning phase, the Traffic Prediction phase is in charge of predicting the future traffic, according to (23). In Fig. 6, a flow chart of the entire MMPP algorithm is depicted. It is worth pointing out that each traffic prediction value is periodically compared to the real incoming data traffic, so that it is possible to know when it is necessary to trigger a new Tuning phase. To this end, the relative prediction error is constantly monitored in order to prevent prediction error drift. In fact, if the relative error is higher than a predetermined threshold, a new identification problem is solved, exploiting the latest available traffic measurements. This mechanism allows following the dynamics of the input traffic, while saving precious computational resources.

1.6 Simulation Results

The simulation tool OPNET Modeler 7.0 [1] has been adopted for developing a satellite simulator and testing the performance of the proposed algorithms. Figure 7 shows the OPNET simulation scenario, at the network level (a GEO satellite system has been considered for its
challenging high propagation delay), node level (Satellite Terminals, User Terminals, Hub Stations, and Network Control Center), and process level (Finite State Machines developed in C-language which implement the proposed algorithms and the associated protocols). The statistical characteristics of the considered applications are reported in the following tables, and one statistical realization for each considered traffic source type is sketched in Fig. 8. In particular, we considered 5 types of applications: 2 Real Time applications, using UDP, namely Voice over IP and Video-conference, and 3 Non Real Time applications, using TCP, namely FTP, E-mail and Web browsing [40]. The simulated scenario is the one shown in Fig. 2, which considers a GEO satellite system ($L \approx 500 \text{ ms}$). The considered spot-beam includes 3 STs. Each of these STs is connected (e.g., via Ethernet LAN) to 5 User Terminals (UTs). Each of these UTs has 5 connections in progress relevant to the 5 applications listed in the tables, such a way that every UT is involved in 1 Voice over IP, 1 Video-conference, 1 FTP, 1 E-mail and 1 Web-browsing connection. This means that $S = 3$ and $C(i) = 5$ ($i = 1, 2, 3$).

The parameters $R_{ij}^{static}$ (see Section 1.2) are all set to zero for any $i$ and any $j$. For Real Time traffic, following the QoS parameters proposed by the ETSI TIPHON project [5], we have considered the maximum queue delay $D_{ij}^{max} = 50 \text{ ms}$ for both Voice over IP and Video-conference connections (the uplink and downlink GEO propagations alone take about 250 ms).

The available uplink capacity in the considered spot-beam $R_{up}^{tot}$ is equal to 10 Mbit/s.

<table>
<thead>
<tr>
<th>Table 1: E-mail parameter definitions</th>
</tr>
</thead>
<tbody>
<tr>
<td>Send Interarrival Time [s]</td>
</tr>
<tr>
<td>Send Group Size</td>
</tr>
<tr>
<td>Email Size (Kbytes)</td>
</tr>
<tr>
<td>Transfer Protocol</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Table 2: File Transfer parameter definitions</th>
</tr>
</thead>
<tbody>
<tr>
<td>Inter-Request Time [s]</td>
</tr>
<tr>
<td>File Size (Kbytes)</td>
</tr>
<tr>
<td>Transfer Protocol</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Table 3: Web Browsing parameter definitions</th>
</tr>
</thead>
<tbody>
<tr>
<td>HTTP Specification</td>
</tr>
<tr>
<td>Page Interarrival Time [s]</td>
</tr>
<tr>
<td>Transfer Protocol</td>
</tr>
</tbody>
</table>
Table 4: Video Conferencing parameter definitions

<table>
<thead>
<tr>
<th>Parameter Information</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Frame Interarrival Time Information</td>
<td>15 frame/s</td>
</tr>
<tr>
<td>Frame Size Information (Kbytes)</td>
<td>unif_int [2,3]</td>
</tr>
<tr>
<td>Transfer Protocol</td>
<td>UDP</td>
</tr>
</tbody>
</table>

Table 5: Voice over IP parameter definitions

<table>
<thead>
<tr>
<th>Parameter Information</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Talk Spurt Length Outgoing(^2) [s]</td>
<td>exponential (0.352)</td>
</tr>
<tr>
<td>Silence Length Outgoing [s]</td>
<td>exponential (0.65)</td>
</tr>
<tr>
<td>Encoder Scheme</td>
<td>G.711 (Silence)</td>
</tr>
<tr>
<td>Voice Frames per Packet(^3)</td>
<td>1</td>
</tr>
<tr>
<td>Transfer Protocol</td>
<td>UDP</td>
</tr>
</tbody>
</table>

Let us point out that all the 95% confidence intervals are not shown in the figures for sake of clarity.

Figure 9 shows the cumulative distribution of the queuing delay \(D_{ij}\) experienced by the packets relevant to Real Time applications, as the parameter \(N\) varies. More precisely, Fig. 9(a) depicts the queuing delay cumulative distribution for the Voice IP datagrams, while Fig. 9(b) depicts the same metric for the Video IP datagrams. Note that by increasing the value of \(N\), the queuing delays are decreased and, in particular, the probability that the queuing delay exceeds the maximum tolerated delay threshold is also decreased. Nevertheless, we also increase the signaling overhead. So, as already stressed in Section 1.4, \(N\) must be selected by trading off these contrasting requirements. In addition, Fig. 9 roughly shows that \(N = 4\) is the lowest value of \(N\) for which we can generally respect the QoS Delay Requirement for Real Time applications, with an acceptable probability. Thus, this value has been taken in the simulations.

In the following simulations, we compare the performance achieved by using the proposed capacity assignment procedure, i.e., the procedure described in Section 1.4 (this case will be hereinafter referred to as \textit{all-dynamic}) which is deeply based on the Markov Modulated Poisson Process (MMPP) prediction algorithm in Section 1.5, with the one achieved by fixedly partitioning the available bandwidth among the three STs (this case will be hereinafter referred to as \textit{all-static}). Note that in the \textit{all-static} case each ST is fixedly assigned a third of the total available uplink capacity and no demand-assignment mechanism is required. The simulation parameters are the same as in the previous simulations, but the available uplink capacity \(R_{up}^{tot}\). As a matter

\(^1\)Indicates the number of “queued emails” to be sent.
\(^2\)Specifies the time spent by the calling party in speech mode in a speech-silent cycle.
\(^3\)This attribute determines the number of encoded voice frames grouped into a voice packet, before being sent by the application to the lower layers.
OPNET Modeler

1. Network level
2. Node level
3. Process level (Finite State Machine)

Figure 7: OPNET Simulation Scenario
Figure 8: Traffic Source parameters and their statistical behaviour

of fact, we will consider the satellite system behavior when the available uplink capacity $R_{up}^{tot}$ is varied with respect to an *average offered uplink capacity* of 5.5 Mbit/s defined as the ratio between the total traffic (expressed in bits) offered to the three STs by the UTs during the simulation time interval and the simulation time interval duration itself. In particular, we considered three cases in which the available uplink capacity is equal to 80%, 100% and 130% of the average offered uplink capacity, respectively. In these cases, the following key performance parameters (corresponding to the targets stated in Section 1.2) are monitored:

- The *bit loss percentile* of the Real Time traffic, which is defined as the ratio between the bits discarded because they have waited more than the maximum tolerated delay, and the sum of the transmitted and discarded bits;

- The *average throughput* (see the definition in Section 1.2) of the Non Real Time traffic.

Figure 10 graphs the *bit loss percentile* for the Voice over IP application as a function of the above-mentioned capacity values. The dashed line refers to the *all-dynamic* case, while the solid line to the *all-static* one. The figure highlights that, if the satellite system is overloaded, the *all-dynamic* approach is remarkably more efficient than the *all-static* one, since the proposed
Figure 9: Queuing delay Cumulative Distribution Functions (CDFs) of Voice (Fig. 9(a)) and Video (Fig. 9(b)) IP datagrams
Figure 10: Voice bit loss percentile comparisons expressed for significant capacity values

Figure 11: Video bit loss percentile comparisons expressed for significant capacity values
Figure 12: TCP average bit rate comparisons expressed for significant capacity values

demand-assignment procedure succeeds in exploiting the advantages of statistical multiplexing. Clearly, as the capacity availability grows, these advantages reduce and for an high capacity availability, the all-static has a slight advantage over the all-dynamic case, due to the fact that the former does not require the demand-assignment procedure (which entails delay and overhead savings).

Figure 11 is similar to Fig. 10, but refers to the video-conference application. The same considerations as in Fig. 10 apply. The lower bit loss percentile values obtained for video-conference depend on the higher maximum tolerated delay of this application with respect to the voice one.

Finally, Fig. 12 graphs the average throughput for the Non Real Time applications. Once again, the same considerations as in Fig. 10 apply.

1.7 Conclusions

This chapter dealt with the problem of the design of a control predictive based demand-assignment algorithm for a satellite network guaranteeing a target Quality of Service (QoS) to Internet traffic, while efficiently exploiting the air interface. The proposed novel algorithm
is in charge of dynamically partitioning the uplink capacity among the connections in progress in the considered satellite spot-beam. Such a partitioning is performed aiming, on the one hand, to match the QoS requirements of each connection and, on the other hand, to maximize bandwidth exploitation. A Control Theory approach, based on a Markov Modulated Chain Prediction Model for the traffic forecast, is adopted to effectively tackle the problem of the delay between the bandwidth request and bandwidth assignment, and the signaling overhead caused by control messages. The algorithms efficiently cope with both the satellite propagation delay and the delays inherent in the periodic nature of the bandwidth demand-assignment mechanism. The demand-assignment algorithm and the proposed Markov Modulated Chain traffic prediction model are shown to be easy to implement and to improve the overall satellite network performance. Performance evaluation is carried out by using OPNET network simulator.
CHAPTER II

A CONTROL BASED TRAFFIC CONTROLLER PROVIDING QOS GUARANTEES

2.1 Introduction

One of the main challenges of the out coming telecommunication networks is the provision of Quality of Service (QoS) guarantees to the IP (Internet Protocol) flows which, most of times, are currently still served according to a best-effort paradigm which does not assure any QoS guarantee. The respect of the QoS guarantees, already a challenging goal in wired networks, is even more challenging in wireless networks in which this goal has to be achieved in conjunction with an efficient exploitation of the available bandwidth which in wireless networks is a much more valuable resource than in wired ones. As a result, in wireless networks, traffic control strategies can be key factors for assuring QoS guarantees and, at the same time, efficiently exploiting the available bandwidth.

In this respect, this chapter proposes an innovative Traffic Controller which assures an efficient available bandwidth exploitation while allowing the respect of the QoS guarantees. This Traffic Controller has been developed for a satellite environment, in the framework of the SATIP6 (SATellite broadband multimedia system for IPv6) project [3] financed by the European Commission in the context of the 5th framework programme (IST-Information Society and Technology programme). Nevertheless, the proposed Traffic Controller can be profitably used in New Generation IP-based Satellites [89] or in any wireless or wired network.

Figure 13 shows the SATIP6 reference scenario [51] where various IP flows coming from the IP Backbone network arrive at an Hub Station (HS) which is in charge of forwarding them towards the satellite. The goal is to efficiently share the uplink bandwidth (i.e., bit rate) which is pre-assigned to the considered HS among the various incoming IP flows meeting the QoS requirements of these IP flows.

The original adopted approach is based on the presence of a control based Traffic Controller which continuously computes a parameter providing an updated measurement of the overall efficiency experienced at the HS, as well as of the efficiencies experienced by each IP flow.
Then, appropriate closed-loop control actions steer the single IP flow efficiencies to track the overall efficiency experienced at the HS. This approach assures flow fairness and bandwidth efficiency, with fairness being defined as the respect of a bandwidth-sharing rule that accounts for the data flow and time constraints of each connection, which will be explained later on, as well as the satisfaction of the QoS guarantees for every IP flow.

The following of the chapter is organized as follows. Section 2.2 presents the Traffic Controller architecture explaining the functions of the various components of such controller. Section 2.3 focuses on the Main Controller which is the core of the Traffic Controller since it includes the control based algorithm which decides the bit rate assignments for each IP flow. Finally, Section 2.4 shows some meaningful simulation results and Section 5.8 drives some conclusion.

### 2.2 Traffic Controller Architecture

The goal of the Traffic Controller is to share the available uplink bandwidth (i.e., the available uplink bit rate), so as to serve all the accepted IP flows respecting their QoS constraints and complying with the principles of fairness and efficiency.

As far as the QoS constraints is concerned, the approach which is currently the most widely used in literature is to identify a set of QoS Classes [104][34][35]. Many different IP flows can
belong to a same QoS Class [118]. A QoS Class is characterized by a set of QoS guarantees [38],[5] (hereafter, also referred to as QoS contract), namely:

1. A minimum bit rate, hereafter referred to as Compliant Bit Rate, which must be granted to the IP flows belonging to the QoS Class in question in whichever traffic condition;

2. A maximum (hereafter indicated as $D_{max}$) and a minimum (hereafter indicated as $D_{min}$) transfer delay which must be experienced by the IP packets belonging to the IP flows in question.

Note that the constraint on the maximum tolerated transfer delay comes from the obvious requirement of guaranteeing IP packet delivery within given delays, while the constraint on the minimum transfer delay comes from the requirement of limiting the delay jitter (whose maximal value is indicated hereafter as $J_{max}=D_{max}-D_{min}$).

Each IP flow is associated to a QoS Class by setting a proper field of the header of all IP packets belonging to the IP flow in question to the value corresponding to the selected QoS Class. This means that all the IP packets relevant to IP flows belonging to a same QoS Class have the above-mentioned IP packet header field set to the same value.

The internal structure of the HS Traffic Controller is shown in Fig. 14. The IP packets coming from the Internet, i.e., the Offered Traffic, are sorted into a set of FIFO (First in First out) Queues. A one-to-one association between queues and QoS Classes exists: each IP packet is enqueued into the FIFO Queue associated to the QoS Class the IP packet in question belongs to.

The core of the Traffic Controller is the Main Controller which takes the decisions concerning the IP packets to be retrieved from the FIFO (First Input First Output) Queues for being forwarded towards the air interface (i.e., towards the satellite). The Main Controller takes its decisions basing on the following parameters, which are monitored every $T_{control}$ seconds (hereafter referred to as Control Period):

1. The Queue Lengths of the FIFO Queues;

2. The Discarded Bit Rates, i.e., the bit rates of the traffic which has to be ruled out by the various FIFO Queues;

3. The Transmitted Bit Rates, i.e., the bit rates of the traffic which is retrieved from the FIFO Queues for being transmitted towards the satellite through the physical interface.
Periodically, every Control Period, the Main Controller, basing on these three parameters and taking into account the Compliant Bit Rate, computes the Assigned Bit Rates, i.e., the bit rates which can be granted to the various IP flows in the next Control Period. The Assigned Bit Rates are expressed as fractions of the Total HS Bit Rate, i.e., the bit rate which is currently assigned to the considered HS. The Assigned Bit Rates are communicated by the Main Controller to the Scheduler, which is in charge of retrieving from the FIFO Queues the IP packets to be forwarded towards the satellite, trying to comply the bit rate assignments decided by the Main Controller.

Note that the Main Controller assumes a continuous-flow model of the inbound traffic, i.e., it bases its computations on continuous measures of bit rates and bit lengths, overlooking the discrete nature of the IP packet flows, while the Scheduler takes into account the discrete nature of the IP packets. This causes a certain difference between the bit rates actually assigned to the IP flows and the ones decided by the Main Scheduler. In this respect, the Scheduler implements a proper bit credit system [56] which aims at keeping close to zero the error between the bit rates actually assigned to the IP flows and the ones decided by the Main Controller.

It is important remarking that the IP packets which have waited in the FIFO Queues more than their maximum tolerated transfer delay $D_{\text{max}}$ are discarded (i.e., are lost).

### 2.3 Main Controller Algorithm

As stated in the previous section, the control action performed by the Main Controller takes place at the beginning of every Control Period, namely every $T_{\text{control}}$ seconds. Conceptually,
the control action consists of two stages:

1. The assignment of the Compliant Bit Rates,

2. The assignment of the Non Compliant Bit Rates.

In the first stage (detailed in Section 2.3.1) the Main Controller assigns a part of the Total HS Bit Rate to the IP flows according to their Compliant Bit Rates; then, in the second stage, (detailed in Section B) the Main Controller assigns the remaining part of the Total HS Bit Rate according to the algorithm described in the following. The sum of the two above-mentioned assigned values for the i-th IP flow will be hereafter referred to as AssignedR_i, whose value will be used throughout the whole following Control Period.

In the following, the various parameters of interest will be referred to the i-th flow and to the Control Period and will be indicated with self-explaining name, as shown in Tab. 6.

### 2.3.1 First Stage: Compliant Bit Rate Assignment

In the first stage of the control action the Main Controller checks the OfferedRate_i of the i-th IP flow (i.e., the average bit rate offered by the IP flow in the last Control Period) and compares this value with the its Compliant Bit Rate, granting to the i-th IP flow a ComplAssignRate_i
equal to the lower of the two values (so as to comply with the QoS contract, while avoiding waste of bandwidth).

The rationale behind this choice is to assign output bit rate on the base of the input (offered) bit rate. Then, the Main Controller computes the \textit{SpareHSRate}, which is equal to the \textit{TotalHSRate} minus the sum of all the \textit{Compliant AssignedRate}_i; note that this value is always positive since the sum of the \textit{Compliant Bit Rates} is always less than the \textit{TotalHSRate}, as guaranteed by a proper IP flow admission control mechanism [52][53][54].

In the second stage, the \textit{SpareHSRate} has to be partitioned into the \textit{Non Compliant Assigned Rate}_i according to the algorithm detailed in the following section. Clearly, for every IP flow \textit{i} the \textit{Assigned Rate}_i is equal to the sum of the \textit{Compliant Assigned Rate}_i plus the \textit{Non Compliant Assigned Rate}_i.

\subsection{Second Stage: Non Compliant Bit Rate Assignment}

This second stage is the core of the control action and the main novelty of this work. The approach used by the Main Controller in order to assign the \textit{SpareHSRate} to the IP flows is based on the concept of comparing the status of each IP flow with the overall HS status and of increasing or decreasing the assignment aiming at assuring an uniform handling of every connection. Figure 15 shows the overall control scheme this second stage is based on, under the assumption that the considered HS is handling \textit{N} IP flows.

For the sake of clarity, the algorithm behind this control scheme is split in three steps detailed in Sections 2.3.2.1, 2.3.2.2, and 2.3.2.3, respectively.

\subsubsection{Computation of Priority and Efficiency of the single IP flows}

The first step of the algorithm is the definition of a parameter, indicated as \textit{Priority (Pr}_i\textit{)}, which takes into account the current “need for bandwidth” of every \textit{i}-th connection

\[ Pr_i = \frac{QLen_i}{T_{control}} + TransmR_i + LossR_i \]  \hspace{1cm} (27)

Note that if both members of Equation 27 were multiplied by \textit{T_{control}}, the left part of the equation \textit{(Pr}_i \cdot T_{control}) would be equal to the sum of the current number of bits in the queue associated to the \textit{i}-th IP flow \textit{(QLen}_i\textit{)}, plus the number of bits that have been transmitted \textit{(TransmR}_i \cdot T_{control}) and the number of those that have been discarded \textit{(LossR}_i \cdot T_{control}) in the \textit{Control Period}. In other words, \textit{Pr}_i \cdot T_{control} represents the total number of bits that would
be stored in an ideal queue, if all the arrived bits were enqueued doing neither transmissions nor discards for all the length of the Control Period.

Then, we introduce the parameter $\text{FlowEfficiency}_i$ as:

$$
\text{FlowEff}_i = \frac{\text{AsR}_i}{\text{Priority}_i} 
$$

High values of $\text{FlowEff}_i$ mean that the bit rate assigned to the i-th IP flow is consistent with its actual need for bandwidth, expressed by $\text{Priority}_i$ and vice versa. This ratio is performed by the component named “Normalizer” in Fig. 15. In fact it normalizes the bit rate assigned to an IP flow taking into account its Priority.

### 2.3.2.2 Computation of Priority and Efficiency of the overall HS

In this second step of the algorithm we extend the approach behind the parameters $\text{Priority}_i$ and $\text{FlowEff}_i$ to the overall HS, in order to achieve overall HS parameters which can be compared with the ones associated to the single IP flows. To do this, we introduce two other metrics, the
**HSPriority** and the \( HSEff \), defined as follows:

\[
HSPr = \sum_{i=1}^{N} Pr_i 
\]

\[
HSEff = \frac{TotalHSR}{HSPr} \tag{30}
\]

All the computations relevant to the first and the second steps are performed by the component named “Averager” appearing in Fig. 15.

### 2.3.2.3 Closed-loop computation of the \( AssignedRate_i \)

The basic idea behind our Controller is to compute the efficiency of each IP flow and to compare it with the HS overall efficiency: the difference is used to assess whether to increase or to decrease the bandwidth assigned to the i-th IP flow.

Referring to the i-th IP flow, the algorithm computes, in a closed-loop fashion, the \( AssignedRate_i \) (i.e., the total bit rate that the i-th IP flow will be assigned in the next Control Period) with the following five step procedure:

1. \( HSEff \) and \( Flow_iEff \) are compared in order to assess if the i-th IP flow is under-served or over-served. The difference between the two above-mentioned parameters is the so-called \( \Delta_iEff \): positive values of this parameter mean that the i-th IP flow is under-served so that its efficiency must be increased, and vice versa;

2. \( \Delta_iEff \) is multiplied by \( Priority_i \) to obtain \( \Delta Rate_i \), which represents the bit rate assignment variation which will align the i-th IP flow efficiency with the overall HS efficiency;

3. \( \Delta Rate_i \) is fed to a PID (Proportional-Integral-Derivative) Controller whose parameters are controlled by a Fuzzy Tuner (see Figure 16), as detailed in section 2.3.2.4. The result is \( \Delta AssignedRate_i \);

4. \( \Delta AssignedRate_i \) is summed to the \( NonCompliantAssignedRate_i \) computed in the previous Control Period;

5. After a precautional non-linear resharing aiming at avoiding negative values of the \( Non Compliant Assigned Rate_i \) (that could decrease the \( Compliant Assigned Rate_i \) and cause the infringement of the QoS contract relevant to the i-th IP flow), the \( Non Compliant Assigned Rate_i \) is summed to the \( Compliant Assigned Rate_i \), thus obtaining the \( Assigned Rate_i \).
2.3.2.4 PID control and Fuzzy tuning

Figure 16 shows the selected PID Controller whose parameters are controlled by a Fuzzy Tuner. This element implements three different sets of rules, relevant to Proportional, Integral and Derivative Actions, as detailed in the following.

*Proportional Action*: the first set of rules concerns the tuning of the PID Controller aggressiveness which is regulated by the $K_P$ parameter. The various IP flows strive to obtain a common and limited bandwidth: when there are many flows with respect to the available bandwidth many of them may be under-served; this would cause high requests of bandwidth from most of the current active flows, resulting in the risk of oscillation and instability. The system manages to reach stability by reducing the aggressiveness of the PID Controller.

In order to define *Congestion* we introduce the following two parameters: *Need* and *Surplus*. The former represents the cumulative request for bandwidth of all the flows that are under-served while the latter represents the cumulative bandwidth of the flows that are over-served, i.e.,:

\[
Need = \sum_{i=1}^{N} \max[\Delta_{AssignRate_i}; 0]
\]  
(31)

\[
Surplus = \sum_{i=1}^{N} \max[-\Delta_{AssignRate_i}; 0]
\]  
(32)

By adopting the two above-mentioned metrics, i.e., (31) and (32), *Congestion*, a new metric to evaluate the traffic load, can be defined as follows:

\[
Congestion = \frac{Need}{Need + Surplus}
\]  
(33)

In the following we exploit the fuzzy logic in order to control the PID parameters. The basic

\[\text{What is striking is that its most important and visible application today is in a realm not anticipated when fuzzy logic was conceived, namely, the realm of fuzzy-logic-based process control}[159].\]
idea of “Fuzzy Logic Control” (FLC) was suggested in [157][158]. FLC provides a nonanalytic alternative to the classical analytic control theory. By utilizing simple triangular fuzzy sets [88] and the Sugeno deduction [145], we implemented three rules in order to decrease $K_i^P$ as Congestion increases, and to decrease $K_i^I$ and $K_i^D$ as $\Delta_i^{Eff}$ decreases. This rule prevents newly established IP flows from pulling pre-existing IP flows away from the equilibrium - as these flows have null assigned bit rate, their priority is very high at the moment of inception and may tend to overcome other requests. This phenomenon is not usually present for bursty sources, as they have some cumulated assigned bit rate.

\textit{Integrative and Derivative Actions}: the second and third fuzzy rules control the parameters $K_i^I$ and $K_i^D$ according to the usual principles of PID tuning [17][18][19]. Note that the tuning of these last two parameters is tailored on each IP flow.

As far as the \textit{Integrative Action} is concerned, it is used to speed up the control when the system is distant from equilibrium, and to turn off it when the equilibrium is approached. In particular, this action is useful in the transients when the FIFO Queues fill up: the approximately linear increase of the queue length - bit rates from bursty sources (FTP, Web-browsing) can be considered almost constant during a burst, and bit rates from stable sources (Voice, Video) have a little variance around their mean value and can be considered, again, almost constant - causes a decrease in the Priority$_i$ of the i-th flow that is almost linear (in a Taylor series), properly controlled by an integrator component.

As far as the \textit{Derivative Action} is concerned, our simulations prove that it introduces minor benefits to the overall system performance.

### 2.4 Simulation Results

The overall control scheme has been simulated by using the discrete event OPNET [1] simulator by MIL3. We have compared its performance with the one of some popular scheduling schemes. We have considered four QoS Classes: Voice, FTP, Video, Web-browsing.

Figure 17 graphs, on a ten minute time-span which allows to have small 95% confidential intervals, the Offered Bit Rates of one IP flow for each of the four considered QoS Classes. The statistical parameters characterizing these IP flows are reported in Tab. 7 while in Tab. 8 the average and peak rate for each traffic application are shown.

Table 9 reports the parameters characterizing the QoS contract of the four considered QoS
Table 7: IP traffic source statistical parameters

<table>
<thead>
<tr>
<th></th>
<th>Datagram length (bits)</th>
<th>Interval times (ms)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Voice</td>
<td>Const. 580</td>
<td>Const. 20</td>
</tr>
<tr>
<td>FTP</td>
<td>Unif. [320,39680]</td>
<td>Unif. [50,150]</td>
</tr>
<tr>
<td>Web</td>
<td>Unif. [320,23680]</td>
<td>Unif. [50,550]</td>
</tr>
<tr>
<td>Video</td>
<td>Unif. [320,23680]</td>
<td>Gauss. avg.=8.1, σ=4.5</td>
</tr>
</tbody>
</table>

Table 8: Average and peak rates of traffic sources

<table>
<thead>
<tr>
<th></th>
<th>R_{avg} (Kbps)</th>
<th>R_{max} (Kbps)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Voice</td>
<td>29</td>
<td>29</td>
</tr>
<tr>
<td>FTP</td>
<td>200</td>
<td>400</td>
</tr>
<tr>
<td>Web</td>
<td>40</td>
<td>240</td>
</tr>
<tr>
<td>Video</td>
<td>1350</td>
<td>1460</td>
</tr>
</tbody>
</table>

The performance achieved by the proposed Competitive Access Controller, are compared with the one of a FIFO Controller and of an Open-Loop Controller detailed in the following.

In the FIFO Controller only one queue is considered: all the IP packets are enqueued in this queue and are served according to a First In First Out (FIFO) discipline without taking into account the QoS requirements of the IP flows.

In the Open-Loop Controller [128] the incoming IP traffic is sorted, according to its QoS Class, to a set of DLBs (Dual Leaky Buckets) [38] which are used to identify the Compliant traffic which is served, on a first priority basis, according to an Earliest Deadline First (EDF) algorithm [108]; the Non Compliant traffic is served, on a second priority basis, again according to an EDF algorithm. The name of the scheduling controller is due to the fact that DLB parameters are statically set without any closed loop mechanism. Open loop refers to the fact that once schedules are created they are not “adjusted” based on continuous feedback. While open-loop scheduling algorithms can perform well in static or dynamic systems in which the workloads can be accurately modeled [78], they generally perform poorly in unpredictable dynamic systems, as it is shown in this section.

Table 9: IP packet QoS parameters

<table>
<thead>
<tr>
<th></th>
<th>D_{max} (s)</th>
<th>D_{min} (s)</th>
<th>J_{max} (s)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Voice</td>
<td>0.1</td>
<td>0.01</td>
<td>0.09</td>
</tr>
<tr>
<td>FTP</td>
<td>4</td>
<td>0.2</td>
<td>3.8</td>
</tr>
<tr>
<td>Web</td>
<td>1.5</td>
<td>0.05</td>
<td>1.45</td>
</tr>
<tr>
<td>Video</td>
<td>0.5</td>
<td>0.02</td>
<td>0.48</td>
</tr>
</tbody>
</table>
Figure 17: Offered Traffic

Figure 18: Link Efficiency vs System Congestion
Figure 18 graphs the Link Efficiency (i.e., the link exploitation) of the considered controllers as a function of System Congestion, this metric being defined as the ratio between the Total HS Bit Rate and the average bit rates of the overall offered traffic. For the Competitive Access Controller, the QoS constraints are never infringed and the IP packet loss is close to zero even for a System Congestion close to 98%. As can be seen in Figure 18, the Competitive Access Controller outperforms the other controllers (in each load condition) and manages to obtain near ideal performance with a Link Efficiency which closely follows the System Congestion.

2.5 Conclusions

The Traffic Controller exposed in this chapter manages to outperform other considered bandwidth sharing algorithms mainly because of three factors:

1. The definition of a proper band-sharing mathematical schematization, that truly highlights the drivers of need for bandwidth in wireless connections;

2. The utilization of a closed-loop controller that results in a high degree of adaptivity and in some degree of forecasting;

3. The implementation of a smart scheduler that properly realizes the calculated band-sharing on the IP packets.

As a result, the overall control system scheme is able to react to different kinds of simulation scenarios and different traffic load conditions, with varying numbers of traffic flows of different nature, minimizing the loss bit rate and maximizing the efficiency of exploitation of the wireless channel bandwidth.
CHAPTER III

DIFFSERV-INTEGRATED ALGORITHMS FOR RESOURCE OPTIMIZATION IN SOURCE SPECIFIC AND GROUP SHARED MULTICAST APPLICATIONS IN INTERNET

3.1 Introduction

In unicast transmissions, the sender transmits data to a single receiver and, if multiple receivers want to receive the same data content, the sender has to transmit multiple copies of data. In multicast transmission, on the other hand, the sender transmits only one copy of data that is delivered to multiple receivers. One of the most challenging objectives in multicasting is to minimize the amount of network resources utilized to compute and setup multicast trees [135][120]. In multicast communication, the routing problem is to find the minimum-weight tree that spans all the nodes in the multicast group [44][154][147]. Multicast communication can be classified into two types:

- Source specific,
- Group shared.

In source specific multicast communication, only one node in the multicast group sends data while all the other member nodes receive data. In group shared multicast communication, each node in the multicast group wants to send/receive data to/from member nodes. A tree that spans all member nodes is called multicast tree. Multicast trees can be classified into two types:

- Source rooted trees,
- Shared trees,

based on the communication strategy. A source rooted tree has the source node as root and is optimized for source specific multicast communications. A shared tree, on the other hand, is optimized for group shared communications, and connects each group member with all the other group members.
The classical optimization problem in multicast routing is the *Steiner tree problem in networks* (SPN) [9] whose objective is to find the least-cost tree connecting the source and the group of destinations with the minimum total cost over all links. If each destination has a bandwidth requirement, then the problem is to find the *least-cost* tree that respects the bandwidth requirements on each path from the source to the receiver. It can be shown that both these problems are at least as complex as the Geometric Connected Dominating Set problem, which is proven to be NP-complete [66][103], and that a polynomial time certificate for the optimal solution exists. Hence, both these problems are NP-complete, and efficient algorithms to solve these problems in polynomial time attain only approximate solutions [147][9].

In source specific communication, multicast sessions may have a large number of receivers with heterogeneous reception capacities. To accommodate this heterogeneity, a *layering scheme* can be used. In the layering scheme, data transmission through the network takes place over *logical channels*. A sender can simultaneously transmit data on multiple channels and a receiver can receive data from multiple channels. Each channel has its own transmission rate, usually computed at the sender side. Receivers subscribe to the layers cumulatively, receiving data at a cumulative rate. Two different layering schemes for data bulk distribution have been proposed in [32][33]. In the former scheme, channel rates are determined so as to minimize the total *completion time*. The completion time is the time length that a receiver takes to download the file. The total completion time is defined as the sum of all receivers’ completion time. In the second layering scheme [33], channel rates are chosen so as to allow dynamic multicast groups, meaning that a receiver can join the group and start to download the file at any time. This is possible since the transmission on each channel is cyclic. Cumulative rates in the second layering scheme match requested rates worse than in the first layering scheme. On the other hand, the second layering scheme addresses the problem of asynchronous multicast group joint, although with a high overhead.

In group shared communication, a possible approach to build a shared tree is proposed in the Core Based Protocol (CBT) [21][22] and in its enhanced version (CBTv2) [20]. The CBT protocol connects each member node to a *core node* using bi-directional shortest paths. The main drawback is the traffic concentration occurring in the core node. In fact, all data sent to a member node from all the other member nodes pass through the same path from the core node to the destination member node. Thus, some links might be overloaded, specially in multicast
applications with high bandwidth requirements. To avoid traffic concentration, a shared tree can be computed by finding as many source rooted trees as the number of member nodes, each of them with all the other member nodes as leaves. The FTM (Feasible solutions using adapted TM algorithm) algorithm [151][107] follows this approach. The source rooted tree connects the root node with the member node by using the path with the greatest capacity. The bottleneck of a path is characterized as the link with the minimum available bandwidth, and the path bandwidth capacity is defined as the available bandwidth of its bottleneck. The path with the greatest bandwidth capacity is called the *widest path*. The FTM algorithm uses information obtained by the Breadth First Search procedure (BFS) [65]. For each member node, the BFS procedure builds a tree rooted at the member node and connecting each network node with this member node by exploiting the *widest path paradigm criterion*.

In this work we present two novel bandwidth efficient algorithms to build multicast trees, and a novel low computational cost multirate layering algorithm to effectively accommodate heterogeneous receiver requested rates.

The first proposed algorithm is called *DiffServ-Integrated Multicast algorithm for Internet Resource Optimization* (DIMRO), and builds source rooted multicast trees for source specific applications. DIMRO takes into account the network link available bandwidth in order to avoid bandwidth bottlenecks in the network. The idea is to keep the average link traffic utilization low by fairly distributing data flows among those least loaded links. To support this algorithm, a novel low computational cost layering algorithm to accommodate heterogeneous receiver requested bit rates is integrated into DIMRO, which, in this way, results in an innovative native multirate multicast algorithm for source specific applications.

The objective function defined in the layering scheme [32], aiming to minimize the total *completion time*, does not meet properties of *fairness functions* defined in [92]. For this reason, we determine channel rates by minimizing the average ratio between requested rate and cumulative rate subscribed by a receiver. This way, our objective function meets properties of *fairness functions*, since each receiver is granted a bit rate whose “distance” to its requested ideal rate is proportional to the ideal rate itself.

The second proposed algorithm is called *DiffServ-Integrated Multicast algorithm for Internet Resources Optimization in Group Shared applications* (DIMRO-GS), and constructs the shared tree for group shared applications. DIMRO-GS builds the shared tree by connecting
each member node to all the other member nodes with a source rooted tree computed using the
DIMRO algorithm.

Both DIMRO and DIMRO-GS algorithms are integrated with the DiffServ Quality of Ser-
vice (QoS) approach [34][35]. The QoS service requested by receivers, mapped into the proper
DiffServ class [81][90], is taken into account in the multicast tree computation. Both algorithms
allow members with less stringent QoS requirements to reuse resources already exploited by
members with more stringent QoS requirements. The results are a better leverage of the net-
work bandwidth resources and an improved QoS perceived by multicast group members. To
summarize, the main innovative features in this work are:

- Seamless integration of DiffServ Quality of Service approach in the proposed multicast
  algorithms;
- Introduction of native multirate algorithms for multicast tree computation;
- Leverage of the network bandwidth resources and an improved QoS perceived by multi-
  cast group members;
- Low computational complexity which effectively leads to time and resource saving.

The remainder of the chapter is organized as follows. In Section 3.2 we describe an original
algorithm for the exact determination, in polynomial time, of logical channel rates. Then, we
present DIMRO, the first multicast algorithm of the chapter for source specific applications in
both non-QoS-aware and QoS-aware networks. In Section 3.3, we tackle the problem of finding
a group shared tree for group shared applications by introducing DIMRO-GS, the second multi-
cast algorithm of the chapter, in both its non-QoS-aware and QoS-aware version. In Section 3.4,
we show numerical results through extensive simulations on an ad-hoc C++ simulator. Finally,
in Section 3.5 we conclude the chapter.

3.2 DIMRO - DiffServ-Integrated Multicast algorithm for Internet
Resource Optimization

In this section we present DIMRO, a novel algorithm to efficiently build source rooted trees
for source specific applications such as live video distribution, software and file distribution,
replicated database server, web site replication, periodic data delivery as sport scores, magazine,
newspaper, etc. In particular, in Section 3.2.1 we present a polynomial time algorithm for
the exact determination of channel rates in a multirate multicast aimed to accommodate heterogeneous receiver requested bit rates. In Section 3.2.2 we explain how classical Steiner based approaches for least cost multicast tree set-up do not tackle the problem of bandwidth bottleneck occurrence. In Section 3.2.3 we describe how DIMRO works in non-QoS-aware networks and show how this algorithm manages to keep low the probability of bottleneck occurrence. Finally, in Section 3.2.4 we point out the main improvements in DIMRO when a QoS-aware network is considered. In this last case the full set of features, which DIMRO is endowed with, is described and analyzed.

### 3.2.1 A polynomial time algorithm for the exact determination of channel rates in a multirate multicast

In source specific communications, multicast sessions may have a large number of receivers with heterogeneous reception capacities. To accommodate this heterogeneity, a *layering scheme* can be used. In a layering scheme, data transmission through the network takes place over *logical channels*. A sender can simultaneously transmit data on multiple channels and a receiver can receive data from multiple channels.

Let us consider a network modelled as a directed graph $G(V, E)$, where $V$ is the set of nodes and $E$ is the set of arcs. The link connecting node $u$ to node $v$ is represented by an ordered couple $(u, v)$. Each node in the directed graph represents a router in the network, while each arc represents a communication link. Communication links in a network may have different properties. A property of a communication link $(u, v)$ can be represented by a weight $c_{uv}$ of the corresponding arc in the graph [9]. In a multirate multicast scenario, data on channel $n$ are transmitted at rate $w_n$. Receivers subscribe to the layers cumulatively, i.e., if a receiver subscribes to layer $j$, it also subscribes to layers $\{1, 2, \ldots, j - 1\}$ and receives data at the cumulative rate $L_j$ shown hereafter:

$$L_j = \sum_{n=1}^{j} w_n$$  \hspace{1cm} (34)

In [32], channel rates are determined so as to minimize the total *completion time*, defined as the sum of all receivers’ completion time, which is the time that a receiver takes to download the file. With $M$ receivers and $K$ channels the algorithm has a computational complexity $O(M^3 \cdot K)$. 

59
In the layering scheme in [33], channel rates meet the following relationship:

\[
w_j = \begin{cases} 
  b & \text{if } j = 1 \\
  \sum_{t=1}^{j-1} w_t & \text{if } j > 1 
\end{cases}
\]  

(35)

where \( b \) is a base rate. Cumulative rates computed by the layering scheme in [33] match requested rates worse than those calculated by the layering scheme in [32]. On the other hand, the layering scheme in [33] allows dynamic multicast groups, meaning that a receiver can join the group and start to download the file at any time. This is possible since the transmission on each channel is cyclic, and a cyclic transmission on each channel is possible because channel rates are given as (35). The objective function of the layering scheme in [33] does not meet properties of fairness functions defined in [92]. For this reason, differently from [33], we determine channel rates by minimizing the average ratio between requested rate and cumulative rate subscribed by a receiver. This way, our objective function meets properties of fairness functions, as it will be clear in the following. We present hereafter an efficient algorithm to solve this problem in polynomial time. This novel layering scheme is deeply exploited by the DIMRO algorithm which works in a multirate multicast environment, as demonstrated in what follows.

In a layering scheme a file is divided into segments. Copies of the same segment are transmitted through different channels [32][33]. The number of used channels, indicated as \( K \) hereafter, is a critical parameter that should be chosen by trading off different contrasting issues, briefly summarized in the following. By increasing the number \( K \) of channels, the number of file segments grows. The more file segments, the more information is needed to rebuild the received file. Furthermore, the more channels, the more copies of the same segment are transmitted through different channels. This implies that, by increasing the number of channels, the overhead increases. For applications with a large number of receivers and high variability in rates, it is unrealistic to provide enough channels to match requested rates of all receivers. The optimal number \( K^* \) of channels should be determined to achieve as much benefit for the receivers as possible, without incurring in excessive overhead.

Let us consider a set of \( M \) receivers with \( N \leq M \) different requested bit rates \( W_1 < \ldots < W_N \). Let us assume, for the reasons explained above, that the sender can setup a maximum number of logical channels, \( K^{\max} \), according to its computation constraints. So, the number of channels \( K \) the sender will use is:

\[
K = \min(K^{\max}, N)
\]  

(36)
If the number $K$ of logical channels is equal to the number $N$ of different requested rates (from (36) it must result $K_{\text{max}} \geq N$), then channel rates $(w_1, \ldots, w_K)$ are trivially chosen as:

$$w_i = W_i - W_{i-1}, \ i \in [1, N], \ W_0 = 0 \quad (37)$$

If a receiver with requested rate $W_j$ subscribes the set of channels $\{1, 2, \ldots, j\}$, then from (34) it will receive data at the cumulative rate $L_j$ as follows:

$$L_j = \sum_{n=1}^{j} w_n = W_j \quad (38)$$

that exactly represents the requested rate at which it would receive data. If the number $K$ of channels is lower than the number $N$ of requested rates, as it is the case in a realistic scenario, then the cumulative rates $L_1, \ldots, L_K$ can be determined by solving the optimization problem $(P_1)$, whose objective is to realize fairness among all receivers.

If we assume that each receiver is assigned a weight $\gamma_m, m \in [1, M]$, then $\eta_1, \ldots, \eta_N$ in (41) are $N$ weights such that $\eta_n, n \in [1, N]$, is the sum of the weights of those receivers asking for rate $W_n$. If, for sake of simplicity, we assign to all receivers the same weight, equal to 1, i.e., $\gamma_m = 1 \ \forall m \in [1, M]$, then $\eta_n$ is the number of receivers asking for rate $W_n$.

**$P_1$: Optimal fair logical channel rate determination**

Given : $K, N, W_n \ \forall n \in [1, N]$ \quad (39)

Find : $L_k, y_{n,k} \ \forall k \in [1, K], \ \forall n \in [1, N]$ \quad (40)

Max : $\phi(L_1, \ldots, L_K) = \sum_{n=1}^{N} \frac{\eta_n}{W_n} (\sum_{k=1}^{K} L_k \cdot y_{n,k}) \quad (41)$

Subject to : $L_k \in \{W_1, \ldots, W_N\}, \ \forall k \in [1, K]$ \quad (42)

$y_{n,k} \in \{0, 1\}, \ \forall k \in [1, K], \ \forall n \in [1, N]$ \quad (43)

$\sum_{k=1}^{K} y_{n,k} = 1, \ \forall n \in [1, N]$ \quad (44)

$\sum_{k=1}^{K} L_k \cdot y_{n,k} \leq W_n, \ \forall n \in [1, N]$ \quad (45)

Let us define $F_n$,

$$F_n = \sum_{k=1}^{K} L_k \cdot y_{n,k}, \ \forall n \in [1, N] \quad (46)$$

indicating the cumulative rate obtained by a receiver with requested rate $W_n$. Constraints (43) and (44) imply that

$$F_n \in \{L_1, \ldots, L_K\}, \ \forall n \in [1, N] \quad (47)$$
Once the cumulative rates $F_n$ are determined by solving the problem $(P_1)$, then the corresponding channels rates are easily computed as follows:

$$w_j = L_j - L_{j-1}, \ j \in [1, K], \ L_0 = 0$$ (48)

Now we present an original algorithm to exactly solve the problem $(P_1)$ in polynomial time. An example of the algorithm in a simplified scenario will be given after its formal description (Fig. 19-23). The algorithm goes through two stages:

1. We build a directed graph $G(V, E)$, where nodes of the graph are the $N$ different requested rates together with an exit node $n_{exit}$, i.e., $V = \{W_1, \ldots, W_N\} \cup \{n_{exit}\}$, thus $|V| = N + 1$,

2. We solve a maximum cost path problem from node $W_1$ to node $n_{exit}$ in the constructed graph $G(V, E)$. At this point, all the nodes belonging to this path will be those cumulative channel rates $L_1, \ldots, L_K$ that form an optimal solution of problem $(P_1)$.

Let us now assume $L_1 = \min\{L_1, \ldots, L_K\} = W_1$. If $L_1 > W_1$, since $L_K > \ldots > L_1 > W_1$ from (48), then receivers that asked for rate $W_1$ could not subscribe any channel without incurring in data loss. In fact, if they subscribed channel $w_1 = L_1 > W_1$, they would be overwhelmed by arriving packets. Thus, this means that it must necessarily be $L_1 = W_1$, as initially assumed.

The proposed algorithm to exactly solve $P_1$, proceeds into steps, hereafter described:

- **Step 1:**

  Starting from node (rate) $W_1$, a link $(1, j)$ from $W_1$ to $W_j$ ($\forall j = 2, \ldots, K + 1$) is added to the graph. Since there are $K$ channels, the set of all possible values for $L_2$ is $\{W_2, \ldots, W_{K+1}\}$, thus at least $K - 2$ values $\{W_{K+2}, \ldots, W_N\}$ remain for the others $K - 2$ cumulative rates $L_3, \ldots, L_K$.

  The cost $c(1, j)$ of each link $(1, j)$ ($\forall j = 2, \ldots, K + 1$) is the partial value of the objective function (41) determined by the choice $L_1 = W_1$ for all those receivers that ask for rates from $W_1$ to $W_{j-1}$. Thus, the cost of link $(1, j)$ is defined as:

  $$c(1, j) = \sum_{n=1}^{j-1} \eta_n \cdot \frac{W_1}{W_n}$$ (49)

- **Step i, $i=2, \ldots, K-1$:**
A link \((i, j)\) from \(W_i\) to \(W_j\) \((\forall j = i + 1, \ldots, K + i)\) is added to the graph. The set \(\{W_{i+1}, \ldots, W_{K+i}\}\) is determined by considering that at this step, values of the cumulative rates, \(L_1 = W_1, \ldots, L_i = W_i\), could have already been assigned (it is only an admissible solution, not necessarily the optimal one). In this limit case, at least \(K-i-1\) values \(\{W_{K+i+1}, \ldots, W_N\}\) must remain for the others \(K-i-1\) values of the cumulative rate \(L_{i+2}, \ldots, L_K\).

The cost \(c(i, j)\) of the link \((i, j)\) is defined as:

\[
c(i, j) = \sum_{n=i}^{j-1} \eta_n \cdot \frac{W_i}{W_n} \tag{50}
\]

The cost \(c(i, j)\) is the partial value of the objective function (41) determined by the choice of the cumulative rate \(W_i\) for all those receivers that ask for rates from \(W_i\) to \(W_{j-1}\).

- **Step \(i, i = K, \ldots, N - 1\):**

A link \((i, j)\) from \(W_i\) to \(W_j\) \((\forall j = i + 1, \ldots, N)\) is added to the graph. The cost \(c(i, j)\) of link \((i, j)\) is defined according to (50). Furthermore, a link from node (rate) \(W_i\) to the exit node \(n_{\text{exit}}\) is added to the graph. In fact at Step \(i, i = K\), all \(K\) values of the cumulative rates \(L_1, \ldots, L_K\) could have already been assigned. If there are no more cumulative rates, i.e., no more channels, then the cumulative rate \(L_K = W_i\) is assigned to all the receivers that ask for rates from \(W_i\) to \(W_N\).

The cost \(c(i, n_{\text{exit}})\) of the link \((i, n_{\text{exit}})\) is defined as:

\[
c(i, n_{\text{exit}}) = \sum_{n=i}^{N} \eta_n \cdot \frac{W_i}{W_n} \tag{51}
\]

Eq. (51) represents the partial value of the objective function determined by the choice of the cumulative rate \(W_i\) for all the receivers that ask for rates form \(W_i\) to \(W_N\).

- **Step \(N\):**

Finally, only one link \((N, n_{\text{exit}})\) from node (rate) \(W_N\) to the exit node \(n_{\text{exit}}\) is added to the graph. Cost \(c(N, n_{\text{exit}})\) of the link \((N, n_{\text{exit}})\) is simply \(c(N, n_{\text{exit}}) = \eta_N\), which is the contribution to the objective function of the choice \(L_K = W_N\).

**Example.** Let us consider \(N = 5\) different requested rates \(W_1 < W_2 < \ldots < W_5\) and let us suppose that only \(K = 3\) channels are available. Let us suppose, for sake of clearness, that \(\eta_n = 1, n = 1, \ldots, 5\).
Figure 19: Determining channel rates: Step 1

Figure 20: Determining channel rates: Step 2

Figure 21: Determining channel rates: Step 3
**Step 1:**

$L_1$ is set to $W_1$ and links from node $W_1$ to nodes $W_2$, $W_3$ and $W_4$ are added to the graph (Fig. 19). There is no link from $W_1$ to $W_5$ because the possible cumulative rates are three ($K = 3$) and it cannot be $L_2 = W_5$. As exposed before, the cost of each link $(W_1, W_j)$ is the contribution of the choice $L_2 = W_j$ to the objective function (41).

**Step 2:**

Links from node $W_2$ to nodes $W_3$, $W_4$ and $W_5$ are added to the graph (Fig. 20). The cost of each link $(W_2, W_j)$ is the contribution of the choice $L_3 = W_j$ to (41).

**Step 3:**

Links from node $W_3$ to nodes $W_4$, $W_5$ and $n_{exit}$, are added to the graph (Fig. 21). The cost of each link $(W_3, W_j)$ is the contribution of the choice $L_3 = W_j$ to the objective function (41) when $L_2 = W_3$, while the cost of the link $(W_3, n_{exit})$ is the contribution of the choice $L_3 = W_3$ to the objective function (41) when $L_2 = W_2$.

**Step 4:**

Links from node $W_4$ to nodes $W_5$ and from node $W_4$ to $n_{exit}$ are added to the graph (Fig. 22). The cost of link $(W_4, W_5)$ is the contribution of the choice $L_3 = W_5$ to the objective function (41) when $L_2 = W_4$, while the cost of the link $(W_4, n_{exit})$ is the contribution of the choice $L_3 = W_4$ to (41) when $L_2 = W_2$ or $L_2 = W_3$.

**Step 5:**
Only a link from node $W_5$ to $n_{exit}$ is added to the graph (Fig. 23). The cost of the link $(W_5, n_{exit})$ is the contribution of the choice $L_3 = W_5$ to the objective function (41) when $L_2 = W_2$, $L_2 = W_3$ or $L_2 = W_4$.

Cumulative channel rates $L_1, \ldots, L_K$ that form an optimal solution of the problem $(P_1)$ can be found by solving a maximum cost path problem in the constructed graph $G(V, E)$.

Let $\Omega$ be the set of all possible values of the objective function $\Phi(L_1, \ldots, L_K)$ of the problem $(P_1)$ on the admissible region. Let $\Pi$ be the set of all possible paths from node $W_1$ to $n_{exit}$ in the graph $G(V, E)$ with maximum number of hops equal to $K$, and let $D_{\Pi}$ be the set of the lengths of paths in the set $\Pi$, i.e., the set of the costs of paths in the set $\Pi$. The graph $G(V, E)$ has been built in a way that the set $D_{\Pi}$ coincides with the set $\Omega$ ($D_{\Pi} \equiv \Omega$). The set of cumulative rates $L_1, \ldots, L_K$ that maximizes the objective function of the problem $(P_1)$ is the set of cumulative rates to which correspond the maximum element of the set $\Omega \equiv D_{\Pi}$. Thus, in order to find the optimal solution of the problem $(P_1)$ it is sufficient to find the longest path from node $W_1$ to $n_{exit}$ in the graph $G(V, E)$.

Since the graph $G(V, E)$ is acyclic [70], the Bellman-Ford algorithm [42] can be used to find the longest path from node $W_1$ to $n_{exit}$ with a number of hop lower or equal to $K$. Nodes of this path but $n_{exit}$ represent the set of optimum cumulative rates $L_1, \ldots, L_K$, i.e., the optimal solution of problem $(P_1)$.

**Computational complexity:** The graph $G(V, E)$ has a number of nodes $|V| = N + 1$ and
a number of links equal to

$$|E| = (K - 1) \cdot (N - K - 1) + \sum_{i=k}^{N} (1 + N - i)$$  \hspace{1cm} (52)

The Bellman-Ford algorithm, which dominates most of the computing time of the algorithm, has a computational complexity $O(|V| \cdot |E|)$. Thus, the complexity of the proposed algorithm for the channel rate determination is $O(K \cdot N^2)$.

### 3.2.2 Least cost source rooted trees

The classical optimization problem in multicast routing is the *Steiner tree problem in networks* (SPN) [9]. If a network can be modelled by a directed graph $G(V, E)$ consisting of a set of nodes $V$ and a set of arcs $E$, each node of the graph represents a router of the network and each arc of the graph represents a link of the network. Let $T = \{r_1, r_2, \ldots, r_m\} \in V$ be a group of nodes in $G$ and $s \in V$, $s \notin T$, the *Steiner tree problem* is to find the least-cost tree connecting the source $s$ with the group of destinations $T$ with the minimum total cost over all links. If each destination $r_k \in T$ has a bandwidth requirement of $W_k$ units on each edge of the path from $s$ to $r_k$, then the *Bandwidth-Constrained Least-Cost Steiner tree problem* has to be solved. This problem is to find the *least-cost* tree that respects the bandwidth requirements $W_k$ on each path from the source $s$ to the receiver $r_k$. Since both the *Steiner tree problem* and the *Bandwidth-Constrained Least-Cost Steiner tree problem* are NP-complete, efficient algorithms to solve these problems in polynomial time attain only approximate solution [147][9].

### 3.2.3 DIMRO in non-QoS-aware networks

By simply minimizing the cost of multicast trees without taking into account the link bandwidth availability, bottlenecks might occur in the network. In fact, since many multicast trees may use the same set of links, these links could be overloaded while other links in the network remain under-utilized.

The DIMRO algorithm proposed in this work follows a different approach to tackle the problem of bandwidth bottlenecks, one of the main reasons for low network performance. The multicast source rooted tree is built by choosing those paths that result least loaded to obtain a smooth load balancing. DIMRO obtains an optimal distribution of network resources which leads to maximize as much as possible the number of multicast trees that can be set up in the network.
Let us consider a multirate multicast scenario where receivers ask for different rates. If $K$ channels with rate $\{w_1, \ldots, w_K\}$ are used in the layering scheme, then $K$ cumulative rate $L_1, \ldots, L_K$ are available. Let $s$ be the source node, and let us suppose that the multicast group is made up of $M$ receivers. The DIMRO algorithm proceeds as it follows.

- **Step 0:**
  Receivers are ordered from the highest rate to the lowest one. At the end of this step we have an ordered set of $M$ receivers $\{r_1, \ldots, r_M\}$, with requested cumulative rates $F_1 \geq F_2 \geq \ldots \geq F_M \ (F_j \in \{L_1, \ldots, L_K\}, j = 1, \ldots, M)$. Receivers from $r_1$ to $r_M$ are connected to the source node progressively. In this way, receiver $r_i$ can reuse resources already exploited on paths from the source to receivers $r_1, \ldots, r_{i-1}$. These receivers, in fact, ask for rates $F_1 \geq F_2 \geq \ldots \geq F_{i-1} \geq F_i$.

Let $S_{\text{path}}(s, r_k)$ be the set of all feasible paths from $s$ to $r_k$. A path $p(s, r_k)$ from source $s$ to receiver $r_k$ is feasible if $b_{uv} \geq F_k$ for all its links, where $b_{uv}$ is the available bandwidth in link $(u, v)$.

- **Step $k$, $k = 1, \ldots, M$:**
  The algorithm chooses the path $\overline{p(s, r_k)} \in S_{\text{path}}(s, r_k)$ that minimizes the following function:
  \[
  f\{p(s, r_k)\} = \sum_{\{(u,v)\in p(s,r_k)\}} \frac{a_{uv}}{(1 - \rho_{uv})^\alpha},
  \]
  \[
  p(s, r_k) \in S_{\text{path}}(s, r_k)
  \]  \hspace{1cm} (53)

  1. $\rho_{uv}$ is the link $(u, v)$ utilization and it is defined as:
  \[
  \rho_{uv} = \frac{B_{uv} - (b_{uv} - F_k)}{B_{uv}} = \frac{B_{uv} - \beta_{uv}}{B_{uv}}
  \]  \hspace{1cm} (54)
  where $B_{uv}$ is the total bandwidth capacity of link $(u, v)$, $F_k$ is the bandwidth exploited on link $(u, v)$ by receiver $r_k$ and $\beta_{uv} = b_{uv} - F_k$ is the residual bandwidth of link $(u, v)$. It is worth pointing out that $\rho_{uv}$ is calculated after that $b_{uv}$ has been decremented by $F_k$.

  2. The exponent $\alpha = \alpha(|V|, |E|, F, \overline{B})$ in (53) is a function of the number of nodes $|V|$ and links $|E|$, the average rate $\overline{F}$ requested by receivers, and the average network link bandwidth $\overline{B}$. According to accurate empirical observations (described
hereafter) supported by extensive simulation results, the model for the exponent $\alpha$ has been chosen as:

$$\alpha(|V_i|, |E_i|, \overline{F_i}, \overline{B_i}) = a \cdot \exp \left\{ -b \cdot \frac{|E_i|}{|V_i|(|V_i| - 1)} \right\} \cdot \exp \left\{ -c \cdot \frac{\overline{F_i}}{\overline{B_i}} \right\} \quad (55)$$

Coefficients $a, b, c$ have been determined by solving the following minimization problem of the quadratic error between the exponent model in (55) and the experimental results:

$$\min \sum_{i=1}^{N} [\alpha_i - \alpha(|V_i|, |E_i|, \overline{F_i}, \overline{B_i})]^2 \quad (56)$$

Starting from a set of chosen points $\Gamma = \{(|V_i|, |E_i|, \overline{F_i}, \overline{B_i}), i = 1, \ldots, N\}$, the optimum exponent $\alpha_i$ was experimentally obtained for each point $(|V_i|, |E_i|, \overline{F_i}, \overline{B_i}) \in \Gamma$ of the set. The coefficient $a$ has been chosen belonging to the interval $[0, 3]$ after an accurate tuning of the model. Values of coefficients $a, b, c$ obtained by solving the minimization problem (56) are $(a; b; c) = (3; 3.9; 16.9)$.

Let us now come back to the $\alpha$ model in (55) to explain why this formula, with its variable aggregations $\overline{F}/\overline{B}$ and $|E|/(|V|(|V| - 1))$, is suitable to model the exponent in (53). By decreasing the value of the exponent $\alpha$, the length of paths found by the DIMRO algorithm is reduced because the difference between the metric of loaded links and the metric of unloaded links is reduced. It could be necessary to reduce the length of paths used by the multicast tree when the average rate $\overline{F}$ requested by the receivers grows, or when the average network bandwidth $\overline{B}$ decreases. In fact, the higher the value of $\overline{F}$, or the lower the value of $\overline{B}$, the more resources on each link are consumed by a single path. Thus, short paths have to be preferred. Simulations show that at the growing of the ratio $\overline{F}/\overline{B}$ the value of the optimal exponent $\alpha$ decreases exponentially.

As far as concerning $|V|$ and $|E|$, if the number of links $|E|$ decreases, there are less paths in the network, and it becomes very important taking care of the resource utilization. By increasing the value of the exponent $\alpha$, the difference between the metric of loaded links and unloaded links increases. If the number of nodes $|V|$ grows but the number of links $|E|$ remains the same, the exponent increases because
the number of paths decreases. Simulations show that at the decreasing of the ratio \( \frac{|E|}{(|V|(|V| - 1))} \) the value of the optimal exponent \( \alpha \) grows exponentially.

3. The binary variable \( a_{uv} \) is equal to zero if link \((u, v)\) already belongs to the tree, otherwise it is 1. If the set \( S_{path}(s, r_k) \) is not empty, let \( \overrightarrow{p(s, r_k)} \) be the path from the source \( s \) to the receiver \( r_k \) that minimizes the function in (53). On each link \((u, v)\) belonging to \( \overrightarrow{p(s, r_k)} \), if \( a_{uv} = 0 \), the value of the available bandwidth is not updated and uses those bandwidth resources already exploited by a path \( \overrightarrow{p(s, r_j)} \), with \( F_j \geq F_k \). Conversely, if \( a_{uv} \neq 0 \), then new resources must be consumed and the new value of the available bandwidth is \( b'_{uv} = b_{uv} - F_k \). Then the binary variable \( a_{uv} \) is set to zero in order not to consider the cost of link \((u, v)\) for those future paths which will exploit it.

Let us point out that at Step \( k \), \( k = 1, \ldots, M \), the optimal path \( \overrightarrow{p(s, r_k)} \in S_{path}(s, r_k) \) can be found using a shortest path algorithm (as Dijkstra or Bellman-Ford algorithm), where the length of each link in the network is set to:

\[
\begin{align*}
    d_{uv} = \begin{cases} 
        \frac{a_{uv}}{(1 - \rho_{uv})^\alpha} & \text{if } b_{uv} \geq F_k \\
        \infty & \text{if } b_{uv} < F_k \text{ or } (u, v) \notin E
    \end{cases}
\end{align*}
\]  

(57)

In this work we use the Bellman-Ford algorithm which finds a spanning tree of the shortest paths from the source to all other nodes of the graph. The path \( \overrightarrow{p(s, n)} \) from node \( s \) to node \( n \), solution of (53), is the one that minimizes the function:

\[
\sum_{\{((u,v) \in \overrightarrow{p(s,n)}\}} d_{uv}
\]

(58)

**Computational complexity:** If the number of receivers is \( M \), the DIMRO algorithm builds the multicast tree by computing for as many as \( M \) times the spanning tree (Bellman-Ford algorithm). Since the time complexity of the Bellman-Ford algorithm is \( O(|V| \cdot |E|) \), then the computational complexity of the algorithm is \( O(M \cdot |V| \cdot |E|) \).

### 3.2.4 DIMRO in DiffServ aware networks

In this section the DIMRO algorithm is integrated into a DiffServ network. Let us consider a multirate multicast scenario where receivers ask for the same data content but different rates and different qualities of service (QoS), mapped into the respective service class according to
[34][35]. In particular, DIMRO can build source rooted tree for real time multicast communication, by adopting the Expedited Forwarding (EF) DiffServ service class [90], and for non real time communication, by adopting the Assured Forwarding (AS) DiffServ service class [81].

The DIMRO algorithm, integrated in a DiffServ network, allows a receiver to reuse the bandwidth already exploited by receivers asking for higher service classes without any extra cost for the network. By building a path from the source $s$ to a receiver $r$, the DIMRO algorithm can reuse some sub-paths already exploited by higher service class receivers. Thus, receiver $r$ obtains a better QoS, and the network saves resources because bandwidth already exploited by other receivers is reused for receiver $r$. DIMRO in a DiffServ network proceeds into steps, described hereafter.

- **Step 0:**

Receivers are ordered according to their service class and rate. Let us consider a set $R$ made up of $M$ receivers. Let $CL = \{cl_1, \ldots, cl_L\}$ be the set of service classes requested by receivers, where $cl_1$ is the highest service class and $cl_L$ is the lowest one. $CL$ is a subset of all service classes supported by the DiffServ architecture [34][35]. The set of receivers $R$ is partitioned into $L = |CL|$ subsets:

$$R = \bigcup_{i=1}^{L} R_i \quad (59)$$

Subset $R_i$ is made up of those receivers asking for service class $cl_i$, $i = 1, \ldots, L$. Receivers in each subset $R_i$, are ordered from the highest rate to the lowest one. Let us consider a partitioned and ordered set $R$ made up of $M$ receivers. Let $r^i_k$ be the receiver $k$ in the subset $R_i$ with requested rate $F^i_k$. $|R_i|$ stands for the cardinality of $R_i$, and $i$ represents the associated service class.

- **Step $j$, $j = 1, \ldots, L$:**

DIMRO connects the source $s$ with all those receivers asking for service class $cl_i$. Initially, the binary variables $a_{uv}$ are set to 1 for each network link $(u, v)$. For receiver $r^i_k$, $k = 1, \ldots, |R_i|$, $a_{uv}$ are set to 0 for all those links $(u, v)$ already used by receivers that ask for higher service class and higher or equal rate then $r^i_k$. In fact, resources already exploited by the tree on these links could be used by $r^i_k$ without any added cost.
Let $b_{uv}(cl_i)$ be the available bandwidth of link $(u, v)$ for the service class $cl_i$. A path $p(s, r_k^i)$ from source $s$ to a receiver $r_k^i$ asking for cumulative rate $F_k^i$ is feasible if $b_{uv}(cl_i) \geq F_k^i$ for all its links. DIMRO algorithm chooses that feasible path $p(s, r_k^i)$ from $s$ to $r_k^i$ that minimizes the function:

$$f\{p(s, r_k^i)\} = \sum_{\{u,v\} \in p(s, r_k^i)} \frac{a_{uv}}{(1 - \rho_{uv}(cl_i))^\alpha},$$

$$p(s, r_k^i) \in S_{path}(s, r_k^i)$$

(60)

where $S_{path}(s, r_k^i)$ is the set of feasible paths from $s$ to $r_k^i$.

Exponent $\alpha$ is defined as in (55), and utilization $\rho_{uv}(cl_i)$ of the link $(u, v)$ is calculated as follows:

$$\rho_{uv} = \frac{B_{uv}(cl_i) - [b_{uv}(cl_i) - F_k^i]}{B_{uv}(cl_i)}$$

(61)

where $B_{uv}(cl_i)$ is the bandwidth capacity of link $(u, v)$ for the class $cl_i$ and $F_k^i$ is the cumulative rate requested by $r_k^i$.

For each link $(u, v)$ belonging to path $p(s, r_k^i)$, if $a_{uv} = 0$ then the available bandwidth $b_{uv}(cl_i)$ is not updated and the new path uses those resources already exploited by other receivers. On the contrary, if $a_{uv} \neq 0$, then new resources have to be exploited and the new value of the available bandwidth becomes $b_{uv}(cl_i) = b_{uv}(cl_i) - F_k^i$. Then the binary variable $a_{uv}$ is set to zero.

The DIMRO algorithm determines the path $p(s, r_k^i)$ by using a modified shortest-path algorithm. Common shortest path algorithms could choose one of those possible paths with zero cost from $s$ to $n$, but $p(s, r_k^i)$ might not be correct. To determine a correct path the algorithm proceeds as it follows.

Starting from node $s$, the first link $(\tilde{u}, \tilde{v})$ belonging to $p(s, r_k^i)$ with $a_{uv} \neq 0$ is found. Among all those paths in the tree exploiting a bandwidth equal or greater than $F_k^i$ and passing through node $\tilde{u}$, it is chosen the one which uses the highest service class. Let us indicate this path with $\tilde{p}_{\tilde{u}}$. The new path $\tilde{p}(s, r_k^i)$ from $s$ to $r_k^i$ is the concatenation of the sub-path from $s$ to node $\tilde{u}$, belonging to path $\tilde{p}_{\tilde{u}}$, and the sub-path from node $\tilde{u}$ to $r_k^i$, belonging to $p(s, r_k^i)$.

**Example.** Figure 24 shows a multicast tree where receiver $r_1$ (node 6) asks for a rate $F_1 = 2$ and a QoS mapped into service class A, receiver $r_2$ (node 8) asks for a rate $F_2 = 3$ and a QoS mapped into service class B, and receiver $r_3$ (node 7) asks for a rate $F_3 = 2$ and a QoS mapped into service class C. Path from source $s$, node 1, to receiver $r_1$ is the first to be built. Receiver
Figure 24: Correct path computation

$r_2$ asks for a service class B but a rate $F_2 = 3 > F_1 = 2$, thus the binary variables $a_{uv}$ are not set to 0 on links of path from node 1 to node 6. Path from node 1 to node 8 is built. Let $p(s, r_i)$ be the path from source $s$ to receiver $r_i$. Receiver $r_3$ asks for a service class C and a rate $F_3 = F_1 = 2 < F_2 = 3$ thus, on each link of paths $p(s, r_1)$ and $p(s, r_2)$, $a_{uv}$ are set to 0 because $r_3$ can reuse resources already exploited by other receivers. By using a common shortest path algorithm, path $p = \{1 \rightarrow 2 \rightarrow 3 \rightarrow 5 \rightarrow 7\}$ might be found from source $s$ to receiver $r_3$ (node 7) because paths $\{1 \rightarrow 2 \rightarrow 3 \rightarrow 5\}$ and $\{1 \rightarrow 4 \rightarrow 3 \rightarrow 5\}$ have both cost equal to zero. The point is that this path $p$ cannot be taken into consideration since link $(3, 5)$ belongs only to the path $\{1 \rightarrow 4 \rightarrow 3 \rightarrow 5 \rightarrow 8\}$ ($p(s, r_2)$ in Fig. 24) and not to the path $\{1 \rightarrow 2 \rightarrow 3 \rightarrow 6\}$ ($p(s, r_1)$ in Fig. 24). For the mentioned reason, it should be clear that the correct path is $\{1 \rightarrow 4 \rightarrow 3 \rightarrow 5 \rightarrow 7\}$. Once the shortest path algorithm has determined a path from the source $s$, node 1, to receiver $r_3$, node 7, the last node of the zero cost path is determined, in Fig. 24 node 5. All paths connecting the source with those receivers that ask for a higher or equal service class and a higher or equal requested rate than $r_3$ are visited (in Fig. 24 path from $s$ to $r_1$ and from $s$ to $r_2$). Only path from $s$ to $r_2$ has a sub-path $\{1 \rightarrow 4 \rightarrow 3 \rightarrow 5\}$ from $s$ to node 5. Thus, path $\{1 \rightarrow 4 \rightarrow 3 \rightarrow 5\}$ is chosen among all path with zero cost from $s$ to node 5. Finally, the correct paths $\{1 \rightarrow 4 \rightarrow 3 \rightarrow 5 \rightarrow 7\}$ from $s$ to $r_3$ is selected.
3.3 DIMRO-GS - DiffServ-Integrated Multicast algorithm for Internet Resource Optimization Group Shared applications

In this section, we present DIMRO-GS, a novel algorithm to efficiently build shared trees for group shared multicast communications. A shared tree is a generalization of a source rooted tree. In a source rooted tree the source node multicasts data to a set of receiver nodes, whereas in a shared tree each member node can multicast data to all the other member nodes (videoconference is probably the most common multicast application requiring a shared tree, but many others applications are conceivable such as distributed games, file sharing, collaborative groupware, replicated database, etc.).

In Section 3.3.1 we explain how DIMRO-GS works in non-QoS-aware networks and show how this algorithm manages to keep low the probability of bottleneck occurrence. In Section 3.3.2 we point out the main improvements in DIMRO-GS when a QoS-aware network is considered. In this last case the full set of features, which DIMRO-GS is endowed with, is described and analyzed.

As briefly introduced in Section 3.1, a possible approach to build a shared tree is proposed in the Core Based Protocol (CBT) [21][22] and in its enhanced version (CBTv2) [20]. The CBT protocol connects each member node to a core node using bi-directional shortest paths. The main drawback with this approach is the traffic concentration occurring in the core node. In fact, all data sent to a member node from all the other member nodes pass through the same path from the core node to the destination member node. Thus, some link might be overloaded, specially in multicast applications with high bandwidth requirements. Moreover, multicast trees set up with this approach are quite easy to be computed, but are far from being optimized. Let us consider an extreme example to catch the inefficiency of this protocol: if two member nodes had a link connecting each other, still their traffic would pass through the core node, no matter the state of their common link.

To avoid traffic concentration a shared tree can be computed by separately finding $M$ source rooted trees (where $M$ is the number of member nodes). The source rooted tree $i$, $i = 1, \ldots, M$ has the member node $n_i$ as root and all the other member nodes $n_j$, $j = 1, \ldots, M$ (with $j \neq i$), belonging to the tree, not necessarily as leaves. The FTM (Feasible solutions using adapted TM algorithm) algorithm, proposed in [151][107], follows this approach. The source rooted tree $i$, $i = 1, \ldots, M$ connects the root node $n_i$ with the member node $n_j$, $j = 1, \ldots, M$, $j \neq i$, by
using the path with the greatest capacity from node $n_i$ to node $n_j$. The bottleneck of a path is characterized as the used link with the minimum available bandwidth, and the path bandwidth capacity is defined as the available bandwidth of its bottleneck. The path with the greatest bandwidth capacity from node $n_i$ to node $n_j$ is called the *widest path*. The FTM algorithm uses information obtained by the Breadth First Search procedure (BFS) [65]. For each member node the BFS procedure builds a tree rooted at the member node and connecting each network node with this member node by exploiting the *widest path paradigm criterion*. After $M$ BFS trees are built rooted at each group member $n_i$, the detailed pieces of information about the *widest path* connecting each pair of member nodes are available.

In the next step, the FTM procedure builds $M$ multicast source rooted trees, one for each group member $n_i$. As far as concerning the member node $n_i$, paths with the maximum bandwidth capacity will be used by the source node $n_i$ to join all the other group member. If two or more paths with the same bandwidth capacity are found at the same time, the one with the least cost is selected. In order to generate each multicast tree, two tables must be available: the *bandwidth capacity* table and the *path cost* table. These tables are both $M \cdot |V|$ in size, where $M$ is the number of members and $|V|$ is the number of network nodes.

**Computational complexity:** The FTM computational complexity is dominated by the BFS complexity which is $O((|V| + |E|) \cdot |V|^2)$. Thus, building $M$ multicast trees by exploiting the FTM algorithm, is $O(M \cdot (|V| + |E|) \cdot |V|^2)$.

### 3.3.1 DIMRO-GS in non-QoS aware networks

In Section 3.2 the DIMRO algorithm has been presented. This algorithm can be extended to the multicast shared tree case. We called this extended algorithm DIMRO-GS (Group Shared DIMRO). If there are $M$ group members, the shared tree can be set up by building $M$ source rooted trees, each one having a different group member as root and all the other group members belonging to the tree, not necessarily as leaves. Each source rooted tree can be built according to the DIMRO algorithm [124]. Let us assume that each group member has the same bandwidth requirement, i.e., $W_k = \tilde{W}$, $k = 1, \ldots, M$. In this case, the first step of the DIMRO algorithm, aimed at ordering receivers according to their bandwidth requirement, is skipped. When $M$ source rooted trees are computed, the shared tree is completed. From then onwards, each member group can send data to all the other members and receive data from all the other
members.

**Computational complexity:** In the shared tree case, if $M$ is the number of group members, then $M$ source rooted trees have to be built. The computational complexity for each source rooted tree by using the DIMRO algorithm is $O(M \cdot |V| \cdot |E|)$, as seen in Section 3.2.3; thus, the computational complexity of the DIMRO-GS algorithm becomes $O(M^2 \cdot |V| \cdot |E|)$.

It has been exposed in the previous subsection that the computational complexity of FTM algorithm is $O(M \cdot (|V| + |E|) \cdot |V|^2)$. Since in most practical case $M \ll |V|$, the computational complexity of the proposed algorithm is lower than the FTM complexity. Furthermore, FTM needs two $M \cdot |V|$ tables that must be stored in memory, while in DIMRO-GS no tables are required.

### 3.3.2 DIMRO-GS in DiffServ aware networks

In this section the DIMRO-GS algorithm is integrated into a DiffServ aware network. Let us consider a multicast scenario where group members in a group shared communications can ask for a different quality of service (QoS). The QoS requirements of each group member are mapped into the associated service class. In particular, DIMRO-GS can build group shared tree for both real time multicast communications, by adopting the Expedited Forwarding (EF) DiffServ service class [90], and non real time communications, by adopting the Assured Forwarding (AS) DiffServ service class [81]. The DIMRO-GS algorithm integrated in a DiffServ aware network allows a group member acting as receiver to reuse the bandwidth already exploited by group members asking for higher service classes, without any extra cost for the network. By building a path from a group member $s$, acting as source, to a group member $r$, acting as receiver, the DIMRO-GS algorithm can reuse some sub-paths already exploited by group members, acting as receivers, with more stringent QoS requirements. Thus, group member $r$ obtains a better QoS and the network saves resources because bandwidth already exploited by other group members is reused for group member $r$. The DIMRO-GS algorithm in a DiffServ aware network proceeds as it follows.

Let us consider a set $R$ made up of $M$ group members. At the first step, group members are ordered from the highest service class to the lowest one. The set $R$ is ordered per service class, thus, if $k < h$, then member group $n_k$ asks for a higher or equal service class than group $n_h$.

**Example.** Let us consider a simple DiffServ architecture with only four different classes of
service: \(A, B, C, D\). Suppose \(A\) is the highest service class whereas \(D\) is the lowest one. Let us indicate a group member \(r\) with the couple \((F; C) = (\text{RequestedRate}; \text{ServiceClass})\) representing the rate and service class requested by group member \(r\). Let us consider the following multicast group: \(R = \{(1; B), (1; A), (1; B), (1; D), (1; C), (1; C)\}\). The algorithm proceeds as it follows:

- **Step 0:**
  The multicast group is sorted per service class in a lexicographic order, as shown hereafter:
  \[R^* = \{(1; A), (1; B), (1; B), (1; C), (1; C), (1; D)\}\.]

- **Step \(i, i = 1, \ldots, M\):**

  DIMRO-GS uses the DIMRO algorithm to builds a source rooted tree with the member node \(n_i\) as root and all the other member nodes belonging to the tree. At the beginning of Step \(i\), the binary variables \(a_{uv}\) are initialized to 1 for each network link \((u, v)\). Let \(cl_j\) be the service class requested by the group member \(n_j, j = 1, \ldots, M, n_j \neq n_i\), acting as receiver, and let \(b_{uv}(cl_j)\) be the available bandwidth for the service class \(cl_j\) on link \((u, v)\). A path \(p(n_i, n_j)\) from the node \(n_i\) to a node \(n_j\) is feasible if \(b_{uv}(cl_j) \geq \tilde{W}\) for all its links (\(\tilde{W}\) is the rate requested by each group member). For each group member \(n_j, j = 1, \ldots, M, n_j \neq n_i\) the DIMRO algorithm chooses that feasible path \(p(n_i, n_j)\) from the node \(n_i\) to a node \(n_j\) that minimizes the function:

  \[
  f\{p(n_i, n_j)\} = \sum_{\{(u,v) \in p(n_i, n_j)\}} \frac{a_{uv}}{(1 - \rho_{uv}(cl_j))^\alpha}, \quad p(n_i, n_j) \in S_{\text{path}}(n_i, n_j)
  \] (62)

  where \(S_{\text{path}}(n_i, n_j)\) is the set of all feasible paths from the node \(n_i\) to a node \(n_j\). For each link \((u, v)\) belonging to path \(p(n_i, n_j)\), if \(a_{uv} = 0\), then the available bandwidth \(b_{uv}(cl_j)\) is not updated, and the new path uses those resources already exploited by another member group, acting as receiver, that asks for a higher or equal service class. On the contrary, if \(a_{uv} \neq 0\), then new resources must be exploited and, thus, the new value of the available bandwidth becomes \(b'_{uv}(cl_j) = b_{uv}(cl_j) - \tilde{W}\). Then the variable \(a_{uv}\) is set to 0. This way the path from the node \(n_i\) to a node \(n_k, k = j + 1, \ldots, M\), can reuse resources already used on path \(p(n_i, n_j)\) without any adding cost for the network.
Table 10: Network parameters

<table>
<thead>
<tr>
<th></th>
<th>α</th>
<th>β</th>
<th>B</th>
<th>nodes</th>
</tr>
</thead>
<tbody>
<tr>
<td>Network 1</td>
<td>0.2</td>
<td>0.4</td>
<td>100 Mbps</td>
<td>100</td>
</tr>
<tr>
<td>Network 2</td>
<td>0.3</td>
<td>0.6</td>
<td>100 Mbps</td>
<td>100</td>
</tr>
</tbody>
</table>

3.4 Simulation Results

3.4.1 Random network model

To ensure a fair evaluation of different routing algorithms, a random network has been generated according to the Waxman’s model [152][40]. In the Waxman’s model, network nodes are randomly distributed across a Cartesian coordinate grid. Links are added to the graph by considering all possible pairs \((u, v)\) of nodes and by using the probability function:

\[
P_e(u, v) = \beta \cdot \exp\left(-\frac{d_{uv}}{\alpha \cdot L}\right)
\]

where \(P_e(u, v)\) is the existence probability of a link between nodes \(u\) and \(v\), \(d_{uv}\) is the Euclidean distance between the node \(u\) and the node \(v\), \(L\) is the maximum possible distance between a pair of nodes, \(\alpha\) and \(\beta\) are parameters belonging to the range \((0,1]\). A high \(\alpha\) value increases the number of connections to nodes further away, while a high \(\beta\) value increases the node degree.

Unlike the original Waxman’s model [152], we suppose that each link added to the network is bi-directional. The bandwidth capacity \(B_{uv}\) of each link \((u, v)\) is randomly generated using an uniform distribution with mean bandwidth \(\overline{B}\). Consequently, the link \((u, v)\) bandwidth capacity \(B_{uv}\) may be different from the link \((v, u)\) bandwidth capacity \(B_{vu}\).

In this work, two different one hundred node random networks are generated, according to Tab. 10. Network 2 has a higher number of links than Network 1, according to the probability function (63). The average bandwidth in both networks is \(\overline{B} = 100 Mbps\).

3.4.2 DIMRO Performance Evaluation

The DIMRO algorithm is compared to the optimal solution of the Steiner tree problem when all link costs are equal to 1. If every links have a cost equal to 1, then the minimum-weight tree that spans all group members is the multicast tree that uses the least number of links.

The minimum-weight tree is found by solving the flow formulation of the Steiner tree problem proposed in [44][154]. We implemented the Integer Linear Programming (ILP) problem in AMPL [64] and solved it with the CPLEX [2] solver.
The DIMRO algorithm and the optimal solution of the Steiner tree problem with unitary costs are compared by using two metrics: the Rejection Rate and the Network Load. For each simulation campaign several experiments have been run to ensure a 95% relative confidence intervals smaller than 5%.

Starting from a completely unloaded Waxman network [152], it is requested to build a fixed number of source rooted trees, in the following the Number of Requested Trees. Multicast groups are sequentially randomly generated. Multicast group members (source and receivers) are randomly chosen among network nodes. The number of receivers for each multicast group is uniformly distributed from 5 to 15, and the bandwidth request of each receiver is uniformly distributed from 0.1 to 2 Mbps.

To evaluate the performance of the DIMRO algorithm in the different simulated scenarios, we use two metrics, the Rejection Rate and the Network Load.

1. The Rejection Rate is defined as:

   \[
   \text{Rejection Rate} = \frac{\text{Number of Rejected Trees}}{\text{Number of Requested Trees}}
   \]

   (64)

   where the Number of Requested Trees is the total number of source rooted trees sequentially generated, while the Number of Rejected Trees is the number of requested source rooted trees that cannot be built because there are not enough resources in the network.

2. The Network Load \( \bar{\rho} \) is defined as:

   \[
   \bar{\rho} = \frac{\sum_{(u,v) \in E} \rho_{uv}}{|E|}
   \]

   (65)

   where \( E \) is the set of network links, \(|E|\) its cardinality, and \( \rho_{uv} \) is the Link Load of link \((u, v)\), defined as:

   \[
   \rho_{uv} = \frac{\text{Used Bandwidth of link } (u,v)}{\text{Bandwidth Capacity of link } (u,v)}
   \]

   (66)

   Figure 25 shows that the DIMRO Rejection Rate in Network 1 is lower than the Rejection Rate of the Optimal solution of the Steiner Tree Problem (OSTP) with \( c_{uv} = 1 \). The Rejection Rate is approximately the same until the Number of Requested Trees is less than 2500. When the Number of Requested Trees overcomes this threshold, the DIMRO algorithm has a lower Rejection Rate.

   Both DIMRO and OSTP Rejection Rates grow at the growing of the Number of Requested Trees because the same bottlenecks occur. These bottlenecks depend on the network topology...
Figure 25: DIMRO and OSTP Rejection Rate in Network 1

Figure 26: DIMRO and OSTP Network Load in Network 1
and cannot be avoided, but the DIMRO Rejection Rate is lower because bottlenecks occur later. Figure 26 shows that the DIMRO Network Load is lightly lower than the OSTP Network Load until the Rejection Rate is the same. Then, since DIMRO rejects less trees, its Network Load is higher than the OSTP one.

Network 2 has a higher number of links than Network 1, according to (63) and the network parameters in Tab. 10. In this second type of network, the DIMRO Rejection Rate is significantly lower than the OSTP Rejection Rate (Fig. 27). This is because less unavoidable bottlenecks exist. An unavoidable bottleneck is a bottleneck that depends on the network topology and not on the algorithm used in the network. For example, if only one path exists between two nodes, all communications between these two nodes must use this path. Since Network 2 has a higher number of links, the number of possible paths between two nodes grows, and it is easier for the DIMRO algorithm to avoid bottlenecks. A lower Network Load (Fig. 26 and Fig. 28) with a higher Rejection Rate (Fig. 25 and Fig. 27) proves that OSTP does not use efficiently network resources. In fact, bottlenecks degrade network performance and not all available resources can be exploited.
3.4.3 DIMRO-GS Performance Evaluation

To evaluate the performance of the DIMRO-GS algorithm and the FTM algorithm, presented in the introduction of Section 3.3, in the different simulated scenarios, we use the two metrics previously introduced slightly adapted to the group shared case, as will be clarified in the following, plus a third metric, the Number of Links.

1. The Rejection Rate is defined as in (64), but now the Number of Requested Trees is the total number of shared trees sequentially generated, and the Number of Rejected Trees is the number of requested shared trees that cannot be built because there are not enough resources in the network. Shared groups are sequentially randomly generated. For each multicast group, members are randomly chosen among network nodes. The number of members for each multicast group is uniformly distributed from 5 to 15, and the bandwidth requirement of each multicast group is uniformly distributed from 0.1 to 2 Mbps.

2. The Network Load is exactly defined as in (65).

3. The Number of Links is the total number of used links to connect each group member.
For each simulation several experiments have been run to ensure a small trust region (95% relative confidence intervals smaller than 5%). Simulation results for both Network 1 and Network 2 show (Fig. 29 and Fig. 31) that at the growing of the Number of Requested Trees, the FTM Rejection Rate is significantly higher than the one of the DIMRO-GS algorithm. This can be explained by considering that the FTM algorithm uses a higher amount of network resources. In fact, at the growing of the Number of Requested Trees, the FTM Network Load is significantly higher than the DIMRO-GS Network Load (Fig. 30 and Fig. 32). This implies that network resources saturate later when the DIMRO-GS algorithm is used, causing a lower Rejection Rate for this algorithm. The lower is the number of links, the more bottlenecks degrade network performance. Figure 31 and Fig. 32 show that the DIMRO-GS algorithm achieves a lower Rejection Rate and a higher Network Load in Network 2. On the contrary, in Network 1 (Fig. 29 and Fig. 30) bottlenecks do not allow to efficiently exploit all network resources.

Moreover, another observation that can be pointed out is that the Number of Links that make up a shared tree built by the FTM algorithm is higher than the Number of Links that make up the same shared tree built by the DIMRO-GS algorithm. This means that the amount of network

Figure 29: DIMRO-GS and FTM Rejection Rate in Network 1
Figure 30: DIMRO-GS and FTM Network Load in Network 1

Figure 31: DIMRO-GS and FTM Rejection Rate in Network 2
resources used by the FTM algorithm is higher than the amount of network resources exploited by the DIMRO-GS algorithm. At the growing of the group size, the Number of Links that make up the shared tree built by the proposed algorithm grows slower than the Number of Links that make up the shared tree built by the FTM algorithm, in both Network 1 and Network 2 (Fig. 33).

Figure 33 shows also that the Number of Links that make up the shared tree built by the DIMRO-GS algorithm is greater in the first network than in the second one. This is because when the number of links in the network grows, the value of the exponent decreases. Thus, paths built by the proposed algorithm are shorter, and the Number of Links that make up the shared tree is lower.

### 3.4.4 DiffServ DIMRO and DIMRO-GS Performance Evaluation

In this section we evaluate the performance of the DIMRO algorithm and the DIMRO-GS algorithm integrated in a DiffServ network, and the simple case, not DiffServ integrated, in which a group member/receiver cannot reuse the bandwidth exploited by another group member/receiver with more stringent QoS requirements.
Network nodes are randomly placed on a Cartesian coordinate grid and links are generated according to the probability function (63). Since we need to generate a DiffServ network, we will consider the bandwidth capacity of a link for each service class. Let us consider a simple DiffServ network with only four service classes that we indicate with $A, B, C, D$, where $A$ is the highest service class and $D$ is the lowest one. Let $B_{uv}(cl)$ be the bandwidth capacity of link $(u, v)$ for service class $cl$, $cl \in \{A, B, C, D\}$. The bandwidth capacity $B_{uv}(cl)$ for each service class on link $(u, v)$ is randomly generated by using an uniform distribution with mean bandwidth $\bar{B}(cl)$. For this simulation campaign, a one hundred node network has been randomly generated, with parameters $\alpha = 0.2$ and $\beta = 0.4$, according to Tab I. The link bandwidth capacity for each service class $cl \in \{A, B, C, D\}$ is uniformly distributed with mean bandwidth $\bar{B}(cl)$ equal to 25 Mbps.

We consider the Network Load as defined in (65) for each service class, when fifty source rooted trees, randomly generated, are built by the DIMRO algorithm. Each receiver asks for a rate uniformly distributed in the range $[0.1, 2]$ Mbps, and a service class randomly selected within the set $\{A, B, C, D\}$. Moreover, a 5 channel layering scheme is used, as described in
Section 3.2.1.

For each simulation several experiments have been run to ensure a small trust region (95% relative confidence intervals smaller than 5%). Figure 34 shows that DiffServ integrated DIMRO performs better than its non integrated version. Furthermore, the Network Load decreases with the service class. In particular, the lowest service class D has the lowest Network Load because D class receivers can reuse bandwidth already exploited by all other receivers. DiffServ integrated DIMRO performs better than its non QoS-aware version, and this gets evident at the growing of the number of receivers. Figure 34 shows that for 30 receivers the Network Load for service classes B, C and D is significantly lower then the Network Load relative to these service classes when DIMRO is not DiffServ integrated (the Network Load is reduced respectively by 23%, 33% and 40%, as can be easily figured out from Fig. 34).

As far as concern the DIMRO-GS algorithm, we consider the Network Load of each service class when a multicast group is randomly generated and the DIMRO-GS algorithm is used to build the corresponding shared tree. The group asks for a rate uniformly distributed in the range $[0.1, 2]$ Mbps, and each group member asks for a service class randomly selected within the set $\{A, B, C, D\}$. 

Figure 34: DIMRO Network Load of DiffServ classes
Figure 35: DIMRO-GS Network Load of DiffServ classes

Figure 35 shows that the DIMRO-GS algorithm integrated in a DiffServ aware network performs better than the simple case (not DiffServ integrated). The Network Load decreases with the service class (the lowest service class D has the lowest Network Load). Moreover, the DiffServ integrated DIMRO-GS outperforms at the growing of the number of group members.

When a group member/receiver reuses resources already exploited by a receiver with more stringent QoS requirements, it obtains a better QoS than that it requires, with no extra cost to the network. We call lucky receivers (lucky members in the DIMRO-GS case) those receivers (members) that use the bandwidth relative to a higher class at least in one link.

Figures 36 and 37 show that the lucky receiver (member) rate, i.e., the percentage of receivers (members) that receive a better quality of service than that they requested, grows as the multicast group enlarges. Figure 36 shows also that DIMRO lucky receiver rate grows at the decreasing of the number of channels, i.e., the number of cumulative rates, in the layering scheme. In fact, by reducing the number of possible cumulative rates, it becomes more probable that receivers with different QoS requirements ask for the same rate and, thus, it is more likely that receivers asking for less stringent QoS requirements can reuse resources already exploited by receivers with more stringent QoS requirements. In other words, by reducing the number
Figure 36: DIMRO Lucky Receivers

Figure 37: DIMRO-GS Lucky Members
of channels, less receivers can obtain the desired rate, but more receivers attain a better QoS, which means less losses, less retransmissions, and an increased throughput on the average.

### 3.5 Conclusions

DIMRO (DiffServ-Integrated Multicast algorithm for Internet Resource Optimization) and DIMRO-GS (DiffServ-Integrated Multicast algorithm for Internet Resources Optimization in Group Shared applications) algorithms, presented in this work, aim to avoid bandwidth bottlenecks effectively, and to achieve an efficient distribution of link loads in the source rooted and group shared multicast tree computation. The proposed multicast algorithms, seamless integrated into the DiffServ Quality of Service approach, by leveraging their native multirate nature, efficiently address the network partition problem. Both these multicast algorithms, embedded with a novel low computational cost multirate layering algorithm to effectively accommodate heterogeneous receiver requested rates, have been described and tested in an integrated DiffServ aware network through extensive C++ based simulations. The simulation results show a better leverage of the network bandwidth resources and an improved QoS perceived by multicast group members with respect to the performance of existing multicast algorithms. Moreover, the low computational complexity which characterizes all the proposed algorithms, effectively leads to time and resource saving.
CHAPTER IV

MOBILE AD HOC NETWORKS (MANET)

4.1 The Notion of “Ad Hoc Network”

A Mobile Ad Hoc Network (MANET) is a network architecture that can be rapidly deployed without relying on pre-existing fixed network infrastructure. The nodes in a MANET can dynamically join and leave the network, frequently, often without warning, and possibly without disruption to other nodes’ communications. In addition, nodes in a ad hoc network can be highly mobile. Thus, the node constellation and the presence or absence of links can rapidly change. Examples of possible applications for MANETs are:

- **Tactical operations**, for fast establishment of military communications during the deployment of forces in unknown and hostile terrain;
- **Rescue missions**, for communication in areas without adequate wireless coverage;
- **National security**, for communications in times of national crisis, where the existing communication infrastructure is non-operational due to a natural disaster or a global war;
- **Law enforcement**, for fast establishment of a communication infrastructure during law enforcement operations;
- **Commercial use**, for quickly setting up communications in exhibitions, conferences, or sale presentations;
- **Education**, for operation of wall-free (virtual) classrooms;
- **Sensor networks**, for communications among sensors which monitor an area and sense the environment for event occurrence.

Nodes in a MANET exhibit nomadic behavior by freely migrating within the terrain, dynamically creating and tearing down associations with other nodes. Groups of nodes that have a common goal can create formations (clusters) and migrate together, similarly to military units on missions or to guided tours on excursions. Nodes can communicate with each other at any
time and without restrictions, except for connectivity limitations and/or limitations due to security provisions. Examples of network nodes are pedestrians, soldiers, sensors, or unmanned robots. Examples of mobile platforms on which network nodes might reside are cars, trucks, buses, tanks, trains, planes, helicopters, or ships.

MANETs are intended to provide a data network that is immediately deployable in arbitrary communication environments and is responsive to changes in network topology. Because ad hoc networks are intended to be deployable anywhere, existing infrastructure may not be present. The mobile nodes are thus likely to be the sole elements of the network. Differing mobility patterns and radio propagation conditions that vary with time and position can result in intermittent and sporadic connectivity between adjacent nodes. The result is a time-varying network topology. MANETs are distinguished from other ad hoc networks by rapidly changing network topologies, influenced by the network size and node mobility. Such networks typically have a large span and contain hundreds to thousands of nodes. MANET nodes exist on top of diverse platforms that exhibit quite different mobility patterns. Within a MANET, there can be significant variations in node speed (from stationary nodes to high-speed aircraft), direction of movement, acceleration/deceleration or restrictions on paths (e.g., cars must drive on a road, but tanks do not). A pedestrian is restricted by built objects while airborne platforms can exist anywhere in some range of altitudes. In spite of such volatility, MANETs are expected to deliver diverse traffic types, ranging from pure voice to integrated voice and image, and even possibly some limited video.

4.2 The Communication Environment and the MANET Model

The following are a number of assumptions about the communication parameters, the network architecture, and the network traffic in a MANET.

- Nodes are equipped with portable communication devices. Lightweight batteries may power these devices. Limited battery life imposes restrictions on the transmission range, communication activity (both transmitting and receiving) and computational power of these devices.

- Connectivity between nodes is not a transitive relation, i.e., if node A can communicate directly with node B and node B can communicate directly with node C, then node A
may not be able to communicate directly with node \( C \). This leads to the hidden terminal problem [149]. Moreover, links are likely to be asymmetrical, i.e., if node \( A \) can communicate with node \( B \), node \( B \) may not be able to communicate to node \( A \).

- A hierarchy in the network routing and mobility management procedures could improve network performance measures such as latency in locating a mobile. However, a physical hierarchy may lead to areas of congestion and is very vulnerable to frequent topological reconfigurations.

- Nodes are generally assumed to be identified by fixed identification numbers, i.e., IDs (e.g., based on IP addresses [125]).

- All the network nodes have equal capabilities. This means that all nodes are equipped with identical communication devices and are capable of performing functions from a common set of networking services. However, all nodes do not necessarily perform the same functions at the same time. In particular, nodes may be assigned specific functions in the network, and these roles may change over time.

- Although the network should allow communication between any pair of nodes, it is envisioned that a large portion of the traffic will be among geographically close nodes. This assumption is clearly justified in a hierarchical organization. For example, it is much more likely that communication will take place between two soldiers in the same unit, rather than between two soldiers in two different brigades.

A MANET is a peer-to-peer network that allows direct communication between any two nodes, when adequate radio propagation conditions exist between these two nodes. If there is no direct link between the source and the destination nodes, multi-hop routing is used. In multi-hop routing, a packet is forwarded from one node to another, until it reaches the destination. Clearly, appropriate routing protocols are necessary to discover routes between the source and the destination, or even to determine the presence or absence of a path to the destination node. Because of the lack of central elements, distributed protocols have to be used. The main challenges in the design and operation of MANETs, compared to more traditional wireless networks, stem from the lack of a centralized entity, the potential for rapid node movement, and the fact that all communication is carried over the wireless medium. In standard cellular wireless
networks, there are a number of centralized entities (e.g., the Base Stations (BS), the Mobile Switching Centers (MSCs), the Home Location Register (HLR), and the Visitor Location Register (VLR)). In ad hoc networks, there is no preexisting infrastructure, and these centralized entities do not exist. The centralized entities in the cellular networks perform the function of coordination. The lack of these entities in MANETs requires distributed algorithms to perform these functions. In particular, the traditional algorithms for mobility management, which rely on a centralized HLR/VLR, and the medium access control (MAC) schemes, which rely on the BS/MSC support, become inappropriate. All communications between all network entities in ad hoc networks are carried over the wireless medium. Due to the radio communications being vulnerable to propagation impairments, connectivity among network nodes is not guaranteed. In fact, intermittent and sporadic connectivity may be quite common. Additionally, since the wireless bandwidth is limited, its use should be minimized. Finally, since some of the mobile devices are expected to be hand-held with limited power sources, the required transmission power should be minimized as well. Therefore, the transmission radius of each mobile is limited, and channels assigned to mobiles are typically spatially reused. Consequently, since the transmission radius is much smaller than the network span, communication between two nodes often needs to be relayed through intermediate nodes, i.e., multi-hop routing is used.

Because of the possibly rapid movement of the nodes and variable propagation conditions, network information such as a route table, becomes obsolete quickly. Frequent network reconfiguration may trigger frequent exchanges of control information to reflect the current state of the network. However, the short lifetime of this information means that a large portion of this information may never be used. Thus, the bandwidth used for distribution of the routing update information is wasted. For these reasons, the design of MANETs needs to allow for a high degree of reliability, survivability, availability, and manageability of the network.

Based on the above discussion, the following features for MANETs are required:

- **Robust routing and mobility management algorithms** to increase the network’s reliability and availability, e.g., to reduce the chances that any network component is isolated from the rest of the network;

- **Adaptive algorithms and protocols** to adjust to frequently changing radio propagation, network, and traffic conditions;
• **Low-overhead algorithms and protocols** to preserve the radio communication resource;

• **Multiple (distinct) routes** between a source and a destination to reduce congestion in the vicinity of certain nodes, and to increase reliability and survivability;

• **Robust network architecture** to avoid susceptibility to network failures, congestion around high-level nodes, and the penalty due to inefficient routing.

The ad hoc networking technology has stimulated substantial research activity in the past 10 years. The rather interesting fact is that although the military has been experimenting and even using this technology for the last three decades, the research community has been coping with the rather frustrating task of finding a ‘killer’ non-military application for ad hoc networks. A major challenge that has been perceived as a possible ‘show stopper’ for technology transfer is the fact that commercial applications do not necessarily conform to the ‘collaborative’ environment that the military communication environment does. In other words, why should a user forward someone else’s transmission, depleting her own battery power and, thus, possibly restricting her use of the network in the future? This question may relate to the issue of billing - if billing is possible (and, in fact, desirable), then nodes that serve as ‘good citizens’ could be rewarded. But billing is, by itself, a significant challenge in ad hoc networks.

Other challenges in deployment of ad hoc networks relate to the issues of manageability, security, and availability of communication through this type of technology. Realizing that technology transfer has not been the motivating factor, the recent research interest in ad hoc networks could be, most probably, attributed to the intellectual challenges that are part of this type of communication environment. Nevertheless, we believe that there is substantial commercial potential of ad hoc networks. Future extensions of the cellular infrastructure could be carried out using this type of technology and may be the basis for fourth generation (4G) of wireless systems. Other possible applications include sensing systems (also referred to as Sensor Networks) or augmentation to the wireless LAN technology.

### 4.3 A Promising Application: “Sensor Networks”

Sensor networks [12] differ from “traditional” ad hoc networks in many aspects. The number of nodes in a sensor network can be several orders of magnitude higher than in ad hoc networks, and the deployment of nodes is usually denser. Moreover, sensor nodes are limited in power,
computational capacities and memory, and they may not have global identification (ID) because of the very large number of nodes and the related overhead.

Due to the above constraints, sensor network protocols and algorithms must be endowed with self-organizing capabilities, i.e., sensors must be able to cooperate in order to efficiently perform networking tasks. The primary design constraints of these algorithms are energy efficiency, scalability and localization.

It has been recently pointed out [78] that energy efficiency in mobile systems can be improved by designing protocols and algorithms with a cross-layer approach, i.e., by taking into account interactions among different layers of the communication process so that the overall energy consumption can be minimized. Hence, interdependencies between physical and network layer functionalities to improve the energy efficiency of the routing tasks should be considered.

A primary requirement of configuration algorithms for large scale ad hoc networks, such as routing algorithms, is scalability, i.e., these algorithms should perform well for wireless networks with an arbitrary number of nodes. The notion of scalability is strictly related to that of localization: in a scalable algorithm each node exchanges information only with its neighbors (localized information exchange) [59]. In a localized routing algorithm, each node selects the next hop based only on the position of itself, of its neighbors, and of the destination node. As a result, the local routing decision of each node strives to achieve a global network objective such as minimum latency or minimum energy consumption. Conversely, in a non-localized routing algorithm a node maintains an accurate description of the overall network topology to select the next hop. This way, the routing problem is equal to the shortest path problem if the hop count is used as the global performance metric or the shortest weighted path if power [129] or cost [155][41] link metrics are used.

It has been shown [102][91] that routing protocols that do not use geographical location information are not scalable, e.g., AODV (Ad hoc On-demand Distance Vector), DSDV (Destination Sequenced Distance Vector) or DSR (Dynamic Source Routing). On the other hand, the recent availability of small, inexpensive and low-power GPS (Global Positioning System) receivers, together with techniques to deduce relative sensor coordinates from signal strengths [84] encourage researchers to develop Geographical Routing [144] (also Position Based Routing) algorithms, which are deemed to be the most promising solutions for critically power-constrained ad hoc networks.
In [112] the authors deal with the interdependencies between topology control [127] and energy efficient geographical routing. The question they try to answer is “How extensive should be the Local Knowledge of the global topology in each node, so that energy efficient geographical routing decisions can be taken?”. The answer to the question is clearly related to the “degree” of localization of the routing scheme. If each node could hold a complete vision (knowledge) of the network topology, it could then compute the “globally” optimal next hop, i.e., the neighboring node on the minimum energy path. However, acquiring increasingly accurate topology information has an increasing cost, i.e., the energy required to exchange the signaling traffic that conveys this information.

Hence, in [112] the authors develop an analytical framework to capture the trade-off between what they refer to as the topology information cost, which increases with increasing range of knowledge of each node, and the communication cost, which may decrease when the knowledge becomes more complete. They apply this analytical framework to different position based forwarding schemes [146][87][63][98][117] and show by Monte Carlo simulations that a limited knowledge is sufficient to make energy efficient routing decisions. With respect to the existing literature on geographical routing, they try to better define the terms “localized” and “neighbor”. A “neighbor” for a certain node is another node which falls into its Topology Knowledge Range, denoted as KR. To summarize, the main contributions of [112] are:

- Introduction of an analytical framework to evaluate the energy consumption of geographical routing algorithms for power constrained large scale ad hoc networks;
- Development of an Integer Linear Programming (ILP) formulation of the topology Knowledge Range optimization problem;
- Detailed comparison of the leading existing forwarding schemes and introduction of a new scheme called Partial Topology Knowledge Forwarding (PTKF);
- Introduction of a PRobe-bAseed Distributed protocol for knowledge rAnge adjustment (PRADA), that allows each node to efficiently select online its topology knowledge range. PRADA is shown to lead to near-optimal energy consumption.
4.4 Mobility Model Introduction

The movement pattern of users plays an important role in performance analysis of mobile and wireless networks. In cellular networks, for example, a user’s mobility behavior directly affects the signaling traffic needed for handover and location management (location updates and paging) \[58\]. The extra signaling messages over the air interface consume radio resources and increase the associated database query load. In addition, mobility has a major effect on the channel holding time in circuit-switched services \[85\][76][119][62\]. The latter has in turn huge influence on the call blocking and dropping probability \[105\]. The modeling of movement is thus an essential building block in analytical and simulation-based studies of these systems. Mobility models are needed in the design of strategies for location updating and paging, radio resource management (e.g., dynamic channel allocation schemes), and technical network planning and design (e.g., cell and location area layout, network dimensioning).

The choice of the mobility model has a significant effect on the obtained results. If the model is unrealistic, invalid conclusions may be drawn. With the increasing number of subscribers and the decreasing cell size in future cellular systems, the mobility pattern of users will even more influence the performance of the network. Models that proved to be a good choice in simulation of macro-cellular environments show some drawback when being applied in micro- and pico-cellular environments \[137\]. Mobility modeling also plays an important role in analysis of algorithms and protocols in wireless local area networks (WLANs) and self-organizing wireless ad hoc networks. Whereas in cellular networks there exists a number of approaches that model the macroscopic movement behavior of users (e.g., random walk from cell to cell, description of the cell residence time), in these cases we need a “microscopic” model.

4.5 Mobility Model Overview

There exists a variety of mobility models that find application in different kinds of simulations and analytical studies of wireless systems. Figure 38 shows a concept map illustrating some criteria which can be used for categorization. Analytical mobility models are, in general, based on rather simple assumptions regarding the movement behavior of users, but they allow the calculation of mathematical expressions with respect to system performance. Several authors derive the distribution of a user’s cell residence time \[85\][160][161][115\]. For example, in \[160\] it is shown that the cell residence time with their model can be described by a generalized gamma
distribution. Combining these “mobility metrics” with traffic models allows to estimate important system performance parameters, such as channel holding time and handover and location update events [85][76][160]. For example, in [85] the authors analyze the mean handover rate for voice calls. In [76] is derived a distribution for the channel holding time. In [160] is also derived a distribution for the channel holding time and the average number of handovers per call.

Let us briefly describe the mobility assumptions used by these authors. In [85] they assume that mobile users are uniformly distributed over a cell. Each user chooses a direction $\phi$ (taken from a uniform distribution in $[0, 2\pi]$) and a speed $v$ (uniformly distributed in the interval $[0, V_{\text{max}}]$). Once these values are chosen, they remain constant until the user crosses the boundary of the cell. In [76] it is exploited a more general model, in which direction changes are also possible within a cell. The model in [160] allows direction changes only to a certain extent ($\Delta \phi_{\text{max}}$).

The authors in [106][61] assume a generally distributed location area residence time as given. From this, they derive the probability distribution of the number of location area crossings for a given distribution of the interservice time (i.e., the time between the beginning of two unblocked calls). Another analytical model is the Brownian mobility model, which models the movement of users based on Brownian motion such that we can calculate the probability
distribution of the physical location of a user at given time $t$, provided that we know her location at a previous time $t_0 < t$. Models used for simulation-based studies describe movement of users in a more detailed manner. On the other hand, in general, they do not allow the derivation of analytical expressions.

For example, the European Telecommunications Standards Institute (ETSI) defined a set of test scenarios for system simulation of UMTS (Universal Mobile Telecommunication System). The document [60] describes mobility models for three environments: an indoor office, an outdoor pedestrian environment, and a vehicular environment. The model for the pedestrian environment uses a Manhattan-like street structure rectangular grid. Pedestrians walk along streets in a straight line and can change their direction at intersections with a given probability. Also speed changes are possible after given intervals. The model for the vehicular environment is a random mobility model with a street structure. Cars move with constant speed ($v = 120 \text{ km/h}$) and can change their direction every $20 \text{ m}$ (with a probability of $20\%$). Only direction changes of up to $\pm 45^\circ$ are possible.

In [37] the ETSI model is extended to get more realistic results. They analyze mobility-related parameters of their model such as the cell residence time and the cell boundary crossing rate, and compare it to the ETSI model.

Let us now consider the different levels of detail in mobility modeling (see Fig. 38). Researchers in vehicular traffic theory distinguish between three levels of description: microscopic, mesoscopic (kinetic), and macroscopic. A microscopic model describes the movement of a single vehicle by its space and speed coordinates at a given time $t$. Such approaches include very detailed “car following” models. At mesoscopic level, the homogenized movement behavior of several vehicles is reflected. For example, a distribution function is derived that describes the number of vehicles at a certain location $(x, y)$ or speed $v$ at time $t$. When modeling on a macroscopic scale, one is interested in high level statistical characterizations, e.g., the density, mean speed and speed variance, and traffic flow of vehicles. An example for a macroscopic movement model used in analysis of wireless systems is the fluid flow model [148]. This family of analytical models describes the mobility in terms of “the mean number of users crossing the boundary of a given area”. A second approach used for modeling the macroscopic movement behavior is the family of gravity models [99]. They are also derived from transportation theory. Such models give an aggregated description of the movement of several users (as the fluid model),
ranging from city scale to international scale. The authors in [109] describe such a model. They use the concept of trips, area and time zones, population groups, and so on. The paper [142] also falls into this category. It models the daily movement of users which follow an activity-based travel demand model.

Another frequently used approach in cellular networks is the family of random walk models, also denoted as Markovian mobility models. They describe movement of individual users from cell to cell. The exact location of a user is of interest but just the cell in which she resides. The model is basically defined by a state-transition diagram in which a cell is represented by a state and the movements between cells by transition probabilities between the states. A user either stays within her cell or moves to one of its neighboring cells with a certain probability. A typical random walk model in one dimension is described in [6]. Two dimensional random walk models are used in [10][47][141]. Recent enhancements include a random walk model presented in [11]. From these models, we can also derive analytical measures for the crossing rates of cell and location area boundaries.

Wireless researchers also invented models for three dimensional movement. The authors in [96] model user movements inside buildings, including vertical movements in staircases and elevators. There exists a variety of generalizations of the model in [76] which are used in simulation-based studies of wireless systems. Basically, this class of mobility models can be described by the following paradigm: users can move freely anywhere in the system area. The values for the user direction $\phi$ are uniformly distributed in $[0, 2\pi]$, i.e., users do not have any preferred direction. The speed values $v$ is usually assigned a uniform or a normal distribution. After a randomly chosen time, usually with an exponential distribution, a user chooses a new direction. The same procedure is performed for speed changes. The stochastic processes for direction and speed change are in general correlated to each other. A node is thereby completely described by its current space vector $(x(t), y(t))$, its current speed $v(t)$, and its current direction $\phi(t)$, where the following constraints generally hold: $0 \leq x(t) \leq x_{max}$, $0 \leq y(t) \leq y_{max}$, $0 \leq v(t) \leq V_{max}$, and $0 \leq \phi(t) \leq 2\pi \forall t$. This is also referred to as random direction mobility model.

In [80][121], it is used a simplified version of the random direction model. All users always have the same constant speed $v_o$ and move with an initial direction $\phi_o$ picked in the interval $[0, 2\pi]$ with a uniform distribution. When a user reaches the border of the simulation, it changes
its direction, “bouncing” back with a new direction angle $\phi = \pi - \phi_o$.

Another random mobility model is the random waypoint model, which is used by several authors in the ad hoc networking community [39][132][49][50][94]. According to this mobility model, a user randomly chooses a destination point in the system area, moves on a straight line to this point with constant speed $v$ (chosen in the interval $[V_{min}, V_{max}]$ with a uniform distribution), and pauses for a certain time before it again chooses a new destination. This model is very similar to a generalized random direction model. The difference is that the destination point is chosen rather than the direction $\phi$. A node is described by its current space vector $(x(t), y(t))$, its current speed $v(t)$, and its current destination point $(X_d(t), Y_d(t))$.

Let us now consider the “degree of randomness” different mobility models have (see Fig. 38). Basically, a mobility model belongs to one of the three following cases:

1. Users can move anywhere in the system plane following a pseudo-random process for speed and direction;

2. Users are bound by streets, buildings, and so on, but use a pseudo-random process for speed and direction choice at crossings;

3. Users are bound to a defined path.

We already gave many examples for the first case, the random waypoint and the random direction model being only two of them. ETSI in-house and pedestrian outdoor models with simple Manhattan-like street structure is an example for the second type. Such models are also described in detail in [109].

A deterministic approach for simulations would allow users to move only on a predefined mobility path, which is the third case for mobility models. Such a path can describe typical movement patterns of pedestrians and vehicles. We can further distinguish two cases. In the first case, the direction and speed are both given, and no random process is incorporated at all. In the second case, the direction trace is given but the speed is chosen randomly. Such traces can exist in different levels of detail (cells, areas, etc.). In [99] the authors describe a family of macroscopic mobility models based on traces. Nevertheless, in practice, tracing the actual mobility behavior of users is a very complicated and resource-draining task and usually such information is hard to obtain by network providers. Thus, researchers often use random
models. Moreover, privacy issues including the confidentiality of certain data, may prohibit the
collection and distribution of such statistics.

4.6 Cellular Mobility Models

Cellular mobility models focus their attention on individual movements since key issues often
involve paging and hand-offs of a single user or MN (Mobile Node). Rarely do more complicated
issues such as group management come into play. As a result, mobility models such as
the following were developed to test the behavior of cellular protocols and strategies:

1. Random Walk Model;

2. Constant Velocity Fluid-Flow Model;


4.6.1 Random Walk Mobility Model

The Random Walk Mobility Model has been proven to be one of the most widely used mobility
models because it describes individual movements relative to cells [161][6][133]. Many entities
in nature move in extremely unpredictable ways. Molecules, for example, move in random
directions, i.e., they do not yield a pattern or sense of direction [143]. The random walk was
developed in an attempt to mimic the erratic movements of certain objects. Specifically, in the
Random Walk Mobility Model, a host moves from its current location to a new location by
randomly choosing a direction and speed. The new speed and direction are both chosen from
pre-defined ranges, \([v_{\text{min}}, v_{\text{max}}]\) and \([0, 2\pi]\) respectively. Each movement in the Random Walk
Mobility Model occurs in a constant time interval, at the end of which a new direction and speed
are calculated. Many derivatives of the Random Walk Mobility Model have been developed
including the one-dimensional, two-dimensional, three-dimensional, and d-dimensional walks.
We mention only the 1-D and 2-D walks, since the 3-D and d-D walks may easily be extrapolated
from those.

In a 1-D walk, we imagine a gymnast standing in the middle of an infinitely long balance
beam. Given the results of a coin flip, the gymnast moves in a particular direction at a random
speed for a certain amount of time. For example, if the coin flip results in heads, the gymnast
moves to the right at the randomly chosen speed. In contrast, if the coin flip results in tails, the
gymnast moves to the left. After repeating this pattern for a large number of times, a 1-D walk is mapped.

In a 2-D walk, we visualize the same gymnast moving on a planar surface. For example, using a similar method as that mentioned in the 1-D walk, we generate a 2-D random walk. Specifically, instead of visualizing a gymnast on a balance beam we expand our environment to include an infinite floor. Instead of flipping a coin, the gymnast uses a spinning dial. After spinning the dial, the gymnast moves in the direction pointed to by the needle at a random speed for a certain amount of time. In doing so, the gymnast randomly moves around a 2-D surface, thus creating a 2-D walk.

In 1921 Polya proved that a random walk on a one- or two-dimensional lattice returns to the origin with complete certainty, i.e., with a 1 probability [153]. This characteristic ensures that the random walk accurately represents a mobility model that emulates the movements of entities around their starting points. The two-dimensional walk is of special interest to researchers developing protocols for cell-related technologies, since the Earth’s surface is modelled using a two-dimensional representation. Unfortunately, the simplicity of the Random Walk Mobility Model is not always sufficient to produce realistic results. Figure 39 shows an example of the movement observed using a 2-D model. The Mobile Node (MN) begins its movement at position (0,0). At each point, the MN randomly chooses a direction between 0 and $2\pi$ and a speed between 0 and 10 m/s. The MN is allowed to travel for a total of 1 second before changing direction and speed. It is easy to note the restricted behavior of the Random Walk Mobility Model, proven by Polya. The node illustrated in Fig. 39 rarely travels far from the origin.

In a special case of the Random Walk Mobility Model, a MN no longer travels for a constant time period before changing direction. Instead, a MN changes direction after travelling a specified distance. Figure 40 illustrates an example of the movement observed according to this modified mobility model. We note that the figure only illustrates a small region of a $200 \times 200$ simulation area. A MN begins its movement in the middle of the simulation area (e.g., at position (0,0)). At each point the MN randomly chooses a new direction between 0 and $2\pi$ and a speed between 0 and 10 m/s. The MN travels for a total of 10 units before changing its direction and speed.

The Random Walk Mobility Model is a **memoryless** mobility pattern because it retains no
Figure 39: Travelling pattern of a Mobile Node (MN) using the 2-D Random Walk Mobility Model

Figure 40: Travelling pattern of a Mobile Node (MN) using the modified 2-D Random Walk Mobility Model
knowledge concerning its past locations and speed values. This characteristic inhibits the practicality of the Random Walk Mobility Model because MNs typically have a pre-defined destination and speed in mind, which in turn affects future destinations and speeds. Other mobility models such as the Random Gauss-Markov Mobility Model attempt to fix this discrepancy.

4.6.2 Fluid-Flow Mobility Model

Fluid mobility models describe macroscopic movements instead of individual or microscopic movements. The behavior of the generated traffic is similar to a fluid flowing through a pipe. As a result, the Fluid-Flow Mobility Model best represents traffic on highways and other similar situations with a constant flow of MNs. In other words, the model is unable to accurately represent the movements of individual MNs. As an example, a deterministic Fluid-Flow Mobility Model is used in [101] to represent the behavioral characteristics of traffic on a one-way, semi-infinite highway. Cars enter and exit the highway at various locations. Haas and Liang [79] confirm that the Fluid-Flow Mobility Model is insufficient for individual movements including stopping and starting, actions commonly associated with an individual walking around town or from class to class.

4.6.3 Random Gauss-Markov Mobility Model

The Random Gauss-Markov Mobility Model was introduced in order to circumvent the undesirable results mentioned above with reference to [79]. In the Random Gauss-Markov Mobility Model, the velocity of a MN at time \( n \) is given by the equation

\[
\nu_n = \alpha \nu_{n-1} + (1 - \alpha) \mu + \sqrt{1 - \alpha^2} \cdot x_{n-1}
\]

(67)

where \( \alpha \) is the tuning parameter used to vary the randomness, \( \mu \) is a constant representing the mean value of \( \nu_n \) as \( n \to \infty \), and \( x_{n-1} \) is a random variable from a Gaussian distribution. Totally random values are obtained by setting \( \alpha = 0 \) and linear motion is obtained by setting \( \alpha = 1 \). Intermediate levels of randomness may be obtained by varying the value of \( \alpha \) between 0 and 1.

The displacement of a MN is given by the equation

\[ s_n = \sum_{i=0}^{n-1} \nu_i \]

(68)

By allowing past velocities and directions to influence future velocities and directions, the Random Gauss-Markov Mobility Model solves the problems encountered in the Random Walk Mobility Model. This way, it also allows studying individual MN movements, thus overcoming the
problems encountered in the Fluid-Flow Mobility Model.

### 4.7 Ad Hoc Mobility Models

In ad hoc wireless mobile networks, the mobility models focus on the individual motion behavior between mobility epochs, which are the smallest time periods in a simulation in which a mobile host moves in a constant direction at a constant speed. Many researchers use the Random Mobility Model [161][138]. According to this model, the speed and direction of motion in a new time interval have no relation to their past values in the previous epoch. This model can generate unrealistic mobile behavior such as sharp turning or sudden stopping. Some authors use modified versions of the random mobility model. In [23], the author describes the movement of MNs in their simulation for the DREAM (Distance Routing Effect Algorithm for Mobility) protocol such that a MN has a random direction at every simulation clock tick, but a constant speed during the entire simulation period. The mobility model in [97] for the LAR (Location-Aided Routing) routing protocol allows MNs to move along a path which is made up of several segments. The segment lengths are exponentially distributed and the direction of each segment is randomly chosen. Speed is uniformly distributed in $[v - \alpha, v + \alpha]$.

In the model proposed in [48], a node chooses its speed, direction and distance based on a pre-defined distribution, then calculates its next destination and the time to reach it. When the node reaches that point, it calculates a new destination and time period to reach it again. The Random Waypoint mobility model [95] is also an extension of random walk. This model breaks the entire movement of a MN into repeating pause and motion periods. A mobile host first stays at a location for a certain time, then it moves to a new random-chosen destination at a speed uniformly distributed in $[0, v_{\text{max}}]$.

In [43], a Markovian model is used to describe the random motion. States represent motion directions. The probability of maintaining the current state (or moving to another state) is specified in the transit matrix. Once in motion, the MN is more likely to keep on going at the current direction and speed. This model is more realistic than the random model. Similar to [43], other models [161][77] consider the relationship between a mobile host’s previous motion behavior and the current movement characteristics such as speed and/or direction. In particular, the author in [77] presents an incremental model in which, after each time increment, speed and direction of current movement randomly diverge from the previous speed and direction, respectively. In
particular, speed $v(t)$ and direction $\theta$ are expressed as below,

$$v(t + \Delta t) = \min[\max[v(t) + \Delta v; 0]; v_{max}]$$

(69)

$$\theta(t + \Delta t) = \theta(t) + \Delta \theta$$

(70)

where $\Delta v$ and $\Delta \theta$ are uniformly picked in reasonable ranges, e.g., $[-A_{max}\Delta t, A_{max}\Delta t]$ and $[-\alpha \Delta t, \alpha \Delta t]$, respectively, being $A_{max}$ the unit acceleration/deceleration and $\alpha$ the maximal unit angular change.

Further, in [138], the author studies the relationship among MNs. This relationship exists while MNs move with the same purpose. As can be noticed in a disaster recovery mission, or in a military setting, several mobile hosts most likely move with a common objective. Two examples given in [138] are the Pursue model, where MNs try to move towards a target, and the Column model, which represents a searching activity. These models try to capture the interdependencies among nodes in their movements, whereas most existing cellular and ad hoc wireless mobility models describe independent node motion behavior.

Ad hoc mobility models differ from cellular mobility models in the network they model. Cellular mobility models require the use of BSs whereas ad hoc mobility models require the cooperation of two or more communicating MNs. Although separate mobility models exist for cellular and ad hoc mobility models, similarities exist between the two categories. In the following of this section we describe the most popular ad hoc mobility models.

4.7.1 Random Mobility Model

The Random Mobility Model for ad hoc networks coincides with the Random Walk Mobility Model for cellular networks. In the Random Mobility Model, the current speed and direction of a MN are both random variables that have no correlation with their current values [86]. Thus, we encounter an unrealistic choppy motion with sudden stopping, sharp turning, and completely random wandering. In order to avoid these problems, many authors modified the Random Mobility Model by changing the calculation of speed, direction, or both.

4.7.2 Constant Velocity Random Direction Mobility Model

Constant Velocity Random Direction Mobility Model [23][86] revised the Random Mobility Model to ensure that every node is assigned the same speed throughout the entire simulation. After a random direction is chosen in the range $[0, 2\pi]$, a MN begins moving. If the MN reaches
a grid boundary, it bounces on the simulation border with an angle determined by the incoming
direction. The MN then continues along this new path.

### 4.7.3 Random Waypoint Mobility Model

The Random Waypoint (RWP) Mobility Model used in [95] includes pause times between
changes in direction and/or speed. In this stochastic model, each node of the network randomly
chooses a destination point, i.e., ‘waypoint’, in a rectangular deployment region $Q$. A node
moves to this destination with a velocity $v$ uniformly chosen in $[v_{\text{min}}, v_{\text{max}}]$. When it reaches
the destination, it remains static for a predefined pause time $t_p$ and then starts moving again
according to the same rule.

It has been observed in several papers [132][27][29][31][36] that the spatial distribution of
nodes moving according to the RWP model is non uniform. Although the initial node positioning
is typically taken from a uniform random distribution, the mobility model changes this dis-
tribution during the simulation. This effect, known as border effect [27], occurs because nodes
tend to cross the center of $Q$ with a relatively high frequency. For a long running time of the
movement process, the stochastic distribution of the nodes converges toward an asymptotically
stationary distribution with the maximum node density in the middle of $Q$. The non-uniformity
of the RWP node distribution has important practical consequences.

First, it reduces the applicability of existing analytical results concerning ad hoc networks,
which are typically based on the uniformity assumption. For example, theoretical results with
respect to routing [16][116], capacity [71][74], connectivity [28][75][140], and minimum power
issues cannot be applied directly in a mobile scenario that employs the RWP model.

Second, the nonuniform distribution implies that the representativeness of the huge amount
of simulation results obtained by using the RWP model could be impaired. This is because the
short-term behavior of the RWP model is quite different from the actual long-term behavior. In
[122] a modified version of the Random Waypoint Mobility Model is proposed so that a MN
travels at a constant speed throughout the entire simulation.

In [30], a generalized version of the RWP model is proposed. In this generalized model, a
node may remain static for the entire simulation time with a given probability. Hence, only a
fraction of the nodes are expected to move. Also, the model considers the fact that nodes are
initially deployed according to an arbitrary spatial distribution. The pause time $t_p$ is allowed to
Figure 41: Travelling pattern of a Mobile Node (MN) using the Random Waypoint Mobility Model

be different after each movement period. Figure 41 shows an example of travelling pattern of a MN using the Random Waypoint Mobility Model starting at a randomly chosen point. We note that the movement pattern of a MN using the Random Waypoint Mobility Model is similar to the Random Walk Mobility Model if pause time is zero and the velocity interval ranges coincide, i.e., \([0, v_{\text{max}}] = [v_{\text{min}}, v_{\text{max}}]\).

4.7.4 Random Direction Mobility Model

The Random Direction Mobility Model [131] was developed in order to overcome a flaw discovered in the Random Waypoint Mobility Model. MNs using the Random Waypoint Mobility Model often choose a new destination that is located in the center of the simulation area, or that requires travelling through the middle of the simulation area. In [131] it is stated that MNs moving with the Random Waypoint Mobility Model appear to converge, disperse, converge again, etc. In order to alleviate this type of behavior and promote a semi-constant number of neighbors, the Random Direction Mobility Model was developed. In this model, MNs choose a random direction rather than a random destination. After choosing a random direction, a MN travels to the border of the simulation area in that direction. As soon as the boundary is reached the MN
Figure 42: Travelling pattern of a Mobile Node (MN) using the Random Direction Mobility Model

stops for a certain period of time, chooses another angular direction in $[0, \pi]$ and continues the process. Figure 42 shows an example path of a MN, which begins at the center of the simulation area, i.e., at (250, 250), using the Random Direction Mobility Model. A slight modification to the Random Direction Mobility Model is the Modified Random Direction Mobility Model [131]. In this modified version, MNs continue to choose random directions but they are no longer forced to travel to the simulation boundary before stopping to change direction. Instead, a MN chooses a random direction and selects a destination anywhere along that direction of travel.

4.7.5 A Probabilistic Version of the Random Mobility Model

In [43], a mobility model which utilizes a probability matrix to determine the position of a particular MN in the next time step is described. Three different states are introduced. State 1 represents the current location of a given MN, state 2 represents the MN’s previous location, and state 3 represents the MN’s next location if the MN moves forward. The probability matrix
used is

$$P = \begin{pmatrix}
P(1, 1) & P(1, 2) & P(1, 3) \\
P(2, 1) & P(2, 2) & P(2, 3) \\
P(3, 1) & P(3, 2) & P(3, 3)
\end{pmatrix}$$  \hspace{1cm} (71)$$

where each entry $P(a, b)$, represents the probability that a MN will go from state $a$ to state $b$. In [43] each node moves randomly with a preset average speed. The following matrix contains the values used in the paper to calculate $x$ and $y$ movements:

$$P1 = \begin{pmatrix}
0 & 0.5 & 0.5 \\
0.3 & 0.7 & 0 \\
0.3 & 0 & 0.7
\end{pmatrix}$$  \hspace{1cm} (72)$$

Probability matrix $P1$ allows a MN to move in any direction as long as it does not return to its previous position. This implementation produces probabilistic rather than purely random movements, which may yield more realistic behaviors. For example, as people complete their daily tasks, they tend to continue moving in a semi-constant forward direction. However, choosing appropriate values of $P(a, b)$, may be difficult, if not impossible, for individual scenarios.

4.7.6 Smooth Random Mobility Model

In [27] the authors employ a combination of principles for direction and speed control that make the movement of users (e.g., pedestrians and cars) smoother and more realistic than in previously known random models. Smooth Random Mobility Model is a random mobility model for movement in two dimensions on a microscopic scale. A new destination is chosen by choosing a new direction $\phi$. The speed and direction changes are both probabilistic. The movement of nodes is not bounded by physical structures (such as streets, buildings, etc.), i.e., nodes are allowed to move anywhere in the simulation plane. Furthermore, there is no correlation between different nodes. This implies that effects like ‘node following’ or ‘group movement’ are not modelled. In [27] two stochastic processes are used: one process determines at what time a mobile station changes its speed, and the other process determines when the direction is changed.

Mobility models where the new choice for speed $v$ and direction $\phi$ is not correlated to previous values (such as in random waypoint model) may provide unrealistic movement behavior with sudden speed changes and sharp turnings. Smooth Random Mobility Model includes both autocorrelation features. The speed is changed incrementally by the current acceleration of the mobile user, and also the direction change is smooth: the direction is changed in several
steps until the new target direction is achieved. This creates a smooth curve rather than a sharp turning.

In a simulation the model proceeds as follows: at the beginning, all nodes are created with an initial speed \( v(t=0) \), which is chosen from a certain speed distribution \( p(v) \). It is used a distribution in which the preferred speed values have a high probability, and uniform distribution is assumed on the entire interval \([0, V_{\text{max}}]\). When a speed change event occurs, a new target speed \( v^* \) is chosen from a set of preferred velocities. Every node belongs to a class which corresponds to a set of speeds that characterizes the movement behavior of the node (e.g., pedestrian, vehicles, etc.). Each class has also different values for the maximum possible acceleration and maximum possible deceleration. A new acceleration or deceleration is chosen to reach \( v^* \) and a new speed \( v(t) \) is calculated according to:

\[
v(t) = v(t - \Delta t) + a(t) \cdot \Delta t
\]  

(73)

where \( \Delta t \) is the interval between two time steps. The principle for direction control is similar to the speed control principle. Each node has an initial direction \( \phi(t=0) \) which is chosen from a uniform distribution. A stochastic process decides when to change direction: a new target direction \( \phi^* \) and a new increment \( \Delta \phi^* \) are chosen, so that the new direction is computed as follows,

\[
\phi(t) = \phi(t - \Delta t) + \Delta \phi^*
\]  

(74)

4.7.7 City Area, Area Zone, and Street Unit Mobility Models

In [109], the authors take an in-depth look at desirable characteristics of mobility models including required inputs and outputs. They represent a basic mobility model with a set of input parameters \( S_{\text{in}} \) and a set of output parameters \( S_{\text{out}} \): i) \( S_{\text{in}} \) includes a population \( P \), which represents specific groups of MNs, a geographical area \( G \) organized into regions, and a time period \( T \), and ii) \( S_{\text{out}} \) includes a collection of functions that determine the location of a MN \( p \) over the set \( G \) at time \( t \).

The authors in [109] are inspired by the Transportation Theory, which is aimed at determining the load a system should carry, given a geographical area of service. In order to calculate a given load, many different variables are considered: the purpose of a trip, the exact route taken including starting and ending points, population groups such as students and/or working people,
periods of high MN activity, a transportation system’s capacity and usage costs, and popular areas attracting many MNs.

By combining these elements with Transportation Theory, in [109] three mobility models are developed: the *city area*, *area zone*, and *street unit* models.

The *city area* model can be viewed as a representation of user mobility and traffic behavior within a large-scale geographical area. A typical city area model possesses two key characteristics according to transportation theory. First, cities usually develop in such a way that the center of the city comprises a high concentration of workplaces and businesses. Surrounding the center of the city is a fairly dense distribution of dwelling areas for the people of the city, which are commonly referred to as urban areas. As we move away from the center of the city, we see a gradual decrease in population density, thus representing suburban and rural areas. The second key characteristic found in a typical city is a street network that supports movements from the center of the city, through urban areas, into the suburban and rural areas. Obviously, the focus in the city area model is to represent large-scale flows of traffic within city limits.

The *area zone* model takes a slightly more refined look at mobility within a city. Instead of looking at the entire city, the area zone model divides the city into regions. This process is done using square-shaped building blocks and an orthogonal grid representing a street network.

Finally, the *street unit* model attempts to model movements of individual MNs. The authors attempt to simulate realistic traffic conditions by minimizing the travelling time for all MNs and implementing safe driving characteristics such as a speed limit and a minimum distance allowed between any two MNs.

The city area, area zone, and street unit models lack specific details, such as calculations for the movements of MNs, because they are theoretical models used to describe simulation environments. These theoretic models create realistic simulation environments that introduce obstacles and strictly defined travel paths. Furthermore, when combined, the city area, area zone, and unit street models create an even more realistic simulation environment. Unfortunately, this high level of accuracy introduces an overwhelming amount of computational effort and complexity in the simulation.
4.7.8 The Metropolitan, National, and International Mobility Models

The Metropolitan, National, and International Mobility Models described in [99] focus on a different range of movement.

The Metropolitan Mobility Model (METMOD) focuses on movements of MNs in a metropolitan area. The geographical region is divided into smaller regions or subsets. A movement connectivity matrix is created and used to describe the probability of a MN moving into an adjacent area within the metropolitan area. Each element \((x, y)\) within this matrix represents the probability of a MN in area \(x\) moving into area \(y\).

The National Mobility Model (NATMOD) models the behavior of MNs moving between metropolitan areas. As in the METMOD case, the NATMOD model divides the entire simulation region into smaller geographical areas; however, these subsets are now entire metropolitan areas. The authors assume that the most popular mode of travel between metropolitan areas is by aircraft. Thus, they use flight information from the Department of Transportation, distances between major metropolitan airports, and the assumption that an equal number of flights exist to and from each airport to derive a traffic volume for the mobility model.

Finally, the International Mobility Model (INTMOD) describes the behavior of MNs travelling between the United States and foreign countries. Data from the United States and other countries were compiled in an attempt to create an accurate traffic volume.

4.7.9 Group Mobility Models

So far we have discussed mobility models that represent multiple MNs whose actions are completely independent of each other. However, in many situations it is necessary to model the behavior of MNs that move together. For example, in many military scenarios a group of soldiers must collectively search a particular plot of land in order to destroy land mines, capture enemy attackers, or simply work together in a cooperative manner to accomplish a common goal. In order to model such situations, group mobility models exist to account for these new cooperative characteristics.

One of the first examples of group mobility is the Exponential Correlated Random (ECR) model proposed in [26]. The model reproduces all possible movements, including individual and group movements, by adjusting the parameters of a motion function. According to this model, the new position \(\vec{b}(t + 1)\) is a function of the previous position \(\vec{b}\), to which a random
deviation \( \bar{r} \) is added,
\[
b(t + 1) = b(t)e^{-\frac{1}{\tau}} + (\sigma\sqrt{1 - e^{-\frac{2}{\tau}}})r
\]
(75)

where: \( b(t) = (r_t, \theta_t) \) is defined for a group or a node at time \( t \); \( \tau \) adjusts the rate of change from old to new (small \( \tau \) causes large change); \( r \) is a random Gaussian variable with variance \( \sigma^2 \). The parameters \( \tau \) and \( \sigma \) drive the groups into different moving patterns, and vary from group to group. The ECR mobility model requires a complete set of \( (\tau, \sigma) \) (one per group) to define the motion of the entire network. The drawback is that it is not easy to force a given motion pattern by selecting the parameters.

In the following we consider two main families of group mobility models and their most popular variations:

1. Simple Group Mobility:
   - Column Mobility Model,
   - Pursue Mobility Model,
   - Nomadic Community Mobility Model.

2. Reference Point Group Mobility Model:
   - In-Place Mobility Model,
   - Overlap Mobility Model,
   - Convention Mobility Model.

4.7.9.1 Simple Group Mobility

In [138] it is pointed out that a random walk movement model may not be sufficient to describe many ‘real-life’ situations. Typically, we should account for dependencies resulting from the interactions between MNs. The Column, Pursue, and Nomadic Community Mobility Models were thus developed [138][139].

The Column Mobility Model. This model is useful for scanning or searching purposes, and represents a set of MNs that have formed a line and are uniformly moving forward in a particular direction. For example, consider a row of soldiers marching together towards their enemy. Each soldier stands next to his/her companions while marching in a uniform manner. A slight
modification of the Column Mobility Model allows the individual MNs to follow one another. For example, consider a group of young children that walk in a single-file line to and from their classroom. In [138] the author describes a version of the Column Mobility Model in which individual MNs are placed in a single-file line and are allowed to move from their initial positions. This process begins by calculating a new reference position for a MN using the equation,

\[
\text{new reference position} = \text{old reference position} + \text{advance vector} \tag{76}
\]

where old reference position is a static variable indicating the initial location of a MN and advance vector is a predefined offset. A second equation is used to calculate the new position of a MN,

\[
\text{new position} = \text{new reference position} + \text{random vector} \tag{77}
\]

where random vector is a random offset and new reference position is the current point of reference. In order to determine their final positions, MNs calculate the sum of their new position with a random vector.

In Fig. 43, three MNs are initially lined up in the lower left-hand corner. The MNs begin moving by travelling six units to the right and five units up, as specified by their advance vector: (6, 5).

Figure 44 illustrates the military example with 12 MNs. The MNs start at the lower left-hand corner of the simulation area and travel towards the upper right-hand corner.
Pursue Mobility Model. In [138] the Pursue Mobility Model is also described. As the name implies, the Pursue Mobility Model attempts to represent MNs tracking a particular target. For example, this model represents police officers attempting to catch an escaped criminal or a swarm of bees attempting to attack a careless camper who inadvertently disturbed their dwelling. The Pursue Mobility Model consists of a single update equation for the new position of each MN,

\[ new\_position = old\_position + acceleration + random\_vector \]  

(78)

The current position of a MN, a random vector, and an acceleration function are combined to calculate the next position of the MN. The acceleration function is used to allow only a limited maximum step in each new movement, while the random vector ensures the random motion of each MN.

Figure 45 illustrates the movements of MNs using the Pursue Mobility Model. The solid black node represents the node being pursued and the white nodes represent the pursuing nodes.

Nomadic Community Mobility Model. A Nomadic Community Mobility Model is useful for representing both military and agricultural situations [138]. Just as ancient nomadic societies moved from location to location, this model represents groups of MNs that collectively move from one position to another. Within each community, or group of MNs, individuals maintain their own personal ‘spaces’ where they move in random ways. This is similar to bedrooms within a house. Each member of the household is assigned their own bedroom where they move in semi-random patterns, yet all household members reside in close proximity to each other.
Many situations may benefit from this type of behavior, in particular agricultural scenarios. Consider a large plot of land divided into smaller subsets that are maintained by a single entity. After each entity finishes tending a particular subset of land, the entire group moves to another large plot of land, divides it into individual subsets, and continues the process. The position update equation for the Nomadic Community Mobility Model is,

\[ new\_position = reference\_position + random\_vector \] (79)

where \( reference\_position \) marks the center of an individual’s plot of land and \( random\_vector \) ensures the random motion of each MN.

The Nomadic Community Mobility Model allows individual movements as long as a MN remains located near the reference position. In other words, the random vector defines how near the MN must be to the reference position. Pursue Mobility Model offers greater flexibility in the placement of a MN.

Figure 46 illustrates the movements observed using the Nomadic Community Mobility Model. The figure illustrates the rigid movements associated with the Nomadic Community Mobility Model as the group travels from one location to another.

4.7.9.2 Reference Point Group Mobility Model

The Reference Point Group Mobility (RPGM) model was developed in [86]. In order to represent the group mobility behavior of the mobile nodes, for each mobility group, the model defines a logical reference center whose movement is followed by all nodes in the group, i.e., the RPGM model describes the group membership of a mobile node by its physical displacement from the
Figure 46: Travelling pattern of Mobile Nodes (MNs) using the Nomadic Community Mobility Model

group reference center. For example, at time $t$, the location of the $i^{th}$ node in the $j^{th}$ group is given by

- Reference location $Y_j(t)$,
- Local displacement $Z_{j,i}(t)$, and
- Node location $X_{j,i}(t) = Y_j(t) + Z_{j,i}(t)$.

The node-dependent local displacement or random motion vector, $Z_{j,i}(t)$, models the effect of the mobile nodes having their own localized movements while following the general group motion defined by the reference center. The RPGM model can effectively generate topologies of ad hoc networks with group-based node mobility for simulation purposes. On the other hand, for mobility or partition prediction purposes, it has two main disadvantages.

First, this model uses an omniscient observer or a ‘God’, where the complete information about the mobility groups including their member nodes and movements are known. Given the distributed nature of the ad hoc network, such global information about the mobility groups are not conveniently available to any mobile nodes. For example, a MN travelling to a destination does not know all the other users that are heading in the same direction. Therefore, the lack of prior knowledge about the mobility groups makes the RPGM model inapplicable for run-time partition prediction.

Second, the RPGM model represents the mobile nodes by their physical coordinates. Given only the instantaneous physical locations of the nodes, it is difficult to discern the nodes’ group
movement patterns and the trend in the network topology changes.

The center motion defines the entire group’s motion behavior, including location, speed, direction, acceleration, etc. Thus, the group trajectory is determined by providing a path for the center. Usually, nodes are uniformly distributed within the geographic scope of a group. Each node is then assigned a reference point which follows the group movement. A node is randomly placed in the neighborhood of its reference point at each step. The reference point scheme allows independent random motion behavior for each node, in addition to the group motion. Each group has a group motion vector $V_{gi}$.

Figure 47 gives an example of a two-group model. In particular, it depicts how a node moves from time tick $\tau$ to $\tau + 1$. First, the reference point of a node moves from $RP(\tau)$ to $RP(\tau + 1)$ with the group motion vector $GM$ ($GM = V_{gi}$). Then, the new node position is generated by adding a random motion vector $RM$ to the new reference point $RP(\tau + 1)$. Vector $RM$ has its length uniformly distributed within a certain radius centered at the reference point and its direction uniformly distributed in $[0, 2\pi]$. This random vector $RM$ is independent from the node’s previous location.
The RPGM model defines the motion of groups explicitly by giving a motion path for each group. A path which a group will follow is given by defining a sequence of check points along the path corresponding to given time intervals. As time goes by, a group moves from one check point to the next, on a continuing basis. Each time the group center reaches a new check point, it computes the new motion vector $\vec{V}_{gi}$ from current and next check point locations and from the time interval. By proper selection of check points, one can easily model many realistic situations, where a group must reach predefined destinations within given time intervals to accomplish its task. The check point scenario file has the advantage of decoupling the motion pattern from the model itself. Many methods can be used to generate a scenario file such as typing in manually, digitizing a route from a map, using outputs from a program or a profile from real world. Moreover, the model has the advantage of providing a general and flexible framework for describing mobility patterns.

The RPGM model was designed to depict scenarios such as an avalanche rescue. During an avalanche rescue, the responding team consisting of human and canine members work cooperatively. The human guides tend to set a general path for the dogs to follow, since they usually know the approximate location of victims. The dogs create their own random paths around the general area chosen by their human counterparts. If appropriate group paths are chosen, along with proper initial locations for various groups, many different mobility applications may be represented with the RPGM model. By proper selection of check point path, initial group location and parameters in the RPGM model, it is easy to model various mobility applications.

We illustrate in the following the use of RPGM in a few representative cases. The first model is a geographical partition model (see Fig. 48).

The entire area is divided into several adjacent regions, with a different group in each region. This model can be used to emulate a battlefield situation, where different battalions are carrying out same operations (e.g., land mine search) in different areas. Each group is in charge of one partition. Another application can be large scale disaster recovery, where different paramedic, police, firemen teams work in separated neighborhoods. This model is referenced to as an In-Place Group Model. Figure 48 gives an example of four groups working in four adjacent areas, with different motion patterns. The second model describes an overlapped operation. Different groups carry out different tasks over the same area. However, the distinct requirements of each task make their mobility pattern quite different. For example, in a disaster recovery area, the
rescue team, the medical assistant team and the psychologist team will be randomly spread out over the area. Yet, each group has a unique motion pattern, speed, scope, etc. In Fig. 49, there are two groups working in the same area. This is the Overlap Mobility Model.

The third model is a convention scenario. It models the interaction between exhibitors and attendees. In a convention, several groups give demos of their research projects/products in separate but connecting rooms. A group of attendees roams from room to room. They may stop in one room for a while and then move on to another room. Or, they may pass through one room quickly. Figure 50 shows a group of attendees roaming around four exhibit rooms. This is called the Convention Model.

Other more complex scenarios which can be modelled with RPGM include:

- A military maneuver with joint aircraft, tank and infantry operations, where each asset has a different mobility pattern which can be handled with check point path profiles;

- A two-level mobility model, for example infantry and helicopters, with slow and fast nodes, etc.

A road map can be easily translated into RPGM check point format. Thus, mobility on highways can also be modeled by RPGM. For example, the Convention Model could also be used to reflect the roaming behavior of drivers on a road network.
Figure 49: Overlap Mobility Model

Figure 50: Convention Mobility Model
In-Place Mobility Model. The In-place Mobility Model is used to partition a given geographical area. Each subset of the original area is assigned to a specific group, which operates only within that geographic subset. This model is useful for simulating situations in which groups of people, who have similar goals, are assigned to limited areas. For example, in [86] this model is suggested for “large scale disaster recovery, where different paramedic, police, and firemen teams work in separated neighborhoods”. Each neighborhood is assigned its own paramedic, police and firemen teams, who never interact with the rescue personnel of nearby neighborhoods.

Each group of MNs moves in conjunction with $G$. After updating their individual $R$, each MN calculates its new position by calculating $RM$ and summing that result with their updated $RP$. Figure 48 illustrates this model. Within the simulation area we see four distinct groups of MNs. Each MN within a single group may participate in an activity different from every other MN in the group. However, the MNs work together in a specific area and leave the rest of the simulation area to be tended by other groups.

Overlap Mobility Model. The second variation of the RPGM model is the Overlap Mobility Model. The Overlap Mobility Model simulates several different groups, each of which has a different purpose, working in the same geographic region. Each group within this model may have different characteristics than other groups within the same geographical boundary. For example, in disaster recovery, one might encounter a rescue personnel team, a medical team, and a psychologist team, each of which have unique travelling patterns, speeds, and behaviors.

In the Overlap Mobility Model, the entire town or disaster area shares the various rescue teams. Each MN within a single group participates in an identical task and each group is responsible for providing their talents to the entire simulation area. Figure 49 illustrates two groups of MNs using the Overlap Mobility Model. The square symbol group follows a $GM$ directed towards the upper left-hand corner while the rhomboidal symbol group follows a $GM$ directed toward the right simulation border. $RP$s and $RM$s of individual MNs are updated as described above.

Convention Mobility Model. The last variation of the RPGM model, described in [86], is the convention scenario. In this scenario, both the conference attendees and the exhibits are represented. According to this scenario, different exhibits are housed in different rooms, which are connected to offer travel between exhibits. Similarly, the Convention Mobility Model divides
a given area into smaller subsets and allows the groups to move in a similar pattern throughout each subset. Again, some groups travel faster than others within a subset. Therefore, the Convention Mobility Model allows one group of attendees to take their time at an interesting exhibit, while others move quickly past a less interesting exhibit. Figure 50 shows an example for the Convention Mobility Model. In this example, the simulation area is divided into several sections or exhibits. Participants at a convention set up will tend to travel from exhibit $A$ to $B$ to $D$ to $C$ as their respective $GMs$ specify. Some groups may spend a significant amount of time at exhibit $B$ and little time at exhibit $D$. All groups, however, will travel in a semi-regular pattern around the room, and all MNs will travel within each exhibit as dictated by their $RP$s and $RM$s.

### 4.7.9.3 Reference Velocity Group Mobility Model

The authors in [150] extend the RPGM model by proposing a velocity representation of the mobility groups and the mobile nodes: each mobility group has a characteristic group velocity. They observe that instead of proximity in physical displacements, a more fundamental characteristic of a mobility group is the similarity of the member nodes’ movements. The node movement can be characterized by the velocity $v = (v_x, v_y)^T$ where $v_x$ and $v_y$ are the velocity components in the $x$ and $y$ directions. The member nodes in the group have velocities close to the characteristic group velocity but slightly deviated from it. Hence, the characteristic group velocity is also the mean group velocity. The membership of the $i^{th}$ node in the $j^{th}$ group is then described as

- **Group velocity** $W_j(t) \sim P_{j,i}(w)$,
- **Local velocity deviation** $U_{j,i}(t) \sim Q_{j,i}(u)$,
- **Node velocity** $V_{j,i}(t) = W_j(t) + U_{j,i}(t)$.

The authors further extend the RPGM model by modeling the group velocity $W_j(t)$ and the local velocity deviation of the member nodes $U_{j,i}(t)$ as random variables each drawn from the distribution $P_{j,i}(w)$ and $Q_{j,i}(u)$, respectively. The distributions can be any arbitrary type, e.g., Gaussian distributions, in order to model the various mobility patterns that may exist for different mobility groups and for the nodes within a mobility group. Analogously to the RPGM model, the characteristic group velocity $W_j(t)$ serves as a reference velocity for the nodes in the
Therefore, this model is called the Reference Velocity Group Mobility (RVGM) model. This velocity-based group representation is the time derivative of the displacement-based group representation in the RPGM model,

\[ V_{j,i}(t) = \frac{dX_{j,i}(t)}{dt} = \frac{dY_j(t)}{dt} + \frac{dZ_{j,i}(t)}{dt} = W_j(t) + U_{j,i}(t) \]  

(80)

In particular, the characteristic group velocity \( W_j(t) \) is the changes over time in \( Y_j(t) \), the displacement of the group reference center. The reference point representation can be derived from the reference velocity representation, by integrating the velocities \( W_j(t), U_{j,i}(t), \) and \( V_j(t) \) over an appropriate time interval, given the initial positions of the group reference center and the mobile nodes.

RVGM model has the following advantages: i) First, it directly provides the mobility parameters of each mobility group such as its mean group velocity and the variance in the node velocities within the group; ii) Second, by modeling the node velocities in a mobility group as a random variable with the distribution \( Q_{j,i}(u) \), the group membership of any mobile node can be easily determined, given the nodal velocity and the velocity distributions of the existing mobility groups.

The velocity representation of RVGM model also provides a clearer characterization of the mobility groups, as graphically illustrated in Fig. 51 through the transformation of the mobile nodes from the physical coordinates (Fig. 51a, left box) to the velocity data space (Fig. 51b, right box). In Fig. 51a, the reference centers of the groups overlap, and the mobile nodes are scattered with no clear groupings. However, in Fig. 51b, the mobile nodes are concentrated around the mean group velocity in their respective mobility groups and the mobility groups are clearly apparent.
Figure 51: Mobile Nodes represented by their a) Physical Coordinates, and b) Velocities
CHAPTER V

A PROBABILISTIC PREDICTIVE MULTICAST ALGORITHM IN AD HOC NETWORKS (PPMA)

5.1 Introduction

Ad hoc networks are collections of mobile nodes communicating using wireless media, without any fixed infrastructure. Conventional multicast routings are inadequate in these scenarios, as mobility can cause rapid and frequent changes in the network topology. Existing multicast protocols fall short since node mobility causes conventional multicast trees to rapidly become outdated. Frequent state changes require constant updates, reducing the already limited bandwidth available for data, and possibly never converging to accurately portray the current topology. Mobility represents the most challenging issue to be addressed by multicast routing protocols. In fact, if a multicast protocol shows robustness to mobility, it often incurs in other shortcomings such as protocol overhead or loops. Conversely, if a protocol is primarily designed to limit or to optimize the network with respect to signaling overhead and power consumption, commonly its performance degrades with the increase of mobility.

Multicast communications can be classified into source specific and group shared. In source specific multicast communication, only one node in the multicast group sends data while all the other member nodes receive data. In group shared multicast communication, each node in the multicast group wants to send/receive data to/from member nodes. A tree that spans all member nodes is called multicast tree. Multicast trees can be classified into source rooted and shared trees, according to the communication strategy. A source rooted tree has the source node as root and is optimized for source specific multicast communications. A shared tree, on the other hand, is optimized for group shared communications, and connects each group member with all the other group members.

Tree based multicast is a very well established concept in wired networks, both for source specific and group shared application support [135]. In the tree based approach, multicast routing uses a source based or group shared tree among sources and receivers, depending on the application requirements. This approach is characterized by high bandwidth efficiency, since
only one path exists between any pair of nodes. The amount of bandwidth resources required
for building up a tree is commonly less than that required for other multicast delivery structures,
since a multicast tree avoids unnecessary duplication of data [154]. This way, the optimiza-
tion routing problem in tree-based multicast is to find the minimum-weight tree that spans all
the nodes in the multicast group [9][123][147]. However, a multicast tree is more subject to
disruption due to link/node failure and node mobility than more meshed structures.

This chapter explores these contrasting issues, and proposes PPMA, a new Probabilistic
Predictive Multicast Algorithm in ad hoc networks, which leverages the tree delivery structure
for multicasting, solving its drawbacks in terms of lack of robustness and reliability in highly
mobile environments. There is, in fact, a tradeoff between the bandwidth resources used to set up
a multicast tree and the tree robustness to energy node consumption and mobility. The primary
objective of the proposed algorithm is to address this tradeoff, by decoupling tree efficiency from
mobility robustness. The intuition this work is based on is that the deterministic nature which
characterizes traditional multicast protocols tends to become their limiting factor when aiming
at robustness and scalability, especially in highly dynamic ad hoc networks. By exploiting the
non-deterministic nature of ad hoc networks, PPMA takes into account the estimated network
state evolution in terms of node residual energy, link availability, and node mobility forecast,
in order to maximize the multicast tree lifetime. The algorithm statistically tracks the relative
movements among nodes to capture the dynamics in the ad hoc network. PPMA estimates
the nodes’ future relative positions, in order to calculate a long lasting multicast tree. To do
so, it exploits the most stable links in the network, while minimizing the total network energy
consumption. We propose PPMA in both its centralized and distributed version, providing
performance evaluation through extensive simulations.

The remainder of the chapter is organized as follows. In Section 5.2 we review the main
wireless ad hoc network multicast routing protocols which the present work is related to. In
Section 5.3 we describe the motivations and goals of this work, and we introduce our novel
probabilistic cost function. In Section 5.4 we explore the terms our cost function is based on,
and point out their need to reach the described goals. In Section 5.5 we present PPMA, a
Probabilistic Predictive Multicast Algorithm in ad hoc networks, in its centralized version, while
in Section 5.6 the distributed version of the proposed algorithm is described. In Section 5.7 we
show numerical results through extensive simulations run on an ad-hoc simulator. Finally, in
Section 5.8 we conclude the chapter.

5.2 Related Work

There have been several multicast routing protocols proposed for wireless ad hoc networks in literature. In the following we present those which the present work is related to, focusing on some tree based algorithms which face the problem of determining a robust and reliable multicast tree in mobile ad hoc networks. We will point out which are their strengths and weaknesses. In particular, in Sections 5.2.1, 5.2.2, and 5.2.3 we present pros and cons of PAST-DM (Progressively Adapted Sub-Tree in Dynamic Mesh) [72], ITAMAR (Independent-tree ad hoc multicast routing) [136], and AODV (Ad Hoc On-Demand Distance Vector Protocol) [132], respectively. In Section 5.3 we then describe how our proposed probabilistic multicast algorithm effectively addresses most of their drawbacks.

5.2.1 PAST-DM: Progressively Adapted Sub-Tree in Dynamic Mesh

PAST-DM [72] utilizes a virtual mesh topology, which has the advantage of scaling very well, since the virtual topology can hide the real network topology regardless of the network dimension. A multicast session begins with the construction of a virtual mesh connecting all group members. PAST-DM gradually adapts the virtual mesh to the changes of the underlying network topology in a fully distributed manner [73]. From the computed virtual mesh, a multicast tree for packet delivery is extracted and then progressively adjusted according to the latest local topology information. At each node, the topology map is represented as a link state table. The entry of this table is the link state information of the virtual network, e.g., the hop distance, which is acquired from the virtual neighbors. Through the link state table, each node has a local view of the whole virtual topology. In PAST-DM, each source constructs its own multicast data delivery tree based on its local link state table. In order to minimize the total multicast tree cost, each source needs to construct a Steiner tree [9], which is extracted from the virtual mesh. To this end, PAST-DM uses a Source-Based Steiner tree heuristic, which relies on the definitions of hop distance, cost, and adapted cost for each virtual link. Hop distance is the minimum distance in terms of number of hops from one of the two nodes of the virtual link to the source node; cost is a generic metric which characterizes the physical link which the virtual link is associate to; and adapted cost of a virtual link is the link cost multiplied by its hop distance to the source. Thus, for any virtual link which has the source node as one of its two nodes, both its distance
value and adapted cost are 0.

PAST-DM computes an approximation of the Steiner tree by exploiting an heuristic, namely the Source-Based Steiner tree algorithm, which takes the defined adapted costs as key metric to approximate the Steiner tree. By applying this heuristic, in PAST-DM the source makes all its neighbors as its children in the multicast tree, and divides the remaining nodes into subgroups. Each subgroup forms a subtree rooted at one of the first-level children of the source. Then, each child recursively repeats the Source-Based Steiner tree algorithm. Consequently, the source node does not need to compute the whole multicast tree; in fact, each child is responsible of further delivering the data packet to all nodes in its subgroup.

5.2.1.1 Main disadvantages

PAST-DM does not explicitly take into account node mobility prediction in the computation of the adapted costs, a key metric for the construction of multicast trees. Consequently, weighting the link cost by a distance that is rapidly changing may result useless, or even incorrect, in a dynamic environment.

5.2.2 ITAMAR: Independent-Tree Ad hoc Multicast Routing

ITAMAR [136] finds a set of pre-calculated alternate trees to promptly react to link failures. This way delay could be reduced whenever a viable backup tree is available at the time of failure of the current tree. Moreover, backup trees have the advantage of making the tree based scheme more robust to node mobility. The basic idea of this multicast routing scheme is that backup multicast trees with minimum number of common links are used to reduce the number of service interruptions. This improves the mean time between route discoveries, and reduces the control overhead and the rate of data loss. At the same time, ITAMAR aims to keep the cost of transmission low. It is worth noting that this method is effective only if the tree failure times are independent [136]. Thus, under the constraint that nodes move independently, multicast trees must have no common nodes, and hence no common edges, to be independent. Since totally independent trees may not be found in many cases, ITAMAR objective is to minimize the dependence between the failure times, i.e., the correlation of the failure times of the main tree and the back up tree.
5.2.2.1 Main disadvantages

The main disadvantage of ITAMAR [136] is that minimizing the correlation between the main tree and the back-up tree is a high computational operation, and may not be convenient in all situations. Also, preventing a link-failure by pre-calculating a tree as much independent from the previous tree as possible, may result in a greater control messaging overhead, which is required for route establishment, than that used for repairing only the failed links while leaving all other tree nodes unchanged. Another disadvantage of ITAMAR is that it does not scale well since it requires every node to have a global knowledge of the network topology.

5.2.3 AODV: Ad hoc On Demand Distance Vector protocol

AODV [132] routing protocol is capable of unicast, broadcast, and multicast communication. One of the main advantages of combining in the same protocol unicast and multicast communication ability is that route information obtained when searching for a multicast route can also increase unicast routing knowledge, and vice versa. Unicast and multicast routes are discovered on demand using a broadcast route discovery mechanism. In order to reduce communication overheads, updates are propagated only along active routes, i.e., routes that have used in the recent past. Broadcast data delivery is provided by using the Source IP Address and Identification fields of the IP header as a unique identifier of the packet. As nodes join their multicast group, a multicast tree composed of group members and nodes connecting the group members is created. A multicast group leader maintains the multicast group sequence number.

5.2.3.1 Main disadvantages

There are some disadvantages in AODV protocol concerning latency, utilization efficiency, mobility robustness, and scalability under particular conditions. As far as latency is concerned, since routes may differ from the shortest paths, the average data delivery latency is expected to be higher than in a shortest path algorithm. In terms of resource utilization efficiency, source routing utilizes a lot of bandwidth due to the use of lists of addresses which increase the size of the data packet headers. Moreover, since AODV keeps hard-states in its routing table, the protocol has to actively track and react to changes in the tree. As far as mobility is concerned, AODV suffers high-rate mobility due to transmission of many routing packets, since each node maintains a routing table entry for each multicast group for which the node is a member or a router. Another disadvantage of AODV is that storm of replies and repetitive updates in hosts’
caches may occur since nodes make use of their routing caches to reply to route queries. This leads to poor scalability performance.

5.3 Problem Setup

5.3.1 Motivations and Goals

The main motivations for the introduction of PPMA, a probabilistic predictive algorithm for multicasting in ad hoc networks, are summarized hereafter:

- The deterministic nature which characterizes traditional multicast protocols tends to become their limiting factor when aiming at robustness and scalability, especially in highly dynamic ad hoc networks;
- The amount of bandwidth resources required for building up a tree is less than that required for other more meshed multicast delivery structures;
- The tradeoff between the bandwidth resource required by a multicast tree, and the tree robustness to energy node consumption and mobility has never been evaluated through extensive simulations, to the best of the authors’ knowledge.

By exploiting the non-deterministic nature of ad hoc networks, PPMA takes into account the estimated network state evolution in terms of node residual energy, link availability, and node mobility forecast, in order to maximize the multicast tree lifetime, and consequently reduce the number of costly tree reconfigurations. The algorithm statistically tracks the relative movements among nodes to capture the dynamics in the ad hoc network. This way, PPMA estimates the nodes’ future relative positions, and computes a long lasting multicast tree. To do so, it exploits the most stable links in the network, while minimizing the total network energy consumption.

To achieve these goals, we individuate a set of general rules that aims to achieve such objectives:

1. The higher battery charge a node avails, the higher its availability to take part in the tree should be;

2. The higher the number of multicast trees a node belongs to, the lower the node availability should be;
3. If the available battery charge goes under a predetermined threshold $E_{\text{min}}$, then a node should no more be considered available to take part to multicast communications;

4. The larger is the distance between two nodes, the smaller their availability to establish a communication should be. Obviously, if the distance is larger than a limit range $D_{\text{max}}$, no link between the nodes should be considered;

5. The more prone a link is to fail or break, the smaller its probability of being included in a branch of a multicast tree should be. Such a property clearly depends on positions, speeds, and directions of nodes.

5.3.2 Transmission Energy Model

An accurate model for node energy consumption per bit at the physical layer is in [83][82]

$$E = E_{\text{trans}} + \beta d^\gamma + E_{\text{elec}}$$  
(81)

where $E_{\text{trans}}$ is the distance-independent energy utilized by transmitter electronics (PLLs, VCOs, bias currents, etc.) and digital processing, $E_{\text{elec}}$ is the energy utilized by receiver electronics, and $\beta d^\gamma$ accounts for the radiated power necessary to transmit over a distance $d$ between source and destination ($\beta$ and $\gamma$ are parameters which depend on the environment; the values considered in this chapter are reported in Tab. 12). As in [112], we assume

$$E_{\text{trans}} = E_{\text{elec}} = E_{\text{elec}}$$  
(82)

Thus, the overall expression for $E$ in (81) simplifies to

$$E = 2 \cdot E_{\text{elec}} + \beta d^\gamma$$  
(83)

5.3.3 Probabilistic Link Cost

In order to synthesize the properties presented in Subsection 5.3.1, we identify a probabilistic link cost function, composed of three multiplicative terms, a Energy Term, a Distance Term, and a Lifetime Term, which will be extensively explained in Subsections 5.4.1, 5.4.2, and 5.4.3, respectively. For two generic nodes $i$ and $j$, we consider as their link cost $C_{ij}$ the following cost function

$$C_{ij} = -\log(P_i^{\mathcal{E}} \cdot P_j^{\mathcal{D}} \cdot P_{ij}^{\mathcal{L}})$$  
(84)
where

\[ P_i^E = P_i^E (E_i, E_i^{\text{min}}, E_i^{\text{max}}, W_i^{\text{out}}, W_i^{\text{max}}) \]  \hspace{1cm} (85)

is the Energy Term for node \( i \) and it weights how much residual energy node \( i \) avails for communications,

\[ P_{ij}^D = P_{ij}^D (d_{ij}, D_i^{\text{max}}(r_{\text{req}}, \epsilon_{\text{mod}}, \eta_{B}^{\text{mod}}), \gamma) \]  \hspace{1cm} (86)

is the Distance Term between node \( i \) and node \( j \), and it takes into consideration the transmission power needed by node \( i \) to communicate with node \( j \), and

\[ P_{ij}^L = P_{ij}^L (d_{ij}, D_i^{\text{max}}(r_{\text{req}}, \epsilon_{\text{mod}}, \eta_{B}^{\text{mod}}), \sigma_{ij}, \Delta t) \]  \hspace{1cm} (87)

is the Lifetime Term between node \( i \) and node \( j \), and it statistically evaluates the probability that the distance between these two nodes remains bounded for \( \Delta t \) seconds by the maximum transmission range of node \( i \), \( D_i^{\text{max}} \) given the current distance. The explanation of the parameters and variables each term depends on, as well as their units, are reported hereafter for the reader’s convenience:

\( E_i [J] \) is the battery state of node \( i \) (residual charge); \( E_i^{\text{min}} [J] \) is the energy threshold under which the node is no more available, and \( E_i^{\text{max}} [J] \) is the node maximum charge; \( \text{min} \) and \( \text{max} \) operators are extended to all nodes in the network;

\( W_i^{\text{out}} [W] \) is the instantaneous power spent for the multicast communications by node \( i \), and \( W_i^{\text{max}} [W] \) is the maximum instantaneous power node \( i \) can consume for communications;

\( d_{ij} [m] \) is the distance between node \( i \) and node \( j \) at time \( t [s] \), and \( D_i^{\text{max}}(r_{\text{req}}, \epsilon_{\text{mod}}, \eta_{B}^{\text{mod}}) [m] \) is the maximum radio range node \( i \) can reach while guaranteeing a bitrate \( r_{\text{req}} [\text{bit/s}] \), given the energy-per-bit \( \epsilon_{\text{mod}} [J/\text{bit}] \) used in radio transmission, and the spectral efficiency \( \eta_{B}^{\text{mod}} [\text{bit/s/Hz}] \) of the adopted modulation scheme;

\( \beta \) and \( \gamma \) are parameters which depend on the environment;

\( \Delta t [s] \) is the time interval used for mobility prediction, and \( \sigma_{ij} [m] \) is the standard deviation of the gaussian process used in the prediction of the distance evolution between nodes \( i \) and \( j \).

All terms are normalized and range in \([0, 1]\) and can be viewed as pseudo-probability terms. Thus, the link cost function \( C_{ij} \) in (84) can be viewed as a pseudo-probability. Actually, only the Lifetime Term \( P_{ij}^L \) is a correctly defined probability, while the other terms are normalized so as to maintain the \( C_{ij} \) values in the same variation range. This way, because of the statistical meaning
of the link cost $C_{ij}$, we can associate a probability metric to the multicast trees computed by the proposed probabilistic predictive algorithm. The Tree Probability is defined as the product of all the costs of the links included in the tree. This tree probability is obviously expected to decrease over time, if nodes are not stationary and if their future movements are unknown. The tree probability can be interpreted as the probability that all the links in a multicast tree will survive at least for a time period $\Delta t$, which is a critical parameter in (87). In Section 5.4 we describe in detail all the terms of the link cost function in (84), why each of them is necessary, and their synergic effect to meet the goals outlined in Subsection 5.3.1.

### 5.4 Probabilistic Link Cost Function Terms

In this Section we describe the properties of each term, which is argument in the link cost function in (84). In particular, in Section 5.4.1 we detail the Energy Term in (85), in section 5.4.2 we present the Distance Term in (86), and in Section 5.4.3 we derive the Lifetime Term in (87).

#### 5.4.1 Energy Term

The main purpose of the Energy Term is to keep all the nodes of the network alive as long as possible while respecting the general rules 1), 2), and 3), presented in Section 5.3.1. Also, we want to interpret the Energy Term associated to node $i$ as the probability of choosing node $i$ during the multicast tree construction. So, the Energy Term varies in the range $[0, 1]$ and is a function of the battery state of the node, i.e., the residual charge. Specifically, it assumes greater values for those nodes that have a greater residual charge. Hence, given $\mathcal{V}$ the set of indexes which correspond to the nodes in the network, a first attempt for the Energy Term associated to node $i$ could be

$$P_i(\mathcal{E}_i) = \left(\frac{\mathcal{E}_i - \min_{n \in \mathcal{V}} \{\mathcal{E}_n^{\min}\}}{\max_{n \in \mathcal{V}} \{\mathcal{E}_n^{\max}\} - \min_{n \in \mathcal{V}} \{\mathcal{E}_n^{\min}\}}\right) \cdot u(\Delta_{E_i}^{\min}) \tag{88}$$

where $\Delta_{E_i}^{\min} = \mathcal{E}_i - \min_{n \in \mathcal{V}} \{\mathcal{E}_n^{\min}\}$ and the unit step $u(\Delta_{E_i}^{\min})$ is defined as

$$u(\Delta_{E_i}^{\min}) = \begin{cases} 1 & \Delta_{E_i}^{\min} \geq 0 \\ 0 & \Delta_{E_i}^{\min} < 0 \end{cases} \tag{89}$$

Roughly, $P_i(\mathcal{E}_i)$ can be viewed as the percentage charge of node $i$, with respect to the maximum possible charge of nodes $(\max_{n \in \mathcal{V}} \{\mathcal{E}_n^{\max}\})$. The reason for the presence of $\min$ and $\max$ operators in (88) is that we want to associate to the Energy Term of a node its physical residual
charge. In fact, if we designed $P_i(E_i)$ as the percentage charge of the maximum charge of node $i$, we would miss the objective of maximizing the lifetime of all the nodes in the network.

This can be better understood through an example. Figure 52 shows $P_i(E_i)$ and $P_j(E_j)$ associated to node $i$ and node $j$, respectively, according to the following definition for the generic node $k$, where no $\min$ and $\max$ operators are used:

$$P_k(E_k) = \frac{E_k - E_k^{\min}}{E_k^{\max} - E_k^{\min}} \cdot u(E_k - E_k^{\min})$$

(90)

where we chose $E_i^{\max} = 3 \ [J]$, $E_j^{\max} = 5 \ [J]$, and $E_i^{\min} = E_j^{\min} = 0 \ [J]$, for sake of clearness. We note that (90) coincides with the percentage charge of the total charge of the node $k$. Also, we note that node $j$ has a larger total charge $E_j^{\max}$ and that, consequently, $P_j(E_j)$ has a smaller slope. If node $i$ and node $j$ had equal charge $E^*$, their probabilities in (90) would be different, since $E_i^{\max} \neq E_j^{\max}$. This way, the charges of the two nodes would not be equally weighted. In particular, node $i$ would be more likely to join multicast trees, although equipped with a smaller energy supply. This simple example evidences the real need for $\min$ and $\max$ operators in (88).

Let us point out a drawback in (88): if a new node $h$, equipped with an energy supply $E_h^{\max}$ higher than any other node in the network, joined the network, we should redefine (88) by replacing $\max_{n \in \mathcal{V}} \{E_n^{\max}\}$ with $E_h^{\max}$, where the $\max$ operator is extended only to the already
deployed nodes. Such an obstacle can be avoided either by replacing \( \max_{n \in V} \{E_{n}^{\text{max}}\} \) with an \textit{a-priori}-known \( E_{\text{max}} \), or with a value large enough to hold for most of the nodes in the network. If the battery charge of node \( i \) becomes lower than its \( E_{i}^{\text{min}} \), such a node cannot join multicast trees anymore; consequently, the term must be equal to zero. Otherwise, if a node is not currently involved in any multicast communication, its availability linearly depends on charge \( E_{i} \). Thus, all other parameters being equal, we choose the most charged nodes among the possible ones. Conversely, if a node is involved in some multicast communications, its availability should be less than the previous case. Moreover, the higher the number of multicast paths a node shares, the lower its availability should be, according to rule 2) in Section 5.3.1. This behavior can be ensured by adding an appropriate exponent to (88). Therefore, the final \textit{Energy Term} has the following expression

\[
P_{E_{i}} = \left[ \frac{E_{i} - \min_{n \in V} \{E_{n}^{\text{min}}\}}{\max_{n \in V} \{E_{n}^{\text{max}}\} - \min_{n \in V} \{E_{n}^{\text{min}}\}} \right]^{\frac{\gamma}{\gamma - 1}} \cdot u \left( \Delta_{E_{i}}^{\text{min}} \right)
\]

where \( \epsilon \) is a positive constant close to 0 whose objective is to avoid a possible division by zero, and \( \Delta_{E_{i}}^{\text{min}} = E_{i} - \min_{n \in V} \{E_{n}^{\text{min}}\} \).

The exponent in (91) is an a-dimensional term, and it increases from 1 to \( \infty \) when \( W_{i}^{\text{out}} \) increases from 0 to \( W_{i}^{\text{max}} \). Thus, as \( W_{i}^{\text{out}} \) grows to \( W_{i}^{\text{max}} \), the Energy Term tends to a two-slope broken line. More precisely, it has slope equal to zero for \( E_{i} \in [E_{i}^{\text{min}}, E_{i}^{\text{max}}] \), and slope equal to \( \infty \) in \( E_{i}^{\text{max}} \). In this limit case the term is always equal to zero but in \( E_{i}^{\text{max}} \).

Figure 53 shows how the Energy Term changes as \( W_{i}^{\text{out}} \) increases from 0 to \( W_{i}^{\text{max}} \), where constants used in (91) can be found in Tab. 12. Since the Energy Term tends to a two-slope broken line as \( W_{i}^{\text{out}} \) grows to \( W_{i}^{\text{max}} \), we can note that the more instantaneous power is spent in multicast communications by node \( i \), the less node \( i \) is available to be part of other multicast communications.

\subsection{Distance Term}

The Distance Term synthesizes the general rule 4) in Section 5.3.1. It takes into account how much power will be spent by node \( i \) to maintain link \((i, j)\) in the multicast tree. An essential piece of information to be known is an esteem of the current distance \( d_{ij}(t) \) between node \( i \) and node \( j \), hereon indicated as \( d_{ij}(t) \). The Distance Term can be expressed as following

\[
P_{D_{ij}}^{D} = \left( \frac{D_{i}^{\text{max}} - d_{ij}(t)}{D_{i}^{\text{max}}} \right)^{\gamma} \cdot u \left( D_{i}^{\text{max}} - d_{ij}(t) \right)
\]

139
Figure 53: Energy Term for different values of the instantaneous power spent in multicast communications

where

\[ D^\text{max}_i = D^\text{max}_i(r_{\text{req}}, \epsilon^\text{mod}_b, \eta^\text{mod}_b), \forall i \in \mathcal{V} \]  

(93)

\( D^\text{max}_i \) is the maximum radio range node \( i \) can reach, and depends on the requested bitrate \( (r_{\text{req}}) \), and on the spectral efficiency \( (\epsilon^\text{mod}_b) \) and energy-per-bit \( (\eta^\text{mod}_b) \) characterizing the used modulation scheme, as previously pointed out. In particular, the higher the requested bitrate, the shorter \( D^\text{max}_i \). [68][69].

Clearly, if \( d_{ij}(t) > D^\text{max}_i \), the communication link \((i,j)\) does not respect the QoS requirements in terms of requested bitrate \( r_{\text{req}} \). Thus, in this case, \( P_{ij}^D \) must be 0. As far as the unit step function \( u(.) \) in (92) is concerned, it is needed to ensure \( P_{ij}^D \) not to be negative for distance \( d_{ij}(t) \) greater than \( D^\text{max}_i \). All these considerations explain why \( P_{ij}^D \) decreases from 1 to 0 as \( d_{ij}(t) \) increases from 0 to \( D^\text{max}_i \). In particular, this decrease is characterized by at least a quadratic trend. In fact, received power decreases over distance with an exponent equal to \( \gamma \), according to our propagation model in (83). In free space \( \gamma \) is equal to 2, while in a real indoor or outdoor environment it ranges in the interval [3,5].
In Fig. 54 the Distance Term for different values of $D_1^{\text{max}}$ in the range $[50, 200]$ m is depicted. It is interesting noting that the larger the maximum radio range $D_1^{\text{max}}$ is, the smoother the decrease of the Distance Term is when the distance between two nodes increases. Conversely, for low values of $D_1^{\text{max}}$, e.g., 50 m, the Distance Term abruptly tends to zero as the distance between two nodes approaches the maximum radio range. As will be clearer later on, for each term modeling the link cost in (84), the more graceful the decrease, the slower the degradation of the link cost itself, and the better the performance achieved by our algorithm.

5.4.3 Lifetime Term

The Lifetime Term synthesizes the general rule 5) in Section 5.3.1. It has a predictive role and is defined as the probability that the maximum predicted distance $d_{ij}^* = \max_{\text{V}_{ij}} (d_{ij}(t + \Delta t))$ between node $i$ and node $j$ is smaller or equal than $D_i^{\text{max}}$, after $\Delta t$ seconds elapsed from time instant $t$. As introduced in (93), $D_i^{\text{max}}$ is the maximum distance transmitter $i$ can reach. Differently from [110][111], we consider the following probability

$$\text{Prob}\{d_{ij}^* \leq D_i^{\text{max}} | d_{ij}^* = \max_{\text{V}_{ij}} (d_{ij}(t + \Delta t))\}$$

(94)
We assume $d_{ij}(t + \Delta t)$ is a stochastic variable, obtained by summing two other stochastic variables, namely the distance, $d_{ij}(t)$, and the relative velocity between the nodes, $V_{ij}$, i.e.,

$$d_{ij}(t + \Delta t) = d_{ij}(t) + V_{ij}(t) \cdot \Delta t \quad (95)$$

The maximum predicted distance of $d_{ij}(t + \Delta t)$, $d_{ij}^*$, is computed in the worst case of node mobility, i.e., the relative velocity vector lies on the line connecting nodes $i$ and $j$, and it is oriented such that the nodes move apart from each other. Hence, we can replace in (95) the relative velocity $V_{ij}$ with its norm $|V_{ij}|$ which we refer to as relative speed,

$$d_{ij}^* = d_{ij}(t) + |V_{ij}(t)| \cdot \Delta t \quad (96)$$

Consequently, the Lifetime Term is

$$P_{ij}^L = \text{Prob}\{(d_{ij}(t) + |V_{ij}(t)| \cdot \Delta t) \leq D_{i}^{\text{max}}\} \quad (97)$$

The determination of this Lifetime Term is a challenging issue, because it generally depends on several factors. It could be well determined only if the current positions and the future destinations of nodes were exactly known, as well as their trajectories. Furthermore, a closed-form expression of $P_{ij}^L$ is hard to be found when all possible statistical parameters are taken into account, even under rough approximations of node mobility. Consequently, some assumptions about movements of nodes are necessary in order to obtain a useful expression of $P_{ij}^L$. In the following, we describe the Probability Density Functions (PDFs) of $d_{ij}(t)$ and $|V_{ij}|$, which are used to compute the Lifetime Term as in (97).

To determine the PDF of $d_{ij}(t)$, it is worth pointing out that its measured value, $\overline{d_{ij}(t)}$, may be prone to errors. In most cases, this error can be accurately described by a gaussian PDF $\mathcal{N}(\overline{d_{ij}(t)}, \sigma_{d_{ij}})$, centered at $\overline{d_{ij}(t)}$ and with variance $\sigma_{d_{ij}}^2$. Because distances are not negative, we propose to weight the gaussian PDF by a unit step function $u(y)$, which is defined in (89), i.e.,

$$P_{d_{ij}(t)}(y) = \frac{1}{\sqrt{2\pi} \cdot \sigma_{d_{ij}}} \cdot e^{-\frac{1}{2} \left(\frac{y - \overline{d_{ij}(t)}}{\sigma_{d_{ij}}}\right)^2} \cdot u(y) \quad (98)$$

The variance $\sigma_{d_{ij}}^2$ can be either set to a fixed value which depends on the measurement system, as in the case of nodes equipped with Global Position System (GPS) capabilities, or to a value proportional to $\overline{d_{ij}(t)}$, as in the case of nodes capable of measuring relative distances by message exchanging. Note that (98) is not a well defined PDF, since its integral between $-\infty$ and $\infty$ is
not equal to 1. Thus, we normalize (98) to 1, by means of a function $A(d_{ij}(t), \sigma_{d_{ij}})$, that is the area of $P_{d_{ij}(t)}(y)$ over the positive axis multiply by $\sqrt{2\pi} \cdot \sigma_{d_{ij}}$. Hence, (98) can be rewritten as

$$P_{d_{ij}(t)}(y) = \frac{1}{A(d_{ij}(t), \sigma_{d_{ij}})} \cdot e^{-\frac{1}{2} \left( \frac{y-d_{ij}(t)}{\sigma_{d_{ij}}} \right)^2 \cdot u(y)}$$

(99)

$$A(d_{ij}(t), \sigma_{d_{ij}}) = \int_{0}^{\infty} e^{-\frac{1}{2} \left( \frac{x-d_{ij}(t)}{\sigma_{d_{ij}}} \right)^2} \, dx = \sqrt{\frac{\pi}{2}} \sigma_{d_{ij}} \cdot \left[ 1 + erf \left( \frac{d_{ij}(t)}{\sqrt{2} \cdot \sigma_{d_{ij}}} \right) \right]$$

(100)

where

$$erf(x) = \frac{2}{\sqrt{\pi}} \int_{0}^{x} e^{-t^2} \, dt$$

(101)

is depicted in Fig. 55, for sake of clarity.

Figure 56 shows the PDF of $d_{ij}(t)$ as in (99) for different values of the measured distance between two nodes, $d_{ij}(t)$. To derive an expression for the relative speed, $|V_{ij}|$, some assumptions about node mobility are required. We assume a zero mean gaussian PDF for each $x$- and $y$-component of the velocity vector $V_i = V_{x,i} \hat{x} + V_{y,i} \hat{y}$, characterizing node $i$, and $V_j = V_{x,j} \hat{x} + V_{y,j} \hat{y}$, characterizing node $j$. This is equivalent to assuming a Brownian motion for each node around its current position in the time interval $\Delta t$. So, for the generic node $i$ we have

$$P_{V_{x,i},V_{y,i}}(v_{x,i},v_{y,i}) = \frac{1}{2\pi \cdot \sigma_i^2} \cdot e^{-\frac{1}{2} \left( \frac{v_{x,i}^2 + v_{y,i}^2}{\sigma_i^2} \right)}$$

(102)

where $\sigma_i^2$ is the variance of the conjuncted stochastic processes $V_{x,i}$ and $V_{y,i}$. Consequently, the amplitude of the velocity vector $|V_i| = \sqrt{V_{x,i}^2 + V_{y,i}^2}$ which we refer to as speed of node $i$, has a
Let us now introduce the relative velocity $V_{ij} = V_{xij} \hat{x} + V_{yij} \hat{y}$ between nodes $i$ and $j$, whose x- and y-components are

$$
\begin{align*}
V_{xij} &= V_{xi} - V_{xj} \\
V_{yij} &= V_{yi} - V_{yj}
\end{align*}
$$

(104)

Since $V_{xi}$, $V_{yi}$, $V_{xj}$, and $V_{yj}$ are gaussian stochastic variables, $V_{xij}$ and $V_{yij}$ are still gaussian stochastic variables. Furthermore, by assuming the x- and y-components of the node velocity vector statistically independent, it results that $V_{xij}$ and $V_{yij}$ are statistically independent too. Hence, $V_{xij}$ has mean equal to the difference of the means of $V_{xi}$ and $V_{xj}$, and variance equal to the sum of the variances of $V_{xi}$ and $V_{xj}$. The same holds for $V_{yij}$. Thus, the conjuncted probability is

$$
P_{V_{xij} V_{yij}}(x, y) = \frac{1}{2\pi \cdot (\sigma_i^2 + \sigma_j^2)} \cdot e^{-\frac{1}{2} \left( \frac{\sqrt{x^2 + y^2}}{\sqrt{\sigma_i^2 + \sigma_j^2}} \right)^2}
$$

(105)

Since $V_{xij}$ and $V_{yij}$ are zero mean gaussian variables, it results that $|V_{ij}|$ has a Rayleigh PDF.
So, taking $\sigma_{|V_j|}^2 = \sigma_t^2 + \sigma_f^2$, it yields

$$P_{|V_j|}(x) = \frac{x}{\sigma_{|V_j|}} \cdot e^{-\frac{x^2}{2\sigma_{|V_j|}^2}} \cdot u(x) \quad (106)$$

Finally, we need to find an expression for $\sigma_{|V_j|}$. A reasonable assumption is that

$$E\{|V_j|\} = |\overline{V_j}| \quad (107)$$

i.e., the mean expected value of (106),

$$E\{|V_j|\} = \frac{\sqrt{2\pi}}{2} \cdot \sigma_{|V_j|} \quad (108)$$

coincides with the measured mean value of $|V_j|$,

$$|\overline{V_j}| = \frac{d_{ij}(t) - d_{ij}(t - \Delta\tau_{ij})}{\Delta\tau_{ij}} \quad (109)$$

where $\Delta\tau_{ij}$ is the time interval between the current distance measure $d_{ij}(t)$ and the last distance measure $d_{ij}(t - \Delta\tau_{ij})$. So, the variance $\sigma_{|V_j|}$ is determined as following

$$\sigma_{|V_j|} = \sqrt{\frac{2}{\pi} \left( \frac{d_{ij}(t) - d_{ij}(t - \Delta\tau_{ij})}{\Delta\tau_{ij}} \right)} \quad (110)$$

The final expression of the Lifetime Term $P_L^{ij}$ can be found by integrating (99) and (106), accordingly to (97) (the rigorous derivation can be found in Appendix). Variables $\Delta t$ and $\sigma_{|V_j|}$ appear always in a product form, so that they actually represent a single variable, which is simply indicated as $\sigma_{\Delta t} = \Delta t \cdot \sigma_{|V_j|}$. Hence,

$$P_L^{ij} = \frac{\text{erf}\left(\frac{\sigma_{\Delta t}}{\sqrt{2}\sigma_{d_{ij}}}\right) + \text{erf}\left(\frac{\sigma_{\Delta t}}{\sqrt{2}\sigma_{d_{ij}}}\right)}{1 + \text{erf}\left(\frac{\sigma_{\Delta t}}{\sqrt{2}\sigma_{d_{ij}}}\right)} - \frac{1}{2} \left( \frac{\sigma_{\Delta t}}{\sqrt{\sigma_{d_{ij}}^2 + \sigma_{\Delta t}^2}} \right)^2 \left( \frac{\sigma_{\Delta t}}{\sqrt{\sigma_{d_{ij}}^2 + \sigma_{\Delta t}^2}} \right)^2 \left. \right|_{1 + \text{erf}\left(\frac{\sigma_{\Delta t}}{\sqrt{2}\sigma_{d_{ij}}}\right) + \text{erf}\left(\frac{\sigma_{\Delta t}}{\sqrt{2}\sigma_{d_{ij}}}\right)}$$

$$\cdot \text{erf}\left(\frac{\sigma_{\Delta t}}{\sqrt{2}\sigma_{d_{ij}}} \frac{D_{\text{max}} - d_{ij}(t)}{\sqrt{\sigma_{d_{ij}}^2 + \sigma_{\Delta t}^2}}\right) + \text{erf}\left(\frac{\sigma_{\Delta t}^2}{\sqrt{2}\sigma_{d_{ij}}} \frac{d_{ij}(t) - \sigma_{\Delta t}}{\sqrt{\sigma_{d_{ij}}^2 + \sigma_{\Delta t}^2}}\right) \quad (111)$$

Although the expression in (111) seems somehow cumbersome, it can be easily computed if the $\text{erf}$ function is approximated.

Figure 57(a) shows the Lifetime Term for different values of $d_{ij}(t)$ in $[0, 90] \text{ m}$ and $\Delta t$ in $[0, 40] \text{ s}$, with $\sigma_{d_{ij}} = 20 \text{ m}$, $\sigma_{|V_j|} = 1 \text{ m/s}$ and $D_{\text{max}} = 60 \text{ m}$. Figure 57(b) shows the Lifetime Term for different values of $\sigma_{d_{ij}}$ in $[0, 60] \text{ m}$ and $\Delta t$ in $[0, 40] \text{ s}$, with $\sigma_{|V_j|} = 1 \text{ m/s}$, $D_{\text{max}} = 60 \text{ m}$ and $d_{ij}(t) = 30 \text{ m}$. In both figures, it can be seen how the Lifetime Term
decreases as $\Delta t$ increases. In fact, if the time interval elapsed from the current esteem $d_{ij}(t)$ is large, the mobility prediction based on such esteem may have poor significance. Moreover, it is worth noting that values of the Lifetime Term close to 1 correspond to low distances between the nodes, low values of $\sigma_{d_{ij}}$, and low values of relative speed, i.e., when the nodes are likely to remain close.

5.5 Centralized PPMA

Algorithm 1 represents the pseudo-code for PPMA, the proposed probabilistic predictive multicast algorithm, in its centralized version. It extends the centralized Bellman-Ford algorithm [42], whose objective is to compute the shortest path from the node $x$ to the source $s$, with respect to a certain metric, for every node $x$ in the network. The centralized Bellman-Ford algorithm builds up a spanning tree which has $s$ as its root. It chooses the father $f_x$ of a node $x$ by minimizing the cost of the associated path toward $s$. The multicast tree connecting the source node with the receiver nodes is extracted from the computed spanning tree. This procedure leads in general to several unicast paths, which inefficiently connect the sender to the receivers with no bandwidth saving. On the other hand, the spanning tree has the good property that it does not distinguish the receiver nodes from the other nodes. This property can be fruitfully used if some nodes want to become a receiver, since a path from the new member node to the sender $s$ has already been computed.

Centralized PPMA, our centralized solution for the computation and set up of multicast trees, differs from the centralized Bellman-Ford in the crucial choice of a node predecessor. There are a number of ways to choose the father of a node in order to create a multicast tree. In the following, we propose two different criteria applied by centralized PPMA to efficiently address this issue, namely the Distance-based criterion and the Link Cost-based criterion.

As can be seen in the initialization phase of Algorithm 1, namely ‘INIT’, we define some useful node sets in order to explain how PPMA works. Let $\mathcal{V}$ be the set of all the nodes of the network, and $\mathcal{M}$ the set of all the multicast groups. For a given multicast group $m \in \mathcal{M}$, and a given maximum number of hops, the current node $x$ will choose a predecessor in its set of potential predecessors $\mathcal{PP}(x)$. We consider a generic node $v \in \mathcal{V}$ a potential predecessor of
Figure 57: Lifetime Term for different transmission ranges (Fig. 57(a)), and for different values of $\sigma_{ij}$ (Fig. 57(b))
Algorithm 1 Centralized PPMA

\textbf{INIT :}
1: nodes set of the network $\rightarrow \mathcal{V}$
2: multicast groups set $\rightarrow \mathcal{M}$
3: $\forall m \in \mathcal{M}$:
4: source $\rightarrow s$
5: current node $\rightarrow x$
6: father of $x$ $\rightarrow f_x$
7: cost of link $(i, j) \rightarrow C_{ij}$
8: path from $x$ to $s \rightarrow \mathcal{P}_x$
9: cost of path from $x$ to $s \rightarrow C_{\mathcal{P}_x}$
10: $\mathcal{H}(x) \equiv \{ h \in \mathcal{V} \mid f_h = x \}$
11: $\mathcal{N}(x) \equiv \{ n \in \mathcal{V} \mid C_{nx} < \infty \}$
12: $\mathcal{A}(x) \equiv \{ a \in \mathcal{V} \mid d_{sa} < d_{sx} \}$
13: $\mathcal{W} \equiv \{ w \in \mathcal{V} \mid \exists \mathcal{P}_w, C_{\mathcal{P}_w} < \infty \}$
14: $\mathcal{F} \equiv \{ f \in \mathcal{V} \mid \mathcal{H}(f) \neq \emptyset \}$
15: $\mathcal{PP}(x) \equiv \mathcal{N}(x) \cap \mathcal{A}(x) \cap \mathcal{W}$
16: $\mathcal{PF}(x) \equiv \mathcal{PP}(x) \cap \mathcal{F}$
17: $\mathcal{PP}_x \equiv \mathcal{N}(x) \cap \mathcal{A}(x) \cap \mathcal{W}$
18: $\mathcal{PF}_x \equiv \mathcal{PP}(x) \cap \mathcal{F}$

Centralized PPMA:
1: for all $m \in \mathcal{M}$ do
2: for all number of hops do
3: for all $x \in \mathcal{V}$ do
4: for all $i \in \mathcal{V} - \mathcal{H}(x)$ do
5: FIND_MULTI_SET($\mathcal{PP}(x)$)
6: FIND_MULTI_SET($\mathcal{PF}(x)$)
7: $\mathcal{PF}_x^\text{in} \leftarrow (\mathcal{K}(x) \cap \mathcal{PF}(x))$
8: if $\mathcal{PF}_x^\text{in} \equiv \emptyset$ then
9: $\mathcal{PF}_x^\text{out} \leftarrow (\mathcal{PF}(x) - \mathcal{PF}_x^\text{in}(x))$
10: if $\mathcal{PF}_x^\text{out} \equiv \emptyset$ then
11: $\mathcal{NF}(x) \leftarrow (\mathcal{PP}(x) - \mathcal{PF}(x))$
12: if $\mathcal{NF}(x) \equiv \emptyset$ then
13: $f_x \leftarrow $ NULL
14: else $\mathcal{NF}(x) \neq \emptyset$
15: $f_x \leftarrow \text{ARGMIN}_{j} \left( C_{\mathcal{P}_j} + C_{jx} \right), \forall j \in \mathcal{NF}(x)$
16: end if
17: else $\mathcal{PF}_x^\text{out}(x) \neq \emptyset$
18: $f_x \leftarrow $ FIND_FATHER_OUT($\mathcal{PF}_x^\text{out}(x)$)
19: end if
20: else $\mathcal{PF}_x^\text{in}(x) \neq \emptyset$
21: $f_x \leftarrow $ FIND_FATHER_IN($\mathcal{PF}_x^\text{in}(x)$)
22: end if
23: end for
24: end for
25: end for
26: end for
Algorithm 2 Choice of predecessor

\[ \text{INIT:} \]
1: \text{current radio transmitting range of node} \ k \rightarrow D_{TX}^k

\[ \text{DISTANCE-BASED CRITERION:} \]
1: \mathcal{K}(x) \equiv \{ \ k \in \mathcal{V} \mid d_{kx} \leq D_{TX}^k \}
2: \text{FIND FATHER OUT} \leftarrow \text{ARGMIN}_z (d_{zx} - D_{TX}^z), \forall z \in \mathcal{F}^\text{out}(x)
3: \text{FIND FATHER IN} \leftarrow \text{ARGMIN}_w (d_{wx} - D_{TX}^w), \forall w \in \mathcal{F}^\text{in}(x)

\[ \text{LINK COST-BASED CRITERION:} \]
1: \mathcal{K}(x) \equiv \{ k \in \mathcal{V} \mid C_{ki} \leq C_{kh}, h^\text{max} = \text{ARGMAX}_h(C_{kh}), h \in \mathcal{H}(k) \}
2: \text{FIND FATHER OUT} \leftarrow \text{ARGMIN}_z (C_{z} - C_{zh}^\text{max}), h^\text{max} = \text{ARGMAX}_h(C_{zh}), h \in \mathcal{H}(z), \forall z \in \mathcal{F}^\text{out}(x)
3: \text{FIND FATHER IN} \leftarrow \text{ARGMIN}_w (C_{w} - C_{wh}^\text{max}), h^\text{max} = \text{ARGMAX}_h(C_{wh}), h \in \mathcal{H}(w), \forall w \in \mathcal{F}^\text{in}(x)

node \ x, \text{ if node} \ v \text{ belongs to the intersection of the three sets defined below}

\begin{align*}
\mathcal{N}(x) &= \{ n \in \mathcal{V} \mid C_{nx} < \infty \} \quad \text{neighbor set} \\
\mathcal{A}(x) &= \{ a \in \mathcal{V} \mid d_{sa} < d_{sx} \} \quad \text{positive advance} \\
\mathcal{W} &= \{ w \in \mathcal{V} \mid \exists P_w, C_{P_w} < \infty \} \quad \text{nodes with a path to} \ s
\end{align*}

(112)

where \( C_{ij} \) is the cost of link \((i, j)\), according to (84); \( d_{xy} \) is the distance between node \( x \) and node \( y \); \( \mathcal{P}_s \) is a feasible path from node \( x \) to the sender \( s \); \( C_{P_s} \) is the cost of the path \( \mathcal{P}_s \), i.e., the sum of the cost of the links that belong to \( \mathcal{P}_s \), formally defined as

\[ C_{P_s} = \sum_{(i,j)\in \mathcal{P}_s} C_{ij} \]  

(113)

We can define the set of potential predecessor of node \( x \), \( \mathcal{PP}(x) \), as

\[ \mathcal{PP}(x) = \mathcal{N}(x) \cap \mathcal{A}(x) \cap \mathcal{W} \]  

(114)

The set \( \mathcal{N}(x) \) contains all the neighbors of node \( x \), and imposes that the predecessor of \( x \) has to be necessarily a neighbor of \( x \), where a node \( n \) is defined as a neighbor of \( x \) if the cost of the link \((n, x)\) is lower than \( \infty \), i.e., \( C_{nx} < \infty \). Note that, according to (84) and (92), the link cost may be \( \infty \) if the distance between the two nodes \( n \) and \( x \) exceeds the maximum radio transmitting range of node \( n \). The set \( \mathcal{A}(x) \) is the positive advance set, and contains all the nodes located inside the circle of radius \( d_{sx} \) around \( s \). The set \( \mathcal{A}(x) \) is needed in (114) in order to avoid loop formation during the construction of the tree, as proved in [112]. The last set in (114), \( \mathcal{W} \), contains all the nodes that know a feasible path towards \( s \). A potential predecessor \( w \) of \( x \) must know a feasible path toward the sender \( s \), otherwise node \( x \) cannot receive any packet from \( s \).
The set $\mathcal{PP}(x)$ can be further partitioned in two subsets: $\mathcal{PF}(x)$, i.e., the set of predecessor father nodes, and $\mathcal{NF}(x)$, i.e., the set of predecessor not-father nodes. It holds that

$$\mathcal{PP}(x) \equiv \mathcal{PF}(x) \cup \mathcal{NF}(x), \quad \mathcal{PF}(x) \cap \mathcal{NF}(x) \equiv \emptyset$$

(115)

1) The first subset, $\mathcal{PF}(x)$, is composed of those potential predecessors that currently have other children, so we can call them potential predecessor fathers, where the child set of node $a$ is defined as the set of nodes whose father is $a$. Hence,

$$\mathcal{H}(a) \equiv \{h \in \mathcal{V} \mid f_h = a\}$$

(116)

We can define a general father set $\mathcal{F}$ as

$$\mathcal{F} \equiv \{f \in \mathcal{V} \mid \mathcal{H}(f) \neq \emptyset\} \quad \text{(nodes with at least one child)}$$

(117)

which all the network nodes that have at least one child belong to. Thus, the predecessor father set can be expressed as

$$\mathcal{PF}(x) \equiv \mathcal{PP}(x) \cap \mathcal{F}$$

(118)

2) The second subset, $\mathcal{NF}(x)$, is composed of the remaining potential predecessors and can simply be defined as

$$\mathcal{NF}(x) \equiv \mathcal{PP}(x) - \mathcal{PF}(x)$$

(119)

In Fig. 58 a simple example of partition of $\mathcal{PP}(x)$ is depicted. If a link between two nodes is available, i.e., the link cost is lower than $\infty$, it is depicted with a solid arrow, while the multicast tree is depicted with dashed arrows. Node $x$, the current node, has to choose its predecessor. The potential predecessors set of $x$, $\mathcal{PP}(x)$, is composed of nodes $b, c$ and $d$. It is easy to verify that all these nodes belong to the sets $\mathcal{A}(x), \mathcal{N}(x)$ and $\mathcal{W}$. In fact, they belong to $\mathcal{N}(x)$ since they are neighbors of $x$. Moreover, they belong to $\mathcal{A}(x)$ since their distance toward $s$ is less than the distance from $s$ to $x$. Finally, they belong to $\mathcal{W}$, since they all have joined the multicast tree, and thus know a feasible path toward the sender $s$. However, as can be seen in Fig. 58, only nodes $b$ and $c$ have not a void child set. In fact, the child set of node $b$, $\mathcal{H}(b)$, is composed of two nodes, nameless in the figure, and the child set of node $c$, $\mathcal{H}(c)$, contains node $d$. Thus, the potential predecessor father set of $x$, $\mathcal{PF}(x)$, is composed of nodes $b$ and $c$, while the not-father set, $\mathcal{NF}(x)$, coincides with node $d$.

In the choice of the predecessor $f_x$ of $x$, Centralized PPMA gives higher priority to the elements of $\mathcal{PF}(x)$ than to the elements of $\mathcal{NF}(x)$. PPMA tries to avoid building up a multicast
Figure 58: Centralized PPMA: partition of the potential predecessor set $\mathcal{PP}(x)$

tree composed of unicast paths, by giving higher priority to those nodes that match a predefined requisite, e.g., being a father. Although such a process could lead to a high tree cost, the total number of links of the spanning tree is kept low. The rationale of this strategy is to exploit those links that are already used to reach some other receivers. In fact, this is the key feature of multicasting. Hence, PPMA prefers father nodes, and connects node $x$ to $s$, when it is possible, by exploiting already used paths.

The set $\mathcal{PF}(x)$ is divided in two other subsets, $\mathcal{PF}^{\text{in}}(x)$ and $\mathcal{PF}^{\text{out}}(x)$, which will be given different priorities in the choice of the predecessor of node $x$, as will be clearer in the following. So we have,

$$\mathcal{PF}(x) \equiv \mathcal{PF}^{\text{in}}(x) \cup \mathcal{PF}^{\text{out}}(x), \quad \mathcal{PF}^{\text{in}}(x) \cap \mathcal{PF}^{\text{out}}(x) \equiv \emptyset$$  \hspace{1cm} (120)

In particular, PPMA prefers to choose the nodes that belong to $\mathcal{PF}^{\text{in}}(x)$ as predecessor of node $x$ than the nodes that belong to $\mathcal{PF}^{\text{out}}(x)$. We propose two different ‘criteria’ to partition the set $\mathcal{PF}(x)$, the ‘Distance-based’ criterion and the ‘Link Cost-based’ criterion, which are described in details in the following subsections.
5.5.1 Distance-Based Criterion

The first criterion, namely the ‘distance-based’ criterion, distinguishes the elements of $PF(x)$ that have a current radio transmitting range $D_{TX}$ large enough to reach node $x$, from those that have not. Thus, according to such criterion, the set $PF_{in}(x) \subseteq PF(x)$ contains the nodes $f^{in}$ that are transmitting to some other node $a$ which is located at a distance $d_{f,a}$ from $f^{in}$ larger than the distance $d_{f,x}$ between $f^{in}$ and $x$. We can formally define $PF_{in}(x)$ as

$$PF_{in}(x) \equiv \{ f^{in} \in PF(x) \mid d_{f,x} \leq D_{f,TX}^{in} \}$$

where $D_{f,TX}^{in}$ is the current radio transmitting range of node $f^{in}$. Differently, a node that has to increase its current radio transmitting range to allow node $x$ to receive the packets it sends, is included in $PF_{out}(x)$. So

$$PF_{out}(x) \equiv \{ f^{out} \in PF(x) \mid d_{f,x} > D_{f,TX}^{out} \}$$

Figure 59(a) shows an example of network topology that explains this concept. A source $s$ and three receivers, $r_1$, $r_2$, and $r_3$, must be connected by a multicast tree. The multicast tree is depicted with unidirectional dashed arrows, while unidirectional solid arrows are used to represent existing links (whose costs are shown nearby), and bidirectional dotted arrows indicate distances. Receivers $r_1$ and $r_3$ have joined the tree through nodes $b$ and $c$, respectively, but receiver $r_2$ has not joined the tree yet. Since both nodes $b$ and $c$ have a child, the potential predecessor set and the predecessor father set of receiver $r_2$ coincide, and are composed of nodes $b$ and $c$, so that $PP(r_2) \equiv PF(r_2) \equiv \{b, c\}$. However, the current radio transmitting range of node $b, D_{b,TX}^{in}$, is larger than the distance $d_{br_2}$ from $b$ and $r_2$, while the radio transmitting range of node $c, D_{c,TX}^{out}$, is smaller than $d_{cr_2}$. Thus,

$$PF_{in}(x) \equiv \{b\}, \quad PF_{out}(x) \equiv \{c\}$$

Thus, receiver $r_2$ is connected to the sender $s$ through the predecessor $b$, which does not have to increase its current radio transmitting range to reach $r_2$. Note that, according to this partition criterion, the costs of the links $(c, r_2)$ and $(b, r_2)$ have not influence in the partition of $PF(r_2)$, i.e., in the priority given to the nodes.
Figure 59: Centralized PPMA: partition of $\mathcal{PF}(x)$ in $\mathcal{PF}^{\text{in}}(x)$ and $\mathcal{PF}^{\text{out}}(x)$ based on the ‘distance’ criterion (Fig. 59(a)), and based on the ‘link cost’ criterion (Fig. 59(b))
5.5.2 Link Cost-Based Criterion

The second proposed criterion for partitioning the predecessor father set $\mathcal{PF}(x)$, namely the ‘link cost-based’ criterion, considers link costs instead of distances. In this case, $\mathcal{PF}_\text{in}(x)$ contains the nodes $f^\text{in} \in \mathcal{PF}(x)$ such that the maximum cost of the links towards their children, $C_{f^\text{in},h^\text{max}}$, is greater than $C_{f^\text{in},x}$, where $h^\text{max}$ is the child of $f^\text{in}$ which corresponds to the maximum cost of the link from $f^\text{in}$. That is,

$$\mathcal{PF}_\text{in}(x) \equiv \{f^\text{in} \in \mathcal{PF}(x) \mid C_{f^\text{in},x} \leq C_{f^\text{in},h^\text{max}}\}$$

$$h^\text{max} = \text{ARGMAX}_h(C_{f^\text{in},h}), \ h \in \mathcal{H}(f^\text{in})$$

Differently, if a node $f^\text{out}$ does not connect a child $h$, whose associated link cost $C_{f^\text{out},h}$ is greater than $C_{f^\text{out},x}$, then $f^\text{out}$ is included in $\mathcal{PF}_\text{out}(x)$. That is,

$$\mathcal{PF}_\text{out}(x) \equiv \{f^\text{out} \in \mathcal{PF}(x) \mid C_{f^\text{out},x} > C_{f^\text{out},h^\text{max}}\}$$

$$h^\text{max} = \text{ARGMAX}_h(C_{f^\text{out},h}), \ h \in \mathcal{H}(f^\text{out})$$

It is worth pointing out that, if we choose the distance $d_{ij}$ between node $i$ and node $j$ as the metric to compute the cost of the link $(i, j)$, the ‘distance-based’ criterion and the ‘link cost-based’ criterion coincide. Anyway, this second criterion can be viewed in general as an extension of the first one, since the link cost could account for the node speed or the battery charge in addition to the distance.

To illustrate this second criterion, Fig. 59(b) shows the same network topology depicted in Fig. 59(a), but with different link costs. Let us assume that the metric of the link cost takes also into account the relative speed of the nodes, i.e., the higher the relative speed between two nodes, the higher the cost of the link connecting them. As in the previous example, in Fig. 59(b) source $s$ and three receivers, $r_1$, $r_2$, and $r_3$, must be connected by a multicast tree, where $r_1$ and $r_3$ have joined the tree through nodes $b$ and $c$, respectively, while $r_2$ has not joined the tree yet. Since both nodes $b$ and $c$ have a child, the potential predecessor set and the predecessor father set of receiver $r_2$ coincide, and are composed by nodes $b$ and $c$,

$$\mathcal{PP}(r_2) \equiv \mathcal{PF}(r_2) \equiv \{b, c\}$$

(126)

However, the cost of the link $(b, r_1)$ is smaller than the cost of the link $(b, r_2)$, while the cost of the link $(c, r_3)$ is larger than the cost of the link $(c, r_2)$. Thus,

$$\mathcal{PF}_\text{in}(x) \equiv \{c\}, \ \mathcal{PF}_\text{out}(x) \equiv \{b\}$$

(127)
Note that, according to this partition criterion, the distances $d_{cr_2}$ and $d_{br_2}$ have influence in the partition of $\mathcal{P}F(r_2)$, i.e., in the priority given to the nodes, since the distance between nodes is included in the computation of the link cost. Thus, the receiver $r_2$ is connected to the sender $s$ through the predecessor $c$, which is already connecting a child through a link with a higher cost than $C_{cr_2}$.

In general, the partition of the set $\mathcal{P}F(x)$ can be obtained by intersecting it with another set, $\mathcal{K}(x)$, that depends on the used kind of partition.

1) In the case of the ‘Distance-based’ partition, the set $\mathcal{K}(x)$ is defined as

$$\mathcal{K}(x) \equiv \{ k \in V \mid d_{kx} \leq D^k_{TX} \}$$

(128)

2) whereas, in the case of the ‘Link Cost-based’ partition, the set $\mathcal{K}(x)$ is defined as

$$\mathcal{K}(x) \equiv \{ k \in V \mid C_{kx} \leq C_{kh_{\text{max}}} \}$$

$$h_{\text{max}} = \text{ARGMAX}_h(C_{kh}), \ h \in \mathcal{H}(k)$$

(129)

Once the set $\mathcal{K}(x)$ is defined, the sets $\mathcal{P}F^{\text{in}}(x)$ and $\mathcal{P}F^{\text{out}}(x)$ are easily obtained as:

$$\mathcal{P}F^{\text{in}}(x) \equiv \mathcal{P}F(x) \cap \mathcal{K}(x)$$

(130)

$$\mathcal{P}F^{\text{out}}(x) \equiv \mathcal{P}F(x) - \mathcal{P}F^{\text{in}}(x)$$

(131)

So far, we defined three sets, $\mathcal{P}F^{\text{in}}(x)$, $\mathcal{P}F^{\text{out}}(x)$ and $\mathcal{N}F(x)$, which contain all the possible predecessors for the current node $x$. As explained in Algorithm 1, PPMA looks for the predecessor of $x$, $f_x$, primarily in the $\mathcal{P}F^{\text{in}}(x)$ set. Then, if that set is void, PPMA searches $f_x$ in the $\mathcal{P}F^{\text{out}}(x)$ set, and finally, if also $\mathcal{P}F^{\text{out}}(x)$ is void, in the $\mathcal{N}F(x)$ set. However, we did not explain how PPMA chooses the predecessor of $x$, among the elements of each set. To this end, we propose three criteria of choosing $f_x$ in a given set $\mathcal{P}(x)$.

1) The first one chooses the node $f_x$ such that the path from the receiver $x$ to the sender $s$ has the minimum cost, i.e.,

$$f_x \leftarrow \text{ARGMIN}_j \left( C_{P_j} + C_{jx} \right), \forall j \in \mathcal{P}(x)$$

(132)

2) The second one chooses the node $f_x$ such that the link from it to the receiver $x$ has the minimum link cost, i.e.,

$$f_x \leftarrow \text{ARGMIN}_j \left( C_{jx} \right), \forall j \in \mathcal{P}(x)$$

(133)
3) The third and last one takes into account the differences between the distances or between
the link costs, whether the partition of the set \( \mathcal{P}(x) \) is ‘Distance’-based or ‘Link Cost’-based,
respectively. In the former case, the difference is between the current radio transmitting range
of the node candidate as predecessor of \( x \) and the distance with \( x \), i.e.,
\[
    f_x \leftarrow \text{ARGMIN}_z (d_{zx} - D_{z_{\text{TX}}}), \forall z \in \mathcal{P}(x)
\]
(134)
In the latter case, the difference is between the cost of the link between the candidate and \( x \)
and the maximum cost of the link between the candidate and its children, i.e.,
\[
    f_x \leftarrow \text{ARGMIN}_z (C_{zx} - C_{zh_{\text{max}}})
\]
(135)
\[
    h_{\text{max}} = \text{ARGMAX}_h (C_{zh}), \ h \in \mathcal{H}(z), \forall z \in \mathcal{P}(x)
\]
Each of the above three criteria can be used by two functions, ‘FIND\_FATHER\_IN’ and
‘FIND\_FATHER\_OUT’, to choose \( f_x \) in the sets \( \mathcal{P}\mathcal{F}^{\text{in}}(x) \) and \( \mathcal{P}\mathcal{F}^{\text{out}}(x) \), respectively. The choice
among the elements of \( \mathcal{N}\mathcal{F}(x) \), instead, is always based on the first criterion, as in (132).

To summarize, as described in Algorithm 1, if the set \( \mathcal{P}\mathcal{F}^{\text{in}}(x) \) is not void, Centralized
PPMA chooses the predecessor of \( x, f_x \), in that set, using the function ‘FIND\_FATHER\_IN’.
Otherwise, if the set \( \mathcal{P}\mathcal{F}^{\text{in}}(x) \) is void, PPMA finds the set \( \mathcal{P}\mathcal{F}^{\text{out}}(x) \), and, if the set is not void,
the predecessor of \( x \) is chosen by the function ‘FIND\_FATHER\_OUT’. If also this set is void,
\( f_x \) is chosen among the elements that belong to \( \mathcal{N}\mathcal{F}(x) \), according to the criterion in (132). In
Algorithm 2 we propose two algorithms, the Distance-based and the Link Cost-based algorithm,
based on the criteria described in (134) and (135), respectively. Each algorithm defines the set
\( \mathcal{K}(x) \) in order to partition \( \mathcal{P}\mathcal{F}(x) \), and also defines the two functions ‘FIND\_FATHER\_IN’ and
‘FIND\_FATHER\_OUT’ to choose \( f_x \) among the elements of \( \mathcal{P}\mathcal{F}^{\text{in}}(x) \) and \( \mathcal{P}\mathcal{F}^{\text{out}}(x) \), respectively.

### 5.6 Distributed PPMA

Algorithm 3 represents the pseudo-code for Distributed PPMA, the proposed probabilistic pre-
dictive multicast algorithm in its distributed version. In order to build up a multicast tree, Distrib-
uted PPMA uses two different types of link costs: a private link cost, denoted with \( C_{\text{priv}} \),
and a public link cost, denoted with \( C_{\text{pub}} \). The private link cost is used by all nodes, except the
receivers not already connected to the tree, to find a minimum cost path toward the source. The
public link cost is used by the receivers not already connected to the tree to aggregate the paths,
which have been computed using the private costs, to form multicast trees. This way, i) the
number of nodes that belong to the tree is reduced, and ii) the overhead for control messages is
kept low.

For each link we have one private link cost, and as many public link costs as the total number
of multicast groups. The private link costs are computed as in (84), whereas the public link costs
either coincide with the private link costs, if the link is not used in a multicast tree, or are set to
0, if the link is part of a multicast tree. The terms ‘public’ and ‘private’ account for the fact that
the two types of costs are or are not ‘visible’ to the not already connected receivers, respectively.

Let $\mathcal{M}$ be the set of the multicast groups for a given multicast group $m \in \mathcal{M}$.

If a node $x$ is a receiver ($x \in \mathcal{R}^m$), it may or may not have just joined the tree.

If the receiver has joined the tree ($x \in \mathcal{T}^m$), it tries to find a new paths, $P_{\text{new}}$, which respects
the positive advance and whose private cost, $C_{\text{priv}}^{P_{\text{new}}}$, is lower than the private cost of the current
path, $C_{\text{priv}}^{P_{\text{current}}}$. If a lower cost path is found, the receiver changes path only if $\text{new\_cost} < \text{current\_cost}$. This condition ensures that the path change is convenient, i.e., the new path has
a sufficiently lower cost than the current one so that resources spent for changing path will be
repaid by the saving induced by the new path, as will be explained in the following. If this
condition holds, the $\text{SWITCH\_FATHER}$ function switches the old predecessor of node $x$, $f_{x}^{P_{\text{current}}}$,
with the new one, $f_{x}^{P_{\text{new}}}$. Then, the $\text{UPDATE\_CURRENT\_COSTS}$ function changes the public
link cost of $P_{\text{current}}$ links with their private costs, and the $\text{UPDATE\_NEW\_COSTS}$ function sets
the private link costs of $P_{\text{new}}$ links to 0.

If a receiver has not joined the tree ($x \notin \mathcal{T}^m$), Distributed PPMA finds the path $P^*$ with
minimum public cost, i.e., $\text{ARGMIN}_P(C_{\text{pub}}^{P})$. The public cost of a path is computed by summing
all the public costs of the links which are part of the path. If such a path is found, the receiver
joins the tree and sends to its new father, $f_{x}^{P^{*}}$, a JOIN message to establish the link. Finally, it
sets to 0 the public costs of the $P_{\text{current}}$ links.

If a node is not a receiver ($x \notin \mathcal{R}^m$), it finds the most convenient path between the available
public paths and private paths, and stores the related father. Whenever another node asks for
the cost of the path to the source, the node will give these pieces of information.

Figure 60 shows how the $\text{new\_cost}$ and the $\text{current\_cost}$ are computed by Distributed
PPMA. For sake of clarity a simple network topology has been chosen. The values of the private
costs are shown nearby the link arrows. A receiver $x$ is currently connected to the source via
Algorithm 3 Distributed PPMA

**INIT:**
1. nodes set of the network → \( V \)
2. multicast groups set → \( M \)
3. set of receivers → \( R^m \), \( \forall m \in M \)
4. multicast tree → \( T^m \)
5. current node → \( x \)
6. path → \( P \in T^m \) (path from \( x \) to \( s^m \))
7. private cost of path \( P \) → \( C^P_{priv} \)
8. public cost of path \( P \) → \( C^P_{pub} \)
9. father of \( x \) via \( P \) → \( f^P \)
10. \( H(x) \equiv \{ h \in V | \exists P \in T^m, f^P_h \equiv x \} \) \{child set\}

**DISTRIBUTED PPMA:**
1. for all \( m \in M \) do
2.   if \( (x \in R^m) \) then
3.     if \( (x \in T^m) \) then
4.       \( P_{new} \leftarrow \text{ARGMIN}_{P_{min}, P_{current}} C^P_{priv} \)
5.       if \( P_{new} \neq \text{NULL} \) then
6.         NEW&CURRENT_COSTS(\( P_{new}, P_{current} \))
7.         if \( \text{new}_\text{cost} < \text{current}_\text{cost} \) then
8.           SWITCH_FATHER(\( f^P_{current}, f^P_{new} \))
9.           UPDATE_CURRENT_COSTS(\( P_{current} \))
10.          UPDATE_NEW_COSTS(\( P_{new} \))
11.        end if
12.     end if
13.   else \( \{ x \notin T^m \} \)
14.     \( P^* \leftarrow \text{ARGMIN}_P C^P_{pub} \)
15.     JOIN(\( f^P_{new} \))
16.     UPDATE_NEW_COSTS(\( P^* \))
17.   end if
18. else \( \{ x \notin R^m \} \)
19.   \( P^\prime_{min} \leftarrow \text{ARGMIN}_P C^P_{priv} \)
20.   \( P^\prime_{min} \leftarrow \text{ARGMIN}_P C^P_{pub} \)
21.   \( P^* \leftarrow \text{ARGMIN}_{P_{min}, P_{min}} C^P_{priv}, C^P_{pub} \)
22.   if \( P^* \neq \text{NULL} \) then
23.     STORE(\( P^*, C^P_{priv}, C^P_{pub} \))
24.   end if
25. end if
26. end for  \{ Daemon running \( \forall x \in V \} \}
NEW&CURRENT_COSTS:
1: \( f_x \leftarrow f_{P_{\text{current}}} \)
2: \( \text{max}_1 \leftarrow \text{MAX}_y (C_y), \forall y \in \mathcal{H}(x) \cap \mathcal{R}^m \)
3: \( \text{max}_2 \leftarrow \text{MAX}(\text{max}_1, C_{f_{\text{current}}} ) \)
4: \( \text{max}_3 \leftarrow \text{MAX}_z (C_{f_{\text{current}}} ), \forall z \in \mathcal{H}(f_x) \cap \mathcal{R}^m - \{x\} \)
5: \( \text{max}_4 \leftarrow \text{MAX}_w (C_{f_{\text{current}}} ), \forall w \in \mathcal{H}(f_x) \cap \mathcal{R}^m \)
6: \( \text{new\_cost} \leftarrow C_{\text{priv}} + \text{max}_2 + \text{max}_3 \)
7: \( \text{current\_cost} \leftarrow C_{\text{priv}}^{\text{current}} - C_{f_{\text{current}}} + \text{max}_1 + \text{max}_4 \)

Figure 60: Distributed PPMA: computation of current\_cost and new\_cost as in Algorithm 3
node $c$, so that $P_{\text{current}} = \overline{x \ c \ a \ s}$ is the current path from $x$ towards the sender $s$. Let us assume that $x$ finds another path towards $s$, $P_{\text{new}} = \overline{x \ b \ a \ s}$, which has a lower private cost than $P_{\text{current}}$, i.e., $C_{\text{priv}}^{P_{\text{new}}} < C_{\text{priv}}^{P_{\text{current}}}$. In Fig. 60, we can see that this condition holds, since

$$C_{\text{priv}}^{P_{\text{current}}} = C_{sa} + C_{ac} + C_{cx} = 1 + 4 + 3 = 8 < 3 = C_{sa} + C_{ab} + C_{bx} = C_{\text{priv}}^{P_{\text{new}}}$$  \quad (136)$$

In such a situation, node $x$ has to check if $\text{new} \_ \text{cost} < \text{current} \_ \text{cost}$. As can be seen in the ‘COMPUTE \_ \text{COSTS}’ subroutine of Algorithm 3, $\text{current} \_ \text{cost}$ is composed of four terms,

$$\text{current} \_ \text{cost} \leftarrow C_{\text{priv}}^{P_{\text{current}}} - C_{f_x x} + \text{max}_1 + \text{max}_4$$  \quad (137)$$

The first term, $C_{\text{priv}}^{P_{\text{current}}}$, is the cost of the current path from $s$ to $x$. The second term, $C_{f_x x}$, is the cost of the link $(f_x, x)$. The third term, $\text{max}_1$, is the highest private cost of the links between $x$ and its receiver children, i.e.,

$$\text{max}_1 \leftarrow \text{MAX}_y (C_{xy}), \forall y \in \mathcal{H}(x) \cap \mathcal{R}$$  \quad (138)$$

$$\mathcal{H}(x) \equiv \{ h \in \mathcal{V} \mid f_h = x \}, \quad f_h \text{ predecessor of node } h$$

The fourth term, $\text{max}_4$, is the highest cost of the links between $f_x$ and its children, for all the children of $f_x$, as shown below,

$$\text{max}_4 \leftarrow \text{MAX}_w (C_{f_x w}), \forall w \in \mathcal{H}(f_x) \cap \mathcal{R}$$  \quad (139)$$

The sum of these four terms is the $\text{current} \_ \text{cost}$ and represents the cost of the sub-tree that connects the source to node $x$, and node $x$ to i) its receiver child, which is associated to the highest private link cost, and ii) the receiver child of $f_x$, which is associated to the highest private link cost, except $x$. Note that Distributed PPMA accounts for the communication broadcast nature in ad hoc networks, since it computes the cost of the sub-tree by summing, for each node belonging to such subtree, only the highest private link cost among all the outgoing links.

In Fig. 60, the cost of the link $(f_x, x)$, i.e., $(c, x)$, is 3 and only the receivers $r_4$ and $r_5$ belong to $\mathcal{H}(x)$. So, $\text{max}_1$ is $C_{xr_5}$, since the cost of the link $(x, r_4)$ is lower than the cost of the link $(x, r_5)$ ($C_{xr_4} = 1 < 2 = C_{xr_5}$). Moreover, $\text{max}_4$ is $C_{cr_6} = 4$, since receiver $r_6$ is the child of node $c$ associated to the highest link cost.

If node $x$ changes its predecessor, it must be ensured that the new path has not a higher cost than the current one. However, we cannot know this, since we should know which links belong
to both the new and the current paths. Thus, we need a worst-case formula that guarantees a convenience for the change of the predecessor. To this end, new\_cost is defined as

$$\text{new\_cost} \leftarrow C_{\text{priv}}^{\text{new}} + \text{max}_2 + \text{max}_3$$

where the first term, $C_{\text{priv}}^{\text{new}}$, is the cost of the new path found by node $x$. The second term, $\text{max}_2$, is the maximum between $\text{max}_1$ and the cost of the link connecting $x$ to its current predecessor $f_x$, i.e.,

$$\text{max}_2 \leftarrow \text{MAX}(\text{max}_1, C_{xf_x})$$

Finally, the third term, $\text{max}_3$, is the highest cost of the links between $f_x$ and its receiver children, except $x$, i.e.,

$$\text{max}_3 \leftarrow \text{MAX}_z(C_{f_xz}), \forall z \in \mathcal{H}(f) \cap \mathcal{R}^m - \{x\}$$

Hence, new\_cost represents the cost of the possible sub-tree connecting the nodes we described in the interpretation of current\_cost. In fact, it may be necessary to reroute the children of the current predecessor of $x$ onto the new path found by $x$.

In the case shown in Fig. 60, $\text{max}_2$ is equal to 3 since the cost of the link between $x$ and its predecessor $f_x = c$, $C_{xc}$, is higher than $\text{max}_1$. In the figure we considered bidirectional links, where the link cost is the same in both directions, for sake of simplicity. With this assumption, $\text{max}_3$ coincides with $\text{max}_4$. The links involved in the computation of new\_cost and current\_cost are shown in figure with a dotted and dash-dotted lines, respectively. Finally, we can compute them as in the following,

$$\text{current\_cost} = C_{sa} + C_{xc} + C_{cr5} + C_{xr5} = 1 + 4 + 4 + 2 = 11$$
$$\text{new\_cost} = C_{sa} + C_{ab} + C_{bx} + C_{xc} + C_{cr6} = 1 + 1 + 1 + 3 + 4 = 10$$

According to this example, node $x$ will change its predecessor since new\_cost is smaller than current\_cost.

### 5.7 Performance Evaluation

This section deals with the performance evaluation of our developed algorithms. In particular, in Subsection 5.7.1 we review the most commonly used mobility models for ad hoc networks in the literature, and describe the mobility model which has been developed and used in this chapter. Then, in Subsection 5.7.2 we describe the simulation scenario, and present the main achieved results.
5.7.1 Network Mobility Model

The network of nodes is represented as \((\mathcal{V}, D)\), where \(\mathcal{V} = \{v_1, \ldots, v_N\}\) is a finite set of nodes in a finite-dimension terrain, with \(N = |\mathcal{V}|\), and \(D\) is a matrix whose element \((i, j)\) contains the value of the distance between nodes \(v_i\) and \(v_j\). As far as the mobility models in ad hoc networks are concerned, the most commonly used models will be briefly described hereafter. Then, we propose our mobility model which is an extension to the Random Walk Model (RWM).

5.7.1.1 Random WayPoint Model (RWPM)

In this model, nodes in a large room choose some destination, and move there at a random speed uniformly chosen in \((0; V_{\text{max}}]\). Once a node has reached its destination point, it pauses for a time \(P [s]\) uniformly chosen in \([0, P_{\text{max}}]\). If the terrain border effects are modeled considering a wrap around\(^1\) environment, in the steady state a uniform distribution of nodes in the whole region is generated, whereas if a bounced back\(^2\) is considered, a lower node density in proximity of the borders can be noticed.

5.7.1.2 Random Walk Model (RWM)

This model has been proposed by Einstein in 1926 to mimic particle’s Brownian movement. A node starts moving by picking up a random direction uniformly in \([0, 2\pi]\). It continues the movement for a certain time, and then repeats the direction selection again. If during this period it hits the region border, different policies can be implemented to take into account the border effect. The node, in fact, can either be bounced back, or continue its movement as if it were in a wrap around\(^1\) environment, or be deleted and replaced\(^3\) by another node which is placed in a random position inside the region. The first two solutions are in this model equivalent, giving both a uniform distribution of nodes in the steady state, while the latter solution concentrates nodes in the center of the region. The choice of speed is the same as in the Random WayPoint Model.

\(^1\)In a wrap around environment, nodes are assumed to seamlessly move from a side of the terrain to its opposite side, as if the two sides coincided, i.e., all the boundaries of the terrain are assumed to be melted together.

\(^2\)In a bounced back environment, nodes are assumed to bounce when approaching one side of the terrain, as a “ball bounces against a wall”.

\(^3\)In a deleted and replaced environment, nodes approaching a side of the terrain are virtually ‘deleted’ and ‘replaced’ by another node, which is placed inside the terrain according to a pre-established rule.
Table 11: Network node mobility

<table>
<thead>
<tr>
<th>Mobility</th>
<th>$V_{\text{max}}$ [m/s]</th>
<th>$A_{\text{min}}$ [m/s$^2$]</th>
<th>$A_{\text{max}}$ [m/s$^2$]</th>
</tr>
</thead>
<tbody>
<tr>
<td>Low</td>
<td>5</td>
<td>-3</td>
<td>2</td>
</tr>
<tr>
<td>Medium</td>
<td>10</td>
<td>-5</td>
<td>3</td>
</tr>
<tr>
<td>High</td>
<td>15</td>
<td>-7</td>
<td>5</td>
</tr>
</tbody>
</table>

5.7.1.3 Random Direction Model (RDM)

This model is similar to the Random Walk Model, differing from that only as far as the node behavior in the border of the region is concerned. In this model a node hitting the border will choose its next direction towards the inside half plane. The angle formed by the node direction and the tangent line of the border is chosen uniformly in $[0, \pi]$.

5.7.1.4 Extended-Random Walk Model (E-RWM)

In this chapter we extend the Random Walk model to include accelerations and decelerations, the main difference being that the speed of a node can change according to an instantaneous acceleration and deceleration, as shown in the following formula which uses parameters in Tab. 11.

\[
v(t + \Delta t_{\text{mob}}) = \max(V_{\text{min}}, \min(v(t) + a \cdot \Delta t_{\text{mob}}, V_{\text{max}}))
\]

(143)

\[
\begin{cases}
  a \in [A_{\text{min}}, 0], & p_a > 0.5 \\
  a \in [0, A_{\text{max}}], & p_a \leq 0.5;
\end{cases}
\]

(144)

\[
\begin{align*}
  x(t + \Delta t_{\text{mob}}) &= x(t) + v(t + \Delta t_{\text{mob}}) \cdot \cos(\phi) \\
  y(t + \Delta t_{\text{mob}}) &= y(t) + v(t + \Delta t_{\text{mob}}) \cdot \sin(\phi)
\end{align*}
\]

(145)

where $p_a$ and $\phi$ are stochastic variables uniformly distributed in $[0, 1]$ and $[0, 2\pi]$, respectively. In (143) the node velocity $v(t)$ can change every $\Delta t_{\text{mob}}$ [s], under the constraint that $V_{\text{min}} \leq v(t) \leq V_{\text{max}}$, and the additional constraint that its derivative $a [\text{m/s}^2]$, the node acceleration, is bounded, i.e., $A_{\text{min}} \leq a \leq A_{\text{max}}$.

Moreover, in the performed simulation nodes belonging to the same multicast group are grouped into a cluster, which is modeled as a set of nodes deployed in a circular area with a given radius. The source of the relative multicast groups is in the center of this area. Inside a cluster, all nodes move with a low relative speed with respect to the source, i.e., nodes in a

---

4Velocity is uniformly distributed with $V_{\text{min}} = 0$ for each scenario.
cluster are assumed to move “coherently”. However, it is worth pointing out that not all nodes in a cluster are receivers or senders.

### 5.7.2 Simulation Results

In this work a random ad hoc network has been generated and simulated with an ad hoc simulator. All the parameters used to run simulations are reported in Tab. 12. The trees built up by Centralized PPMA and Distributed PPMA are compared to the trees built up by the Steiner algorithm [9], which builds multicast trees by minimizing their total cost. Each algorithm builds a tree with two different link cost functions: the first includes the Distance Term only, whereas the second is the Global Term, i.e., it includes all the terms (the Energy, the Distance, and the Lifetime Term, described in Subsections 5.4.1, 5.4.2, and 5.4.3, respectively). For each simulation several experiments have been run to ensure 95% relative confidence intervals smaller that 5%.

Starting from a completely unloaded randomly generated ad hoc network, 2 source rooted multicast trees are built. Multicast groups are sequentially randomly generated. Multicast group members (source and receivers) are randomly chosen among ad hoc network nodes. In particular, as summarized in Tab. 13, we consider multicast requests from Small Groups (5 receivers as mean) and Large Groups (11 receivers as mean), in order to test the network under different load conditions.

### Table 12: Simulation parameters

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>$E_{elec}$</td>
<td>5 [nJ/bit]</td>
</tr>
<tr>
<td>$\beta$</td>
<td>100 [pJ/bit · m$^3$]</td>
</tr>
<tr>
<td>$\gamma$</td>
<td>3</td>
</tr>
<tr>
<td>$[E_{min}, E_{max}]$</td>
<td>[0.2, 2] [J]</td>
</tr>
<tr>
<td>$W_{max}$</td>
<td>0.55 [W]</td>
</tr>
<tr>
<td>Terrain</td>
<td>300 · 300 [m$^2$]</td>
</tr>
<tr>
<td>Nodes</td>
<td>65</td>
</tr>
<tr>
<td>$\Delta t_{mob}$</td>
<td>0.5 [s]</td>
</tr>
<tr>
<td>Packet size</td>
<td>128 [bit]</td>
</tr>
<tr>
<td>$r_{req}$</td>
<td>32 [Kbps]</td>
</tr>
</tbody>
</table>

### Table 13: Multicast group size

<table>
<thead>
<tr>
<th>Group Size</th>
<th>$N_{mean}$</th>
<th>$\sigma$ (standard deviation)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Small</td>
<td>5</td>
<td>2</td>
</tr>
<tr>
<td>Large</td>
<td>11</td>
<td>3</td>
</tr>
</tbody>
</table>
In Fig. 61 the average Tree Lifetime for PPMA and the Steiner algorithms in a medium and high mobility environment for small multicast group size is shown, while in Fig. 62 the same metric for large multicast group size is shown. The curves related to the Distance Term show performance that gets worse with a larger multicast group size. Instead, curves related to the Global Term are less correlated to the multicast group size. Moreover, the increase of mobility affects the Distance Term based trees, shortening their lifetime. Conversely, trees built by also considering the Lifetime Term, as network mobility grows improve their lifetime in the case of the Steiner algorithm or keep their lifetime constant in the case of PPMA. In particular, Distributed PPMA achieves the objective of maximizing the tree lifetime while optimizing the network power consume. The tree lifetime of Distributed PPMA using the Global Term metric is much longer than all other curves related to PPMA algorithms, including those obtained with Centralized PPMA. This is due to the lower number of receivers connected by the Distributed algorithm, which is also reflected by a lower number of tree switches needed. Moreover, Centralized PPMA rebuilds all the multicast trees every update time step, whereas Distributed PPMA tends to keep alive the links already exploited when it is possible. Finally, we can note that Distributed PPMA suffers a little bit more the increase of network mobility and multicast group size than Centralized PPMA. In fact, the latter builds up trees with an almost constant lifetime with respect to the grow of mobility and group size, while the former looses about 20 seconds for every increase of mobility or group size.

In Fig. 63 the number of Connected Receivers for PPMA and Steiner algorithm in a medium and high mobility environment for small multicast group size is depicted, while in Fig. 64 the same metric for large multicast group size is shown. The number of connected receivers increases at the increase of both group size and mobility. Analogously, differences among Distance Term based trees and Global Term based trees get more evident at the growing of both group size and mobility. In particular, we have to remark that the Steiner algorithm does not exploit the link cost function as efficiently as PPMA does. In fact, for small multicast group size we can see that the Steiner algorithm performs initially better by using as link cost the Distance Term only, whereas at the end of the simulations it performs much better with the Global Term. Conversely, Centralized and Distributed PPMA have always the curves related to the Global

\[ \text{Group size is uniformly distributed with } N_{\text{mean}} \text{ and } \sigma \text{ in the table.} \]
Figure 61: Tree lifetime for PPMA and Steiner algorithm for small multicast groups in a medium mobility (Fig. 61(a)) and high mobility environment (Fig. 61(b))
Figure 62: Tree lifetime for PPMA and Steiner algorithm for large multicast groups in a medium mobility (Fig. 62(a)) and high mobility environment (Fig. 62(b))
Figure 63: Number of connected receivers for PPMA and Steiner algorithm for small multicast groups in a medium mobility (Fig. 63(a)) and high mobility environment (Fig. 63(b))
Figure 64: Number of connected receivers for PPMA and Steiner algorithm for large multicast groups in a medium mobility (Fig. 64(a)) and high mobility environment (Fig. 64(b))
Term higher than those related to the Distance Term. Finally, Distributed PPMA is more sen-
sible to network mobility than Centralized PPMA, although both algorithms perform better in
high mobile environment.

In Fig. 65, the average Battery Charge of the nodes for PPMA and Steiner algorithm in
a medium and high mobility environment for small multicast group size is shown, while in
Fig. 66 the same metric for small multicast group size is shown. The Energy Term affects the
slopes of the curves which include it in the link cost function. The algorithms using the Global
Term outperform those using the Distance Term only, and that is true especially for the Steiner
algorithm. However, Centralized PPMA has a slope comparable to the Steiner algorithm, the
latter requiring a computational cost very larger than the former. The curves related to PPMA
algorithms change slope more rapidly than those related to the Steiner algorithm, since more
rapidly the number of connected receivers reach 0. The gain of Distributed PPMA which uses
Global Term with respect to Distributed PPMA which uses Distance Term is remarkable, as
depicted in the figure. Moreover, Distributed PPMA which uses the Global Term metric also
performs better than Centralized PPMA which uses only the Distance Term metric.

In all these bunches of simulations it is consistently shown that our predictive PPMA al-
gorithm outperforms the Steiner algorithm, in both the Small and Large group simulation sce-
narios. Moreover, PPMA manages to better leverage the available network resources than the
competing algorithm, by exploiting the mobility predictive features it is endowed with. PPMA
shows also good robustness properties to mobility, as can be pointed out from Fig. 61-64.

5.8 Conclusions

This chapter proposed PPMA, a new Probabilistic Predictive Multicast Algorithm in ad hoc net-
works, which leverages the tree delivery structure for multicasting, overcoming its limitations
in terms of lack of robustness and reliability in highly mobile environments. PPMA exploits
the non-deterministic nature of ad hoc networks, by taking into account the estimated network
state evolution in terms of node residual energy, link availability, and node mobility forecast, to
maximize the multicast tree lifetime. The algorithm statistically tracks the relative movements
among nodes to capture the dynamics in the ad hoc network. This way it estimates the nodes’
future relative positions, in order to calculate a long lasting multicast tree. To do so, it exploits
the more stable links in the network, while minimizing the total network energy consumption.
Figure 65: Average available node energy for PPMA and Steiner algorithm for small multicast groups in a medium mobility (Fig. 65(a)) and high mobility environment (Fig. 65(b))
Figure 66: Average available node energy for PPMA and Steiner algorithm for large multicast groups in a medium mobility (Fig. 66(a)) and high mobility environment (Fig. 66(b))
We proposed PPMA in both its *centralized* and *distributed* version, providing performance evaluation through extensive simulation campaign run on an ad hoc simulator.
FINAL CONCLUSIONS AND FUTURE RESEARCH

This thesis presented resource optimized algorithms for efficient bandwidth sharing in telecommunication systems and, for sake of clarity, has been divided into three parts. The first part, composed of Chapter I and Chapter II, presented demand-assignment mechanisms in GEO satellite systems. The second part, made up of Chapter III only, proposed resource-optimized algorithms for multicast applications in wired networks. Finally, the third part, constituted of Chapter IV and Chapter V, dealt with the peculiar characteristics of a special kind of wireless networks, namely Ad Hoc Networks, and proposed PPMA, a new Probabilistic Predictive Multicast Algorithm for efficient multicast communications in ad hoc networks.

Let us now consider the main contributions of each part of this thesis separately, draw the main conclusions of this work, and outline our future research topics.

The first part of the thesis dealt with the problem of the design of a control-predictive based demand-assignment algorithm for a satellite network guaranteeing a target Quality of Service (QoS) to Internet traffic, while efficiently exploiting the air interface. The proposed novel algorithm is in charge of dynamically partitioning the uplink capacity among the connections in progress in the considered satellite spot-beam. Such a partitioning is performed aiming, on the one hand, to match the QoS requirements of each connection and, on the other hand, to maximize bandwidth exploitation. In particular, in Chapter I, a Control Theory approach, based on a Markov Modulated Chain Prediction Model for the traffic forecast, is adopted to effectively tackle the problem of the delay between the bandwidth request and bandwidth assignment, and the signaling overhead caused by control messages. The algorithms efficiently cope with both the satellite propagation delay and the delays inherent in the periodic nature of the bandwidth demand-assignment mechanism. The demand-assignment algorithm and the proposed Markov Modulated Chain traffic prediction model are shown to be easy to implement and to improve the overall satellite network performance. Performance evaluation is carried out by using OPNET network simulator. In Chapter II, a satellite controller for bandwidth reservation is proposed, which can be profitably used in New Generation IP-based Satellites [89] or in any wireless or wired network. The controller is shown to outperform other considered bandwidth sharing algorithms mainly because of three factors: i) the definition of a proper band-sharing mathematical schematization, that truly highlights the drivers of need for bandwidth in wireless connections,
ii) the utilization of a closed-loop controller that results in a high degree of adaptivity and in some degree of forecasting, and iii) the implementation of a smart scheduler that properly realizes the calculated band-sharing on the IP packets. As a result, the overall scheme is able to react to different kinds of simulation scenarios and different traffic load conditions, with varying numbers of traffic flows of different nature, minimizing the loss bit rate and maximizing the efficiency of exploitation of the wireless channel bandwidth.

The second part of this thesis presented two multicast algorithms for wired networks. DIMRO (DiffServ-Integrated Multicast algorithm for Internet Resource Optimization) and DIMRO-GS (DiffServ-Integrated Multicast algorithm for Internet Resources Optimization in Group Shared applications) algorithms, presented in Chapter III, aim to avoid bandwidth bottlenecks effectively, and to achieve an efficient distribution of link loads in the source rooted and group shared multicast tree computation. The proposed multicast algorithms, seamless integrated into the DiffServ Quality of Service approach, by leveraging their native multirate nature, efficiently address the network partition problem. Both these multicast algorithms, embedded with a novel low computational cost multirate layering algorithm to effectively accommodate heterogeneous receiver requested rates, have been described and tested in an integrated DiffServ aware network through extensive C++ based simulations. The simulation results show a better leverage of the network bandwidth resources and an improved QoS perceived by multicast group members with respect to the performance of existing multicast algorithms. Moreover, the low computational complexity which characterizes all the proposed algorithms, effectively leads to time and resource saving.

The third and last part of this thesis described the main features of a promising wireless technology, namely MANET (Mobile Ad Hoc Network), and proposed PPMA, a new Probabilistic Predictive Multicast Algorithm in ad hoc networks, which leverages the tree delivery structure for multicasting, overcoming its limitations in terms of lack of robustness and reliability in highly mobile environments. PPMA exploits the non-deterministic nature of ad hoc networks, by taking into account the estimated network state evolution in terms of node residual energy, link availability, and node mobility forecast, to maximize the multicast tree lifetime. The algorithm statistically tracks the relative movements among nodes to capture the dynamics in the ad hoc network. This way it estimates the nodes’ future relative positions, in order to calculate a long lasting multicast tree. To do so, it exploits the most stable links in the network,
while minimizing the total network *energy consumption*. We proposed PPMA in both its *centralized* and *distributed* version, providing performance evaluation through extensive simulation campaign run on an ad hoc simulator.

Our future research on multicast in wired networks will focus on dynamic multicast groups. The aim will be to develop efficient online procedures to add new receivers to the multicast tree and to release resources when receivers leave the multicast tree, so as to extend the current approach to capture the dynamics in the network.

Our future research on ad hoc networks will consider one of its most promising applications, namely Wireless Sensor Networks, whose main characteristics have been shortly described in Chapter IV. In this context, we will develop a localization protocol for geographical routing (also called localized routing), i.e., a routing algorithm where data packets are forwarded by a node to its neighbor based on their respective positions, where the neighborhood of each node is constituted by the nodes that lie within a certain radio range. Geographical routing are known to be scalable, which is a key feature in densely deployed networks such as sensor networks, whereas other routing strategy such as those belonging to the traditional proactive and reactive families are not. In order to implement a geographical routing algorithm, it is necessary to devise a scalable localization protocol to determine the absolute/relative positions of nodes and possibly assign to each sensor, or to a group of them for sake of scalability, an identification (ID), so as to allow the geographical routing algorithm to take efficient forwarding decisions. As a future research work we plan to develop an efficient localization protocol, since we strongly advocate geographical routing for sensor networks. To the best of our knowledge, few works have focussed on this issue as of this writing.
APPENDIX

In this appendix we provide the rigorous derivation of the Lifetime Term expression in (111).

To this end, we assume for \( d_y(t) \) and \( |V_y| \) the following PDF

\[
P_{d_y}(y) = \frac{1}{A(d_y(t), \sigma_{d_y})} \cdot e^{-\frac{1}{2} \left( \frac{y - d_y(t)}{\sigma_{d_y}} \right)^2} \cdot u(y)
\]

\[
P_{|V_y|}(x) = \frac{x}{\sigma_{|V_y|}} \cdot e^{-\frac{x^2}{2 \sigma_{|V_y|}^2}} \cdot u(x)
\]

where \( d_y(t) \) is the measured value and \( A(d_y(t), \sigma_{d_y}) \) is defined as

\[
A(d_y(t), \sigma_{d_y}) = \int_0^\infty e^{-\frac{1}{2} \left( \frac{x - d_y(t)}{\sigma_{d_y}} \right)^2} \, dx = \sqrt{\frac{\pi}{2}} \sigma_{d_y} \cdot \left[ 1 + erf \left( \frac{d_y(t)}{\sqrt{2} \cdot \sigma_{d_y}} \right) \right]
\]

Our objective is computing the following probability

\[
Prob\{ (d_y(t) + |V_y|) \cdot \Delta t \leq D_y^{\text{max}} \} = Prob\{ |V_y| \cdot \Delta t \leq D_y^{\text{max}} - d_y(t) \} = \int_{-\infty}^\infty P_{d_y}(y) \left( \int_{-\infty}^{D_y^{\text{max}} - y} \frac{1}{\Delta t} \cdot P_{|V_y|} \left( \frac{x}{\Delta t} \right) \, dx \right) \, dy = \int_{-\infty}^\infty \frac{1}{A(d_y(t), \sigma_{d_y})} \cdot \frac{1}{\Delta t^2 \cdot \sigma_{|V_y|}^2} \cdot \int_0^{D_y^{\text{max}}} e^{-\frac{1}{2} \left( \frac{y - d_y(t)}{\sigma_{d_y}} \right)^2} \cdot \left( \int_{-\infty}^{D_y^{\text{max}} - y} x \cdot e^{-\frac{2 \Delta t^2 \sigma_{|V_y|}^2}{2 \Delta t^2 \sigma_{|V_y|}^2}} \cdot dx \right) \, dy = \int_0^{D_y^{\text{max}}} e^{-\frac{1}{2} \left( \frac{y - d_y(t)}{\sigma_{d_y}} \right)^2} \cdot \left( 1 - e^{-\frac{(D_y^{\text{max}} - y)^2}{2 \Delta t^2 \sigma_{|V_y|}^2}} \right) \, dy \tag{146}
\]

Since the variables \( \Delta t \) and \( \sigma_{|V_y|} \) appear only in a product form, in the following we simply indicate them as a single variable, namely \( \sigma_{\Delta t} = \Delta t \cdot \sigma_{|V_y|} \). So (146) simplifies to

\[
\int_0^{D_y^{\text{max}}} e^{-\frac{1}{2} \left( \frac{y - d_y(t)}{\sigma_{d_y}} \right)^2} \cdot \left( 1 - e^{-\frac{(D_y^{\text{max}} - y)^2}{2 \sigma_{\Delta t}^2}} \right) \, dy = \int_0^{D_y^{\text{max}}} e^{-\frac{1}{2} \left( \frac{y - d_y(t)}{\sigma_{d_y}} \right)^2} \cdot \frac{1}{\sqrt{2 \pi} \sigma_{\Delta t}} \cdot \left[ erf \left( \frac{D_y^{\text{max}} - d_y(t)}{\sqrt{2} \cdot \sigma_{d_y}} \right) + erf \left( \frac{d_y(t)}{\sqrt{2} \cdot \sigma_{d_y}} \right) \right] + \left[ erf \left( \frac{D_y^{\text{max}} - d_y(t)}{\sqrt{2} \cdot \sigma_{d_y}} \right) - erf \left( \frac{D_y^{\text{max}} - d_y(t)}{\sqrt{2} \cdot \sigma_{d_y}} \right) \right] + \left[ erf \left( \frac{d_y(t)}{\sqrt{2} \cdot \sigma_{d_y}} \right) - erf \left( \frac{d_y(t)}{\sqrt{2} \cdot \sigma_{d_y}} \right) \right]
\]

177
Eq. (147) is the final expression of the Lifetime Term, the same it appears in (111).
LIST OF ACRONYMS

4G  Fourth Generation
AF  Assured Forwarding
AODV  Ad hoc On-demand Distance Vector
BFS  Breadth First Search procedure
BS  Base Station
CAC  Connection Admission Control
CBT  Core Based Protocol
CDF  Cumulative Distribution Function
CL  Controlled Load
DAMA  Demand Assigned Multiple Access
DiffServ  Differentiated Service
DIMRO  DiffServ-Integrated Multicast algorithm for Internet Resource Optimization
DIMRO-GS  DiffServ-Integrated Multicast algorithm for Internet Resource Optimization in Group Shared applications
DREAM  Distance Routing Effect Algorithm for Mobility
DSDV  Destination Sequenced Distance Vector
DSR  Dynamic Source Routing
DVB  Digital Video Broadcast
DVB-RCS  DVB Return Channel via Satellite
ECR  Exponential Correlated Random
EDF  Earliest Deadline First
EF Expedited Forwarding

E-RWM Extended-Random Walk Model

ETSI European Telecommunications Standards Institute

FIFO First Input First Output

FLC Fuzzy Logic Control

FTM Feasible solutions using adapted TM

FTP File Transfer Protocol

GEO Geostationary Earth Orbit

GPRS General Packet Radio Service

GPS Global Positioning System

GS Guaranteed Service

GSM Global System for Mobile Communications

HLR Home Location Register

HS Hub Station

ID IDentification number

IEEE Institute of Electrical and Electronics Engineers

IETF Internet Engineering Task Force

ILP Integer Linear Programming

INTMOD International Mobility Model

IP Internet Protocol

ISDN Integrated Service Digital Network

IST Information Society and Technology

ITAMAR Independent-Tree Ad hoc Multicast Routing
ITU  International Telecommunications Union

KR  Topology Knowledge Range

LAN  Local Area Network

LAR  Location-Aided Routing

MAC  Medium Access Control

MANET  Mobile Ad Hoc Network

METMOD  Metropolitan Mobility Model

MMPP  Markov Modulated Poisson Process

MN  Mobile Node

MSC  Mobile Switching Center

NATMOD  National Mobility Model

NCC  Network Control Center

OSTP  Optimal solution of the Steiner Tree Problem

PAST-DM  Progressively Adapted Sub-Tree in Dynamic Mesh

PDF  Probability Density Function

PID  Proportional-Integral-Derivative

PLL  Phase-Locked Loop

PPMA  Probabilistic Predictive Multicast Algorithm

PRADA  PRobe-bAsed Distributed protocol for knowledge rAnge adjustment

PSTN  Public Switched Telephone Network

PTKF  Partial Topology Knowledge Forwarding

QoS  Quality of Service

RDM  Random Direction Model
RPGM  Reference Point Group Mobility

RVGM  Reference Velocity Group Mobility

RWP  Random Waypoint Mobility Model

SATIP6  SATellite broadband multimedia system for IPv6

SPN  Steiner tree Problem in Networks

ST  Satellite Terminal

TCP  Transmission Control Protocol

TM  Takahashi-Matsuyama approximation for the Steiner problem in graphs

UDP  User Datagram Protocol

UMTS  Universal Mobile Telecommunication System

UT  User Terminal

VCO  Voltage Controlled Oscillator

VLR  Visitor Location Register

WLAN  Wireless Local Area Network

WRP  Wireless Routing Protocol

xDSL  Digital Subscriber Line

ZRP  Zone Routing Protocol
REFERENCES


[22] BALLARDIE, T., FRANCIS, P., and CROWCROFT, J., “Core Based Trees (CBT) and Architecture for Scalable Inter-domain Multicast Routing,” in Proceedings of ACM SIGCOMM’93, (San Francisco, USA), September 1993.


Dario Pompili was born in Rome, Italy, in 1977. He received the “Laurea” degree in Telecommunications Engineering in 2001, magna cum laude, from the University of Rome “La Sapienza”, Italy, with a thesis titled “Quality of Service in a broad band satellite network for Internet access”. The thesis, developed within the BRAHMS (BRoadband Access for High speed Multimedia via Satellite) Project, promoted by IST (Information Society of Technologies) and financed by EU, focused on the study and development of an IP Traffic Control to integrate IntServ (Inegrated Service) and DiffServ (Differentiated Service) Quality of Service (QoS) different approaches, introduction of hard states in RSVP (ReSerVation Protocol) to limit the amount of RSVP protocol signaling packets in transit along the satellite path and TCP (Transmission Control Protocol) enhancements for the satellite path.

From June 2001 he has been working at the same university on the European Union IST BRAHMS and SATIP6 (SATellite broadband multimedia system for IPv6) projects aiming at identifying, designing and demonstrating a new set of services and functionalities upon an integrated Terrestrial and Satellite broadband network.

In 2003 he worked on Sensor Networks at the Broadband and Wireless Networking Laboratory (BWN-Lab), Georgia Institute of Technology, Atlanta, as a visiting researcher. His main research interests are in Wireless Sensor Networks, Underwater Acoustic Sensor Networks and Quality of Service in Broadband Satellite Networks.

He attended several technical stages: i) One month at the Advanced School in Mobile Computing at “Scuola Normale Superiore”, Pisa, September 2003; ii) A week in July 2002 at the “VI National School CIRA - Centro Interuniversitario per le Ricerche in Automatica”, Bertinoro (Italy), attending lectures on Linear Systems and Mobile Robotics; iii) Two weeks in September 2000 at “G. Reiss Romoli High School”, L’Aquila (Italy), for a technical stage organized by Telecom Italia S.p.A., dealing with a development of a Business Plan of a tlc operator who has to decide whether is worth entering in a new foreign market.

The following is a list of his publications, as of this writing:

- D. Pompili and M. Vittucci, “PPMA, a Probabilistic Predictive Multicast Algorithm in Ad
Hoc Networks”, submitted for publication, December 2004


• D. Pompili, F. Delli Priscoli, and G. Santoro, “Enhanced IP-based satellite architecture compatible with many satellite platforms”, proceedings of IST 2004


• F. Delli Priscoli, D. Pompili, and G. Sette, “A Control Theory based Medium Access Control for the downstream satellite uplink of SATIP6 project”, proceedings of IST 2003


• F. Del Sorbo, G. Lombardi, and D. Pompili, “SIP based services and Virtual Home UMTS environment in the IST research project FUTURE”, proceedings of GLOBECOM 2002

• F. Del Sorbo, G. Lombardi, and D. Pompili, “UMTS access network architecture for multimedia services”, proceedings of IWDC 2002


• Dario Pompili, “Quality of Service in a broad band satellite network for Internet access”, Laurea Thesis (MS Thesis), University of Rome, “La Sapienza”, May 2001, (in English with an introduction in Italian)