UNIVERSITÀ DEGLI STUDI DI NAPOLI FEDERICO II

Facoltà di Ingegneria

Corso di Dottorato di Ricerca in Ingegneria Informatica ed Automatica
XXIII Ciclo
Dipartimento di Informatica e Sistemistica

RESOURCE AND TRAFFIC MANAGEMENT TECHNIQUES FOR PERFORMANCE OPTIMIZATION IN WIRELESS MESH NETWORKS

FRANCESCO PAOLO D'ELIA

Ph.D. Thesis

TUTOR

Prof. Giorgio Ventre

COTUTOR

Prof. Stefano Avallone

COORDINATOR

Prof. Francesco Garofalo

November 2010

Abstract

Since Internet was born, it has steadily evolved. New devices, new needs, new applications, new protocols, and this evolution is not going to slow down in the near future. As a result of the introduction and standardization of wireless access technology, we are living through an inevitable replacement of the old wired technology. We are going through a wireless revolution, which brings with it new ways to access information: simply, anywhere, anytime.

The new Internet challenge is to embody the new generation of wireless technology, giving users mobility and ubiquitous access. Wireless technology allows to extend Internet to rural areas, building blocks, home devices. Mesh networking did grow up and did become a solid technology, and it is ready to be widely adopted. Wireless communication, though, is affected by interference, the noise signal which may significantly affect the Signal to Interference-plus-Noise Ratio (SINR), leading to transmission errors. Moreover, wireless links have their own set of transmission parameters, which have to be properly tuned in order to suit the mesh network to the traffic dynamics.

Thanks to the availability of cost-effective radio interfaces, it is possible to endow each wireless node with multiple radios, thus exploiting simultaneous transmissions on the available orthogonal channels and reducing interference. However, a fundamental issue in the design of multi-radio wireless mesh networks is the joint channel assignment and routing problem. The problem consists in how to assign the available channels to the radio interfaces, and how to find the set of flow rates that have to be routed on each link in order to achieve a given objective. The joint problem, though, is NP-complete, and mainly heuristics have been proposed, which solve the two problems separately.

Therefore, in this Thesis we solve the routing problem and focus on the channel assignment problem in multi-radio wireless mesh networks, trying to find a channel assignment such that a given set of nominal flow rates can be scheduled. Such problem is shown once again to be NP-complete, hence heuristics are developed.

We study the influence of typical wireless parameters, like transmission data rate, transmission power and transmission frequency, on the overall performance

of a wireless mesh network and present two different approaches to channel assignment.

Assuming that the set of flow rates is given, we introduce a channel, rate and power assignment algorithm that aims at reducing the interference inside the mesh network, while trying to ensure scheduling for the nominal flows.

Since the wireless parameters configuration depends on the set of network flow rates, when the traffic changes, then the channel assignment has to be recomputed, i.e., it has to be adapted to the new rates. Therefore, we analyze the issues that arise when channels are re-assigned without taking into consideration the current channel assignment, and present an efficient channel and rate re-assignment algorithm.

Once the channel assignment and routing problems are solved, we have the proper channel configuration and the nominal set of flows to be routed on the wireless links. The traffic routing operations, though, are still performed by the network routing agent, which usually route traffic along the minimum hop-count path. In this Thesis, then, we present a routing protocol that is aware of the precomputed flow rates, and tries to route packets in order to obtain an average link utilization that is as close as possible to the nominal one. We modify a minimum hop-count routing protocol so as to consider variable costs for each link, and we propose a local search heuristic to find the best link cost that allows to optimize the link utilization.

Internet evolution has also been accompanied by new protocols, new communication paradigms and new user interactions. Peer-to-Peer applications are an exemplary case of the new trend of Internet, and they create nowadays more than half the Internet traffic. One of the most distinguishing characteristics of Peer-to-Peer applications is that they communicate by means of overlay networks, unaware of the physical network topology they are built upon. Therefore, Peer-to-Peer applications represent a strong engineering challenge for the network management, since the only way to control the traffic is to control the decisions that the applications make at the overlay level.

We study the overlay/underlay issues in the wireless mesh networking context, and outline how controlling P2P traffic is even more stringent, due to the mesh configuration sensibility to traffic patterns, and to the restraint of resources. To control the Peer-to-Peer application traffic, we have to somehow guide the overlay applications in the choice of the connection end points, providing information about the physical underlying network.

To propose innovative solutions to the overlay traffic optimization problem, the IETF ALTO Working Group has been recently founded. The aim of the ALTO Working Group is to define a structured architecture to support the cooperation between overlay and physical networks.

In this context, then, we develop and implement a new component for the

ALTO Service architecture, the ALTO Agent, designed to guide the overlay topology creation and to help the Peer-to-Peer application in selecting the best nodes inside a wireless mesh network. In order to choose the best available nodes and to evaluate the performance of wireless paths, we introduce an interference-aware path metric. With the path metric we propose, each link is penalized by taking into account the interference with previous hops on the path.

Given the importance of testing our solution in a realistic scenario, we study the integration problem of OMF-based wireless testbeds into the PlanetLab testbed. Thanks to our efforts, we have been able to exploit a realistic large scale testbed to test our ALTO Agent solution.

Finally, the ALTO Agent has been installed inside a wireless mesh network, and it has been used to retrieve performance metrics about end-to-end connections between peer mesh nodes. We evaluate the performance of the ALTO Agent into our integrated wireless testbed and show the improvements obtained for the application end user and the reduced costs for the network operator.

Acronyms and Abbreviations

ACK	Acknowledgement Packet
AE	Address Extension
ALARM	A new Location Aware Routing Metric
AODV	Ad-Hoc On-Demand Distance Vector
AODV-MLW	Ad-Hoc On-Demand Distance Vector with Modified Link Weights
AS	Autonomous System
BEB	Binary Exponential Backoff
BER	Bit Error Probability
BSCA	Balanced Static Channel Assignment
BSS	Basic Service Set
CA	Collision Avoidance
CCK	Complementary Code Keying
CERTH	Centre for Research and Technology - Hellas
CSMA/CA	Carrier Sense Multiple Access with Collision Avoidance
CSMA	Carrier Sense Multiple Access
CWB	Contention Window Based
DA	Destination Address
DCF	Distributed Coordination Function
DHT	Distributed Hash Table
DIFS	Distributed Interframe Space
DS	Distribution Service
DSM	Distribution System Medium
DSS	Distribution System Service
DSSS	Direct-Sequence Spread Spectrum
EC	Experiment Controller
EMULAB	Network Emulation Testbed
ESA	Extended Service Area
ESS	Extended Service Set
ETOP	Expected number of Transmissions On a Path
ETT	Expected Transmission Time
ETX	Expected Number of Transmissions

FCPRA	Flow-based Channel, Power and Rate Assignment
FCRA	Flow-based Channel and Rate Assignment
FIRE	Future Internet Research and Experimentation
GENI	Global Environment for Network Innovations
HTTP	Hypertext Transfer Protocol
HWMP	Hybrid Wireless Mesh Protocol
IAR	Interference-Aware Routing metric
IBAM	Interference and Bandwidth Aware routing path Metric
IBSS	Independent Basic Service Set
IEEE	Institute of Electrical and Electronics Engineers
IETF	Internet Engineering Task Force
IETF ALTO	Internet Engineering Task Force - Application Layer Traffic Optimization
ILP	Integer Linear Program
ISO/OSI	International Organization for Standardization/Open System Interconnection
ISP	Internet Service Provider
KAUMesh	Karlstad University Wireless Mesh Testbed - Sweden
LACA	Load-Aware Channel Assignment
LAN	Local Area Network
MA	Management Authority
MAC	Media Access Control
MANETs	Mobile Ad Hoc Network
MBSS	Mesh Basic Service Set
MCS	Modulation and Coding Scheme
MesTiC	Mesh based Traffic and Interference aware Channel assignment
MIC	Metric of Interference and Channel-switching
MID	Multiple Interface Declaration
ML	Mesh Link
MMPDU	MAC Management Protocol Data Units
MP	Mesh Point
MPR	MultiPoint Relay
MPLS	Multiprotocol Label Switching
MSDU	MAC Service Data Units
MSE	Mean Square Error
MVCRA	Minimum Variation Channel and Rate Re-Assignment
MVCRA-R	Minimum Variation Channel Re-Assignment
NAT	Network Address Translation
Net-X	Networking eXtensions
NIC	Network Interface Controller
NITOS	Network Implementation Testbed using Open Source code
ODIE	ORBIT Driven Integrated Experimentation
OFDM	Orthogonal Frequency Division Multiplexing

OLSR	Optimized Link State Routing protocol
OMF	cOntrol and Management Framework
ORBIT	Open-Access Research Testbed for Next-Generation Wireless
P2P	Peer-to-Peer
PERR	Path Error
PDIE	PlanetLab Driven Integrated Experimentation
PFR	Pre-computed Flow Rates
PHY	Physical Layer
PID	Peer Identifier
PL-Edge Nodes	PlanetLab Edge Nodes
PL	PlanetLab
PLE	PlanetLab Europe
PREQ	Path Request
PREP	Path Reply
PSK	Phase Shift Keying
QoS	Quality of Service
RA	Receiver Address
RERR	Route Error
RC	Resource Controller
RREQ	Route Request
RREP	Route Reply
RANN	Route Announcement
RTS-CTS	Request to Send - Clear to Send
SA	Slice Authority
SINR	Signal to Interference-plus-Noise Ratio
TA	Transmitter Address
TC	Topology Control
TCP	Transmission Control Protocol
TE	Traffic Engineering
TTL	Time To Live
UDP	User Datagram Protocol
VINI	Virtual Network Infrastructure
WCETT	Weighted Cumulative Expected Transmission Time
WiLEE	WireLEss Experimental
WLAN	Wireless Local Area Network
WM	Wireless Medium
WMN	Wireless Mesh Network

Contents

Chapter 1

Introduction

1.1 Context and Motivation

When the first computer network in the world started in the 1960s, the things were different from what are today. It was designed by a research team, it had mainly research purposes and counted only few nodes. Protocols and architectures were designed for a small size, trusted network, where the users were fixed.

ARPANET was an experimental network developed by the DARPA inside the US Department of Defense, and it was designed to connect together the available supercomputing facilities for military research. Initially, the network had to be fast, reliable, and capable of withstanding an enemy attack destroying a network node or link. From those few original computer nodes, computer networks have known a tremendous growth rate and soon the network evolved into the disorganized network of millions of computers we know today as Internet.

In less than twenty years, Internet has undergone a rapid expansion, going from to a few nodes network, to a planetary scale network.

The first index of such expansion has been the lack of IP addresses availability. The initial IP network protocol was designed with a network address field of 32 bit length, i.e., 2^{32} different addressable IP devices. The IP address exhaustion is one of the first initial design limits. Introduction of the NAT devices and finalization of the IPv6 protocol allowed to strive for the lack of addresses availability.

Internet growth and evolution have been strictly related to the users and ap-

plications behavior. Sudden popularity of Peer-to-Peer paradigm, social networks and online gaming brought more uncoordinated actors into scene, with ever growing bandwidth and low delay requirements, new applications, new protocols and an ever growing need for ubiquitous connectivity. New Internet services, like e-commerce, video on-demand, IPtv broadcast transmissions, need ever increasing and demanding requirements, like security, Quality of Service and privacy. Networking technologies are upgrading and expanding faster and faster. This rapid development has brought new services and new access technologies that are now in our everyday lives.

Mobility and ubiquitous network access are now possible with the ultimate wireless technologies, exploiting GSM bands, microwave bands, optical bands, etc. This adds complexity to the original Internet, underlines the current protocols limits, and brings new challenges in the new protocols development.

Actual Internet big challenge is to be able to provide user connectivity everywhere, anytime, i.e., to create an always-on, always-connected world. Hence, current research is focused on the development of new methods for supporting users mobility, trying to exploit the latest wireless access technologies. Mobility networking started in the mid-90s, together with the WiFi development and 802.11 standardization. Thanks to the low cost availability of wireless radio devices, new hardware was brought onto the shelves and soon acquired increasing popularity. Nowadays, mobile radio interfaces are implemented on laptops, cell phones, game stations, and so on, and we developed a strong addiction to the easy information access they lead to.

We can say in all conscience that we are going through a real wireless revolution: simply, we are going through the replacement of wires used for communication, by wireless environment at all levels, both for client connectivity, and for backbone connectivity. Local area network (LAN) has been replaced by WLAN (Wireless LAN), just like, similarly, fixed line telephones have been replaced by Mobile phones.

Nowadays wireless is everywhere, mobile network operators are heavily investing to deliver data to cars, home appliances and other unconventional plat-

forms through wireless connectivity, laying the foundations upon which the Internet of things will be built. WiFi first, WiMAX and Mesh Networks nowadays, are starting to create a new generation of seamless networks that extend from small portable devices to house automation devices, in-car devices and beyond.

Wireless Mesh Networks (WMNs) will be the first big changer. Mesh networks can provide entire regions, blocks or buildings with an Internet connection faster than what is currently available over the telephone cable lines.

Companies like Verizon, Sprint and Cingular are already rolling out high-speed Internet networks and other telephone companies providers are on their way. To understand how WMNs work, we can start with the familiar home WiFi network: we have one wireless Access Point, which broadcasts to all the computers in the house, each with its own WiFi receiver. If we want to deploy several Access Points, we will have to interconnect them with wired connections.

Mesh networks are composed of nodes connected in a completely wireless manner. Each node and each user can join the network and actively contribute with packet routing, thus helping expanding the overall mesh wireless area coverage. In a Mesh Network, each computer may not only be an access point client, but it may also perform routing functions, so that information can travel longer distances, in a series of short hops, and can have multiple paths to choose from, thus improving reliability. Wireless home devices, then, may exploit the multi-hop wireless links in order to reach the Internet gateway.

Mesh networks can be inexpensive, quickly built or expanded, and highly reliable. Such a network might be used in a neighbourhood to create a large wireless network composed of many individual homes, connected to each other by short-range WiFi and sharing a single Internet connection to the outside world. Or, on a smaller scale, a low-power mesh network could link all of the electronics and appliances inside the home environment.

We are going towards a world with fewer wires, but an atmosphere stuffed with radio waves. The wireless medium presents some characteristics that make it unique and at the same time hard to control. One of the biggest obstacles in the creation of the pervasive wireless connectivity is the interference and spectrum

availability problems. Radio space is a precious resource, and wireless communications have to share the radio space with services like TV, radio, air traffic control, garage door openers and so. All such services need their own electromagnetic frequency channels to keep from interfering with something else. Valuable radio space will be freed up eventually, when the switching to an all-digital television will take place.

Another doubt that could be sort out is the bandwidth availability. Skeptics could point out that wireless will never deliver the same maximum bandwidth as wires, cables or fiber-optics. Moreover, many of the wireless concepts also depend on battery power, traditionally the weak link in mobile technology. But the first phase of the wireless revolution has begun, and once a technology has been shown to work, focused engineering efforts can obtain new and fresh resources from it. An exaggerated forecast is that someday every electronic device will have an Internet address. We remark that this may be an exaggeration, of course, but it could not be far off from what ubiquitous wireless would make possible.

1.2 Contributions

Thanks to low management costs and easiness of deployment, wireless connectivity has drastically changed the way Internet access is provided to the final users. Thanks to the complete reduction in wired infrastructure, Mesh Networks have shown to be advantageous not only for the final users of the network, but also for the network manager. Given the large adoption of wireless networks, and the recent introduction of WMNs, one of the hot topic in current research papers is wireless mesh network performance optimization in terms of throughput.

Wireless Mesh Networks are all wireless backbone infrastructure networks, capable of multi-hop packet routing over completely wireless links. With a wired network, the speed of a link will always be the speed that wire is capable of, and the speed we can get today will be the speed we get tomorrow and the day after and so on. With a completely wireless network, instead, each link is characterized by several configuration parameters that directly influence its performance.

Given two nodes, u and v, a link $u \rightarrow v$ between them will be possible if the Signal to Interference-plus-Noise Ratio ($SINR$) measured at node v is greater than a particular threshold γ_{r_m}, where $SINR_{uv} = f(P_u, G_{uv}, n_v, \sum_{x \rightarrow y} I(x \rightarrow y))$.

Therefore, a wireless link will depend on the transmission power P_u, the gain factor G_{uv}, the noise n_v, and the interference signal powers I (due to the active transmissions on generic interfering links $x \rightarrow y$), measured at node v. Moreover, the transmission data rate has direct influence on the γ_{r_m} threshold value.

One of the issues with wireless speed is the fact that it depends on the environmental noise, and the noise signal, if powerful enough, may eventually completely block a wireless link. Another issue is that the presence and speed of a link depend a lot on the distance between the nodes (through the gain factor G), and implicitly, on whatever happens in between the two nodes in terms of external interference.

When considering the $SINR$ value between two nodes, the internal interfering sources are those near-by nodes transmitting on the same wireless channel.

An usually adopted solution to reduce the interference due to internal sources is to provide each mesh node with multiple radio interfaces. By configuring the multiple radio interfaces with the orthogonal communication channels that are made available by the IEEE 802.11 standards, it is possible to reduce the number of the interfering links.

Therefore, while external interference sources are in general out of our control, managing the correct channel assignment inside the network is required, in order to reduce interference from internal sources.

The channel assignment determines the number of links sharing the same channel. Because of transmissions collision, it may happen that only one link at a time can be active on a given channel c, while the other stations have to hold themselves back. The channel assignment, then, determines which wireless links have to share the same channel capacity and, in the last instance, their resulting available bandwidth.

With a channel assignment algorithm, it is possible to determine directly the amount of bandwidth that is granted to the wireless links. In principle, we could think to provide the most busy links with a channel with lower interference level,

hence with greater available bandwidth.

Challenging research issues in wireless mesh networks are the channel assignment and the routing problems: while the first is the problem to find an assignment of channels to radios, the latter is the problem to find a set of flow rates for every network link which optimize a given objective (for example, typical objectives are to maximize the throughput, or to maximize the minimum end-to-end rate).

Throughout the Thesis, we outline how the two problems are not independent.

In recent literature, several works are focused on the wireless mesh network optimization, and in particular, several channel assignment approaches have been proposed. However, several issues are still open and in this work we address some of them.

The wireless link has some inner characteristics that make it different from the classical wired link. For example, the possibility to set the transmission power and rate, together with tuning the proper frequency channel.

In this Thesis, we define the channel assignment problem and then analyze the impact of tuning the transmission power and rate of a link on the efficiency of a channel assignment algorithm.

Moreover, since the channel assignment algorithm has direct influence on the bandwidth available on each link, one soon realizes that the channel assignment may be tailored to a particular traffic pattern. To cope with a particular amount of traffic, then, we should assign channels in order to allow the bandwidth on a link to exceed the expected traffic request.

Different traffic patterns will eventually require different assignment of channels. The channel assignment, then, should be recomputed when a variation in the traffic matrix occurs.

The channel assignment re-computation is an actual research issue in the field of wireless mesh networks.

In this Thesis, we design an adaptive channel assignment algorithm that can readapt channels to the new traffic patterns. Such an algorithm works on smaller time scales with respect to classic channel assignment algorithms.

We present a simple heuristic that takes the current channel assignment into

account and, minimizing the number of radio interfaces to reconfigure, assigns the channels trying to cope with a variation in the traffic flows in the best possible way. Moreover, the proposed algorithm also attempts to determine the most suitable transmission rate for each network link.

Therefore, the WMN configuration is strictly bound to the traffic pattern, and it has to continuously adapt to their dynamics.

Finally, starting from observations made in Section 1.1, and taking into account the new applications, new paradigms and new user interactions developing in the last years, my attention focus on the P2P applications which gather together all these new features.

Since more than half the current Internet traffic is created by applications communicating by means of the P2P paradigm, we study how P2P applications traffic affected the nominal WMN behavior.

Then, we study the P2P traffic, we analyze it and outline how the peer nodes communicate with each other by creating an overlay network. This means that the P2P traffic is a direct consequence of the overlay network creation, i.e., it is a direct effect of the application decisions. The overlay network structure is distributed, and each node inside the P2P network contributes in the creation of the overlay. We study the main issues in the interaction between overlay and physical networks, and in particular we focus on the P2P file sharing Bittorrent application in the context of Wireless Mesh Networks.

In a Bittorrent file sharing tracker-based scenario, each P2P node selects a subset of the available peer nodes, and it will establish connections mostly with them. The peer selection procedure is made by randomly picking up nodes in the P2P swarm and it currently does not take into consideration the actual physical network configuration. This may lead to serious inefficiencies.

Therefore, we ask ourselves if it is possible to optimize the overlay network creation, in order to control P2P traffic and adapting it to the actual physical network configuration.

The IETF Working Group ALTO [1] defined a service, the ALTO Service, that can be used to provide P2P clients with information of the underlying net-

work. Since the P2P traffic is the result of application level decisions, the aim is to control the P2P traffic by advising the application decisions. In other words, the ALTO service may be used by the network provider to guide the P2P application in the creation of the overlay network.

In this context, then, we implement and deploy an ALTO-compliant service. Then, by means of a modified tracker, we create a P2P overlay network topology through the use of the ALTO service.

Finally, we explore the ALTO standard in the wireless mesh network context, and design and implement an ALTO Agent. We show that with the ALTO Agent it is possible to successfully contain the P2P traffic inside the Mesh Network, leading to performance improvements and optimization of network resources.

In the next Section we illustrate in detail the structure of the Thesis, with particular attention to the Chapters organization.

1.3 Structure of the Thesis

The Thesis is structured in two parts.

In Part I, which includes Chapters 3, 4 and 5, we present the traffic flow optimization problem, and in particular we focus on the channel assignment and routing problems.

In Part II, which includes Chapters 6, 7, 8, 9 and 10, we analyze the P2P traffic behavior and we underline its relations with the application decisions. In particular, we focus on the optimization of both the overlay network topology creation and P2P application performances.

The Chapters are structured as follows.

- In Chapter 2 we recall a few background notions of the Wireless Network 802.11 standard, we introduce the 802.11s Wireless Mesh Network standard and we present some of the typical issues that arise when trying to optimize the mesh network performances.

In Chapters 3, 4 and 5 we introduce the traffic flow optimization problem in

Wireless Mesh Networks. We present the channel assignment and the routing problems and we propose some algorithms for their solutions.

- In Chapter 3 we present the channel assignment problem. First of all, we present a system model to formalize the problem. We define the collision domain of a link and we show how to compute the total utilization of a collision domain. Then, we state our problem and we present two different approaches to the solution of the channel assignment problem. Both our approaches are designed to exploit the inner features of the wireless links, trying to adapt the wireless configuration to the traffic flow rates. We study the parameters configuration of the wireless link and try to tune transmission power and rate to decrease the interference level in the wireless network. First, we present a channel assignment that, together with tuning the transmission rate and power on a link, tries to decrease the maximum total utilization among all the collision domains in a wireless network. Then, after showing the need for a dynamic wireless network configuration, we present a channel re-assignment algorithm. Starting from a current channel assignment, the algorithm tries to adapt to the new traffic flow rates while keeping the number of radio reconfigured to a minimum. For each of the channel assignment approaches simulation results are given.

- In Chapter 4 we present a cross-layer traffic demand aware channel assignment. Applications would provide network routers with information about traffic flow rates and traffic destinations. Together with routing paths information, the channel assignment algorithm would then be able to adapt the wireless links configuration to the traffic requests. We break up the channel assignment algorithm in two components, the probe component, responsible for the traffic information exchange, and the channel assignment component, responsible for the assignment execution. To implement and test the proposed channel assignment, we take advantage of a framework based on OLSR [2] and Net-X [3]. Finally, we run our algorithm on the Wireless Mesh Network testbed of the Karlstad University, in Sweden.

- In Chapter 5 we address the routing problem. Given an objective function, like maximizing the aggregate throughput, we consider the optimization problem and its solution, i.e., the nominal traffic flow rates on the wireless links. Then, we actually deal with the problem of using the links, in terms of average throughput, as dictated from the optimization problem solution. On each wireless link we try to obtain an average utilization as close as possible to the nominal one. For this purpose, we introduce the AODV-MLW routing protocol, which is able to route packets along the minimum cost path. Then, by exploiting the Tabu Search algorithm, conveniently tuning its research parameters, and bearing the set of nominal traffic flows in mind, for each wireless link in the network we find the set of costs that leads to the optimal average link utilization. Finally, we compare AODV-MLW with AODV in terms of throughput performance.

In Chapters 6 through 10, we explore the Application-layer traffic optimization problem in the mesh networks field. In particular, we focus on the P2P applications and the overlay networks they build up.

- In Chapter 6 we present the issues stemming from the uncoordinated overlay and underlay networks interaction. We cite several works introduced in literature that study the overlay/underlay interaction problem, and we explore what are the main overlay networks features. Overlay decisions collide with those made at the underlying level, especially in terms of overlay packet routing operations and topology creation. Some of the main inefficiencies are very harmful both for the ISPs and for the final application user: routing path oscillations and excessive physical distance between overlay neighbour nodes are some of the major threats to the correct ISP network management. Then, we focus on the overlay/underlay issues when the physical network is a WMN, finding that even more problems arise. What emerged is that overlay applications may benefit from cross-layer information exchange, especially in the WMN context. Finally, we look for a new cross-layer architectural model in order to exchange information between layers and to optimize both application and network performance.

Modified applications would then be able to provide resource information to the mesh nodes and thus get help in the overlay topology creation and routing. At the same time, a new, properly designed network metric would help applications in resource locating and in choosing the most performing nodes.

- In Chapter 7 we focus on the problematical interaction between ISP traffic engineering operations and P2P applications. While P2P application objective is to improve user performances, ISPs want to decrease management costs and to enforce their own routing paths decision. A cooperation between ISPs and P2P applications is then desirable. In this Chapter we introduce the solution proposed by the IEEE Working Group ALTO [1], and we show how a simple Bittorrent tracker-based file sharing application may exploit the ALTO Service. In particular we aim at exploiting the ALTO Service in the creation of the overlay network topology.

- In Chapter 8 we present a new wireless path metric for multi-radio wireless mesh networks. Following the observations made in Chapter 6, we introduce a new path metric that has a twofold meaning. We briefly describe the inadequacy of commonly used link metrics in modeling intra-path interference and we stress the importance of a new path metric. To decrease the interference on a path, we imagine that a link metric should be able to evaluate the actual interdependencies among the interfering wireless links. Then, we introduce a new interference aware path metric and explore the possibility to use it to rank nodes inside a wireless mesh network. By measuring the end-to-end path metric values, a nodes ranking service may assign preferences to the peer nodes inside a wireless mesh networks. In this way, we could also provide P2P applications with information about Peer-to-Peer connection performances in a WMN.

- In Chapter 9 we present a testing architecture for Wireless Mesh Networks experiments. We focus on how we integrated an OMF-based (cOntrol and Management Framework) [4] wireless testbed into the PlanetLab testbed [5].

It is then possible for a PlanetLab user, to run experiments that involve both PlanetLab wired nodes and OMF-based wireless nodes. With our approach, multiple concurrent experiments are possible, since our Scheduler is able to reserve resources for each PlanetLab slice. Finally, we show how we use our integrated testbed setup to conduct a P2P traffic management experiment. In particular, by deploying an ALTO Service and a modified Bittorent tracker, we explore the gain that an optimization technique based on the ALTO Service can lead to.

- In Chapter 10 we present an implementation of the ALTO Service in a Wireless Mesh Network context. We refer to a file sharing tracker based Bittorrent P2P application. We explore how a random peer selection is harmful for a Wireless Mesh Network, and we introduce a WMN localized ALTO Agent, which is responsible for providing mesh peers ranking functionalities. Therefore, we implement a hierarchical ALTO compliant architecture, where the Bittorrent tracker is the client for the ALTO Service. By intervening on the peers list returned by the tracker to the active peer mesh nodes, the ALTO Agent is able to help the P2P nodes in picking up the best available mesh peers. When contacted by the asking peer, the Bittorrent tracker would contact the ALTO Server that, together with the help of the ALTO Agent and its mesh specific performance metrics, would sort in decreasing order of performance the available peer nodes. We implement and test the proposed architecture on a OMF-based wireless mesh testbed, integrated into the PlanetLab testbed, and showed the performance improvements, both in terms of application user download times and traffic volume crossing the mesh network boundaries.

Finally, we give some conclusions in Chapter 11.

1.4 Related publications

Part of the material presented in this Thesis has appeared in some conference proceedings or journals, or has been submitted for publication. In particular:

- The work presented in Chapter 3 has appeared in [6], [7] and is going to appear in [8]. Moreover, a paper has been submitted for publication in [9].
- The work presented in Chapter 4 has appeared in [10].
- The work presented in Chapter 5 has appeared in [11].
- The work presented in Chapter 6 has appeared in [12] and [13].
- The work presented in Chapter 9 is going to appear in [14].
- The work presented in Chapter 10 has been submitted for publication in [15].

Chapter 2

Background

In this Chapter we present a quick introduction to 802.11 wireless standard and to the 802.11s wireless mesh standard. We outline the main wireless standard features and the extensions that are brought by the new wireless mesh network paradigm. In particular, we describe the basic wireless mesh infrastructure and its peculiar advantages over the classical wireless networks: low deployment and management costs thanks to a complete absence of any wired architectural support. These advantages have been the key in success of wireless mesh networks deployment.

One of the most fitting wireless mesh network use case is the broadband Internet connectivity access in those remote areas where wired infrastructure is too expensive or too hard to enforce. Several wireless mesh networks have been built in small and medium sized towns and city wide wireless mesh networks have already been realized, like the one in [16], realized by the researchers of the Massachusetts Institute of Technology (MIT). Moreover, Wireless ISP are making large use of the wireless mesh network model to offer easily and quickly their services with reduced investments.

The most consolidated model of Wireless ISP is that of mobile operators offering broadband wireless mesh connectivity that complements mobile cellular devices, assuring mobility support. Operators like Swisscom Mobile [17], Sunrise [18], Orange [19] and Cablecom [20], followed this approach. Moreover, FON [21] and other free network communities may push towards the creation of

global community networks.

The throughput performance optimization in a wireless mesh network, though, is not simple. Wireless communications are affected by interference. The closest wireless links sharing the same frequency channel, in fact, cannot communicate simultaneously. In order to cope with the interference problem, the IEEE standards provide several orthogonal non interfering communication channels. Each channel has a reserved central working frequency and a reserved working bandwidth. The orthogonal channels are those channels whose frequency spatial distance is sufficiently high to avoid the respective bandwidths to intersect. By deploying multiple radios on each node, the correct orthogonal channels assignment is mandatory to reduce the interference problem.

Moreover, each wireless link has some characteristic parameters, like transmission rate and transmission power, whose values have direct influence on the degree of interference and on the available bandwidth on a link. Given the set of source destination couples and the traffic matrix in the network, we will have the corresponding traffic flows that have to be routed on the wireless links. The configuration of the wireless links parameters has a direct influence on the available bandwidth on the current link. In order to cope with the actual traffic matrix, then, the parameters configuration has to be properly adapted. The purpose is to obtain on each link a resulting available bandwidth that exceeds the traffic flow rate value that has to be routed on that link.

In Section 2.3, we explore what are the main wireless mesh networks engineering challenges, focusing on the channel assignment problem and on the routing flow optimization problem.

2.1 Wireless Networks Overview

Wireless Networks are standardized in the 802.11-2007 [22] amendment. The central device in the wireless architecture is the Access Point (AP) station, which is conforming to the MAC and PHY layer 802.11 specifications. 802.11 compliant wireless devices have to authenticate and associate with an AP in order to gain

network access. Even if the traffic destination is within communication range, all the frames have to flow through the associated AP. Therefore, the AP and the associated wireless devices form a star topology, a wireless single-hop network where the devices have to relay on the AP to exchange frames with other associated users. This kind of infrastructure is called Basic Service Set (BSS).

We refer to Independent Basic Service Set (IBSS) when no AP is included. In an IBSS, nodes can communicate directly with each other, if and only if they are in the mutual communication range, since no relay functions are defined for an IBSS. Anyway, the BSS is the most used type of infrastructure deployed: each station has to rely on an AP for communication, hence it needs a direct link to the AP in order to join the BSS. While the AP is stationary and has to relay on a power line, the wireless station may be battery powered and be free to roam.

A wireless link is defined as a single physical path over the Wireless Medium (WM). A wireless link may be established if two stations are within the mutual communication range, and it allows the two stations to exchange MAC Service Data Units (MSDUs). In 802.11, every successful transmission has to be acknowledged by the destination. Since the ACK is a shorter frame and is transmitted at a robust Modulation and Coding Scheme (MCS), its Packet Error Rate is much smaller than the usual frames. A correct transmission from A to B, then, requires correct reception of both the data frame (received from B) and the ack frame (received from A).

Therefore, when evaluating the performance of a wireless link $a \rightarrow b$, in terms of frame no-error transmission probability, the probability p of no-error transmission will be $p = p_f \cdot p_r$, where p_f is the probability of no-error transmission from a to b and refers to the correct transmission of the data frame, and p_r is the probability of no-error transmission from b to a and refers to the correct transmission of the ack frame.

The AP stations and their BSSs may be interconnected with each other through a Distribution Service (DS), forming an Extended Service Set (ESS). A typical set up is shown in Figure 2.1. The total area covered by the interconnected BSSs is called Extended Service Area (ESA). Users can move seamlessly within the ESA.

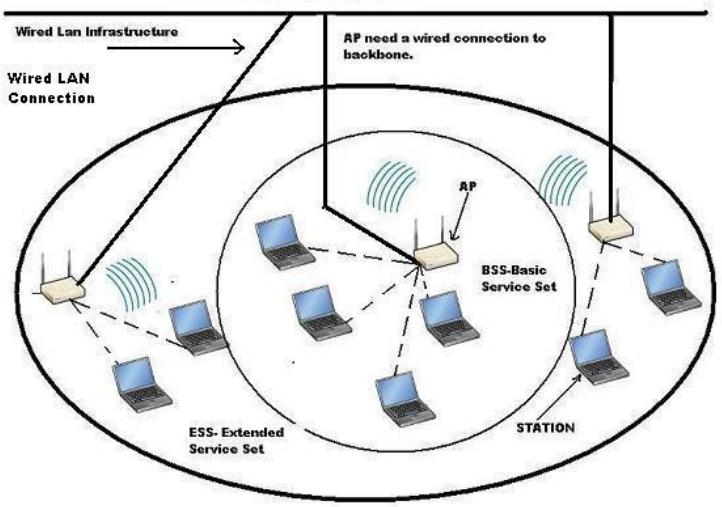


Figure 2.1: Traditional Wireless LAN deployment

The ESS appears as a single logical network at the Logical Link Control layer, hence APs can forward frames within the ESS. Users within the same ESS, then, belong to the same Local Area Network.

To build an ESS, APs use the Distribution System Service (DSS). APs relay on the Distribution System Medium (DSM) to provide the DSS. The DSM may not be a 802.11 based network. An example of DSM is a 802.3 based Local Area Network (LAN). Moreover, even if two APs are within mutual transmission range, APs do not use the WM to exchange frames, but they exploit the DSS. This may represent one big limitation of the 802.11 standard, since the APs need a separate infrastructure to communicate with each other. Building a 802.3 based, in fact, requires a wired network cabling of the area covered by the APs.

This may not only be unnecessarily difficult and expensive, but it may be also not possible. The wireless area coverage advantage is particularly interesting in the high client mobility situations. Use cases are the military field application, or the critical situations. For example, in the aftermath of an earthquake, or in a military battlefield, wired network cabling is not possible, and APs wireless direct communication is highly desirable.

Finally, APs often have bridge functionalities, in order to deliver MSDUs to

non 802.11 networks, allowing the AP to communicate with the 802.3 LAN that builds the DSM. For addressing stations in different BSSs, at the MAC-layer, the 802.11 standard introduces four new address fields:

- The Source Address (SA), which holds the MAC address of the station originating the frame.
- The Destination Address (DA), which refers to the final destination station address.
- The Transmitter Address (TA), which states the address of the station forwarding the frame.
- The Receiver Address (RA), which indicates the next intended receiver station in the ESS.

By filling the address fields, it is possible to forward packets between stations located in different BSSs. For this purpose, it is important to know the exact location of the wireless station. In particular, the DS needs to know where the destination station is, i.e., it needs to identify the AP to which the message should be delivered. Each wireless station in a BSS must maintain an association with an AP belonging to the same BSS. To this requirement, three services are introduced:

- Association, which is used to establish the initial association between station and AP; the station is then identified, together with its address. After the association phase the AP communicates the information to the other APs in the ESS;
- Re-Association, which is used to transfer the established association to another AP, allowing a device to move from one BSS to another;
- Disassociation, which is invoked when the station leaves the ESS or it is shut down.

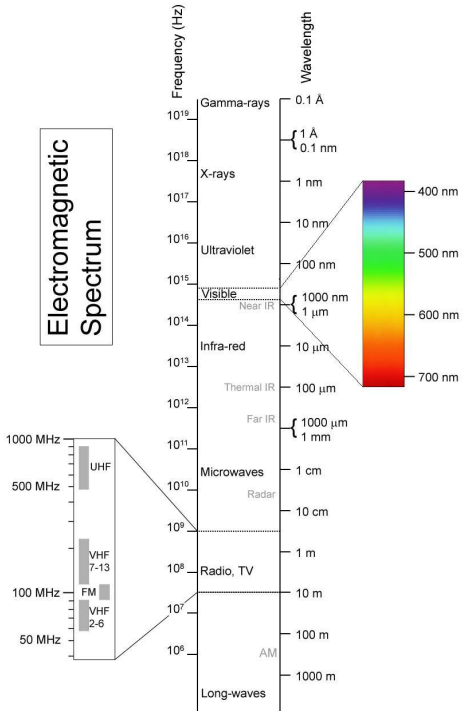


Figure 2.2: The electromagnetic spectrum

Media Access Control Mechanisms Outline

In the present wireless IEEE 802.11 standard, the widely used multiple access method is the Carrier Sense Multiple Access with Collision Avoidance (CSMA/CA), which is a modification of the Carrier Sense Multiple Access (CSMA) protocol. Under CSMA protocols, a carrier sensing scheme is used and the sender does not transmit the data if the channel is sensed busy.

In addition, the source also employs additional collision avoidance (CA) mechanisms. Collision avoidance is used to improve CSMA performance by not allowing wireless transmission of a node if another node is transmitting on the channel within the wireless range, thus reducing the probability of collisions. Moreover, the collision avoidance technique improves the performance of CSMA by attempting to divide the wireless channel equally among all transmitting nodes within the same collision domain.

The source node that is ready to transmit a packet picks a backoff value which

is uniformly distributed between $[0, CW_{min}]$, where CW_{min} is the minimum Contention Window size, and waits for the channel to be idle. If the channel is idle for a DCF Interframe Space (DIFS) period, the node starts decrementing its back-off value. The data frame is transmitted once the back-off value reaches zero. If the channel becomes busy while decrementing, the node pauses its backoff and resumes it once the channel appears again idle for DIFS period. If the source fails to get the ACK frame, it assumes a packet collision happened and it doubles the backoff window (Binary Exponential Backoff or BEB).

In the RTS-CTS mode of IEEE 802.11, the source and destination nodes observe a 4-way handshake (including ACKs) to avoid collision and to help solving the hidden terminal problem that often arises in wireless networks. The hidden terminal problem may originate when a node A is within the range of the receiver node R but not within the range of the sender S. Therefore, node A cannot hear the transmission from S to R and may start transmitting. This, in turn, may result in a packet collision which prevent R from correctly decoding the signal transmitted from S.

Then, CSMA/CA can be supplemented by the exchange of a Request to Send (RTS) packet, sent by the sender S, and a Clear to Send (CTS) packet, sent by the receiver R. Node A, then, can hear the CTS packet from R and will refrain from transmitting for a given time. The amount of time the node should wait before trying to get access to the medium is included in both the RTS and the CTS frame. In general, all nodes within the same range of node S or R will hold back from transmitting for the duration of their transmission.

Physical Specifications Outline

At the Physical layer, the 802.11 family includes several modulation techniques. The most popular are those defined by the 802.11b/g and 802.11a protocols. These protocols identify a restricted portion of the microwave spectrum (see Figure 2.2) and within the defined portion, they identify a discrete number of frequencies on which the transmission can be modulated. The number of the available frequencies define the number of the available transmission channels.

channel	frequency (MHz)	North America	Japan	Most of world
1	2412	Yes	Yes	Yes
2	2417	Yes	Yes	Yes
3	2422	Yes	Yes	Yes
4	2427	Yes	Yes	Yes
5	2432	Yes	Yes	Yes
6	2437	Yes	Yes	Yes
7	2442	Yes	Yes	Yes
8	2447	Yes	Yes	Yes
9	2452	Yes	Yes	Yes
10	2457	Yes	Yes	Yes
11	2462	Yes	Yes	Yes
12	2467	No	Yes	Yes
13	2472	No	Yes	Yes
14	2484	No	Yes	No

Figure 2.3: List of 802.11b/g available channels

Each radio can transmit on one of the available frequencies, but its transmitting power level has to respect a standardized power mask, i.e., the band occupied by the transmitted signal has to be symmetrical bounded around the central frequency. The power mask is the tool employed to tailor the available spectrum, obtaining a fixed number of channels, all with the same fixed available band.

In particular, the 802.11b and 802.11g protocols use the 2.4 GHz band. Since this band is used by many other wireless devices, like microwave ovens, cordless telephones, video senders, baby monitors, wireless keyboards and Bluetooth devices, 802.11b/g equipment may occasionally suffer interference, hence performance degradation.

Without entering into details, the 802.11b/g protocols use direct-sequence spread spectrum (DSSS) and orthogonal frequency-division multiplexing (OFDM) signaling methods, respectively. The 802.11b, exploiting the CCK/Barker PSK-based modulation scheme, can reach a maximum raw data rate of 11Mb/s, while the 802.11g standard operates at a maximum bit rate of 54Mb/s. The two modulation schemes are fundamentally different and efforts have been directed for letting 802.11b and 802.11g devices work well together.

The 802.11a protocol uses the 5 GHz band, which offers at least 12 non-overlapping channels rather than the 3 offered in the 2.4 GHz frequency band.

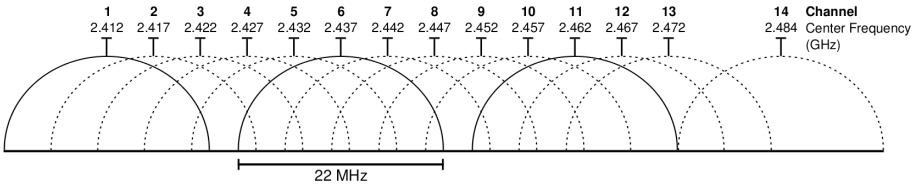


Figure 2.4: Wi-Fi channels in 2.4 GHz band

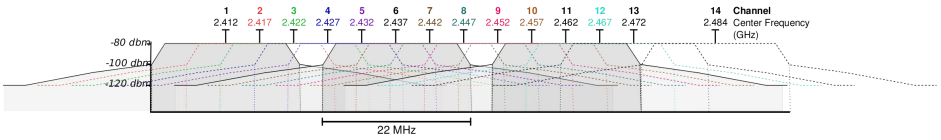


Figure 2.5: Wi-Fi channels in 2.4 GHz band

Moreover, the 802.11a reaches the 54Mbit/s maximum data rate, exploiting an OFDM based modulation technique.

The segment of the radio frequency spectrum used by 802.11 varies between countries. In the US and Europe, 802.11a and 802.11g devices may be operated without a license. Frequencies used by channels one through six of 802.11b and 802.11g fall within the 2.4 GHz amateur radio band.

In 802.11 standard, each of the available bands are divided into channels, analogously to how radio bands are sub-divided but with greater channel width and overlap. For example, in Figure 2.4, we can see how the $2.4000 - 2.4895$GHz band is divided into 14 channels, each of 22MHz width, spaced 5MHz apart. In Figures 2.3 and 2.6 we show the wifi channels availability in the 802.11b/g and 802.11a bands respectively, in different world regions. Channels availability is separately regulated by each country, and it depends on how each country decides to allocate the available radio spectrum to various services. For example, while Japan allows the use of all the 14 channels (with the exclusion of 802.11g from channel 14), Spain initially allowed only channels 10 and 11 and France allowed only 10, 11, 12 and 13 (they now both follow the European model, i.e., allowing channels 1 through 13).

Besides specifying the central frequency of each channel, 802.11 also specifies

channel	frequency (MHz)	United States	Europe	Japan		Singapore	China	Israel	Korea	Turkey
		40/20 MHz	40/20 MHz	40/20 MHz	10 MHz	20 MHz	20 MHz	20 MHz	20 MHz	20 MHz
183	4915	No	No	No	Yes	No	No	No	No	No
184	4920	No	No	Yes	Yes	No	No	No	No	No
185	4925	No	No	No	Yes	No	No	No	No	No
187	4935	No	No	No	Yes	No	No	No	No	No
188	4940	No	No	Yes	Yes	No	No	No	No	No
189	4945	No	No	No	Yes	No	No	No	No	No
192	4960	No	No	Yes	No	No	No	No	No	No
196	4980	No	No	Yes	No	No	No	No	No	No
7	5035	No	No	No	Yes	No	No	No	No	No
8	5040	No	No	No	Yes	No	No	No	No	No
9	5045	No	No	No	Yes	No	No	No	No	No
11	5055	No	No	No	Yes	No	No	No	No	No
12	5060	No	No	No	No	No	No	No	No	No
16	5080	No	No	No	No	No	No	No	No	No
34	5170	No	No	No	No	No	No	Yes	Yes	Yes
36	5180	Yes	Yes	Yes	No	Yes	No	Yes	Yes	Yes
38	5190	No	No	No	No	No	No	Yes	Yes	Yes
40	5200	Yes	Yes	Yes	No	Yes	No	Yes	Yes	Yes
42	5210	No	No	No	No	No	No	Yes	Yes	Yes
44	5220	Yes	Yes	Yes	No	Yes	No	Yes	Yes	Yes
46	5230	No	No	No	No	No	No	Yes	Yes	Yes
48	5240	Yes	Yes	Yes	No	No	No	Yes	Yes	Yes
52	5260	Yes	Yes	Yes	No	No	No	Yes	Yes	Yes
56	5280	Yes	Yes	Yes	No	No	No	Yes	Yes	Yes
60	5300	Yes	Yes	Yes	No	No	No	Yes	Yes	Yes
64	5320	Yes	Yes	Yes	No	No	No	Yes	Yes	Yes
100	5500	Yes	Yes	Yes	No	No	No	No	Yes	No
104	5520	Yes	Yes	Yes	No	No	No	No	Yes	No
108	5540	Yes	Yes	Yes	No	No	No	No	Yes	No
112	5560	Yes	Yes	Yes	No	No	No	No	Yes	No
116	5580	Yes	Yes	Yes	No	No	No	No	Yes	No
120	5600	No	Yes	Yes	No	No	No	No	Yes	No
124	5620	No	Yes	Yes	No	No	No	No	Yes	No
128	5640	No	Yes	Yes	No	No	No	No	Yes	No
132	5660	No	Yes	Yes	No	No	No	No	No	No
136	5680	Yes	Yes	Yes	No	No	No	No	No	No
140	5700	Yes	No	Yes	No	No	No	No	No	No
149	5745	Yes	No	No	No	Yes	Yes	No	Yes	Yes
153	5765	Yes	No	No	No	Yes	Yes	No	Yes	Yes
157	5785	Yes	No	No	No	Yes	Yes	No	Yes	Yes
161	5805	Yes	No	No	No	Yes	Yes	No	Yes	Yes
165	5825	Yes	No	No	No	Yes	Yes	No	Yes	Yes

Figure 2.6: List of 802.11a available channels

a spectral mask defining the permitted distribution of power across each channel. The power masks defined for the 802.11b/g radios are shown in Figure 2.5. Since channels are 22Mhz wide, the mask requires that the signal must be attenuated by at least 20dB from its peak energy at 11MHz from the central frequency.

From Figure 2.4, we can see that stations can only use channels 1, 6, 11 and 14 without overlapping. In fact, it is often assumed that the power of the signal in a channel band extends no further than 11Mhz from the central frequency.

The presence of the orthogonal channels is fundamental for the interference reduction.

Each radio interface has several configuration parameters, from frequency channel to raw transmission data rate. If we model the interference through the physical model, we consider a transmission to be successful if the $SINR$ measured at the receiver is greater than a given threshold γ_{r_m}. The $SINR$ equation is reported in equation 2.1.

$$SINR_{uv} = \frac{G_{uv}P_u}{\sum_{x \to y \neq u \to v} G_{xv}P_x + n_v} \geq \gamma_{r_m} \tag{2.1}$$

In equation 2.1, we have contribution of transmission power P_u, gain factor G_{uv}, the noise n_v. The term at the denominator, $G_{xv}P_x$ takes into account all the active links using the same channel as link $u \rightarrow v$, i.e., interference signals are considered in the same way as noise signals. The signal received from the generic node x, then, interferes with the signal from u and the final effect is a decrease in the overall $SINR$ at node v.

Starting from Figure 2.4, we try to exploit the non-interfering channels availability. Radios transmitting on orthogonal channels will not interfere with each other, hence they can be active simultaneously. A generic node y transmitting on an orthogonal channel will not affect the $SINR$ on node v, hence will avoid interference. If link $u \rightarrow v$ is set on channel 1, a possibility would be to assign orthogonal channels to all the nodes communicating in the same interference range of node v. With a proper configuration, if the number of orthogonal channels is sufficient, then, the interference is defeated and all the nodes in the same interference region can transmit without obstructing one another.

2.2 Wireless Mesh Networks

Multi-radio Wireless Mesh Networks (WMNs) are introduced to provide high-speed broadband connectivity in both rural and urban areas, to both fixed and mobile users, with low operational management costs, easy network maintenance and high robustness.

We have seen in Section 2.1, that, in case of traditional 802.11, high speed ubiquitous wireless connectivity could be obtained through a dense Access Points deployment. In such a scenario, while APs may be cheap, the wired infrastructure needed for APs interconnection is expensive and often impracticable. To cut down costs and to overcome practical deployment issues, then, the solution adopted in the WMNs is to choose wireless APs interconnection.

Wireless Mesh Networks come out as an evolution of MANETs (Mobile Ad-hoc Networks). Ad hoc networks consist of completely wireless nodes with high mobility, and are characterized by total lack of backbone infrastructure. Moreover,

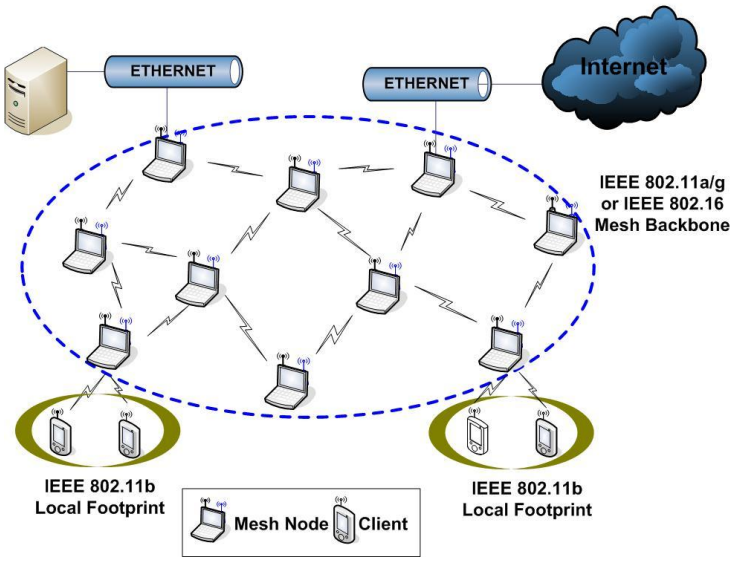


Figure 2.7: Typical Wireless Mesh Networking

mobile nodes in MANETs can operate both as a host or as a router, i.e., they can forward packets on behalf of nodes that are not in the same range as their destinations. On the other hand, mesh networks consist of fixed or mobile wireless nodes connected by multi-hop wireless links, and a complete absence of wired infrastructure. While in standard WLANs, AP stations accomplish a central role in the overall network structure, the WMN is fully distributed.

Mesh nodes can be of two kinds: client or router. Mesh routers form the mesh network backbone, they have minimal mobility and provide access for the clients, similarly to a classical AP. This is a typical WMN configuration and it is called Backbone WMN configuration. In Figure 2.7 we show a small wireless mesh network topology example. Beyond conventional access points typical functionalities, e.g. bridging/gateway functionalities, mesh routers can relay on multi-hop wireless communication links to reduce the transmitting power, while having the same area coverage. Therefore, they can forward and send frames over multiple hops, i.e., they are able to communicate with other mesh routers outside the mutual transmission range.

If mesh routers density is sufficiently high, each mesh router may establish

several wireless connections with many different routers, thus providing robustness against nodes and links failures. Mesh routers, then, create an auto-configured backbone network and supply network access to mesh clients; moreover, while gateway routers supply Internet connectivity to the mesh nodes, bridging functionalities are needed to integrate heterogeneous wireless networks. It is common that different radio technologies are installed on mesh routers: an highly directive antenna is used for long range communications and backbone connectivity, and an isotropic antenna is often used for client connectivity. On their side, mesh clients have simpler software and hardware platforms, they do not have gateway or bridging functionalities and they have to relay on mesh routers in order to obtain network access. Finally, while mesh routers are essentially stationary, mesh clients are more flexible and can be installed in mobile terminals.

Wireless Mesh Networks are currently under definition in the IEEE 802.11 Task Group "S", [23]. IEEE 802.11s develops the Wireless Mesh Network amendment, which is still not finalized. The basic element in the 802.11s amendment is the Mesh Point (MP). Unlike 802.11 devices, MPs are able to exchange frames with other MPs outside the mutual transmission range, through multiple wireless hop links. A MP can establish a Mesh Link (ML) with any mesh device in its transmission range to which an 802.11 link exists. In particular, the mesh device to which an MP has established a Mesh Link with, is denoted as peer MP. In order to set up an ML, two MPs have to perform an 802.11s peer link management protocol over the 802.11 wireless link. The concatenation of Mesh Links defines a Mesh Path.

Each MP, then, can act as a traffic source or destination, or it can just forward packets on behalf of other nodes. In the 802.11s amendment a Mesh Basic Service Set (MBSS) is defined, with the same Physical layer specifications defined in the IEEE 802.11 standard and introducing some extensions to the MAC layer. MBSSs support broadcast and unicast messages forwarding in auto-configurable multi-hop wireless networks. Like in the 802.11 BSS, membership in a MBSS does not imply that direct transmission is possible with all the users in the MBSS. Differently from the 802.11 BSS, though, in a MBSS the communication with

other MPs in the same MBSS is possible as long as a Mesh Path exists between the two endpoints.

The main objective of 802.11s is to define a layer-2 domain that supports higher layer protocols in a complete transparent way. Hence, path selection, packet forwarding and routing techniques are all defined at the Data Link layer. Moreover, 802.11s has to integrate the IEEE 802 standards, i.e., it has to support unicast, multicast and broadcast traffic. To this purpose, the wireless mesh standard 802.11s introduces the Mesh header field.

The Mesh header field has a minimum length of 4 bytes and a maximum length of 16 bytes. The first byte contains the Mesh Flags. In particular, the first bit is the Address Extension (AE) flag field, while the remaining bits are reserved. When the AE flag is set, the MP uses a six-address scheme. The second byte indicates the Mesh TTL: every node forwarding the current frame has to decrement such value in order to avoid endless routing inside the network. The third and fourth byte are used for Mesh End-to-End sequence numbering, as the 802.11 sequence control field is set hop by hop. The Mesh End-to-End sequence number is used to eliminate duplicate packets and to avoid unnecessary retransmissions.

Finally, we give a quick overview of the 802.11s Link Management and Path Selection solutions. Mesh Points may use active (transmission of probe frames) or passive (listening for beacon frames) peer discovering. When a Mesh Link is established, a cost c_a is computed and associated with the new ML. In particular, the c_a cost is computed as follows:

$$c_a = [O_{ca} + O_p + \frac{B_t}{r}] \cdot \frac{1}{1 - e_f}$$

where:

- O_{ca} is the channel access overhead which depends on the Modulation and Coding Schemes;
- O_p is the Protocol overhead;
- B_t is the number of bits that have been transmitted in the test frame;

- r is the bit rate;
- e_f is the frame error rate for the current test frame.

This link metric, called airtime link metric, is mandatory, but it is possible to include other vendor specific link metrics. Airtime link metric is used by the path selection algorithms to compute the best source destination end-to-end path. Path selection algorithms are layer-2 algorithms, since they find the best route working at layer-2. Despite the fact that many path selection algorithms may be implemented in a MBSS, the 802.11s standard defines one path selection protocol that has to be implemented by all the MPs. The default protocol is the Hybrid Wireless Mesh Protocol (HWMP).

The HWMP protocol relies on three main MAC Management protocol data units (MMPDUs):

- Path Request (PREQ);
- Path Reply (PREP);
- Path Error (PERR).

Each path message to be forwarded, has a path cost value that is updated hop by hop, by adding the actual air link metric to the current path cost value. Moreover, the TTL path message value is decremented (this value is independent of the TTL field in the Mesh MAC header).

The HWMP protocol has three different operative modes:

- An on demand path selection procedure, which is similar to those adopted by the Ad-hoc On-demand Distance Vector (AODV) routing protocol [24];
- A tree-based path selection procedure, which consists in building a tree structure where a specific MP in mesh network is chosen as root. The MP root has to:
    1. establish paths with all the MPs in the network, by proactively sending PREQ messages to all the MPs;

2. send Route Announcement (RANN) messages, to allow MPs to build an on demand route path toward the root;

- Null path selection, indicating that the MP is not enabled to frame forwarding.

The tree-based path selection is often preferred when the direct path is not known. While the on demand path selection algorithm is invoked, and the path discovery procedure is still running, it is possible to forward packets on the tree-based path, through the MP root. When the direct path is found by the on demand protocol, then, packets can be forwarded more efficiently along the new path.

Regarding the pro-active path selection phase, the tree structure is built as follows: first, the MP elected as root sends a broadcast PREQ message; this PREQ message requires a PREP message response from the contacted MPs. Then, once the PREP frame is received, the path is established with the responding MP. There is also the eventuality that the MP node spontaneously contacts the root MP. The MP updates its path table if

- the PREQ message contains a more recent sequence number, or
- there is a new path having a lower path metric value, with a proper sequence number.

Finally, the RANN message sent by the MP root is used to update each MP with the root identification.

2.3 Typical Issues in Wireless Mesh Networks

In Section 2.2, we have seen how mesh routers are able to establish direct wireless links with other mesh routers that are in the same transmission range area. Moreover, they can exchange packets with other mesh nodes outside the mutual transmission range by entrusting packets to intermediate nodes, through wireless multi-hop frame forwarding.

The absence of any wired network infrastructure makes the WMNs an affordable and manageable solution. Multi-hop wireless communication links, though, are subject to environmental noise and interference problems.

In this Section, we analyze the obstacles in the wireless mesh network optimization, together with the main challenging research issues. Starting from the importance of an appropriate channel assignment algorithm, we focus on the optimization of the traffic paths and finally outline the need for a proper overlay network construction in the context of a Wireless Mesh Network.

2.3.1 The need for a proper channel assignment

In Section 2.1, we have shown that for each wireless link $u \rightarrow v$ set on channel c, if the $SINR$ value at v is below a given threshold, then the information signal is not recognizable anymore. From equation 2.1, it turns out that all the interfering transmissions on the same channel c contribute to decrease the $SINR$ at node v.

The only way to reduce the interference problem is to set all the interfering mesh radios to orthogonal channels. Mesh routers can set their operational radio frequencies in the 802.11a or 802.11b/g band: this means that 3 or 12 orthogonal channels are available, respectively. If we had just one radio interface available on each node, all the routers in the mesh network should transmit at the same frequency, i.e., they can only exploit one channel. Otherwise, different radio frequencies would lead to disconnected subsets of nodes, since nodes can communicate with each other only if a common channel is available.

Due to the availability of cost-effective radio interfaces, a possible solution to the interference problem comes with equipping mesh routers with several radio devices. With more than one available radio per node it is possible to better exploit the available spectrum, by accurately setting different radios to different channels. With this solution, routers can transmit or receive simultaneously on different frequencies. In order to best exploit all the available spectrum, then, it is crucial to correctly assign the available channels to each radio device. One challenging issue related to the configuration of Multi-Radio WMNs is the channel assignment problem.

Assigning the channel to a wireless link, though, has also the effect of defining the available bandwidth on that link. The available bandwidth on a link, in fact, is a portion of the nominal capacity. This portion is as smaller as the number of links sharing that particular channel is bigger.

For example, we may consider two links $u \rightarrow v$ and $x \rightarrow y$, both set on channel c and close enough to be in the same interference range. We can also hypothesize that we have such a parameters configuration that the two links cannot be active at the same time. Otherwise, the drop in $SINR$ at receivers will avoid correct information reception. Hence, only one node at a time can be active between u and x.

If we denote as C the channel capacity and as f the traffic flow that is transmitted by the radio x, in an observation interval T we have that $C = \frac{f}{T}$. If another radio u has to transmit, it will have to contend the wireless medium with the radio x. Assuming contention fairness, each radio will have roughly half the time it had to transmit when it was all alone. This means that in the same observation interval T the two radios will have the chance to transmit a traffic f', roughly half the traffic they would transmit if there were no contention. In this case, the channel capacity C' for the radio x we will be $C' = \frac{f'}{T} \approx \frac{\frac{f}{2}}{T} = \frac{C}{2}$.

Then, if two links are configured to communicate on the same channel and are close enough, interference will arise and the interfering links will share the channel capacity. Interfering links have then just a portion of the total nominal bandwidth: the channel assignment operation determines the number of links sharing the same channel and, consequently, the available bandwidth on a link. It is straightforward that the links carrying most of the traffic should have most of the bandwidth, i.e., they should be assigned to the least busy channels in order to minimize the interference.

It goes without saying, then, that the assignment of the available radios has to be tailored to the current traffic routing.

2.3.2 How to properly route traffic

Together with the channel assignment problem, another challenging research issue is the routing problem, i.e., the problem to find a set of traffic flow rates for all the links in the network that optimizes a given objective. Typical optimization objectives may be the maximization of aggregate network throughput, minimization of end-to-end delay or minimization of a given cost function.

Finding the correct traffic rate for each link is then a key step for the optimization of the overall network. We can see that the channel assignment problem and the routing problem are related to each other. Finding the correct traffic rate for each link means that the knowledge of the available bandwidth is necessary. In wireless mesh networks, though, the available bandwidth on a link depends on the number of links sharing that particular channel, which is a direct consequence of the channel assignment phase. But this can also be seen the other way around. If the pre-computed traffic flow rates are predicted on a particular link, then the channel assignment should be aware of this information. In particular, the channel assignment should set up the available channels in such a way that, on each link, the available bandwidth exceeds the nominal flow rate. But the flow rate information can be available only after the routing problem is solved. It is clear, then, that both problems are related and they have to be solved jointly.

The joint channel assignment and routing problem is NP-complete [25], and this means that an heuristic for the joint channel assignment and routing problem is needed. A common approach is to first solve the routing problem, i.e., to determine the amount of traffic flow to be routed on each link, and then solve the channel assignment problem, i.e., to assign the available channels in such a way that the resulting available bandwidth exceeds the traffic flow rate on each link. With this approach we relate the channel assignment to the flow rates. Starting from a source-destination traffic matrix, once the flow rates are found, we can tailor the channel assignment to the newfound flow rates, finally shaping all the configuration parameters of the wireless mesh network to suit the traffic flows. When the traffic sources start transmitting, the traffic, though, is still routed by the routing algorithm along the minimum cost path. If all the links have unitary costs,

the routing algorithm will always route packets along the minimum hop-count path. The routing path, then, will always be the same, regardless of the nominal flow rates that are the solution of an optimization problem.

What we would like to do, instead, is to use the wireless links as returned by the solution of the optimization problem. If we see the nominal flow rates as the nominal average utilization of the wireless links, then, we would like to enforce on each link an average throughput as close as possible to the nominal utilization.

A further optimization problem consists in realizing an average link utilization as close as possible to the traffic flow rates returned by the solution of the optimization problem. This problem may require changes to the routing algorithms and to the way the routing paths are chosen.

2.3.3 Overlay network traffic management

In Section 2.3.2 we pointed out that knowledge of source-destination couples is fundamental to achieve a given optimization objective and solve the optimization problem. Then, the traffic matrix is required for the routing problem solution, and the wireless mesh network configuration has to suit the traffic dynamics, trying to adapt to the new traffic patterns.

Recently, an ever growing portion of network traffic is generated by users that share their own resources through Peer-to-Peer protocols. It has been estimated that Peer-to-Peer traffic represents more than 50% of the overall Internet traffic. When dealing with wireless mesh networks, link traffic knowledge is fundamental for a proper link configuration and channel assignment phase. Since Peer-to-Peer traffic represents a large portion of total traffic, this means that an information exchange between Peer-to-Peer application clients and mesh routers concerning the amount of traffic may be needed for an optimal network configuration.

Moreover, Peer-to-Peer applications tend to build an overlay network among all the peer nodes. In this new overlay network, Peer-to-Peer applications usually impose their own overlay routing strategies, based on their own metrics, and they tend to establish connections with subsets of overlay nodes. The P2P traffic pattern, then, is a direct consequence of particular decisions taken at the application

layer.

The overlay topology building phase and routing are totally unaware of the underlying wireless network, and metrics for the overlay links are picked by measuring the wireless network performance by means of a set of tools. Some overlay applications also independently perform active measurements that allow them to continuously monitor dynamic parameters like loss and available bandwidth. Direct consequence of these performance measurements is the generation of overhead traffic, which leads to an increase in the overall network traffic.

Moreover, peer nodes in an overlay network exchange most of the data with their overlay neighbourhood, which is a subset of the available peers in the overlay network. The neighbourhood of a given peer is chosen randomly, and this may lead to other inefficiencies.

For example, the overlay neighbours of a Peer-to-Peer node may be several hop away in the physical network, even outside the mesh network or the provider network. This may lead several links or gateway nodes to be traffic overloaded. As a consequence, the actual wireless network configuration may not be suited for these new traffic flows. Another challenging issue, then, is the overlay/underlay routing interaction in the context of wireless mesh networks. A new architectural model strongly based on the exchange of information between overlay and wireless mesh network is needed. The main objective is to guide the creation of the overlay networks, trying to optimize it in order to control the Peer-to-Peer traffic and to adapt it to the wireless mesh network requirements.

2.4 Chapter Conclusions

In this Chapter we presented a quick overview of the 802.11 wireless architecture. We presented the basic wireless devices and underlined their typical star shaped network structure, together with the limits of this kind of infrastructure. Then, we introduced the 802.11s standard, presenting the main features of the new wireless mesh standard.

Something new in the 802.11s standard is the focus on the layer-2 function

implementations: path selection, and routing techniques have been shifted at the layer-2 of the ISO/OSI protocol stack. With particular attention to the wireless mesh networks, we focused on what are the main challenges when trying to optimize a given objective function. The wireless link is characterized by several configurable parameters, like the transmission channel central frequency, the transmission rate and the transmission power.

The performance of a wireless link, in terms of available bandwidth, is dictated by the proper configuration of all the typical wireless parameters. The current wireless link configuration and the available bandwidth, then, have to be related to the actual traffic that has to be routed on the links. For example, the bandwidth available on a link is just a fraction of total channel capacity and it depends on the number of links sharing the same channel. Hence, the wireless network has to accurately choose the wireless links parameters, in order to adapt their features to the traffic flows.

Moreover, we outlined how an ever growing portion of current Internet traffic is represented by P2P applications, which are able to organize themselves and manage their traffic through the creation of overlay networks. We pointed out how P2P traffic flow characteristics depends on these overlay networks, and how the interaction with the physical network is the source of management inefficiencies. Then, a way to tame P2P traffic pattern, reducing inefficiencies, is to somehow control the overlay network, i.e., to influence application decisions, through a cross-layer information exchange.

The remainder of the thesis is organized in two Parts, each focused on a distinctive optimization issue arising in wireless mesh networks.

In detail, in Part I we are interested in the traffic flow optimization problem, focusing on the channel assignment and routing problems. We try to best adapt the wireless link configuration, in terms of channels assigned and routing paths chosen, to the optimal traffic flow configuration. In Part II, we analyze the P2P traffic dynamics and its relations with the decisions taken at the application layer, in particular regarding the overlay set-up procedure. With this basis, then, we focus on the Peer-to-Peer traffic optimization, trying to adapt the traffic flow to

the current mesh network configuration.

Part I

Traffic flow optimization in Wireless Mesh Networks

Chapter 3

The Channel Assignment Problem

In Chapter 2 we described the structure of the wireless mesh networks and in Section 2.3 we presented some of the critical issues in management and configuration of a wireless mesh network.

The use of wireless links has the consequence that communication is affected by interference due to links using the same transmission channel. Interference may occur between different hops of the same path or between two different neighbouring paths. For high density wireless mesh networks, interference drastically degrades network throughput, inducing very low network performance. Mesh wireless communication, in particular, takes place over the communication channels defined by the IEEE 802.11a or 802.11b/g standards. If channels are orthogonal, i.e., their operational frequencies are far enough that the communication bands do not overlaps, simultaneous transmissions are possible and interference is avoided. In particular, IEEE 802.11b/g and IEEE 802.11a standards define 3 and 12 orthogonal channels.

Usage of different orthogonal channels on wireless links belonging to the same interference region is the best way to cope with the interference issue. If mesh nodes are equipped with a single radio, though, using different channels is possible only by performing dynamical radio switching operations. In fact, the wireless network would result partitioned in subset of nodes and communication would be possible only between the nodes sharing the same channel. A possible solution to use multiple channels within single radio networks would be to frequently switch

the radio interfaces to the appropriate communication channel.

In order to correctly perform switching operations, thus avoiding the deafness problem, i.e., transmitter and receiving radio interfaces being on different channels, fine-grained synchronization between nodes is required. This solution would require some compromises that may prevent the practical implementation: synchronization between nodes may be hard to enforce and synchronization errors may arise; moreover, frequencies switching times may disrupt the current network operations and prevent most real-time applications from working correctly.

Thanks to the availability of cost-effective radio interfaces, endowing mesh routers with multiple radio interfaces is a recent solution to improve performance in a wireless mesh network. In multi-radio wireless mesh network, no channel switching is needed, radio interfaces may be configured on different orthogonal channels and simultaneous transmissions over different channels are possible. This results in reduced interference and increasing in the overall network throughput.

One of the main challenges in configuring a multi-radio wireless mesh network is the assignment of the available channels to the radio devices, given the variety of objectives that can be pursued and the computational complexity of the resulting problems.

Moreover, the channel assignment problem has been revealed to be interdependent with the routing problem, and a joint solution of both problems is required. In fact, the channel assignment determines the links sharing the same channel, and, ultimately, their available bandwidth. To solve the routing problem, i.e., finding the set of flow rates for every link that optimizes a given objective, the available bandwidth returned by the channel assignment is required. We would have this information only after solving the channel assignment problem.

On the other side, the channel assignment algorithm needs to be aware of the traffic flows expected on the links. With such information, the channel assignment algorithm can ensure that on each link the available bandwidth exceeds the nominal traffic flows that have to be routed. However, this information is available only after the solution of the routing problem.

The two problems are then interdependent and have to be solved jointly. The joint problem, though, has been shown to be NP-complete [25], and mainly heuristics have been proposed. A possible solution to the joint problem may consist in three steps:

1. **Solve the routing problem to obtain the pre-computed flow rates:** For each network link, a set of pre-computed flow rates is computed, based on a given optimization function. Since we do not have a channel assignment, we may either assume that only one channel is available, or that there is no interference. Both assumptions are valid. The former assumption, though, does not take into account the possibility of simultaneous transmissions over links that are assigned orthogonal channels. Therefore, while the former assumption may lead to underestimate the flow rates, the latter assumption may lead to overestimate them, since the interference may prevent simultaneous transmissions over the interfering links.

2. **Solve the channel assignment problem:** Taking into consideration the pre-computed flow rates obtained in the step 1, channels are assigned in the attempt to verify the scheduling condition for the flow-rates.

3. **Adjust the pre-computed flow rates:** Given the channel assignment returned by the step 2, if flow rate scheduling cannot be achieved, the pre-computed flow rates are adjusted in order to obtain a feasible set of flow rates.

In particular, we will assume that the set of pre-computed flow rates is given, i.e., they have been computed based on an optimization objective and they are supplied as input to the channel assignment algorithm.

Moreover, in this Chapter, we will formally define the problem to find, if exists, a channel assignment which guarantees the scheduling of the set of pre-computed flow rates. This problem is NP-complete, too, and we develop greedy heuristics having the property that the returned channel assignment is invariant for scaling of the pre-computed flow rates.

Finally, we show that typical wireless links parameters like transmission rate and transmission power affect the performance of the channel assignment algorithm, reducing the interference level perceived by links sharing the same channel, thus helping in scheduling the flow rates.

In Section 3.1 we formalize the channel assignment problem and provide a sufficient condition for the scheduling of the pre-computed flow rates. Then, in Section 3.2 we present the state of the art on channel assignment algorithms. Finally, in Section 3.3 and Section 3.4 we show two different approaches to channel assignment.

Often, when approaching the channel assignment problem, the peculiar features of the wireless networks are disregarded, i.e., the wireless links are considered just like wired links. Our aim is to correctly characterize the wireless links and its peculiar features. In both the proposed approaches, we consider the impact of typical wireless radio parameters, like transmission rate and transmission power, on the performance of the channel assignment.

In Section 3.3, we describe a centralized static channel assignment algorithm that is designed to reduce as much as possible the interference inside the wireless network, while adapting the wireless parameters to the pre-computed set of flow rates. The proposed algorithm assigns channels from the very beginning, at the set-up phase of the wireless network, when all the radios have yet to be configured. Basically, it has no time or complexity constraints, and it is run one time only, or on very large time scales. We present, then, a channel, rate and power assignment algorithm and evaluate its performances.

In Section 3.4, instead, we consider a channel re-assignment algorithm. Since the channel assignment depends on the set of pre-computed flow rates, when the flow rates change, the channel assignment should be recomputed. Hence the need for adapting channels to the new flow rates.

Frequent channel assignment re-computations, though, are not desirable and the channel assignment proposed in Section 3.3 may not be suitable in this situations. In fact, a new execution of the channel assignment procedure does not take into consideration the current channel assignment and thus will likely return

a completely different assignment of channels with respect to the current one. Enforcing the new channel assignment will eventually require changing the channels assigned to several radios, and such a procedure may disrupt the normal operation of the WMN for the time required to establish the new assignment.

Unlike the algorithm presented in Section 3.3, then, the channel reassignment algorithm we present in Section 3.4 is designed to adapt to the traffic flow variations. We present a simple heuristic that takes the current channel assignment into account and aims to adjust the minimum number of channels in order to cope with a variation of the set of pre-computed flow rates in the best possible way. It is thought to be run on much smaller time scales and its main requirements are low execution times and low computational complexity.

Since the algorithm is planned to adapt the current static channel assignment to the new traffic flows, it has to reach a good compromise between interference reduction and number of radio reconfigurations needed. This was not required for the static algorithm, which could then focus on best adapting channels to the flow rates.

Finally, the two algorithms reflect two different needs, they both have raison d'etre, and they are complementary. They have different time scales and they should be both used to adapt the wireless network to the transient traffic pattern. While the algorithm in Section 3.3 is deployed at the setup time and it is then eventually rerun at large time scales, the adaptable channel assignment proposed in Section 3.4 may be deployed on smaller time scales to reasonably cope with traffic variations.

3.1 System Model

In this Section we describe the model we used to formalize the channel assignment problem and we introduce notations that will be referred to all along the Chapter.

We assume that each mesh router u is equipped with $k(u) \geq 1$ radio interfaces and there are $|\mathcal{C}|$ available channels. For every radio, we assume a variable transmission power and rate. In particular, the transmission rate can be selected in the

(increasingly) ordered set $\{r_m\}_{m=1}^{M}$, while the transmission power can be selected in the set $\{w_m\}_{w=1}^{W}$

The impact of the interference can be formally accounted for through either one of the interference models: the *protocol* model that assumes interference to be an all-or-nothing phenomenon and the *physical* model of interference, which considers a transmission successful if the Signal-to-Interference and Noise Ratio (SINR) at the receiver is sufficiently high to decode the signal.

With the physical model, then, interference is modeled as noise. We have a given threshold at the receiving node, and the transmission is successful if the SINR exceeds such threshold, so that the signal may be correctly extracted with an acceptable Bit Error Probability (BER).

With the protocol model, instead, the transmission is successful when the receiving node is inside the transmission range of the sender, and outside the interference range of other nodes. While the transmission range may be set accordingly with the SINR thresholds, the interference range definition is vague. Under the protocol model, interference is then a binary process and it occurs when the node is within interference range of an active node. Clearly, the binary decision is not accurate and it can hardly model the real interference mechanisms, being too conservative.

For example, under the protocol model, a node inside the interference range of an active node cannot correctly receive data from the intended transmitter. This may be overly conservative, since the signal may be correctly decoded if the SINR is greater than the threshold. On the other hand, if a node is outside the interference ranges, small interferences may still be present and the overall effect may not be negligible.

In the light of these considerations we will adopt the *physical* model. In particular, the SINR at receiver v when a signal is transmitted by u is defined as:

$$SINR_{uv} = \frac{G_{uv}P_u}{\sum_{x \to y \neq u \to v} G_{xv}P_x + n_v} \tag{3.1}$$

where P_u is the power emitted by u to transmit to v, G_{uv} is the gain of the radio channel between u and v, and n_v is the thermal noise at receiver v. The

sum $\sum_{x \to y \neq u \to v} G_{xv} P_x$ is the interference signal power extended to all the signals coming from the generic node x, but node u. Again, G_{xv} is the gain, P_x is the transmitting power of node x which is transmitting on the same channel as node v. If u transmits at rate r_m, the receiver v can correctly decode the signal if $SINR_{uv} \geq \gamma_{r_m}$, where γ_{r_m} denotes the minimum SINR required to correctly decode a signal modulated at the rate r_m. It is a known result that the higher is the transmission rate, the higher is the SINR threshold.

We model the WMN as a directed graph $G_I = (V, E_I)$, where V is a set of nodes each representing a mesh router. Given two nodes $u, v \in V$, the directed edge $u \to v \in E_I$ if and only if, in the absence of transmissions on other links, the signal to noise ratio at v is larger than the SINR threshold for one of the available transmission rates r_m, i.e.,

$$\frac{G_{uv} P_u}{n_v} \geq \gamma_{r_m} \tag{3.2}$$

The capacity $c(u \to v)$ of the directed link $u \to v$ is the transmission rate r_m used by u to transmit to v. The capacity of the link is then set to the highest transmission rate for which the signal to noise ratio is larger than the corresponding threshold. G_I is referred to as the *potential communication graph* since an edge $u \to v \in E_I$ indicates that u can transmit to v provided that they are assigned a common channel.

A link $x \to y \in E_I$ is said to *potentially interfere* with a link $u \to v \in E_I$ if a transmission on link $x \to y$ prevents a transmission on link $u \to v$. The set of all the links that interfere with link $u \to v$, is then referred to as the potential collision domain and is denoted by

$$\mathcal{N}(u \to v) = \left\{ x \to y \in E_I \mid \frac{G_{uv} P(u \to v)}{G_{xv} P(x \to y) + n_v} < \gamma_{c(u \to v)} \right\} \tag{3.3}$$

The word potentially means that none of the links in $N(u \to v)$ may be active at the same time as link $u \to v$, if and only if they share the same channel. From equation 3.3 we see that the potential collision domain depends on the transmission rates and transmission powers selected for that link, and $x \to y \in \mathcal{N}(u \to v)$ do not mean necessarily that $u \to v \in \mathcal{N}(x \to y)$.

A channel assignment $\mathcal{A}$ assigns a set $\mathcal{A}(u)$ of channels ($|\mathcal{A}(u)| \leq k(u)$) to each node $u \in V$. Thus, $\mathcal{A}$ induces a new graph model $G = (V, E)$ where two nodes u and v are connected if $u \rightarrow v \in E_I$ *and* they share at least one common channel. In case u and v share multiple channels, the set E may include as many links between the two nodes as the number of common channels. To differentiate among those links and stress that a link has been assigned channel c, we use the notation $u \xrightarrow{c} v$. Moreover, we note that a link in E_I does not have any corresponding link in E if the end nodes do not share any channel.

A link $x \xrightarrow{c} y \in E$ interferes with $u \xrightarrow{c} v \in E$ if a simultaneous transmission on $x \xrightarrow{c} y$ prevents v from correctly decoding the signal from u, i.e., $x \rightarrow y \in \mathcal{N}(u \rightarrow v)$ and they share the same channel c. We denote the collision domain of a link $u \xrightarrow{c} v$, i.e., the set of all the links that interfere with it, by

$$\mathcal{D}(u \xrightarrow{c} v) = \left\{ x \xrightarrow{c} y \in E \mid x \rightarrow y \in \mathcal{N}(u \rightarrow v) \right\} \tag{3.4}$$

Equation 3.4 simply states that the collision domain of a link $u \xrightarrow{c} v$ is the subset of all the nodes belonging to $\mathcal{N}(u \rightarrow v)$ sharing the same channel as $u \xrightarrow{c} v$. This means that none of the links in $\mathcal{D}(u \xrightarrow{c} v)$ can be active at the same time as $u \xrightarrow{c} v$. Moreover, by definition $u \xrightarrow{c} v, v \xrightarrow{c} u \in \mathcal{D}(u \xrightarrow{c} v)$ and $x \xrightarrow{c} y \in \mathcal{D}(u \xrightarrow{c} v) \nRightarrow u \xrightarrow{c} v \in \mathcal{D}(x \xrightarrow{c} y)$. We note that $v \xrightarrow{c} u$ *must* belong to $\mathcal{D}(u \xrightarrow{c} v)$ as a single radio cannot transmit and receive simultaneously.

3.1.1 The channel assignment problem

As previously described, the first step of an approximate solution for the channel assignment and routing problem typically provides a pre-computed flow rate f for every link $u \rightarrow v \in E_I$. The channel assignment problem is to find, if any exists, a channel assignment such that the given set of pre-computed flow rates can be scheduled. It will now follow a sufficient condition for a set of pre-computed flow rates to be feasible. We note that if only one link is active in each collision domain, then there is no interference in communications. At the same time, if the scheduling period T is sufficiently long to avoid simultaneous transmission, then no interference will affect communications.

In order to ensure scheduling, then, we have to plan a scheduling period that is long enough to allow all the links in the collision domain to carry the nominal amount of traffic. For each link e, the amount of nominal traffic to be sent in a scheduling period T is equal to $f(e)T$, where $f(e)$ is the pre-computed flow rate for the link e. The time needed to transmit $f(e)T$ bit at a rate $c(e)$ is $\frac{f(e)}{c(e)}T$. If we consider link e, to avoid collision during the scheduling time T, we have to verify that $\sum_{e_0 \in \mathcal{D}(e)} \frac{f(e_0)}{c(e_0)} T \leq T$. That is,

$$\sum_{e_0 \in \mathcal{D}(e)} \frac{f(e_0)}{c(e_0)} \leq 1 \qquad \forall e \in E \tag{3.5}$$

Therefore, we established that equation 3.5 is a *sufficient* (but not necessary) condition for a set of pre-computed flow rates to be scheduled. From equation 3.5 we have that the channel assignment affects the scheduling condition, since it determines the number of nodes belonging to the collision domains in the network. More precisely, channel assignment, transmission rate and transmission power are all the three parameters that affect the composition of the collision domains and hence have direct influence on condition 3.5. For conciseness, we define

$$U_{tot}(e) = \sum_{e_0 \in \mathcal{D}(e)} \frac{f(e_0)}{c(e_0)} \tag{3.6}$$

which we refer to as the *total utilization* of the collision domain of link e. Our objective, given a wireless mesh network with a potential communication graph $G_I = (V, E_I)$, and a set of pre-computed flow rates, i.e., $f(e) \forall e \in E_I$, is to determine a channel assignment algorithm that ensure the traffic scheduling condition, i.e.:

$$\max_{e \in E} \sum_{e_0 \in \mathcal{D}(e)} \frac{f(e_0)}{c(e_0)} \leq 1 \tag{3.7}$$

In [26], a proof of the NP-completeness of the problem 3.7 is reported. This means that an optimal channel assignment algorithm that may solve the problem 3.7 (decision problem) in polinamial time does not exist. Hence, in order to approach the scheduling issue in channel assignment, heuristics are proposed.

A simple approach consists in decomposing the problem into two steps. First, a channel assignment is found, which eventually does not verify the scheduling condition 3.7. Then, the original flow rates are adjusted in order to obtain a new set of flow rates that verify the scheduling condition, given the computed channel assignment. With the current channel assignment and the set of flow rates f that cannot be scheduled, if we consider a right scaling factor $\lambda > 1$, the new set of flow rates $\frac{f}{\lambda}$ may satisfy the scheduling condition. With this approach, then, we obtain a new set of nominal flow rates, that we expect to be different from the original ones. Our objective is to find a channel assignment so that the corresponding set of flow rates is as close as possible to the original set of pre-computed flow rates, i.e., we want to minimize the scaling factor λ, where $\lambda > 1$.

Therefore, given the pre-computed set of flow rates, $f(e) \quad \forall e \in E$, we try to find a channel assignment algorithm that minimizes the scaling factor λ, so that the new set of nominal flow rates, i.e., $\frac{f(e)}{\lambda} \quad \forall e \in E$, is feasible with the new channel assignment. From equation 3.6 we can see that the total utilization of a collision domain is in a direct relationship with the pre-computed flow rates. Then, the λ scaling factor is represented by the maximum total utilization among the collision domains in the wireless network. In the attempt to keep the total utilization of the collision domains below 1, then, the channel assignment objective is to minimize the maximum total utilization of all the collision domains in the network. Such a problem turns out to be the optimization version of the decision problem stated above (equation 3.7), and hence it is NP-complete, too. In conclusion, channel assignment algorithms are heuristics that try to minimize $\max_{e \in E} U_{tot}(e)$.

3.1.2 Effect of rate and power on the total utilization

From equation 3.6 it is straightforward that the total utilization of link e is a function of the capacities of all the link in the collision domain of link e. Reducing the transmission rate on a link e leads to an increase in the total utilization of that link, and one may conclude that in order to reduce the maximum total utilization in the network, the optimal transmission rate is r_M, i.e., the maximum transmission rate available.

	$\lvert\mathcal{D}(e)\rvert$	$U_{tot}(e)$	$\lvert\mathcal{D}(e_1)\rvert$	$U_{tot}(e_1)$	$\lvert\mathcal{D}(e_2)\rvert$	$U_{tot}(e_2)$
$c(e)\downarrow$	$\downarrow ?$	$?$	$=$	$\uparrow$	$=$	$=$
$P(e)\uparrow$	$\downarrow ?$	$\downarrow ?$	$=$	$=$	$\uparrow ?$	$\uparrow ?$

Table 3.1: Summary of the impact of transmission power and rate ($e \in \mathcal{D}(e_1)$, $e \notin \mathcal{D}(e_2)$)

Decreasing the transmission rate, though, may help reducing the collision domain. In fact, decreasing the transmission rate on link e brings with it a lower SINR threshold, which means the transmission on more links may be compatible with the transmission on e (see equation 3.4). In general, $\mathcal{D}(e|c(e) = r_i) \subseteq \mathcal{D}(e|c(e) = r_j)$ for $i < j$. This means that reducing the transmission rate on link e helps reducing the size of its collision domain. As a consequence, there will be less terms to sum in the computation of the total utilization of link e. Therefore, there could be the chance to reduce the total utilization of the collision domain of link e.

At the same time, the size of the collision domain of other links does not change, since transmission rates has no effect on the composition of other collision domains. It is true, though, that reducing the transmission rate on link e also affects the total utilization of other collision domains, since the ratio $\frac{f(e)}{c(e)}$ increases.

Increasing the transmission power on a link e has the same effect on the composition of the collision domain of link e as reducing the transmission rate. Indeed, the numerator of the SINR at the receiver increases and thus some links may leave the collision domain of link e. Since an increase in the transmission power does not affect the term $\frac{f(e)}{c(e)}$, $U_{tot}(e)$ may only decrease (in case some links leave $\mathcal{D}(e)$). For the same reason, an increase in the transmission power on link e does not affect the utilization of the collision domains that already include link e. However, link e may join new collision domains, thus increasing both their cardinality and total utilization. Our findings are summarized in Table 3.1, where $\downarrow$ denotes a decrease, $\uparrow$ an increase, $?$ a possibility, and $=$ no variation.

3.2 Channel assignment algorithms: State of the art

Wireless mesh networks have been the subject of much research work recently. In particular, the potential benefits of using multiple radios per node have been explored.

In [27], multiple radios per node are used with an identical channel assignment, i.e., the first radio is assigned channel 1, the second radio is assigned channel 2 and so on. Such an approach clearly preserves connectivity but does not make any effort to reduce interference.

In [28], a hybrid channel assignment scheme is proposed, where some radios are statically assigned a channel while the remaining radios can dynamically change their frequency channel.

In Section 3.2.1 we present some works that attempt to minimize the interference. They try to minimize collision domains size, or the number of links using the same channel, without taking into consideration the routing problem. In Section 3.2.2 we present some works that take into consideration the joint channel assignment and routing problem. We divide them into centralized and distributed solutions.

Then, in Section 3.2.3 we present studies that try to cope with flows feasibility and that consider the channel assignment problem jointly with the congestion control problem. Finally, in Section 3.2.4 we present some works that analyze the channel reassignment problem, after a variation in the traffic pattern.

3.2.1 Interference aware channel assignment algorithms

Many proposals aim to assign channels so as to minimize some network-wide measure of interference and do not study the channel assignment problem in conjunction with the routing problem.

For instance, a centralized channel assignment algorithm, which is presented in [29], aims to limit interference while preserving connectivity. Given a K-connected potential communication graph, the goal is to find a channel assignment which minimizes the maximum among the size of the collision domain of all the

links subject to the constraint that the induced graph must still be K-connected.

A polynomial time recursive heuristic based on the use of a conflict graph is proposed in [30]. Each network link is associated with a link conflict weight. Two objectives are considered which lead to two different variants: GreedyMax, that aims to minimize the maximum link conflict weight at any link, and GreedyAvg, that aims to minimize the average link conflict weight over all the links.

A distributed channel assignment algorithm together with a distributed routing protocol are proposed in [31]. At any time, each node joins the neighbour which minimizes the cost to reach a gateway and sends all the packets destined to the wired network to such neighbour. Joining a new neighbour requires to update the routing tables of all the nodes along the paths to the previous and the new gateways. Each node only assigns channels to the radios used to communicate with its children nodes. Finally, channels are selected based on their usage by the interfering nodes.

In [32], the properties of an s-disjunct superimposed code are exploited to achieve an interference-free channel assignment under certain conditions. Each node is associated with a binary vector, denoted as its channel codeword, having as many components as the number of available channels. For each node, the available channels are divided into two categories: primary channels and secondary channels. A value 1 in the channel codeword denotes a primary channel and a value 0 a secondary channel.

The goal is to assign channels to a node based on its codeword and the channel codewords of its interferers. Intuitively, a node should favor a channel that is secondary to all its interferers. The authors propose a couple of algorithms assuming that the channel code of the network is an s-disjunct superimposed code.

In [33], both centralized and distributed algorithms are presented, which aim to minimize the number of pairs of links that are interfering.

3.2.2 Traffic aware channel assignment algorithms

Other proposals study the joint channel assignment and routing problem, i.e., the problem to find an assignment of channels to radios and a set of flow rates for

every network link which optimize a given objective. Common optimization objectives maximize the aggregate network throughput, to verify if a traffic demand vector is achievable or maximize the minimum end-to-end rate.

We note that such proposals actually do not consider a distributed routing protocol, the routing problem they solve being just the problem to determine a sustainable rate (or, equivalently, the available bandwidth) on every network link.

An iterative routing algorithm based on traffic profiles is proposed in [34]. Given the set of initial link flow rates, channels are assigned in the attempt to have the resulting available bandwidth on each link exceed the link flow rate. The available bandwidth values are estimated as a fraction of the link capacity and are used as input to the routing algorithm, which computes the shortest feasible path for every flow of the given traffic profile. The resulting flow allocated on each link is used as link flow rate for the next iteration, in which a new channel assignment is computed.

Centralized algorithms

A centralized channel assignment algorithm, presented in [35], takes into account the traffic generated by mesh clients. Each mesh router periodically captures packets generated by the mesh clients and measures the number of senders and per second utilization for each channel. Then, it ranks each channel based on the number of clients and the channel utilization. Each link has assigned the highest ranked channel that does not conflict with the channel assignment of its neighbours.

MesTiC [36], in turn, is a rank-based channel assignment, where the rank of a node is a function of its aggregate traffic, its number of hops from the gateway and its number of radio interfaces.

In [37], a centralized channel assignment is proposed which aims to maximize the transmission rates at which source nodes can transmit while preserving fairness among flows. The channel assignment problem is formulated as a multi-objective non-linear mixed-integer optimization problem, which is NP-complete. Hence, a randomized greedy heuristic is proposed.

In [38] the authors develop a centralized solution to the joint logical topology

design, interface assignment, channel allocation and routing problem. Given the expected traffic demands between the source-destination pairs, the authors present a linear mixed-integer program to assign channels and route the given load. For a more comprehensive survey of channel assignment algorithms, we refer the reader to [39].

In [25] an approximate solution for the joint channel assignment and routing problem is developed which optimizes the network throughput subject to fairness constraints. The traffic load that each mesh router collects from its clients and has to route towards the mesh gateways is assumed to be known. An ILP (Integer Linear Program) is formulated to find the link flow rates and the channel assignment that maximize the fraction (the same for all the nodes) of the traffic load that each node gets delivered to the wired network. Solving the LP relaxation provides a possibly unfeasible channel assignment. The channel assignment algorithm aims to fix this un-feasibility. The flow on the graph is then readjusted and scaled to ensure a feasible channel assignment and routing.

Distributed algorithms

Alternatively to static channel assignments, Makram et al. [40] present a dynamic distributed channel assignment based on clustering, with the purpose of subdividing the channel assignment problem into small local sub-problems that are easier to handle.

Dhananjay et al. [41] present a distributed protocol for channel assignment and routing in dual-radio mesh networks. The authors focus on the case where every mesh node has two radios and one has to operate on 802.11a channels while the other on 802.11b/g channels.

In [42], a distributed joint channel assignment, scheduling and routing algorithm is presented and in [26], tuning the transmission rate is exploited to present a channel and rate assignment heuristic.

3.2.3 Channel assignment and flow scheduling

The problem of how to verify the feasibility of a given set of flows between source-destination pairs is investigated in [43]. The goal is to determine the maximum scaling factor for the flows that still satisfies the constraints on the number of radios per node and the scheduling constraint. The binary variables in these constraints are approximated by appropriate continuous variables.

The resulting LP is solved by using a primal-dual approach based on shortest path routing. The solution provides a possibly unfeasible channel assignment and a set of link flow rates. A greedy channel assignment algorithm and a subsequent flow scaling are then used to ensure a feasible channel assignment and routing.

Moreover, in order to produce an interference free link schedule, a scheduling algorithm is proposed in [25].

The channel assignment problem has also been studied jointly with the congestion control problem in [44] based on [45]. TCP congestion control mechanism is analyzed in [46] as an approximate distributed algorithm solving a network utility maximization problem. This analysis is used in [45] for multi-hop wireless networks. In particular, the Shannon's formula is used to model the capacity of wireless links and the optimal source rates and transmission powers are determined in order to maximize the network utility.

In [44], instead, multi-radio nodes with fixed transmission powers are considered and the optimal source rates and channels are calculated such that the network utility will be maximized.

3.2.4 Channel re-assignment algorithms

All the papers mentioned so far, however, only deal with the problem to determine a channel assignment and do not consider the problem how to re-configure the wireless mesh network after a change in the traffic flows.

The problem of re-configuring channels in a wireless mesh network has been tackled in [47]. Such a work does not consider the standard CSMA/CA access technique but assumes the existence of a link layer synchronization among the nodes which enables them to organize their data transmissions in different time

slots with no contention. Hence, the proposed algorithm reconfigures the channel assignment and the link scheduling as a consequence of a change in the traffic matrix.

Our approach proposed in Section 3.4 is fundamentally different, as it assumes the standard contention based access technique and hence it does not perform link scheduling.

3.3 Flow-based Channel, Power and Rate Assignment Algorithm

In Section 3.1 we defined the system model and showed that the channel and routing problems are related and have to be jointly solved. We decomposed the problem into three steps, i.e., solving the routing problem determining the pre-computed flow rates, then solving the channel assignment problem trying to ensure the pre-computed flow rates feasibility, finally, adjusting the given set of flow rates in order to obtain a new set of flow rates that can be scheduled with the current channel assignment.

Moreover, we found out that if the channel assignment configuration verifies the condition 3.7, then the set of the pre-computed flow rates $f(e), \forall e \in E$ is feasible.

Such problem is NP-complete and we try to find a channel assignment that minimizes the degree of adjustment (the scaling factor λ) of the pre-computed set of flow rates.

Since this problem turns out to be the optimization version of the decision problem defined in equation 3.7, in this Section we propose a channel assignment heuristic that tries to minimize the maximum total utilization (defined in equation 3.6) among all the links in the network.

The total utilization of the collision domain of a link e has been defined as

$$U_{tot}(e) = \sum_{e_0 \in \mathcal{D}(e)} \frac{f(e_0)}{c(e_0)}$$

In Section 3.1 we outlined the relation between transmitting data rate, trans-

```
FCPRA(G_I(V, E_I), {f(e)}_{e∈E_I})
 1  A(u) ← {1}    ∀u ∈ V
 2  E ← { u -1-> v | u → v ∈ E_I }
 3  c(e) ← r_M, U_tot(e) ← Σ_{e_0∈D(e)} f(e_0)/c(e_0)    ∀e ∈ E
 4  Q ← {e}_{e∈E_I}
 5  while Q is not empty
 6       do (u → v) ← EXTRACT_MAX(Q)
 7          E ← E − { u -c_T-> v }
 8          OPTIMIZE(G_I(V, E_I), u, Q)
 9          OPTIMIZE(G_I(V, E_I), v, Q)
10          S_C ← A(u) ∩ A(v)
11          ▷ Determine the set S of candidate channels
12          for each c ∈ S
13               do U_max(c) ← TUNE_POWER_RATE(c)
14          SELECT_CHANNELS
            ▷ Update U_tot(e)    ∀e ∈ E
```

Figure 3.1: Pseudo-code FCPRA

mitting power and collision domain size, and we outlined the importance of transmission rate and power tuning for decreasing the maximum total utilization among the collision domains in the network, hence improving the channel assignment performance.

The Total Utilization of a link e is the sum of the ratios $\frac{f(e_0)}{c_0}$ for all the links e_0 that are in the collision domain of e. In Section 3.1 we defined the collision domain of a link e with equation 3.4.

The collision domain is related to the γ_{r_m} threshold, wich is the minimum SINR required to correctly decode the signal transmitted at the rate r_m. Moreover, we recall that the higher the transmission rate, the higher the γ threshold.

The idea behind the proposed channel assignment algorithm is to exploit the impact of the transmission power and rate on the total utilization of the collision domains.

In this Section we present the FCPRA (Flow-based Channel, Power and Rate Assignment) algorithm, a greedy heuristic for the channel assignment problem

defined in Section 3.1 that exploits the ability to tune transmission power and rate to minimize the maximum total utilization among all the collision domains.

The FCPRA algorithm, whose pseudo-code is shown in Figure 3.1, inserts all the links of the potential communication graph into a priority queue Q and extracts all of them one-by-one to originate one or more links to be included in E (hence connectivity is preserved). For each extracted link $u \rightarrow v$, the set of candidate channels is computed first. Then, for each such channels c, the transmission power and rate values are determined in order to minimize the maximum total utilization in case a link $u \xrightarrow{c} v$ is added.

Based on the flow rate associated with the link and the maximum total utilization resulting from adding a link on each candidate channel, the set of links $u \xrightarrow{c} v$ to be added to E is determined.

We analyze below the steps taken by the algorithm.

Initialization (lines 1–4)

A channel can be assigned to a link only if one radio on both the end nodes can be set to that channel. In order to ensure that all the extracted links share a common channel, we initialize one radio of every node to a fixed channel, say it channel 1. Thus, a temporary link on channel 1 exists between every two nodes connected in the potential communication graph and is initially included in E.

Each such link is also assigned the highest transmission rate and the power resulting from the heuristics in [48].

Extracting a link and determining the candidate channels

Links are extracted from Q in decreasing order of priority, where the priority of a link $u \rightarrow v$ is the total utilization of the collision domain of the corresponding temporary link $u \xrightarrow{c_T} v$ (c_T is channel 1 unless modified by the optimization step).

When a link $u \rightarrow v$ is extracted, the goal is to determine the set of links $u \xrightarrow{c} v$ that will replace the temporary link $u \xrightarrow{c_T} v$ in E. Links can be clearly established on channels in common to both end nodes. We denote by $\mathcal{S}_C$ the set of channels shared by the end nodes. Additionally, we allow at most one radio on each end

node to be assigned a new channel. Thus, the set $\mathcal{S}$ of candidate channels for the establishment of links $u \xrightarrow{c} v$ includes $\mathcal{S}_C$ and is determined as follows.

If both the end nodes of the extracted link have available radios (i.e., radios with no channel assigned), then all the available channels become candidates. If only one of the end nodes of the extracted link has available radios, then we may select a channel from among those assigned to the radios of the other end node. If both end nodes have no available radios, we can only select channels in common to the end nodes. The initialization of one radio of every node to channel 1 ensures that every pair of end nodes share at least one common channel, hence the set $\mathcal{S}$ is always non-empty.

The *optimization* step (lines 8–9), which is described hereinafter, preserves this condition despite it can remove the initial assignment of channel 1 to the radios of the end nodes.

Determining power and rate for each candidate channel (Figure 3.2)

Since each link is initially assigned the minimum transmission power (that ensures a connected bidirectional potential communication graph) and the maximum transmission rate, we observe that increasing the power and decreasing the rate on the extracted link cannot decrease the total utilization of the collision domains of the other links (see Table 3.1). Thus, tuning power and rate may only help in case the total utilization of the collision domain of the extracted link exceeds those of the other collision domains.

For each channel $c \in \mathcal{S}$ we consider all the links $x \xrightarrow{c} y$ which would have $u \xrightarrow{c} v$ in their collision domain and compute the total utilization of their collision domain. We note that at this moment the link $u \xrightarrow{c} v$ does not exist in E and thus, in case it were established, all such total utilizations would be increased of the same amount, i.e., $\frac{f(u \xrightarrow{c} v)}{c(u \xrightarrow{c} v)}$.

To the purpose of determining the power and the rate which minimize the maximum total utilization, we only consider the maximum among such total utilizations, which we denote by $U'_{max}(c)$ (line 2).

If a link $u \xrightarrow{c} v$ were established, we would also need to consider the total

TUNE_POWER_RATE(c)

1 $\mathcal{D}_1 \leftarrow \left\{x \xrightarrow{c} y | u \xrightarrow{c} v \in \mathcal{D}(x \xrightarrow{c} y)\right\}$
2 $U'_{max}(c) \leftarrow \max_{x \xrightarrow{c} y \in \mathcal{D}_1} U_{tot}(x \xrightarrow{c} y)$, $U''_{max}(c) \leftarrow 0$
3 $m \leftarrow M$, $c(u \xrightarrow{c} v) \leftarrow r_M$, $w \leftarrow i | P_i = P(u \rightarrow v)$
4 $w_{min} \leftarrow w$, $min \leftarrow U_{tot}(u \xrightarrow{c} v)$
5 **while** $w < W$ and $U_{tot}(u \xrightarrow{c} v) > U'_{max}(c)$ and $U''_{max}(c) < U'_{max}(c)$
6 **do** $w \leftarrow w + 1$, $P(u \xrightarrow{c} v) \leftarrow P_w$
7 $U''_{max} \leftarrow \max_{x \xrightarrow{c} y \notin \mathcal{D}_1 \,|\, u \xrightarrow{c} v \in \mathcal{D}(x \xrightarrow{c} y)} U_{tot}(x \xrightarrow{c} y)$
8 **if** $U_{tot}(u \xrightarrow{c} v) < min$ and $U''_{max}(c) < U'_{max}(c)$
9 **then** $min \leftarrow U_{tot}(u \xrightarrow{c} v)$
10 $w_{min} \leftarrow w$
11 $m_{min} \leftarrow M$, $min \leftarrow U_{tot}(u \xrightarrow{c} v)$
12 **while** $m > 1$ and $U_{tot}(u \xrightarrow{c} v) > U'_{max}(c)$
13 **do** $m \leftarrow m - 1$, $c(u \xrightarrow{c} v) \leftarrow r_m$
14 **if** $U_{tot}(u \xrightarrow{c} v) < min$
15 **then** $min \leftarrow U_{tot}(u \xrightarrow{c} v)$
16 $m_{min} \leftarrow m$
17 $c(u \xrightarrow{c} v) \leftarrow r_{m_{min}}$
18 $P(u \xrightarrow{c} v) \leftarrow P_{w_{min}}$
19 $U_{max}(c) \leftarrow \max\left(U'_{max}(c), U_{tot}(u \xrightarrow{c} v)\right)$

Figure 3.2: Pseudo-code TUNE_POWER_RATE

utilization of its collision domain. If $U_{tot}(u \xrightarrow{c} v)$ (computed without the term $\frac{f(u \xrightarrow{c} v)}{c(u \xrightarrow{c} v)}$) is greater than $U'_{max}(c)$, then we try to iteratively increase the transmission power on $u \xrightarrow{c} v$ in the attempt to reduce the size of the collision domain of $u \xrightarrow{c} v$ and hence its total utilization (see Table 3.1). As we discussed, increasing the transmission power does not affect $U'_{max}(c)$ but may lead $u \xrightarrow{c} v$ to interfere with other links. In such a case, we need to evaluate the total utilization of the collision domains of such links (7) and stop increasing the transmission power if the maximum among such total utilizations exceeds $U'_{max}(c)$.

If $U_{tot}(u \xrightarrow{c} v)$ still exceeds $U'_{max}(c)$, we can try to iteratively decrease the transmission rate on $u \xrightarrow{c} v$ in the attempt to reduce the size of its collision do-

SELECT_CHANNELS

1 $\{c_i\}_{i=0}^{|\mathcal{S_C}|} \leftarrow \mathcal{S_C} \cup \underset{c \in \mathcal{S}-\mathcal{S_C}}{\operatorname{argmin}} U_{max}(c)$
  ▷ Sort $\{c_i\}$ in increasing order of $U_{max}(c_i)$
2 $F \leftarrow f(u \rightarrow v)$, $i \leftarrow 0$
3 **while** $F > 0$
4 **do** $E \leftarrow E \cup \{u \xrightarrow{c_i} v\}$
5 $\mathcal{A}(u) \leftarrow \mathcal{A}(u) \cup \{c_i\}$, $\mathcal{A}(v) \leftarrow \mathcal{A}(v) \cup \{c_i\}$
6 **if** $i < |\mathcal{S_C}|$ and
  $F \geqslant [U_{max}(c_{i+1}) - U_{max}(c_i)] \cdot$
  $\sum_{k=0}^{i} c(u \xrightarrow{c_k} v)$
7 **then** $I \leftarrow U_{max}(c_{i+1}) - U_{max}(c_i)$
8 **else** $I \leftarrow \frac{F}{\sum_{k=0}^{i} c(u \xrightarrow{c_k} v)}$
9 **for** $k \leftarrow 0$ to i
10 **do** $f(u \xrightarrow{c_k} v)$ += $I \cdot c(u \xrightarrow{c_k} v)$
11 F -= $I \cdot c(u \xrightarrow{c_k} v)$
12 $i \leftarrow i + 1$

Figure 3.3: Pseudo-code SELECT_CHANNELS

main. Unlike increasing the transmission power, decreasing the transmission rate does not make $u \xrightarrow{c} v$ possibly join new collision domains but causes an increase in the term $\frac{f(u \xrightarrow{c} v)}{c(u \xrightarrow{c} v)}$ which is added both to $U_{tot}(u \xrightarrow{c} v)$ and $U'_{max}(c)$.

Therefore, we keep on trying lower transmission rates as long as $U_{tot}(u \xrightarrow{c} v)$ remains greater than $U'_{max}(c)$. Also, the transmission rate is actually decreased if it allows to reduce the total utilization. Then, $U_{max}(c)$ is set to the maximum between $U'_{max}(c)$ and $U_{tot}(u \xrightarrow{c} v)$.

Selecting channels (Figure 3.3)

The pre-computed flow rate $f(u \rightarrow v)$ can then be split over links established on the candidate channels. Since we allow at most one radio on each end node to be assigned a new channel, we can only consider one channel that is not already shared by the end nodes. We select the channel that minimizes $U_{max}(c)$ for $c \in \mathcal{S} - \mathcal{S_C}$. Such a channel and the channels in S_C constitute the actual set $\{c_i\}$ of the candidate channels to carry the flow rate $f(u \rightarrow v)$. In order to minimize the

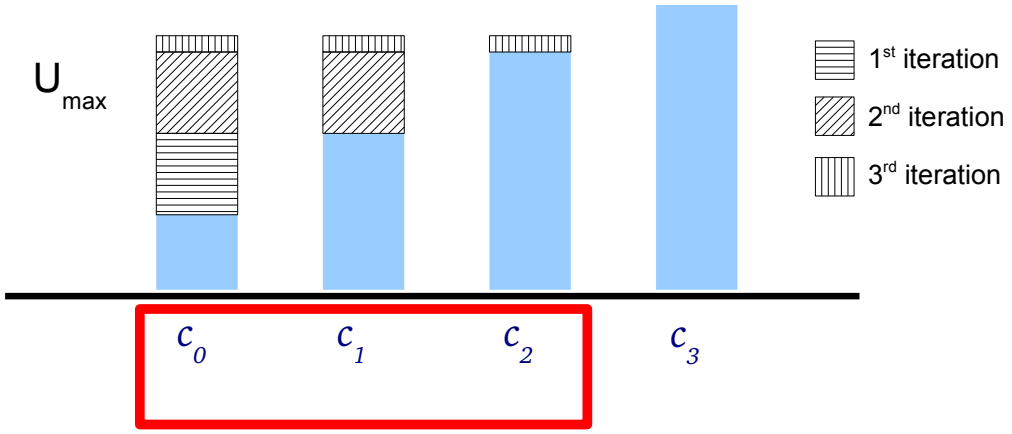


Figure 3.4: Selecting channels

maximum total utilization, the flow rate $f(u \rightarrow v)$ is split in the following way.

We begin allocating the flow rate on the candidate channel associated with the smallest U_{max} value (c_0, if we assume the set $\{c_i\}$ ordered for increasing values of U_{max}) until either $U_{tot}(u \xrightarrow{c_0} v)$ equals $U_{max}(c_1)$ or all the flow rate has been allocated. In the latter case, only the link $u \xrightarrow{c_0} v$ is established between u and v. In the former case, we keep on allocating the flow rate equally on c_0 and c_1 until either $U_{tot}(u \xrightarrow{c_0} v)$ and $U_{tot}(u \xrightarrow{c_1} v)$ equal $U_{max}(c_2)$ or all the flow rate has been allocated, and so on.

At the end of these steps (lines 2–12), the links $u \xrightarrow{c_i} v$ on which part of the pre-computed flow rate $f(u \rightarrow v)$ has been allocated are included in the set E of links of the graph G.

Figure 3.4 illustrates the case where four channels are candidate and the available flow is only split among the first three channels. In the first iteration, we start allocating the flow rate only on channel c_0, until the total utilization $U_{tot}(u \xrightarrow{c_0} v)$ equals $U_{max}(c_1)$. Then, in the second iteration we keep allocating the flow rate on channel c_0, and we start allocating the same amount of flow rate also on channel c_1. The second iteration ends when either $U_{tot}(u \xrightarrow{c_0} v)$ and $U_{tot}(u \xrightarrow{c_1} v)$ equal $U_{max}(c_2)$. Therefore, in the third iteration step, we consider the three channels c_0, c_1 and c_2 and start allocating the remaining amount of flow rate. In this example it was possible to allocate all the traffic flow on only three channels.

```
OPTIMIZE(G_I(V, E_I), u, Q)
 1  if 1 ∉ A(u) or |A(u)| < k(u)
 2     then return
 3  L ← {x → y ∈ E_I | x = u ∨ y = u}
 4  for each x → y ∈ L − Q
 5        do if x -1-> y ∈ E
 6              then return
 7  for each x → y ∈ L ∩ Q
 8        do if A(x) ∩ A(y) − {1} = ∅
 9              then return
10  for each x → y ∈ L ∩ Q
11        do select c ∈ A(x) ∩ A(y) − {1}
12           E ← E ∪ {x -c-> y} − {x -1-> y}
13  A(u) ← A(u) − {1}
    ▷ Update U_tot(e)   ∀e ∈ E
```

Figure 3.5: Pseudo-code OPTIMIZE

Optimization step

In the initialization phase of the FCPRA algorithm one radio of every node is assigned channel 1. Given the limited availability of radios, this choice may be too wasteful. The optimization step is introduced to remove the initial assignment of channel 1 while still ensuring that every pair of connected nodes share at least a common channel.

The OPTIMIZE function, shown in Figure 3.5, takes as input one end node of the extracted link, e.g., u, and the priority queue, and checks whether it is possible to remove channel 1 from u's radios. If none of u's radios is assigned channel 1 (node u might have already been passed to OPTIMIZE as end node of a previously extracted link) the function returns. OPTIMIZE also returns in case u has radios with no channel assigned, since removing channel 1 from a radio would not enlarge the set of candidate channels.

To remove channel 1 from u's radios while preserving the property that every pair of connected nodes in the network share a common channel, we need to analyze the set $\mathcal{L}$ of the links leaving or entering node u. We distinguish two

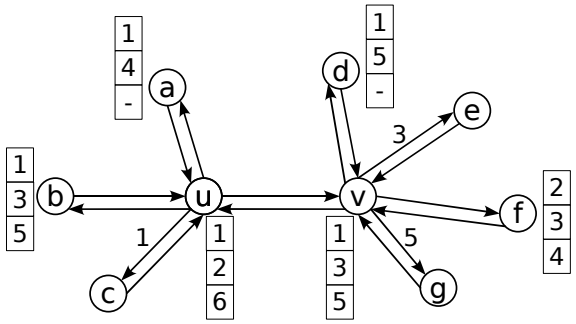


Figure 3.6: Example of the optimization step

cases:

1. **The link has already been extracted from Q:** such a link should not have been assigned channel 1, otherwise it is not possible to remove channel 1 from u's radios.

2. **The link has not been extracted from Q yet:** the end nodes must share at least one channel other than channel 1, otherwise it is not possible to remove channel 1 from u's radios.

The OPTIMIZE function returns as soon as a link is found which does not obey the above conditions. If no such link is found, channel 1 can be removed from among u's radios. In such case, we need to assign a new temporary channel to all the links leaving or entering u that are still in the queue Q.

In Figure 3.6 we show a example of the optimization step. We suppose that link $u \rightarrow v$ has been extracted. Numbers reported next to a node represent the channels assigned to its radios, while the number close to a link represents the channel it has been assigned, if that is the case. In this example, channel 1 can be removed from v's radios but not from u's radios.

Notes on increasing the transmission power

In the attempt to reduce the size of the collision domain of the extracted link, we increase its transmission power. Consequently, it might happen that the power level is sufficient to reach new neighbours, meaning that new links could be added

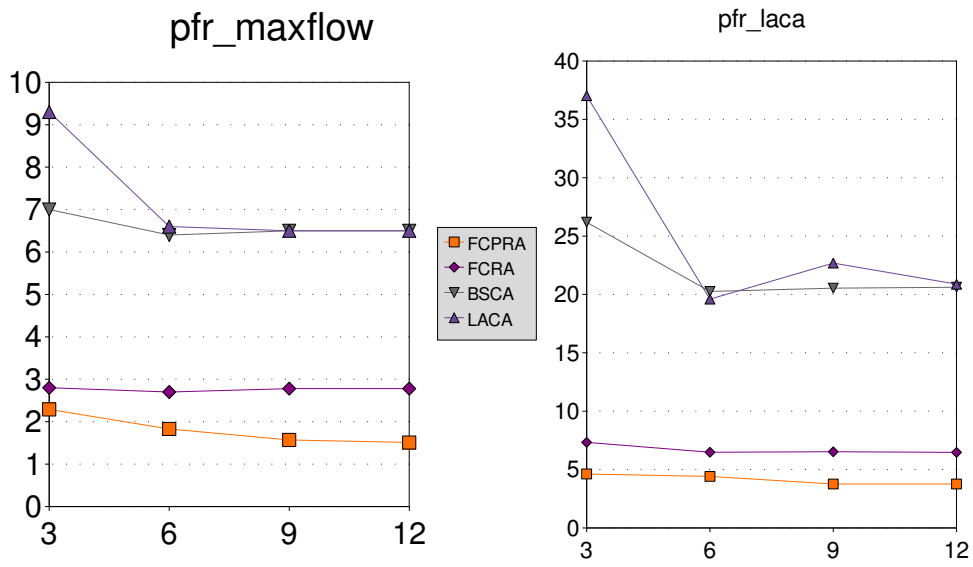


Figure 3.7: 25-nodes topologies

to the communication graph. In this circumstance we do not exploit this possibility, due to the complexity it adds to the algorithm.

Indeed, not only we need to evaluate the impact of new links on the total utilization of other collision domains, but we also need to associate the new links with proper flow rates, which relates to the method used to compute the flow rates.

3.3.1 Performance evaluation

The aim of this Section is to evaluate the effectiveness of different channel assignment algorithms by comparing the resulting minimum scaling factor $\lambda = \max_{e \in E} U_{tot}(e)$.

We compare our FCPRA algorithm to FCRA [26], LACA (Load-Aware Channel Assignment) [34] and BSCA (Balanced Static Channel Assignment) [43]. We consider two classes of topologies with 25 and 50 nodes, respectively, and a different distribution of radios per node. For the first (second) class, we generate 20 connected topologies by randomly placing nodes in a 300×300 (400×400) square meter area.

The thresholds γ_{r_m} are determined by using the *BER-SINR* curves of the IEEE

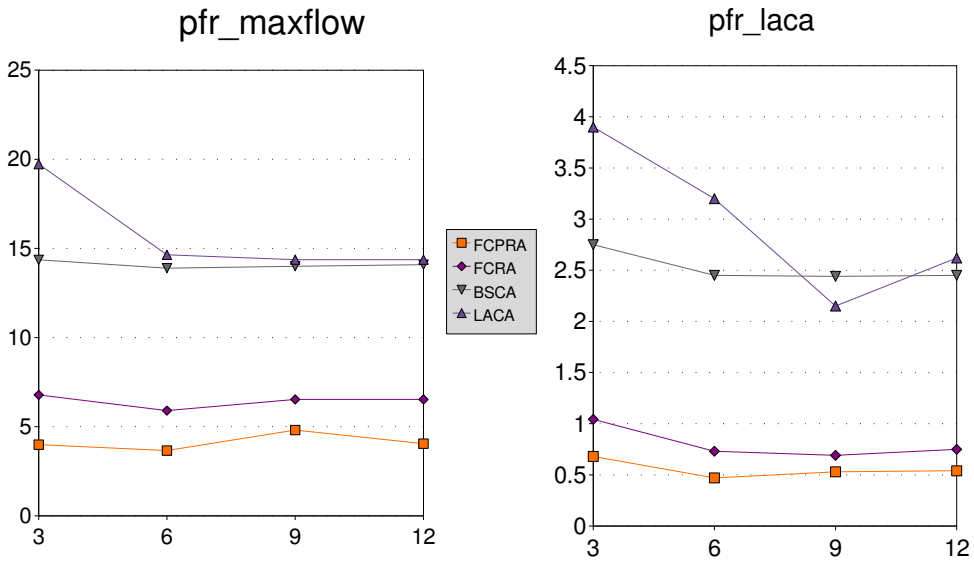


Figure 3.8: 50-nodes topologies

802.11a/g modulation schemes [49] with target bit-error-rate (BER) of 10^{-6} and packet size of 1024 bytes.

For BSCA and LACA, the link capacities are set according to the values commonly advertised by 802.11a vendors: $54Mb/s$ when the end nodes are within $30m$, $48Mb/s$ when within $32m$, $36Mb/s$ when within $37m$, $24Mb/s$ when within $45m$, $18Mb/s$ when within $60m$, $12Mb/s$ when within $69m$, $9Mb/s$ when within $77m$, and $6Mb/s$ when within $90m$.

We consider two scenarios differing in the way the set of pre-computed flow rates are computed. We denote by *pfr_maxflow* the method proposed in [26], and by *pfr_laca* the method proposed in [34] to compute the set of initial flow rates. We also consider different numbers of available channels: 3, 6, 9, and 12. The results of the simulations are shown in Figures 3.7 and 3.8 for the 25 and 50 nodes topologies, respectively.

Each Figure illustrates both the cases where *pfr_maxflow* and *pfr_laca* are used. For each value of the total number of available channels, the Figures show the average values of $\lambda = \max_{e \in E} U_{tot}(e)$ over all the 20 topologies of each class for every channel assignment algorithm. We can observe that the best performance

is achieved by FCPRA, as in every scenario it exhibits the smallest scaling factor required to obtain a set of feasible flow rates.

Therefore, these results confirm the effectiveness of tuning both the transmission power and rate.

3.4 Minimum Variation Channel and Rate Re-Assignment Algorithm

Differently from the approach followed in Section 3.3, we introduce below a channel re-assignment algorithm.

Since the beginning of this Chapter, we recognized channel assignment and computation of optimal traffic flows to be interconnected problems that need to be solved jointly. In particular, we decomposed the joint problem in three steps: first, we found the solution to the routing problem, consequently, we assigned channels trying to obtain the right available bandwidth to ensure scheduling, and, as a third step we eventually scaled the original flow rates in order to satisfy the scheduling condition.

This chained relationship between pre-computed flow rates and the channels assigned to the radio interfaces means that a variation of the pre-computed flow rates, i.e., a variation in the traffic profile to be routed, leads to a need for a new channel recomputation. A traffic pattern variation, then, raises the need for a new channel assignment.

Channel assignment algorithms, like the one presented in Section 3.3, are designed to assign channels from scratches. They do not take into consideration a possible current channel assignment and, with a different set of pre-computed flow rates, they are likely to return a completely different configuration of radios.

Enforcing such a new channel configuration may not be desirable, since it represents a serious threat to the stability of network operations and to the performance perceived by time sensitive applications. In fact, frequent channel assignments will require the reconfiguration of an extended number of radios, where the time for a single radio channel switching is in the range of hundreds of millisec-

onds. The time needed for the radio reconfiguration may lead to lost of connectivity and timeouts.

Moreover, the channel assignment problem is NP-complete, hence the proposed channel assignment algorithms are heuristics that do not provide the optimal solution. Finally, heuristics may have a non negligible computational complexity, given the complexity of the channel assignment problem, and frequent channel assignments recomputations may not be feasible.

For these reasons, we introduced a simple channel assignment heuristic that takes the current channel assignment into account and aims to adjust the minimum possible number of channels in order to cope with the variation of the set of pre-computed traffic flows in the best possible manner.

We constrained the maximum number of radios that can be reconfigured: this will likely prevent the algorithm form obtaining the optimal solution that best adapts to the new flow rates.

In addition, the proposed channel reassignment algorithm takes advantage of the transmission rates made available by the IEEE standards, 802.11b/g and 802.11a. In fact, by properly tuning transmission rates on each link, the proposed algorithm can reduce the perceived interference that is induced by the neighbour links sharing the same channel. The algorithm, then, finds out the most suitable transmission rate for each network link.

The channel reassignment algorithm makes use of the complete network topology and the set of pre-computed flow rates. It is naturally suited to a centralized implementation and can be run, for example, by a network management station. On condition that mesh routers are capable of exchanging the required information, a distributed implementation is also possible and is under study. If the flow rate information and the channel rates and frequencies associated to each link are spread among all the mesh routers, all the necessary information to run the algorithm are provided. A possible solution to exchange such information among nodes may be represented by modifying the link state routing protocol, for example extending the Topology Control message used by OLSR (Optimized Link State Routing) [2].

Our proposed approach thus assumes the availability of a solution to the channel assignment problem given an original set of pre-computed flow rates. Then, we assume a new set of pre-computed flow rates is provided. Our proposed algorithm takes as input the current channel assignment (computed by a channel assignment algorithm for the original set of pre-computed flow rates) and the new set of pre-computed flow rates.

Finally, the channel reassignment algorithm here presented and the $FCPRA$ described in Section 3.3 can easily coexist, since they address the same problem (the channel assignment), but with different scopes and time scales. In a typical network scenario, we may have already assigned the available channels with the $FCPRA$ algorithm, but we may deem too expensive to mess up the actual network configuration with a new $FCPRA$ rerun only to meet the updated flow rates. In the near future, then, we may decide to face the new flow rates with subsequent channel reassignment enforcements, and delay the $FCPRA$ algorithm for a late possible rerun.

The goal of the proposed reassignment algorithm is to minimize the total utilization over all the collision domains, subject to the constraint that the number of required changes to the channels assigned to radios must not exceed a given value.

In the attempt to minimize the total utilization, we propose to adjust the transmission rate on the network links. Thus, our proposed channel re-assignment algorithm, presented in this Section, adjusts the channel and the transmission rate on each link, while considering the impact on the total utilization of the other collision domains.

We illustrate the operation of our algorithm through the pseudo-code shown in Figures 3.9 to 3.13.

The Minimum Variation Channel and Rate Re-Assignment algorithm (MVCRA-R) takes as input the current assignment of channels (i.e., the communication graph G and the set of channels $\mathcal{A}(v)$ assigned to each node v), the new set $f(e)$ of pre-computed flow rates and the $MaxNumChanges$ parameter, which determines the maximum allowed number of changes to the channels assigned to the radios. In order to determine what radios need to be assigned a new channel, we

MVCRA-R$(G(V,E), \{\mathcal{A}(v)\}_{v\in V}, \{f(e)\}_{e\in E}, MaxNumChanges)$

$U_{tot}(e) \leftarrow \sum_{e_0\in\mathcal{D}(e)} \frac{f(e_0)}{c(e_0)} \quad \forall e \in E$
$Q \leftarrow \{e\}_{e\in E}$
$Num\_Changes \leftarrow 0$
while $Q \neq \emptyset$ AND $(Num\_Changes < MaxNumChanges)$
  do $(u \overset{c_{old}}{\rightarrow} v) \leftarrow$ EXTRACT_MAX(Q)
    $\left(c_{selected},\ c(u \overset{c_{selected}}{\rightarrow} v)\right) \leftarrow$ MIN_UTOT$(u, v, \mathcal{C})$
    CHANGE_IF$(u, c_{selected})$
    CHANGE_IF$(v, c_{selected})$
    $E \leftarrow E - \{u \overset{c_{old}}{\rightarrow} v\} \cup \{u \overset{c_{selected}}{\rightarrow} v\}$
    while Q_P is not empty
      do $(s \rightarrow t) \leftarrow$ EXTRACT_MAX(Q_P)
        if $\mathcal{A}(s) \cap \mathcal{A}(t) \neq \emptyset$
          then $\mathcal{S} \leftarrow \mathcal{A}(s) \cap \mathcal{A}(t)$
            $\left(c_{selected},\ c(s \overset{c_{selected}}{\rightarrow} t)\right) \leftarrow$ MIN_UTOT$(s, t, \mathcal{S})$
            $E \leftarrow E - \{s \overset{c_{old}}{\rightarrow} t\} \cup \{s \overset{c_{selected}}{\rightarrow} t\}$
          else
            $\mathcal{S} \leftarrow \mathcal{A}(s)$
            $(k_c, n_c) \leftarrow$ MIN_DISRUPT$(t, c) \quad c \in \mathcal{S}$
            $c_{selected} \leftarrow \underset{c\in\mathcal{S}}{\operatorname{argmin}}\, n_c$
            $c(s \overset{c_{selected}}{\rightarrow} t) \leftarrow$ ADJUST_RATE$(s \overset{c_{selected}}{\rightarrow} t, 0)$
            CHANGE_IF$(t, c_{selected})$
            $E \leftarrow E - \{s \overset{c_{old}}{\rightarrow} t\} \cup \{s \overset{c_{selected}}{\rightarrow} t\}$

Figure 3.9: Pseudo-code MVCRA-R

first compute the total utilization of all the collision domains as determined by the current channel assignment and the new set of pre-computed flow rates (line 1).

All the links of the communication graph are then inserted into a priority queue Q and are extracted one by one (line 4) in decreasing order of priority, where the priority of a link $u \rightarrow v$ is the total utilization of its collision domain. The $Num\_Changes$ variable holds the current number of channel adjustments and should not exceed the $MaxNumChanges$ parameter.

When a link $u \rightarrow v$ is extracted (we denote by c_{old} the channel it is currently assigned), the goal is to determine a new channel c (independently from the channels currently assigned to u and v), and a new rate r, that minimize the total utilization of its collision domain (line 5). This is achieved by invoking the MIN_UTOT function (figure 3.10), which analyzes the effects of assigning each of the potential channels to link $u \rightarrow v$ and returns the most convenient one. In particular, in order not to take decisions that might considerably aggravate the total utilization of other collision domains, the MIN_UTOT function also considers, for each channel c in the set $\mathcal{S}$, the total utilization of the collision domain of all the links $x \xrightarrow{c} y$ which would have $u \xrightarrow{c} v$ in their collision domain. In case a link $u \xrightarrow{c} v$ were established, all such total utilizations would be increased of the same amount, i.e., $\frac{f(u \xrightarrow{c} v)}{c(u \xrightarrow{c} v)}$.

In order to keep track of the effects of assigning a channel c to the extracted link on such collision domains, it suffices to only consider the maximum among such total utilizations, which is denoted by $U'_{max}(c)$ (line 2 in figure 3.10). If a link $u \xrightarrow{c} v$ were established, we would also need to consider the total utilization of its collision domain. $U_{tot}(u \xrightarrow{c} v)$ may be decreased by reducing the transmission rate on $u \xrightarrow{c} v$, because a lower rate may allow to reduce the size of the collision domain.

For the purpose of determining the most suitable rate, the ADJUST_RATE function (figure 3.11) is invoked. Such a function starts by considering the highest rate possible and then proceeds by iteratively trying lower rates, as long as $U_{tot}(u \xrightarrow{c} v)$ is greater than $U'_{max}(c)$. We note that the rate is actually decreased only if it allows to reduce the total utilization $U_{tot}(u \xrightarrow{c} v)$ (line 5 in Figure 3.11). Then,

```
MIN_UTOT(u, v, S)
1  for each c ∈ S
2      do U'_max(c) ← max_{x -c-> y | u -c-> v ∈ D(x -c-> y)} U_tot(x -c-> y)
3         r_c ← ADJUST_RATE(u -c-> v, U'_max(c))
4         U_max(c) ← max(U'_max(c), U_tot(u -c-> v))
5  return ( argmin_{c∈S} U_max(c), r_c )
```

Figure 3.10: Pseudo-code MIN_UTOT

```
ADJUST_RATE(e, U_max)
1  m ← max{i ∈ 1...M | e ∈ E_I ∧ c(e) = r_i}
2  m_min ← m, min ← U_tot(e)
3  while m > 1 AND U_tot(e) > U_max
4      do m ← m − 1, c(e) ← r_m
5         if U_tot(e) < min
6           then min ← U_tot(e)
7                m_min ← m
8  return r_{m_min}
```

Figure 3.11: Pseudo-code ADJUST_RATE

the MIN_UTOT function computes the collision domain of $u \xrightarrow{c} v$ considering the rate returned by ADJUST_RATE and uses the $U_{max}(c)$ variable to hold the maximum between $U_{tot}(u \xrightarrow{c} v)$ and $U'_{max}(c)$. MIN_UTOT returns the channel that minimizes $U_{max}(c)$ and the rate selected for that channel.

Since the channel $c_{selected}$ returned by MIN_UTOT may not be currently assigned to any radio on u and v, it is necessary to take appropriate actions. In particular, the CHANGE_IF function is invoked (lines 7 and 8 in Figure 3.12) to set a radio interface of nodes u and v to $c_{selected}$.

The CHANGE_IF function (figure 3.12) gets as input the node u and the channel c that has to be assigned to one of u's radios. If channel c is already assigned to one of u's radios, then nothing else needs to be done (lines 1-2). Otherwise, we attempt to assign channel c to the radio of u that causes the least disruption in the network configuration.

CHANGE_IF(u, c)
1 **if** $(c \in \mathcal{A}(u))$
2 **then** return
3 $(k, n) \leftarrow$ MIN_DISRUPT(u, c)
4 $\triangleright$ Add links to the queue of pending links
5 $Q_P \leftarrow Q_P \cup u \xrightarrow{k} x \in E \mid c \notin \mathcal{A}(x)$
6 $\mathcal{A}(u) \leftarrow \mathcal{A}(u) - \{k\} \cup \{c\}$
7 $\triangleright$ Pending links are removed from Q
8 $Q \leftarrow Q - Q_P$
9 $Num\_Changes\,{+}{+}$

Figure 3.12: Pseudo-code CHANGE_IF

MIN_DISRUPT(u, c)
1 $W_k \leftarrow \left\{ u \xrightarrow{k} w \in E \mid c \notin \mathcal{A}(w) \right\} \quad k \in \mathcal{A}(u)$
2 $n \leftarrow \min_{k \in \mathcal{A}(u)} |W_k|$
3 $K \leftarrow \{k \in \mathcal{A}(u) \mid |W_k| = n\}$
4 **if** $|K| = 1$
5 **then return** $(k \in K, n)$
6 $n \leftarrow \min_{k \in K} |W_k \cap (E - Q)|$
7 **return** $\left(\operatorname{argmin}_{k \in K} |W_k \cap (E - Q)|, n \right)$

Figure 3.13: Pseudo-code MIN_DISRUPT

For this purpose, the MIN_DISRUPT function is invoked (figure. 3.13), which computes, for every channel k currently assigned to node u, the set W_k of links that would be disrupted by switching a radio on u from channel k to c. Clearly, all links originating from u and ending on a node which already has channel c assigned to one of its radios would not be disrupted. MIN_DISRUPT returns the channel k that would disrupt the minimum number of links (and such minimum number). The rationale of MIN_DISRUPT is thus to minimize the number of links to be repaired as a consequence of a channel switching and hence to limit the so called ripple effect [34].

The operations of MIN_DISRUPT are illustrated with reference to figure 3.14,

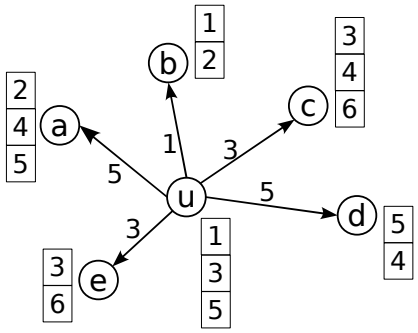


Figure 3.14: Example to illustrate the operations of MIN_DISRUPT

where channel 2 has to be assigned to one of the radios on u. In such example, $\mathcal{A}(u) = \{1, 3, 5\}$ and $W_1 = \emptyset$ (because $2 \in \mathcal{A}(b)$), $W_3 = \{u \xrightarrow{3} c, u \xrightarrow{3} e\}$ and $W_5 = \{u \xrightarrow{5} d\}$ (because $2 \in \mathcal{A}(a)$). Hence, channel 1 would be returned because switching a radio on u from 1 to 2 does not break any link. In case multiple channels k would cause the disruption of the minimum number of links, another criterion is adopted to break the tie. In particular, we select the channel that minimizes the number of links that would be disrupted among those that have not been extracted yet from the queue Q. The reason is that extracted links have been already processed and breaking their channel would require them to be re-processed again.

MIN_DISRUPT returns the channel k that can be replaced by c on node u. Consequently, all the links on u that were using channel k must be assigned a new channel. In order to do so, CHANGE_IF inserts all such links into the queue Q_P of the *pending* links, i.e., links that need to be assigned a new channel (line 5 in figure 3.12). Then, the pending links are removed from the queue Q, since they will be processed when extracted from the queue Q_P. Finally, $Num\_Changes$ is increased to reflect the channel adjustment on u and the CHANGE_IF function returns.

As mentioned above, a call to CHANGE_IF from MVCRA-R (lines 7 and 8 in Figure 3.12) may bring some links into a pending state, where they need to be assigned a new channel. MVCRA-R will thus extract the links from Q_P one by one, until the queue is empty (line 11). For each extracted link $s \rightarrow t$, if we are

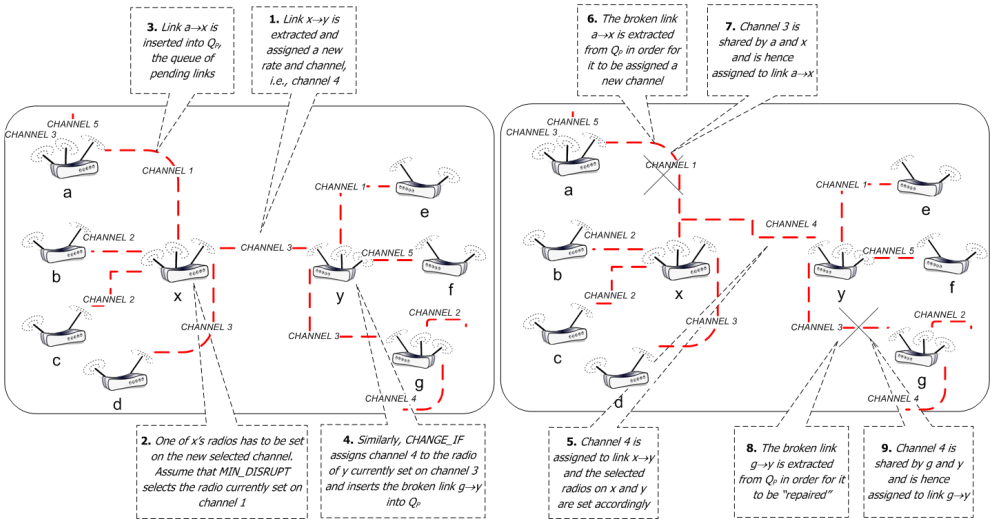


Figure 3.15: Using example for clarification of the MVCRA-R algorithm

lucky enough that s and t share a set of common channels, then (lines 12 to 15) the channel from such a set and the rate minimizing the maximum total utilization, as determined by the MIN_UTOT function, are selected. Otherwise (lines 17 to 22), channels assigned to one end point (e.g., s) are considered and one of them is assigned to the other end point (e.g., t).

To determine the most suitable channel, the MIN_DISRUPT function is invoked to determine how many links would be disrupted by assigning each of such candidate channels to t. The channel minimizing the number of such links is selected. Then, ADJUST_RATE is invoked in the attempt to minimize the total utilization of the link extracted from Q_P. Finally, CHANGE_IF is invoked to actually assign the selected channel to one of the radios on t.

MVCRA-R simple application case

To further clarify the operations of MVCRA-R, Figure 3.15 illustrates the steps of the algorithm with reference to a sample topology. Herein after, we refer to Figure 3.15 and we explain the algorithm step by step.

(Step 1.) First, we assume that link $x \xrightarrow{3} y$ has the highest priority and is extracted from the queue Q. The MIN_UTOT and ADJUST_RATE procedures

are invoked to determine the new rate and the new channel for the extracted link. We assume that channel 4 is determined as the channel minimizing $U_{tot}(x \xrightarrow{3} y)$. Hence, channel 4 has to be assigned to one radio on x and y. Then we have to properly tune the network interface cards.

(Step 2.) The CHANGE_IF procedure is invoked to select which radio on x has to switch to channel 4. We suppose that MIN_DISRUPT returns channel 1.

(Step 3.) Consequently, link $a \rightarrow x$ is broken and is put into the queue of pending links Q_P to be assigned a new channel.

(Step 4.) Similarly, the CHANGE_IF procedure is invoked to select which radio on y has to switch to channel 4. We assume the radio currently set on channel 3 is selected. This implies that link $g \rightarrow y$ is now broken and is put into the queue Q_P.

(Step 5.) Channel 4 is then assigned to the extracted link $x \rightarrow y$ by configuring the selected radios on x and y accordingly.

(Step 6.) Link $a \rightarrow x$ is extracted from Q_P and needs to be assigned a new channel since no radio of x is set on channel 1.

(Step 7.) Since nodes a and x share a channel, i.e., channel 3, we assigned it to link $a \rightarrow x$.

(Step 8.) Link $g \rightarrow y$ is extracted from Q_P and needs to be assigned a new channel since no radio of y is set on channel 3.

(Step 9.) Nodes g and y share channel 4, which is thus assigned to link $g \rightarrow y$. The queue Q_P is now empty and MVCRA-R continues by extracting a new link from Q.

3.4.1 Large simulation campaign results

In this Section we present the results of the simulations carried out to evaluate the performance of the MVCRA-R algorithm. We compare MVCRA-R to FCRA

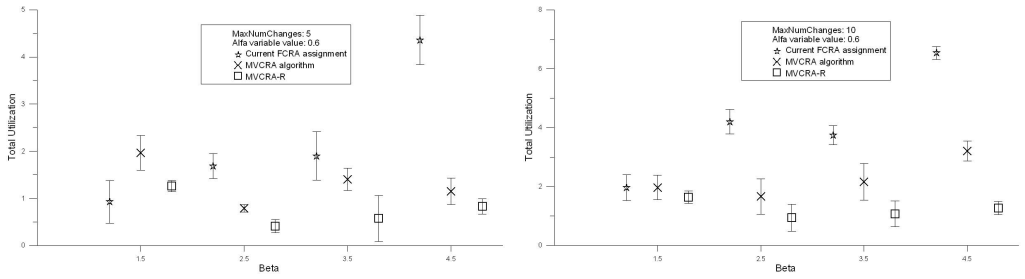
(Flow-based Channel and Rate Assignment) [26] and our MVCRA (Minimum Variation Channel Re-Assignment) [6] in terms of different performance indices: maximum total utilization, number of radios that have to switch channel and average network throughput.

Simulation setup

We consider a wireless mesh network of 25 nodes, each of which is equipped with two or three radios with an omnidirectional antenna. Nodes are randomly placed in a 300 $\times$ 300 square meter area. The standard 802.11 MAC layer is adopted and 6 non-overlapping channels are considered. The available transmission rates are those provided for 802.11a (6, 9, 12, 18, 24, 36, 48 and 54 $Mbps$).

The initial set of pre-computed flow rates is determined as follows. A subset of mesh nodes is identified as source or destination of traffic flows. The traffic demand for each source-destination pair is determined according to a random variable uniformly distributed in $[1, 5]Mbps$. Also, all the link disjoint paths between a source and a destination are computed and the corresponding traffic demand is equally split over all such paths. The sum of the amount of traffic routed on a link over all the source-destination pairs determines the flow rate on that link. FCRA is then used to compute the initial channel assignment.

In the simulations illustrated hereinafter we evaluate the effects of a random variation in the traffic demands, which clearly cause a variation in the set of pre-computed flow rates. The random variation in the traffic demands is determined by two parameters, α and β. With probability α, the traffic demand between a source-destination pair is zeroed. For non-zeroed source-destination pairs, the traffic demand is determined by a random variable uniformly distributed in $[\beta, 5\beta]Mbps$, i.e., the boundaries of the uniform random variable are scaled by a factor β with respect to the uniform random variable used to generate the initial set of traffic demands. Given the new set of traffic demands, the new set of pre-computed flow rates is determined as described above.



(a) $MaxNumChanges$=5 (b) $MaxNumChanges$=10

Figure 3.16: Performance degradation with the current channel assignment

Performance degradation with the current channel assignment

The aim of this set of simulations is to show the performance degradation occurring if the current channel assignment is kept despite a variation in the traffic demands.

A simulation consists in computing the channel assignment for the initial set of pre-computed flow rates by using FCRA and then varying the set of traffic demands as described in Section 3.4.1. Then, we measure the maximum total utilization resulting from the current channel assignment with the new set of pre-computed flow rates.

Starting from the current channel assignment, we also run MVCRA and MVCRA-R and measured the achieved maximum total utilization. Each such simulation is repeated 10 times and the distribution of the maximum total utilization achieved by each algorithm is represented in the Figures by the interval average value $\pm$ standard deviation.

Figures 3.16a and 3.16b show the results obtained when MVCRA and MVCRA-R use $MaxNumChanges$ equal to, respectively, 5 and 10. We performed different sets of simulations using different values of β (from 1.5 to 4.5). We also performed simulations with different values of α, but we just report the results obtained for $\alpha = 0.6$.

Figures 3.16a and 3.16b show that for a very slight variation in the traffic demands ($\beta = 1.5$, $\alpha = 0.6$), there is no advantage in re-adjusting channels by using MVCRA or MVCRA-R. However, as the variation in the traffic demands

becomes more marked (with $\beta = 2.5$ already), the need to recompute channels is more and more evident.

Thus, this set of simulations show that recomputing channels is advantageous even for slight variations in the traffic demands. Also, by comparing Figure 3.16a and Figure 3.16b, we can observe that the performance of MVCRA and MVCRA-R improves with respect to FCRA by increasing of $MaxNumChanges$ and MVCRA-R outperforms MVCRA. Such observations are confirmed by the results shown in the next subsections.

Evaluation of the number of interfaces changing channel

This subsection aims to compare the number of radio re-configurations required by FCRA and MVCRA-R when the traffic demands are subject to a given variation.

Figure 3.17 reports the results obtained for different values of $MaxNumChanges$ (from 4 to 22) and for a given pair of values for α and β ($\alpha = 0.3$, $\beta = 5.5$). FCRA is independent of the value of $MaxNumChanges$ and different numbers of radio-configurations achieved by FCRA as shown in the Figure are explained by considering that the probability distributions of the initial set of traffic demands and of its variation are the same for all the simulations, but different simulations are clearly characterized by different outcomes of those random variables.

From Figure 3.17 it can be firstly noted that MVCRA-R satisfies the constraint that the number of interfaces that need to change channel should be less than $MaxNumChanges$ with a good approximation. Only in a few cases the number of radios that need to switch channel exceeds $MaxNumChanges$ (by at most 2 or 3 units). This is due to the need of emptying the Q_P queue of pending links after the last link is extracted from the Q queue.

Secondly, we note that the number of radio re-configurations required by MVCRA-R keeps smaller than those required by FCRA until $MaxNumChanges$ grows quite large (above 20). In the next subsections we will compare the performance of FCRA and MVCRA-R for different values of $MaxNumChanges$, thus keeping in mind that, for values below 20, MVCRA-R allows to save a number of

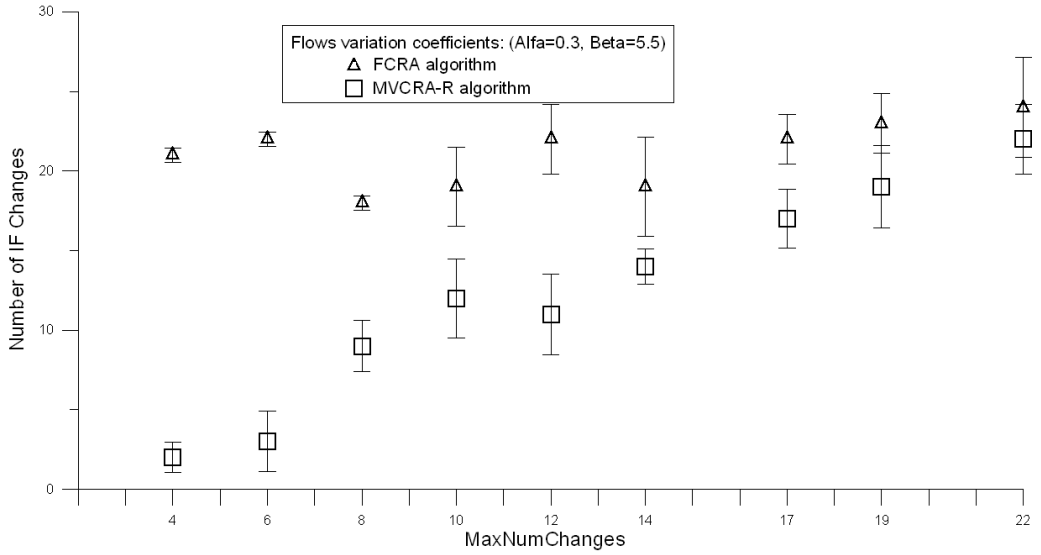


Figure 3.17: Total number of radio re-configurations needed for different values of $MaxNumChanges$

radio re-configurations with respect to FCRA.

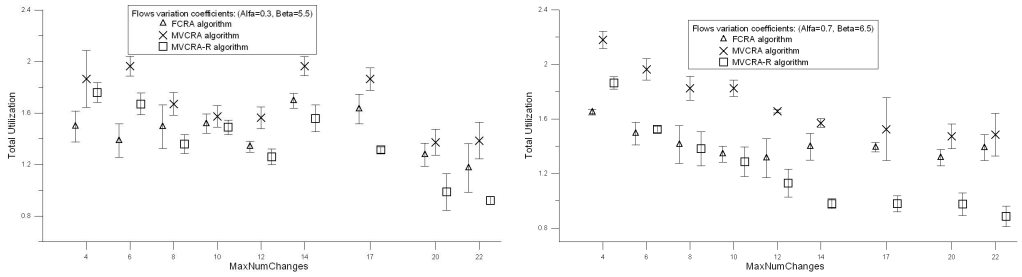
Evaluation of the maximum total utilization

This set of simulations compares MVCRA-R to MVCRA and FCRA in terms of the achieved maximum U_{tot}. In each simulation, the channel assignment for a given initial set of pre-computed flow rates is determined by using FCRA. Then, a variation in the traffic demands is considered and the maximum U_{tot} achieved by MVCRA and MVCRA-R (with the current channel assignment and the new set of pre-computed flow rates) is compared to the maximum U_{tot} achieved by a new execution of FCRA (with the new set of pre-computed flow rates).

Figures 3.18a and 3.18b report the maximum U_{tot} achieved for two different variations in the traffic demands ($\alpha = 0.3$, $\beta = 5.5$ and $\alpha = 0.7$, $\beta = 6.5$, respectively) and for different values of $MaxNumChanges$.

It can be observed that for very small values of $MaxNumChanges$ (up to 6) FCRA outperforms MVCRA-R as it achieves a smaller maximum U_{tot}.

However, as $MaxNumChanges$ increases, MVCRA-R performs better and better than FCRA. Such result, together with the result of the previous subsection,



(a) $\alpha = 0.3,\ \beta = 5.5$ (b) $\alpha = 0.7,\ \beta = 6.5$

Figure 3.18: Maximum U_{tot} for different values of $MaxNumChanges$

shows that for proper values of $MaxNumChanges$ (from 8 to 20 in the considered scenario) MVCRA-R both achieves a smaller U_{tot} and requires a smaller number of radio re-configurations with respect to FCRA.

Also, the maximum U_{tot} achieved by MVCRA-R is always smaller than that of MVCRA, thus proving that the presented enhancements to the algorithm are effective.

Evaluation of the network throughput

Here we compare MVCRA-R to MVCRA and FCRA in terms of network throughput. These simulations are carried out by using the *ns-2* network simulator. We use the standard 802.11 MAC protocol, omnidirectional antennas and the two-ray ground reflection propagation model. The initial set of traffic demands is generated by UDP traffic sources starting at time 0. The initial channel assignment is computed by using FCRA. We wait some time for the network to reach a steady state and then (at about time 40 seconds) we simulate a variation in the traffic demands ($\alpha = 0.3,\ \beta = 5.5$).

We evaluate the average throughput (over the period from the time when a steady state is reached till the end of the simulation - time 200 seconds) achieved in three cases: (i) the current channel assignment is unchanged; (ii) FCRA is used to compute a new channel assignment considering the new set of pre-computed flow rates; (iii) MVCRA-R is used to re-assign channels starting from the current assignment and the new set of pre-computed flow rates. The results shown in Fig-

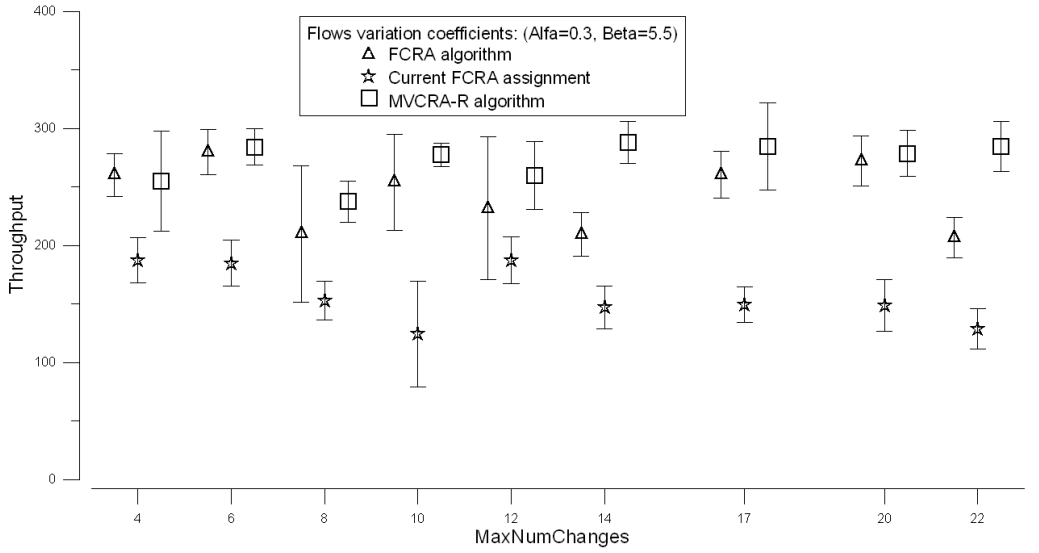


Figure 3.19: Average network throughput for different values of $MaxNumChanges$

ure 3.19 (in $kbps$) are averaged over ten simulation runs. The channel switching operation on a radio interface is simulated as a random time interval (of duration uniformly distributed between $0.1s$ and 1s) in which the radio cannot receive nor transmit packets.

Figure 3.19 shows that the lowest throughput is always achieved in the case that the channel assignment is unchanged, thus proving the need of re-computing channels. We can also observe that for very small values of $MaxNumChanges$ (up to 6) a complete re-computation of channels by FCRA achieves the best throughput. However, for slightly larger values of $MaxNumChanges$, the highest throughput is achieved in the case that channels are re-adjusted by MVCRA-R. This result shows that, by properly setting the value of $MaxNumChanges$, MVCRA-R achieves a higher throughput than FCRA with a smaller number of radio re-configurations.

In order to show the impact of switching radios on the network throughput, we show in Figure 3.20 the average throughput on short time intervals ($5s$) obtained for FCRA and MVCRA-R in one of the simulations described above (specifically, one with $MaxNumChanges$=8). We recall that, after about 40 seconds from the

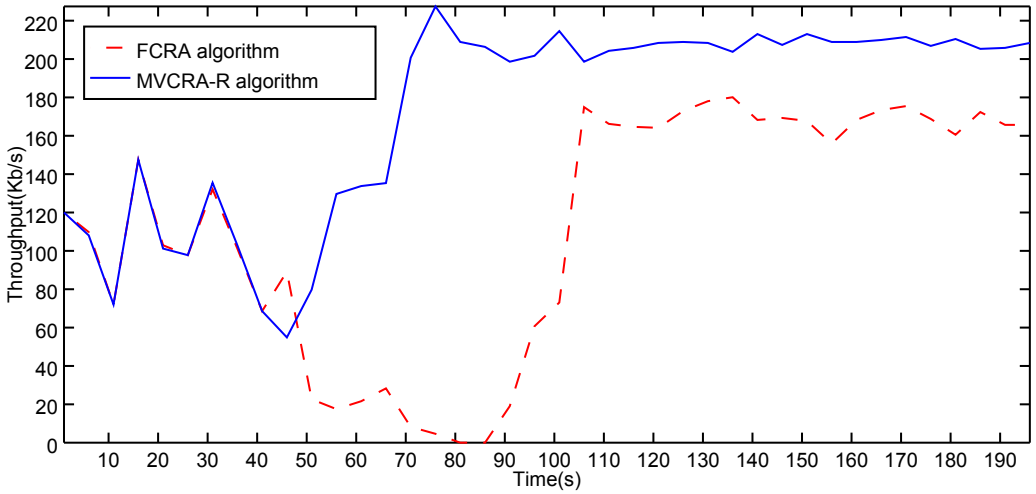


Figure 3.20: Network throughput as a function of time ($MaxNumChanges$=7)

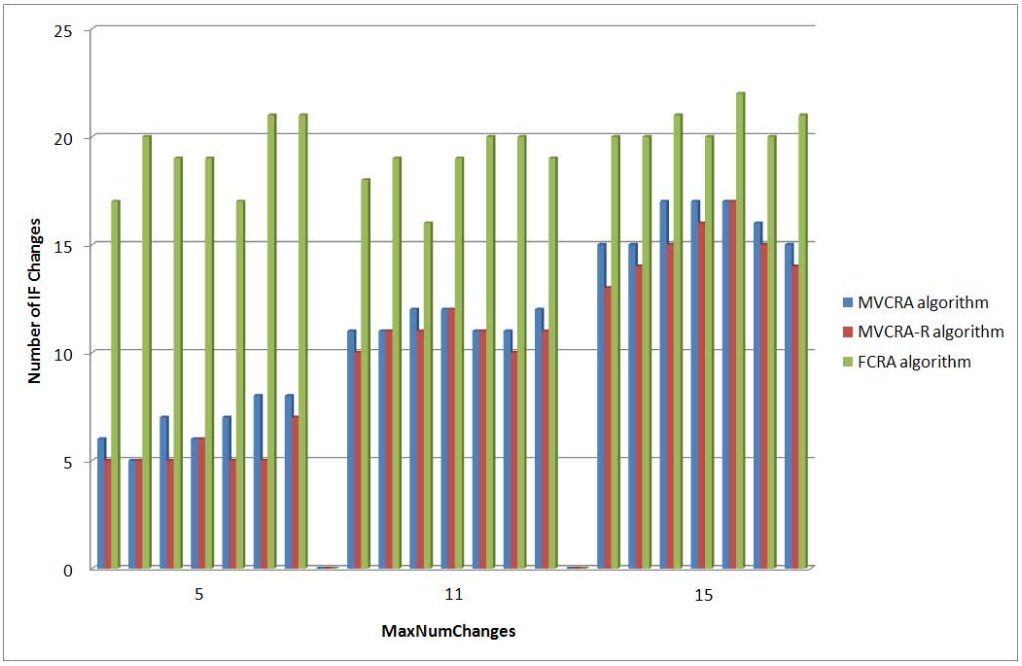
beginning of the simulation, a variation in the traffic demands is simulated and the assignment of channels is then determined by running FCRA or MVCRA-R.

Figure 3.20 clearly shows that the large number of radio re-configurations required by FCRA causes a long time interval where the network throughput is very low. MVCRA-R, instead, requiring a smaller number of radio re-configurations, causes less disruption and the throughput rapidly grows till the steady state value. As expected, given the previous results, the steady state throughput achieved by MVCRA-R is higher than FCRA.

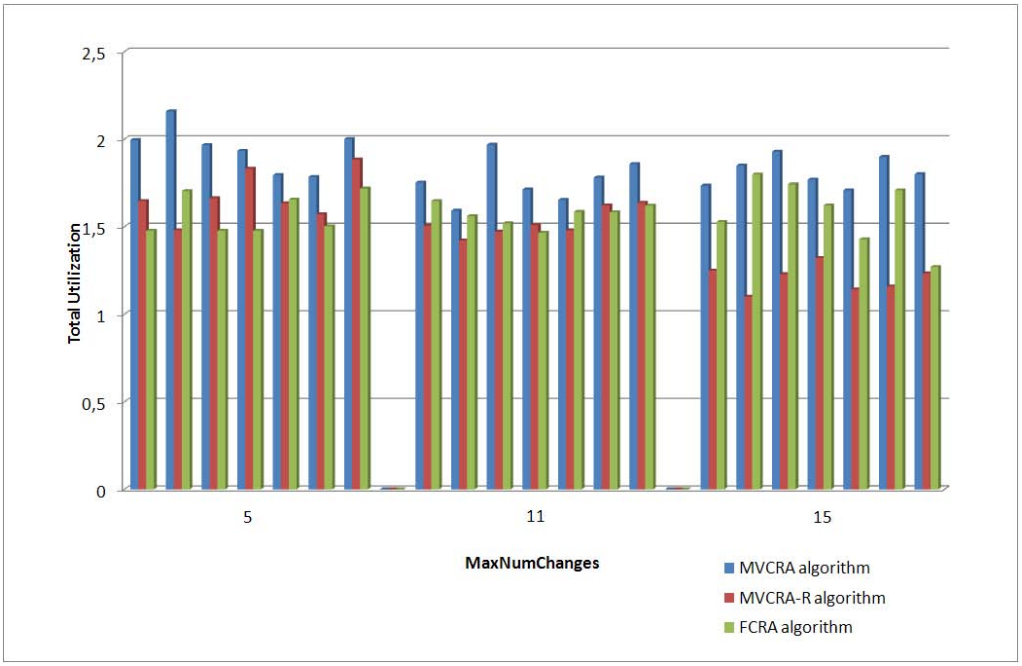
Repeated variations

We conducted another set of simulations to verify whether the performance of MVCRA-R degrades after a number of consecutive variations in the pre-computed flow rates. Specifically, we simulated 7 consecutive variations in the traffic demands using $\alpha = 0.3$ and $\beta = 5.5$. After every variation, MVCRA-R and MVCRA are fed with the current channel assignment and the new set of flow rates, while FCRA is only fed with the new set of flow rates and computes a new channel assignment from scratch. Also, we considered three different values of $MaxNumChanges$ (5, 11 and 15) for MVCRA-R and MVCRA.

Results are shown in Figure 3.21. In particular, Figure 3.21a shows the number



(a) Total number of radio re-configurations needed in case of repeated flow variations



(b) Maximum U_{tot} in case of repeated flow variations

Figure 3.21: Repeated variations in the traffic demands

of radios changing channel after each flow variation. It can be observed that, for all the variations in the traffic demands, both MVCRA-R and MVCRA are able to satisfy (with a few exceptions) the constraint on the maximum number of radio re-configurations, whereas FCRA requires a larger number of radios to switch channel. Instead, Figure 3.21b shows the maximum U_{tot} achieved after each flow variation. In line with the results shown in the previous subsections, it can be observed that, for small values (e.g., 5) of $MaxNumChanges$, FCRA achieves a slightly lower maximum U_{tot}, whereas, for slightly larger values of $MaxNumChanges$, MVCRA-R outperforms both FCRA and MVCRA.

Therefore, these results show that the performance of MVCRA-R does not degrade after repeated variations in the traffic demands, but MVCRA-R is able to adapt to repeated flow variations and keep a high level of performance.

3.5 Chapter Conclusions

In this Chapter we addressed the channel assignment problem, a fundamental design issue in multi-radio wireless mesh networks. We considered two approaches.

In Section 3.3, we presented a new channel, power and rate assignment heuristic and showed that it outperforms previous proposals. With the $FCPRA$ algorithm, we consider as inputs the pre-computed flow rates returned by the routing problem solutions, and we try to tune transmission power, rate and channel of a link in order to satisfy the sufficient condition for a set of flow rates to be scheduled.

The channel algorithm is designed to assign the links parameters at large time scales. Hence, it may be computational and time demanding. Moreover, it had the drawback that if run after a traffic flow variation, it may return a completely different channel assignment, thus requiring time to enforce the new configuration.

Then, following the second proposed approach, we introduced in Section 3.4 the Minimum Variation Channel and Rate ReAssignment (MVCRA-R) algorithm, whose aim is to overcome some drawbacks with channel assignment algorithms being re-computed upon every variation in the traffic pattern. The MVCRA-R

algorithm takes into account the current channel assignment and the new set of pre-computed flow rates and attempts to minimize the maximum total utilization over all the collision domains, while constraining the number of radios that can be assigned to a new channel. The MVCRA-R algorithm leverages the possibility to adjust the link transmission rates and presents some solutions to limit the ripple effect. We performed extensive simulation studies that confirmed that MVCRA-R outperforms the other assignment solutions we compared it to.

The aims were twofold: in Section 3.3 we proposed a channel assignment that can cope with the nominal traffic flow rates, ignoring the current network situation, hence focusing on the satisfaction of the scheduling condition. Such an algorithm is run once in a while and it may be tolerant to non-realtime computation times; in Section 3.4 we planned a channel assignment algorithm that would be run more frequently, after traffic flow rates variations. While the priority is still the satisfaction of the scheduling condition for the flow rates, we tried to reduce the algorithm complexity and the impact of frequent reassignments on the throughput performance of the network.

Chapter 4

An hybrid approach to channel assignment

In Chapter 2 we exposed some of the critical issues concerning traffic optimization in multi-radio Wireless Mesh Networks, and in Chapter 3 we faced up to the channel assignment problem, showed its various aspects and presented some heuristic methods to come to a good assignment. IEEE Working Groups have standardized several orthogonal frequency bands that, together with proper channel assignment, allow multi-radio mesh routers to simultaneously communicate on orthogonal channels.

We showed how each wireless link has a collision domain, defined in equation 3.3, and how the size of the collision domain affects the available bandwidth on each link. We showed, then, how the routing problem and the channel assignment problem are rigorously interconnected and have to be treated as one.

Channel assignment algorithms, based on the pre-computed flow rates knowledge, are able to properly tune transmission parameters configuration, therefore bounding the entity of collision domains in the network. Finally, channel assignment decides how many wireless links have to share the same channel *and* consequently the amount of available bandwidth.

As outlined in Section 2.3, knowledge of the amount of traffic that has to be routed over wireless links is fundamental for a proper channel assignment.

In this Chapter, our objective is to plan a channel assignment scheme that takes advantage of some information from the application layer and that is able to opti-

mize both network resources usage and overlaying application performance. Such channel assignment algorithm would be inherently cross-layer, since it would involve information and parameters from both DataLink layer and Application layer. We envision that generic applications would supply all the routers in the network with information about the amount of traffic they intend to produce and the destination nodes of such traffic.

Together with information about the source-destination routing paths, the channel assignment algorithm would exploit this information and will adapt the channel assignment to the traffic requests, trying to obtain, on each link, an available bandwidth that exceeds the amount of traffic to be routed on that link.

Moreover, in this Chapter we present a generic framework which can be used to experiment with channel assignment algorithms in WMNs. The framework is based on OLSR [2], a 2.5 channel abstraction and scheduling layer based on Net-X [3] and a modified Network Interface Controller (NIC) driver capable of fast channel switching.

Net-X comes with an hybrid channel assignment approach, as it applies a semi-dynamic assignment to its fixed interface (used primarily for receiving data from neighbours) and a dynamic assignment to the switchable interfaces (used to transmit data to its neighbours). The assignment of the available channels to the fixed interface will be made by the channel assignment algorithm we introduce in this Chapter.

Finally, we use the framework to develop a traffic demand-aware channel assignment algorithm. We introduce the traffic demand idea and, based on the observation that traffic demands are mostly not uniformly distributed in a WMN (e.g. demands are higher close to internet gateways), we formulate a channel assignment algorithm that exploits this circumstance.

In particular, we designed and implemented two main components, one responsible for keeping updated the traffic demand information among the nodes, the other responsible for the channel assignment procedure, with the task of assigning the least used channels to the most busy wireless links. Then, we implemented our algorithm in a distributed fashion and we tested it on the KAUMesh,

the wireless mesh testbed of the Karlstad University.

In this Chapter, in Section 4.1 and 4.2, we introduce the KAUMesh wireless testbed and present the main Net-X module features. Then, in Section 4.3, we present a quick overview on the OLSR routing protocol, and finally in Section 4.4 we present the traffic demand-aware channel assignment algorithm introduced here.

4.1 Overview on the KAUMesh wireless testbed

In this Section we present KAUMesh, an experimental Wireless Broadband Mesh Network based on 802.11a/b/g WLAN based devices (the mesh node) that has been deployed at the Karlstad University Campus, in Sweden.

KAUMesh comprises 20 stationary mesh routers, which can provide connectivity to standard 802.11b/g based clients as any normal WLAN Accesspoint. Mesh backhaul connectivity is established through mesh nodes which, by means of wireless links, relay packets among each other and towards the gateway mesh nodes, allowing the mesh clients access to Internet.

The number of interfaces used for backhaul access to create the mesh physically limits the number of transmissions and receptions that can take place simultaneously. The routers are all equipped with 6 radio interfaces, but in our experiments we use only three radios, two for backhaul connectivity, one for mesh client network access.

KAUMesh is a multi-radio, multi-channel mesh solution, based on Net-X [3]. It exploits the OLSR routing algorithm, and is adapted to run on a high-performance network processor platform and has been extended using several additional functionalities.

One interface is used for transparent client access, using typically 802.11b/g mode. The mesh backhaul connectivity, instead, is established using two radio interfaces. One of these interfaces is called ”fixed interface” and is assigned for relatively long time intervals to a ”fixed channel”. This fixed channel is selected by the channel assignment algorithm, and used to receive and eventually send

packets.

The second interface at each node is called ”switchable interface”. It can be switched dynamically between any of the remaining channels, and is only used to send packets to neighbours. Maximum and Minimum Channel Switch Intervals can be configured through command line.

All data sent to a node must be over the fixed channel, because the node is guaranteed to always listen to the fixed channel. With this interface configuration, connectivity is maintained and all channels are used. A node may send data to a neighbour using the fixed interface, if both nodes use a common fixed channel; otherwise, the switchable interface is tuned to the neighbour's fixed channel, and data is sent out over the switchable interface. The second interface, then, is dynamically switched on a packet-by-packet basis. In that sense, the mechanisms do not require coordination channels.

In any case, we should still manage the radio interfaces switching operations. Therefore, what we are looking for is a transparent mechanism to manage the frequently switching channels operation on the dynamic interfaces, while leaving us the freedom to decide which channel assignment to enforce on the ”fixed” radio interfaces. To this end, we exploit the Net-X module features. As we will show in Section 4.2, the Net-X module introduces one single virtual interface to abstract multiple interfaces. This makes the dynamic radio switching operations easier, since they are all managed by the Net-X module.

When a packet has to be sent, it is given to the virtual interface. Then, according to the destination channel, it is inserted in the correct channel queue of the switchable interface. All the radio switching operations, then, are managed by the Net-X module, leaving us free to focus only on the channel assignment issue.

In addition, to obtain a distributed algorithm, we need to distribute information among network nodes. This is made possible by exploiting the broadcast messages sent by the proactive OLSR routing protocol. The OLSR routing protocol will be explored later in Section 4.3. By suitably modifying the broadcast messages of OLSR, it was possible to spread both information regarding current radio channel configurations, and the information needed to the channel assign-

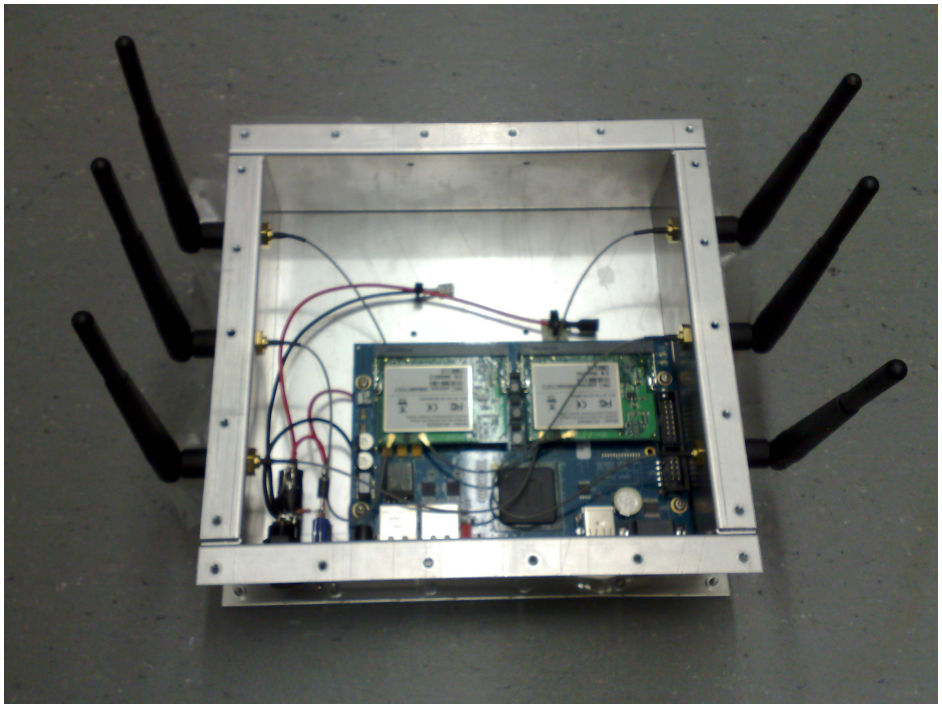


Figure 4.1: A KAUMesh wireless router

ment algorithm.

Each node periodically sends out broadcast ”hello” messages on all channels. The broadcast messages are received by all the 2-hop neighbours. Hello messages include the fixed channel of the node, and the fixed channels of all 1-hop neighbours. A node can thus learn about the fixed channels used by all nodes in its 2-hop neighbourhood.

For monitoring purpose, each mesh node has a wired Ethernet connection to a managed network in order to send monitoring data to a monitoring server, located in the fixed network. The wired connection may also be used for sending configuration information to the mesh nodes as well as for downloading kernel versions to the mesh nodes.

In Figure 4.1 we show one typical KAUMesh wireless router, together with the multiple radio interfaces it is equipped with.

Each mesh node has been built using commercial off-the-shelf single board platforms, based on the Cambria GW2358-4 Network Computer (see http://www.

gateworks.com), which comes with Four Type III Mini-PCI Sockets for the wireless radio interface cards. Each board is equipped with 3 x 802.11a/b/g Atheros miniPCI WLAN radio cards. One card is used for client access, the other two for establishing the mesh backhaul in the 802.11a band.

4.2 Net-X module

The Net-X[3] project was born to give full support to practical utilization of next-generation interface capabilities. For example, the ability to frequently switch between different frequency bands, data rates or transmitting powers, and the ability to equip multiple radios.

Due to the poor support of current operating systems for advanced wireless interface functionalities, using the Net-X project, implementing channel assignment protocols for multi-radio wireless mesh networks becomes a much easier task. In particular, Net-X provides a supporting architecture that takes full control over radio interfaces and channels used.

One important feature of the Net-X implementation is that one single virtual interface is used to abstract multiple radio interfaces, that is Net-X hides the notion of multiple interfaces availability.

The Net-X approach is to develop a channel abstraction layer that is responsible for interface switching operations. Moreover, the Net-X module abstracts out the details of the procedures that are employed to switch interfaces, providing a transparent control layer to routing and link layer protocols. The channel abstraction layer module logically belongs to the link layer: it is independent of the specific adopted drivers, and it shows one virtual interface device to the network layer, thus hiding the multi-interface support management.

This solution is particularly suitable to a Linux implementation, since Linux is able to bond multiple network interfaces into a single virtual interface, using a ”bonding driver” that is placed between network layer and device drivers. The channel abstraction layer is implemented as a new feature of the bonding driver. In particular, Net-X strengthens the Linux operating system by introducing features

at the channel abstraction layer. In particular, we have three main components, i.e., the unicast component, the broadcast component and the scheduling and queuing component. Their peculiarities will be analyzed below.

Unicast Component

One of the main features of the Net-X module is that it allows network layer to control the channels used to reach a neighbour, instead of just the used interface. In single radio networks or in those multi-channel networks where there is a one-to-one mapping between channels and interfaces (e.g., m different channels over m different radios), there is a direct correspondence between channels and interfaces, hence it is possible to refer to the used channel to reach a node by referring to the used interface.

When the number of radios is smaller than the number of used channels and switchable interface protocols are implemented, we need a more fine-grained control over the channels used to reach a neighbour. Hence, with this feature, it is now possible to implement protocols that allow single radio nodes to send data to different nodes over different channels. Moreover, a single interface could be used to reach different nodes using different data rates or transmitting powers. This feature is enforced with a specific module, which maintains a specific unicast table. Each unicast table entry is a tuple, made of destination IP address, destination interface receiving channel, and the real interface card that will be used to transmit the packet. This table is populated by the multichannel protocol.

When a packet is received from network layer, its IP destination address is looked up into the table in order to check the interface and the channel to be used for actual transmission.

Broadcast Component

It allows to specify which channels broadcast packets have to be sent on. Hence, multichannel broadcast support module is incorporated into the Linux kernel. The broadcast component keeps a broadcast table, which maintains a list of channels on which copies of broadcast packets have to be sent and a list of the real interfaces

to be used. Changing the broadcast table entries over time grants the flexibility to select the set of channels used to broadcast packets.

When a broadcast packet is received from network layer, the broadcast table is checked and the broadcast packet copies are sent over each listed channel, using the proper radio interfaces.

Scheduling and queuing Component

Queue module to provide enhanced support for buffering and scheduling. Interface channel switching operations are needed to enable communication with different neighbours. Each radio interface has separate packet queues for each of the available channels. Moreover, each radio interface has different scheduling functions for sending packets, that can be modified to provide higher priority to some channels.

The queue component receives packets from the unicast or broadcast component, together with information about channel and interface to use for transmission. The queue module, then, puts the received packet into the corresponding interface channel queue. A switching operation is needed if the neighbour fixed interface is set on channel c and there is no interface set on channel c. It may happen, though, that all the local radio interfaces are being used, i.e., a packet is being transmitted and other packets are waiting in the specific interface channel queue. Immediate switching would lead to loss of queued packets, and hence it is not feasible. When an immediate switching is not advised, new packets have to be queued for the time a switching operation is agreed.

Moreover, channel switching operations take time and too frequent switching operations may interrupt actual applications traffic. In some cases, then, it is better to send multiple packets on a channel queue before switching to a new channel. A scheduling functionality, that allows to transmit queued packets while reducing frequent switching, is then introduced. When switching to a new channel queue, the scheduler stays on that channel queue for at least T_{min} seconds. If the queue is continuously loaded, the channel switching takes place after T_{max} seconds, where $T_{max} > T_{min}$. Before actually switching the channel, the component queries the

device driver asking whether or not all the packets sent to the interface since last switch have been indeed transmitted.

If packets are still pending, then the switching is delayed by T_{defer} seconds and late packets have chance to be sent out. In any case, if a switch operation is initiated, the channel queue is flushed.

4.3 OLSR link state routing algorithm

In this Section we give a quick overview of the OLSR routing algorithm [2], giving a closer look to what routing messages are used, to how they are exchanged and to how it is possible to exploit them to carry current channel assignment information.

The OLSR routing protocol is a link state proactive routing protocol. This means that routing control messages are periodically exchanged between network nodes. Each node selects a set of neighbour as MultiPoint Relays (MPRs): only nodes elected as MPRs can forward OLSR routing messages and retransmit the selector broadcast packets.

Thanks to the MPRs relay role, routing control messages are flooded all over the network by reducing the number of total transmissions. The idea behind MPRs, then, is to optimize the classical flooding mechanism and to adapt it to the interference-sensible wireless networks. The MPR set is calculated in such a way that a generic selector can reach, through the neighbours in the MPR set, all 2-hop neighbours. In particular, MPRs spread broadcast information only for their own selectors. Nodes selected as MPR announce themselves in control messages, specifying for which nodes they have reliability. Moreover, MPRs are used to form the route from a given node to any destination in the network.

OLSR routing protocol define three types of messages:

1. **HELLO** messages, shown in Figure 4.2, are used for link sensing, neighbour detection and MPR signaling. The emission interval is of 2 seconds. In particular, the ”Willingness” field specifies the willingness of a node to be selected as a MPR, while the Link Code field is used to specify information about the link between the sender interface and the following list of

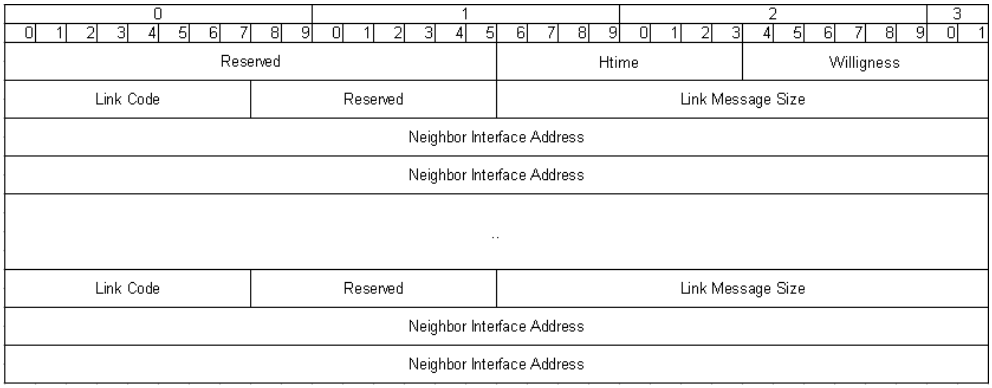


Figure 4.2: OLSR HELLO message

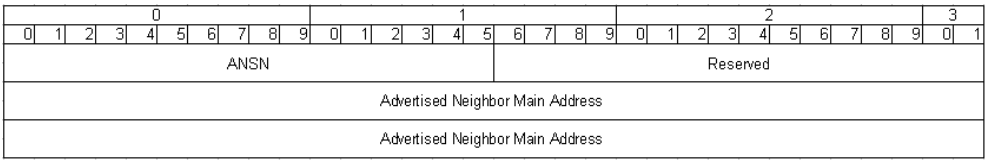


Figure 4.3: OLSR TC message

neighbours.

2. Topology Control (**TC**) messages, shown in Figure 4.3, are introduced to perform the task of topology declaration through link state advertisement. A node must at least disseminate information about links between itself and its neighbours, in order to properly allow the routing table computations. The addresses of the advertised neighbours are put into the Advertised Neighbour Main Address field. TC messages emission interval is of 5 seconds.

3. Multiple Interface Declaration **MID** messages, used to declare the presence of multiple interfaces on a node. In this case the MID emission interval is the same as for the TC message.

Through the exchange of OLSR messages, each node acquires information about the network. In particular, the most remarkable messages are:

1. **Local Link Information:** information about links to neighbours, like local and neighbour interface addresses.

2. **Neighbour Information:** information about neighbours, 2-hop neighbours, MPRs and MPR selectors. For each neighbour, address, status and node's willingness to be selected as MPR information are stored. For each 2-hop neighbour, status and address information about the intermediate 1-hop neighbour is stored. Finally, the MPR selector list stores the addresses of those neighbours that have selected the current node as a MPR.

3. **Topology Information:** each node in the network has complete knowledge about the network topology. This information is acquired from TC messages and is used for routing table computations. For each destination, a node stores information about the last hop through which that destination can be reached (typically an MPR).

Finally, the routing table is computed according with the information contained in the local link information database and the topology set. In particular, for each destination, each routing table entry contains information about the closest next hop in the route, the wireless interface that has to be used to reach the next hop and the hop-count distance. To build the routing table, the link state Dijkstra algorithm is deployed.

4.4 The Distributed Channel Assignment Algorithm

In this Section we describe our distributed channel assignment solution and how we developed it taking advantage of the framework described in Sections 4.2 and 4.3.

In Figure 4.4 we reported the framework architecture. Implemented by the Linux kernel's networking stack, the interface device drivers control access to the interface hardware. The channel abstraction layer described in Section 4.2 resides between the network layer and the interface drivers, having the capability to "bond" the multiple interfaces into a single virtual interface, presented to the network layer. The OLSR module (also known as Olsrd) is an user space daemon that can be easily extended via plugin modules using dynamically linked libraries. This enables the possibility to send broadcast traffic using OLSR (useful

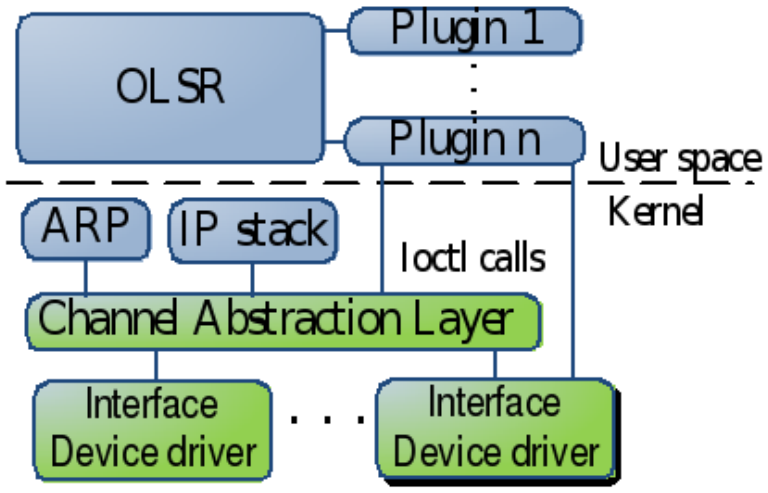


Figure 4.4: Framework architecture

for broadcast-based applications) and also change OLSR functionalities without changing any code in the OLSR daemon.

In particular, we extended the OLSR module with a traffic demand-aware channel assignment plugin. We exploited the Net-X channel assignment approach described in Section 4.2.

We use two radio interfaces for backhaul connectivity, where one interface is considered semi-dynamic, and is used primarily to receive data, while the other interface is used mainly to transmit data and it rapidly switches over different channels. Moreover, with this approach, connectivity and stability are always granted, all channels are used, there is no ripple effect when assigning channels, hence we can focus on reducing interference, not worrying about consequences like possible network fragmentation. Based on this radio dynamics, our algorithm is used to assign channels to the receiving interfaces of all the nodes in the network, i.e., to the semi-dynamic radio interfaces, used mainly for receiving purposes.

In Section 4.4.1 we describe the Channel Assignment plugin we developed, its two main components, and how we implemented it on the KAUMesh testbed. Finally, we show the obtained performance improvements.

4.4.1 The Distributed Channel Assignment Plugin

The goal of our channel assignment algorithm in multi-radio WMNs is to minimize interference while improving the aggregate network capacity and maintaining the connectivity of the network. While the network connectivity is obtained by exploiting the Net-X approach, throughput maximization problem has still to be explored. In particular, we note how channel assignments can be improved if traffic demands at mesh routers are considered into the assignment process.

This knowledge may be successfully exploited, by ensuring that interfering nodes with high individual demands are assigned to less loaded channels. We refer to this as an hybrid traffic demand-aware channel assignment.

In particular, given the Net-X based interface configuration, we focus our attention on the receiving interface of a node. If we want to reduce interference by considering the amount of traffic that has to be routed on a given wireless link, we have then to refer to the amount of traffic that each node expects to receive on the fixed wireless interface.

We define the term traffic *demand* as the amount of data that a node receives from other nodes per unit time. Obtaining the realistic amount of traffic each source destination couple exchanges is a challenging task, and it is independent from the channel assignment problem. Different techniques can be applied in order to measure and predict traffic demands, for example by predicting future demands based on historical information [50]. Since we are mainly interested in the channel assignment, we assume that the demands are provided by an external entity. One possibility is to supply applications with a cross-layer interface in order to have explicit information about traffic demands directly from the source applications.

Our demand-aware channel assignment's plugin is composed by two components: the probe component and the channel assignment component.

The probe component

The probe component is responsible for maintaining the demand information up-to-date among the nodes. We assume that each traffic source is aware of the traffic

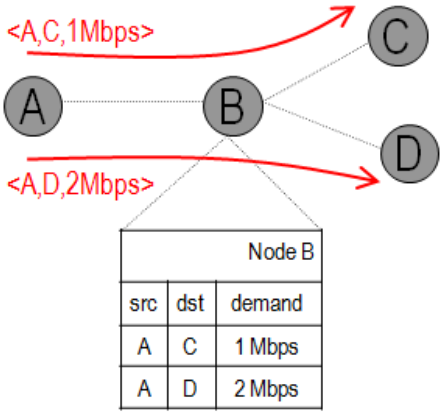


Figure 4.5: Probe Messages status update

it asks to be routed. Hence, each traffic source generates probe messages that contain traffic demand information to be spread through the network. This information is propagated via HELLO messages (in its one and two hop neighbours), or via unicast probe messages along the demand path, both generated periodically.

A demand path is composed by a source node, intermediary nodes (responsible to forward traffic between source and destination) and a destination node. The unicast probe messages may incur additional overhead to the system, but it allows the dissemination of demands information along the nodes in the demand path. Each traffic demand probe message spreads information about traffic source, traffic destination and demand of traffic. At the same time, each node receiving a probe message maintains its list of <Source, Destination, Demand>.

In Figure 4.5 we see how node B updates its traffic demand status when probe messages are received. Node A generates two traffic flows of 1Mb/s and 2Mb/s towards node C and D, respectively, and generates two probe messages, as shown in Figure 4.5. When node B receives the probe messages, before forwarding them, it updates its traffic demand database: it now has two entries, one about the 1Mb/s flow from A to C and the other about the 2Mb/s flow towards D. Node B is now aware that it will receive a total amount of 3Mb/s on its receiving interface and it can supply this information to the channel assignment component.

```
DEMAND-AWARE_ASSIGNMENT(n, c_cur)
 1  C ← {c_1, ··· , c_C}
 2  S ← 1 × C
 3  S ← INIT(0)
 4  if T(n) = 0
 5     then return c_cur
 6  D ← d_k   ∀k ∈ T
 7  SORT(D)
 8  for each d_k ∈ D
 9       do
10          c_k ← argmin_{c∈C} S
11          S(c_k) += d_k
```

Figure 4.6: Pseudo-code DEMAND-AWARE_ASSIGNMENT

The channel assignment component

Due to application traffic and users communication, each wireless node receives a given amount of traffic on the receiving interface. Since the amount of available bandwidth on a channel is bounded by the number of nodes sharing that channel (see Chapter 3), in order to have enough available bandwidth to guarantee the traffic demands, each node should select the appropriate channel based on the incoming traffic. The criterion behind the channel assignment algorithm is that nodes with high individual demands are assigned to less loaded channels.

The channel assignment component implements a distributed channel assignment algorithm in order to satisfy the traffic demands. The algorithm is shown in Figure 4.6 and it works as follows.

First, the traffic demand vector $D = (d_1, d_2, \cdots, d_{\mathcal{D}})$ is built according to the information spread by the probing messages through the network. Moreover, a temporary data structure, $\mathcal{S}$, is also created, which will be used next to decide which is the best channel to assign.

If a node n is not on any demand path $\mathcal{T}$, such node will eventually be marginal, hence the channel assignment algorithm is not called and the receiving interface is kept on the configured channel c_{cur}, i.e. one of the channels from the

	36	64	112
first allocation step	7		
second allocation step	7	6	
third allocation step	7	6	3
fourth allocation step	7	6	3,2
fifth allocation step	7	6	3,2,1

Table 4.1: Traffic Aware Channel Allocation example

set $C = (c_1, c_2, \cdots, c_C)$ (lines 1-6).

After collecting all the traffic demand information, the demand vector D is sorted in descending order (line 7). Starting from the node with the largest traffic demand, then, each receiving interface is assigned the channel which is currently associated with the least load (lines 8-11). Therefore, computing the $\mathcal{S}$ structure, each node is aware of what currently is the channel occupation in terms of traffic demands.

For simplicity, we represented $\mathcal{S}$ as a simple vector. Actually, the implemented algorithm, thanks to the information coming from the OLSR daemon, maintains a data structure for every node in the same interfering region, including the IP address, the channel occupied by the fixed radio interface and the traffic demand d_i expected on that interface, expressed in Mb/s.

Thanks to the probe messages, each node in the network has the same knowledge about the traffic demands, i.e., each node can run the traffic aware channel assignment algorithm with the same input parameters, hence resulting in the same channel allocation. Each node will go through all the same allocation steps, computing the same demand allocation table, and it will then set its own receiving radio interface accordingly.

For example, if the sorted traffic demand vector is $D_{sort} = (7, 6, 3, 2, 1)$ for five different nodes, and the available channel set is $C = (36, 64, 112)$, the channel allocation is shown in Table 4.1. At the first iteration, all channels are considered free, hence, starting with the node with the largest traffic demand, i.e., 7Mb/s, the channel 36 is assigned. Then, at the second iteration, the node with the traffic demand of 6Mb/s is considered. Since channel 36 is occupied by the node with the

largest demand, channel 64 will be assigned. Therefore, at the end of the second iteration, we have 7Mb/s of traffic on channel 36, 6Mb/s on channel 64. The same holds for the node that has to allocate the traffic demand of 3Mb/s and so on, until all the traffic demands are allocated.

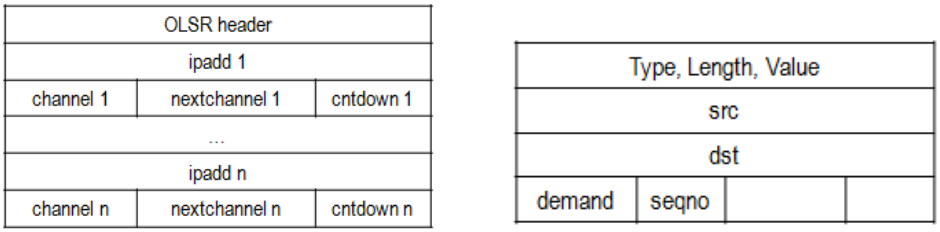
Since every node has the same knowledge about the traffic demands, the algorithm will go through the same steps on each node, thus resulting in the same channel allocation. For the example in Table 4.1, the node with a traffic demand of 2Mb/s would run the channel assignment algorithm and would compute its assigned channel, i.e., channel 112.

4.4.2 Implementation details and experimental evaluation

We implemented both channel assignment and probe messages as an Olsrd plugin and they are both shown in Figure 4.7.

In particular, the channel assignment message is implemented as an OLSR HELLO message extension, and it is broadcasted every 2 seconds. The broadcast message forwarding is an MPRs task, as usual in the OLSR architecture. Each node broadcasts channel assignment information about each neighbour node. The channel assignment broadcast message is shown in Figure 4.7a. After the common OLSR header, it follows the IP address of the announced node and the channel assignment information. In particular, we have the channel that the announced node is currently using on its fixed interface, the next channel that it is currently switching on, and the countdown to the switching procedure. Upon receiving the periodical channel assignment message, each node updates its channel table with the <IPaddress, current channel, next channel, countdown> information. Knowing the countdown value, the node receiving the update knows exactly when a channel switching is going to happen and can synchronize with it. Therefore, each node knows exactly when to update the channel table with the correct value of the current channel.

In Figure 4.7b, we show the unicast probe message. The soft-state probe messages are sent periodically each 5 seconds on the UDP socket on port 2000. Aggregate demands are specified in a common file. After the header fields, the message



(a) Channel assignment broadcast message (b) Traffic demand probe message

Figure 4.7: Channel assignment and traffic demand probe message implementation

specifies the source of the traffic demands, the destination node and the amount of traffic. This message is sent unicast from source towards the destination and each node on the routing path updates its demand table with the information carried by the probe packet. Each intermediate node updates the soft-state information with the <src, dest, demand, seq.no>, and broadcasts this information to the 2-hop neighbourhood. The information contained into the demand table is continuously considered by the channel assignment algorithm.

We evaluated the proposed framework and algorithm in the KAUMesh testbed4.1. KAUMesh consists of 20 Cambria GW2358-4 nodes, each equipped with 3 wireless cards, running Linux 2.6.22, MadWifi 0.94 and the proposed framework. Figure 4.8 depicts the evaluated topology composed of nine nodes mounted in the ceiling of lab rooms and corridors in the engineering building of Karlstad University. The wireless cards where used in the 5 GHz 802.11a band and interface rate set to 6Mb/s.

In this experiment we used the aggregated network throughput as the performance metric for evaluation of our algorithm. We compared the throughput obtained using our algorithm with the local-balancing algorithm proposed by [51] using three 802.11a non-overlapping channels.

In the local-balancing approach, every node obtains the information on the number of nodes assigned to a specific channel, in its one and two hop neighbourhood via periodic hello message. With such information, each node proba-

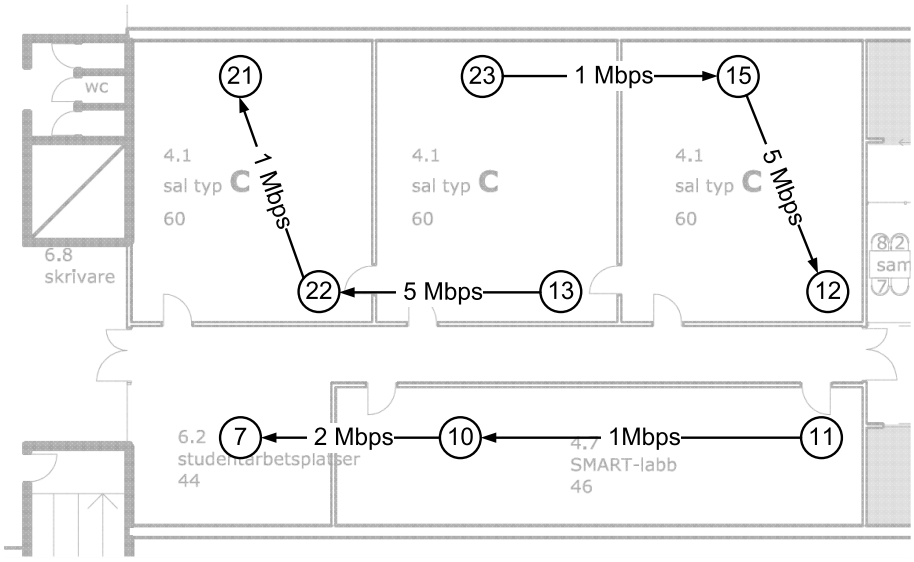


Figure 4.8: Evaluation topology

bilistically re-assigns the receiving radio to the channel that is less utilized by its neighbours, trying to reduce the number of nodes using a given channel. Since the number of combinations of channel allocation is large, we have averaged the best and worst case scenario for the local-balancing algorithm.

We generate six UDP flows at three different data rates (5, 2 and 1 Mb/s), representing the assigned node's demands. Table 4.2 compares the aggregate throughput obtained by both algorithms, given by an average over five runs. As we can see, the demand-aware channel assignment has an improvement factor of 30.5% compared to the local-balancing. This is because with the demand-aware algorithm the nodes have complete knowledge of the traffic demands, hence they can assign their interface to a channel that will support the overall demands, at the same time reducing the interference. This is not the case for the local-balancing algorithm which, for example, in the worst case assigns the highest demands to the same channel.

	Demand-Aware	Local-Balancing
Agg. througput (Mb/s).	13.7 $(+/-0.4)$	10.5 $(+/-0.18)$

Table 4.2: Aggregate throughput versus channel assignment strategies

4.5 Chapter Conclusions

In this Chapter we presented a framework based on the OLSR routing algorithm and the Net-X 2.5 channel abstraction layer. We used this framework to conduct experiments and explore the channel assignment issues in a Wireless Mesh Network. In particular, we developed and implemented a distributed channel assignment algorithm and tested it on the KAUMesh Wireless Mesh Testbed. Our purpose was to improve WMN overall throughput performance by reducing the interference among wireless nodes by exploiting traffic flows information. Our algorithm was then made of two components, one responsible for the traffic demand flow information spreading among the nodes and one responsible for the channel assignment algorithm. We implemented our solution and evaluated it by comparing with the local-balancing approach developed in [51]. We outline how our algorithm outperforms the local-balancing approach, by showing more than 30% throughput performance improvements.

Chapter 5

The routing problem

In Section 2.3 and in Chapter 3 through Chapter 4, we showed how the channel assignment problem is one of the big issues in Wireless Mesh Networks. In particular, we showed how the channel assignment problem is strictly related to the problem of finding the amount of flow to route on each link in order to achieve a given optimization objective.

A solution to such a joint channel assignment and routing problem provides a channel for each radio and a route for each traffic demand, subject to the constraint that the resulting amount of flow routed on each link must not exceed the bandwidth available on that link, as determined by the channel assignment.

Once a solution to the joint channel assignment and routing problem is devised, another problem arises and regards how to route packets in order to achieve the computed flow rate on every link. To this end, recent work [25] showed how to build an interference-free scheduling algorithm. However, a scheduling algorithm requires synchronization among nodes and modified Distributed Coordination Functions (DCFs).

The aim of this Chapter is to present a routing protocol which is aware of the pre-computed flow rates.

One of the properties we want is the capability to keep the link utilization close to the computed flow rate. We deem this property a primary requirement, since the link flow rates come out as the result of an optimization problem and thus trying to achieve them on the network links is the best way to pursue the optimization

objective. Also, as we described in Chapter 3, the obtained computed flow rates can be scheduled, meaning that they take into account the channel capacity and the number of nodes sharing the same channel in a neighbourhood. The divergence from the computed flow rates may break the scheduling condition, with the possible consequence of increasing collisions.

In the light of such considerations, we start analyzing a destination based routing protocol, AODV (Ad-hoc On-Demand Distance Vector), and we propose a modification that allows AODV to be aware of the pre-computed flow rates.

AODV routing decisions are based on links costs, and routes are chosen in order to minimize the overall cost on the routing path. Standard AODV link weight is unitary, the routing path hence minimizes the hop-count distance between two nodes. One possible modification to the AODV protocol would be to consider arbitrary link weights, so that the best routing path would not be all the times the shortest one.

Therefore, in this Chapter, we analyze the problem of finding the weights for all the links in the network in order to achieve a given set of flow rates.

Finally, we developed a modified version of AODV, named AODV-MLW (AODV with Modified Link Weights), that makes use of such link weights. All the experiments have been carried out by comparing AODV and such modified version of AODV.

5.1 State of the art on routing metrics

A number of papers have been published proposing new routing metrics for wireless mesh networks. Among the first metrics being presented, the expected transmission count metric (ETX) [52] aims to find high-throughput paths on multi-hop wireless networks. ETX minimizes the expected total number of packet transmissions (including retransmissions) required to successfully deliver a packet to the ultimate destination. The ETX metric incorporates the effects of link loss ratios, asymmetry in the loss ratios between the two directions of each link, and interference among the successive links of a path.

Instead, the metric proposed in [27] assigns weights to individual links based on the Expected Transmission Time (ETT) of a packet over the link. The ETT is a function of the loss rate and the bandwidth of the link. The individual link weights are combined into a path metric called Weighted Cumulative ETT (WCETT) that explicitly accounts for the interference among links using the same channel.

In [53], the authors propose a load and interference-aware routing metric named Contention Window Based (CWB) metric. Such metric assigns weights to individual links based on both channel utilization and the average Contention Window used on these links. The individual link weights are combined into path metric that accounts for load balancing and interference between links that use the same channel.

In [54], a routing metric is developed that evaluates each link's effective share of the medium. The Interference-Aware Routing metric (IAR) MAC-level measurements determine the percentage of time each transmission wastes due to interference from other nodes. This wastage occurs in the form of backoff and waiting time, as well as failed transmissions. The goal is to select paths that exhibit the least interference.

Unlike previous efforts, such as the popular ETX, the work in [55] accounts for the fact that MAC protocols incorporate a finite number of transmission attempts per packet. Thus, the performance of a path depends not only on the number of the links on the path and the quality of its links, but also, on the relative positions of the links on the path. Based on this observation, a new path metric, ETOP (Expected number of Transmissions On a Path) is proposed which accurately captures the expected number of link layer transmissions required for reliable end-to-end packet delivery. The authors analytically compute ETOP, which is not trivial, since ETOP is a noncommutative function of the link success probabilities.

The work in [56] is, to the best of our knowledge, the only one taking the channel assignment into account. The proposed routing metric, ALARM (A new Location Aware Routing Metric) is derived from WCETT and includes a component to account for the impact of the distance between links using the same channel. Thus, ALARM still does not take into account the flow rates resulting

from a traffic-aware channel assignment algorithm.

5.2 AODV with Modified Link Weights

In this Section we present an enhanced version of AODV, AODV with Modified Link Weights and denoted as AODV-MLW, which allows to use arbitrary link weights. Such a feature is required to have a destination-based routing protocol that is aware of the pre-computed flow rates returned by a joint channel assignment and routing algorithm, with the procedure described in Chapter 4.

5.2.1 AODV Protocol Overview

We recall that AODV is a reactive routing protocol, i.e., information about nodes and routes are only collected when needed. Common routing protocols are proactive, that is each router discovers information about nodes and routing paths even if there is no traffic demand for those destinations. The AODV routing protocol uses Route Requests (RREQs), Route Replies (RREPs), and Route Errors (RERRs) messages.

When a source node with a given Originator IP needs a valid route to a destination IP address, a RREQ packet is broadcasted to find a route. The range of dissemination of such RREQs is indicated by the TTL in the IP header.

Every route table entry at each node must have the sequence number for the IP address of the destination node for which the route table entry is maintained. This sequence number is called the ”destination sequence number” and it is updated whenever a node receives new information about the sequence number from RREQ, RREP, or RERR messages that may be received related to that destination. AODV requests from each node in the network to maintain the destination sequence number to guarantee loop-free routes towards that node. A destination node increments its own sequence number in two circumstances:

- Immediately before a node originates a route discovery, it has to increment its own sequence number. This prevents conflicts with previously established reverse routes towards the originator of a RREQ.

- Immediately before a destination node originates a RREP in response to a RREQ, it has to update its own sequence number to the maximum of its current sequence number and the destination sequence number in the RREQ packet.

In order to verify that information about a destination is new, the node compares its current sequence number with that obtained from the incoming AODV message.

The only other circumstance in which a node may change the destination sequence number is in response to a lost or expired link to the next hop towards that destination. In this case, for each destination that uses the next hop, the node increments the sequence number and marks the route as invalid.

Whenever any new routing information for an affected destination is received by a node that has marked that route table entry as invalid, the node should update its route table information according to the information contained in the update.

The RREQ message is shown in Figure 5.1. Among others, we have the Destination Sequence Number field, which is the latest sequence number received in the past by the originator for any route towards the destination. We also have an Originator Sequence Number, which is the sequence number to be used in the route entry pointing towards the originator of the route request.

Moreover, each RREQ packet is characterized by a unique ID, which is incremented every time a new RREQ packet for a given destination is generated. The ID field is used by intermediate nodes to discard duplicate RREQ packets.

When a node receives a RREQ, it first creates, or updates a route to the previous hop without a valid sequence number, then checks to determine whether it has received a RREQ with the same Originator IP Address and RREQ ID. If such a RREQ has been received, the node silently discards the newly received RREQ.

If the RREQ packet is not discarded, the node first increments the hop count value in the RREQ by one, to account for the new hop. Then the node searches for a reverse route to the Originator IP Address. If that is the case, the route is created, or updated using the Originator Sequence Number from the RREQ message. This reverse route will be needed if the node receives a RREP back to the node that

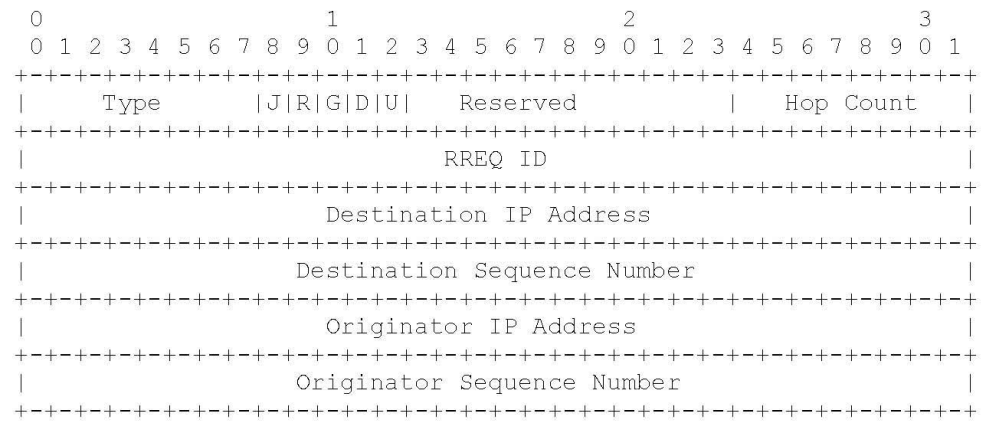


Figure 5.1: AODV RREQ type message

originated the RREQ. When creating the reverse route table entry, the next-hop is the node from which the RREQ was received (which is not necessarily the originator of the request) and the hop-count is set to that included in the received RREQ packet, increased by one.

If a node does not generate a RREP, it broadcasts the RREQ with the new Hop Count field. Then, the Destination Sequence number for the requested destination is set to the maximum of the corresponding value received in the RREQ message, and the destination Sequence value currently maintained by the node for the requested destination.

When the RREQ packet reaches the destination or a node inside the network with a valid route entry for the destination, the route is made available via a unicast RREP packet, which is sent back to the source of the RREQ packet.

When the destination is found (analogously when an intermediate node has the route to the destination), then, a RREP packet is created and is sent to the next-hop towards the originator. Moreover, the hop-count value is incremented at each node, so that, in the end, the originator node can update its routing table entry properly. When receiving the RREP packet, the node searches its routing table for the originator node (luckily it will have a reverse route) in order to know how to forward the RREP message.

Finally, nodes monitor the link status of next hops in active routes. When a

link break in an active route is detected, a RERR message is used to notify other nodes that the loss of that link has occurred. The RERR message indicates those destinations which are no longer reachable by way of the broken link. In order to enable this reporting mechanism, each node keeps a ”precursor list”, containing the IP address for each of its neighbours that are likely to use it as a next hop towards each destination. The information in the precursor lists is most easily acquired during the processing for generation of a RREP message.

In fact, the intermediate node receiving the RREP message updates the forward route entry by placing the last hop node (from which it received the RREQ, as indicated by the source IP address field in the IP header) into the precursor list for the forward route entry. Moreover, it also updates its route table entry for the node originating the RREQ by placing the next hop towards the destination in the precursor list for the reverse route entry.

5.2.2 AODV-MLW

AODV routing protocol finds minimum hop paths to each destination and does not allow to use arbitrary link weights. We developed an extension for AODV that enables to assign different pair of weight values to each bidirectional network link. The main concern was to leave unchanged the format of the messages. Both the RREQ and the RREP messages provide an 8 bit field that is labeled as Reserved. It is set to 0 on transmission and ignored on reception. We then chose to modify this byte in order to carry the weight information we need. The modified AODV RREP message is shown in Figure 5.2.

As mentioned earlier, not only RREP message information is used to update routing tables, but for reverse routes, RREQ packets are used. Since the weights are unidirectional, we need to take into account for direct and reverse weight when adding or updating the entries in the routing tables. In order to do so, in the RREQ packet an additional 8 bit field is needed (we would take the 8 bit flag field as well) for reverse weight values. With these changes, the RREQ packet will carry the information needed for reverse route entries (for intermediate nodes), while the RREP will deliver the final weight information and path to the originator of

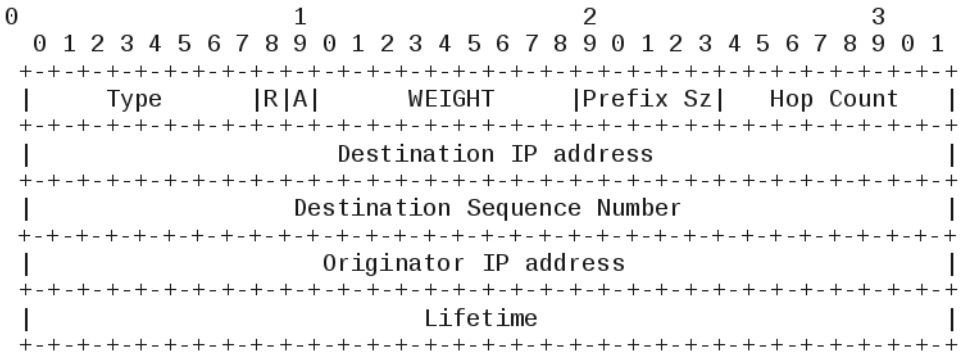


Figure 5.2: AODV-MLW RREP type message

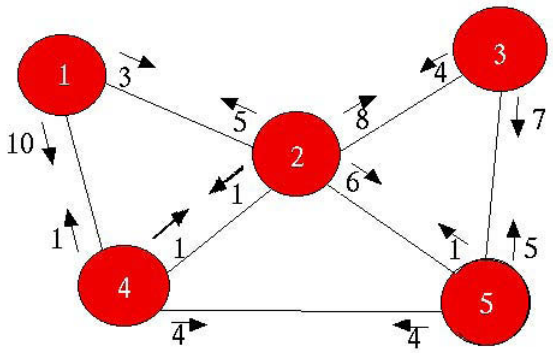


Figure 5.3: Example topology for the modified AODV message exchange

the RREQ packet.

When the source of the packets tries to send a message, a RREQ message is generated and is broadcasted through the network. It is assumed that each node has the weights associated to the links towards and from the neighbours (direct and reverse weights). A RREQ type packet gets processed by the receiving node if the RREQ packet received is new, according to the ID field, and the weight fields in the RREQ type packet are the smallest received for that destination.

The following example illustrates how the AODV-MLW protocol works.

Figure 5.3 shows a simple topology, with unidirectional weights associated to the links. Node 1 is the source, trying to send packets to Node 3, the destination. Node 1 needs to reach Node 3 and generates a RREQ packet, with both direct and reverse weight field set to 0. Direct and reverse weight values are updated hop-by-

hop. Hence, when a node receives a RREQ packet from a neighbour, by checking the direct and reverse weight fields, it knows the weight of the route from the RREQ originator node to itself so far (direct weight field), and that it can reach the RREQ originator with a route of known reverse weight value. The direct or reverse directions are to be intended from the RREQ originator node perspective.

The first nodes to receive the RREQ message are Node 2 and Node 4. When Node 2 receives the RREQ packet request from Node 1, it adds the direct and reverse weight information, regarding the connection 1-2, to the packet fields. Node 2, then, adds the reverse route entry for Node 1, setting the weight entry value to 5, and checks if it is the destination or if it knows the route to the destination Node 3.

Moreover Node 2 adds a new routing table entry that will be checked for the eventual RREP message to be forwarded to the originator of the RREQ: the neighbour from which the RREQ with the lowest direct weight is received, is cached and the weight of the route entry is the direct weight carried by the RREQ packet so far.

While the reverse route is used for all the ordinary packets, the just added routing entry is used for RREP message forwarding only. The RREP message forwarding mechanism will be explained later. If Node 2 is not the destination or it does not have the route to Node 3, the RREQ packet is broadcasted. When the RREQ packet forwarded by Node 4 arrives to Node 2, by checking the reverse weight field in the RREQ packet, Node 2 updates its reverse route. In Figure 5.3 it can be seen that Node 2 now can reach Node 1 through Node 4 and the reverse route entry of Node 2 to Node 1 must be updated to 2.

When the RREQ packet arrives at Node 3, a reverse route entry is cached and a RREP packet is created, with Node 1 IP as the destination IP field. The weight value of the RREP message is set to 0 and is updated each time a new hop is crossed. The weight field value carried by the RREP message through the network is the direct weight value of the route so far, from Node 1 to Node 3, and is updated hop-by-hop.

For example, if RREP message is forwarded from Node 3 to Node 2, the

weight field (after adding the weight value of last hop) would be 8. It is vital for the RREP message to follow the direct route (backwards) and not the reverse route. In fact, since weights are unidirectional, direct and reverse routing paths between two nodes are different. Here, Node 2 knows that a RREP packet to Node 1 must be sent through the direct link 1-2 and not through Node 4, despite the fact that the shortest path from Node 2 to Node 1 is through Node 4. Finally, the RREP packet is delivered to Node 1. When RREP message is received from Node 1, routing table entry to Node 3 is cached, with the weight information taken from the RREP packet (incremented with the last hop weight), that is a value of 11, and with Node 2 as the next hop.

Finally, we note that the use of an 8-bit field in the RREQ and RREP messages constrains the weight of a path to be less than 255. Thus, link weights should be properly scaled such that it cannot occur the case of a path weight exceeding 255.

5.3 Computing link weights for AODV-MLW

The goal of this Section is to find a proper set of link weights that allow our modified version of AODV to utilize every network link in proportion to the corresponding pre-computed flow rate. We assume that, given a set of traffic demands, a solution to the joint channel assignment and routing problem is provided, which supplies such flow rate value for each link. In traditional destination-based routing protocols, links are assigned a weight and some information on the network topology is disseminated among routers. Each router then builds its own routing table and forwards a packet along the minimum weight path to the destination of the packet. Thus, the set of link weights heavily affects how the traffic is distributed across the network.

Given the traffic demands, it is not easy to determine the set of link weights that lead to a desired flow distribution, because the forwarding decisions taken by each router depend on the weight of all the network links. Besides MPLS [57], routing protocols do not allow to freely distribute flows between source and destination. Moreover, when weight-based routing protocols are used, standard weights

configurations (often unitary weights leading to minimum hop-count paths), cannot realize the nominal flows inside the network. For such protocols, the problem of flows distribution becomes a weights choice problem.

For routing algorithms like AODV, if a traffic matrix and the physical network configuration (for example link capacities) are known, there is a direct correlation between the link weights choice and the flow distribution inside the network. In our case, the routing protocol objective would be to use the link weight distribution in order to obtain optimal flows inside the network domain. Unfortunately, it has been shown [58] that such traffic-bounded weight configuration problems are NP-hard. Hence, we resort to heuristics to find an approximate solution in polynomial time.

Given the set of traffic demands between source-destination node pairs and the set of link weights $\vec{w} = (w_1, w_2 \cdots w_{|E|})$, where $|E|$ is the total number of network links, the corresponding flow distribution can be computed by finding the minimum weight paths between all the source-destination node pairs. Therefore, given the set of link weights $\vec{w} = (w_1, w_2 \cdots w_{|E|})$ and denoted by $\vec{f} = (f_1, f_2 \cdots f_{|E|})$ the resulting vector of link flow rates, we consider an operator T such that:

$$\vec{f} = T(\vec{w}) \tag{5.1}$$

If we denote by $\vec{f'} = (f'_1, f'_2, \cdots, f'_{|E|})$ the set of pre-computed flow rates, the determination of a set of link weights leading to a link utilization close to the pre-computed flow rate on every link can be formalized as the minimization of the objective function $\mathcal{F}$, reported in equation 5.2:

$$\mathcal{F} = \min_{\vec{w}} \Psi(\vec{w}) = \min_{\vec{w}} \sum_{n=1}^{|E|} \left([T(\vec{w})]_n - f'_n \right)^2 \tag{5.2}$$

where $\Psi(\vec{w})$ is the *fitness* function.

This is a non-linear problem and the T-operator is not known in an explicit form, so no linear programming nor derivative minimization methods can be applied. In order to find the best weights, a local search technique like Tabu Search is used. Given an arbitrary initial solution $\vec{w}_0$ and a space of feasible solutions $\mathcal{W}$,

a neighbourhood $N(\vec{w}_0) \subseteq \mathcal{W}$ is defined and a new solution $\vec{w} \in N(\vec{w}_0)$ is sought such that $\mathcal{F}(\vec{w}) < \mathcal{F}(\vec{w}_0) = \mathcal{F}_0$. Being the neighbourhood at each iterative step the real search space for the algorithm, it is crucial to design it wisely: it cannot be too small or otherwise the found solution is likely to be suboptimal. On the other side, as the size increases, computational time becomes critical. Moreover, since the algorithm starts with an arbitrary solution, it is important that the quality of the final solution does not strongly depend on the initial choice. At every iteration, the neighbourhood is chosen as follows.

Be $\vec{w} = (w_1, w_2, \cdots, w_{|E|})$ an initial possible solution, and K the set of admissible weights values (i.e. $w_i \in K$). Then, a vector, say it $\vec{w}'$, of $N(\vec{w})$ is obtained by changing just a single weight, say it i, in $\vec{w}$ such that $w'_i = k$ with $k \in K$ and $w'_j = w_j$ for $j \neq i$. Accordingly, the cardinality of $N(\vec{w})$ is $\mathcal{N} = (|K| * |E|) - 1$.

To explore the whole solution space while avoiding cycling, a tabu list is maintained which includes all the previously visited solutions. In every iteration, a solution from the neighbourhood is a possible candidate only if it has a better fitness and it is not in the tabu list. The search for a solution in the neighbourhood having a better fitness may be computationally rather expensive. Indeed, starting from the previous solution, a single weight is chosen and incremented by a unit. However, such a small increase may not alter any shortest path between the source-destination node pairs and hence the fitness of the candidate solution may be the same as the previous solutions.

In order to find a candidate solution with a different fitness, a number of iterations of such a process may be required. The neighbourhood construction strategy and exploration introduced may then be time consuming, so that the time required by the search process to find the optimal solution could be too high. On the other side, it is necessary to perform the weight evaluation over short periods of time, especially when this computation has to be executed somewhere inside a domain network, wherein only limited computing power is available. A less resource consuming neighbourhood construction may help reducing the research speed of the Tabu Search algorithm. Moreover, optimal solution may be far from the starting solution, so that an efficient exploration strategy has to be introduced. This

could be obtained by dynamically choosing the neighbourhood extension, that is, by considering the neighbourhood size as a function of the optimization degree of the algorithm at that particular iteration. The new neighbourhood is then built as follows.

To reduce the number of iterations as much as possible, then, we use an adaptive approach. After a fixed number I of iterations that do not produce a change in the fitness, the selected weight in the previous solutions is increased by a value l, which initially equals 1 and then is doubled every time I iterations do not alter the fitness value.

However, it may happen that many consecutive different solutions have the same lowest fitness values, and being chosen, they lead to a stall in the search. The reason is that, since the differences between the elements of the neighbourhood reside on a single vector component and this difference may be small, routing decisions (and thus the T function) may not be affected by this small weight variations, leading to the same fitness values for many consecutive iterations. To avoid having the same lowest fitness value for several steps, that is being stuck near a local optimal solution, the value $\mathcal{F}(\vec{w})$ is also stored, so that a feasible solution $\vec{w}'$ is tabu even if it has the same fitness value of a previously inspected solution $\vec{w}$. In this way we add some diversification to the search process, in the sense that the solutions closer to the current one are avoided, and new regions of W are explored.

The performance of the Tabu Search algorithm, with the different choices of the neighbourhood, become comparable with that of the genetic algorithm (properly tuned), as regards the quality of the final solution. Obviously, the choice of the genetic algorithm is not feasible, though, due to the high running times.

5.4 Performance Evaluation

In this Section we present the results of the simulations carried out to evaluate the performance of the AODV-MLW routing protocol.

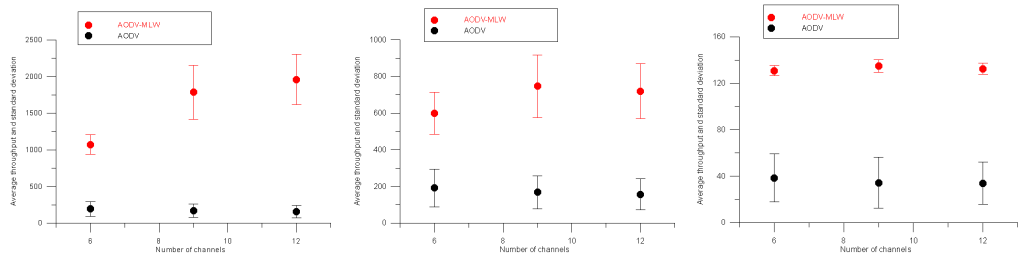
Throughput with varying traffic demands

In this Section, a comparison between the AODV-MLW algorithm and AODV is presented in terms of throughput and robustness to changes in the traffic demands. Simulations, carried out with the *ns-2* network simulator, are set up as follows. We consider wireless mesh networks of 25 nodes, each of which is equipped with two or three radio interfaces. Simulations are performed in case 6, 9 or 12 channels are available. Given a set of traffic demands, we use the joint channel assignment and routing algorithm proposed in Section 3.3 to find a channel for every radio and the set of pre-computed flow rates. The set of traffic demands and the set of pre-computed flow rates are fed to the Tabu Search heuristic proposed in Section 5.3 to compute the link weights used by the proposed AODV-MLW.

Moreover, when dealing with flow optimization, it is very important to estimate the traffic matrix, since this information is vital for the evaluation of the nominal flows over the wireless links. However, since the traffic matrix is time variable, especially for wireless networks, traffic estimation is very difficult: the nominal flows to be realized are obtained for a given, static, traffic matrix, while network conditions vary with time. Even if traffic matrix would be known every time instant, evaluation of optimal flows would be time consuming and not feasible for very short time periods. Hence, it is important to have a routing protocol with the ability to keep the throughput and optimization degree of the network at an acceptable level, with variable network conditions.

Throughput performance for AODV-MLW and AODV, is considered when different traffic variations take place inside the network.

The throughput achieved by AODV-MLW and AODV is shown in Figure 5.4a through 5.4c for a different number of available channels. In particular, Figure 5.4a shows the case where the source nodes generate exactly the traffic profile used to compute the channel assignment, the set of pre-computed flow rates and the AODV-MLW link weights. Figures 5.4b and 5.4c instead show the case where the source nodes generate a traffic pattern that more or less slightly diverges from the given set of traffic demands. The goal is to show that AODV-MLW is more robust than AODV to variation in the predicted traffic demands.



(a) No traffic demand variation (b) Nearly 65% variation in traffic demands (c) More than 90% variation in traffic demands

Figure 5.4: Comparison of AODV-MLW and AODV in terms of throughput

We computed the average throughput over the simulation time for ten different topologies and show in the Figures an interval centered at the average throughput over the ten topologies and having a width equal to the standard deviation. The AODV-MLW forwarding paradigm is shown to achieve a higher throughput than AODV for all the traffic demand variations, thus showing a better capability to adapt to traffic variations. In particular, even when the actual traffic profile diverges from the given traffic demands, the AODV-MLW outperforms AODV.

We also measure the Mean Square Error (MSE) between the average link utilizations and the pre-computed link flow rates to evaluate the ability of AODV-MLW to achieve given utilizations on network links.

Table 5.1 reports the average MSE values measured for AODV-MLW for different variations in the traffic demands and for a different number of available channels. The results are again averaged over ten different topologies. In particular, Table 5.1 shows that AODV-MLW can obtain links utilization that are close to the nominal flows, even when the traffic profile may be different from the original one.

5.5 Chapter Conclusions

In conclusion, in this Chapter we developed a modified version of AODV that makes use of arbitrary link weights. This proved to be necessary, in order to cope with the routing problem.

Traffic Loads	Average MSE for **AODV-MLW**	Average MSE for **AODV**
no traffic demand variation	0.021	2.712
about 60% variation	0.022	2.820
about 90% variation	0.022	3.016

Table 5.1: MSE values for both AODV-MLW and L2.5 algorithms

Since the traffic flows are returned as the solution of the optimization problem, the main purpose is to obtain the actual rates on the links as close as possible to the nominal flows. Common optimization criteria are throughput maximization, to verify the feasibility of a traffic matrix, to maximize the minimum end-to-end rate, or to impose fairness to the traffic rates. AODV with Modified Link Weights can then be used for different purposes. Then, we proposed a technique to compute link weights such that a destination-based routing protocol can achieve a given link utilization on its links.

We are given the set of pre-computed flow rates, together with a set of traffic demands, and returned the set of link weights that can be used by AODV-MLW. Since the proper link weights computation is a NP-hard problem, we solved the problem numerically, designing an heuristic technique based on Tabu Search.

Finally, we showed through *ns-2* simulations that a traffic flows aware routing protocol is better suited to the WMN framework than a simple routing protocol which routes packet on the minimum hop-count path. AODV-MLW is stronger in terms of throughput and robustness to variation in traffic changes.

Part II

Application layer traffic optimization in Wireless Mesh Networks

Chapter 6

Overlay/Underlay issues in Wireless Mesh Networks

Recent analysis of the network traffic shows that an ever growing portion of it is generated by user communities that share their own resources in a Peer-to-Peer fashion (see Figure 6.1). Internet Service Providers' attitude towards Peer-to-Peer applications, so far, has been of considering them more as potential threats rather than as an opportunity and a customer demand to meet. This is mostly due to the fact that Peer-to-Peer applications still represent a significant traffic engineering challenge for ISPs, since they tend to build their own overlays in a way that is largely independent of the Internet routing and topology.

Moreover, Wireless Internet Service Provider (WISPs) are making large use of the Wireless Mesh Network model to easily and quickly offer their services in urban and rural areas with reduced investments. In the WMN context, the problem of controlling P2P traffic is even more stringent, due to the limitedness of resources. In this Chapter, we discuss these issues and invoke the adoption of an architectural model that allows the exchange of information among Peer-to-Peer applications and mesh routers, according to a cross-layer approach, where not only lower layers (data link) but also upper layers (application) influence the network layer.

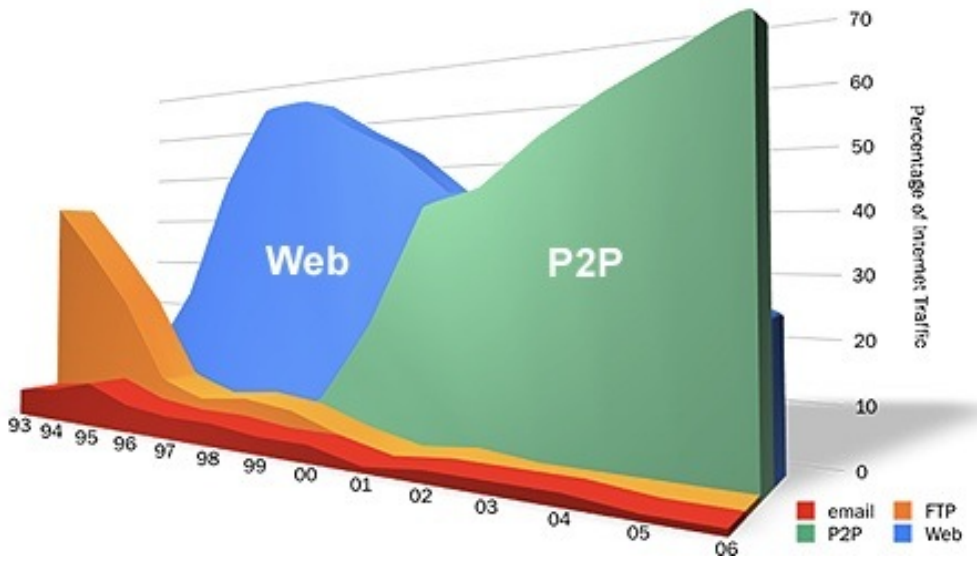


Figure 6.1: More than 50% of Internet traffic is generated by P2P applications

6.1 Overview

Overlay networks are virtual communications infrastructures implemented ”on top of” an underlying physical network such as the Internet. An example of overlay network is shown in Figure 6.2. Routing of packets in an overlay is usually performed at the application level. Overlay networks are usually created either to force some special routing in the network (e.g. multicast routing, or QoS routing) or to organize application-level entities in order to efficiently provide some form of service to large scale communities (e.g. Peer-to-Peer networks for file sharing or video streaming).

Systems based on overlay of peers are inherently distributed and, if properly designed, more robust and scalable, due to their decentralized nature and to the absence of single points of failure. Nonetheless, overlay routing decisions collide with those made by underlay routing, i.e. ISP routing decisions. While overlay routing objectives are ultimately applications objectives, i.e. compliance with a given QoS specification or latency optimization, underlay routing decisions are guided by objectives such as load balancing, minimization of cross-ISP traffic and minimization of links utilization. As a consequence of such a dichotomy, several

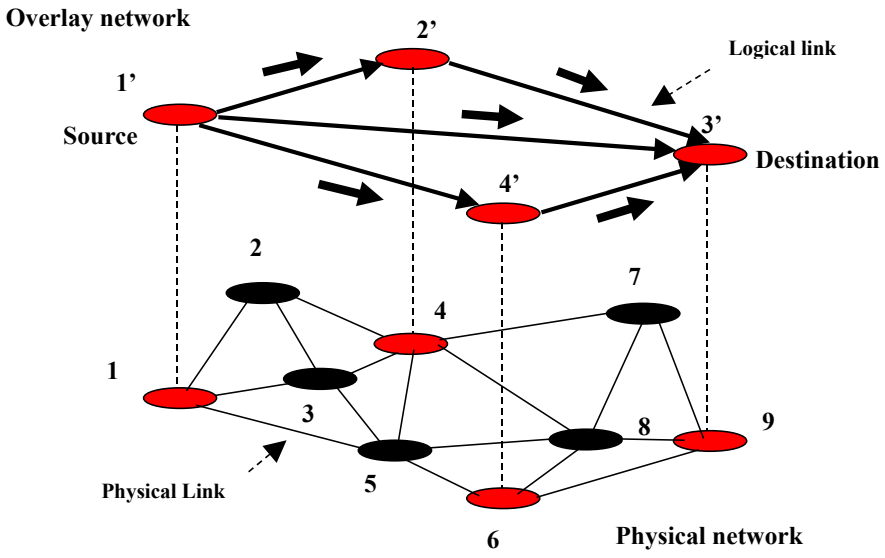


Figure 6.2: An example of overlay network

inefficiencies may result. For example, a problem that can arise is traffic routing oscillations: if the overlay routing moves a great amount of traffic from one path to the other, ISP as a result may notice a bandwidth decrease and change the weight link balance in order to redistribute traffic. This, in turn, could induce a subsequent overlay routing change that can trigger repeatedly underlay routing change, bringing the overall system to an endless traffic imbalance: this is an example of negative consequences of no information exchange between the two routing layers. The interactions between the two layers are shown in Figure 6.3. Moreover, due to the fact that overlay routing does not know physical nodes positions, it is not uncommon that adjacent nodes of an overlay network are in different ASs. Such a topology arrangement leads to traffic crossing network boundaries multiple times, thus overloading links which are frequently subject to congestion, while an equivalent overlay topology with nodes located inside the same AS could have had same performance. From what we described above, it emerges that overlay routing may benefit from some form of underlay information recovery, or in general from cross-layer information exchange. When creating an overlay network, the choice of the nodes (i.e. the network topology) can be done

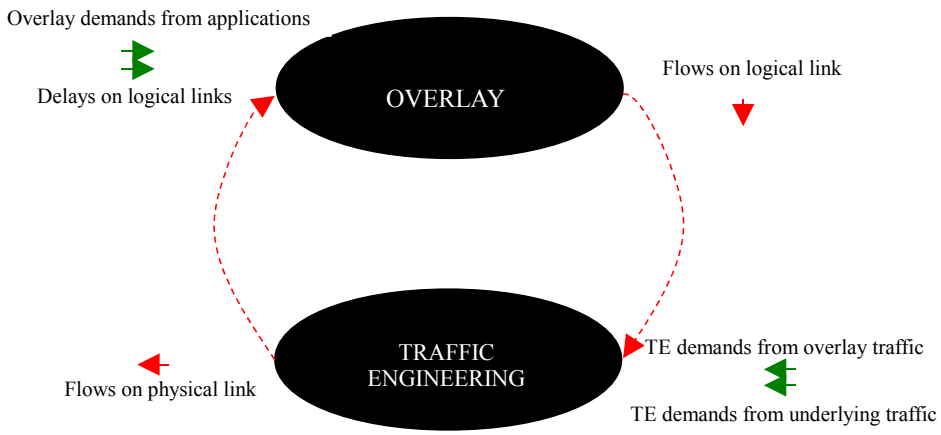


Figure 6.3: Interaction between overlay optimizer and TE's optimizer

by taking advantage of information from the underlay network. The problem of creating network informed overlays has been studied both for general overlays and for specific Peer-to-Peer applications. Since Peer-to-Peer traffic is nowadays predominant in the Internet, this problem is of great interest for Internet Service Providers.

Cross-layer optimization techniques are also widely adopted in the context of Wireless Mesh Networks, where they are often applied to make optimal routing decisions in combination with the spectrum resource allocation. In such a context, negative interactions among overlay and underlay routing decisions may jeopardize the whole network ability to keep operating. A couple of nodes can communicate only if a common communication channel exists, in WMN. In Section 2.3 we presented some of the main issues that have to be addressed in WMNs. In particular, in Chapter 3, we focused on the channel assignment issue. Here, we quickly recall the fundamental points. When nodes share a common channel, the total link capacity is shared as well: the number of nodes in the same interference range, using the same channel implicitly limits the link bandwidth. It is clear, then, that channel assignment implicitly limits the link bandwidth. Moreover, the channel assignment main objective is to realize link bandwidths so that is possible to route traffic inside the network. The more accurate the description of the traffic incoming into the network, the more efficient the bandwidth usage benefits. To

this purpose, one important parameter is the traffic demand of a given P2P application. Providing such information to the underlay network could result in better resource utilization in the underlay. In the case of a WMN, overlay applications could give information to mesh routers, such as peer nodes sharing a certain software content and the bandwidth requirements for the file transfer. The Wireless Mesh Network, in turn, could use this information to readapt channel assignment in order to satisfy new traffic requests.

The overlay building procedure often employs application-level measurements to most suitably organize the overlay topology. Such an approach has been proved to be suboptimal and inefficient. A possible alternative would be to build some kind of ”oracle” service, realized and maintained by the underlay network, i.e. the ISP, that could instruct users about the best way to organize their overlay. The adoption of such an approach is seen as mutually beneficial for both users and Internet Service Providers. In the rest of this Chapter we review some of most recent approaches for underlay/overlay cooperation that have been proposed in the literature. Subsequently we describe the benefits of supporting cross-layer interactions in a WMN and we illustrate our proposal of adapting an oracle service in the specific context of Wireless Mesh Networks.

6.2 Peer-to-Peer Traffic Optimization

The literature in the field of wireless mesh networks presented in Section 3.2, mostly ignores the characteristics of the network traffic and does not attempt to exploit any possible knowledge of its nature. The interaction between overlay and underlay routing has been mainly studied in the context of wired networks.

The work in [59], for instance, acknowledges that designing overlay services to independently probe the Internet, with the goal of making informed application-specific routing decisions, is an untenable strategy. For this reason, the authors propose a shared ”routing underlay”, which is an architectural element that sits between overlay networks and the underlying Internet. Overlay networks query the routing underlay when making application-specific routing decisions. The un-

derlay, in turn, extracts and aggregates topology information from the underlying Internet. The underlay routing is organized in a two-layer architecture. The lower layer, denoted as Topology Probing Kernel, provides underlay primitives such as a graph of the known network connectivity at a specified resolution (e.g., Autonomous System (AS) level, router level, physical link level), the actual path a packet follows from one point to another, the characteristics of specific paths (e.g., in terms of AS hops, router hops, measured latency). The upper layer, denoted as Library of Routing Services, is designed to answer higher-order questions about the overlay itself using the primitives exported by the lower layer. For instance, the upper layer may provide services such as finding the nearest neighbours to a node or finding disjoint paths between two nodes. Several recent works have analyzed the problem of improving network performance through Overlay/Native networks collaboration.

In [60] the authors study the routing performance of a particular layer when very few information is known from the other layer, i.e., non-cooperative interaction occurs. In the paper one layer is considered as Leader and it tries to predict and react to the other layer's actions (the Follower), trying to avoid performance deterioration. The leader layer tries to optimize its own performance, trying to preserve the optimization objective of the other layer (friendly approach) or with no regard for the other layer performances (hostile approach). As regards the overlay layer, the strategies consist in paying attention to not trigger underlay optimization decisions, trying to keep the load balance of the underlay network intact (that is, the friendly approach), or, for the hostile approach, to completely defeat the underlay optimization objectives.

For native layer, the friendly strategy consists in modifying the optimization step, with the extra consideration of keeping the native route of the same length as in the last optimization step: in this way, the overlay layer will have no need in changing routing paths, thus maintaining good load balance in the network.

The hostile strategy, contrarily, consists in strongly limiting the overlay layer free path choice: in this way, underlay optimization do not need to take into account possible traffic matrix variations due to overlay traffic switching. To manip-

ulate overlay routing, link latency is modified in such a way to push the overlay nodes to keep the same routing table. At the same time, the stability of the network is studied in terms of route flaps numbers, which is the result of the conflicts between layers and it is determined by the number of multi-hop overlay paths: it is shown that for both strategies stability is reached in few steps. The optimization objectives are end-to-end delay minimization for overlay users and network cost minimization for underlay network. In the paper, no attention is dedicated to information exchange between the two layers.

Despite the great performance obtained by the hostile strategies, the damage brought to the other layer is unsustainable; on the other hand, the friendly performance gives stability to the network, at cost of reduced performance. In the end, performance improvement can be obtained by just one layer, and in general the performance is strictly dependent of the other layers strategy, with poor information knowledge. This convinced us that, through cross-layer collaboration it is possible to achieve a more balanced use of network resources, leading to better performance for both overlay and native routing.

In [61], the interaction between overlay and underlay routing is studied for a single AS (Autonomous System). Again, overlay routing objective is to minimize end-to-end delay, while underlay routing tries to minimize a given cost function. Game-theoretic models are used to study the interaction between the layers, where each layer, for each step, tries to optimize the given objective. Nash Equilibrium is found for simple-case topologies and it is shown that underlay performance will never be improved in the Nash routing game when overlay routing is present. Moreover, routes oscillations are found for both overlay and underlay, due to differences in optimization objectives. It is shown that the selfish behavior of the overlay layer can rise the value of the global cost for the underlay layer and, moreover, the optimization steps taken by the overlay routing in each round, i.e. when it plays a best-reply Nash routing game with underlay, can worsen even the performances of the overlay user instead of improving them. It is then revealed that when no explicit collaboration is present, overlay optimization criteria are not always the ones that really minimize the objective function, i.e. best-reply strat-

egy is not the best strategy in this case. Besides, when overlay traffic percentages are not small, underlay's cost is shown to be dramatically increased by the interaction with overlay routing: a certain coordination between the two layers in regards of the optimization moments would lessen the misalignment degree. Even the authors of [61] feel the importance of knowledge exchange and mutual awareness in cross-layer routing optimization, and an example would be information given to the overlay about the optimization algorithm of the underlay routing. In this way the upper layer would be able to predict underlay decisions, thus optimizing the overlay routing decisions. Still, information estimation and cross-layer strategies are strongly called.

In [62] P2P traffic is directly addressed. Once again P2P user perceived performance is the main issue and it is solved thanks to the overlay network: differences in overlay and underlay routing are analyzed, like topology creation, routing criteria and different dynamics. In particular, the overlay topology creation problem is addressed. Each node should pick the neighbour with smallest delay, possibly with the highest throughput, and in the same AS: in the paper the cross-layer information exchange is welcome and an "oracle" service is studied. The "oracle" is a service supplied by the ISP (i.e. the underlay network manager) to the overlay P2P users: very different information is provided, like link delay, bandwidth estimations, etc. The "oracle" service takes as input a list of P2P nodes sharing a known content (even from different ASs) and gives back the nodes list ranked according to different performance metrics. In this case, the P2P application won't have to perform such measurements by itself (and these measurements would be available to all P2P applications) and thus, measurements overhead traffic is avoided and the topology creation process is optimized. Moreover, the "oracle" gives ISPs a way to control overlay topology creation decisions, and, ultimately, overlay routing. An example of guided overlay topology creation is showed in Figure 6.4, extracted from [62]. The paper shows how explicit cross-layer information exchange (in this case information flow goes from underlay to overlay) can really improve overlay operations (topology creation in this case) and at the same time improve underlay network usage (AS hops are minimized and congested link

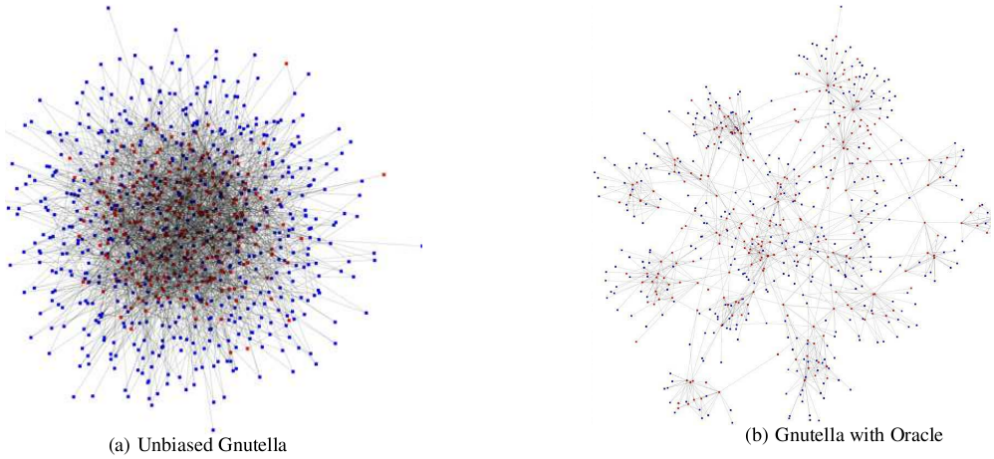


Figure 6.4: Topologies obtained without and with the Oracle

could be unloaded). Despite the difficulties in oracle management, the results suggest that cross-layer information exchange is a valid starting point in solving the overlay-underlay frictions.

In [63] the problem of the interaction between routing layers is addressed when route failures take place at the physical layer. For both underlay and overlay layers a dynamic routing protocol (Open Shortest Path First) is employed and average route flaps number is considered while evaluating performances. In particular, two scenarios are considered: one with no interaction between layers and sub-optimal results regarding route flaps and network cost (due to unawareness between layers and overlap of functionalities between levels); another scenario, where awareness of underlay routing at the overlay is considered, and better results are found in terms of route flaps and cost function optimization. Once again, independent operations of the two layers return insufficient performances, high number of route flaps and general system instability. Moreover, two approaches are followed to improve interaction between layers: overlay awareness of the underlay routing through information access at the native layer, and underlay routing awareness of the overlay layer and consequent parameters tuning in order to favor overlay routing performances. In particular, better results are obtained with the

second approach, where the native layer tries to suit the overlay traffic, helping the overlay layer to reach the best performances according to different metrics.

In [64] collaboration between layers is invoked, especially in terms of control traffic, to obtain optimality in terms of underlay network costs, while protecting P2P users performance. The introduced paradigm, P4P architecture, consists in explicit information exchange between providers and application users, i.e. a portal. The main feature of this architecture is the p-distance information interface, which collects and gives information about physical network's status and indications about possible application traffic choices. The authors call these portals iTrackers, and each network is responsible for each own iTracker, since each portal collects information for one particular domain. Each provider can set the p-distance value based on link cost or link congestion, while applications can combine the provider information with delay or bandwidth requirements, through application level optimization algorithms. Moreover, thanks to the iTracker, it is possible to divide management responsibilities between provider and application. In particular, the MLU minimization problem is decomposed in many different and independent problems, each one linked to a given application session: in this way a distributed solution to the optimization problem is possible, thanks to the information exchange between layers. Once again it is shown that through cross-layer information exchange it is possible to reduce completion time and bottleneck traffic, both in terms of intra-domain and inter-domain transmissions.

In [65], the authors evaluate the gain that can be obtained from using network coding when running file sharing applications in wireless mesh networks, and conclude that some gain can actually be achieved, although not as much as in a wired network. In [66], a two-tier architecture is introduced for file sharing in wireless mesh networks: the lower tier is composed of mobile mesh clients, which provide the content (files/resources to share) to the P2P overlay; the upper tier is composed of stationary mesh routers, and implements a distributed hash table for locating resources within the network.

Starting from the well-known Chord protocol for resource sharing in wired networks [67], some of the authors of [66] propose in [68] an adaptation (called

MeshChord) that accounts for peculiar features of wireless mesh networks. The work in [68] complements that in [66]. While the emphasis in [66] is on the procedure for dealing with client mobility at the lower tier of the architecture, in [68] the authors are concerned with the efficiency of Distributed Hash Table (DHT) implementation in the upper tier of the architecture. In particular, the exploited peculiarities of the Wireless Mesh Networks are the availability of a wireless infrastructure and the 1-hop broadcast transmission, while at the same time particular care is directed towards mobility and signalling overhead. Like in [67], MeshChord maps each node ID and each shared resource ID into a hash table entry key.

In [68], though, the authors try to exploit node location inside the network, when assigning IDs to nodes: observing packets exchange due to the Chord protocol, the traffic is seen to be more intense when peers are close-by in terms of ID, and then nodes that are physically close-by are assigned to close-by IDs. This approach, though, requires nodes positions. Moreover, to speed up the search operations, each node captures a lookup packet even if it is not the destination of the lookup packet. In this context, the lookup packet is used to find the IP address of a node with a known ID, and it is sent to. The concept is that the node capturing the packet could be able to answer to that request, and in that case it could answer to the requesting node directly, thus speeding up the requesting phase. The paper, finally, shows how a protocol designed for wired-connected networks can be adapted to the features of the wireless networks, thanks to cross-layer techniques, and that it needs to be adapted in order to improve performances.

The problem of providing ”standard” ways of interoperation between overlay and underlay networks has been recently considered in the IETF, where a Working Group on ”Application Layer Traffic Optimization” (ALTO) has been recently chartered [1]. The objective of this working group is ”to design and specify a service that will provide applications with information to perform better-than-random initial peer selection”.

6.3 P2P in Wireless Mesh Networks

In the previous Section, we have discussed the issues stemming from application layer routing decisions being unaware of the underlying network. In particular, the work appeared in the literature has shown the negative consequences of selfish routing strategies at both the overlay and the underlay level. The aim of this Section is to revisit such issues in the context of wireless mesh networks. WMNs are expected to provide Internet connectivity to end-users and, consequently, to carry a large amount of P2P traffic. Such volume of traffic, in the absence of cooperation between overlay and underlay layers, may have even more negative consequences in WMNs than in wired networks.

Firstly, P2P applications generate a large amount of overhead traffic for maintenance operations. For instance, each peer may autonomously probe the underlying network in the attempt to guess some network performance. In wireless multi-hop networks, where the available bandwidth is cut down by interference, overhead traffic has much more harmful consequences than in wired networks. Hence, one of our goals is to define an architecture where peers are aided by the ISP in the selection of a group of peers to be contacted. Thus, peers no longer need to probe the network by themselves.

Secondly, the wireless nature of links in multi-radio WMNs requires the tuning of parameters such as frequency channel, transmission rate and transmission power. The proper parameters configuration can determine the existence of a link and the bandwidth available on the link itself and on the links in the neighbourhood. As shown in Chapter 3, the configuration of such parameters is strictly inter-dependent with the traffic expected to flow on every wireless link. Since the joint channel and routing problem is NP-complete, the typical approach is to find an approximate solution by solving the two problems separately. The flows optimization problem is solved first, finding the amount of traffic flow that crosses a network link so that a given objective function is optimized. Then the channel assignment problem turns out to be the problem to assign the channel and the transmission parameters so that the nominal flows are feasible.

It is thus clear that a given channel assignment is tailored on a certain dis-

tribution of flows inside the network. A different distribution of flows instead may result in poor network performance. Since P2P applications can move large amount of traffic in the attempt to improve the performance they experience, it is crucial that their decisions match the underlying network configuration. This last point motivates us to study the overlay/underlay routing interaction problem from a somewhat different perspective than previous research. Indeed, the underlay routing has always been looked at as a sort of black box, giving a little chance, if any, to modify its behavior. Our goal, instead, is to design an architectural model for the exchange of information between the overlay network and the underlying wireless mesh network. We envision that some (or even all of the) mesh routers are equipped with some enhanced functionalities that allow them to interact with overlay applications.

In a typical scenario, overlay applications may provide a mesh router with a list of neighbours sharing the desired resource and an estimate of the bandwidth required by such a communication. The mesh router will then rank each peer based on how the corresponding traffic matches the actual network configuration. Alternatively, the mesh router may re-compute a new network configurations (channels, transmission powers and rates) if one exists that is better suited to accommodate the new and the previous traffic requests.

The architecture we envision largely adopts cross-layer principles, as parameters at lower layers heavily impacts the behavior at the application layer and vice versa. The need of mixing information from different layers to take proper decisions stems from the strict inter-dependence that exists in wireless mesh networks between the distribution of flows inside the network and the configuration of data link parameters such as the frequency channel, the transmission rate and the transmission power.

6.4 Chapter Conclusions

Starting from recent traffic analysis, P2P applications create more than half of current Internet traffic. Along with their success, recent works have shown the in-

efficiency and instability following independent interaction between P2P/overlay and underlay routing. Routing paths oscillations and inefficient overlay topology creation lead to a decrease in application performance and to increased management costs. When dealing with wireless mesh networks, even more problems arise. Resource shortage and complex wireless parameters configuration suggest the need for a cooperative cross-layer routing approach, with particular attention to the possibility to convey information about underlay network to the P2P applications. The cross-layer routing protocol would take routing decisions based on heavy information exchange between all the layers, from the MAC level up to the application level, in the attempt to optimize overlay and underlay routing paths and overlay topology creation, while improving both network and application performances.

The first step is to define and evaluate underlay metrics that could be useful in helping applications, indicating the most resourceful and high-performance nodes. Moreover, such a metric should be considered in the context of wireless mesh networks. In Chapter 8 we introduce a wireless mesh network path metric that evaluates end-to-end paths between users taking into account links utilization and intra path self-interference.

Then, a new architectural model is to be designed in order to exchange information between layers. For this purpose, applications should be designed in order to interact with the proposed architecture. The model we envisage is characterized by overlay nodes giving information about traffic information and possible P2P resource availability to the underlay network. In Chapter 7 we introduce the IETF Working Group solution, the ALTO Service, which approaches the cross-layer interaction issues in the context of P2P applications. The goal is to help applications by providing information about the underlying network, thus helping them to optimize the overlay network, while optimizing the management of underlying network resources.

In Chapters 9 and 10, then, we exploit the ALTO Service, placing it in the context of wireless mesh networks. In particular, we present the problems that arise in the integration of wireless and wired testbeds and, finally, we extend the

ALTO Service and implement it in a realistic wireless mesh network environment.

Chapter 7

The Application Layer Traffic Optimization problem

In Chapter 6 we presented an overview on the main issues taking origin from applications being unaware of the underlying network.

We remarked that the situation is problematic both from ISP and P2P applications perspective:

ISPs: they are unable to perform efficient traffic engineering operations due to the inability to control traffic and to influence the peer selection mechanism.

P2P Applications: they are unable to build an optimal topology, since they ignore the physical topology characteristics; moreover, they often probe underlying links to measure some performance metrics, obtaining unstable results.

One of the greatest inefficiencies that arise from such an uncoordinated interaction is that neighbour nodes in the overlay network are many ASs away. With such an arrangement, traffic flows cross several times ISP boundaries, thus overloading critical links often subject to congestion. Moreover, since ISP economic agreements are based on the volume of traffic that cross their boundaries, ISP management costs drastically increase as well. Finally, Peer-to-Peer clients generate overhead traffic by independently performing network metrics measurements.

Some form of cooperation between ISPs and P2P applications is then highly likable, since it will be beneficial for both. Among a given set of available peers,

P2P clients exchange data mostly with a subset of such overlay nodes. We refer to this subset of peer nodes as the neighbourhood of the P2P node. In particular we aim at guiding the creation of the overlay topology, by providing P2P clients with ISP preferences on the available neighbour peers.

In this Chapter, in particular, we refer to a Bittorrent tracker-based file sharing application, and in particular in Section 7.1 we give an overview of the Bittorrent protocol mechanisms. In Section 7.2, we introduce the solutions proposed by the ALTO IEEE Working Group, and show how a simple file sharing Peer-to-Peer application may exploit an ALTO Service [1].

7.1 Tracker-based Bittorrent Protocol

Bittorrent application has been introduced to allow large amount of popular resources to be rapidly distributed. Resources examples are software house new operative system releases, digital video or videogame distribution, that are usually distributed as large dimension ISO file dvd images. Given the nature of the shared content, it is not uncommon the creation of the so called flash crowds, i.e., in a few days a large amount of peer users require the same resource. Clearly, Client-Server solutions are not suitable for these situations, due to the large number of traffic requests in small time intervals.

Bittorrent idea is to spread the popular file exploiting the domino effect among all the interested peer nodes. The set of all the nodes sharing the same resource file is called swarm and each node running a Bittorrent client is called peer. Bittorrent protocol does not specify any kind of resource file research mechanism. This means that the research operation has to be out of band, i.e., it has to relay on external procedures. Before sharing, each Bittorrent resource file is decomposed in small chunks. Then, a torrent file is created.

The torrent file stores metadata about the file that is shared, which peers own which chunk, and the address of the tracker, the entity that coordinates the content distribution. Peers that want to join the swarm have to first obtain the torrent file, then connect to the tracker. The tracker will inform the peers from which

peer they have to download each chunk. Bittorrent clients contact different peers on different TCP connections, while the single chunk downloading is made on a single TCP connection to a single peer.

Bittorrent peers use a tit-for-tat like strategy to optimize download speed, called optimized unchoking. Each peer has a limited number of slot reserved to upload data to other peers. Upload bandwidth is allocated taking into account the download bandwidth, encouraging fair trading. When a peer is not uploading in return to our own uploading, the client will choke the connection with the uncooperative peer and will allocate more resources to moore cooperative peers.

Such a strict policy may result in suboptimal situations, though. For example, when new peers are unable to receive data because they do not have available chunks to trade, or when two peers do not exchange data because neither of them takes the initiative, despite having a good connection between them. To counter these possible side effects, Bittorrent uses a mechanism called optimistic unchoking. Periodically, the client will allocate an upload slot to a random uncooperative peer, unchoking it, therefore searching for more cooperative peers and giving second chances to previously choked peers. Thus the name optimistic unchoking. Moreover, Bittorrent downloads chunks following the rarest-first approach, that ensures the rarest chunks to be downloaded first. As an alternative a random chunk downloading approach is available.

In Figure 7.1 we show the basic interaction steps of the Bittorrent protocol. The peer that owns the complete resource file is called seeder. When a peer completes the file download, it becomes a new seeder. The first seeder is called initial seeder. First, the peer sharing a file has to fragment the data in a given number of identically sized chunks (usually between 32KB and 4MB). For each chunk, using the SHA-1 hash function, a hash is created and recorded in the torrent file (usually a file with .torrent extension). When a chunk is downloaded, the hash of the chunk is compared to the hash in the torrent file in order to check for errors. Torrent files have two sections, an info section which contains information about the shared file, and an announce section, with the URL address of the tracker. In order to download the shared file, then, the client connects to the tracker specified

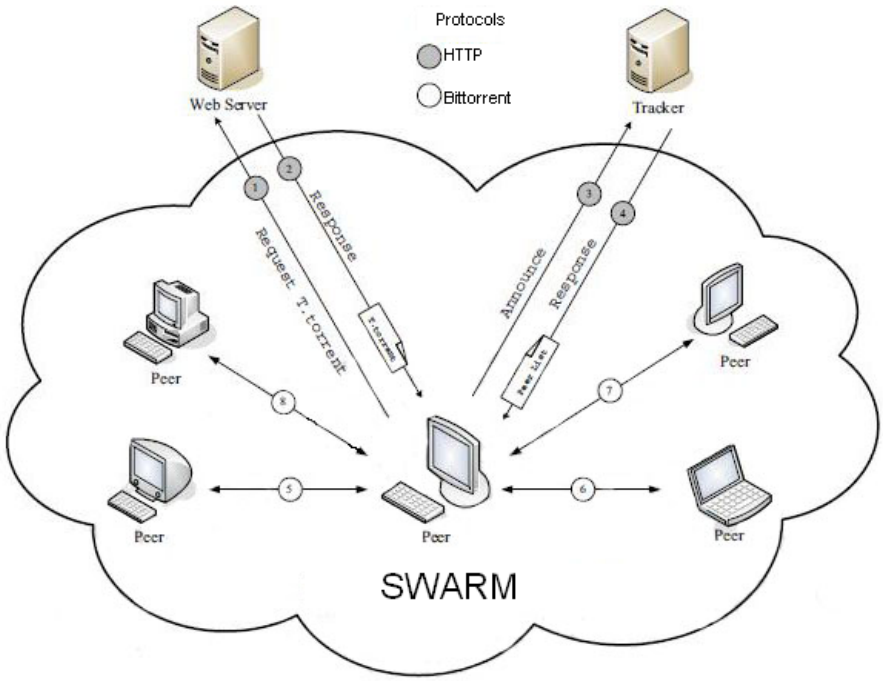


Figure 7.1: Bittorrent protocol basic interaction steps

in the torrent file. The tracker returns the list of peers that are in the swarm, together with information about the particular chunks they are sharing, allowing the peer client to get the missing chunks.

A peer node that wants to join the swarm, contacts the tracker server, obtains the peer nodes list and randomly contacts the peers to upload/download content from. A random peer selection may lead to several inefficiencies from an ISP point of view. In fact, this means that a node may choose to connect to a peer that is many hops away in the underlying network. This is a serious inefficiency if the same resource is available at a node closer to the asking peer. In particular, the Peer-to-Peer traffic may cross several AS and ISPs, drastically increasing the provider management costs. In Wireless Mesh Networks, random peer selection has even worse consequences, given the limited amount of resources. It may happen that several peer nodes are all inside the same mesh network. In this case, the best choice would be to get the required content from the nodes inside the mesh network. Instead, with a random peer choice, the worst case is when all peer

nodes download from nodes outside the WMN. This may lead to an overloading of the wireless links that connect to the mesh gateways, and a consequent packet loss that may disrupt the normal operation in the WMN.

IETF Working Group ALTO has introduced a mechanism to influence the peer nodes choice, hence giving the possibility to convey information about the network's preference to applications. In Section 7.2 we explore the proposed solution and show how it can be adapted to the P2P deployment in WMNs.

7.2 Application Layer Traffic Optimization

Idea behind the Application Layer Traffic Optimization is to provide P2P applications with knowledge, from the provider perspective, about the underlying network. This kind of information may be exploited by the P2P application, improving user perceived performance while at the same time optimizing resource usage. In a possible overlay network topology creation, the ALTO Service would be very helpful in the choice of the connection endpoints. The ALTO Service, directly controlled by the network manager, is also aware of the network resources availability, the critical bottlenecks, and the location of the peer nodes inside the network. An ALTO Server, then, provides Oracle-like (see Chapter 5) functionalities, supplying network information to applications. Typical information may be node connectivity properties, location of endpoints and costs for paths between network hosts.

Therefore, applications may no longer need to relay on measuring link performance metrics by themselves, and obtain information from the ALTO servers. Since the ALTO Service is deployed in order to provide network provider preferences on overlay nodes connection choices, it is understood that the service is bound to the particular network region the provider controls.

We may expect, then, to have localized ALTO Services reflecting the features of the particular considered network. When an overlay client is interested in obtaining network information, it has then to resolve first the correct ALTO server, responsible for the particular ISP or Autonomous System it is interested in. To

properly locate the correct ALTO Server, a hierarchical ALTO Service structure may be implemented, resembling the established DNS Server structure and reflecting the chance of having localized ALTO Servers: a main ALTO Server may exchange information with several ALTO Servers to collect answers for a client request, or, alternatively, provide the client with the local ALTO Server address.

The ALTO protocol structure is composed of two main services:

- **Server Information Service:** This service provides the list of information that can be retrieved from the ALTO server (or the other ALTO Servers that refer to the one contacted), like cost metrics and possible operations.
- **ALTO Information Services:** Several kinds of information may be provided by the ALTO server, each one encoding different kinds of network provider preferences. An example is represented by the Map Service, providing two kinds of information:
  - Network Map, providing network endpoints location (e.g., in terms of AS domain);
  - Cost Map, providing costs for potential connections.

  More services are available, like a Map Filtering Service, to filter some query results at the ALTO server, avoiding waste of bandwidth and client CPU usage; Endpoint Property Service, to allow Clients to query for some specific end point property, like location or connectivity type; Endpoint Cost Service allows clients to query for costs based on endpoints, obtaining the endpoints ranked based on some metric cost.

Since many endpoints may be close in terms of network connectivity, to improve scalability, ALTO may threat several nodes as one single entity, based on the Network Location property. Such grouping is referred to as ALTO Network Map.

As we showed above, Network Location is an important property, since we would prefer to let peers inside the same AS or ISP communicate with each other

instead of communicating with outside nodes. Network Map, then, may group internal nodes, based on several granularity degree, encouraging internal peering. In order to group peers, each group is identified by a Network Location ID, called PID and defined by the network provider. Each PID expresses some form of network aggregation and can denote subnets, ASs or sets of AS. PIDs represent, then, an indirect way to point out a network aggregation, they do not reveal explicit confidential information on ISP internal network composition and they improve scalability. Moreover, network costs may be provided in terms of PID groups.

Network costs are stored into the Cost Map, they indicate network operator preferences among the available PIDs and they can be updated independently of the Network Map entries. The Cost Map refers to both syntax and semantics of the presented information. In particular, each cost map entry has both type and mode fields, that specify both what the cost represents (i.e., routing metric cost, AS hop count, etc.), and how the values should be interpreted (e.g., numerical or ordinal). Moreover, since the costs may be defined between network locations, each characterized by a PID, the Cost Map may be strictly related to the Network Map.

Protocol Overview

The ALTO Service strongly relays on the HTTP implementation, and the interface is designed to exploit the HTTP infrastructure. In this way, it is possible to count on flexible server implementation strategies and simple redistribution of ALTO Information. Together with HTTP caches availability and HTTP authentication and encryption mechanisms, it is not to underestimate the fact that many P2P applications employ an HTTP client. In particular, the ALTO messages are denoted with the HTTP content-type ”application/alto”.

Therefore, we have that ALTO Request/Response messages are standard HTTP Request/Response messages, following the standard rules for the HTTP headers, with some components defined by the ALTO Protocol. Eventually, the ALTO Request input parameters are encoded and sent in the HTTP Request Body. If the request is successfully accomplished, the ALTO Response object contains the type

Service	**Operation**	**HTTP Method and URI Path**
Server Information	List Servers	*GET /info/servers*
Server Information	Capability	*GET /info/capability*
Map	Network Map	*GET /map/core/pid/net*
Map	Cost Map	*GET /map/core/pid/cost*
Map Filtering	Network Map	*POST /map/filter/pid/net*
Map Filtering	Cost Map	*POST /map/filter/pid/cost*
Endpoint Property	Lookup	*GET /endpoint/prop/<name>* *POST /endpoint/prop/lookup*
Endpoint Cost	Lookup	*POST /endpoint/cost/lookup*

Table 7.1: Overview of ALTO Requests

and data members that clarify the type of data encoded in the response and the actual requested data. In any case, the client is free to ignore the ALTO information.

In Table 7.1 we report a summary of the supported ALTO requests. In particular, the Server Information service provides information about available ALTO Servers and the list of the supported functionalities. The Map service provides clients with information about Network Map and Cost Map; the Network Map lists for each PID the network locations within the same PID, while the Cost Map returns the cost (together with type and value specification) for each path connecting each PID couple. Map Filtering service allows clients to specify some filtering criteria in order to obtain just a subset of the Map Service information. Finally, the Endpoint Property lookup service allows clients to lookup for particular endpoints properties, like PID or connectivity parameters. The Endpoint Cost feature allows clients to supply endpoints lists to the ALTO Server and to obtain in return costs among such endpoints, i.e., to obtain ranked lists of endpoints.

In Conclusion, IETF Working Group ALTO has introduced a mechanism to influence the peer nodes choice, hence giving the possibility to convey information about the underlaying network composition and status to applications. Next, in Section 7.2.1, we show a possible use case and later in Chapter 10, we will show how we implemented an ALTO Service in the context of wireless mesh networks.

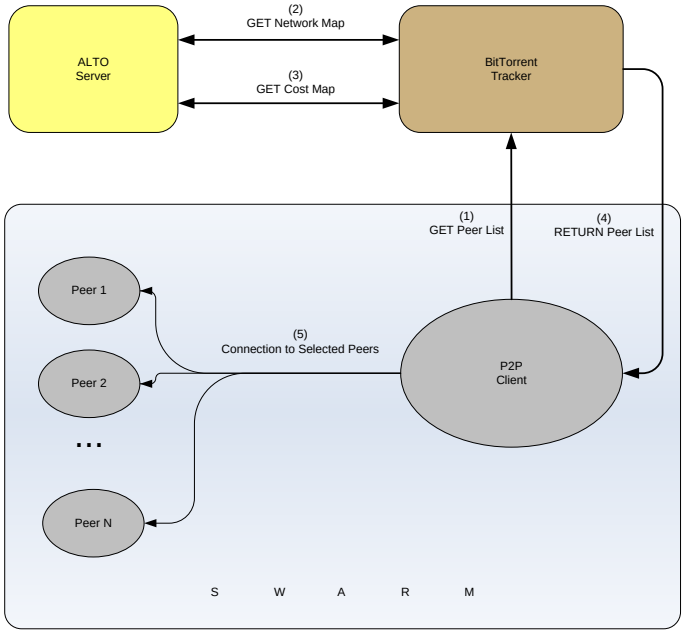


Figure 7.2: Bittorrent Tracker contacting an ALTO Server

7.2.1 ALTO Service: Bittorrent Tracker Server use case

In Section 7.1 we presented the Bittorent P2P protocol, and we underlined how the swarm and the peer selection phase was managed by a centralized Tracker server. Moreover, we emphasized how P2P applications may take advantage of underlaying network information, like endpoints location and ranking, in order to improve performance while optimizing network resources with more efficient traffic patterns. Acting as an ALTO Client, a Bittorrent tracker may obtain this kind of information from an ALTO Server.

Trackers usually manage large amounts of Peer-to-Peer clients, and they may obtain ALTO information from the ISPs the clients connect to, and manage such clients with the same aggregation level done by the ALTO servers.

In Figure 7.2 we show how a Bittorrent tracker, acting as an ALTO client, contacts the ALTO Server and exploit ALTO information to rank the peers inside the swarm. Below we explore the single steps:

Step (1) The P2P client joins the swarm and contacts the tracker for a list of nodes.

Step (2) The Tracker requests the Network Map using the Network Map query, obtaining all the PIDs covered by the ALTO server. The PIDs are characterized by IP prefixes, hence the Tracker can easily map each client IP address to the corresponding PID. Since asking peers may belong to different ISPs, the Tracker may have to contact multiple ALTO Servers.

Step (3) The Tracker requests the Cost Map for the acquired PIDs from the ALTO Server.

Step (4) The P2P Tracker finally returns the list to the client. The returned peer list is obtained from the ALTO server Network Map and Cost Map information.

Step (5) The P2P client connects to the selected peers.

7.3 Chapter Conclusions

In this Chapter we introduced the IETF ALTO Working group solution to the P2P traffic optimization issue. We introduced the ALTO Service and presented its purpose, i.e., to convey information about the underlying physical network to the P2P applications. We presented the ALTO protocol structure, its main functionalities and services, like the ALTO Information Service, providing information about endpoints location and costs for connecting to such endpoints.

Finally, we introduced the Tracker-based Bittorrent P2P paradigm and showed a simple use case where, thanks to the ALTO Service, such application have the potential to both improve user perceived performance and reduce network inefficiencies.

Chapter 8

An Interference Aware Path Metric for Wireless Mesh Networks

In this Chapter we present a new link metric for multi-radio wireless mesh networks. The link metric is used to pick the best available path to route packets, as well as to rank nodes inside the wireless network, following the approach described in Chapter 7.

The motivation behind a context metric takes origin from the insufficiency of common metrics in managing interference among wireless links. Common wireless multi-hop routing protocols are usually designed to find low cost paths between network nodes. The cost of a path is then the sum of the single link costs. The advent of multi-radio wireless mesh networks highlights the weakness of such an approach. As we saw in Section 2.3, the main issue in wireless technology is the interference among the wireless links sharing the same frequency channel. The interference is reduced by deploying on each node several radio interfaces, each set on a different non-interfering channel. Since the number of orthogonal non-interfering channels is limited (see Chapter 3), to maximize the benefit of multiple radios it is crucial to utilize those paths that have the minimum number of interfering links.

Typical link metrics like ETX [69] or ETT [70] are clearly unsuitable for minimizing interference, since links are considered independently from each other, while the nature of the interference problem intrinsically involves several links together.

Throughout the Chapter we develop a self-interference aware routing metric, which takes into account the actual interdependencies among the interfering wireless links.

First, in Section 8.1, we consider the most common used wireless link metrics and show their inadequacies. Then, we justify the interference-aware context metric and introduce it in Section 8.2 and later on, in Chapter 10 we show how we can exploit this path metric to rank nodes in a WMN and integrate this service in a centralized ALTO Agent.

8.1 Weak spots in current wireless link metrics

Common routing protocols are designed to choose the best among the possible routing paths in a wireless network. Such routing protocols take into consideration the cost of a path and pick the path that minimize such cost. Usually, the cost of a path is computed as the sum of the single links metrics, where typical link metrics are the Round Trip Time, ETX, expected transmission count, or ETT, expected transmission time.

Evaluating path performance as the sum of single link costs leads to an overestimation of the path performance value. The main problem is that the overall path performance cannot be measured by the simple sum of the component links. In multi-radio wireless mesh network, for example, we have correctly assigned a channel to each radio, every link has a collision domain and the available channel capacity is affected directly by the traffic that is routed on the interfering links. In a multi-hop path between two end hosts, then, the best choice would be to choose the minimum number of interfering links, since the induced interference would reduce the per link capacity.

Throughout this Section, metrics already mentioned in Chapter 5 are now described in more detail. As outlined in Section 5.1, one of the most famous single link metric is the ETX [69] (and his extension ETT [70] for multi-radio wireless networks) metric. The ETX metric is defined as to measure the expected number of transmissions to correctly send a packet over a wireless link. If we denote

by p_f and p_r on a wireless link the packet loss probability in forward and reverse direction respectively, we can compute the probability that a packet cannot reach the destination and denote as p such probability. Since each packet has to be acknowledged on reception, we have that a packet is not correctly received with a probability p, where:

$$p = 1 - (1 - p_f) \cdot (1 - p_r) \tag{8.1}$$

The p_f and p_r probabilities can be computed by means of broadcast probe packets transmission. Each node periodically broadcasts a probe packet, tracks down the number of received probe packets from each neighbour (it is considered a standard time window of ten second) and includes such information into its own broadcast probe. Then, p_r is derived from the number of probe packets received, p_f is derived from the information about themselves stored in the received last probe packet. Then, we will have that a packet is correctly received and acknowledged after k attempts with a probability $s(k)$, where:

$$s(k) = p^{k-1} \cdot (1 - p) \tag{8.2}$$

Therefore, from equation 8.2 we have that the expected number of transmissions to correctly deliver a packet, denoted as ETX, is given by:

$$ETX = \sum_{k=1}^{+\infty} k \cdot s(k) = \frac{1}{1 - p} \tag{8.3}$$

Consequently, a path metric will be the sum of all the ETX metrics on the single wireless links on the path.

The ETX metric has some limits in multiradio wireless networks scenario, though. For example, when multiple radios are adopted, with both 802.11b and 802.11a technologies, the ETX metric tends to choose 802.11b based links. This is due to two ETX limits. First, the ETX metric considers only packet loss ratios and not the available link bandwidths. Second, when using the ETX metric, it is a fact that shortest paths will be preferred over longer paths. When only 802.11b technology is deployed on wireless nodes, ETX is not able to exploit multi-radio

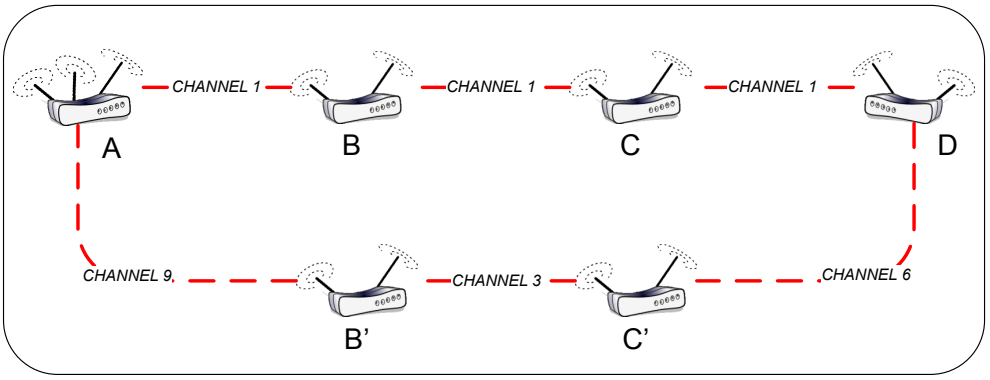


Figure 8.1: ETX/ETT link metric insufficient when estimating path performance

availability and it will eventually choose sub-optimal paths, since it is not able to prefer interference-free paths, i.e., to prefer those paths that are able to properly reuse the available wireless channels.

In [70] the authors propose the ETT metric, an extension to the ETX metric. The ETT metric is introduced to take into consideration the link bandwidth. If we denote as S the packet size and as B the link data rate, ETT is defined as:

$$ETT = ETX \cdot \frac{S}{B} \tag{8.4}$$

In Figure 8.1, a simple wireless network topology emphasize the inadequacy of single link metric in evaluating the overall path performance. In particular, there are two routes between nodes A and D, one through nodes B and C, the other route through nodes B' and C'. We consider all channels to be orthogonal between each other and we assume that all links have the same quality. Since links on path $A \rightarrow B \rightarrow C \rightarrow D$ are all set on Channel 1, the path is highly affected by interference which negatively influences the overall path performance. On the other hand, all the wireless links on the $A \rightarrow B' \rightarrow C' \rightarrow D$ path are set on orthogonal channels, hence reducing the overall interference along the path.

Judging the overall path performance by summing the single ETT/ETX link metrics may lead to awkward situations: in Figure 8.1 the $A \rightarrow B \rightarrow C \rightarrow D$ path may have better ETT/ETX metric performance than the $A \rightarrow B' \rightarrow C' \rightarrow D$ path, leading to inefficient routing paths. Therefore, single path performance cannot be

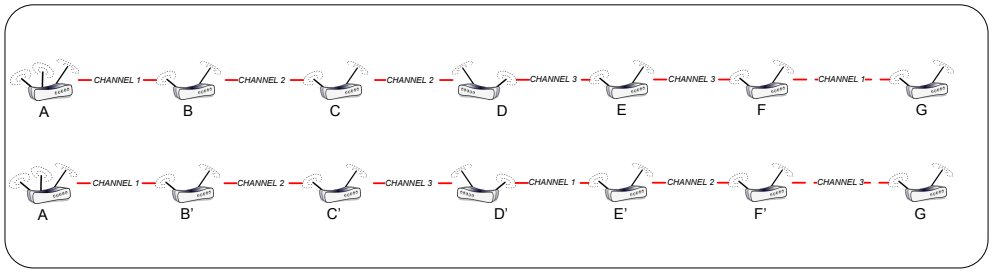


Figure 8.2: WCETT link metric insufficient when considering mutual interference between links in a routing path

considered examining the single links in isolation of each other, but we have to study link metrics that take into consideration all the interdependencies that arise on the route followed so far by the packets.

Path metrics have been introduced in literature, like the WCETT metric [71], trying to model interference to some extent. The WCETT metric is defined as follows:

$$X_j = \sum_{Hop\ i\ is\ on\ channel\ j} ETT_i \qquad 1 \leq j \leq k \tag{8.5}$$

$$WCETT = (1-\beta) \cdot \sum_{i=1}^{n} ETT_i + \beta \cdot \max_{1 \leq j \leq k} X_j \tag{8.6}$$

From equation 8.5, we have that X_j is the sum of the transmission times on all the hops set on channel j. In equation 8.6, it is considered the maximum among the X_j on a path $\mathcal{P}$, since the path throughput is tied down by the bottleneck channel. The first term in equation 8.6 is the sum of all the transmission times along all the hops in the network, i.e., the total resource consumption on that path. The second term reflects the bottleneck channel that will have the most influence on the overall path throughput, and it is introduced in order to capture the self interference effect. The β parameter is introduced to weight the two terms, i.e., β is comprised between 0 and 1.

WCETT metric basic idea is to penalize more those routing paths that use the same channel multiple times. The model proposed by the authors, though, does not take into consideration the distance between nodes and the real packet colli-

sion events. In fact, even if two nodes use the same channel, the physical distance between them may be high enough to avoid interference and packet collisions. In this case, then, the WCETT metric does not take into consideration spatial reuse and as a consequence, we could underestimate good paths. An example case is reported in Figure 8.2.

In Figure 8.2 we have two paths between nodes A and G, each with a particular wireless channel configuration. Each node has two available radios, and we hypothesize that channel 1,2 and 3 are all orthogonal between each other and all links have the same quality. Roughly, we may consider two links to be interfering if they are set on the same channel and they are within two hop distance. The ideal path is $A \rightarrow B' \rightarrow C' \rightarrow D' \rightarrow E' \rightarrow F' \rightarrow G$, since there is no interference on the one and two hop links. From WCETT point of view, both $A \rightarrow B' \rightarrow C' \rightarrow D' \rightarrow E' \rightarrow F' \rightarrow G$ and $A \rightarrow B \rightarrow C \rightarrow D \rightarrow E \rightarrow F \rightarrow G$ paths have the same performance, since they both use the same number of channels on the same number of links. This means that WCETT interference model is not able to properly choose the best paths. In WCETT, all links in a path sharing the same channel are considered to be in the same collision domain, while this may not always be the case.

In [72] and [73], the authors proposed the MIC metric, together with a procedure to find the optimal route between two hosts, under this metric. However, the metric is forced to comply with some constraints that may not be aware of link interdependencies and this may lead to path route inefficiencies. The MIC metric is defined as follows:

$$\alpha = \frac{1}{N \cdot \min ETT} \tag{8.7}$$

$$IRU_k = ETT_k \cdot N_k \tag{8.8}$$

$$CSC_i = \begin{cases} w_1 & \text{if previous hop is on different channel} \\ w_2 & \text{otherwise} \end{cases} \tag{8.9}$$

$$MIC(\mathcal{P}) = \alpha \cdot \sum_{Link\ k\ is\ on\ path\ \mathcal{P}} IRU_k + \sum_{Node\ i\ is\ on\ path\ \mathcal{P}} CSC_i \tag{8.10}$$

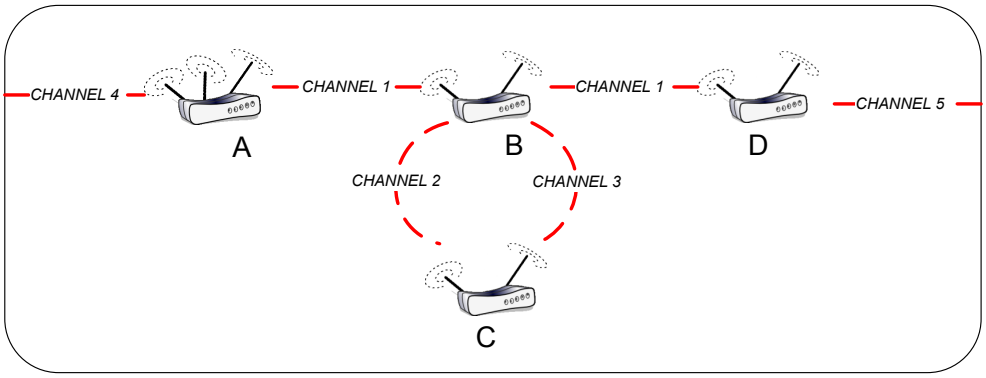


Figure 8.3: MIC link metric insufficient when modeling self-interference with all previous hops

In equation 8.7, N is the total number of the nodes in the network, and $min(ETT)$ is the smallest ETT in the network. Equation 8.8 defines the *Interference- aware Resource Usage*, which is introduced to consider inter flow interference: when node k is transmitting, N_k is the number of neighbours which interfere with the transmission. For each link, the greater N_k, the greater the penalization. At the same time, equation 8.9 defines the *Channel Switching Cost*, which reflects the self-interference, since it gives heavier weights to paths with consecutive links using the same channel: in equation 8.9 immediate previous hop is considered, while in [73] the authors propose an interference model extension for more than one hop. A major drawback in the MIC metric, though, is the limited amount of memory extent. Moreover, again, self-interference is considered in terms of hop-count distance.

In Figure 8.3 we report a simple case that may be critical if the MIC metric is considered. We consider four wireless nodes, each with two wireless network interfaces. We consider all the channels to be orthogonal and equivalent link condition. Due to possible memory extent problem, self interference situations like the one in Figure 8.3 may not be modeled and due to lower cost, path $A-B-C-B-D$ may be preferred to direct path $A-B-C-D$. Considering the interference only in terms of hop count distance, then, may lead to inefficient path choices. If link $B-D$ would have been penalized taking into consideration its collision domain, instead of its last hop, the path metric would have been able

to avoid the ambiguous situation.

8.2 Interference and Bandwidth Aware Routing Metric

In Section 8.1 we considered several link and path routing metrics and showed their inadequacy in modeling interference and link interdependencies that often occur in multi-radio wireless mesh networks. In particular, this is true both for classic link metrics like ETX and its extension, ETT, and for more advanced metrics like WCETT and MIC which try to take into consideration somehow the interference along a routing path.

Therefore we introduced IBAM, an Interference and Bandwidth Aware routing path Metric. In IBAM, each link is evaluated based on how previous links in the path have been chosen, hence on each link we take into account the impact of previous links on the path. For instance, for a path $\mathcal{P}$, the cost may be computed as:

$$\begin{aligned} cost(\mathcal{P}) &= c(link_1) + c(link_2|link_1) + c(link_3|link_2, link_1) + \cdots + \\ &\quad + c(link_n|link_{n-1} \cdots link_1) \end{aligned} \tag{8.11}$$

where $c(link_n|link_{n-1} \cdots link_1)$ notation points out that cost of $link_n$ is a function of the previous links on path $\mathcal{P}$.

In Figure 8.4, we report a small wireless topology case and show how the link cost depends on the links crossed so far on the path $\mathcal{P} = A \rightarrow F$. We suppose all links have the same quality. For simplicity, we assume that node A selected the link on channel 1 to communicate with node B on path $\mathcal{P}$. Clearly direct link $B \rightarrow F$ cannot be considered alone. Since we know that packets are flowing from Node A on channel 1, choosing link $B \rightarrow F$, which is also set on channel 1, would imply intra-flow self interference. Link $B \rightarrow F$ on path $\mathcal{P}$, then, would have higher weight than when considered alone. Based on the example, then, we see that we have to condition our choice and to properly weight links based on previous hops on the path.

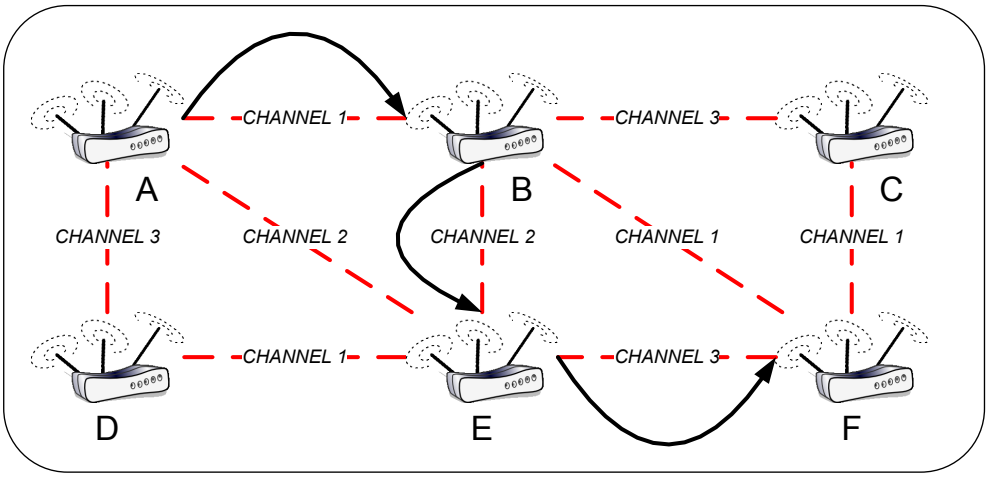


Figure 8.4: Computing path performance metric

We define the IBAM path metric as follows:

$$IBAM(\mathcal{P}) = \sum_{k \in \mathcal{P}} \frac{ETT_k}{C_0 - \sum_{j \in \mathcal{J}_k} \frac{f_j}{C_j}} \tag{8.12}$$

$$\mathcal{J}_k = \mathcal{D}(k) \bigcap \mathcal{P} \tag{8.13}$$

where $\mathcal{D}(k)$ is the collision domain defined in Chapter 3, equation 3.4, reported here for the sake of convenience. Given the set E of all the links in the network, we have that the collision domain $\mathcal{D}$ of link $u \xrightarrow{c} v$ is computed as:

$$\mathcal{D}(u \xrightarrow{c} v) = \left\{ x \xrightarrow{c} y \in E \mid x \to y \in \mathcal{N}(u \to v) \right\} \tag{8.14}$$

where in equation 3.3 we defined the potential collision domain $\mathcal{N}(u \to v)$ as:

$$\mathcal{N}(u \to v) = \left\{ x \to y \in E_I \mid \frac{G_{uv} P(u \to v)}{G_{xv} P(x \to y) + n_v} < \gamma_{c(u \to v)} \right\} \tag{8.15}$$

In equation 8.15, $x \to y \in E_I$ means that x and y can communicate provided that they are assigned the same channel.

From equation 8.12 we can see that for each link k on the path $\mathcal{P}$, the IBAM metric considers the ETT [70] value of that link, where the ETT is computed as shown in equation 8.4. Moreover, for each link k we consider the set $\mathcal{J}_k$ (computed as in equation 8.13) of all the links j that are in the same collision domain as link k and are on the same path $\mathcal{P}$. Therefore, with the set $\mathcal{J}_k$ we consider the context of link k and we are able to take into consideration all the previous hop on the path that interfere with link k. According to equation 8.14, the link context we consider is the actual physical interference transmission range, thus avoiding all the ambiguity of the metrics considered in Section 8.1.

For each link j belonging to the local context of link k we sum the ratio between the traffic flow on link j, f_j, and the available nominal capacity C_j. The motivation behind this is that we want to take in consideration the real utilization of the collision domain $\mathcal{D}$ of a link k, and not only the number of the previous hops interfering with the current link k. One of the side effects, in fact, may be to overpenalize those links with a large set of inactive interfering links, while choosing the links with smaller collision domains, but with higher traffic activity. Here, the rationale is that we do not equally penalize all the interfering links, but we also take into consideration the traffic routed on those links. Finally, C_0 is a constant value.

8.3 Chapter Conclusions

In this Chapter we showed inadequacy of current routing path metrics in Wireless Mesh Networks that consider simple sums of single wireless link metrics and presented a self-interference aware routing metric that is based on the local context of a link. In our metric, we defined the local context of a link k as the set of all the previous hops in a path $\mathcal{P}$ that belong to the collision domain of link k. For each next-hop link k on a path $\mathcal{P}$, then, we penalized a link taking into account the total utilization of all the links in the collision domain of link k. In Chapter 10 we will show how we can exploit wireless specific metrics in order to provide overlay networks with performance information about peer nodes.

Chapter 9

Testing architecture for Wireless Mesh Networks experiments

The ultimate success of the Wireless Mesh Network paradigm in large scale deployments depends on the ability to test it in real world scenarios. A typical application scenario which is worth to be investigated is the Peer-to-Peer traffic management. The creation of large scale testbeds for evaluating wireless mesh technologies and protocols, and for testing their ability to support real world applications in realistic environments, is then a crucial step. In this Chapter we present how we integrated an OMF-based (OMF stands for cOntrol and Management Framework) wireless testbed in the planetary-scale PlanetLab testbed, making it possible for PlanetLab users to run experiments spanning on both PlanetLab and on nodes belonging to the OMF-based wireless testbed.

In Chapter 10 we will exploit this framework to conduct experiments and to test our implementation of the ALTO Server and ALTO Agent in the context of Wireless Mesh Networks. The possibility of running this kind of experiments highlighted several real-world issues which could be investigated thanks to our hybrid experimental scenario.

9.1 Introduction

Testing Wireless Mesh Networks in large scale deployments and with realistic traffic loads is of crucial importance [74]. Due to the inherent difficulty of captur-

ing all the relevant aspects of the real behavior of these systems in analytical or simulation models, research on WMNs has always heavily relied on experimental testbeds [75]. Setting up large-scale wireless mesh testbeds, though, is a difficult and costly process.

Moreover, since wireless mesh networks are usually employed as access networks to the Internet, taking into account the complexity of the real Internet is part of the efforts to be made for a realistic assessment of these networks. To this purpose, we reckon that it is important to carefully evaluate the impact of Peer-to-Peer applications on wireless mesh access networks and their resource management schemes, since nowadays more than 50 percent of the overall Internet traffic is produced by applications of this kind.

In this Chapter we present an architectural integration of geographically distributed OMF-based wireless testbeds in the global scale PlanetLab environment [5]. We show a practical example of such an architecture by describing the integration of the WiLEE testbed, an OMF-based wireless testbed located in Naples, Italy, with the PlanetLab Europe testbed.

OMF (*cOntrol and Management Framework*) is a well-established tool to manage wireless testbeds. Originally developed for the ORBIT wireless testbed at Winlab, Rutgers University, OMF is now deployed in several testbeds in Australia, Europe, and in the U.S. [4]. Our system allows the seamless integration of the OMF-resources into the global scale PlanetLab infrastructure, creating a synergic interaction between the two environments. In particular, thanks to our contribution, PlanetLab users may run experiments involving resources provided and controlled by the OMF-based wireless testbeds. The contribution we present into this Chapter is in line with current ongoing efforts towards the so called ”federation” of experimental infrastructures, an approach that appears as the most reasonable way to build large-scale heterogeneous testbeds. The problem of heterogeneous testbeds federation is currently under investigations of both the GENI initiative in the U.S. [76] and the FIRE initiative in Europe [77]. For instance, federation between PlanetLab and EMULAB [78] is currently being investigated in the context of the GENI initiative, as reported in [79].

Other efforts have been made in order to integrate OMF-based testbeds in PlanetLab, like [80] and [81]. Those approaches differ significantly from ours in the level of integration achieved, as they see the OMF testbed as a single resource, that can be attached to a PlanetLab set of nodes as a whole, i.e. only one experiment at a time is allowed on the OMF testbed. In our approach, as we employ a Scheduler which is able to reserve single nodes and wireless channels of the OMF-based testbed for slices, multiple concurrent experiments are possible, which improves significantly the utilization of the OMF-based wireless testbed.

Finally, in Chapter 10 we also illustrate how we use the integrated testbed setup to conduct an experiment aimed at evaluating the ALTO optimization technique for the Bittorrent file sharing application in the context of WMNs.

In Section 9.2 we present some of the recent development regarding testbeds integration. Then, in Section 9.3 we provide a brief description of the PlanetLab testbed. In Section 9.4 we describe the OMF testbed management framework and the NITOS Scheduler component [82], developed at CERTH (Centre for Research and Technology - Hellas). Finally, in Section 9.5 we describe the integration steps that we developed to allow for distributed experiments involving both PlanetLab and geographically distributed OMF-based wireless mesh testbeds.

9.2 Testbeds integration: State of the art

In this Chapter we are going to present an architectural solution to integrate a number of local OMF-based wireless testbeds with the global-scale PlanetLab environment.

Our solution is a first technical solution towards the federation of these two kinds of testbeds. An attempt to add heterogeneity in PlanetLab by integration of ORBIT testbeds is presented in [81]. In this paper, the authors propose two models of integration.

The first model (PDIE, *PlanetLab Driven Integrated Experimentation*) is intended to serve PlanetLab users who want to extend their experiments to include wireless networks at the edge without changing the PlanetLab interface, while the

second model (ODIE, *ORBIT Driven Integrated Experimentation*) is intended to serve ORBIT wireless network experimenters who want to improve their experiments by adding wired network features without major changes to their code. This approach is different from ours, as it just supports the execution of one experiment at a time and employs tunnels.

A similar approach was taken in [83]. The authors aimed at integrating the VINI virtual network infrastructure [84] with OMF-based testbeds. The intention was to enable Layer 3 experimentations by allowing users to create virtual topologies spanning both wired and wireless links. Our approach also intends to recreate in the testbed the same operational situation that exists in real networks, in which a private mesh is connected to the Internet through NAT gateways. For achieving a full-fledged federation of the two environments, other issues need to be fully solved, such as the creation of a single sign-on mechanisms for the two environments. We will study the problems in the next Sections.

9.3 PlanetLab Overview

PlanetLab is a geographically distributed testbed for deploying and evaluating planetary-scale network applications in a highly realistic context. Nowadays the testbed is composed of more than 1000 computers, hosted by about 500 academic institutions and industrial research laboratories. In the rest of this Chapter we will refer to PlanetLab Europe, a European-wide testbed which is federated with PlanetLab since 2007. Thanks to this integration, an experiment created in PlanetLab Europe may comprise nodes belonging to PlanetLab, and viceversa.

Figure 9.1 shows a conceptual view of the current architecture of the PlanetLab testbed, whose node set is the union of disjoint subsets, each of which is managed by a separate authority. Two such authorities exist so far: one located at Princeton University (PLC) and another located at Université Pierre et Marie Curie UPMC in Paris (PLE). Each testbed authority contains an entity called *Slice Authority* (SA), which maintains state for the set of system-wide slices for which it is responsible. The slice authority includes a database that records the persis-

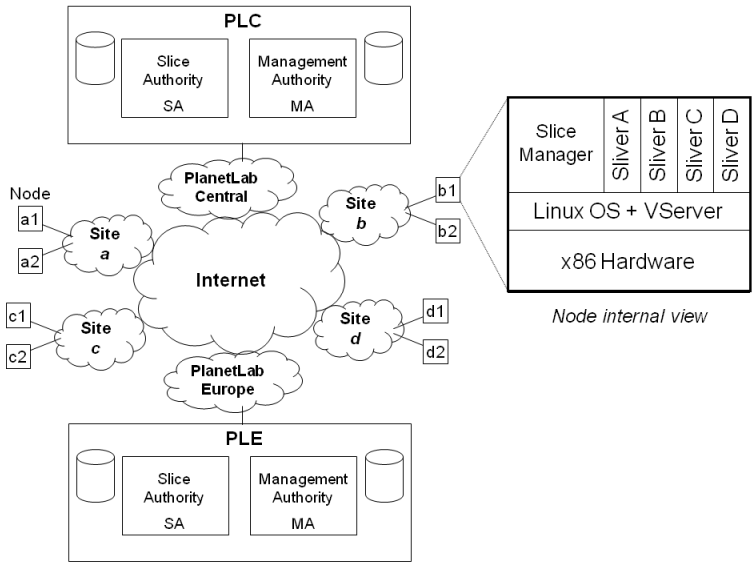


Figure 9.1: Conceptual PlanetLab architecture.

tent state of each registered slice, including information about every user that has access to the slice [85]. Testbed authorities also include a so called *Management Authority* (MA), which is responsible for installing and managing the updates of software running on the nodes. It also monitors these nodes for correct behavior, and takes appropriate action when anomalies and failures are detected. The MA maintains a database of registered nodes at each site. Each node is affiliated with an organization (owner) and is located at a site belonging to the organization.

To run a distributed experiment over PlanetLab, users need to be associated to a *slice*, which is a collection of virtual machines (VMs) installed on a user-defined subset of all the testbed nodes. Slices run concurrently on PlanetLab, acting as network-wide containers that isolate services from each other. A slice instance on a particular node is called a *sliver*. Slivers are Virtual Machines created in a Linux-based environment by means of the VServer virtualization technology. By means of so-called *contexts*, VServer hides all processes outside of a given scope, and prohibits any unwanted interaction between a process inside a context and all the processes belonging to other contexts. VServer is able to isolate services with respect to the filesystem, memory, CPU and bandwidth. However, it does not

provide complete virtualization of the networking stack since all slivers in a node share the same IP address and port space. The adoption of VServer in PlanetLab is mainly motivated by the need of scalability, since up to hundreds of slivers may need to be instantiated on the same physical server [86].

9.4 The OMF framework

OMF (cOntrol and Management Framework) is a platform which supports the management and the automatic execution of experiments on a networking testbed. An OMF testbed is made of several nodes, each equipped with several wireless interfaces and used for users experiments; moreover, it has a few other software components, providing the infrastructure layer needed to govern the system and to define its configuration.

These software components support all the phases of an experiment, from the resources definition to the experimental data collection. The most important component is the *Experiment Controller* (EC), which provides also the interface to the user. The EC is fed with an experiment description, provided by the user, and it takes care of organizing the testbed resources in order to accomplish the required experimental steps.

Such an experiment description is a script written in the OEFDL language, a domain-specific language derived from Ruby. The EC interacts with the *Aggregate Manager*, the entity responsible for managing and controlling the status of the testbed resources.

The EC also interacts with the *Resource Controllers* (RCs), components running in each of the testbed nodes. RCs are responsible of performing local configuration steps, e.g., configuring the channels on the WiFi interfaces, and controlling the applications running in the single nodes.

9.4.1 The NITOS scheduler

In an OMF testbed, resources are of two types: nodes and spectrum. In its basic form, OMF assigns resources to users following a First-Come-First-Served strat-

egy: the user supplies an experiment description and the system tries to assign the resources requested by the experiment if they are available. OMF can be customized, though, to support some kind of resource reservation. In the NITOS and WiLEE testbeds, we adopted an extended version of OMF which allows the execution of multiple experiments in parallel on the same testbed, by guaranteeing that the requested resources are actually exclusively assigned to each experiment for its entire duration. This is achieved by assigning a different subset of nodes and wireless channels to each user for a specific time interval, through a testbed resource scheduler, the NITOS Scheduler, described in [87]. These subsets are reserved in advance through the Scheduler and the access to them is enforced during experiment time so that users can have access only to the resources they had previously booked.

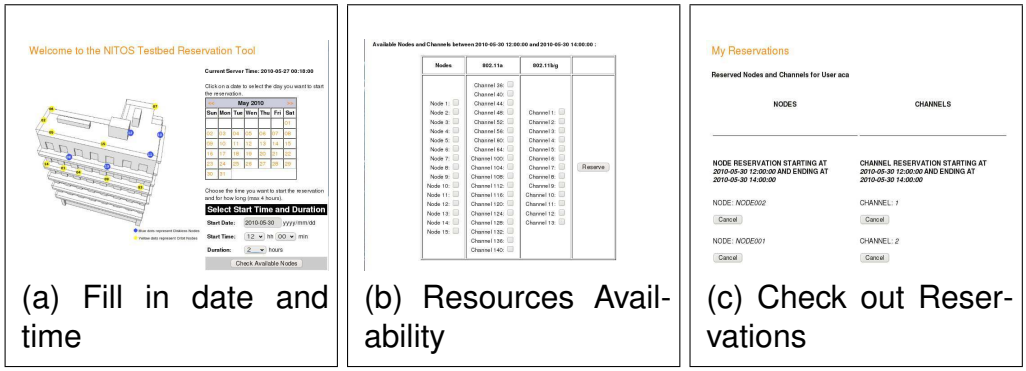
While allowing the concurrent execution of multiple experiments on the same testbed, the NITOS Scheduler does not support the integration of the testbed resources with other experimental facilities, like PlanetLab. For this reason we decided to extend it, in order to enable the seamless integration of OMF and PlanetLab Europe resources, as explained in the following Section.

Scheduler Scheme

The slices are created on the testbed following the user reservation. The reason that we can do spectrum slicing is that the user has to reserve nodes and spectrum for a specified time range first, and then he can login to the testbed and execute his experiments. When the reservation procedure is over, the system is aware of the resources that he needs and for how long he needs them. During this time range, no other users can use any of these nodes or spectrum that may cause interference. The reservation procedure is illustrated in this Section.

First of all, the user has to decide on which date and time he would like to reserve a slice. The time is slotted with each slot duration set to 30 minutes. Then, he checks for resource availability. The testbed resources are the nodes and the wireless spectrum.

The scheduler keeps all reservations in a database. A reservation is a set of



(a) Fill in date and time

(b) Resources Availability

(c) Check out Reservations

Figure 9.2: Resource reservation steps

nodes, spectrum and a time range. When a user checks for available nodes, the scheduler is actually checking its database for any possible records in the time range that the user specified. Finally, it returns only the available set of nodes and spectrum at the specified time range. In this way, the system ensures that both the time and the frequency division requirements will be met.

The time division requirements will be met, since the scheduler allows the users to reserve only the nodes which have not already been reserved by any users in the specified time range.

At the same time, the frequency division requirements will also be met, since the scheduler allows the users to reserve only the channels (frequencies) which have not already been reserved by any users in the specified time range.

The resource reservation steps are shown in Figure 9.2. At this point, the user can properly select nodes and spectrum that he required for his experiments. When he selects the resources, the scheduler database is being updated with his data.

The reservation procedure ends here. From this point on, the scheduler is responsible for ensuring that the user will use only the reserved slice for the specified time period.

OMF extension to support NITOS scheduler features

The scheduler is primarily composed of two parts: a user interface, which is responsible for guiding the user through the reservation process, and a system com-

ponent, which controls the slices by ensuring that users experiments will only use the reserved resources. The user interface role has been illustrated in the previous subsection, while the scheduler encapsulation system will be illustrated in this subsection.

Until now we have described the part of the scheduler which is focused on the user and his choices at reservation. This, however, may not be enough. We need to ensure that the experimenters will stick on their choices and, even if they try, the system will not allow them to use any resources that they have not reserved.

In order to do that, we have chosen to extend OMF. Here, we give a detailed description of the additions and the extensions we had to make inside such framework to integrate spectrum slicing support.

Before anything else, we need a way for OMF and the scheduler's database to communicate. For this purpose, we have added one more service group to Aggregate Manager named scheduler and we have added one more service to the inventory service group. Next, we show what these services are responsible for.

The inventory service group is developed inside OMF and provides a set of webservices that provide general information about the testbed (such as node names, IP addresses, etc). This information is stored in a database residing on the testbed server and the inventory service group reads this database to return the proper response. Our addition here is a service which gets a node location (that is its coordinates) based on its IP address. We note here that the node location is the same information on both the scheduler and the testbed databases and, thus, we can use it to do the matching. We have added this service because when an experiment is executed, OMF does not know a node's location; only its IP address.

Now that scheduler knows the exact location of the node, it can use the scheduler service group to get any information needed from the scheduler's database. Namely, the services provided by this group provide functionality to get node reservations based on its coordinates, the spectrum that this reservation contains and the user that owns it. Furthermore, it provides services that can do the matching between a channel or a frequency number and the respective spectrum identification number as it is stored in the database. All this information will be used

by Resource Controller (RC), which decides whether to allow the user use the channel or not.

So, the RC is responsible for deciding whether the resources declared in the experiment can be allocated to the user. In order to make such decisions, the RC has to ask the scheduler's database if the specified resources have been reserved by the user. So, when the experiment sets the wireless card channel, this information is passed to the RC, which now knows the channel along with the corresponding IP address. All that is needed is the user identification to check with the schedulers database if the current channel (and, of course, node) should be allocated to that user.

However, this is not straightforward, since the user usually logs into the node as administrator, so he has full privileges. So, we need to track where did he use the username that he also used for registering. The scheduler is designed in such a manner that, when a user registers to the system, then an account with the same username and password is automatically created to the testbed's server. The user uses this account to both access the user interface and the testbed server (using secure shell connection). This can solve our problem, since we can say for sure that the user that is running the experiment is logged into the console with the same username that he has made his reservation. This information, though, relies on the testbed server, while the RC runs on the client side, i.e., on the nodes. Therefore, we need to pass such information from the server to the clients. This operation is done by the Experiment Controller, the OMF service that is running on the server side and is responsible for controlling the experiment execution.

Using its built-in message passing mechanism, EC tells the RC the username of the experimenter. Finally, the RC now owns what he needs to do the matching, except the date. The system should not rely on the user to keep the clock of his clients synchronized. This is why EC sends, along with the username, the current date and the RC adjusts its clock accordingly.

At this point, RC has all the information needed to check with the scheduler if the requested resources should be allocated to the user. Using the web services we described above, the RC checks if there is a reservation at that time and if

the spectrum reserved for such reservation matches the channel that the user has requested.

If all goes well, then the RC lets the experiment execution move on. Otherwise, it notifies the EC that a resource violation took place and it stops its execution (without assigning any resource). When the EC receives that message, the execution is terminated immediately and an ERROR message is thrown back to the experimenter describing the resource violation.

9.5 PlanetLab and OMF integration

Our main goal has been to integrate a local OMF-based wireless testbed with the global scale PlanetLab Europe infrastructure. The architecture we envisioned, however, is general, and it allows to integrate any OMF-based wireless testbed in PlanetLab.

In PlanetLab, slice creation and resource allocation are decoupled. In fact, when a slice is first created, a best effort service is associated with it and resources are acquired and released by the slice during its entire lifetime. Therefore, slices are not bound to sets of guaranteed resources. Such an approach has been deliberately chosen in the original PlanetLab design. PlanetLab, in fact, has not been designed for controlled experiments, but to test services in real world conditions [88].

Thanks to the integration we achieved, PlanetLab users can add to their slices, in addition to PlanetLab nodes, nodes belonging to the integrated OMF-based wireless testbed, i.e., the WiLEE testbed. In [80], it has been realized integration of an OMF-based wireless testbed in PlanetLab Europe, but only one slice at a time was allowed to include nodes of the wireless testbed. By leveraging and extending the features of the NITOS Scheduler, it is possible to allow multiple PlanetLab slices to include resources from the same OMF-based wireless testbed. Each of these slices is given a subset of nodes and wireless channels, resources which have to be reserved in advance by means of the extended NITOS Scheduler.

In the integrated scenario, PlanetLab users can either run isolated experiments

on the OMF-based wireless testbed, i.e., experiments which involve only nodes of the OMF-based wireless testbed, or can use the wireless testbed as an access network for the set of *co-located PlanetLab nodes*, i.e. a set of PlanetLab nodes which are in range of wireless transmission with the OMF-based wireless testbed. In the latter case, PlanetLab users have also to reserve a wireless interface from one of the co-located PlanetLab node. The wireless interfaces are the access technologies that allows the co-located PlanetLab nodes and the OMF-based wireless testbed to communicate.

The architecture we propose is depicted in Figure 10.1. It consists of the following elements:

- A PlanetLab site S whose nodes are equipped with one or more WiFi interfaces that allow them to be connected to a local wireless OMF testbed. In the following these nodes are called *PlanetLab Edge Nodes* (PL-Edge Nodes).
- The PlanetLab Europe Central server (PLE), which hosts the information on the PlanetLab Europe testbed, e.g. user accounts, slices.
- The OMF testbed and its components: the Aggregate Manager, the Experiment Controller and the Gateway Service.
- The extended NITOS Scheduler, used to manage the reservation of resources shared through booking.

The Gateway Service is implemented in a Linux box and acts as a Network Address Translator. It is needed for enabling Internet access to the nodes of the OMF-based wireless testbed, whose NICs are assigned private IP addresses.

The PL-Edge nodes are multi-homed PlanetLab nodes which can act as clients for the OMF wireless testbed.

Operation steps: example

In the following we list the sequence of the steps needed to execute an experiment using an OMF testbed at site S as access network for PlanetLab.

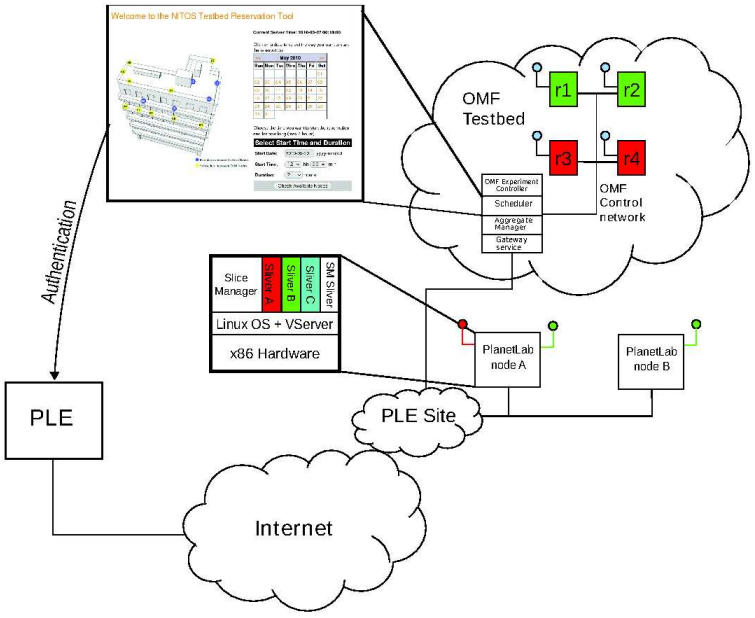


Figure 9.3: OMF-PlanetLab integrated architecture.

The experiment is going to be executed over a specific time interval

$$T = [T\_START, T\_END]$$

1. Then, the PlanetLab user U adds one or more PlanetLab Edge Nodes (PL-edge nodes) to his slice;

2. U logs into the Scheduler at site S and books the resources (nodes, channels, WiFi interfaces of PL-edge nodes nodes) he needs for his/her experiment over time interval T, providing the slice identifier. According to PlanetLab resource management scheme, resources are actually associated with slice rather than with the user that performed the reservation;

3. Within a time interval T, each slice's user is allowed to access the OMF EC (Experiment Controller) to perform experiments involving the assigned OMF resources.

The procedure for running isolated experiments on the OMF-based wireless

testbed is similar and just implies that no wireless interfaces belonging to the co-located PlanetLab nodes are reserved.

9.6 Chapter Conclusions

In this Chapter we presented an architecture designed to integrate local OMF-based wireless testbeds with the planetary-scale wired testbed, PlanetLab.

In Chapter 10 we exploit such architecture and we carry out experiments aimed at assessing the performance of the ALTO Service (see Chapter 7), proposed in the context of wireless mesh networks. The possibility of running experiments in such a hybrid experimental scenario highlighted several real-world issues, such as the impact on P2P systems of NAT traversal systems, that could only be reproduced thanks to our integrated environment.

Chapter 10

Implementation of an ALTO Agent in a Wireless Mesh Network

We have seen in Chapter 2 and throughout the thesis, how Wireless Mesh Networks (WMNs) have gained more and more attention as a way to provide Internet connectivity in both rural or urban areas. In Section 2.3 we showed WMNs main limitations, in particular the bandwidth availability, that comes mainly from sharing the wireless medium among several users and from external interference sources. Therefore, it is important for WMNs to optimize the spectrum resource usage.

In Chapter 6 we outlined how the applications based on the P2P file sharing paradigm show such limitations. We have seen that the main problem about Peer-to-Peer applications is that the overlay communication among endpoints is not aware of the underlying network configuration, which leads to an inefficient usage of resources. In Chapter 7 we showed the IETF Working Group ALTO initiative, which has introduced a mechanism to influence the peer nodes choice, hence giving the possibility to convey information about the network's preference to applications.

In particular, we will consider our solution to the Bittorrent case. As explored in Section 7.1, Bittorrent trackers are the software entities which store information about shared resources and nodes that share those resources. We called swarm the process of disseminating a resource among peers. A peer node that wants to join the swarm, contacts the Tracker, obtains a list of peer nodes and randomly

contacts the peers from uploading/downloading the desiderated resource. We have seen how a random peer selection may lead to several inefficiencies from an ISP point of view. In fact, this means that a node may choose to connect to a peer that is many hops away in the underlying network. This is a serious inefficiency if the same resource is available at a node closer to the asking peer. In particular, the Peer-to-Peer traffic may cross several AS and ISPs, drastically increasing the provider management costs.

In Wireless Mesh Networks, random peer selection has even worse consequences, given the limited amount of resources. It may happen that several peer nodes are all inside the same mesh network. In this case, the best choice would be to get the required content from the nodes inside the mesh network. Instead, with a random peer choice, worst case is that all peer nodes download from nodes outside the WMN. This may lead to an overloading of the wireless links that connect to the mesh gateways, and a consequent packet loss that may disrupt the normal operation in the WMN. Moreover, it may happen that the random choice leads to the selection of some mesh peers, but without taking into consideration connectivity performances between the peers, the choice may be suboptimal.

Exploiting a cross-layer approach, in this Chapter we introduced a strategy for P2P Bittorrent traffic optimization in WMNs. Our strategy is founded on the exchange of WMNs topology and peers performance information between application clients and underlying routing daemons.

We implemented an ALTO compliant service that takes a peer list from a Tracker as input, and returns the same nodes ranked with respect to the asking peer. Such information is passed on to the Bittorrent application by means of a Tracker server, purposely extended. This model, then, does not require any modification at the client side.

10.1 The ALTO architecture

In Section 7.2 we introduced the IETF ALTO solution to the Peer-to-Peer traffic optimization. The Application Layer Traffic Optimization architecture is a recent

IETF (Internet Engineering Task Force) Working Group, which has the objective of defining an architecture for allowing the optimization of distributed applications by providing them with information on the underlying network topology [1]. Distributed applications are comprised of nodes, e.g., peers in case of p2p applications, which exchange data by creating connections among themselves. Each node chooses a set of other nodes for creating such connections, but with no information about the underlying network topology, the choice of such nodes might be sub-optimal and lead to a waste of resources.

The ALTO architecture tries to provide such information. We showed in Section 7.2 that the main component of the architecture is the ALTO Server. The purpose of the ALTO Server is to reply to requests done by ALTO clients, for example P2P Bittorrent Clients, with the requested knowledge about the topology. In case of WMNs, the ALTO Service could help a peer located in the WMN to establish connections towards other nodes located in the same WMN. With the ALTO extension we propose, we try to reply to requests taking into account also the cost for reaching the mesh peers, returning the list of available peers in order of relative cost from the sender. This strategy reduced the amount of traffic that crossed the WMN border, and in addition it also reduced the download times, since the clients download from the best peers.

10.1.1 Network Address Translation (NAT)

In this Section we give a quick overview on the Network Address Translation mechanisms, often employed in the wireless mesh network context. The Network Address Translation operation consists in modifying the address field in the IP packet, changing the mesh node private address with the gateway public interface IP address. The NAT devices have been introduced to cope with the IPv4 addresses exhaustion. Thanks to the NAT devices, it is possible to use private IP addresses on internal networks that are behind a router that grants Internet access and that is equipped with a network interface configured with a single IP public address. The NAT operation, then, allows several clients to access the Internet services without consuming costly public addresses.

The translation operation is usually performed by a NAT server inside the WMN, installed at the gateway nodes. The NAT translation is required when the mesh nodes do not have public IP addresses. Therefore, their ”hidden” IP addresses have to be mapped into the single IP public address of the mesh gateway. All the outgoing packets, then, will appear as originated from the same mesh gateway. The NAT stores all the IP mappings in its translation tables. Then, on the reverse path, the packets are mapped back to the originating IP address using the information stored in the NAT tables.

When the traffic originates in the wireless mesh network, the NAT expedient works, since the device is able to build the translation tables: the mesh nodes may be able to access to a remote website on an outside server, but an outside client could not access to a website hosted in a server inside the private network. No server may be hosted on an internal node, then, since the NAT devices have no way to determine the internal node for which the incoming packets are destined.

Applications such as Peer-to-Peer file sharing, require users to carry out both client and server capabilities as well. Therefore, such applications represent a problem if users are inside WMNs, behind NAT devices, since incoming transmissions cannot be delivered to the proper internal node, and they cannot make it through the NAT.

Moreover, applications of this kind, often carry IP addresses and port numbers information inside the application data, hence requiring substitution and advanced translation techniques.

Despite having real benefits, the Network Address Translation has also serious drawbacks, that may prevent the operation of some applications. In Section 10.1.2, we showed how the ALTO Agent may help mesh nodes to solve the NAT traversal issues in the context of file sharing P2P applications.

10.1.2 The ALTO Agent

The main contribution of this Chapter is the definition of a strategy for the optimization of the tracker based P2P traffic and the definition of an ALTO-compliant architecture which acts accordingly.

We implemented an ALTO server and consistently with the ALTO concepts we conveyed to the ALTO server information from the perspective of different network regions. In particular, we implemented an ALTO WMN manager, called ALTO Agent, responsible for the peer ranking inside WMNs. In our model, the ALTO agent is implemented on a node inside the WMN, preferably on a mesh gateway. It has direct access, then, to the routing information and link performances of the WMN. Therefore, the ALTO Agent can properly rank peer nodes inside the WMN based on purely wireless metrics, taking into consideration interference and total utilization of the collision domains of wireless links. We may have several WMNs, each with a proper ALTO Agent installed. Then, each Agent may announce itself to the ALTO server, declaring itself and supplying information about the IP range it is responsible for. Moreover, since most WMNs are behind a NAT, the ALTO agent accomplishes the NAT address translation operations, too. In this Section we will show the integrated ALTO Service architecture.

The application entity that would require the ALTO service is the Bittorrent tracker server. The Bittorrent tracker server is contacted by Peer-to-Peer application clients in order to retrieve the list of peers sharing the same resource. The tracker, owing a list of the peer clients belonging to the same swarm, would provide the peer list to the ALTO server, together with the IP address of the asking peer. The network information that is returned by the ALTO server is the same peer list, reordered in decreasing order of priority, where the priority of a peer is its ranking with respect to the asking peer. Finally, the tracker would turn the ordered peers list to the asking peer. The ALTO server entity would require no modification to the application clients, since all the ALTO information exchange is managed tracker-side.

All the ALTO agents are responsible for a particular IP address range and they are contacted directly by the ALTO server.

We structured the ALTO service as follows. We set up a main ALTO server and several ALTO agents. When contacted by a tracker with a peers list, the ALTO server checks if some of the peers in the list can be handled by the available ALTO agents. To get to know that, the ALTO Server may simply check the IP addresses

with the IP range that is managed by the available ALTO Agents. If that is the case, then, the ALTO Server can ask the agent responsible for that IP range to order the peers located behind the WMN according to their distances from the asking peer. An agent has direct knowledge of the subnet it is responsible for and it can exploit fine-grained information when asked to rank a set of nodes. We installed an ALTO agent on a mesh router inside a multi-radio wireless mesh network and exploited the routing metric information directly available at mesh nodes. The agent was then able to rank nodes based on routing metric costs, thus exploiting local resource availability: if a mesh node was to contact the ALTO server, the corresponding mesh agent would return a set of nodes belonging to the same wireless mesh network of the asking node.

The architecture we propose is depicted in Figure 10.1.

We implemented the ALTO agent as a plugin for *Olsrd* [2], the well-known routing daemon for wireless mesh networks. This choice allowed us to get all the information we needed without the need for additional control messages, since the network topology information and the links performance metric are collected by the Olsrd daemon. The plugin has access to the topology, i.e, the nodes, the edges between them and the cost of the edges. From this knowledge, by applying the Dijkstra algorithm [89], the plugin is able to compute the best paths from the asking peer to all the co-located peers, to evaluate such paths, and then to sort them. Moreover, this choice gave us the opportunity to freely choose between several link cost metrics, for example between the one proposed in [90] or the one presented in Chapter 8, without any change to the architecture.

With the ALTO Agent implementation, we also try to cope with the problem of private addresses translation, which is very relevant in the WMN context, since the private internal wireless mesh network is often mapped into one single public IP address (see Section 10.1.1).

The connections that originate from the WMN private network for outside destinations are altered by the NAT so as to present to the outside destination the same, public, IP address. The NAT processing is completely transparent, and the mesh nodes are not aware of the IP translation.

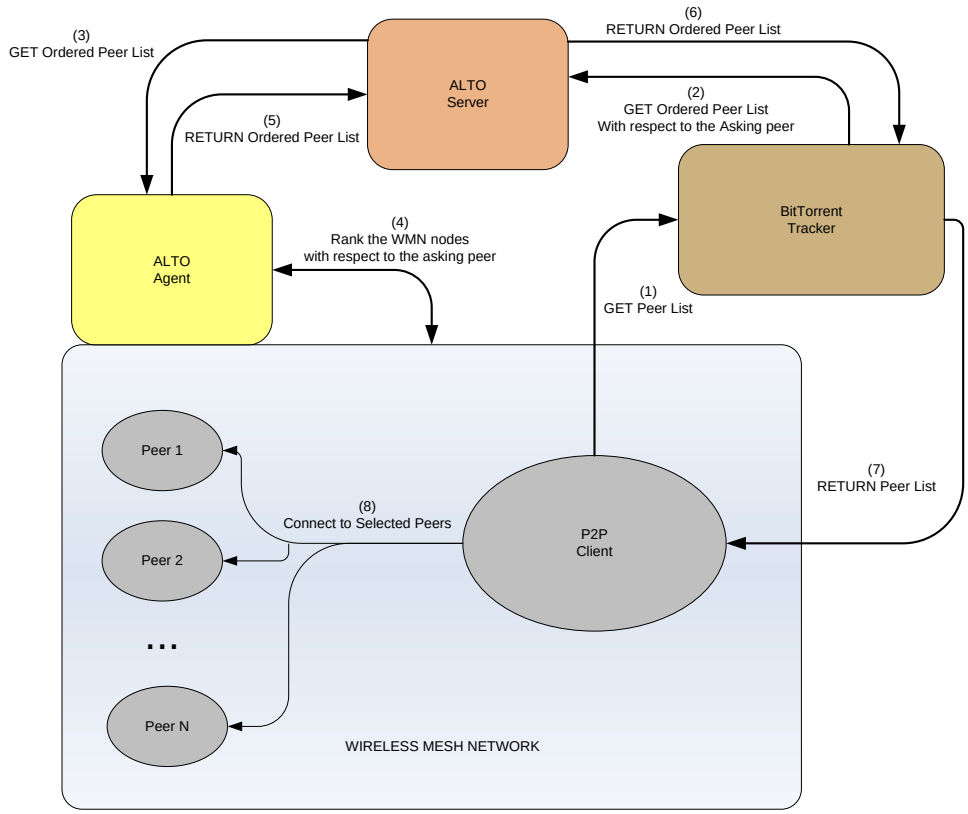


Figure 10.1: Architecture of the ALTO Service. **(1)** The peer node queries the Tracker for a list of peers; **(2)** The Tracker is configured to rely to the ALTO Server, providing the peer list to be ordered and the IP address of the asking Peer; **(3)** The ALTO Server verifies the Agent presence and delegates the mesh node ranking to the Agent; **(4)** The Agent rank the nodes and **(5)** return the ranked list to the Server; **(6)** The Server returns the peer list to the Tracker first, **(7)** to the peer node later; **(8)** Finally, the node establishes connections with the nodes on the optimized peer list.

Therefore, as the peers are seen by the Tracker with their public IP addresses, two peers located behind the same wireless mesh network will receive a list of public IP addresses. At best, the internal nodes will be able to connect to the mesh nodes inside the mesh network only through the gateway, i.e., only through a public to private address translation and vice versa. In fact, to the mesh peers it will appear that the peer to connect to has a public, external address, hence the next-hop towards the destination has to be the mesh gateway. The only entity that

would be able to understand what is going on is the NAT.

Finally, in the worst case, when contacted by the internal mesh node, the NAT translation table may not be complete, preventing the NAT to climb up to the proper internal peer.

We leveraged the ALTO architecture, in particular the ALTO Agent, for providing information about other nodes inside the same WMN along with the required private IP addresses.

A simple solution would be to set the NAT gateway to statically forward some public ports to the peers, so that peers can contact each other on their assigned public ports. Peers may always be visible with each other, but this strategy is not optimal, though, since it requires that all the internal Peer-to-Peer connections must go through the NAT. Therefore, we propose the following solution.

Upon receiving the peer list from the ALTO Server, the ALTO Agent would translate all the public addresses of the mesh peers into their relative local addresses. Then the list would be sorted and finally returned to the asking peer. The asking peer, in turn, receives the local addresses of the co-located peers and can connect to them directly on the best path, as enforced by the routing daemon.

In case the asking peer is located inside the WMN and an Agent has been setup, the first elements of the returned list are the co-located peers, ordered by their distance from the asking peer (if the asking peer is outside the mesh network, the mesh peers are ordered by their distance from the gateway node). The remainder of the list is made of the other peers of the swarm, ordered by their distance in terms of ASes (Autonomous Systems) hop count from the asking peer. By ordering the list in this way, we try to minimize the inter-ASes traffic, which is generally more expensive for the provider.

In case the request of the ALTO Server to the ALTO Agent fails, it means either the asking peer is not behind a WMN or the ALTO Agent is unavailable. In such a case, the peers are ordered by their distances in term of ASes from the asking peer. In particular, the modified Tracker replies to requests with lists in which the first peers belong to the same AS as that of the asking peer. The information about the AS of a peer is got by the tracker at runtime by a request to

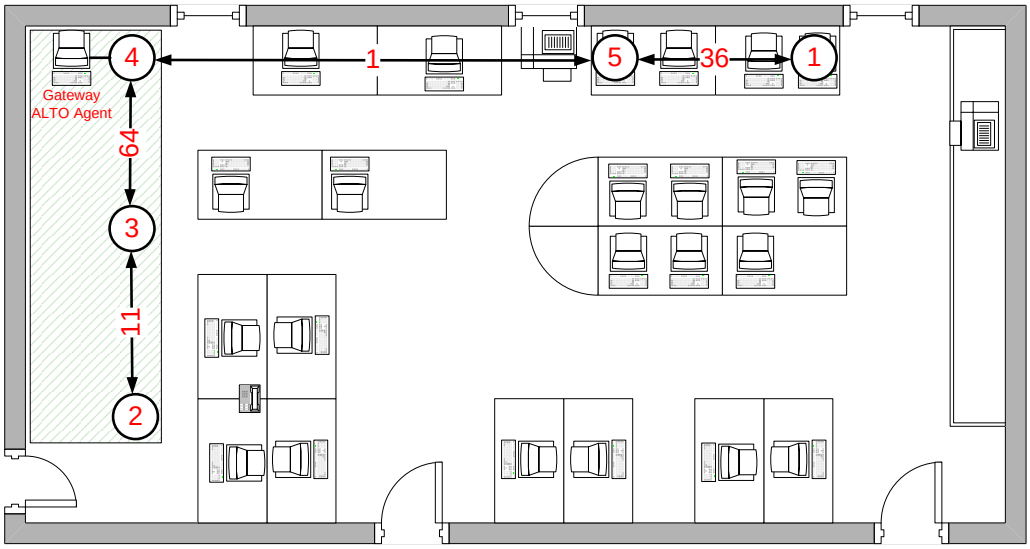


Figure 10.2: Wireless mesh network topology

the RIPE database [91]. This is coherent with our strategy, which is to keep traffic as local as possible to lower the cost for the ISP while optimizing application performance when possible, i.e. when the peers are behind the same WMN.

10.2 Experimental evaluation

The objective of the experiments we conducted in this Section is to show the proposed architecture at work and to verify the goodness of the optimization strategy we proposed.

We created a distributed setup for our experiment involving the use of a wireless mesh testbed, connected to Internet through a mesh gateway.

The tests were performed on the WILEE (WIreLEss Experimental), a wireless mesh network testbed, located at the University Federico II of Naples, Italy. It consists of:

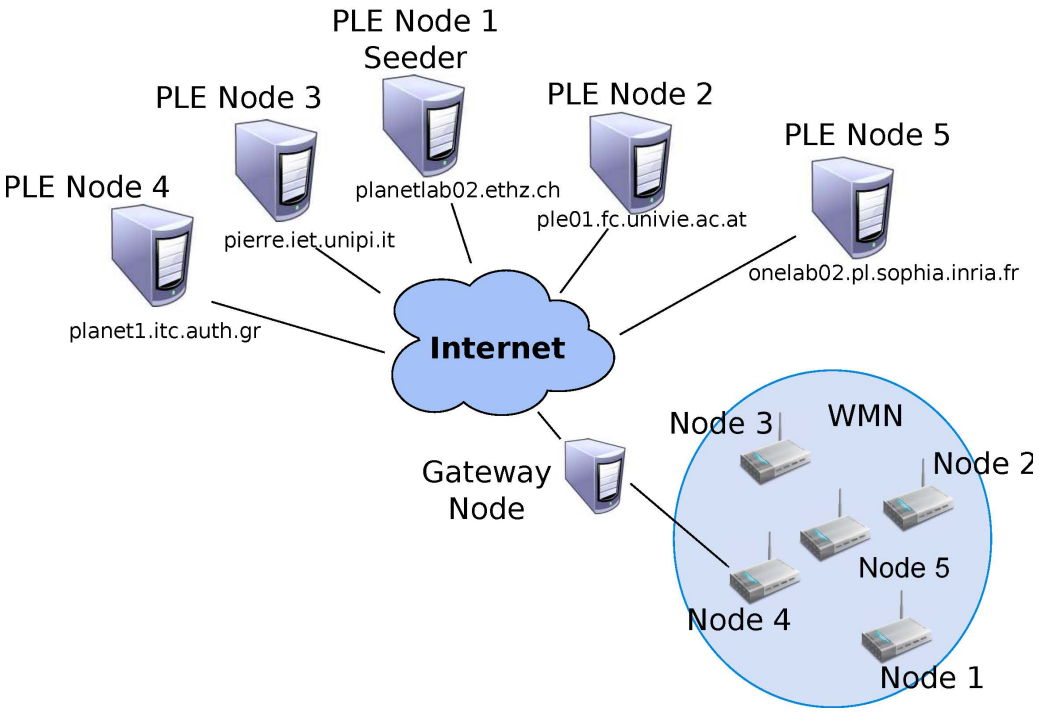


Figure 10.3: Experimental testbed setup

- 8 Netgear WG302Uv1 access points (5 of them will be used in the experiments) based on the network processor Intel XScale IXP422B at 266Mhz, with 32 Mbyte of DRAM memory and 16 Mbyte of flash disk. Each node is equipped with two 802.11a/g wireless cards. As operating system we employed OpenWrt [92], the well known Linux distribution for embedded devices;

- 1 Linux machine acting as gateway towards the Internet and hosting the *OMF Experiment Controller (OMF EC)*, the extended *NITOS Scheduler*, the *OMF Aggregate Manager*, described in Chapter 9, and the *ALTO Agent*;

First, we created a slice involving five PlanetLab Europe nodes; to such slice, we added five wireless nodes from the WILEE testbed by using the extended NITOS Scheduler.

The wireless nodes were configured exploiting the facility offered by OMF to form a WMN topology, and we took advantage of the mixed 802.11a/g orthogonal

channels availability to reduce interference. The wireless mesh network topology and radio interfaces configuration are shown in Figure 10.2. Node 4 is connected to the OMF Gateway, NAT server and ALTO Agent. The channel configuration is depicted in Figure 10.2. Since each node has both 802.11a and 802.11g compatible antennas, we were able to allocate 802.11a (channels 36 and 64) and 802.11g (channels 1 and 11) channels without losing connectivity.

Then, an open source Bittorrent client, *Transmission* [93], was installed on PlanetLab Europe and WILEE nodes.

The tests consisted of the execution of a complete file transfer between seeders, i.e., hosts with the file to spread, and few leechers, i.e., the hosts that download the file. One PlanetLab Europe node and one WILEE node were chosen as seeders of the Bittorrent swarm and the file shared was of approximately 50MB. The scenario of the experiments is illustrated in Figure 10.3.

Therefore, we ran two sets of experiments. For the first set we employed a standard Bittorrent Tracker, *Quash* [94]; for the second set we employed a modified version of Quash, which is able to exploit our proposed architecture.

The ALTO server and the ALTO Agent functionalities were those described in Section 10.1.2.

We focused on two kinds of performances. First, we underlined how the ALTO service deployment has the effect of confining the traffic within the mesh network (or in general within the AS). This is a fundamental requirement for cost reduction in network management. At the end of each experiment, i.e., when all the peers downloaded the file, we measured the traffic which had been either originated or destined to nodes located behind the OMF Gateway, i.e., the wireless nodes. Our objective was to demonstrate that the traffic crossing the WMN boundaries was minimized by using our modified tracker and ALTO Agent. Then, we underlined the increase in the overall performance perceived by the P2P application user. The introduction of the ALTO Agent allowed the WMN user to choose the best mesh nodes inside the mesh networks and this led to faster download times. Therefore, for each test we measured the time taken for all the mesh peers to download the file. Our objective is to show the download time saved, together with the amount

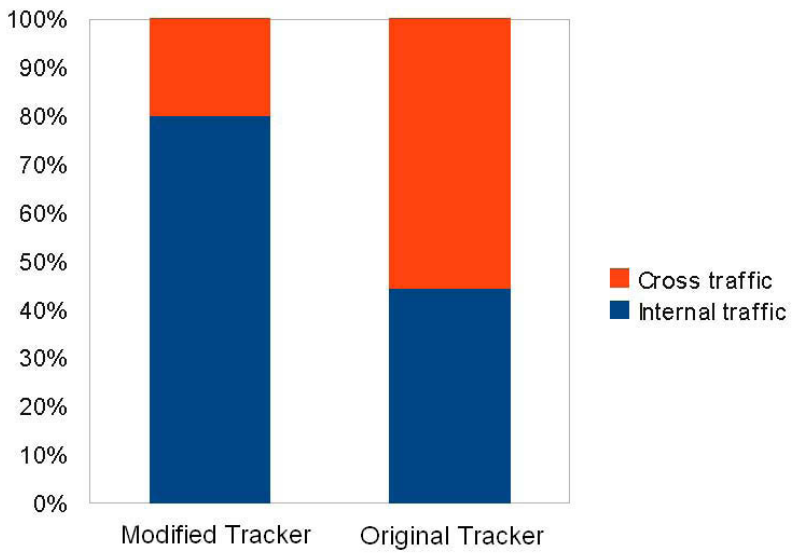


Figure 10.4: Percentage of exchanged cross traffic with and without the ALTO Service

of traffic which is kept local to the WMN.

In Figures 10.4 and 10.5, we report the results of two experiments, one where the standard tracker was deployed, the other with the modified version of the tracker. The results are averaged over 5 different simulations. The Figures show that the amount of Bittorrent traffic flowing through the OMF Gateway, i.e., the amount of traffic exchanged between the nodes inside the WMN and the external nodes, was significantly lower when the ALTO Service was exploited. In particular, the results show a reduction of about 66% of the overall amount of traffic crossing the WMN borders. This means that the ALTO Service significantly helps the network provider in localizing the traffic inside the WMN (or, similarly, inside the same Autonomous System). We remark here that the wireless mesh nodes were all configured with private IP addresses. Therefore, the ALTO Agent is responsible for the necessary translation operation of the private IP addresses inside the ranked peers list returned to the tracker.

In Figure 10.6, we show the download times, averaged on 5 repeated simulations, for the Bittorrent clients located inside the wireless mesh network. On

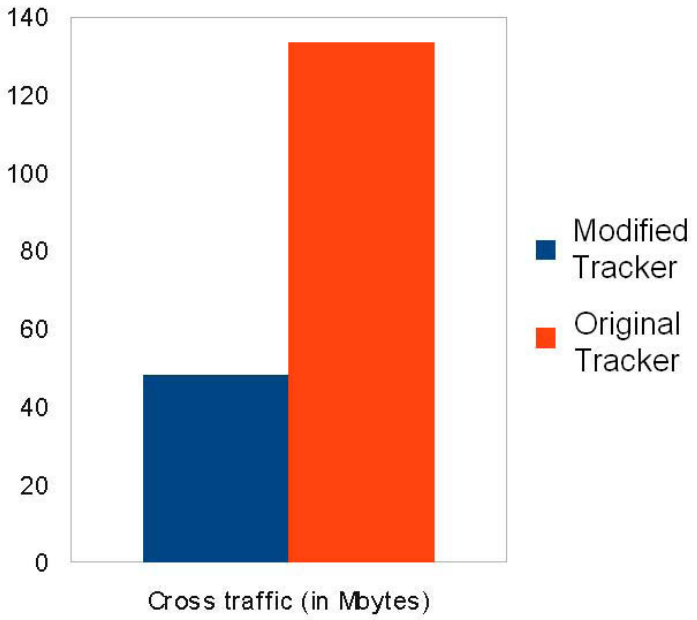


Figure 10.5: Amount of exchanged cross traffic with and without the ALTO Service

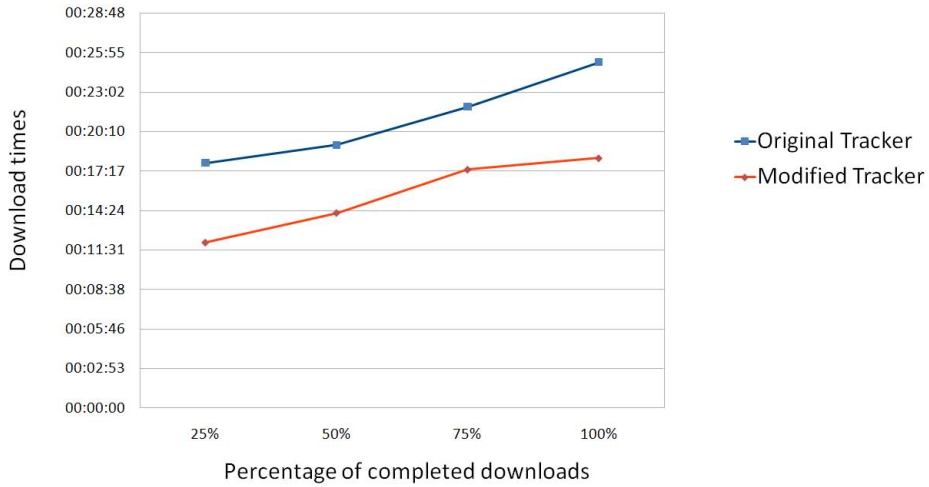


Figure 10.6: WMN nodes download times with and without the ALTO Service

the x-axis are represented the percentages of the nodes that have completed the download, while on the y-axis are represented the complete file download times. It results that the download times for the mesh clients are always shorter when the ALTO Service is invoked. In particular, we have a reduction in the download times of about 27%. This means that the ALTO Service not only ensures traffic locality, but, thanks to the mesh ranking capabilities of the ALTO Agent, is also able to improve Bittorrent performance, reducing download times.

10.3 Chapter Conclusions

In this Chapter we addressed the P2P traffic management problem in the context of Wireless Mesh Networks. A way to control P2P applications traffic is to control the choice of the connection endpoints in a P2P swarm, i.e., to influence the overlay topology creation step. We referred to tracker-based file sharing Peer-to-Peer applications and to the IETF Application Layer Traffic Optimization Working Group solutions. We exploited an ALTO compliant solution and extended it with an ALTO Agent. The ALTO Agent is deployed at wireless network access gateways, hence it is able to directly evaluate links performance and to point out the best routing path between two nodes. Therefore, the ALTO Agent is responsible for ranking the peers inside the mesh network and for helping the P2P applications to choose the best nodes inside the mesh network.

Finally, we implemented the ALTO Server and the wireless ALTO Agent and we deployed them in a realistic wireless mesh network testbed. Then, we showed the benefits of the ALTO architecture both for the P2P application end user and for the network operator, in terms of quicker download times and inter-network traffic volume reduction. In particular, the tests we performed showed a reduction of about 27% of download times and of 66% of the amount of traffic crossing the wireless mesh network boundaries.

Chapter 11

Conclusions

Wireless Mesh Network performances strongly depend on the proper wireless parameters configuration. The wireless medium is affected by environmental noise, interference problems and by shortage of spectrum availability. Moreover, multi-hop mesh networks are even more sensible to interference since the communication path is affected by both inter-path interference, due to transmissions on adjacent paths, and intra-path interference, due to transmissions on hops on the same path. We saw how the multi-radio interfaces availability and the IEEE 802.11a/g orthogonal channels standardization can help reducing the internal interferences. Hence, the problem of minimizing interference has become the problem of correctly assigning the available orthogonal channels to the radio interfaces.

In this Thesis, we looked into the traffic optimization issues that arise in the context of wireless mesh networks and we showed how the overall wireless mesh network configuration has to be necessarily adapted to the expected traffic flow rates. We started addressing the channel assignment problem, we showed how it is strictly connected to the traffic flows routing problem, and how we needed to solve the two problems jointly. The joint problem has been shown to be NP-complete and therefore we relied on approximate solutions. We presented a typical approach that consisted in obtaining the solution of the routing problem, i.e., the traffic flow rates, and then assigned the channels trying to ensure the scheduling condition for the nominal flows.

Therefore we assumed to know the optimal flows, solution of the routing prob-

lem, and focused on the problem of finding an assignment of channels to the available radios that ensured the attainability of such flows. This problem was again NP-complete and we proposed some heuristics.

In particular, we showed how typical wireless parameters affected the interference perceived by the links sharing the same transmission frequency and we found out how an accurate parameter configuration can help verifying the flow scheduling condition. Therefore, instead of considering the wireless mesh links in the same way as physical links, the proposed channel assignment algorithms tried to exploit the unique physical characteristics of wireless links, optimizing the choice of parameters such as transmission rate and transmission power on a link.

In this Thesis we studied two different approaches to the channel assignment problem. We presented FCPRA, a centralized channel assignment algorithm that takes into account the impact of transmission power and transmission data rate on the interference reduction in a multi-radio wireless mesh network.

Then, we observed that, since the channel assignment depends on the traffic flows on each link, a variation in these flow rates would lead to a new resulting channel assignment. Moreover, we outlined how frequent channel assignment re-computations are not recommended if the current channel configuration is not taken into consideration. In fact, in this case the channel assignment algorithm would return a channel configuration that may be completely different from the current one. The time required to reconfigure all the radios in the network may lead to packet losses and connections dropping. Therefore we presented MVCRA-R, a centralized rate and channel re-assignment algorithm that, starting from the current configuration of radio interfaces, tries to adapt the configuration of the channels to the new traffic flows as much as possible, by reducing the number of radio interfaces that will need to be reconfigured. The proposed algorithms, FCPRA and MVCRA-R, are both centralized algorithms that assume that the solution of the routing problem is given.

For an effective wireless mesh network configuration, knowledge of the exact amount of traffic that has to be routed on the wireless links is fundamental. Then, in Chapter 4, we presented an hybrid distributed channel assignment algorithm

that takes advantage of traffic information and tries to optimize both resource usage and application performance. We assumed that a generic application would provide information about the amount of traffic it is going to create and the destination nodes. Knowing the actual routing paths between the mesh end points, the channel assignment algorithm was then able to assign channels so that the least occupied channels were assigned to the busiest links. We implemented our solution by exploiting two different components. While the first component was responsible for keeping updated the traffic demand information among the mesh nodes, the second component was responsible for the actual channel assignment algorithm.

We had the chance of testing our proposed solution on the KAUMesh, the wireless mesh testbed of the Karlstad University, exploiting an experimental framework based on OLSR and Net-X. Thanks to the special features of such framework, we were able to implement such distributed channel assignment algorithm and to thoroughly test it.

After solving separately the channel assignment and routing problems, we ended up having a channel assigned to each radio and a set of traffic flows for all the links in the network. The set of flows would be the result of an optimization problem, and they are found satisfying a given objective function. We focused, then, on the problem of how to properly route packets in the network so to obtain on each link an average traffic flow as close as possible to the nominal one, returned by the solution of the optimization problem. Standard routing algorithms make routing decisions trying to minimize the overall path cost. AODV, one of the most commonly deployed routing algorithms, takes into consideration only unitary link costs, therefore the chosen path is always the minimum hop-count path.

Therefore, we proposed AODV-MLW, a modified version of the AODV routing protocol, which allowed to use arbitrary link costs. In particular, we computed the link costs so to obtain an average link utilization as close as possible to the optimal one. The problem of computing the link costs that result in the optimal link utilization, though, is non linear. Moreover, the routing operator that returns the

links utilization from the link costs, is not known in an explicit form. To determine all the link costs, then, we used a local search algorithm, Tabu Search. The algorithm started from an initial solution (a solution is represented by the costs vector), chosen arbitrarily, and created a set of possible solutions of the problem. Such set is called neighbourhood of the considered solution. For the Tabu Search algorithm implementation, we defined a fitness function, used to assess the quality of a solution vector, and we defined the rule to populate the neighbourhood of the current solution.

In conclusion, if we wanted to optimize the available resources, both in terms of proper channel assignment, transmission parameters configuration and optimal routing path choice, it is mandatory to take into consideration the exact traffic pattern. In fact, we tailored the wireless mesh network configuration on a particular traffic distribution, and we continuously tried to keep such configuration updated to the traffic variations. We deemed very precious the traffic knowledge: finding a way to control traffic would help the wireless network management and, of course, would also improve the overall network performance.

In particular, we outlined how recent traffic analysis revealed that an ever growing share of Internet traffic is created by P2P applications. Applications of this kind create overlay networks among the peer users, where the overlay topology is created without any knowledge of the underlying network.

In this Thesis we showed some of the inefficiencies that arise from the interaction of overlay networks with the physical networks they are built upon. One of the inefficiencies is that neighbour overlay nodes are many hops away in the physical network, even in terms of AS-distance.

This uncoordinated interaction is critical from the application user point of view, due to unstable performances. Moreover, the network provider is unable to apply its own traffic engineering mechanisms. Hence, it is hoped that some form of cooperation between the network manager and the P2P application may be enforced. We presented the IEEE ALTO Working Group proposal: the physical network provider would guide the overlay topology creation, by providing P2P clients with preferences on the available neighbour peers. The motivation behind

the ALTO Working Group is to convey to the upper layer information about the underlying physical network, from the provider point of view, hence allowing the P2P application to choose the closest available nodes. Typical information that the ALTO Service may be able to provide are nodes connectivity properties and nodes location.

In this Thesis we referred to the tracker based file sharing Bittorrent applications. Peers would contact the tracker, and they would obtain the list of the peer nodes belonging to the same swarm. Then, they would randomly select a subset of such peer nodes and they would start exchanging data with them. This means that a wireless mesh client may even choose to exchange data with nodes outside the wireless network, although there are available internal peers inside the mesh network, eventually overloading the wireless links towards the mesh gateways. Moreover, even if internal nodes were chosen, a random choice may result in a traffic pattern that could not be acceptable given the current network configuration.

In this Thesis, we defined a strategy for Bittorrent traffic optimization and we designed an ALTO-compatible architecture that implemented our strategy. In particular, we presented an ALTO wireless manager, the ALTO Agent, installed on a gateway node inside the mesh network, and responsible for the local peer mesh nodes. The Agent has direct access to routing information, wireless link performance and current wireless mesh network configuration.

The Agent would be able to rank the internal mesh nodes with respect to an asking peer, by estimating the end-to-end paths performances. After showing inadequacy of common link metrics in taking into account interference among the wireless links in a routing path, we introduced an interference aware path metric for wireless mesh networks. We proposed a metric that would take into account the interdependencies among the interfering wireless links, allowing to reduce the overall interference level within the chosen multi-hop wireless path, hence choosing the most efficient routes. In particular, we penalized a link by taking into account the total utilization of all the links in its collision domain.

The Agent, then, is able to properly rank the mesh peers based on typical wire-

less path performances. Moreover, since wireless mesh networks are often behind a NAT server, the Agent is also responsible for the address translation operations, allowing mesh nodes with private addresses to communicate with external or internal nodes, unhindered.

Finally, we focused on the problem of measuring the performances of the proposed solutions for wireless networks in a large-scale testbed. We presented the OMF technology and integrated an OMF-based wireless testbed into the planetary scale PlanetLab testbed. With this integration, it has been possible to run experiments that would extend both on the PlanetLab testbed and on the mesh nodes belonging to the OMF-based wireless testbed. Therefore, we were able to test our implementation of the ALTO Server and ALTO Agent in a realistic testbed scenario, thus allowing us to show the real improvements that our architecture led to.

Concluding, the wireless mesh networks represent a considerable step towards the complete replacement of wired infrastructure with wireless communications. Moreover, they are the big push towards the creation of a pervasive network that may allow user to connect to Internet anywhere, anytime. We saw throughout this Thesis, that wireless mesh networks optimization comes through interference reduction and ability to adapt the wireless configuration to the current traffic pattern. Current research in wireless mesh network performance optimization is valid and active, but there are still many questions unanswered and many challenges to face, especially considering the constantly evolving nature of Internet. With this thesis, then, we tried to contribute to the research in the field, proposing new algorithms, new techniques and solutions.

Bibliography

[1] A. A. IETF Working Group, "Application-layer traffic optimization (alto)," http://www.ietf.org/html.charters/alto-charter.html.

[2] T. Clausen and P. Jacquet, "Optimized link state routing protocol (olsr)," Tech. Rep., October 2003.

[3] C. Chereddi, P. Kyasanur, and N. H. Vaidya, "Net-x: a multichannel multi-interface wireless mesh implementation." *Mobile Computing and Communications Review*, vol. 11, no. 3, pp. 84–95, 2007. [Online]. Available: http://dblp.uni-trier.de/db/journals/sigmobile/sigmobile11.html#ChereddiKV07

[4] T. Rakotoarivelo, M. Ott, G. Jourjon, and I. Seskar, "Omf: A control and management framework for networking testbeds," *ACM SIGOPS Operating Systems Review,*, vol. 43, no. 4, pp. 54–59, 2009.

[5] B. Chun, D. Culler, T. Roscoe, A. Bavier, L. Peterson, M. Wawrzoniak, and M. Bowman, "Planetlab: An overlay testbed for broad-coverage services," *ACM SIGCOMM Computer Communication Review,*, vol. 33, no. 3, pp. 3–12, July 2003.

[6] S. Avallone, F. P. D'Elia, and G. Ventre, "A traffic-aware channel re-assignment algorithm for wireless mesh networks," in *Proc. of European Wireless*. IEEE, April 2010, pp. 683–688.

[7] S. Avallone, D. Pellegrino, P. Peruggini, F. P. D'Elia, and G. Ventre, "A new channel, power and rate assignment algorithm for multi-radio wireless mesh networks," in *Wireless Days, 2008. WD '08. 1st IFIP*, nov. 2008, pp. 1–5.

[8] ——, "A new channel, power and rate assignment algorithm for multi-radio wireless mesh networks," *Telecommunication Systems Journal*, vol. 51, no. 4, 2012.

[9] S. Avallone, F. P. D'Elia, and A. Kassler, "A traffic-aware channel and rate re-assignment algorithm for wireless mesh networks," *Submitted to Transactions on Mobile Computing*.

[10] M. C. Castro, P. Dely, A. J. Kassler, F. P. D'Elia, and S. Avallone, "Wintech '09," in *Proceedings of the 4th ACM international workshop on Experimental evaluation and characterization*. New York, NY, USA: ACM, 2009, pp. 79–80. [Online]. Available: http://doi.acm.org/10.1145/1614293.1614308

[11] S. Avallone, F. P. D'Elia, and G. Ventre, "Layer-2.5 routing in multi-radio wireless mesh networks," in *Sensor, Mesh and Ad Hoc Communications and Networks Workshops, 2009. SECON Workshops '09. 6th Annual IEEE Communications Society Conference on*, june 2009, pp. 1–6.

[12] S. Avallone, R. Canonico, and F. P. D'Elia, "Overlay/underlay routing issues in wireless mesh networks," in *M4D 2008 - 1st International Conference on M4D, Karlstad, Sweden, 11-12 Dec08*. University of Naples, 2008. [Online]. Available: http://urn.kb.se/resolve?urn=urn:nbn:se:kau:diva-3837

[13] ——, "Ottimizzazione del traffico p2p nelle wireless community network," in *Conferenza GARR - Network Humanitatis*, 2009.

[14] G. Di Stasi, R. Bifulco, F. P. D'Elia, S. Avallone, and R. Canonico, "Experimenting with p2p traffic optimization for wireless mesh networks in a federated omf-planetlab environment," in *IEEE WCNC 2011 Conference*, 2011.

[15] F. P. D'Elia, G. Di Stasi, S. Avallone, and R. Canonico, "Evaluation of the alto architecture for bittorrent traffic optimization in wireless mesh networks," in *Submitted to the 25th IFIP TC 7 Conference*, 2011.

[16] MIT, "Roofnet project page," http://pdos.csail.mit.edu/roofnet/doku.php.

[17] "Swisscomm mobile," http://www.swisscom.ch/res/mobile/.

[18] "Sunrise," http://www1.sunrise.ch/.

[19] "Orange," http://www.orange.com/.

[20] "Cablecom," http://www.cablecom.ch/.

[21] "Fon," http://www.fon.com/.

[22] "Ieee 802.11-2007, wireless lan medium access control (mac) and physical layer (phy) specifications, june 2007." June 2007.

[23] "Draft standard for information technology - telecommunications and information exchange between systems - lan/man specific requirements- part 11: Wireless medium access control (mac) and physical layer (phy) specifications: Amendment: Ess mesh networking, ieee," March 2007, unapproved draft P802.11s/D1.02, Mar. 2007.

[24] C. Perkins, E. Belding-Royer, and S. Das, "Ad hoc on-demand distance vector (aodv) routing," Tech. Rep., July 2003.

[25] M. Alicherry, R. Bhatia, and E. Li, "Joint channel assignment and routing for throughput optimization in multiradio wireless mesh networks," *IEEE Journal on Selected Areas in Communications*, vol. 24, no. 11, pp. 1960–1971, November 2006.

[26] S. Avallone, I. F. Akyildiz, and G. Ventre, "A channel and rate assignment algorithm and a layer-2.5 forwarding paradigm for multi-radio wireless mesh networks," *IEEE/ACM Transactions on Networking*, vol. 17, no. 1, pp. 267–280, February 2009.

[27] R. Draves, J. Padhye, and B. Zill, "Routing in multi-radio, multi-hop wireless mesh networks," in *Proc. of ACM Mobicom '04*, 2004, pp. 114–128.

[28] P. Kyasanur and N. H. Vaidya, "Routing and interface assignment in multi-channel multi-interface wireless networks," in *Proc. of IEEE WCNC*, vol. 4, 2005, pp. 2051–2056.

[29] J. Tang, G. Xue, and W. Zhang, "Interference-aware topology control and qos routing in multi-channel wireless mesh networks," in *Proc. of ACM MobiHoc*, 2005, pp. 68–77.

[30] M. K. Marina and S. R. Das, "A topology control approach for utilizing multiple channels in multi-radio wireless mesh networks," in *IEEE BroadNets*, vol. 1, 2005, pp. 381–390.

[31] A. Raniwala and T. Chiueh, "Architecture and algorithms for an ieee 802.11-based multi-channel wireless mesh network," in *Proc. of IEEE INFOCOM*, vol. 3, 2005, pp. 2223–2234.

[32] K. Xing, X. Cheng, L. Ma, and Q. Liang, "Superimposed code based channel assignment in multi-radio multi-channel wireless mesh networks," in *Proc. of ACM MobiCom*, 2007, pp. 15–26.

[33] A. P. Subramanian, H. Gupta, S. R. Das, and J. Cao, "Minimum interference channel assignment in multi-radio wireless mesh networks," *IEEE Transactions on Mobile Computing*, vol. 7, no. 12, pp. 1459–1473, December 2008.

[34] A. Raniwala, K. Gopalan, and T. Chiueh, "Centralized channel assignment and routing algorithms for multi-channel wireless mesh networks," *ACM Mobile Computing and Communications Review*, vol. 8, no. 2, pp. 50–65, April 2004.

[35] K. N. Ramachandran, E. M. Belding, K. C. Almeroth, and M. M. Buddhikot, "Interference-aware channel assignment in multi-radio wireless mesh networks," in *Proc. of IEEE INFOCOM*, 2006, pp. 1–12.

[36] H. Skalli, S. Ghosh, S. K. Das, L. Lenzini, and M. Conti, "Channel assignment strategies for multiradio wireless mesh networks: Issues and solutions," *IEEE Communications Magazine*, vol. 45, no. 11, pp. 86–95, November 2007.

[37] B. Bakhshi and S. Khorsandi, "A maximum fair bandwidth approach for channel assignment in wireless mesh networks," in *Proc. of WCNC*. IEEE, April 2008, pp. 2176–2181.

[38] A. H. M. Rad and V. W. S. Wong, "Joint logical topology design, interface assignment, channel allocation, and routing for multi-channel wireless mesh networks," *IEEE Transactions on Wireless Communications*, vol. 6, no. 12, pp. 4432–4440, December 2007.

[39] J. Crichigno, M.-Y. Wu, and W. Shu, "Protocols and architectures for channel assignment in wireless mesh networks," *Ad Hoc Networks*, vol. 6, no. 7, pp. 1051–1077, 2008.

[40] S. A. Makram, M. Gunes, A. Kchiche, and M. Krebs, "Dynamic channel assignment for wireless mesh networks using clustering," in *Proc. of ICN*, April 2008, pp. 539–544.

[41] A. Dhananjay, H. Zhang, J. Li, and L. Subramanian, "Practical, distributed channel assignment and routing in dual-radio mesh networks," in *Proc. of ACM SIGCOMM*, August 2009, pp. 99–110.

[42] X. Lin and S. Rasool, "A distributed joint channel-assignment, scheduling and routing algorithm for multi-channel ad-hoc wireless networks," in *Proc. of IEEE INFOCOM*. IEEE, May 2007, pp. 1118–1126.

[43] M. Kodialam and T. Nandagopal, "Characterizing the capacity region in multi-radio multi-channel wireless mesh networks," in *Proc. of ACM MobiCom*, 2005, pp. 73–87.

[44] A. H. Mohsenian Rad and V. W. S. Wong, "Joint optimal channel assignment and congestion control for multi-channel wireless mesh networks," in *Proc. of IEEE ICC*, vol. 5, 2006, pp. 1984–1989.

[45] M. Chiang, "Balancing transport and physical layers in wireless multihop networks: jointly optimal congestion control and power control," *IEEE Journal on Selected Areas in Communications*, vol. 23, no. 1, pp. 104–116, 2005.

[46] S. H. Low, "A duality model of tcp and queue management algorithms," *IEEE/ACM Transactions on Networking*, vol. 11, no. 4, pp. 525–536, 2003.

[47] A. A. Franklin, A. Balachandran, C. Murthy, and M. Marina, "Demand based state aware channel reconfiguration algorithm for multi-channel multi-radio wireless mesh networks," in *Proc. of CARMEN*. IEEE, 2010, pp. 1–6.

[48] J. Park and S. Sahni, "Power assignment for symmetric communication in-wireless sensor networks," in *Proc. of ISCC*. IEEE, June 2006, pp. 591–596.

[49] D. Qiao, S. Choi, and K. G. Shin, "Goodput analysis and link adaptation for ieee 802.11a wireless lans," *IEEE Transactions on Mobile Computing*, vol. 1, no. 4, pp. 278–292, 2002.

[50] E. Rozner, Y. Mehta, A. Akella, and L. Qiu, "Traffic-aware channel assignment in enterprise wireless lans," *2007 IEEE International Conference on Network Protocols*, no. 4, pp. 133–143, October 2007. [Online]. Available: http://ieeexplore.ieee.org/lpdocs/epic03/wrapper.htm?arnumber=4375844

[51] V. Raman and N. H. Vaidya, "Adjacent channel interference reduction in multichannel wireless networks using intelligent channel allocation," Tech. Rep., 2009.

[52] S. J. D. Couto, D. Aguayo, J. Bicket, and R. Morris, "A high-throughput path metric for multi-hop wireless routing," *Wireless Networks*, vol. 11, no. 4, pp. 419–434, 2005.

[53] L. T. Nguyen, R. Beuran, and Y. Shinoda, "A load-aware routing metric for wireless mesh networks," in *Proc. of IEEE ISCC*, July 2008, pp. 429–435.

[54] S. Waharte, B. Ishibashi, R. Boutaba, and D. Meddour, "Interference-aware routing metric for improved load balancing in wireless mesh networks," in *Proc. of IEEE ICC*, May 2008, pp. 2979–2983.

[55] G. Jakllari, S. Eidenbenz, N. Hengartner, S. V. Krishnamurthy, and M. Faloutsos, "Link positions matter: A noncommutative routing metric for wireless mesh network," in *Proc. of IEEE INFOCOM*, April 2008, pp. 744–752.

[56] E. Alotaibi and S. Roy, "A location-aware routing metric (alarm) for multi-hop, multi-channel wireless mesh networks," in *Proc. of IEEE WCNC*, 31 2008-April 3 2008, pp. 2081–2086.

[57] R. C. E. Rosen, A. Viswanathan, "Multiprotocol label switching architecture," http://www.ietf.org/rfc/rfc3031.txt.

[58] B. Fortz and M. Thorup, "Internet traffic engineering by optimizing ospf weights," in *Proc. of IEEE INFOCOM 2000*, vol. 2, 2000, pp. 519–528.

[59] A. Nakao, L. Peterson, and A. Bavier, "A routing underlay for overlay networks," in *SIGCOMM '03: Proceedings of the 2003 conference on Applications, technologies, architectures, and protocols for computer communications*. New York, NY, USA: ACM, 2003, pp. 11–18.

[60] S. Seetharaman, V. Hilt, M. Hofmann, and M. Ammar, "Preemptive strategies to improve routing performance of native and overlay layers," may. 2007, pp. 463–471.

[61] Y. Liu, H. Zhang, W. Gong, and D. Towsley, "On the interaction between overlay routing and underlay routing," vol. 4, mar. 2005, pp. 2543–2553.

[62] V. Aggarwal, A. Feldmann, and C. Scheideler, "Can isps and p2p systems co-operate for improved performance?" *ACM SIGCOMM Computer Communications Review (CCR)*, vol. 37, no. 3, pp. 29–40, July 2007.

[63] S. Seetharaman and M. Ammar, "On the interaction between dynamic routing in native and overlay layers," apr. 2006, pp. 1–12.

[64] H. Xie, Y. R. Yang, A. Krishnamurthy, Y. G. Liu, and A. Silberschatz, "P4p: provider portal for applications," in *SIGCOMM '08*, 2008.

[65] A. A. Hamra, C. Barakat, and T. Turletti, "Network coding for wireless mesh networks: A case study." in *WOWMOM*. IEEE Computer Society, 2006, pp. 103–114. [Online]. Available: http://dblp.uni-trier.de/db/conf/wowmom/wowmom2006.html#HamraBT06

[66] L. Galluccio, G. Morabito, S. Palazzo, M. Pellegrini, M. E. Renda, and P. Santi, "Georoy: A location-aware enhancement to viceroy peer-to-peer algorithm." *Computer Networks*, vol. 51, no. 8, pp. 1998–2014, 2007. [Online]. Available: http://dblp.uni-trier.de/db/journals/cn/cn51.html#GalluccioMPPRS07

[67] I. Stoica, R. Morris, D. Karger, F. Kaashoek, and H. Balakrishnan, "Chord: A scalable peer-to-peer lookup service for internet applications," *Computer Communication Review*, vol. 31, no. 4, pp. 149–160, oct 2001.

[68] S. Burresi, C. Canali, M. E. Renda, and P. Santi, "Meshchord: A location-aware, cross-layer specialization of chord for wireless mesh networks (concise contribution)." in *PerCom*. IEEE Computer Society, 2008, pp. 206–212. [Online]. Available: http://dblp.uni-trier.de/db/conf/percom/percom2008.html#BurresiCRS08

[69] D. S. J. D. Couto, D. Aguayo, J. Bicket, and R. Morris, "A high-throughput path metric for multi-hop wireless routing," *Wirel. Netw.*, vol. 11, no. 4, pp. 419–434, 2005.

[70] R. Draves, J. Padhye, and B. Zill, "Routing in multi-radio, multi-hop wireless mesh networks," in *MobiCom '04: Proceedings of the 10th annual international conference on Mobile computing and networking*. New York, NY, USA: ACM, 2004, pp. 114–128.

[71] ——, "Routing in multi-radio, multi-hop wireless mesh networks," in *MobiCom '04: Proceedings of the 10th annual international conference on Mobile computing and networking*. New York, NY, USA: ACM, 2004, pp. 114–128.

[72] Y. Yang, J. Wang, and R. Kravets, "Designing routing metrics for mesh networks," *Proceedings of the IEEE Workshop on Wireless Mesh Networks (WiMesh). IEEE Press*, 2005.

[73] ——, "Interference-aware load balancing for multihop wireless networks," Tech. Rep., 2005. [Online]. Available: http://www.cs.uiuc.edu/research/techreports.php?report=UIUCDCS-R-2005-2526

[74] I. F. Akyildiz, X. Wang, and W. Wang, "Wireless mesh networks: a survey," *Computer Networks,*, vol. 47, no. 4, pp. 445–487, 2005.

[75] B. Blywis, M. Günes, F. Juraschek, and J. Schiller, "Trends, advances, and challenges in testbed-based wireless mesh network research," *ACM/Springer Mobile Networks and Applications (MONET),*, vol. 15, no. 3, pp. 315–329, June 2010.

[76] C. Elliott, "Geni: Opening up new classes of experiments in global neworking," *IEEE Internet Computing,*, vol. 14, no. 1, pp. 39–42, 2010.

[77] A. Gavras, A. Karila, S. Fdida, M. May, and M. Potts, "Future internet research and experimentation: the fire initiative," *SIGCOMM Computer Communication Review,*, vol. 37, no. 3, pp. 89–92, 2007.

[78] "Network emulation testbed," http://www.emulab.net/.

[79] U. o. Utah and P. U. Proposal, "Statement of work: Exploring federation of testbeds with diverse models," Tech. Rep., 2008. [Online]. Available: http://www.geni.net/docs/dev_emu-plab-fed.pdf

[80] G. D. Stasi, S. Avallone, and R. Canonico, "Lecture notes of the institute for computer sciences, social informatics and telecommunications engineering," in *Proceedings of ICST QShine 2009*, vol. 22. Las Palmas de Gran Canaria (Spain): Springer, November 2009.

[81] R. Mahindra, G. Bhanage, G. Hadjichristofi, S. Ganu, P. Kamat, I. Seskar, and D. Raychaudhuri, "Integration of heterogeneous networking testbeds," in *Proceedings of TridentCom 2008*, Innsbruck (Austria), March 2008.

[82] N. N. I. T. Laboratory, "Nitos testbed scheduler," http://nitlab.inf.uth.gr/NITlab/index.php/scheduler.

[83] G. C. Hadjichristofi, A. Brender, M. Gruteser, R. Mahindra, and I. Seskar, "A wired-wireless testbed architecture for network layer experimentation based on orbit and vini," in *Proceedings of ACM WINTECH 2007*, Montréal, Québec (Canada), September 2007, pp. 83–90.

[84] A. Bavier, N. Feamster, M. Huang, L. Peterson, and J. Rexford, "In vini veritas: Realistic and controlled network experimentation," in *Proceedings of ACM SIGCOMM 2006*, Pisa (Italy), September 2006.

[85] L. Peterson, S. Muir, T. Roscoe, and A. Klingaman, "Planetlab architecture: An overview," Tech. Rep. PDN–06–031, May 2006.

[86] S. Soltesz, H. Pötzl, M. E. Fiuczynski, A. Bavier, and L. Peterson, "Container-based operating system virtualization: a scalable, high-performance alternative to hypervisors," *ACM SIGOPS Operating Systems Review,*, vol. 41, no. 3, pp. 275–287, 2007.

[87] A. Anadiotis, A. Apostolaras, D. Syrivelis, T. Korakis, L. Tassiulas, L. Rodriguez, I. Seskar, and M. Ott, "Towards maximizing wireless testbed uti-

lization using spectrum slicing," in *Proceedings of TridentCom 2010*, Berlin (Germany), May 2010.

[88] L. Paterson and T. Roscoe, "The design principles of planetlab," *ACM SIGOPS Operating Systems Review,*, vol. 40, no. 1, pp. 11–16, January 2006.

[89] E. W. Dijkstra, "A note on two problems in connexion with graphs," *Numerische mathematik*, vol. 1, no. 1, pp. 269–271, 1959.

[90] D. S. J. D. Couto, D. Aguayo, J. Bicket, and R. Morris, "A high-throughput path metric for multi-hop wireless routing," *Wireless Networks*, vol. 11, no. 4, pp. 419–434, 2005.

[91] "Ripe network coordination centre," http://www.ripe.net/.

[92] "Openwrt," http://openwrt.org/.

[93] "Transmission project," http://www.transmissionbt.com/.

[94] "Quash," http://code.google.com/p/quash/.