Budapest University of Technology and Economics
Department of Telecommunications and Media Informatics

RESOURCE OPTIMIZATION IN OPTICAL NETWORKS AND PEER-TO-PEER TRAFFIC IDENTIFICATION IN IP NETWORKS

Ph. D. Theses

Perényi Marcell

Research supervisors:
Dr. Tibor Cinkler, Dr. Sándor Molnár
Department of Telecommunication and Media Informatics
Budapest University of Technology and Economics

Budapest, Hungary
2009

Preface

My theses cover two different areas of telecommunications, namely *configuration and resource optimization of optical networks* and *traffic identification and analysis of Peer-to-peer (P2P) applications in IP networks*. These two fields of science – together with the progressing of multimedia applications and services – have played an important role in the development of Internet and computer networks in recent years.

The traffic of telecommunication networks and Internet has increased significantly in recent years. The unprecedented demand for bandwidth has to be handled from the network provider's side. The only way to cope with the rapid development and to solve the capacity issue on a long term is the application of *optical technology*. Optical transmission successfully solves the capacity issues, but the continuously changing traffic is still a challenge for the operators – especially on lower aggregation levels where the traffic is more variable. Even though the provisioning of *(dynamic) demands* is more problematic than the static configuration, dynamic networks have recently attracted more attention.

In my theses, *I introduce a fast, generic, statistical utilization based method for joint dimensioning of grooming capability and the number of wavelengths* for dynamic demands. The method does not depend on the routing algorithm, the network topology, the protection scheme, and the traffic load.

Considering physical effects in the provisioning, configuration and routing of optical networks is a popular research area. The evolution of optical networks seems to tend towards a *fully reconfigurable network* where the control and the management plane (CP and MP) have new functions, such as determining the signal quality, tuning the wavelength frequency, setting dispersion compensation units, and – by using variable optical attenuators – setting the channel powers. Traditional functions, such as Routing and Wavelength Assignment (RWA), will naturally remain the main function of the CP and MP.

I have given the exact Integer Linear Programming (ILP) formulation of the joint optimization problem of RWA and determining of signal powers for static demands. My patented method performs routing in the optical layer and adjusts channel powers in an optimal way while observing physical effects. The other, more complex method performs routing in the optical and electronic layer jointly and supports 3R signal regeneration, wavelength conversion and grooming as well. Consequently, it optimizes three factors jointly: routing in the electronic and in the optical layers as well as determining signal powers.

Currently, optical networking is more dominant in transport and backbone networks. However, a clear process of bringing the optical termination closer and closer to the end-user (Fiber to the Curb/Building/Home, FTTC/FTTB/FTTH) can be noticed. In the near future, many broadband customers will have direct optical access. Optical connection does not only serve as a high-speed Internet access, but it also allows the receiving *High Definition TV (HDTV) channels*, replacing the traditional cable TV services. Optical access can satisfy both needs at the same time by providing high bandwidths and by having much spare bandwidth for potential further utilization.

Another way to ease the traffic load of the networks is to apply *multicast delivery*. Despite its bandwidth saving and the many possible applications (e.g., high definition TV), the multicast service is currently not directly available to the end users. However, it is an essential feature in the core of the transport network.

I have introduced a *novel ILP formulation for multicast routing* in multi-layer optical networks. I have shown the cost efficiency of *optical layer multicasting over electronic layer multicasting*, and how the cost of components, traffic load and grooming ratio affect the gain. I have also shown which parameters influence the benefits of *reconfiguration of dynamic multicast trees* and the how they influence it. I have determined the optimal length of the reconfiguration period.

Applications based on the P2P principle generate the major part of Internet traffic and likely also a significant part of the unidentifiable traffic. Therefore P2P traffic is one of the main factors responsible for an unprecedented demand for bandwidth in telecommunication networks. P2P applications are also involved in the distribution of illegal content (e.g., music, video and commercial applications), causing loss for the authors and publishers, who are against this kind of usage.

As an aftermath, recent popular P2P applications deploy various techniques to hide their presence and traffic. Traditional port based and pattern based identification methods can often not be used anymore.

I have introduced novel flow dynamics based identification methods to identify P2P traffic. I have constructed an effective and compound *heuristic method for the detection of general P2P traffic*. The method relies on the recognized robust and inherent characteristics of the traffic generated by major P2P applications. It is able to separate P2P and non-P2P traffic flows. The accuracy of the method was validated on a test dataset. Applying the method on real-life, large scale traffic traces, I investigated and compared P2P and non-P2P traffic at packet level, flow level, and aggregate level, focusing on the similarities and differences between the two traffic types.

P2P technology is also intensely used for providing *communication services*, including "presence service", chat, Internet telephony (Voice over IP, VoIP), video conferencing, and file transfer. Many of these services

offer a real alternative to traditional telephony services provided by landline and mobile phone operators. To avoid competition, operators are interested in regulating, controlling, filtering or even blocking P2P traffic origination from such sources.

My theses include methods for the identification of Skype traffic (a specific P2P application). The methods are based on typical characteristics not applied so far in flow dynamics based identification. Three methods (related to each other) were proposed to recognize different components of Skype traffic with the final aim of detecting Skype voice calls. The methods were validated and applied on real traffic traces. The comprehensive traffic analysis revealed unique properties of Skype traffic at flow level and aggregate level. The results show that Skype has a unique daily profile, that the duration of Skype calls has an exponential-like distribution, and that characteristic properties of Skype traffic are very similar in a fixed network and in 3G mobile networks.

The rest of the Dissertation is organized as follows. The dissertation consists of two main parts (PART 1 and PART 2). PART 1 introduces the results achieved in the *field of optical networking*, while PART 2 presents the results stemming from the *field of traffic identification and analysis*. Each part starts with a short introduction and also explains the general considerations related to all investigated problems in that field. The problem statement and related articles are presented in the specific chapter dealing with the problem.

Chapter 1 – as the first chapter of PART 1 – gives an introduction to optical networks. It describes the relevant technologies, protocols, and the applied simulation and modeling techniques (detailed models are shown in the specific chapters). Chapter 2 is dealing with the dimensioning of optical network resources (grooming capability and wavelengths) and introduces a novel optimization method. My new ILP formulations, which take into account physical effects in the joint optimization of routing and signal levels, are described in Chapter 3. Chapter 4 is dealing with static multicast routing. It presents a novel ILP formulation for the routing of unicast and multicast demands. Chapter 4 also compares the cost efficiency of optical layer branching and electronic layer branching of multicast trees. Chapter 5 studies the reconfiguration of dynamic multicast trees from several aspects.

Chapter 6 – as the first chapter of PART 2 – gives an introduction to traffic identification and analysis. Chapter 7 describes the traffic measurements used for verification and traffic analysis purposes. Chapter 8 and Chapter 9 are dealing with flow dynamics based traffic identification. Chapter 8 proposes a heuristic method for the identification of general P2P traffic and presents the results of a comprehensive traffic analysis. Chapter 9 contains the method constructed for the identification of Skype traffic and traffic analysis results (stemming from fix and mobile environments).

Chapter 10 concludes the Dissertation by describing the significant achievements.

Finally, the APPENDIX (Chapter 11) includes the detailed documentation of my new optical network simulator that was used for investigating most of the problems related to optical networks.

Acknowledgements

I would like to thank my supervisors; Tibor Cinkler, whose help in the field of optical networking and support from the last years of the M.Sc. study were essential in becoming a researcher, and Sándor Molnár, who taught me the mathematical background of traffic identification and analysis. Their assistance and advices, all the way through the Ph.D. study years, were essential.

My work was done in a research cooperation framework between Ericsson Hungary and High Speed Network Laboratory (HSNLab) at the Budapest University of Technology and Economics. Their financial support allowed me to concentrate on work and to present my results at international conferences.

Thanks to P. Varga and L. Kovács, Magyar Telekom and Ericsson Hungary for their help in the traffic measurements.

Thanks to István Maricza for the reviewing and improving the quality of many of my publications.

I am grateful to my friends and colleges at the Department of Telecommunication and Media Informatics for their invaluable help and comments during the years.

Naturally, I appreciate the psychical and mental support of my parents, József Perényi and Anikó Ékes, my brothers, Kristóf and Zsolt, and all other members of my family.

Last but not least, I wish to thank my faithful friends and those persons, who were the closest to me, for all the fun we had together and which was essential for productive work.

Budapest, Hungary
May 13, 2009

Marcell Perényi

PART 1
Resource Optimization in Optical Networks

1. Introduction to Optical Networks

The discovery that an appropriately doped, thin strand of silica acts as a waveguide (Section 8.2 of [1]) in a certain domain of the electromagnetic spectrum, has had a fundamental impact on digital communications in the late 1970s. Since silica (SiO_2) is available in abundant quantities (as opposed to materials used in electric transmission lines) and efficient means of fiber manufacturing are available, the potential cost of the optical fiber is low. In addition, fiber-optic devices outperform conventional transmission media, like electronic cables or radio waves, in terms of reliability and data transfer rate. Signals through optical fibers are transmitted using electromagnetic waves of wavelengths between 1260 and 1625 nanometers (the attenuation of the transmitted signal is the lowest in this range in modern fibers, see Fig. 1). The propagation characteristics of this domain in lightguides allow the very efficient shielding of the transported signal from external sources of interference; conversely, the interference caused by the signal itself (including crosstalk between fibers) is negligible as well. These advantages allow a favorable bit-error rate (as low as 10^{-12} in practical applications) for data transfer. The useful bandwidth of a single strand of fiber can be measured in tens of THz, which is significantly higher than that of most other media. As a consequence, it is fairly sure that optical communications will be a dominant technology in the upcoming decades.

1.1. Optical Technology

This subsection gives an overview of the physical phenomena, technologies and principles applied in optical networking.

1.1.1. Physical Medium and Light Sources

Taking full advantage of the bandwidth provided by silica fibers poses numerous challenges. Allocating the whole bandwidth to a single digital data stream would require a bit transfer rate of about 50 terabits per second, which is currently too fast for semiconductor switching devices to handle. Transmitters used to produce the optical carriers are mostly built using Light Emitting Diodes (LEDs), Edge-Emitting Semiconductor Laser Diodes or Vertical Cavity Surface-Emitting Lasers (VCSELs) (Chapter 2 of [2], Chapter 5 of [3], Chapter 3 of [4]). LED devices have large rise times that limit their modulation response bandwidth to less than 200 MHz; this value increases to a few GHz for edge-emitting semiconductor lasers and near 50 GHz for VCSELs. These modulation rates are, at best, a thousand times smaller than the bandwidth of the optic fiber; this implies that electronic TDM (Time-Division Multiplexing) methods cannot be used to fully exploit the available resources and some form of multiplexing is required in the optical domain. The most promising alternative is a combination of OTDM (Optical Time-Division Multiplexing) and WDM (Wavelength-Division Multiplexing) technologies (Section 8.1 of [1], Chapter 19 of [5], Chapter 2 of [6]).

The propagation of light inside the fiber is a complex process, but in outline light waves travel along the core of the fiber and reflect from the boundary of the cladding. The cladding is designed to have a higher refractive index, which allows the light waves to be guided through the core. Early *multimode fibers* have a relatively thick core (50-100 μm), which allows several modes of propagation. Each mode corresponds to a certain distribution of electric field gradient regarding the cross-section of the fiber. These fibers are limited in distance (few kilometers) and bit rate (up to 20 Mbit/s), since different modes propagate at different speeds leading to *multimodal dispersion*. *Single mode fiber* removes multimodal dispersion by allowing only one propagation mode, since the core width is around 8-10 μm, a small multiple of the wavelength of the signal.

The two major causes of signal loss through fiber are material absorption and Rayleigh scattering. The effects of these can be seen in Fig. 1. The local minima correspond to different frequency bands used in optical networks. The peaks between bands are mostly caused by absorption by water vapor in the fiber; this has been reduced over the last few years by the use of newer types of commercial fiber. The minimum point is around 0.25 dB/km of loss, which enables a distance traveled of around 100 km before the signal to noise ratio drops too low.

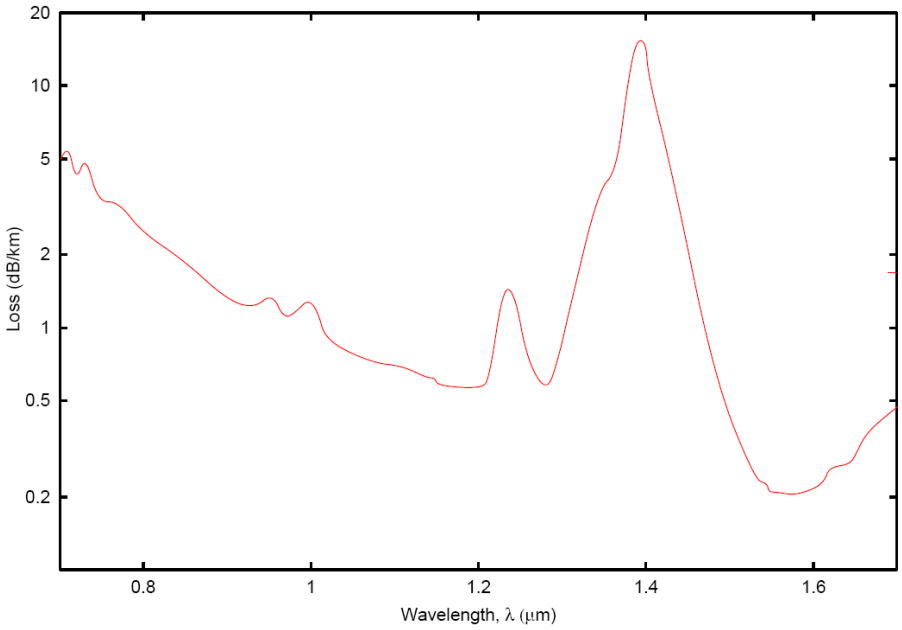

Fig. 1. Attenuation of optical fiber as a function of the used wavelength

Historically, the first window ranged between 800-900 nm; but since losses are high in this region, it is mostly used for short-distance communications. The second window is around 1300 nm, and has much lower attenuation. The region has zero dispersion. The third window is around 1500 nm, and is the most widely used. This region has the lowest attenuation losses and hence it achieves the longest range. However, it has some dispersion, and dispersion compensators are used to remove this.

Table I. Transmission windows used in optical fibers

Band	Description	Wavelength Range
O band	original	1260 to 1360 nm
E band	extended	1360 to 1460 nm
S band	short wavelength	1460 to 1530 nm
C band	conventional ("erbium window")	1530 to 1565 nm
L band	long wavelength	1565 to 1625 nm
U band	ultra long wavelengths	1625 to 1675 nm

Other factors also affect the transmission of data through single mode fiber. As fibers are typically not entirely cylindrical, they are slightly birefringent. The propagation speed depends on the polarization of the wave. Since a light-wave consists of two orthogonally polarized modes, these will propagate at different speeds leading to *Polarization-Mode Dispersion (PMD)*. *Chromatic dispersion* occurs since the propagation time is frequency dependent, and some optical pulses are chirped; the exact frequency changes slightly with time. This means that pulses can broaden, shorten, and even be reversed over time. These effects can be controlled and exploited by dispersion compensating fiber, which changes the refractive index of the fiber to create enough waveguide dispersion to compensate for chromatic dispersion at a given wavelength.

Non-linear factors may also cause significant distortion especially at higher bit rates. *Stimulated Brillouin scattering (SBS)* (Section 8.3.4 of [4]) occurs, when the power of the signal exceeds a certain level and produces acoustic vibrations in the transfer medium. The beam may undergo Brillouin scattering from these vibrations, usually in opposite direction to the incoming beam. As a consequence, the reflective index of the medium increases suddenly.

Stimulated Raman Scattering (SRS) (Section 8.3.3 of [4]) is a similar phenomenon – it modifies the frequency spectrum of the signal by strengthening higher frequencies and weakening lower frequencies.

Self-phase modulation (SPM) (Section 8.3.5 of [4]) is a consequence of the fact that the refractive index of the medium is influenced by the intensity of the signal. Such an intensity change can occur, for example, at the rising edge of the signal and deforms the shape of it.

With the correct choice of fiber properties, the current available long haul DWDM systems typically use channel spacing of 50 GHz and up to 128 wavelengths at 10 Gbit/s spanning 4,000 km before full regeneration is required. A full discussion of light propagation in optical fibers is covered in Chapter 2 of [7].

To achieve the quoted spans between full regeneration, optical amplification is necessary. This enables amplifying the optical signal without having to convert it into electrical form, preferably over a wide range of frequencies used by a DWDM system, and with a high-output flat gain spectrum, i.e. it is wished to avoid amplifying some signals more than others.

Erbium-Doped Fiber Amplifier (EDFAs) (Section 6.4 of [4]) is one of the most common types of amplifiers used in DWDM systems (other types with less favorable characteristics include Semiconductor Optical Amplifiers (Section 6.2 of [4]) and Raman Amplifiers (Section 6.3 of [4])). Their bandwidth is wide enough for multi-channel amplification (Section 6.4.5 of [4]); the ultimate limiting factor for applications is the amplifier noise (Amplified Spontaneous Emission) (Section 6.4.4 of [4]).

1.1.2. Multiplexing and Demultiplexing

WDM ([6], Chapters 1 and 16 of [8]) is basically an analogy of Frequency Division Multiplexing in the infra-red domain. Independent streams of data are modulated using different frequencies and sent through the same piece of fiber. At the receiver, several parallel frequency sensitive filters can be used to separate (demultiplex) the signals from each other. Demultiplexing can be done by *diffraction-based demultiplexers* or *interference-based demultiplexers* (Section 8.2.2 of [4]), but probably the most often used approach is the *Arrayed Waveguide Grating* (Sections 8.2.2 and 8.2.5 of [4]).

Literature distinguishes *dense* and *coarse WDM* technologies depending on the spacing of the wavelength channels: the Dense WDM (DWDM) recommendation described in ITU-T G694.1 [9] supports channel spacing values of 100, 50, 25 and 12.5 GHz with, respectively, allowing approximately 115, 229, 458 and 916 channels in the C and L bands (see Table I) [45, Section 19.3.3]. The Coarse WDM (CWDM) scheme (ITU-T G694.2 [10]) defines 18 channels with channels spacing larger than 2 THz in the spectral bands designated by the letters O, E, S, C and L.

Multiplexing and demultiplexing optical signals is an essential component in WDM networks; optical signals need to be split and joined depending on the frequency of the signal. There are two main types of demultiplexer, active and passive. Passive types, such as stimax gratings [11] or arrayed waveguide gratings [13], can typically be also used as multiplexers. Acoustically tunable filters [12], an example of an active demultiplexer, have the added ability to dynamically select multiple wavelengths but perform in an inferior manner to passive components on the two main goals: to minimize loss of the pass-band frequencies and to minimize signal from rejected bands. Other goals include thermal stability and flat pass-band stability to cope with slight changes in actual wavelength frequency switching capabilities.

Section 1.1.1 stated that the noise caused by crosstalk between individual optical fibers is negligible. Unfortunately, the same is not true for wavelength channels in the same fiber. Several phenomena are responsible for the significant amount of crosstalk possible in WDM systems:

- *Heterowavelength linear crosstalk* (Section 8.3.1 of [4]) is caused by the imperfect characteristics of optical filters, i.e. other wavelengths also pass through the filter besides the target wavelength.
- *Homowavelength linear crosstalk* (Section 8.3.2 of [4]) can occur due to multiplexer and demultiplexer leakage or to internal switch multipath interaction.
- *Nonlinear Raman crosstalk* (Section 8.3.3 of [4]) has the same origin as SRS.
- *Cross-phase modulation (XPM)* (Section 8.3.5 of [4]) is similar to SPM, but in this case intensity change of other wavelength channels modifies slightly the refractive index of the transfer medium.
- The term *four-wave mixing (FWM)* (Section 8.3.6 of [4]) refers to the phenomenon when the coexistence of multiple wavelengths in the same fiber creates also new wavelengths to appear. The effect of FWM can be decreased by using uneven channel spacing or deliberately allowing some dispersion.

1.1.3. Switching

In the context of digital communications, the term *switching* refers to the controlling of network elements in such a way that certain signals transported in the network are routed and forwarded from their source towards their intended destination. Three basic types of switching are distinguished according to the type and format of the routed signals.

In *packet switching*, transmitted information in the network is organized into packets comprising client payload and header. Network devices may process headers of packets in order to be able to forward them correctly.

Circuit switched networks create a continuous circuit between two endpoints. Information sent along the circuit will not be altered by the network, except by the physical transmission impairments caused by the physical properties of the transmission media. This might mean the end-to-end preservation of the analog waveform (resulting in a system providing fully *"transparent"* connections), but it can also refer to the preservation of the digital logic value of the signal only (resulting a *"transcendent"* or *"opaque"* system).

Former system may use their network resources more efficiently. Transmitted information is not processed by network elements along the way, switching devices are configured by the control plane at the setup phase of a connection.

Burst switching can be regarded as an intermediate solution between packet and circuit switching. It switches very long streams of bits.

Packet or burst switching is beneficial, if the traffic is changing rapidly and unpredictably. This assumption is right, if we are close to the access network. On the other hand, as we are leaving the access network towards the backbone, demands (which are, in fact, an aggregation of access network traffic) become more stable with lower variance. In such a case it is more sensible and economical – especially because packet switching networks are premature and expensive in their current state – to set up preconfigured high-capacity connections (pipes) between endpoints of the transport network. In my dissertation I always assume a transport (backbone) network and circuit switching.

It is worth to investigate switching functionality more specifically inside optical cross-connect (OXC) devices of the optical network.

There are several ways to implement an OXC. It can be realized in the electronic domain: all the input optical signals are converted into electronic signals after they are demultiplexed (i.e. wavelengths are separated). The electronic signals are then switched by an electronic switch module (electronic backplane). Finally the switched electronic signals are converted back into optical signals by using them to modulate lasers and then the resulting optical signals are multiplexed by optical multiplexers onto outlet optical fibers. This is known as an "OEO" (*Optical-Electrical-Optical*) design. Whilst this is a flexible approach, it has a key limitation: the electronic circuits limit the maximum bandwidth of the signal. Such architecture prevents an OXC from performing with the same speed as an all-optical cross-connect, it is not transparent to the network protocols used, and power consumption (generating heat by dissipation) can be also critical. On the other hand, it is easy to monitor signal quality in an OEO device, since everything is converted back to the electronic format at the switch node. An additional advantage is that the optical signals are 3R regenerated, so they leave the node free of dispersion and attenuation. An electronic OXC is also called an *opaque OXC*.

The second approach to implement an OXC is to perform switching in an all-optical way. *Photonic cross-connects* (PXC, or *transparent OXC*) switch optical signals without converting them to the electrical domain. Specifically, optical signals are first demultiplexed, and then the separated wavelengths are switched by optical switch modules (optical backplane). After switching, the optical signals are multiplexed again onto output fibers. Such switch architecture offers fully transparent operation in terms of data rate and framing protocol. However, it does not allow easy monitoring and assuring of optical signal quality.

In the era of dense WDM, large numbers of fibers, and the spreading of mesh topologies, the logical size of the switch is critical. Large switches, containing thousands of input and output ports, can be created by several technologies (also by some that are experimental), including Micro-Electro-Mechanical System (*MEMS*) switches, *liquid crystal* (LC) technology or by *cascading smaller switches* in some configuration.

The Beneš architecture [14] follows the second approach by combining 2x2 switches to realize an *nxn* switch. The main drawback of the technique is that the signal power can decrease significantly due to the long chain of small switches.

The MEMS devices used in optical switches are arrays of tiny mirrors fabricated in silicon and controlled to steer lightpaths around the switch [15]. They can be built either in a flat configuration with one fixed axis, or with two adjustable axes (3D MEMS). These devices are currently capable of switching hundreds of ports; they are non-blocking, and some have switching speeds of around 10 ms [16][17]. However, large MEMS switches are very complex devices, requiring a large amount of electronic control circuit, and have yet to be demonstrated in a large scale deployment environment.

The phase and optical properties of LCs can be changed by adjusting the temperature of the surrounding environment or the electric field. Switching functionality can be achieved by setting the LC molecules in different orientation.

The so called *translucent OXC* should be regarded as a compromise between opaque and transparent cross-connects. In this architecture the switch stage contains an optical switch module and an electronic switch module. Thus optical signals can be switched either by the optical or the electronic backplane. Usually the optical backplane is preferred for the purpose of transparency. When the optical switch module's switching interfaces are all busy or an optical signal needs signal regeneration, wavelength conversion, or grooming via OEO conversion, then the electronic module is used. Translucent OXC provides a compromise of full optical signal transparency and comprehensive optical signal monitoring. It also provides the possibility of signal regeneration at each node.

Optical Add-and-Drop Multiplexer (OADM) nodes can be viewed as a special case of an OXC, which can insert and extract data to and from an optical WDM ring without the need to first convert the signals on all of the WDM channels to electronic signals. (Note that ring topology was prevailing at the beginning of optical networking.) Wavelength channels to drop are converted to the electronic domain (OE); other wavelength

channels traverse the node unaltered. In early OADMs the entire bandwidth assignment was carried out manually; reconfiguration was also performed manually and took much time. In addition, typically the reconfiguration of only a limited number of wavelengths was supported.

The technology called *ROADM* (Reconfigurable Optical Add/Drop Multiplexers) meant a real breakthrough for WDM networks by providing the flexibility and functionality required in present complex networking environments. ROADMs allow service providers to configure and reconfigure add and drop capacity at a node remotely, reducing operating expenses by eliminating the time and complexity involved in manual reconfiguration. ROADMs also allow automatic power balancing of signals according to the paths of demands.

1.1.4. Wavelength Conversion and Grooming

One constraint on creating light-paths through a network is the wavelength continuity constraint, which states that the same wavelength has to be used on each link in the path. A lightpath also requires all OXCs traversed along the assigned route to be set to states in which the wavelengths belonging to the lightpath on subsequent links are connected. This constraint *might result in a routing failure* when there is spare capacity on all links along the path, but not for the same wavelength. *Wavelength conversion* can be used to avoid this constraint. There are two fundamental ways of achieving this: opto-electrical and photonic.

Opto-electrical conversion changes the incoming signal to electrical form and then retransmits it. This kind of wavelength converting transponders rapidly took on the additional function of signal regeneration. Throughout the development of transponders more and more advanced features appeared, in chronologic order: 1R, 2R, and 3R regeneration.

Early 1R detects the incoming analogue signal, amplifies, and retransmits in a "garbage in garbage out" manner; it is entirely modulation format transparent. 2R regeneration adds reshaping; here we detect the digital pulses and retransmit those. 3R adds retiming onto 2R; we have to know the bit rate to be able to retime the digital signal and use either a global clock or clock recovery scheme.

Photonic conversion [18] relies on physical properties, such as cross-gain modulation (XGM) using optical gratings, cross-phase modulation (XPM) using interferometers such as Mach-Zehnder Interferometers, or four wave mixing (FWM). By combining some effects, 3R regeneration can be gained [19]. However, photonic conversion is an immature technology, and it is not clear how long it will take to become more cost effective than opto-electrical conversion.

Many types of optical networks face the challenge of efficiently utilizing the bandwidth of optical fibers (see Section 1.1.1). WDM provides only a partial solution, since demands, originating from the access networks, usually require various, much smaller magnitudes of bandwidth. In order to provide a finer granularity of bandwidth – by allowing multiple calls to share the capacity of a wavelength channel –, it is desirable to deploy an additional *sub-wavelength multiplexing technique*, like *Time Division Multiplexing* (TDM). The process of grouping client demands with appropriate bandwidths into lightpaths is collectively called *grooming* [50][51].

This two-level multiplexing scheme definitely leads to better resource usage, but significantly increases the complexity of routing. Arranging demands even on a single link consisting of multiple wavelength channels with uniform capacities is a close analogy of the bin-packing problem [23], which is known to be *NP-complete*.

It is also worth to differentiate static and dynamic grooming (there is a wide range of literature available on the subject [20][21][22][23][24][25]). If the traffic in the network does not change frequently (which is assumed to be true in backbone and transport networks carrying traffic aggregates), the network operator may have enough time to find the best solution of the NP-hard optimization problem and route all calls accordingly. This is called the *static grooming problem*.

However, as we are moving closer to the access network, traffic demands are getting more variable, i.e. the set of active calls in the network is continuously changing, which renders achieving the optimal routing topology in the network at any time instant – without re-routing of current active calls – almost impossible. Therefore the optimality criteria of *dynamic grooming problem* should also involve that some demands are non-reroutable.

Not all network devices are necessarily equipped with grooming capability. Optical networks, where only designated nodes are able to perform grooming, are called *sparse grooming capability* networks. On the other hand, in *limited grooming capability networks*, the numbers of grooming ports are limited in the devices. Naturally, the combination of these two properties can also occur.

1.1.5. Protection against Failures

Given that an optical network is complex and relies on many components, protection against failures is essential. Failures can happen in numerous ways: some may render a lightpath or set of lightpaths completely unusable (e.g., a fiber cut). Some will degrade a lightpath so that the bit error rate increases to an unacceptable level; bad thermal stability of lasers or multiplexing equipment might cause this. Finally, some failures might leave lightpaths operable, but stop reconfiguration (e.g., failure of switch control hardware or software).

My dissertation does not intend to give an in depth analysis of various protection techniques and algorithms. However, I briefly discuss here the most important protection techniques and classify them.

There exists a range of mechanisms to deal with recovery from a failure. They differ primarily in the speed of recovery from a failure, the range of failures covered, and the excess capacity needed to recover. These properties are mostly influenced by the network level (layer) where the protection mechanism is implemented.

It is an open problem usually to decide which layer should react to a failure in the current or in one of the underlying layers. Several factors should be considered, e.g. which layer can handle the failure the fastest, or which is the most economical solution. Duplicated efforts, caused by multiple layers responding to a single failure, are definitely inefficient. Different optical layer protection schemes may have various advantages over higher network layers [26], and the correct balance between optical layer protection and higher layer protection requires co-ordination and depends on the exact network parameters [27].

We can classify protection mechanisms from several aspects (see Fig. 2 for example). *Restoration* means that demands are rerouted when a failure really happens, no advanced engineering is performed. On the other hand, in case of *protection* proper reactions are already prepared (e.g., spare paths are pre-computed), which allow very fast reaction in order of tens of milliseconds.

We can also decide whether to protect individual demands (*end-to-end path protection*), individual light-paths or physical link (latter two are called *segment protection*).

Protection mechanisms usually consider *single failure*, and do not count with *multiple failures*. Protection against multiple failures would require pointlessly high amount of spare-resources. Therefore usually 2nd and further failures are handled by means of restoration.

Some techniques may protect against *link and device failures*, while others protect against *link failures only*.

Mechanisms can also apply *dedicated* or *shared protection* scheme (at both segment and end-to-end level). In case of dedicated protection, a protection path is assigned to every working path, which implies higher network load (even two times more). The traffic is directed to the protection path in case of failure on the working path (1:1). However, it is also possible that the same traffic is flowing on both paths at the same time (1+1), and receiver can choose the better quality signal. "Handover" is also more seamless in this way.

In the case of shared protection, protection paths belonging to different working paths may possess common network resources. This scheme leads to more efficient resource usage and handover can be also very fast. On the other hand, in general, dedicated protection tolerates multiple failures in the network better.

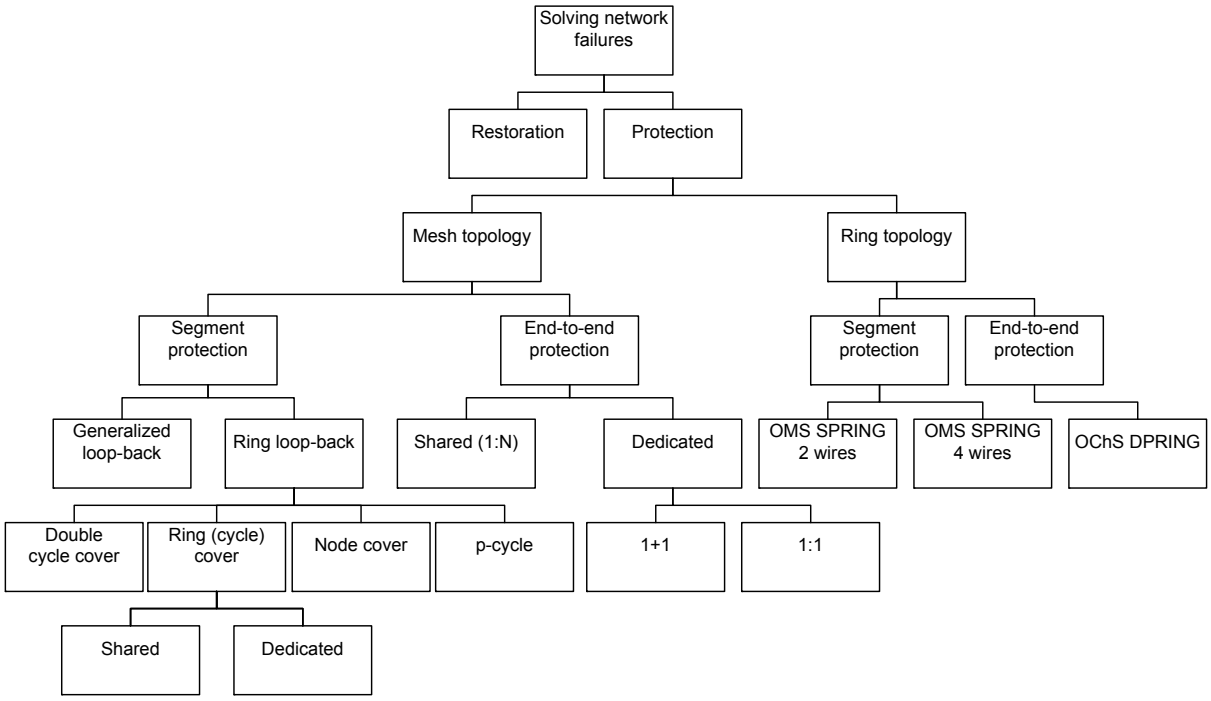

Fig. 2. Classification of protection techniques against failure

1.2. Transport Network Architectures and Protocols

In this section I discuss the classification of network elements from several aspects, namely functional, architectural and administrative classification.

1.2.1. Functional Grouping of Network Elements

Traditionally the functional elements of the transport network infrastructure are classified into three distinct categories called planes: the *transport*, the *control* and the *management planes* (the term "plane" is intended to express the orthogonal nature of this partitioning to the layering and administrative grouping concept discussed in Section 1.2.2 and 1.2.3, respectively).

According to G.8081 [28], "The transport plane provides bidirectional or unidirectional transfer of user information, from one location to another. It can also provide transfer of some control and network management information. The transport plane is layered; it is equivalent to the »Transport Network« defined in ITU-T Rec. G.805 [29]."

The control plane has several purposes, ranging from the efficient configuration of connections within a transport layer network to protection and restoration functions; these are detailed in G.8080 [30]. Finally, "The management plane performs management functions for the transport plane, the control plane and the system as a whole. It also provides coordination between all the planes. The following management functional areas identified in ITU-T Rec. M.3010 are performed in the management plane: fault management; configuration management; accounting management; performance management; security management (FCAPS)." [28].

1.2.2. Layered Network Reference Model

The architectures of communication networks, especially those of the transport networks, can be quite complex. In order to facilitate their discussion, it is often helpful to organize network components into *architectural layers*. A layer is a group of functional elements that might provide services to the directly adjacent upper layer and might rely on services provided by the immediate underlying layer. Other layers should not rely on the internal processes of a particular layer.

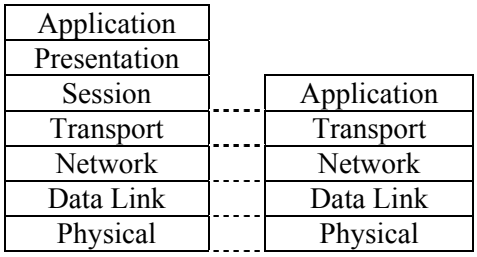

Fig. 3. Layer diagram of the ISO OSI reference model (left) and that of the TCP/IP reference model (right)

With the original purpose of facilitating interconnection between systems, ISO/IEC defined a layered reference model, OSI (Open Systems Interconnection) [31], in which network components were grouped into seven individual layers with respect to their functionality: Physical, Data Link, Network, Transport, Session, Presentation and Application (see Fig. 3, left). DARPA has created a similar model consisting of four layers several years earlier, which has been extended to five layers and is now often referred to as the TCP/IP reference model (see Fig. 3, left).

Many network architectures are compatible with the OSI and TCP/IP type of layering and the principle has been of fundamental importance ever since its introduction. However, very often, it is more convenient to use layers to represent and correspond to individual network protocols; such representations, which are often called *protocol stacks*, might not be entirely compatible with the OSI or the TCP/IP structure, i.e. one-to-one correspondence between protocol stacks and the layers of the OSI or TCP/IP reference models is not always possible. For this reason, in the rest of my dissertation, I will use the term "layer" as a reference to an element of a protocol stack.

1.2.3. Administrative Grouping of Network Elements

In addition to the vertical layering concept discussed above, ITU-T has defined a way of partitioning large-scale networks horizontally into administrative domains and subnetworks (Section 5.3 of [29]). This subdivision serves a complex purpose: it separates the administrative and routing areas of different network operators providing services over the same network; it also enhances network scalability by encouraging the use of hierarchical (inter-domain) routing procedures. Networks comprising multiple administrative domains are

referred to simply as *multi-domain networks*. However, multi-domain related problems are out of scope of my dissertation; optimization techniques and methods proposed here apply for *single domain* only.

1.3. Transport Network Architectures and Technologies

In this section, I give an overview of the relevant network architectures, technologies, concepts and recommendations.

1.3.1. Synchronous Digital Hierarchy (SDH) and Synchronous Optical Network (SONET)

SONET (USA and Canada) and SDH (elsewhere) are two closely related multiplexing protocols designed, respectively, by Telcordia [33] and ITU-T [34][35][36] as replacements and extensions to their respective PDH *T-* and *E-carrier* counterparts. They are capable of multiplexing top-level PDH signals in additional hierarchical levels in exact synchrony up to the aggregate data rates of approximately 40 Gbit/s using fiber optic media in the physical layer (Chapter 2 of [1]) [32] (Chapters 2 and 3 of [37]) (Section 12.2 of [2]) (Section 8.6 of [3]).

SONET/SDH signals are formed by the periodic, synchronous transmission of *frames* conforming to pre-defined formats. Their synchronous nature allows for the simple realization of a wide range of devices, such as *terminal multiplexers, add-drop multiplexers and digital cross-connect* systems (Chapter 2 of [1]). These devices allow the creation and interconnection of self-healing rings, which add a certain level of survivability to the network by the means of link protection.

Next-generation SONET/SDH (NgSDH) protocols were introduced to transfer not only PDH signals, but also a wide range of clients (also with variable bandwidth), including IP, ATM and Ethernet. There can be a large amount of unused bandwidth left over, due to the fixed sizes of concatenated containers. *Generic Framing Procedure (GFP)* introduced in G.7041 [38] and *Virtual Concatenation (VCAT)* together allow mapping of payloads of arbitrary bandwidth into the virtually concatenated container. *Link Capacity Adjustment Scheme (LCAS)* allows for dynamically changing the bandwidth via dynamic virtual concatenation. These features make SDH a flexible, integrated, universal transport network solution [39] (Section 2.8 of [1]).

As the original SONET/SDH design was planned to be *compatible with WDM* [10, Section 6.1], significant research has been devoted to the issues about efficiently deploying SONET/SDH over WDM [22][23][24][25]. In a linear or ring SONET/SDH network, additional parallel channels can be included by supporting them on new wavelengths in the underlying WDM infrastructure; therefore, no additional fibers are required.

While accommodating a given set of calls, the two most important measures that should be minimized are the number of parallel SONET/SDH rings and the number of SONET/SDH add/drop multiplexers used. It has been shown that minimizing either of the variables might result in the other being non-minimal; hence joint optimization methods are required. Traffic grooming (see Section 1.1.4) also raises a couple of problems, many of which have been proven to be NP-complete (e.g., minimizing the number of wavelengths used in a SONET-over-WDM ring network is Karp-reducible to the bin packing problem). Quite obviously, these problems can be generalized to WDM mesh networks as well (Chapter 9 of [21]).

Unfortunately, in its present state, it cannot fully utilize the bandwidth of fiber optics. Its client data rate granularity is too high: clients can only allocate channels with data rates multiple of STS-1 (50 Mbit/s). Furthermore, the SONET/SDH standards only define operation on a limited set of network topology types (linear, ring and interconnected rings), which might prove to be a hindrance of further scalability.

1.3.2. Ethernet

Ethernet is originally a family of frame-based computer networking technologies for local area networks (LANs) standardized as IEEE 802.3 by IEEE [40]. According to the original concept, all devices attached to the network were communicating over a shared medium. Soon Ethernet evolved into the complex networking technology by replacing shared medium with point-to-point links connected by Ethernet hubs and/or switches to reduce installation costs, increase reliability, and enable point-to-point management and troubleshooting. Ethernet is transmitting data on the physical wire in form of frames containing preamble, start of frame delimiter and the payload. The simple framing structure allowed of creating simple and cheap hardware.

Through its rapid development the data rate of Ethernet gradually increased, beginning with the early 10Mbit/s version through 100 Mbit/s, 1Gbit/s to 10Gbit/s. Ethernet supports various types of transfer medium, including *coaxial cable* (e.g., 10BASE2 standard), *twisted pair* (10BASE-T, 100BASE-T, 1000BASE-T) *and fiber optics* (100BASE-FX, 1000BASE-SX/LX/CX). 10 gigabit version of Ethernet support also multimode (10GBASE-LX4 and SR) and single mode (10GBASE-LR/ER) fibers for short distance (26-82 m and 240-300 m) and long distance (10-40 km) communication, respectively. There is a set of 10 gigabit standards (10GBASE-SR/LR/ER) – corresponding to the former ones and hence using the same types of fiber and support the same

distances – that were designed to interoperate with OC-192 / STM-64 SONET/SDH equipments. This also implies slightly lower data rates, corresponding to the SONET/SDH carriers.

Since Ethernet is an inexpensive, economical solution, it is often used in Metropolitan Area Networks (MAN) and it is also getting popular in Wide Area Networks (WAN). IEEE is already developing the next generation of the standard allowing 40 Gbit/s and 100Gbit/s of transfer rate and an operating distance of up to 40 km. The upcoming standards use solely optical fiber as transfer medium and provide appropriate support for OTN (Optical Transport Network), but preserves original frame format and MAC addressing.

1.3.3. Internet Protocol (IP)

The potential role of IP as a convergence layer and abundance of physical resources of WDM makes the IP/WDM combination a promising solution in tomorrow's backbone networks (Chapter 18 of [21], [41]).

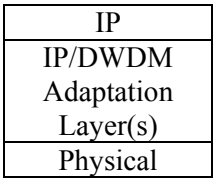

Fig. 4. "IP over WDM" solution requires an adaptation layer

The most obvious problem about this pairing rises from the profound differences between the two components: as WDM is a physical-layer solution and IP is a packet-relay networking protocol, *any realization requires one or more intermediate adaptation layers* (see Fig. 4). In practice, such realizations are easily imaginable by using a large number of intermediate layers, such as SONET, ATM or Ethernet.

1.3.4. Multi-Protocol Label Switching (MPLS)

MPLS [42][43] is a packet-forwarding protocol designed for high–performance applications in mesh networks. It imposes very few requirements on the client layer and minimal processing demands on the forwarding equipment.

MPLS works by prefixing packets with an MPLS header, containing one or more "labels", which are collectively called *label stack*. The forwarding of the packet is done based on the contents of the labels, which allows "protocol-independent packet forwarding" that does not need to look at a protocol-dependent routing table and *avoids the expensive IP longest prefix match* at each hop.

MPLS *supports traffic engineering* (MPLS-TE) by being able to define arbitrary paths for network flows through the use of Label Distribution Protocols. MPLS clearly separates routing and forwarding (i.e. control and transport), thus a single forwarding mechanism can employ a wide range of routing procedures (with a strong emphasis on *constraint-based routing*) and routing protocols may base their decisions on *multiple metrics*, such as bandwidth availability, latency, link costs, etc. (Section 5.5.4 of [37]).

1.3.5. Generalized Multi-Protocol Label Switching (GMPLS)

By generalizing the label-switching paradigm, MPLS proposal has been extended for non-packet orientated networks, such as optical networks, through the *Generalized MPLS (GMPLS)* proposals. It realizes that switching cannot be done according to a pre-pended packet header only, but can be performed according to the underlying architectural layers and protocol stack. I.e., GMPLS is not limited to packet forwarding anymore, instead, it assumes coexistence of Packet-Switching Capable, Layer 2 Switching Capable, TDM-Switching Capable, Lambda-Switching Capable and Fiber-Switching Capable devices (Section 5.5 of [37], [44]). GMPLS propagates inter-operability between these switching layers to integrate/reduce/bypass unnecessary ones. It also reuses MPLS-TE protocol suite and mechanisms.

Networks capable of operating multiple switching technologies are referred to as *Multi-Region Networks* [112]. *Vertical integration*, i.e. co-operation between the control planes of the different technologies, is highly desirable. Generalized MPLS supports this requirement by defining collaborative mechanisms between the control planes of the data planes based on different switching technologies. Vertical integration complements *horizontal integration*, which refers to the facilitation of inter-operability between the control planes of separate routing areas (autonomous systems) based on a common switching technology.

GMPLS defines three layer inter-operability models: the *overlay model*, the *augmented model* and the *peer model*, which is also known as the *unified or integrated model* [112]. The upper layer may be a framing layer (e.g., SONET/SDH) encapsulating IP packets, while the lower layer is typically a WDM layer. However, GMPLS supports IP directly over WDM by eliminating the intermediate layers.

The *overlay model* is the legacy model having two separate control planes or instances of routing and signaling protocols. The upper layer acts as a client to lower one; the topology of the lower layer is invisible for the client. They are communicating through the User-Network Interface (UNI). The model does not significantly facilitate interworking between the control planes of adjacent layers, assuming layers operated by different service providers that share a low-trust business relationship. On the other hand, the separation of the control planes allows better failure isolation, scalability, survivability, domain security (exposure of control and topology information), and independent evolution of technologies [113]. This model is opaque and prone to the so-called unknown adjacency problem that results from the duplication of the same routing functionality [112] in different layers.

The *peer model* allows the complete unification of the control planes: it allows a complete exchange of control and signaling information (through the Network-Network Interface, NNI) between them while using a common addressing space. A single instance of the control plane is used for addressing, routing, and signaling. The peer model is applicable to a single administrative domain. Obviously, the algorithmic control of multiple data planes is more complex than that of a single one, but more efficient solutions can be achieved this way [113].

The augmented model serves as an intermediate solution between the overlay and peer models. It allows the exchange of a limited amount of routing information between the client and the server layer.

Throughout my dissertation I always assume that the two layers (optical and electronic) are interconnected according to the peer model.

1.3.6. Automatically Switched Optical Network (ASON)

The ASON is a recommendation proposed by ITUT [28][30] comprising a complete control plane reference architecture for automatically switched transport networks [45], that is, transport networks based on SDH [36] and OTN [46]. In the recommendation, three basic connection types are supported: *Permanent Connections*, which are set up and torn down by the management plane; *Soft Permanent Connections*, whose user-to-network part is established by the management plane as a Permanent Connection, while the network-network part is maintained as a Switched Connection; and, finally, *Switched Connections*, which are established on user demand by the signaling/control plane (this involves the dynamic exchange of signaling information between signaling elements of the control plane) [28]. The recommendation describes a control plane whose functionality is threefold:

1. It facilitates the efficient configuration of Switched and Soft Permanent Connections within the transport layer network;
2. Provides capability for the reconfiguration of connections that have already been set up;
3. Performs a restoration function [30]. The User-Network Interface used by ASON is described in the associated OIF UNI specification [47]. The high-level overview of the ASON architecture is shown in Fig. 5.

The term *Automatic Switched Transport Network* (ASTN) used to refer to any transport network where configuration connection management is implemented by means of a control/signaling plane (e.g., ASON [30] or GMPLS [47]) (Section 6.2 of [48], Section 3.2.5 of [28]). As of 2007, the ASTN recommendation has been integrated into ASON.

The *OTN Recommendation* [46] is intended to serve as an extension of SONET/SDH (Sections 8.4.2 of [5]). It provides better error correction, better aggregation and reconfiguration capability, transparent client signal transport, switching scalability, dynamic provisioning and a better support for WDM (interoperability) (Section 19.9 of [5]). Basically, OTN provides the transport layer for the control and management functions described in ASON.

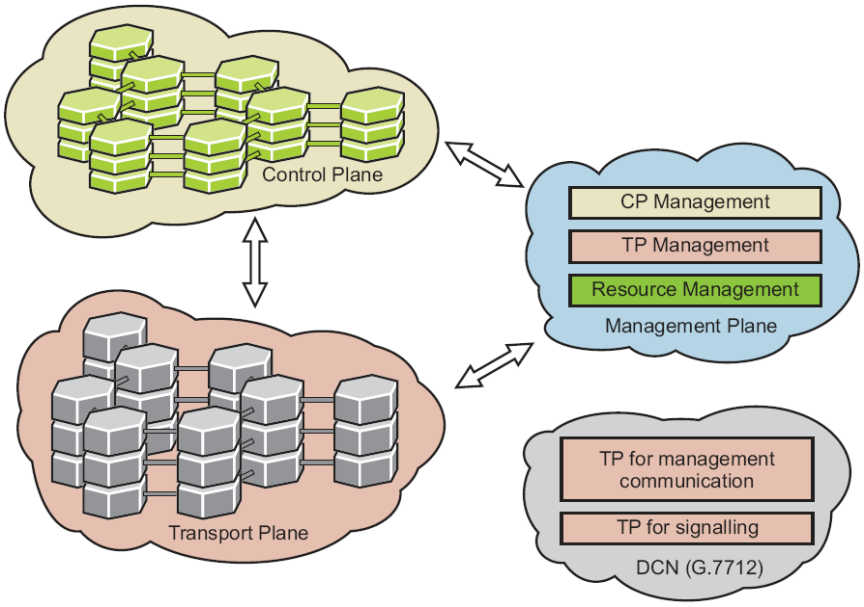

Fig. 5. High-level overview of ASON Architecture

1.3.7. Wavelength-Routing Networks

Wavelength-Routing (WR) networks are circuit-switched networks consisting of OXCs (see Section 1.1.3) devices interconnected by optical fibers and providing transparent (or opaque) end-to-end broadband connections between network endpoints.

A WR network (Chapter 9 of [1], Part III of [8], and Section 10.3 of [49]) provides excellent means of utilizing the capabilities of the WDM technology in arbitrary mesh topologies. These networks (and generally WDM fiber optics) can be efficiently deployed in the transport planes of core and backbone networks that carry high-density aggregate traffic between subnetworks.

A central issue in the provisioning of WR network resources is the allocation of wavelength channels to lightpaths. This process is called *Routing and Wavelength Assignment (RWA)*. A *lightpath* provides end-to-end, circuit-switched connections between a pair of physical nodes using a single, previously specified wavelength.

First, in order to establish a lightpath, a sequence of physical links between the endpoints should be found; the resulting sequence will be the assigned route or path of the lightpath. Second, a certain wavelength, which is available (free) on all links, has to be assigned to the lightpath. Conventional OXCs are incapable of wavelength conversion, therefore the *wavelength continuity constraint* applies: a lightpath must use the same wavelength on any two consecutive links traversed; clearly, a lightpath must use the same wavelength throughout its whole length.

A WR network providing fully *transparent* connections assign a single, dedicated lightpath to each traffic demand.

Whenever a wavelength conversion is needed in a node, the OXC should switch the respective signal to the electronic layer; the electronic layer returns it to the WR network layer on a different wavelength. With the aid of the electronic layer, *3R signal regeneration, wavelength conversion*, and even electronic TDM multiplexing (*grooming*) are all possible.

Traffic grooming can cure bandwidth efficiency issues and increases resource utilization; on the other hand it makes the RWA problem more complex.

A WR network providing *semi-transparent* (transcendent, opaque) end-to-end connections may assign several consecutive lightpaths to each traffic demand.

In large-scale WR networks (Trans- or intercontinental backbone networks), signal degradation due to *transmission impairments* (Section 8.2.2 of [1]) cannot be neglected; lightpaths cannot be arbitrarily long as signal regeneration is not yet possible purely in the optical layer. Considering *physical effects in routing* has recently attracted much attention. Section 3 of this work presents a significant achievement on this field by introducing a novel RWA algorithm that takes into account physical effects (see Chapter 3).

1.4. Simulation and Modeling

A double-layer network is assumed, where the upper layer is an electronic, TDM capable time switching capable, while the lower layer is a wavelength switching capable one.

It is also assumed that the two layers are interconnected according to the *peer* (or *vertically integrated*) *model* according to the multi-region network node framework, i.e. throughout routing the control plane has information on both layers and both layers take part in accommodating a demand. Peer-model allows optimal routing using the resources of both layers jointly. Note, that all of my algorithms and results are – probably with minor modifications – applicable to overlay or augmented interconnection models as well.

I decided to use a general network model for routing in two layer networks with grooming and with different types of nodes and arbitrary topologies. The model must be able to handle any regular mesh topology. For all these reasons I chose *wavelength graph (WL Graph, WLG) modeling technique* to represent the network. The WL graph corresponding to the logical network is derived from the physical network (i.e., it can be regarded as a virtual representation of the physical network, see Fig. 6) considering the topology and capabilities of physical devices. WL graph modeling allows and significantly facilitates performing *routing* and *wavelength assignment* at the same time, which is collectively known as the *RWA* problem.

The WLG itself is a directed graph described by a quadruple: *G(V,A,C,B)*, where *V* and *A* are the set of (logical) nodes and (logical) directed edges, respectively. The term "logical" is used in order to distinguish elements of the WLG from the elements of the physical network (i.e., physical nodes and links). The set of costs and capacities assigned to the edges are denoted by *C* and *B*, respectively. To model physical impairments, the set of length of fibers (*L*) should be also taken into account.

If the aim is to model bidirectional traffic only, undirected WLG can also be used. However, I decided to use the directed version, since in my dissertation unidirectional end-to-end traffic and multicast traffic (with designated source) are concerned.

A simpler version of the model has been first proposed in [91]. Integer Linear Programming (ILP) formulation of the static RWA problem with grooming and protection has been given in [100].

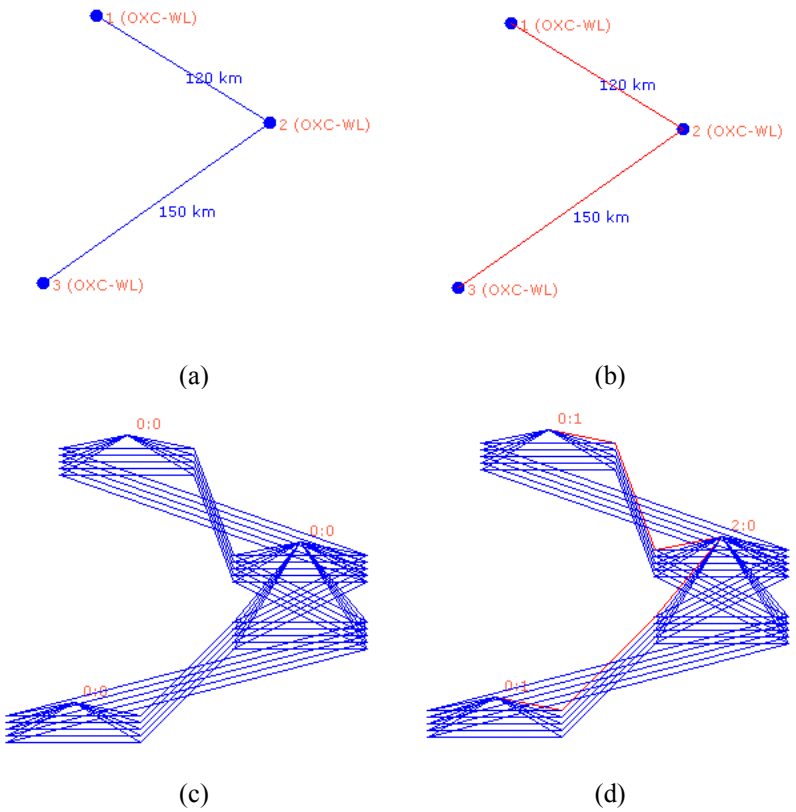

Fig. 6. Physical topology of the network (a), Physical topology with two routed demands (b), WLG (virtual) representation of the network (c), WLG representation with two routed demands (d)

The types of nodes can also be quite different: Optical Add-and-Drop Multiplexers (OADM), Optical Cross-Connects (OXC: optical core) with full or limited, optical or opto-electrical WL conversion or even an Opto-Electrical Cross-Connect (OEXC: electrical core). Furthermore, some of these nodes support grooming, typically

with limited number of optical ports. *All these properties can be considered in the WL graph* model, together with different protection techniques of traffic demands.

The network consists of *physical devices (physical nodes) and optical fibers (physical links)* connecting the physical devices. Both ends of a fiber are attached to an interface (IF) of the corresponding physical device. A physical device contains an internal switching fabric and some IFs. The number of available WLs in a fiber is the minimum of the WLs supported by the end IFs. *Every link and every physical device has a specific logical representation in the WL graph* (see Fig. 7) depending on the capabilities and properties of the physical device.

A physical link is derived to as many logical edges as the number of available WLs in the link. The logical sub-graph of a physical device (see Fig. 7) depends on the capabilities of the device. Every edge in the graph has a capacity and a cost of usage. The capacity of the edge usually equals to the WL capacity, which depends on the used carrier (typically 2.5 Gbps or 10 Gbps). The cost of the edge is determined by its functionality (WL edge, O/E conversion, etc.).

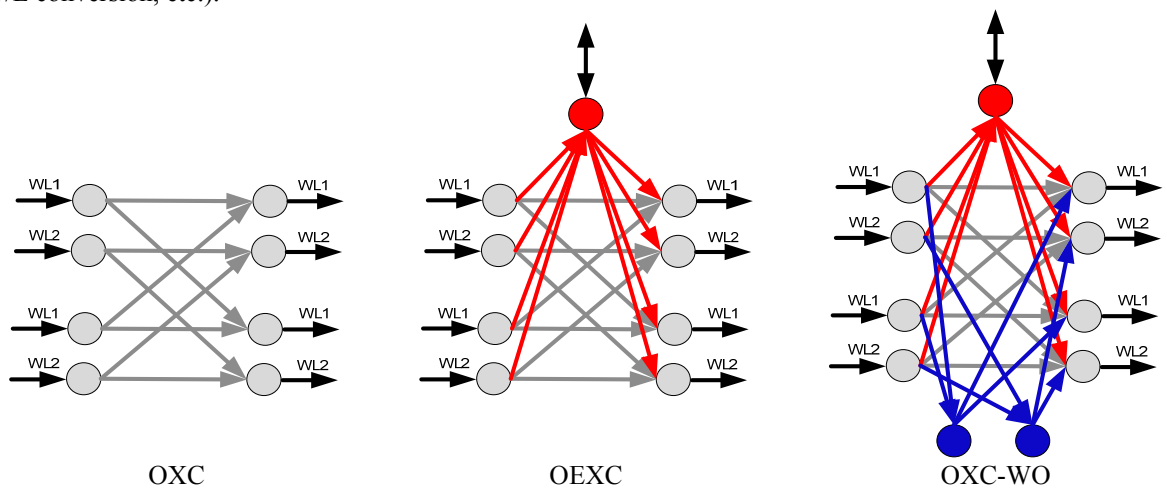

Fig. 7. Representation of physical devices by a sub-graph in the WLG: Optical Cross-connect (OXC) (a), OXC with (electronic) wavelength conversion (OEXC) (b), OEXC with optical branching capability (OXC-WO) (c)

Fig. 6 illustrates the creation and maintenance of the WLG. Here a simple physical topology is visible comprising 3 physical nodes and 2 physical links. The nodes are all OEXCs shown in Fig. 7/b. The topology implies that the 1st and 3rd node have one (used) physical interface, while the 2nd physical node has two interfaces. The physical links are 120 km and 150 km long, respectively. The number of supported wavelengths is 5 on each interface, which makes 5 wavelength channels available in each fiber. Fig. 6 shows how WLG (Fig. 6/c) is derived from the physical topology (Fig. 6/a): physical devices are substituted by the corresponding sub-graph, while fibers are replaced by a number of parallel edges – depending on the number of supported wavelengths.

Fig. 6/b and Fig. 6/d show the routing after inserting two demands (demand 1→2 and demand 3→2). The used physical links (Fig. 6/b) are marked with red color. Fig. 6/d shows lightpaths assigned to the demands. In this simple case one lightpath is assigned to each demand. Fig. 6/d also illustrates how the WLG facilitates the joint usage of two layers to accommodate the demands. E.g., the 1st demand starts from the electronic layer and enters the optical layer in the 1st node, passes the 120 km fiber on the 1st wavelengths, arrives at the 2nd node and returns to the electronic layer here.

Fig. 7 shows the sub-graph of some exemplary physical devices, i.e. the virtual representation of the device in the WLG. The topology of the sub-graph and the costs assigned to the (internal) edges are determined by the functionalities. Fig. 7/a depicts a simple OXC: incoming wavelength on any interface can be switched to the same wavelength on any outgoing interface (no wavelength conversion is possible). The OEXC in Fig. 7/b supports wavelength conversion by routing the demand up to the electronic layer; O/E and E/O conversions are represented by red edges. The device shown in Fig. 7/b is a special one able to perform optical lightpath branching, i.e. in case of the blue nodes, the incoming nodal degree is not equal to the outgoing nodal degree.

For the details of my versatile simulator, based on WLG, see Chapter 11 in the APPENDIX. Most of the proposed, new methods were implemented and evaluated by this tool.

2. Grooming Capability and Wavelength Number Dimensioning

In switched multi-layer optical networks traffic grooming is the key of efficient bandwidth utilization. Without it a single traffic demand would occupy a full wavelength channel. However, it is not necessary to install grooming capability in all nodes of a network. Furthermore it is also unnecessary to equip all grooming nodes with full grooming capability. We can reduce the costs by deploying only the necessary grooming capacity. *In this chapter, I introduce algorithms based on statistical utilization analysis, which determine not only the necessary number of grooming ports, but also the necessary number of wavelengths in the network.* I compare these dimensioning results for different protection techniques (i.e. dedicated or shared end-to-end path protection).

The chapter is organized as follows. Section 2.1 formulates the problem, Section 2.2 presents the proposed algorithms, Section 2.3 evaluates the obtained simulation results, and finally Section 2.4 concludes the chapter.

2.1. Problem Formulation

The network topology and the number of fibers are assumed given as well as the estimated busy hour traffic. The capacity of wavelength channels, the cost of grooming capability and the cost of grooming ports are also given in advance.

Dynamic traffic demands are assumed and three kinds of protection schemes (since one of the goals is to compare resource requirements of these protection schemes): *no protection* (as a baseline), *dedicated* end-to-end path protection, and *shared* end-to-end path protection.

The simplest, but most expensive approach is to equip all nodes with full wavelength conversion capability, i.e. all wavelength channels can be terminated in a node and switched via the electronic space-and-time switch. However, this solution is very expensive.

A wiser and more economical approach is to estimate the required number of grooming ports in each node and to decide whether grooming capability is needed at all in the specific node. The decision depends strongly on the network topology, on the number of fibers and wavelengths per link as well as on the traffic conditions.

The objective is to find the optimal (in terms of cost) network configuration that can satisfy all demands with an upper bound on the allowed blocking ratio.

The used network and grooming models are described in details in Section 1.4. *I have solved a similar, but simpler problem in [52], where the grooming facility location was the task without determining the number of ports or the number of wavelengths per fiber.* A similar problem was discussed in [53].

There are some papers on sparse wavelength conversion capability dimensioning, but their model does not support grooming [54][55][56].

Numerous publications have been presented for the dimensioning of grooming ports after my paper had been published in 2005. Correia et al. proposed different grooming node placement strategies for sparse grooming capability networks in [57]. They separated the problem into a static grooming node placement sub-problem and heuristic, dynamic routing problem. Awwad et al. introduced heuristic methods (including one based on genetic algorithm) to minimize the total cost of routing in the network without hindering blocking in [58]. In [59] the authors proposed a dynamic routing algorithm and an analytical model to evaluate the effects of the number of grooming nodes in the network on the blocking performance.

The authors of [60] have given an ILP formulation and a heuristic approach for the allocation of grooming ports in a limited grooming capability network. Another heuristic approach was given in [61] to optimize grooming capacity for dynamic traffic.

According to my knowledge, apart from the method proposed in the chapter, no other methods apply the statistical utilization approach to address the problem of grooming port dimensioning. Furthermore, my method should be considered as a high-level heuristic, which can be used in conjunction with other routing algorithms (or even protection techniques).

2.2. Algorithms

In this section I present three *heuristic algorithms* in Section 2.2.1, 2.2.2 and 2.2.3, respectively, which use *iterative simulations* to determine the number of grooming ports necessary in each node and/or the number of wavelengths on each link. This means that they start from an initial network configuration and use simulation to measure different characteristics of the network; then they modify the network configuration based on the outcome of the simulation, and start a new simulation round. The iteration stops when such a network configuration is reached, which is the same as the previous one, i.e. the algorithm cannot further improve it. I will refer to this configuration as a “balanced state” hereafter.

2.2.1. Dimensioning the Number of Grooming Ports

The input of this algorithm is the network topology. For the reason of simplicity I suppose that the number of wavelengths (WL) is the same on each link, though the algorithm can be easily extended to handle different number of WLs on each link. Let the number of fibers connected to node n be denoted by R_n.

I assume that the initial number of wavelengths on the links is relatively large, thus the blocking in the network is primarily caused by the nodes (namely by the lack of grooming ports) and not by the lack of link capacity.

Initially the number of O/E and E/O converter ports was $WL \cdot R_n$ in all nodes. This is the theoretical maximum on the number of grooming ports needed, since this many ports ensure that every wavelength on each interface can be routed to the electronic layer of the switch. The numbers of O/E and E/O ports are the same. Each O/E conversion reduces the number of free O/E ports by one. This may happen in the following cases:

- Every time when a wavelength in a node is routed up from the optical layer to the electric layer to perform wavelength conversion or traffic grooming. A reverse conversion occurs when a wavelength channel leaves the optical layer and enters the optical layer. Reverse conversion reduces the number of available E/O ports by one.
- Note that a wavelength may carry multiple (multiplexed) traffic demands! The conversion (reverse conversion) of such a wavelength "consumes" only a single O/E (E/O) port too – irrespectively of the number of demands groomed together in that wavelength.
- Also note that if a node is the source (destination) of a traffic demand then one E/O (O/E) port is used necessarily!

Throughout the simulations the number of available O/E and E/O ports in a switch changes independently. However, due to the symmetry of the demands, their statistical properties are the same. I assumed unidirectional demands and that all nodes have the same chance to become either the source or the destination of a demand; this explains the symmetry.

In each logical time step of the simulation the numbers of available O/E and E/O ports are logged for each node separately. From this data two histograms for each node (Fig. 8) are constructed. The value of the O/E histogram at n expresses the ratio of simulation time when the number of available O/E ports was equal to n. The E/O histogram means the same for the E/O port utilization. Fig. 8 shows the histograms of a less loaded (practically underutilized) and a heavily loaded (overloaded) node.

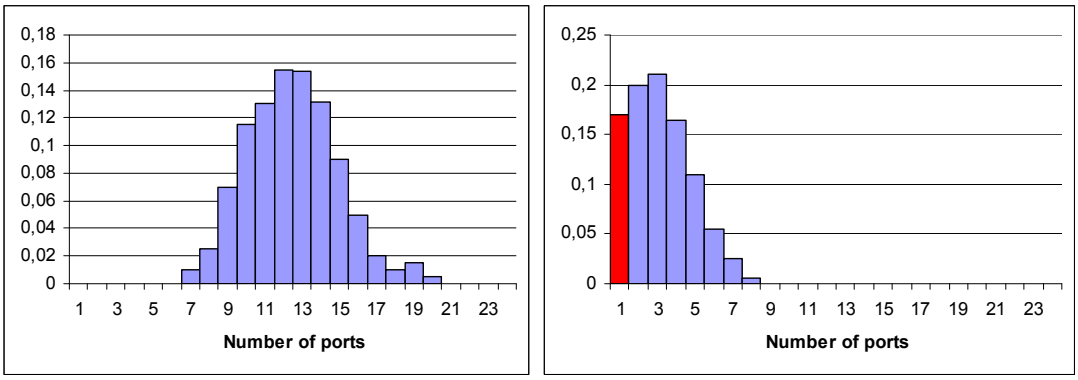

Fig. 8. The distribution of the number of available ports in a less loaded node (left) and in a heavily loaded node (right)

The value of such histogram at zero is of great importance. This shows the ratio of time when the node was unable to perform further wavelength conversion or grooming because it had no free O/E or E/O ports. This value is not equal to the blocking rate of the node (pB_n) because a demand passing through the node does not necessarily reach the electric layer. It may be handled only by the optical layer of the switch. Thus pB_n is lower than the value of the histogram at zero, but there is a connection between them. The value of pB_n determines the blocking rate of the network as follows (assuming that it is the same for every node):

$$avgB = 1 - (1 - pB_n)^i \approx i \cdot pB_n \qquad (0.1)$$

where i denotes the number of nodes an average demand passes, which is the average distance of nodes plus one.

We modify the grooming port number so that the resulting per node blocking rate is lower than TH_n. This can be done the following way:

$$k := k_{\max} - 1, \text{ that } \sum_{i=0}^{i=k_{\max}} p_i \le TH_n \qquad (0.2)$$

where p_i is the relative frequency of the event that the number of free ports is i (the i^{th} column of the histogram).

The number of the ports is decreased by k. The value of k is obtained by summing up the values from left to right of the histogram (the left tail of the distribution) until the sum reaches the threshold TH_n. This procedure may be regarded as the "shifting" of the histogram to the left. In the special case, when k_{max} equals to 0, k is -1. This means that the number of ports is increased by one.

According to my experiences, for small values of TH_n (<0.03) this procedure should be iterated, because the balanced state is not reached in one step. For larger values of TH_n it takes only one step to reach the balanced state.

2.2.2. Dimensioning the Number of Wavelengths

In this case I want to determine how many wavelengths are necessary on each link. The number of grooming ports in the nodes is supposed to be sufficiently large so that the blocking of the network is caused by the link capacities. I follow a logic similar to the one in the previous subsection. In this case, however, the used capacity of each link is logged in each time step of the simulation in order to construct the histograms at the end. The aim is to determine if a link is underutilized or over-utilized and to decrease the number of wavelengths on underutilized links and to increase it on over-utilized ones.

If the number of wavelengths on the link is denoted by WL, then the free capacity of the link may vary between 0 and $WL \cdot C_{WL}$, where C_{WL} is the capacity of a single wavelength. Every wavelength has the same capacity in our model.

The free capacity of each link is calculated from the used capacities. The histograms (Fig. 9) are constructed the same way as the grooming port dimensioning algorithm did it. Before constructing the histograms the values of free capacity are quantified to avoid having histograms of $WL \cdot C_{WL}$ columns. Quantifying is necessary because running simulations which produce such fine-grained results would take too much time. The $WL \cdot C_{WL}$ capacity range was divided into N parts uniformly.

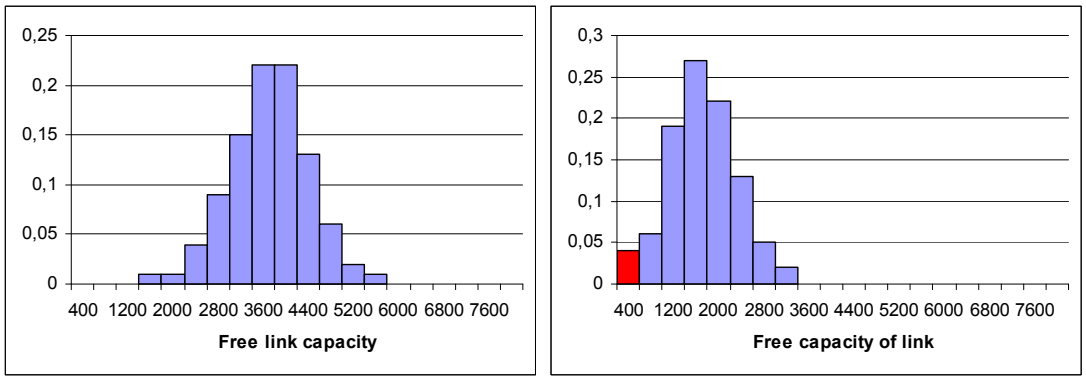

Fig. 9. The distribution of the free link capacity on a less loaded (left) and a heavily loaded link (right)

Determine the blocking caused by one single link is not easy. Hence the following assumptions are made:

- A demand leaving a given network node on a given interface cannot pass the link if there is no wavelength available with a free capacity larger than the bandwidth of the demand. In such a case the link blocks the demand.
- The unused capacities on different wavelengths are independent.

A demand gets blocked by a link if the bandwidth of the demand is larger than the maximum of the free capacities (c^*) of the wavelengths on the link. For the sake of simplicity, I will refer to measure as the *"largest free block"* on the link. Only the *sum of the free capacities* is known on the link (C), but I want to determine the *maximum of these free capacities*. I.e. a connection between these two probability variables has to be found:

$$C = \sum_{i=1}^{WL} c_i \text{ and } c^* = \max(c_i) \qquad (0.3)$$

where c_i is the free capacity of the i^{th} wavelength, and WL is the number of wavelengths on the link.

Simulation results suggested the following approximation of c^* (except when the number of wavelengths is 1, in this case $C = c^*$):

$$c^* = 1000 \cdot \left(1 - e^{-k \cdot C}\right) \qquad (0.4)$$

where k is a constant dependent on the number of wavelengths.

The blocking rate of a single link (pB_l) can be approximated with the following sum:

$$pB_l = \sum_{i=0}^{N-1} P\left(bw > (i+1) \cdot \frac{WL \cdot C_{WL}}{N} \right) \cdot P\left(c^* < (i+1) \cdot \frac{WL \cdot C_{WL}}{N} \right) \qquad (0.5)$$

where N is the number of columns in the histogram and bw is the bandwidth of the demand we want to route.

The number of wavelengths is modified similarly to the number of grooming ports in the previous algorithm. The histogram is being "shifted" to the left while pB_l is smaller than the blocking rate threshold (TH_l) of the link. Obviously the histogram can only be shifted by an integer multiple of the wavelength capacity (due to the fact that only whole wavelengths can be added or removed). The number of wavelengths on the link is modified according to the following formula:

$$k := k_{\max} - 1 \text{, so that } pB_l = \sum_{i=0}^{i=\frac{N}{WL}k_{\max}} P\left(bw > (i+1) \cdot bw_N\right) \cdot P\left(c^* < (i+1) \cdot bw_N\right) \qquad (0.6)$$

where $bw_N = \frac{WL \cdot C_{WL}}{N}$.

Having a look at the formula, it can be seen that the number of wavelengths can be decreased by any arbitrary number, but can be increased only by one.

In every step of the iteration the above described procedure is applied to every link of the network. Reaching a balanced state may require several (i.e. more than one) iterations depending on the value of TH_l. In case of starting from a "lower state" surely more steps are needed, because the number of wavelengths can be increased by only one in each step.

2.2.3. Dimensioning both the Number of Grooming Ports and the Number of Wavelengths

Now I will combine the previous two algorithms. The combined algorithm optimizes the number of grooming ports and the number of wavelengths at the same time. The cost of the wavelengths is supposed to be larger than that of the grooming ports. Therefore, the primarily goal is to decrease the number of wavelengths.

This algorithm finds the balanced state through iterations by all means. In the starting configuration every link has only one wavelength, and the number of grooming ports is R_n in all nodes.

Just as previously, after every simulation the histograms are calculated and modifications are made accordingly. After each step it is true that the number of grooming ports in all nodes is less than or equal to

$$\sum_{i=1}^{R_n} WL_{n,i} \qquad (0.7)$$

where $WL_{n,i}$ is the number of wavelengths used on the i^{th} interface (link) of node n, and R_n is the number of the links connected to node n. The algorithm applies the following heuristic rules:

1. *Increase both the port number and the wavelength number*: when the number of ports in a node exceeds the maximum, the number of wavelengths is increased by one on each adjacent link, and the number of ports is set to the maximum as well.
2. *Decrease the number of ports*: the same rule used for decreasing the number of ports in the first algorithm. If suitable values of TH_n are used (not too small) one step convergence is likely.
3. *Modifying the number of wavelengths and the number of ports:* the same rule used to modify the numbers of wavelengths in the second algorithm. Additionally, if the number of wavelengths on a certain link changes, the number of ports in the adjacent nodes are set to the maximum.

The network reaches a balanced state, if the algorithm changes neither the number of ports nor the number of wavelengths anymore. The algorithm may start from two types of initial configurations. Starting from a "lower state" (bottom up dimensioning) means that both the number of ports in all nodes and the number of wavelengths on all links are lower in the initial configuration than in the balanced state. Starting from an "upper state" (top down dimensioning) means that in the initial configuration all resources are over-dimensioned.

Network topology and traffic load considerably influence the result of the simulation rounds and the outcome of the whole algorithm along with other important simulation parameters like TH_n and TH_l. The algorithm runs until the given TH values are reached on every link and node.

The iteration might not reach the balanced state in all cases; but, instead, it starts oscillating around it. The oscillation can be eliminated by changing the traffic pattern. Another option is, regarding that the amplitude of the oscillation is very small, to stop the iteration and to consider one of the oscillating states as the balanced state.

2.3. Simulation Results

Two reference network topologies were used in the simulations. NSF network [118] contains 14 nodes and 21 links (see Fig. 15), while the US Longhaul consists of 28 nodes and 45 links. The traffic was generated by my own program. Both inter-arrival times and holding times of the demands have an exponential distribution. The traffic intensity can be adjusted by setting the parameters of the distributions. Source and destination nodes of demands are chosen randomly. In the simplest scenario shortest path routing is applied without protection.

Firstly, the behavior of the algorithm optimizing the number of grooming ports (introduced in Section 2.2.1) is investigated. In the initial configuration there were 7 wavelengths on each link and the maximal number of grooming ports in all nodes. The volume of the traffic was tuned so that the blocking rate of the network should not exceed 1‰. Thus the experiment starts from an oversized network and wants to observe the effect of increasing TH_n, i.e., allowing more and more blocking in the nodes. The results are presented for NSF and Longhaul as well, though they are quite similar in case of both networks.

According to Fig. 10, the network blocking rate is rising exponentially as the per node blocking rate threshold (TH_n) is increasing. The intensity of the growth is proportional to the average node distance, which is 2.14 in NSF net and 3.36 in Longhaul network.

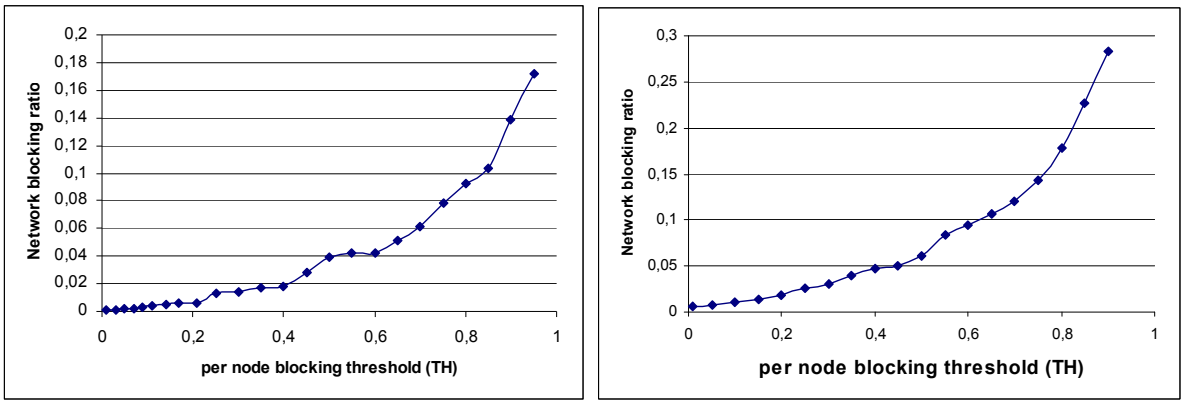

Fig. 10. The blocking rate of the network as a function of the per node blocking rate threshold (TH_n) for NSF net (left) and for Longhaul (right).

Fig. 11 shows the total required number of grooming ports in the network as a function of the blocking rate. The curve falls steeply around zero. It infers that ensuring very low blocking rates requires a high number of grooming ports, which significantly boosts network costs. In the case of "NSFnet", for example, allowing a blocking rate of 18‰ instead of 1‰ reduces the required number of grooming ports (the cost) by almost 30%. These curves allow network providers to find a trade-off between blocking and cost and to make a decision accordingly. After choosing the desired blocking rate for the network, the value for TH_n can be obtained from Fig. 10, and by applying this TH_n, the algorithm can determine the number of grooming ports in each node.

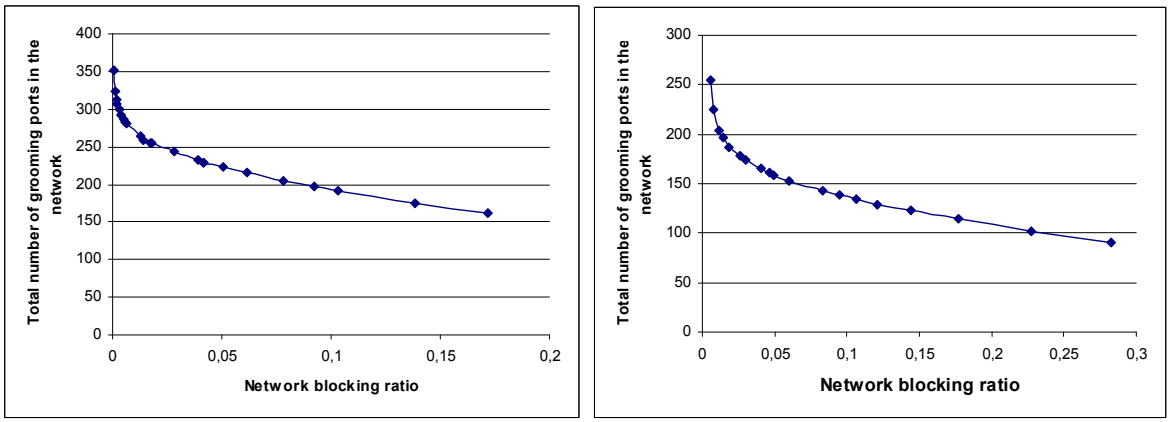

Fig. 11. The number of grooming ports required to reach a certain blocking rate of the network in case of "NSFnet" (left), and "Longhaul" (right).

Now the results of the second algorithm, used to determine the number of wavelengths (introduced in Section 2.2.2), will follow. In the initial configuration there were 7 wavelengths on every link, and the maximal number of grooming ports in every node. The volume of the traffic was adjusted again to ensure a blocking rate under 1‰ in the network. Afterward, the per link blocking rate threshold (TH_l) was started to increase. The numbers of

grooming ports in the nodes were set to the maximum – according to the current number of wavelengths – not to make this a limiting factor. The impact of changing TH_l is similar to the altering of TH_n, so similar conclusions can be drawn.

Fig. 12 shows the blocking rate of the network as a function of the per link blocking rate threshold (TH_l). In this case the exponential increase is not obvious. For small values of TH_l the blocking rate of the network is close to zero; for medium values of TH_l the curve is more or less linear, and after a given point it starts increasing exponentially. However, in general, modification of the TH_l value affects the blocking rate of the network in a similar way as the modification of TH_n.

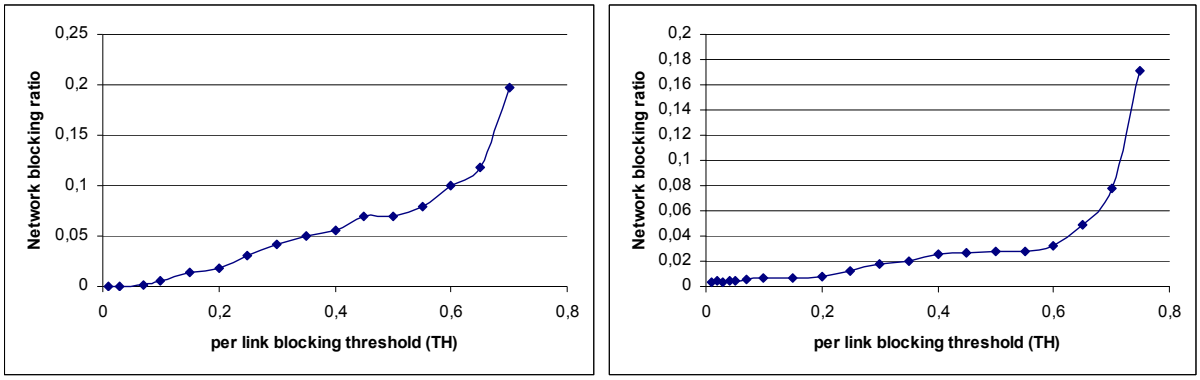

Fig. 12. The blocking rate of the network as a function of the per link blocking rate threshold (THl) for "NSFnet" (left) and for "Longhaul" (right).

Fig. 13 shows the blocking rate of the network as a function of the total number of wavelengths in the network. (The total number of wavelengths in the network is the sum of wavelengths on individual links.) The required number of wavelengths falls rapidly close to zero, which suggests that it is not worth to demand a blocking rate below a certain level, since it requires an extremely high number of wavelengths and becomes very costly. A service provider can find a trade-off between network blocking and costs (number of wavelengths) with the aid of this curve. In case of "NSFnet", for example, allowing 0.18‰ blocking rate instead of 0.5‰ decreases the required total number of wavelengths approximately by 30%.

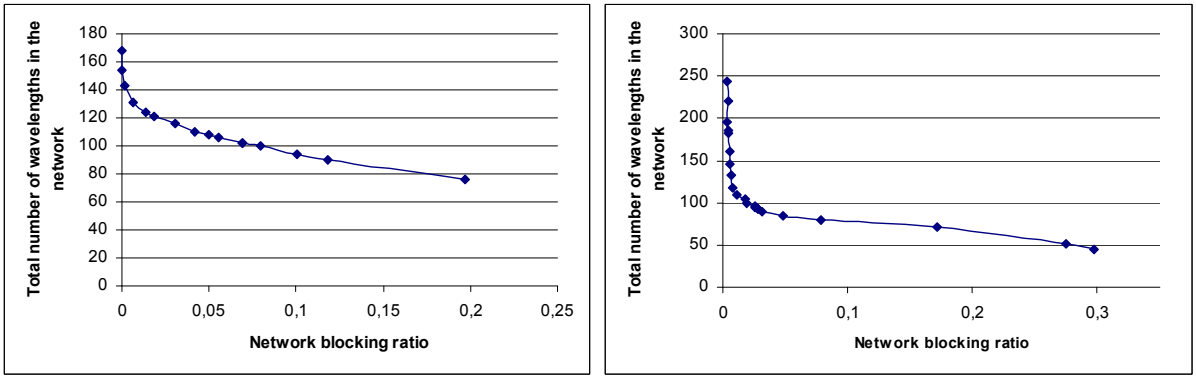

Fig. 13. The total number of wavelengths required to reach a given blocking rate of the network in case of "NSFnet" (left) and "Longhaul" (right).

Finally, the behavior of the third algorithm (see Section 2.2.3), which jointly optimizes the number of grooming ports and the number of wavelengths, is investigated. "NSFnet" reference network was used for this simulation. The value of TH_n was set to 0.4, while the value of TH_l was 0.1. The average inter-arrival time and the average holding time had a value of 30 and 18, respectively. The volume of the traffic can be arbitrary, of course, since the purpose of the algorithm is to find the appropriate network configuration for any given traffic set.

The algorithm started from both upper and lower states. In the *upper state* (corresponding to an *oversized network*) the network comprised 10 wavelengths on each link and the maximal number of grooming ports in every node. On the other hand, in the *lower state* (corresponding to an *undersized network*) there was only one wavelength on each link, but still the maximal number of grooming ports in the nodes. The algorithm found the same balanced state in both cases. There was only one wavelength difference on one of the links and one port number difference in two nodes between the two final states. However, the convergence time was significantly

different: in case of starting from the lower state, the algorithm found the balanced state in 11 steps; while, in case of starting from the upper state, it took only 8 steps.

Fig. 14 shows the trajectories of starting the algorithm from the two different initial configurations. Note that the convergence applies not only for the total number of wavelengths and the total number of grooming ports, but also for the wavelength and port number of individual links and nodes, respectively. As Fig. 14 illustrates, if the algorithm starts from an upper state, the convergence requires only 3 steps less, but the network reaches a configuration – very close to the balanced state – already after 3-4 steps.

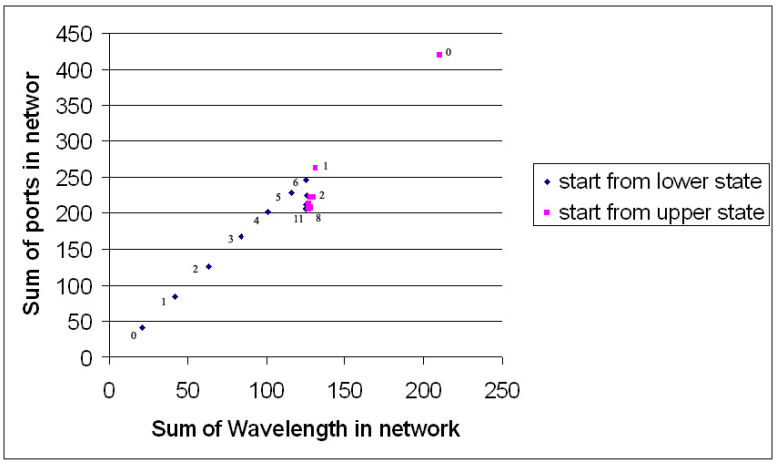

Fig. 14. The trajectories of the third algorithm in case of starting it form a lower state or an upper state.

The results of the algorithm, the so-called balanced state, can be seen in Fig. 15. The numbers on the links mean the required numbers of wavelengths, and the numbers on the nodes represent the required numbers of grooming ports. The thickness of the links and the radius of the nodes are also proportional to their capacity.

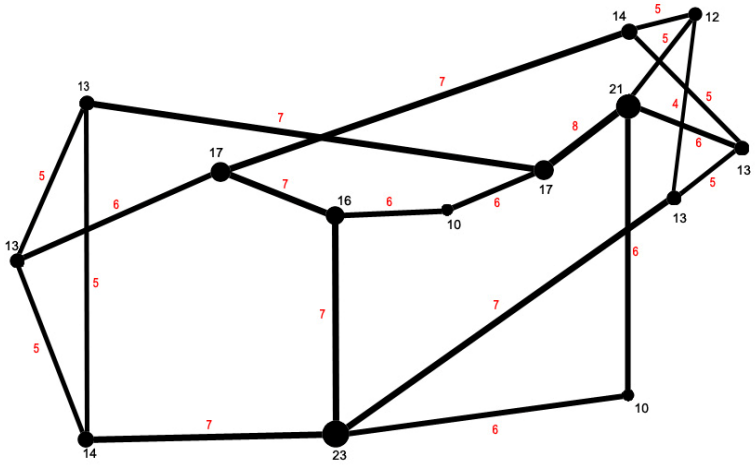

Fig. 15. The number of wavelengths and the number of grooming ports in the balanced state for "NSFnet" reference network (determined by the third algorithm)

2.3.1. Simulation Results for the Case with Protection

In this subsection the aim is to *compare different protection techniques* in terms of resource requirements. NSF-net network topology was used with uniform traffic demands between the nodes. A medium-sized network was chosen to shorten the simulation time. The traffic load was set – by adjusting the mean inter-arrival time and the mean holding time of the traffic – to keep the blocking ratio under an acceptable level in case of all protection schemes.

Firstly, the behavior of the algorithm optimizing the necessary number of grooming ports (see Section 2.2.1) is investigated. The initial configuration contained 14 wavelengths on each link, and the maximal number of grooming ports in every node. The volume of the traffic was set to keep the blocking rate of the network under 1%, thus the algorithm was started from a well oversized network and started to increase TH_n (i.e., allowing higher blocking in each node).

Fig. 16 (left) shows an exponential-like increase in the network-level blocking rate as the per node blocking rate threshold (TH_n) rises. The simulations resulted in similar exponential-like curves in all protection scenarios. The fact that the bandwidth requirement of shared protection – and especially dedicated protection – is higher than that of the unprotected case slightly appears in the curves. Furthermore, results suggest that the network blocking ratio is mostly affected by the per node blocking threshold, not by the protection technique.

It is more interesting to see, however, how many grooming ports are necessary to provide a given network blocking ratio in different protection scenarios (Fig. 16, right). *In this sense there is significant difference between the protected and unprotected scenarios. In case of dedicated protection, about two times more grooming ports are necessary in the network compared to the unprotected case. Shared protection needs only about one and a half times more ports than no-protection.*

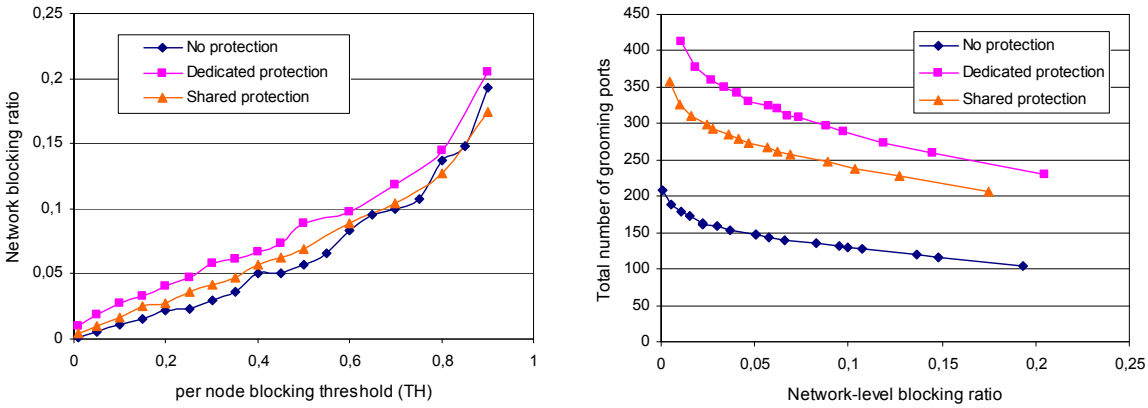

Fig. 16. Network blocking ratio as a function of the per node blocking threshold (left)
Total number of required grooming ports as a function of the network-level blocking ratio (right)

Similar conclusion can be drawn for all protection schemes: if moderate network blocking ratio is allowed, the number of grooming ports can be significantly reduced. All curves are falling rapidly close to the zero-blocking region. For example, if 5% blocking ratio is allowed instead of 1%, the necessary number of grooming ports can be reduced by 20%. This gain was found to be similar in all protection cases. Naturally, this gain is inexpressively high if we compare the required number of grooming ports in this case to the number of grooming ports in a full grooming capable network. The results also suggest that it is not wise to decrease the number of grooming ports after a certain point because the gain decreases; i.e., not much more cost can be spared, but the network blocking ratio rises.

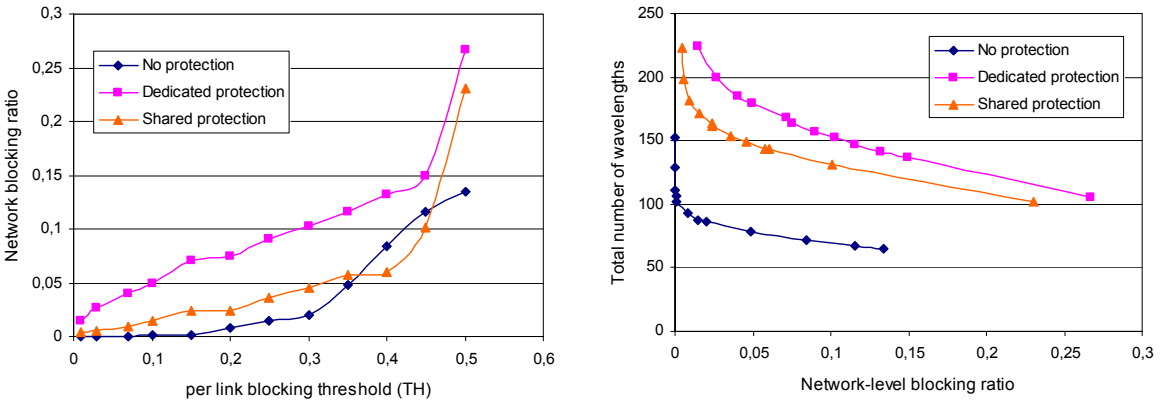

Fig. 17. Network blocking ratio as a function of the per link blocking threshold (left)
Total number of required wavelength as a function of network blocking ratio (right)

Fig. 17 shows the results of the second algorithm (see Section 2.2.2), which is used to determine the number of wavelengths. In the initial configuration there were 14 wavelengths on each link and the maximum number of grooming ports in every node as in case of grooming port dimensioning. The volume of the traffic was again adjusted to keep the blocking rate of the network *under 1% in all protection scenarios*. After that, the algorithm started to increase the per link blocking rate threshold (TH_l). The number of grooming ports was always set to the maximum in all nodes according to the current number of wavelengths; i.e., this time the lack of wavelengths meant the bottleneck, not the lack of grooming ports.

The results are by some means similar to the grooming port dimensioning case. However, Fig. 17 (left) shows a huge difference between the blocking curves belonging to different protection scenarios. Especially the dedicated protection case shows higher network blocking ratio for the same value of the per link blocking threshold. This phenomenon is due to the fact that the capacity of a link (i.e., the number of wavelengths) cannot be adjusted with such a fine granularity as the number of grooming ports. Thus a small difference in the per link blocking threshold may cause a large modification in the number of wavelengths. Consequently, the network blocking ratio can raise or fall suddenly.

The necessary number of wavelengths as a function of the network blocking ratio (Fig. 17, right) shows a similar picture as in the case of grooming port dimensioning. The curves are falling even more steeply close to the zero-blocking value, thus the sparing gain is higher than in the previous case. This sudden fall is also due to the rough granularity of the link capacities.

There is significant difference in the necessary number of wavelengths between the protected and unprotected cases. In case of dedicated protection, about two times more wavelengths are necessary in the network compared to the unprotected case. Shared protection needs only about one and a half times more wavelengths. The results showed that, in protected networks, *the gains* in the required number of grooming ports and in the required number of wavelengths *are similar*.

Finally, the third algorithm (see Section 2.2.3) was investigated, which optimizes the number of grooming ports and the number of wavelengths at the same time. The value of TH_n was set to 0.4, while the value of TH_l was 0.1. The average inter-arrival time and the average holding time had a value of 300 and 250, respectively.

The objective was to compare the number of grooming ports, the number of wavelengths in the network in balanced state and also the number of iterations to reach this balanced state.

I tried starting the algorithm again both from an upper (oversized network) and a lower state (undersized network) as well.

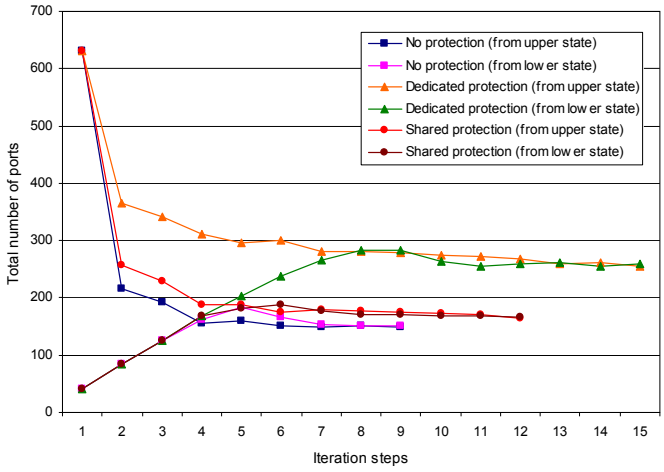

Fig. 18. Total number of grooming ports in the network in the iteration steps

The algorithm found the appropriate balanced state in all protection scenarios; no matter whether from lower or upper state it was started. In case of shared protection and no protection there was negligible difference between the two appropriate balanced states reached from upper state and from lower state. Only the total number of grooming ports was slightly different. In case of dedicated protection, the results were interesting. Neither the balanced state reached from the upper state, nor the balanced state reached from the lower state was stable. More interestingly both iterations ran into the same oscillation trajectory. Of course, the amplitude of the oscillation was very low, so one of the oscillation states can be considered as the balanced state.

If I start the iterative algorithm from upper state, it reaches the balanced state faster – irrespectively of the protection scheme, or at least approaches the balanced state in a few steps. From lower state it takes more iteration steps to reach the balanced state. The number of required iteration steps slightly depends on the applied protection scheme as well. In case of protection the network needs more iteration steps to reach the balanced state. This fact is likely not only the consequence of the more complex routing, but also of the increased traffic load. Higher traffic load causes longer iteration in itself.

Fig. 18 shows how the total number of grooming ports in the network is changing during the iteration steps, while Fig. 19 shows the altering of total number of wavelengths. In the case of shared protection, both the number of wavelengths and the number of grooming ports is less than expected.

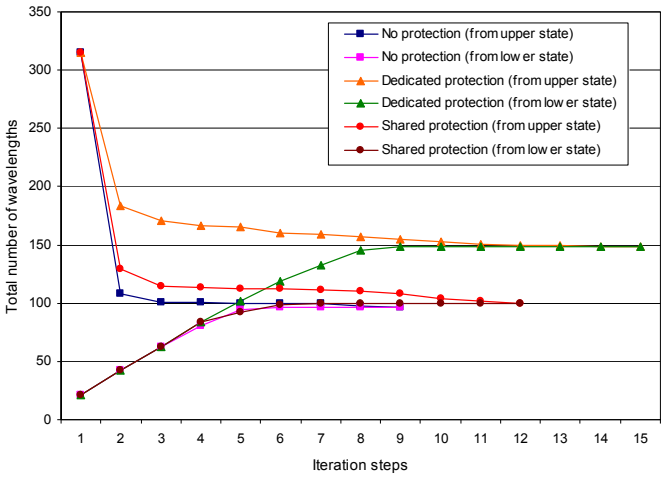

Fig. 19. Total number of wavelengths in the network in the iteration steps

2.4. Conclusion

I proposed algorithms for optimizing the number of grooming ports in each node and the number of wavelengths on each link in partial grooming capability networks. These algorithms use iterative simulations to dimension the network for a predefined traffic.

The resulting network configuration is homogeneous from the aspect that the per link and per node blocking rates are approximately the same on all links and in nodes, respectively. The proposed algorithms, as another advantage, do not assume any central intelligence and may run in a distributed way.

Applying the algorithms, I compared the resource requirement of different protection schemes and showed that significant amount of equipment can be saved by proper dimensioning.

3. Signal Power Based Routing

In both metropolitan optical networks (MON) and long haul optical networks (LHON) the signal quality is often influenced by the physical impairments. Therefore a proper impairment based routing decision is needed. In this chapter I propose new routing and wavelength assignment (RWA) methods where the control plane has influence on the signal power of the Wavelength Division Multiplexed (WDM) channels. Recently in nearly all kinds of reconfigurable optical add-drop multiplexers (ROADM) the signal power can be tuned via variable optical attenuators (VOA) from the control plane.

Increased data management capabilities on individual wavelengths are also needed to exploit the benefits of ROADM in metro and backbone WDM networks. For instance, ROADM rings are very sensitive to topology changes and need strict monitoring and control of wavelength power to keep the system in balance. The real innovation lies in the system engineering related to the ROADM function, addressing per-wavelength power measurement and management, and per-wavelength fault isolation. Almost every optical system vendor has commercial ROADM with wavelength monitoring functions (see e.g. [62]-[65]).

The next step towards a fully reconfigurable WDM optical network is the deployment of tunable Small Form-factor Pluggable (SFP) interfaces, where the wavelength allocation is modified according to the network changes. The tunable dispersion compensation elements mean another innovation. Nowadays these ready-made products can be purchase [66], [67].

The evolution of optical networks seems to tend towards a fully reconfigurable network where the control and the management plane (CP and MP) have new functions, such as determining the signal quality, tuning the wavelength frequency, setting dispersion compensation units, and – by using variable optical attenuators – setting the channel powers. Of course, traditional functions (such as routing) remain the main function of the CP and MP. Routing and Wavelength Assignment (RWA) plays a central role in the control and management of optical networks. Many excellent papers deal with the design, configuration, and optimization of WDM networks (e.g., [68], [69], or Section 1.1.4). However, these papers do not consider the physical parameters of the fully reconfigurable optical network in the RWA method.

The authors of [80] proposed a hierarchical RWA model for the provisioning of high-speed connections, where physical effects are estimated and taken into account in lightpath computation. In [77] a general OSNR network model was developed and the OSNR optimization problem was formulated as a non-cooperative game between channels. Several other papers are taking into account physical constraints in routing ([81]-[86]). However, there is a difference between these and the method proposed in this chapter. These methods either perform wavelength allocation in a heuristic way thought to be beneficial in terms of physical effects, or they calculate the physical metrics in advance, before routing the demands.

In this Chapter I propose a new ILP based RWA algorithm where the control plane handles the routing and the signal power allocation jointly. The proposed method can be used in existing WDM optical networks where the nodes support signal power tuning.

I give the exact integer linear programming (ILP) formulation of the method for both single and multilayer networks. In the first case (Section 3.3), I assume that no signal regeneration is allowed along the path. While in the more complex multilayer case (Section 3.4), 3R signal regeneration, grooming and wavelength conversion can all be done in the electronic layer.

The algorithm finds the global optimum, if it exists, for a certain network topology, physical constraints and demand set.

3.1. Physical Considerations

Currently the power of certain channels within a fiber is set to equal levels. This is one of the remaining practices of point-to-point optical networks. Naturally using this kind of channel power allocation is a technical simplification. The other reason for using the same channel powers is the nonlinear effects which in this case have the smallest impact on the signal quality [70]-[76]. The new idea is to use different channel powers according to the length of the path of the connection request to fulfill the optical signal to noise ratio (OSNR) to achieve bit-error free detection. E.g., for a long distance connection, we can increase the signal power of the dedicated wavelength, while for a short distance connection lower wavelength power is satisfactory. Due to the nonlinear effects there is an upper bound on the totally inserted optical power in one fiber, which has to be observed. Consequently, if we increase the signal power of one channel, the signal powers of other channels in the same optical fiber have to be decreased. In recent years, there have been some publications which apply the same idea of different channel power allocation. This problem was considered in [77]-[79].

3.1.1. Physical Feasibility

As mentioned before, the proposed algorithms use different channel powers in the same optical fiber. This approach introduces many new problems related to physical feasibility. All physical effects were already investigated using equal channel power allocation. The only difference in our case is that the impacts of the effects are different for each channel, since the signal powers are different. In case of linear effects, the signal power has no influence on the dispersion and its compensation schemes. The only linear effect which has signal power dependency is the crosstalk in the nodes. I assume that using the well-known power budget design process the effects of the crosstalk can be eliminated.

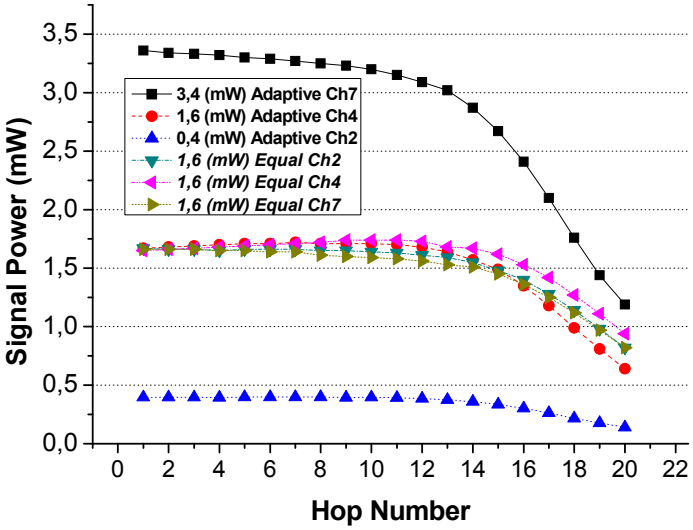

Fig. 20. Signal power dependency from the number of EDFAs in chain

More interesting question is how the Erbium Doped Fiber Amplifiers (EDFAs) react to the use of different channel power allocations. For this purpose, I made simulations using the VPI TMM/CM Version 7.5 simulation tool [87]. I assumed a system with 8 channels which are multiplexed and then amplified using EDFA rate propagation modules. I aimed at investigating the difference between the uniform and the adaptive channel power allocations. Following the amplifier an attenuator was placed with attenuation equal to the gain of the EDFA. The results can be seen in Figure 1. On the horizontal axis the number of hops is plotted, i.e., the number of EDFAs and attenuators connected in a chain. On the vertical axis the "powers of ones" (the power of the signal level of one) is plotted, where the "power of ones" is the power in case the transmitter is switched on. The first three curves represent the adaptive channel power allocation where the "powers of ones" are set to 0.4 mW, 3.4 mW and 1.6 mW, for channel two, seven and all other channels, respectively. These values are used because applying adaptive signal power routing on the COST 266 topology these values are the maximum, minimum and average channel powers, respectively. The values are plotted for channel two, four and seven. As it was expected, if the number of hops increases, after a certain number of inline amplifiers "the power of ones" decreases, i.e., the signal quality becomes very poor. The interesting thing is that if the number of consecutive amplifiers is lower than this value, "the power of ones" remains nearly the same. Because of this behavior, the EDFA supports adaptive signal allocation. To compare the performance of the two power allocation schemes, exactly the same simulations were made as described formerly but without allowing different channel powers. In this case, the "powers of ones" were set to the same value (1.6 mW) for all channels (channel 2, 4 and 7). See the last three curves in Figure 1. It can be seen that the three curves are very close to each other. The only difference is due to the wavelength dependency of the EDFA. It is interesting that the results obtained for adaptive and uniform allocation schemes are nearly the same. Difference can be seen only for high numbers of hops, where the signal quality is very poor. These results lead to the conclusion that the so far deployed EDFAs behave similarly in case of both uniform and non-uniform channel power allocations in a single optical fiber.

The other interesting question is about the nonlinear effects, since these effects highly depend on the used signal powers. The only solution is to limit the signal power inserted in one optical fiber. This must be done in both allocation schemes. Another problem is the maximum allowed difference between the maximum and the minimum channel powers. In the current case, this is an input parameter of the algorithm. Determining this value is a hard task and is out of the scope of my work. Finally to conclude: according to the results, adaptive signal power allocation scheme can be implemented in optical systems deployed so far. Moreover, the author knows

existing WDM optical networks operating without any error, where different channel powers – though not intentionally – are used, since the power tuning was not performed for the inserted channels in the OADMs.

3.1.2. Relation between Channel Power and Maximum Allowed Distance

To investigate the relation between the signal power and the maximum allowed distance, I consider a noise limited system where other physical effects can be taken into account as power-penalty:

Considering a chain of amplifiers the OSNR of the end point can be calculated as follows [88]:

$$OSNR_{dB}=58+P_{in}-\Gamma(dB)-NF_{dB}-10\cdot\log N-M \qquad (0.8)$$

where noise figure (*NF*) is the same for every amplifier and the span loss (*Γ(dB)*) is the same for every span. P_{in} is the input power in dBm, *M* is the margin for other physical effects, and *N* is the number of spans. I assume that there is an inline amplifier in every *l* km. This means that if the length of the link is *L, N* is $\lfloor L/l \rfloor$, i.e. the integer part of the division.

Having in mind that

$$Q_{dB}=OSNR_{dB}+10\cdot\log\left(\frac{B_0}{B_e}\right) \qquad (0.9)$$

where B_0 is the optical bandwidth and Be is the electronic (digital) bandwidth of the receiver.

The logarithmic Q_{dB} and the linear Q have the following relation:

$$Q_{dB}=20\cdot\log(Q) \qquad (0.10)$$

Substituting equation (0.9) and (0.10) into (0.8), a linear relation can be obtained between the maximum allowable distance and the signal power.

$$L=L_c\cdot P_{mW} \qquad (0.11)$$

where P_{mW} is the input power in mW, *L* is the maximum allowable distance, and L_c is the linear factor between them.

$$L_c=1/10^{\left(\frac{20\cdot\log Q-10\cdot\log\left(B_0/B_e\right)-58+\Gamma_{dB}+NF+M}{10}\right)} \qquad (0.12)$$

For typical constant values used in telecommunications the L_c is between 500 and 2000.

The effects of an optical node on the signal quality are similar to the impact of an about 90 km long optical fiber, since it has nearly the same attenuation. Using the approximation mentioned previously in the routing algorithm, when the physical effects are taken into consideration, each node was substituted with a 90 km long optical fiber. Naturally, more accurate models [89], [90] can be implemented for characterizing the networks.

3.2. Network and Routing Model

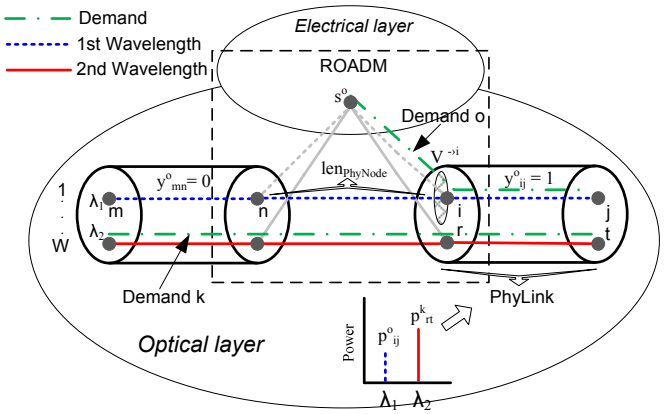

Fig. 21. Model of the switching device with optical and electronic switching capabilities, grooming and 3R regeneration (in the electronic layer)

The model of an ROADM switching device assumed in the simulations is shown in Fig. 21. The device can perform optical switching and – through the electronic layer – wavelength conversion, grooming and 3R signal regeneration. The device illustrated in Figure 2 has an input and an output interface with one *physical link* (or

fiber) connected to each. Each physical interface supports two *wavelengths* (W=2), marked by blue dashed and red solid lines. The signal powers of the wavelengths in the right hand side physical link are different, p_{ij}^{o} and p_{rt}^{k} – as shown by the small subfigure. The example also comprises two demands (indicated by dash-dotted line): *demand k* passes through the switch in the optics, while *demand o* originates in this device, in the electronic layer (s^o). A certain length of fiber ($len_{PhyNode}$) is assigned to each internal edge, e.g., edge *(n, i)*, which corresponds to the amount of signal distortion that the switching functionality introduces in the demand path. The edges representing O/E or E/O conversion are shown by grey color.

In the routing, I assume that WL conversion, grooming and signal regeneration are possible only in the electronic layer, and that the *noise and the signal distortion accumulate along the lightpath*. Actually, re-amplification, re-shaping, and re-timing – which are collectively known as 3R regeneration – are necessary to overcome these impairments. Although 3R optical regeneration has been demonstrated in laboratories, only electronic 3R regeneration is economically viable in current networks.

The constraints of maximum input power in each fiber, and maximum allowed distance as a function of the input power of the lightpath have to be met.

In addition, I differentiate two routing cases and propose an ILP formulation for each (presented in 3.3 and 3.4).

In the first case (referred to as *single-layer network*), I assume that a whole lightpath is assigned to each demand from the source to the destination node. The signal enters into the optical layer at the source node and leaves it at the destination node. Wavelength conversion, grooming or regeneration is not allowed elsewhere along the path.

In the second case (referred to as *multilayer network*), the path of a demand may consist of several lightpaths, i.e. it can enter and leave the electronic layer multiple times if necessary and efficient. In addition in the second case grooming is also applicable.

3.3. ILP formulation of Signal Power Based Routing in Single-Layer Networks

In this section, I introduce the ILP formulation of Signal Power based Routing for single-layer networks.

3.3.1. Constants

The WL graph contains nodes (V) and directed edges or arcs (A). Edge *(i, j)* represents one edge in the WL graph. $V^{\rightarrow i}$ (illustrated by a checkered ellipse in Fig. 21) and $V^{i\rightarrow}$ represent incoming and outgoing edges of node i, respectively. Symbol A^{sw} denotes the set of edges in the WL graph representing switching function inside a physical device; other edges represent wavelengths of a physical link (A^{pl}). The set of demands in the network is denoted by O.

$$P_{pl}^{max} = \text{4-20 dBm typically 10dBm} \qquad (0.13)$$

Constant P_{pl}^{max} means the upper limit of total power in physical link *pl* in dBm.

$$len_{ij} \qquad (0.14)$$

Constant len_{ij} is the length of the physical link which the wavelength belongs to.

$$len_{PhyNode} = \text{90 km typically} \qquad (0.15)$$

Constant $len_{PhyNode}$ corresponds to the length of the fiber a switching device induces to the path of the demand.

$$L_c = 1200 \qquad (0.16)$$

Constant L_c is the factor of the linear relation between the input power of a demand (in mW) and the maximum distance the signal is allowed to reach.

$$\alpha \qquad (0.17)$$

Constant α expresses tradeoff between optimization objectives: minimal routing cost or minimal power.

$$s^o,\ t^o \qquad (0.18)$$

Symbols s^o and t^o represent source and target of demand o.

$$\beta = \frac{n}{W} \cdot P_{pl\ lin}^{max} \qquad (0.19)$$

Constant β is the maximum allowed signal power for one channel in mW, where n is a real number between 1 and W, and W is the number of wavelengths in a fiber.

$$P_{pl\ lin}^{max} = 10^{\left(P_{pl}^{max}/10\right)} \qquad (0.20)$$

Constant $P_{pl\ lin}^{max}$ means the upper limit of total power in physical link *pl* in mW.

3.3.2. Variables

$$p^o \in \left[0..\frac{\beta}{P_{pl\ lin}^{max}}\right],\ \forall o \in O \qquad (0.21)$$

Variable p^o denotes the input power of demand *o* divided by $P_{pl\ lin}^{max}$.

$$p_{ij}^o \in \left[0..\frac{\beta}{P_{pl\ lin}^{max}}\right],\ \forall (i,j) \in A,\ \forall o \in O \qquad (0.22)$$

Variable p_{ij}^o means the power of demand *o* on edge *(i, j)* divided by constant $P_{pl\ lin}^{max}$.

$$y_{ij}^o \in \{0,\ 1\},\ \forall (i,j) \in A,\ \forall o \in O \qquad (0.23)$$

Variable y_{ij}^o tells whether demand *o* uses edge *(i, j)*.

(E.g., variable $y_{mn}^o = 0$ in Fig. 21 because demand *o* does not pass through edge *(m, n)*, which represents the first wavelength. On the other hand, $y_{ij}^o = 1$ because demand *o* does use edge *(i, j)*.)

3.3.3. Objective Function

$$\alpha \cdot \sum_{\forall o \in O} \sum_{\forall (i,j) \in A / A_{sw}} y_{ij}^o + (1-\alpha) \cdot \sum_{\forall o \in O} p^o \qquad (0.24)$$

The objective function expresses that the sum of the used edges should be minimized together with the sum of input powers of demands. If we want to minimize the total cost of the routing, constant cost factors should be assigned to each edge.

Constant α decides whether optimization emphasis is on minimal routing cost (α is close to 1) or on minimal input power (α is close to zero).

3.3.4. Constraints

$$\sum_{\forall o \in O} \sum_{\forall (i,j)^{pl}} p_{ij}^o \le 1,\ \forall pl \in PhyLinks \qquad (0.25)$$

$$p_{ij}^o \le y_{ij}^o,\ \forall (i,j) \in A,\ \forall o \in O \qquad (0.26)$$

$$\sum_{\forall j \in V^{\to i}} p_{ji}^o - \sum_{\forall k \in V^{i \to}} p_{ik}^o = \begin{cases} -p^o & \text{if } i = s^o \\ 0 & \text{if } i \notin \{s^o, t^o\}, \\ & \forall i \in V,\ o \in O \\ +p^o & \text{if } i = t^o \end{cases} \qquad (0.27)$$

$$\sum_{\forall j \in V^{\to i}} y_{ji}^o - \sum_{\forall k \in V^{i \to}} y_{ik}^o = \begin{cases} -1 & \text{if } i = s^o \\ 0 & \text{if } i \notin \{s^o, t^o\}, \\ & \forall i \in V,\ o \in O \\ +1 & \text{if } i = t^o \end{cases} \qquad (0.28)$$

$$\sum_{\forall o \in O} y_{ij}^o \le 1,\ \forall (i,j) \in A \qquad (0.29)$$

$$\sum_{\forall (i,j) \in A^{sw}} y_{ij}^o \cdot len_{PhyNode} + \sum_{\forall (i,j) \in A^{pl}} y_{ij}^o \cdot len_{ij} \le$$
$$\le L\left(p^o\right) = L_c \cdot p^o \cdot P_{pl\ lin}^{max},\ \forall o \in O \qquad (0.30)$$

3.3.5. Explanation

Constraint (0.25) ensures that the sum power of demands traversing a physical link (fiber) cannot exceed the maximum allowed power of that link. The first sum on the left enumerates all demands, while the second one all WL graph edges belonging to a certain fiber. Constraint (0.26) guarantees that if the power of demand *o* in edge (i,j) is larger than zero, then edge (i,j) is used by demand *o* (i.e., the value of the y indicator variable is set to 1). Constraints (0.27) and (0.28) express the flow-conservation constraint of the power and the y decision variables, respectively, for every demand. Constraint (0.27) also ensures that a lightpath has the same signal power along the whole way. Constraint (0.29) guarantees that a given edge can be used by only one demand. Constraint (0.30) ensures that the total length of demand *o* should be less than the distance allowed by the input power of demand *o*. The left side of the equation sums the products of the *y* decision variables and the related fiber lengths. This way the total length of fibers (and signal penalties induced by switching functionalities and expressed by fiber length as well) used by demand *o* can be calculated. On the right side the maximum allowed distance is computed by multiplying the signal power variable by constants.

3.4. ILP Formulation of Signal Power Based Routing in Multilayer Networks

In this section, I introduce the ILP formulation of Signal Power based Routing for multilayer networks, which can provide optimal solution for the joint problem of RWA with grooming and of determining the signal powers of lightpaths.

3.4.1. Variables and Constants

The symbols are similar to those in 3.3.1. In addition, the set of lightpaths is denoted by *L*. A path in the WL graph is considered as a lightpath if it goes only in the optical layer without going up to the electronic layer. A lightpath does not traverse any electronic node except for the source and destination nodes.

$$p_{EF} \in \left[0..\frac{\beta}{P_{pl\ \ lin}^{max}} \right],\ \forall (E, F) \in L \qquad (0.31)$$

Variable p_{EF} denotes the input power of lightpath *(E, F)* divided by constant $P_{pl\ \ lin}^{max}$.

$$p_{ij}^{EF} \in \left[0..\frac{\beta}{P_{pl\ \ lin}^{max}} \right],\ \forall (i, j) \in A,\ (E, F) \in L \qquad (0.32)$$

Variable p_{ij}^{EF} means the power of lightpath *(E, F)* on edge *(i, j)* divided by constant $P_{pl\ \ lin}^{max}$.

$$x_{\substack{EF \\ ij}}^{o} \in \{0,\ 1\},\ \forall (i, j) \in A,\ o \in O,\ (E, F) \in L \qquad (0.33)$$

Variable $x_{\substack{EF \\ ij}}^{o}$ expresses whether demand *o* uses lightpath *(E, F)* on edge *(i, j)* or not.

$$y_{ij}^{EF} \in \{0,\ 1\},\ \forall (i, j) \in A,\ (E, F) \in L \qquad (0.34)$$

Variable y_{ij}^{EF} indicate whether lightpath *(E, F)* uses edge *(i, j)* or not.

$$y_{ij} \in \{0,\ 1\},\ \forall (i, j) \in A \qquad (0.35)$$

Variable y_{ij} expresses whether edge *(i, j)* is used by the routing or not.

The same constants and calculated constants are used that were defined in 3.3.1.

3.4.2. Objective Function

Minimize:

$$\alpha \cdot \sum_{\forall (i,j) \in A} y_{ij} + (1-\alpha) \cdot \sum_{\forall (E,F) \in L} p_{EF} \qquad (0.36)$$

The objective function (0.36) expresses that the routing cost (including network resources) should be minimized together with the total of signal powers. If we want to minimize the sum cost of the routing, constant cost factors should be assigned to each edge. Constant α decides whether optimization emphasis is on minimal routing cost (α is close to 1) or on minimal signal power (α is close to zero).

3.4.3. Constraints

$$\sum_{\forall (i,j)\in pl,\ (E,F)\in L} p_{ij}^{EF} \le 1,\ \forall pl \in PhyLinks \tag{0.37}$$

$$x_{ij}^{o\ EF} \le y_{ij}^{EF} \le y_{ij},\ \forall o \in O,\ i, j \in V,\ (E, F) \in L \tag{0.38}$$

$$y_{ij}^{EF} \le \sum_{\forall o\in O,} x_{ij}^{o\ EF},\ \forall (i, j) \in A,\ (E,F) \in L \tag{0.39}$$

$$y_{ij} \le \sum_{\forall (E,F)\in L} y_{ij}^{EF},\ \forall (i, j) \in A \tag{0.40}$$

$$p_{ij}^{EF} \le y_{ij}^{EF},\ \forall i, j \in V,\ (E, F) \in L \tag{0.41}$$

$$\sum_{\forall j\in V^{\to i}} p_{ji}^{EF} - \sum_{\forall k\in V^{i\to}} p_{ik}^{EF} = \begin{cases} -p_{EF} & \text{if } i = E \\ 0 & \text{if } i \notin \{E, F\}, \\ & \forall i \in V,\ (E, F) \in L \\ +p_{EF} & \text{if } i = F \end{cases} \tag{0.42}$$

$$\sum_{\forall j\in V^{\to i}} \sum_{\forall (E,F)\in L} x_{ji}^{o\ EF} - \sum_{\forall k\in V^{i\to}} \sum_{\forall (E,F)\in L} x_{ik}^{o\ EF} = \begin{cases} -1 & \text{if } i = s^o \\ 0 & \text{if } i \notin \{s^o, t^o\},\ \forall i \in V,\ o \in O \\ +1 & \text{if } i = t^o \end{cases} \tag{0.43}$$

$$\sum_{\forall E,\ F} y_{ij}^{EF} \le 1,\ \forall (i, j) \in A \tag{0.44}$$

$$\sum_{\forall o\in O,\ (E,F)\in L} x_{ij}^{o\ EF} \cdot b^o \le B,\ (i, j) \in A \tag{0.45}$$

$$\sum_{\forall (i,j)\in A_{sw},} y_{ij}^{EF} \cdot len_{PhyNode} + \sum_{\forall (i,j)\in A_{pl}} y_{ij}^{EF} \cdot len_{ij} \le \le L(p_{EF}) = L_c \cdot p_{EF} \cdot P_{pl\ lin}^{max},\ (E, F) \in L \tag{0.46}$$

3.4.4. Explanation

Constraint (0.37) explains that total power of lightpaths traversing a physical link or fiber (denoted by *pl*) cannot exceed the maximum allowed power of that link. The left side of constraint (0.37) calculates the sum of the power of lightpaths going through those edges that belong to physical link *pl*. Constraint (0.38) is straightforward: it expresses that edge *(i, j)* is used by lightpath *(E, F)* if any of the demands – multiplexed into lightpath *(E, F)* – uses that edge. Similarly it also tells that edge *(i, j)* is used by the routing if any of the lightpaths uses that edge. Constraint (0.39) states that edge *(i, j)* is used by lightpath *(E, F)* only if it is used by at least one demand. I.e., lightpath *(E, F)* does not use any unnecessarily edge *(i, j)*. Similarly, constraint (0.40) expresses that edge *(i, j)* is used by the routing only if it is used by at least one lightpath. I.e., no lightpath is created unnecessarily. Constraints (0.39) and (0.40) are optional, since these rules are implicitly expressed by the objective function. Constraint (0.41) simply means that if the power of a lightpath on an edge is greater than zero, then that edge is used by the lightpath. Constraint (0.42) assures that the signal power of a lightpath – assuming amplification (e.g., EDFA) – is the same along the whole path. Constraint (0.43) expresses flow conservation constraint for demands. Constraint (0.44) assures that each edge is used by at most one lightpath. Constraint (0.45) expresses the grooming constraint, i.e., the sum bandwidth of multiplexed demands cannot exceed wavelength capacity. Constraint (0.46) expresses the relation between the physical distance traversed by the lightpath and the signal power of the lightpath.

3.5. Simulation Results

It is a very hard task to illustrate the efficiency of the algorithm since it gives obviously better results than the traditional RWA algorithms. This is due to the additional degree of freedom, namely the tunability of the signal power. In this section, I illustrate some of the benefits of the algorithm having in mind that for different input parameters the results would be slightly different.

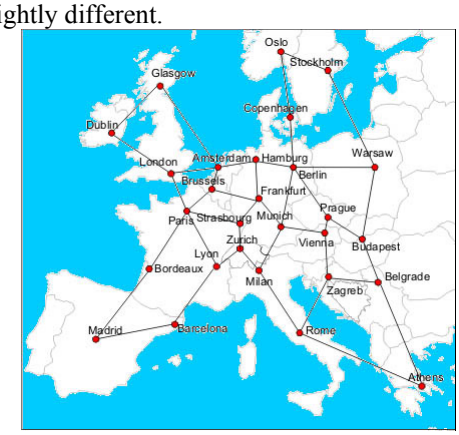

Fig. 22. COST 266 European reference network topology

In the simulations, the well-known COST 266 reference network [92] was used. The nodes are fully optical nodes and the signal cannot be 3R regenerated or converted into the electronic layer once it has entered the optical layer (i.e. the single-layer case is assumed, since the multilayer case proved to be unsolvable for even moderate size networks). The demands were generated randomly. The single layer routing scheme (from Section 3.3) was applied. The constants of the routing algorithm were as described in section 3.3.1.

ILOG CPLEX solver [93] was applied to solve ILP instances and obtain results.

To present the benefits of the algorithm, the maximum number of demands which can be routed in one network scenario was calculated. The absence of solution can have two reasons: the RWA does not succeed or the distance between the source and destination node is too long (i.e., the signal quality will be inadequate). It has to be mentioned that the proposed algorithm finds the global optimum of the routing problem, which is an NP-hard problem. Therefore, in some cases, to find the maximum demands which can be routed takes long time, approximately one week for the COST 266 network with 8 wavelengths ($W = 8$) and $n = 8$ – where n means the maximum allowed deviation of signal power from the traditional power allocation scheme (see Equation (0.19)). The "maximum number of routed demands" means the number of successfully routed demands from a randomly generated demand set. If a certain number of demands could be routed, the algorithm increases the number of demands and routes it again. This process continues until it is not possible to route more demands anymore. This way, it is possible to find the maximum number of demands which can be routed. The bandwidths of the demands were equal to the capacity of one channel. The source and destination pairs were chosen randomly. This timescale problem is not a significant drawback of the proposed algorithm since in real networks this kind of routing problem will not occur. Finding the global optimum (e.g., for COST 266 network with 8 wavelengths, $n = 1.5$ and 60 demands), takes approximately 10 minutes, which is a really fast RWA solution.

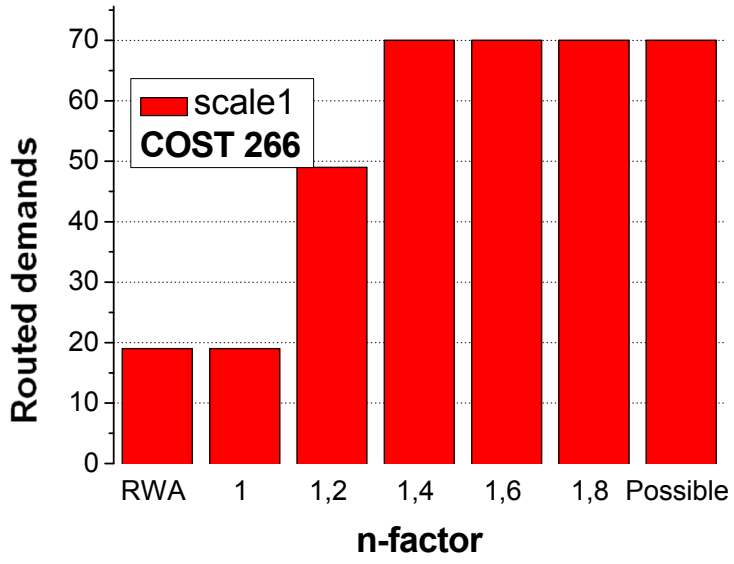

Fig. 23. Maximum number of routed demands versus n-factor parameter in case of COST 266 topology

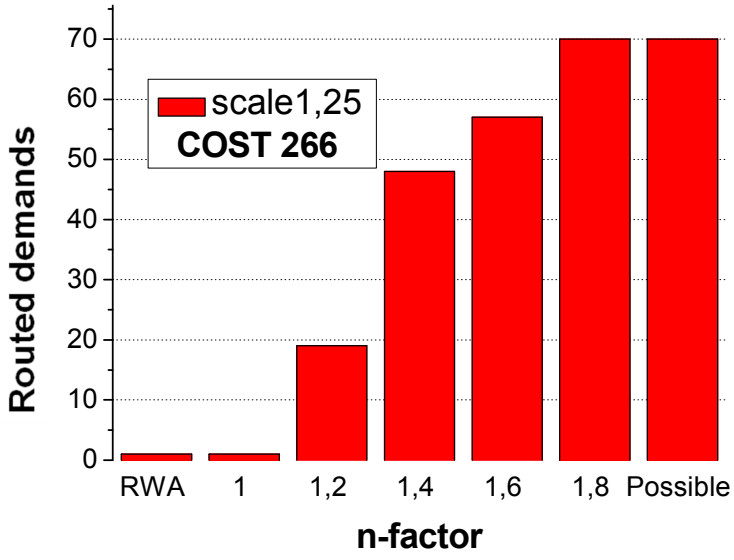

Fig. 24. Maximum number of routed demands versus n-factor parameter in case of COST 266 topology, scale 1.25

The proposed algorithm is compared to the traditional RWA algorithm (Fig. 23 and Fig. 24). The y-axis shows the maximum number of routed demands, while the x-axis shows the used routing schemes. *RWA* means that we used the traditional routing scheme where each channel has the same signal power. The $n = 1$ routing scheme is similar to the *RWA* routing scheme. The only difference is that in case of $n = 1$ the channel powers can be lower than the average of the powers. In the *RWA* case, this variation is not allowed. In $n > 1$ cases, the proposed routing algorithm is used with n equal to the depicted numbers. The result marked as "possible" means the number of maximum routed demands in a case when physical effects are neglected. The *scale parameters* mean that the lengths of all network links are modified by multiplying the original lengths with the *scale parameter*. In Fig. 23, the scale is 1.0, i.e., the original link lengths (geographical distances) are used. In Fig. 24, the scale parameter is 1.25.

In Fig. 23 and Fig. 24, it can be seen that the traditional RWA algorithm can route 19 and 1 demands, respectively. While, by increasing the n-factor, more and more demands can be routed until a limit is reached, where the RWA problem is infeasible in itself (without considering physical effects).

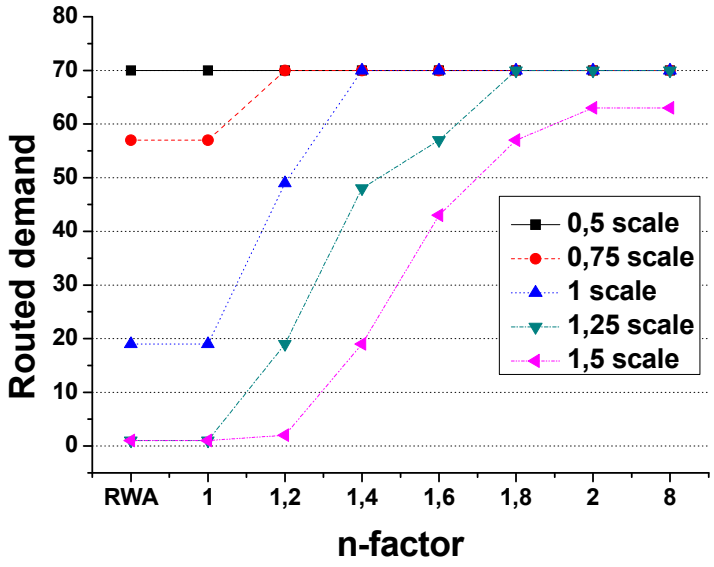

Fig. 25. Maximum number of routed demands versus n-factor parameter in case of COST 266 topology, for different scale parameters

In Fig. 25, the network size was scaled the way as mentioned before. Each line represents a scaled network topology. The meanings of the axis are the same as in Fig. 23 and Fig. 24. The only difference is between the last data columns. In Fig. 23 and Fig. 24 the last set of data is marked as "*possible*" where the influences of the physical effects are neglected. In case of Fig. 25 there is no such kind of results. Instead, we compared with a high *n*-factor value ($n=8$). It can be seen that while increasing the scale of the network the maximum number of demands which can be routed decreases. This is due to the fact that increasing the lengths of the network increases the influence of the physical effects as well. To have the same number of routed demands, we have to increase the range where the signal power can be tuned, i.e. the n-factor. The other interesting property is the case of the 1.5 times scaled topology: while increasing the n-factor 70 demands (as in the other cases) cannot be routed. Constraint (0.25) is responsible for this behavior. Constraint (0.25) expresses that the sum power of demands traversing a fiber cannot exceed the maximum allowed power of that link. Therefore the critical nonlinear region of the signal power will not be reached.

To investigate the dependency of the proposed method on the number of wavelengths, simulations were made using the COST266 network topology and different number of wavelengths (see Fig. 26). The figure shows that while increasing the number of channels the maximum number of routed demands is increasing. This behavior is as expected when solving the RWA problem. The interesting property is that if the number of wavelengths is doubled and the n-factor is high enough, the maximum number of routed demands is more than double in each case. This behavior is due to the way how the proposed algorithm works. If there are more wavelengths, there are also more possible variations how the signal power can be allocated. Consequently, if the number of wavelengths is increased, the performance of the proposed algorithm will improve. However, as it was mentioned before, for higher number of wavelengths (32-64) it takes very long time to find the maximum number of demands which can be routed, since it needs many tests to find the exact number of demands which can be routed. The other timescale problem occurs when the number of demands is very close to the maximum number of demands which can be routed. This kind of simulations can have long running time, more than a week. In other cases the running time of the algorithm has a timescale of minutes even for higher number of wavelengths.

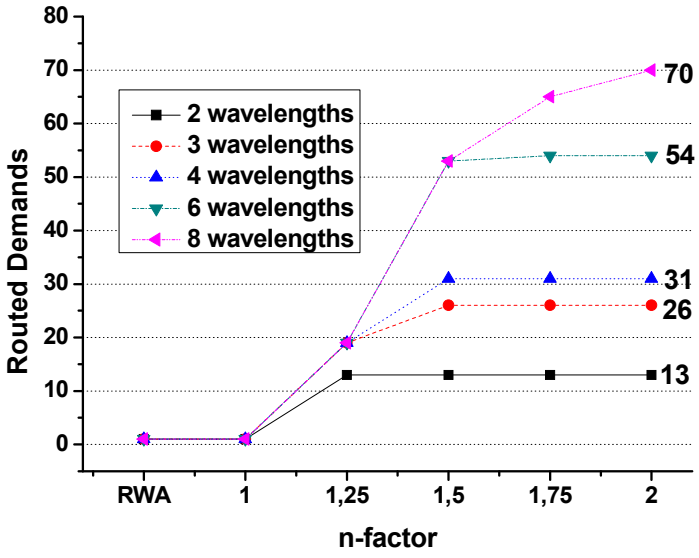

Fig. 26. Maximum number of routed demands versus n-factor parameter in case of COST 266 topology, for different wavelength numbers

3.6. Conclusion

In this chapter I presented new RWA algorithms where the power of WDM channels can be adjusted. The proposed algorithms perform joint optimization of routing (RWA) and of determining signal powers of WDM channels. The proposed methods can be used in existing WDM optical networks where the nodes support signal power tuning. I gave the exact ILP formulation of the problems to find the global optimum. In the simpler single layer case, it is not allowed to use the electronic layer at all along a path, except for the source and destination nodes, while in the more complex multilayer case electronic layer can perform 3R regeneration, grooming and wavelength conversion. In the second case, a full joint optimization of RWA with grooming (according to a given demand set) and with determining the power of lightpaths (observing physical constraints) is carried out. The multilayer optimization proved to be too complex for even small networks, while the single layer ILP formulation is practically applicable for moderate size networks. However, it is still worth looking for fast heuristic approaches. For such heuristic methods, the ILP based optimal solutions can be regarded as a baseline for comparison.

4. Multicast Routing in Optical Networks

In this chapter, I investigate and evaluate the performance of multicast routing in grooming capable multi-layer optical wavelength division multiplexing (WDM) networks. New wavelength-graph models are proposed for network equipments capable of optical-layer branching of light-paths. I evaluate the cost-effectiveness of multicasting in both electronic and optical-layer. The high scalability of optical multicast against unicast is also shown. All routing and technical constraints are formulated in ILP and realized in my versatile simulator.

In the recent years, the traffic load of transport networks has increased significantly due to the rapid growth of Internet and network based applications. There comes a time when the network providers cannot satisfy the traffic demands by merely enlarging network capacities. Many types of demands, responsible for the heavy traffic load, can be regarded as multicast (point to multipoint) demand instead of ordinary unicast (point to point) demand. Applications include digital media broadcasting (e.g. IP-TV, IP-Radio, etc.) or digital media distribution and streaming. Multicast in a transport network is especially useful when the content has high bandwidth requirements, e.g. distribution of one or several digital TV channels from the distributor of the content to the local providers. Other possible application of multicast is described in [94].

In spite of its benefits in terms of bandwidth savings, today the multicast service is not made available to the end users by most commercial ISPs due to a number of practical reasons. This means that today a huge amount of bandwidth is wasted due to multipoint delivery based on application-layer multicasting (ALM) i.e. unicast-based distribution. In this sense, a recent application that may impel the operators to open the multicast service is TV peer-casting. This application is starting to take an unnecessarily high share of the network capacity as the same streaming information comes in and out of the network for thousands of users by unicast relaying.

Nonetheless, even though not directly available to end users, the multicast service is an essential feature present in the core of the transport network because it is the key to the scalable implementation of the triple-play concept: TV channels are usually multicasted from a content distributor to local caches/relays near the end users.

In general, it can be said that it is less costly to implement multicast in the lowest layers of the network hierarchy; however, when the underlying technology is connection-oriented – as it is the case of optical networks – the number of supported connections becomes a strict bound. In the case of wavelength-routed optical networks, this limit is set by: the number of available wavelengths, the amount of multipoint units in optical nodes, their fan-out and the optical power budget. Given this limitation, optimizing light tree construction is quite a relevant challenge in next generation multicast-capable optical networks.

There are quite a few papers in the field of optimizing the cost of static multicast routing in transport networks (for dynamic approaches see Chapter 5). Madhyastha et al. proposed in [95] a heuristic method for the problem of multicast routing and wavelength assignment in WDM ring networks. They proposed new node architectures with electronic or optical Drop-and-Continue (D&C) feature. The authors of [96] presented an analytical model of grooming problem represented as non-linear programming formulation. They compare the results with heuristic approaches. Several heuristic tree formation algorithms are proposed in [97]. The study assumes a network with sparse splitting capabilities, i.e. only some of the nodes are able to perform splitting of lightpaths in the network. The authors of [98] use an ILP formulation to solve the optimal routing and wavelength assignment problem, and show that a network with only a few splitters and wavelength converters can efficiently transfer multicast demands. Two heuristic optimization algorithms are proposed in [99] minimizing the number of allocated wavelengths in the network. The authors aim to perform optimal QoS (Quality of Service) routing and optimal wavelength assignment together. Mustafa et al. [104] also presented an ILP formulation and heuristic solutions assuming grooming for minimizing the number of electronic-layer equipments and the number of wavelengths.

4.1. Problem Formulation

A two-layer network is assumed, where the upper, electronic layer is time switching capable, while the lower, optical layer is a wavelength (space) switching capable one. The electronic layer can perform traffic grooming, i.e. multiplexing low bandwidth demands into a single WL channel. The two layers are assumed to be interconnected according to the *peer model* [112] or vertically integrated according to the multi-region network node framework, i.e., while routing, the control plane has information on both layers and both layers take part in accommodating a demand. Note, that the result is applicable to overlay or augmented interconnection models as well.

The network topology and the number of fibers are assumed given as well as the description of traffic demands. The capacity of WL channels and the cost of routing, (e.g. space switching, optical to electronic conversion, WL branching, etc.) can also be given in advance. We assume static traffic consisting of unicast and multicast traffic demands. A unicast demand has one source, one destination node and a given bandwidth, while a multicast demand has more than one target node. The objective is to reach all destination nodes from the source, while observing all routing and technical constraints.

4.2. Node Models

A physical device is modeled by a sub-graph. It represents all IFs of the device and the capabilities of its internal switching fabric. The WL graph model (together with my ILP framework) can support devices with different capabilities appearing in the network at the same time. The model is easily extendable; type of devices can be changed later, if new internal models are introduced.

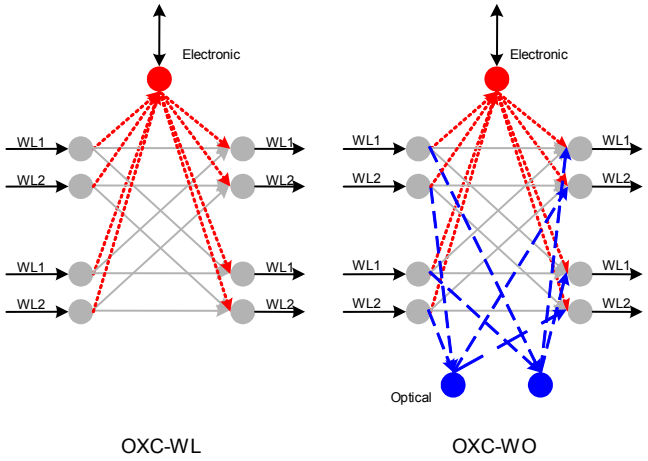

Fig. 27. Sub-graph of an OXC-WL device in the wavelength graph (left). Sub-graph of an OXC-WO device in the wavelength graph (optical splitting capable) (right).

A sub-graph of a versatile physical device is depicted in Fig. 27. The equipment is a combination of an OXC with WL-conversion and an OADM: it can originate and terminate traffic demands, as well as perform space-switching. WL-conversion is possible only through the electronic layer. This is illustrated by an electronic node in the sub-graph, while other (pair of) logical nodes correspond to interfaces. Fig. 27 assumes two fibers connected to the device, and two WLs per fiber, which results in two input and two output interfaces – because all edges are directed. We will use this complex node and its extension in the simulations.

The OXC-WO (OXC with WL-conversion and Optical splitting) devices should be considered as an extension of the OXC-WL type. It introduces a new functionality: optical splitting of light-paths. The branching function is represented by dashed edges in the sub-graph. By adding these specific logical edges into the sub-graph, we can accurately determine the cost of optical branching. The in-degree of splitting nodes is always less than or equal to one, while the out-degree is more than or equal to two.

Branching in this device is, however, possible in the electronic layer as well. To accomplish electronic branching, the demand must be first routed up to the electronic layer. The returned (branched) lightpaths should not necessarily use the same WLs. Thus WL-conversion can also be performed in addition to branching (and 3R processing) of the signal.

4.3. ILP Formulation

We used the following ILP formulation to route multiple multicast trees in the network (Note, optical branching requires a few extra constraints in addition to these):

$z_{ij}^{or} \in \{0,\ 1\}$ indicates whether sub-demand *o* of multicast tree *r* uses edge *(i, j)* or not.

$x_{ij}^{r} \in \{0,\ 1\}$ indicates whether (end-to-end or multicast demand) *r* uses edge *(i, j)*.

$y_{ij} \in \{0,\ 1\}$ indicates if edge *(i, j)* is used by any of the demands.

$$\sum_{\forall j \in V_i^+} z_{ji}^{or} - \sum_{\forall k \in V_i^-} z_{ik}^{or} = \begin{cases} -1 & \text{if } i = s^r \\ 0 & \text{if } i \notin \{s^r,\ t^{or}\} \\ +1 & \text{if } i = t^{or} \end{cases} \qquad (0.47)$$

for every (logical) node $i \in V$, tree *r* and *o* sub-demand.

V_i^+ denotes the set of nodes that can reach node *i* by the use of one directed edge, while V_i^- is the set of nodes reachable from *i* by one directed edge. Capitals *A*, *V*, V_E, *O*, *R* denote the set of edges (arcs), nodes (vertices), electronic nodes, sub-demands and the set of multicast trees respectively. The source of tree *r* is denoted by s^r, while targets are denoted by t^{or}, where *o* is the corresponding sub-demand.

$$z_{ij}^{or} \le x_{ij}^{r},\ \forall (i,\ j) \in A,\ \forall o \in O,\ \forall r \in R \qquad (0.48)$$

$$x_{ij}^{r} \le \sum_{\forall o \in O} z_{ij}^{or},\ \forall (i,\ j) \in A,\ \forall r \in R \qquad (0.49)$$

$$\sum_{\forall j \in V_i^+} x_{ji}^{r} = \sum_{\forall k \in V_i^-} x_{ik}^{r} \le 1,\ \forall i \notin V_E,\ \forall r \in R \qquad (0.50)$$

$$\sum_{\forall j \in V_i^+} x_{ji}^{r} \le \begin{cases} 0 & \text{if } i = s^r \\ 1 & \text{if } i \ne s^r \end{cases},\ \forall i \in V_E,\ \forall r \in R \qquad (0.51)$$

$$\sum_{\forall r \in R} x_{ij}^{r} \cdot b^{r} \le B_{ij},\ \forall (i,\ j) \in A \qquad (0.52)$$

$$x_{ij}^{r} \le y_{ij},\ \forall (i,\ j) \in A,\ \forall r \in R \qquad (0.53)$$

$$y_{ij} \le \sum_{\forall r \in R} x_{ij}^{r},\ \forall (i,\ j) \in A \qquad (0.54)$$

$$\sum_{\forall j \in V_i^+} y_{ji} = \sum_{\forall k \in V_i^-} y_{ik} \le 1,\ \forall i \notin V_E \qquad (0.55)$$

Variables:

$$z_{ij}^{ro} \in \{0,\ 1\},\ \forall (i,\ j) \in A,\ \forall o \in O,\ \forall r \in R \qquad (0.56)$$

$$x_{ij}^{r} \in \{0,\ 1\},\ \forall (i,\ j) \in A,\ \forall r \in R \qquad (0.57)$$

$$y_{ij} \in \{0,\ 1\},\ \forall (i,\ j) \in A \qquad (0.58)$$

Objective function:

$$\text{Minimize} \sum_{\forall (i,\ j) \in A} c_{ij} y_{ij} \qquad (0.59)$$

(0.47) expresses flow-conservation of each sub-demand of each tree in all nodes. (0.48) tells that edge *(i, j)* must be allocated if any sub-demand of tree *r* wants to use it, while (0.49) ensures that edge *(i, j)* is not allocated in vain. (0.50) ensures that light-paths cannot branch (or disappear) in non-electronic nodes. Constraint (0.52) says that the aggregate bandwidth of demands using edge *(i, j)* should be less than the capacity of one WL. (0.53) expresses that a tree can only use an edge, if it is allocated for the routing. In addition (0.54) ensures that edges used by none of the trees are not allocated. (0.54) can be omitted, however, since it is implicitly included in the minimization objective. Constraint (0.55) is similar to (0.50), but on the highest hierarchy-level. It expresses that the number of allocated input and output edges should be equal in non-electronic nodes. The objective function (0.59) expresses that the total cost of allocated edges should be minimized, i.e. we are looking for a minimal-cost routing solution in the network.

The above mentioned formulation can be applied for a mix of unicast and multicast demands as well. The routing problem of solely unicast demands has equivalent ILP formulation using only a two-level hierarchy of variables. Although that formulation contains less variables and inequalities, the solution time does not differ significantly from the solution time of the general formulation. The ILP solver likely recognizes the simplifications and shortcuts, thus it does not cause significant (more than an order of magnitude) deviation in the solution times.

4.3.1. Technical Constraints

We applied several technical constraints to influence certain properties of the routing. These constraints reflect real-world technical restrictions. Two types of constraints are considered: branching limitations and restrictions concerning size of multicast trees.

Branching Limitation

Most of the current switching devices are not able to perform neither optical nor electronic layer branching of the signal due to technical or software restrictions. Thus it is necessary to limit the splitting of the signal in both layers. The electronic layer branching can be constrained by the following simple inequality:

$$\sum_{\forall j \in V_i^-} x_{ij}^{r} \le \alpha_i,\ \forall i \in V_E,\ \forall r \in R \qquad (0.60)$$

The branching limit in node *i* is denoted by constant α_i.

The all-optical branching of the signal has further implications. The power of the signal decreases by 3 dB when it is split in two. In case of splitting in more than two the attenuation is more significant, which impairs correct detection in a receiver.

The number of times the signal is split can be constrained for each sub-demand, because it determines the quality of the signal in the receiver. The number of optical splits can be constrained by the following formula:

$$\sum_{\forall i \in V_{osplit}} z_{ij}^{or} \le L^{or}, \ \forall o^r \in O, \forall r \in R, \qquad (0.61)$$

where L^{or} means the upper limit for sub-demand *o* of tree *r*.

Constraint (0.61) sets an upper limit for the number of splits along the path of each sub-demand of each tree. The number of optical splits along the path can be calculated by counting the number of edges originated from optical splitting nodes. Note, that we only calculate the total number of splits along the path, not the maximum number of consecutive splits along the path. Although that property is more important, it can be formulated only by too complex linear constraints.

Limitation on the Size of the Multicast Tree

Sometimes the optimal solution of a multicast tree produces too long paths between the source and some of the targets. It also implies higher delay, which is unacceptable in some applications or harms the QoS (Quality of Service) agreement. The length (measured in hops in this case) of the path can be limited by the following formula, referred to as *depth-limit constraint*:

$$\sum_{\forall (i,j) \in A} z_{ij}^{or} \le \beta^{or}, \ \forall o \in O, \forall r \in R \qquad (0.62)$$

Different requirements can be specified for each tree *r* and sub-demand *o* by assigning different β^{or} constant values. However, inadequate selection of β^{or} values can render the whole routing problem infeasible. In order to measure the length of the path in some metric such weight factors should be introduced that represent the distance.

It is also possible that size of the whole tree should be limited. The number of links contained by the tree can be restricted by the following formula (*tree-size limit*):

$$\sum_{\forall (i,j) \in A} x_{ij}^{r} \le \mu^{r}, \ \forall r \in R, \qquad (0.63)$$

μ_r is a constant value meaning the tree-size limit.

A switching device is called a leaf-node, if it does not branch or relay the multicast tree, but terminates it. It implies that the out-degree of such a node equals to zero. The width of the tree is defined by the number of leaf nodes in it. The *breadth-limit* of the tree can be set up by the following inequalities:

$$\sum_{\forall j \in V_i^-} x_{ij}^{r} \le \kappa_i \cdot (1 - v_i^r), \ \forall i \in t^r, \forall r \in R \qquad (0.64)$$

$$(1 - v_i^r) \le \sum_{\forall j \in V_i^-} x_{ij}^{r}, \forall i \in t^r, \forall r \in R \qquad (0.65)$$

$$\sum_{\forall i \in t^r} v_i^r \le \eta^r, \forall r \in R \qquad (0.66)$$

Variables:

$$v_i^r \in \{0, 1\}, \forall i \in V_{dr}, \forall r \in R, \qquad (0.67)$$

Variable $v_i^r \in \{0, 1\}$ expresses whether tree *r* is relayed (0) or terminated (1) in node *i*, while κ_i is a properly chosen constant of a large value. Constant η^r expresses the breadth-limit. The sum in (0.66) should only include the electronic nodes of the target switching devices, since these are the only candidate leaf nodes of the tree. Because of the other routing constraints other leaf nodes (other than the target nodes) are not possible.

4.3.2. Soft Constraints

All technical constraints and several of the routing constraints can be reformulated as soft-constraints. It means that a technical or routing restriction/preference is not involved anymore in the ILP constraints; rather it is included in the optimization objective. If a solution does not observe a recommendation, it is not rejected, but penalized. The necessity of a recommendation can be controlled by adjusting the value of the penalty. The advantage of this method is that we can sooner obtain a feasible solution, instead of waiting for a solution observing all the limitations. This technique is especially useful when the whole problem together is infeasible, but lessening the requirements can lead to a feasible solution. For example the soft depth-limit constraint can be expressed by the following formula:

$$\sum_{\forall (i,j) \in A} z_{ij}^{or} \le \beta^{or} + q^{or}, \ \forall o \in O, \forall r \in R \qquad (0.68)$$

$$q_i^{ro} \in R^+, \forall o \in O, \forall r \in R \qquad (0.69)$$

$$\text{Minimize} \sum_{\forall (i,j)\in A} c_{ij} y_{ij} + \sum_{\substack{\forall o\in O,\\ \forall r\in R}} c^{or} q^{or} \qquad (0.70)$$

q^{or} is a positive variable representing the deviation from the recommended value of the limit, while c^{or} is a constant meaning the weight of the penalty. Other constraints can be reformulated as soft constraints in the same way.

4.4. Simulations Results

In all simulations NRS core network topology [101] was used with 5 WLs per link.

The measures "number of destination nodes", "number of light-trees" and "solution cost" form a three dimensional space, which is hard to visualize. Therefore the two most important sections of this space is illustrated in Fig. 28. The *left subfigure* displays the cost of routing as a function of the increasing number of targets assuming different number of light-trees (in the legend). The *right subfigure*, on the other hand, shows the cost of routing as a function of the increasing number of light-trees (sources) assuming different number of target nodes. Fig. 28 shows that multicast routing scales well with both the increasing number of light-trees and target nodes. The cost curve does not rise beyond a limit, it has a saturation section. It means inserting a new light-tree or further nodes into the current trees has gradually decreasing cost.

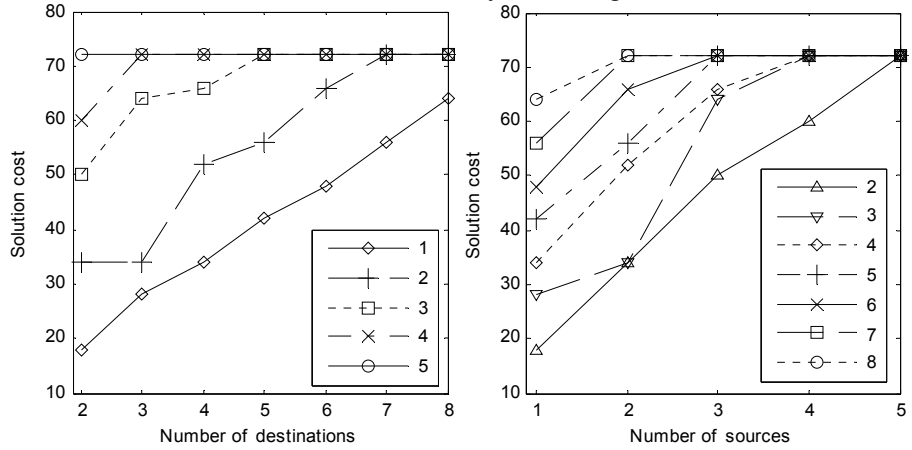

Fig. 28. Cost of routing as a function of the increasing number of targets for different number of sources (left). Cost of routing as a function of the increasing number of sources for different number of targets (right).

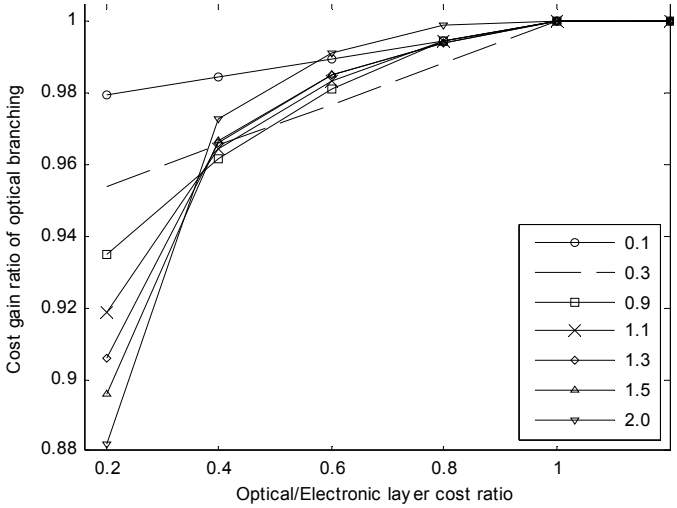

Fig. 29. Cost gain ratio of optical branching versus electronic layer only branching as a function of optical-electronic cost ratio. Different curves assume different WL costs.

Fig. 29 tells that significant cost can be saved if optical layer branching capability is introduced into the network. The figure depicts the cost of routing in an optical branching capable network compared to an only

electronic branching capable network. It shows that the lower the price of the optical layer, the more costs can be saved. This suggests that optical layer branching is particularly worth, if the electronic layer has high cost, which is in accordance with the real-world conditions. The cost gain depends, however, on several properties of the network and the set of demands.

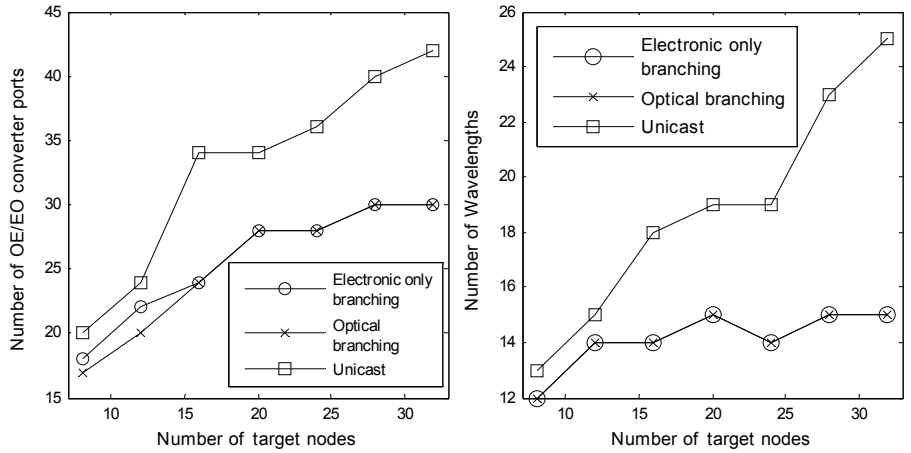

Fig. 30. Required number of converter ports (left) and wavelengths (right) as a function of the number of target nodes for unicast and multicast routing (with and without optical branching).

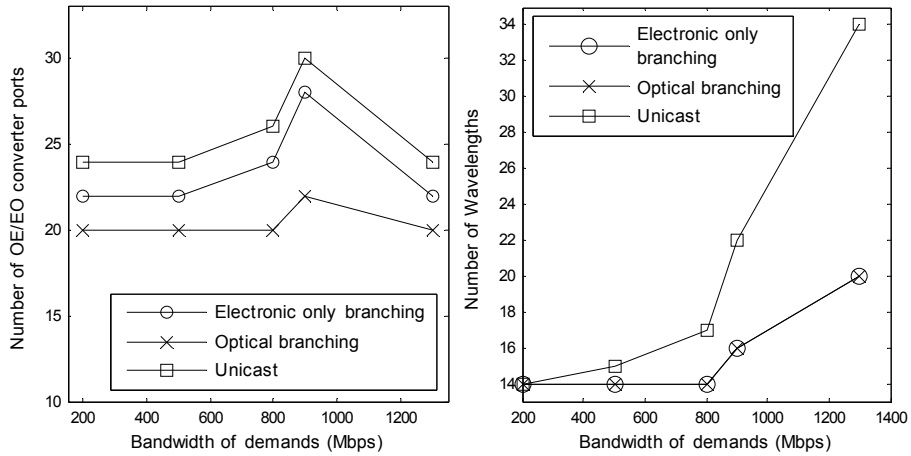

Fig. 31. Required number of converter ports (left) and wavelengths (right) as a function of the bandwidth of demands for unicast and multicast routing (with and without optical branching).

Fig. 30 shows that the required number of converter ports and wavelengths rises with the increasing number of target nodes. Optical branching outperforms electronic branching only if the number of nodes participating in the trees is low – compared to the total number of nodes in the network. As more and more nodes are involved in the trees, the advantage of optical branching disappears. It is likely due to the fact that in member nodes of the trees the electronic layer must be reached anyway, thus there is less necessity for optical layer branching in these nodes. Both the required number of converter ports and WLs scale well with the increasing number of demands, unlike in the case of unicast routing, where the required number of resources increases more rapidly. In the case of unicast, a set of end-to-end demands equivalent to the multicast demands were routed.

Fig. 31 also shows the required number of converter ports and WLs, but as a function of the increasing bandwidth of the demands. Four trees were routed with three target nodes each. In such a situation, optical branching clearly uses less converter ports than electronic branching. However, optical branching does not decrease the required number of WLs. Unicast – naturally – performs far worse in all measurements. Grooming is less and less applicable if the bandwidth of the demands is close to the WL capacity. The sudden fall of the number of OE/EO ports can be explained by this fact. In case of tiny bandwidths multicast routing does not really show its power, since grooming compensates some of the disadvantages: a huge amount of WLs can be spared by multiplexing tiny demands into a single WL channel.

4.5. Conclusion

In the chapter, I proposed WL graph models and a new ILP formulation to route unicast and multicast demands in WDM networks. I evaluated the cost and the resource usage of multicast routing with and without optical layer branching of light-trees and unicast routing as well. I showed that optical branching usually performs better than branching in the electronic layer only. However, if many nodes are involved in the trees then electronic layer is used anyway, and optical branching loses its gain. In case of tiny bandwidths, grooming can well compensate the drawback of unicast over multicast.

5. Reconfiguration of Multicast Trees

In this chapter I study *dynamically changing multicast trees* in two-layer optical networks. In this scenario, the continuous changing of multicast endpoints causes the *degradation of the tree*. Therefore a huge amount of network *resources can be spared by regular reconfiguration*. The benefit of reconfiguration is investigated for different routing algorithms and reconfiguration periods.

In dynamic multicast trees the member tree leaves are continually changing. New destination nodes may log in to the tree to receive the content, while other nodes may leave the tree and return at a later time. This corresponds to a scenario where IP membership drives optical tree set up. In a real setting, the tree would be due to the aggregation of multiple multicast sessions or it could be given by a selected set of individual ultra broadband multicast sessions.

An example can be a digital media distribution service, where the audience is varying in time. New customers appear, who subscribe to the content, and other customers with expired subscription leave the network. In this case a customer does not necessarily mean an individual home user, but also a local provider (e.g. a local cable-TV provider).

Another example can be a virtual LAN service, where LAN broadcast has to be delivered to all endpoints. In contrast with the previous scenario, this application is less sensitive to minor interruptions in transmission caused by reconfiguration of the multicast tree.

The continuous changing of tree leaves causes the degradation of the multicast tree in terms of switching and transmission resources as the tree diverges from the optimum. This degradation can be overcome by regular reconfiguration of the tree in order to take advantage of resources saved up.

However, there are also some drawbacks of reconfiguration: it may require a lot of computation time as determining the Steiner tree is an NP-complete problem. However, considerable saving can be obtained by using faster heuristic methods trading-off speed and optimality. Reconfiguration can cause a short disruption in the data transmission flow or cause packet reordering, which is sometimes not acceptable by the application and should be avoided with some technique. Reconfiguration implies an additional signaling overhead as well.

Quite a few papers were published in the field of optimizing the cost of multicast routing (light-trees) in optical networks. Since the problem of routing demands optimally is often infeasible or time-consuming, several heuristic approaches were proposed and their performance was compared with ILP (Integer Linear Programming [102])-based optimal solutions.

The problem of static multicast optical wavelength routing was deeply investigated for ring and mesh networks (Chapter 4). Recently the optimization of *dynamically changing multicast trees* attracted more attention. Several provisioning methods of dynamic trees (assuming grooming) are discussed in [105][106][107]. In the dynamic case, the goal is usually to minimize the blocking ratio, not to route all demands (according to some constraints) as in the static case. This problem in general is even more resource- and computation-intensive then the static version. I found, however, that some sub-problems of routing (e.g. optimization of a single tree, or several trees separately) can be solved optimally by ILP. Therefore, it is worth to compare the performance of dynamic routing algorithms to the optimal solution, and to calculate the benefit.

In [108] traffic engineering is performed through dynamic traffic grooming in grooming-capable WDM networks in the unicast scenario.

The authors of [109] proposed a dynamic wavelength assignment algorithm for multicast to minimize call blocking probability by maximizing the remaining network capacity in each step. Chowdhary et al. [110] addressed a similar problem by provisioning on-line multicast demands with the objective of increasing the resource utilization and minimizing the blocking probability for future arriving requests.

Boworntummarat et al. introduced light-tree-based protection schemes against single link failure in [111]. ILP formulations were developed to measure and compare the minimum spare capacity requirement of the proposed protection strategies.

According to my knowledge, no previous work analyzed the effect of regular reconfiguration of light-trees, investigating the degradation of dynamic routing algorithms, and comparing the dynamically changing costs to the optimum.

5.1. Smooth Transition to Tree Reconfiguration

Although here we do not intend to solve this problem, we suggest some techniques to show that it is feasible.

A solution for interruption-sensitive application (e.g. media streaming) is a soft switch-over from one light tree to the new one. In this case the new light-tree is set up, before the old one is torn down. There is a short period when both trees exist and are able to transmit data at the same time. In order to prevent loss of sequence during the change of tree, the transmission can be held for a short time at the ingress to guarantee that all the packets are flushed out of the original tree. Alternatively, the first packets that travel through the new tree are buffered at the egress node until an end-of-transmission signaling packet arrives through the old tree.

5.2. Problem Formulation

The main difference compared to the problem formulation (Section 4.1) of Chapter 4 is that here I assume *traffic consisting of dynamic multicast delivery demands*. As explained before, these demands may correspond to an individual ultra high-speed IP multicast session or to a set of aggregated sessions that share most of the leaves. The heuristics for multiple session aggregation into a single light tree falls out of the scope of this work. The same consideration is made regarding joint optimization of light trees and light tree merging: for the sake of simplicity, in this study light-trees are optimized separately, although, joint optimization could yield a higher gain at a higher computational cost.

A multicast tree consists of multiple so-called sub-demands, which can share resources in the network (e.g. their bandwidths are not additional). One sub-demand is assigned to each destination node of the tree. The source of every sub-demand is the single source node of the multicast tree. Destination nodes of the tree change dynamically: new nodes can log in the tree or existing nodes can log out at any time. Paths of new sub-demands have to be calculated online while paths of leaving nodes need to be torn down as carefully as possible by not affecting other sub-demands.

The active session time (holding time) and the idle (inter-arrival) time for every destination node are assumed to have an exponential distribution. The traffic load can be determined by appropriately setting the rate parameters (λ) of the distributions.

The objective is to reach all current destination nodes from the source in every time step.

5.3. Network Model

The sub-graph of a versatile physical device used in dynamic multicast simulation is the same as the one depicted in Fig. 27 (left). The equipment is a combination of an OXC with WL-conversion and an OADM: it can originate and terminate traffic demands, as well as perform space-switching. WL-conversion and splitting (branching) of light-trees can only be performed in the electronic layer.

5.4. Routing Algorithms

Several algorithms were applied to route the demands in the network in order to compare their costs and performances. A simple example illustrating the different outcome of the algorithms is shown in Fig. 32.

5.4.1. ILP Routing and Formulation

ILP always provides the optimal cost of routing the current demands in the system, thus it serves as a baseline for comparison. However, this does not necessarily mean that the numbers of certain resources (e.g. wavelengths, O/E, E/O conversion ports) are all minimal as well.

On the other hand, ILP routing usually consumes much time. The routing time of one multicast tree is still acceptable even for larger networks. This time varied from 3 seconds to 180 seconds on a 2.8 GHz Pentium for COST266 network [92], which consists of 28 nodes and 41 links. If we want to route several trees together by introducing grooming much more cost can be spared, but the solution becomes unacceptably time consuming. Therefore it is only possible to route different trees separately one after the other.

An important disadvantage of ILP routing is that the consecutive configurations are very dissimilar, thus reconfiguration of the paths of demands (including switching devices along the path) is unavoidable.

I used the ILP formulation introduced in Section 4.3 to route multiple multicast trees in the network. This formulation is able to route unicast and multicast demands as well, or even demands from both types at the same time.

5.4.2. Accumulative Shortest Path (Dijsktra's Algorithm)

Accumulative shortest path algorithm is fast and simple. It can be applied for routing a new demand by not interrupting the current active sub-demands in the network. On the other hand, the resultant routing topology of this algorithm is very far from optimal (and thus costly).

The accumulative shortest path algorithm works as follows: routes are calculated between the source and the destination nodes one after the other. The algorithm operates directly on the logical network (wavelength graph). The source and the destination nodes of a sub-demand are the electronic nodes of the corresponding physical device. The cost of already reserved edges of the graph is set to zero, which means it can be used for free.

Paths to leaving destination nodes are cleared. Edges that are not used by the multicast tree anymore are de-allocated. Dijsktra's algorithm never modifies paths of existing sub-demands, which unfortunately often results in longer paths.

5.4.3. Minimal Path Heuristic (MPH)

The MPH algorithm transforms the original wavelength graph into a virtual graph and applies Prim's algorithm [115] to form a minimum cost spanning tree. A virtual graph is a full mesh, in which only the single source and all the destination electronic nodes are presented. The weight of an edge in the virtual graph expresses the cost of the shortest path in the original wavelength graph (which implies that the shortest paths have to be calculated for every node pair in both forward and backward directions). Prim's algorithm is applied in this "upper-layer" virtual graph. After the minimal cost spanning tree is found, the paths are traced back into the original wavelength graph. The cost of already used edges of the virtual graph equals to zero when the spanning tree is being updated after a new destination node logged in. This ensures that paths of existing sub-demands are not modified. Details of MPH algorithm can be read in [116].

5.4.4. Tree Routing

This algorithm is similar to the MPH algorithm, except that it operates in the wavelength-graph, not in a derived "upper layer", virtual graph. It applies the same Prim's algorithm to determine a minimal cost spanning tree in the WL graph. Updating the tree and modification of the edges costs are also similar to the former case.

A phenomenon can occur in the cases of both tree routing and MPH that needs attention: trees can branch in nodes, where splitting is not allowed (i.e. in non-electronic nodes). These forbidden branches need to be corrected by a post-processing. In fact, it is pretty simple to solve the problem by moving both branched paths up to the electronic layer.

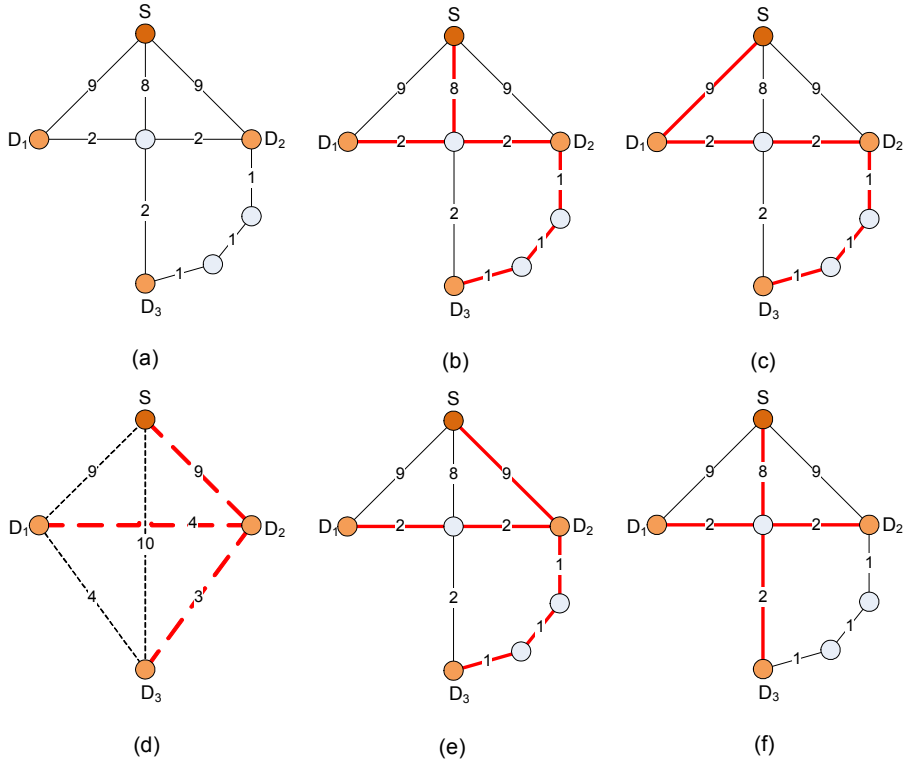

Fig. 32. Original topology with the source node and three leave nodes (a), tree routing (b), accumulative shortest path routing (c), MPH virtual topology and routing (d), MPH routing (e), ILP optimal routing (f)

5.5. Simulation Results

The simulations were carried out using the COST 266 European reference network [92] (unless otherwise indicated) with the same traffic pattern for all algorithms.

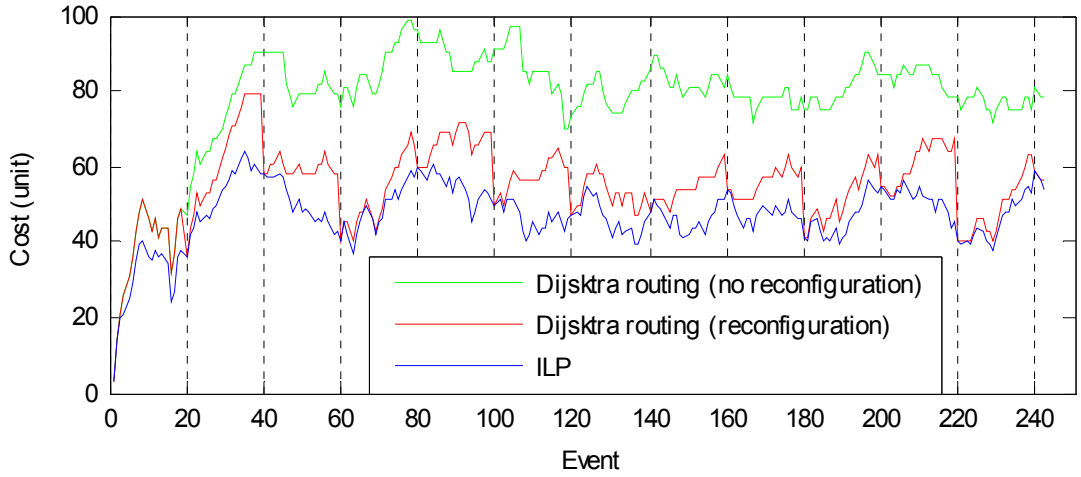

Fig. 33. The cost of routing as a function of elapsed events for Dijsktra's algorithm with (middle curve) and without (upper curve) reconfiguration compared with optimal ILP solution (lower curve)

In Fig. 33, the cost of routing is plotted as a function of elapsed events. Every change of the light-tree (i.e., a destination node enters or exits the tree) is considered as an "event". In Fig. 33, the (lower) blue curve (marked as ILP) represents the optimal cost in every step, while the (top) green one stands for the case where no reconfiguration was applied. The (middle) red curve shows the effect of the regular reconfiguration in every 20^{th} event.

In my experiment Dijsktra without reconfiguration exceeds the optimal solution by more than 60 percent on average. The reconfiguration curve usually diverges rapidly from the optimal curve. It has the same cost, though, as the optimal one in every 20^{th} event because of the reconfiguration. Although reconfiguration is clearly beneficial (according to Fig. 33), it surely depends on the network topology, the applied dynamic routing algorithm and the reconfiguration period as well.

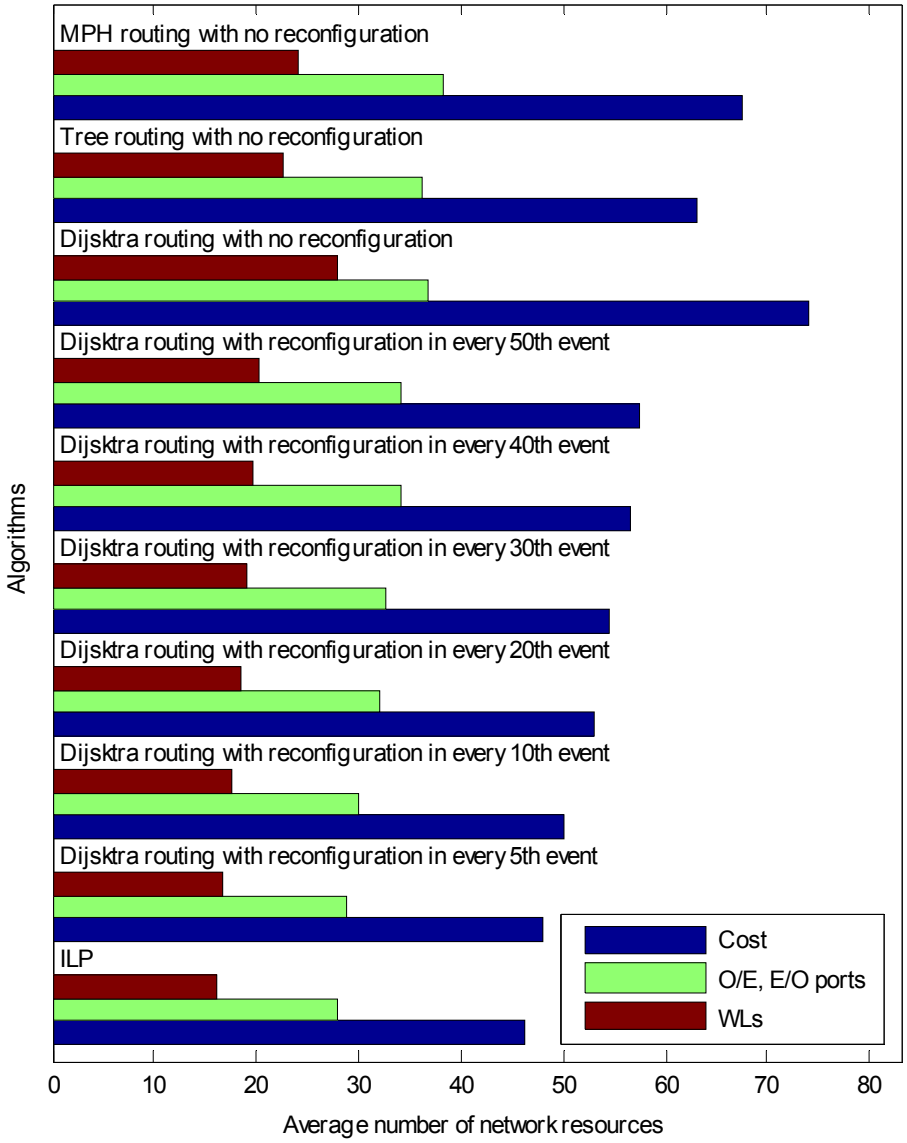

Fig. 34. The average routing cost, conversion ports (O/E, E/O) usage and WL usage of different algorithms and (Dijsktra's) shortest path algorithm with different reconfiguration periods

Therefore I also investigated the cost (namely the sum cost of used edges of the WL graph) and network resource usage of different routing algorithms (described in Section 4) and accumulative shortest path routing (Dijsktra) with different reconfiguration periods. The results are depicted in Fig. 34. It is clear, that all of the algorithms (without reconfiguration) are far from optimal: in the current simulation the additional cost is around 34 to 57 percent compared to the optimum. Much cost can be spared by regular reconfiguration. As expected, the shorter the period of reconfiguration, the closer the average cost approaches the optimal value. However, we should know, that reconfiguration can be computation-intensive and has other disadvantages (see Section 5.1). These drawbacks are not yet taken into account in the cost.

The results are very similar for network resources necessary to realize the routing: i.e. the number of required O/E and E/O conversion units and the number of wavelengths (Fig. 34). One interesting fact is that Dijsktra's algorithm without reconfiguration has an outstanding WL usage, while the usage of opto-electronic converters is behind MPH routing. Both WL and conversion port usage approach the optimal value by decreasing the length of reconfiguration period.

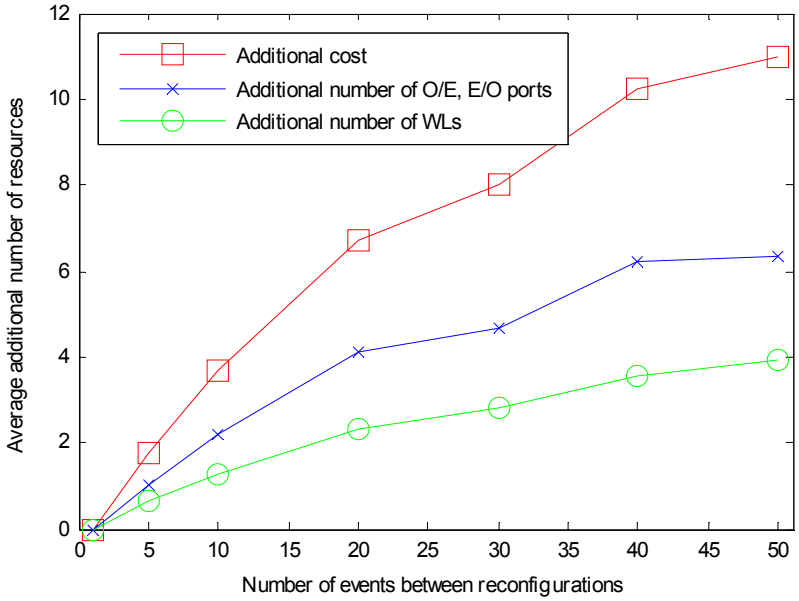

Fig. 35. The average additional cost of routing (upper curve), number of O/E, E/O conversion ports (middle curve) and number of WLs (lower curve) as a function of the length of reconfiguration period

Another interesting issue is determining how the length of the reconfiguration period affects the cost gain. The average additional cost of routing as a function of the length of reconfiguration period is depicted in Fig. 35. The figure shows a saturating curve with decreasing slope. This means, that in order to reach high cost gain, frequent reconfiguration is necessary. There is not much difference between cost gains, when the periods are long. The required number of WLs and conversion ports follows the same rule: both have a decreasing slope.

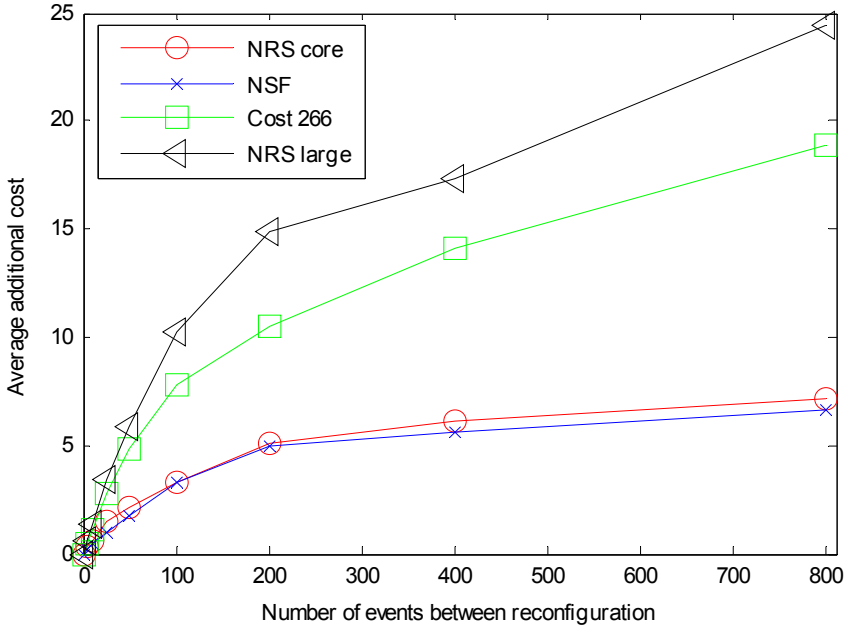

Fig. 36. The average additional cost for different network topologies

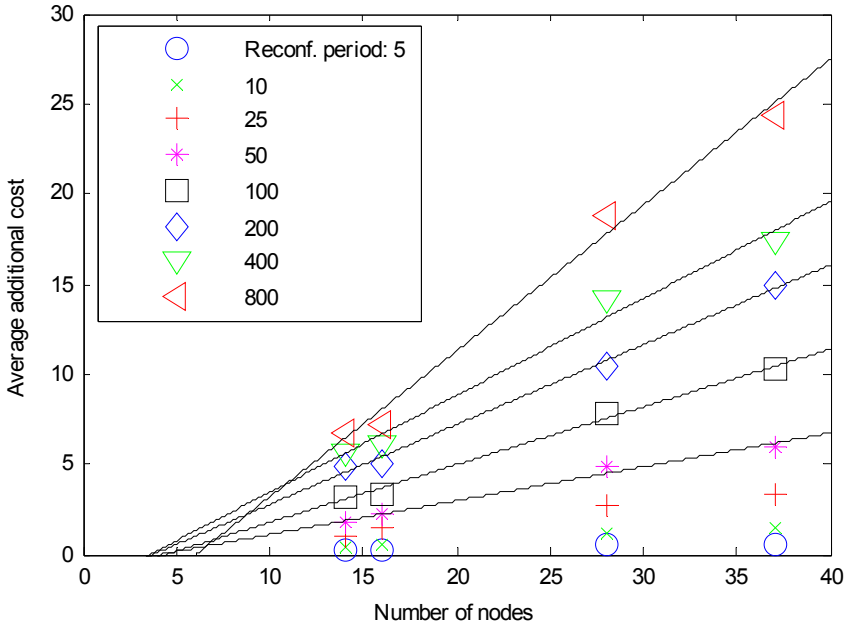

Fig. 37. The average additional cost for different network topologies

I repeated the same measurement for several reference networks to study how the additional cost curve (as a function of the reconfiguration period) looks like in case of different topologies. The same amount of traffic was injected in all of the networks. I obtained similar saturating curves again for all topologies (see Fig. 36). However, the slopes of the curves differ. For larger networks, the additional cost rises more rapidly as the length of the reconfiguration period is increased.

Table II. Reference networks used in the simulations

Topology	Number of nodes	Number of links
NRS core [101]	16	23
NSF net [118]	14	21
COST 266 [92]	28	41
NRS large [101]	37	57

Therefore I depicted the additional cost as a function of the number of nodes in the network (Fig. 37). The symbols mean different lengths of reconfiguration periods; the linear regression was also computed for most of the data series to show the clear linear trend. I found similar relationship between the average additional cost and the number of links in the network. However, the trend is not obviously linear in that case.

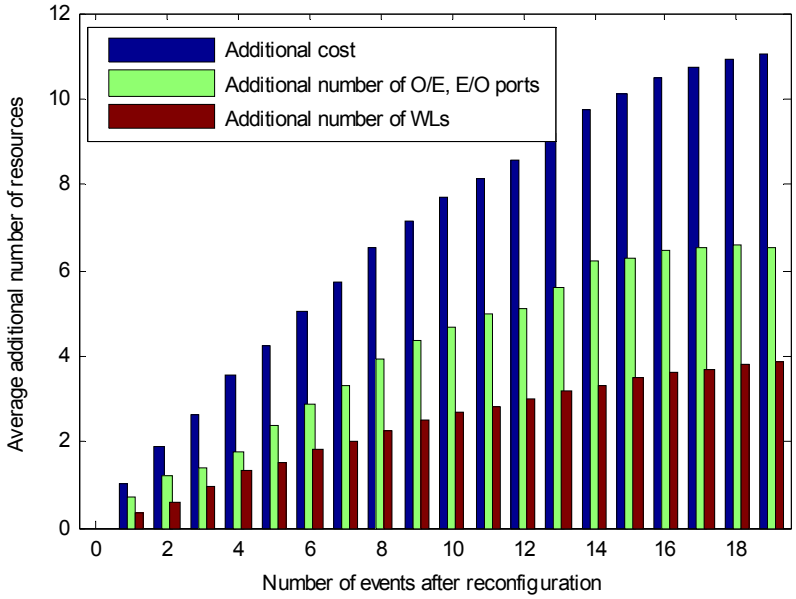

Fig. 38. The average additional cost of routing (higher bar), number of O/E, E/O conversion ports (middle bar) and number of WLs (lower bar) after reconfiguration as a function of elapsed events

Fig. 38 shows how fast the cost of the optimized reconfigured light-tree diverges from the optimal curve. This is also a saturating curve with decreasing slope, similar to the left one. This suggests that in the first few steps the cost of the tree is quickly diverging from the optimal curve, then during the next few events this divergence is slowing down. The same kind of divergence is true in terms of conversion ports and WLs as well: after reconfiguration the multicast tree hastily uses more network resources compared to the optimal topology.

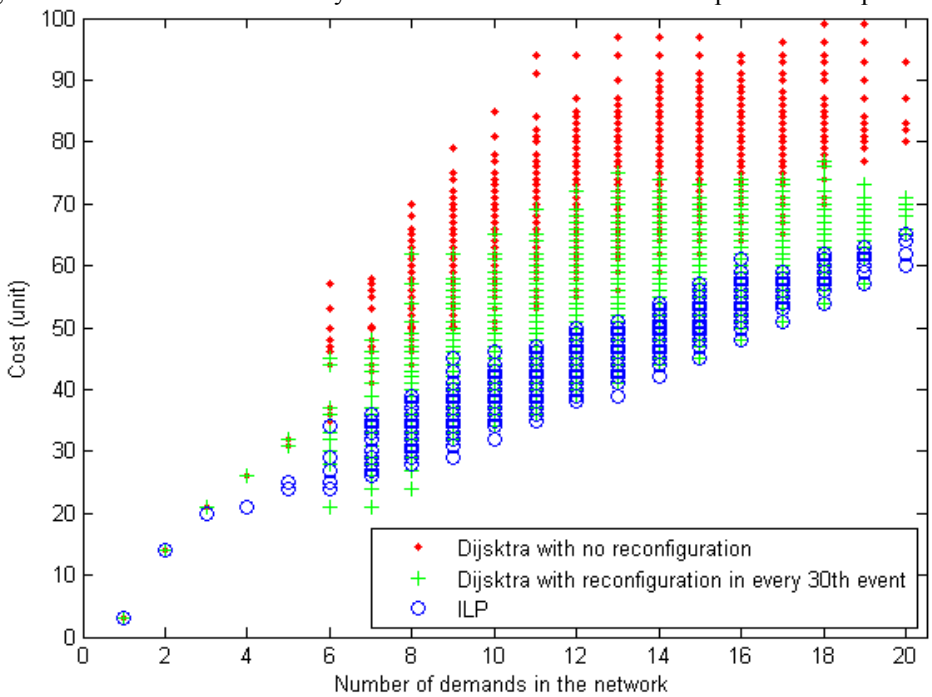

Fig. 39. The cost of routing as a function of the number of destination nodes of the light-tree

The next figure (Fig. 39) displays the cost of routing as a function of the number of destination nodes of the light-tree. Each data point corresponds to one time-step in the simulation. The figure compares shortest path routing with and without reconfiguration to the optimal solution. As expected, the routing cost naturally raises as the number of the destinations increases. The signs show the typical ranges of the dynamically changing cost for the routing methods. It is noticeable that the range of shortest path with reconfiguration is somewhere between the optimum- and the no-reconfiguration range.

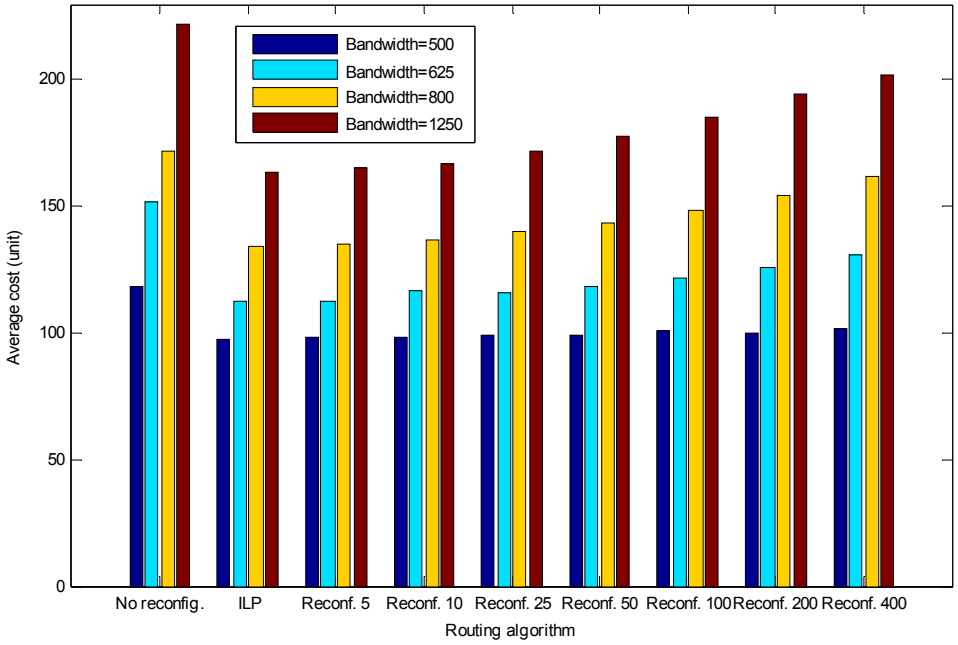

Fig. 40. Average routing cost for different bandwidth of demands

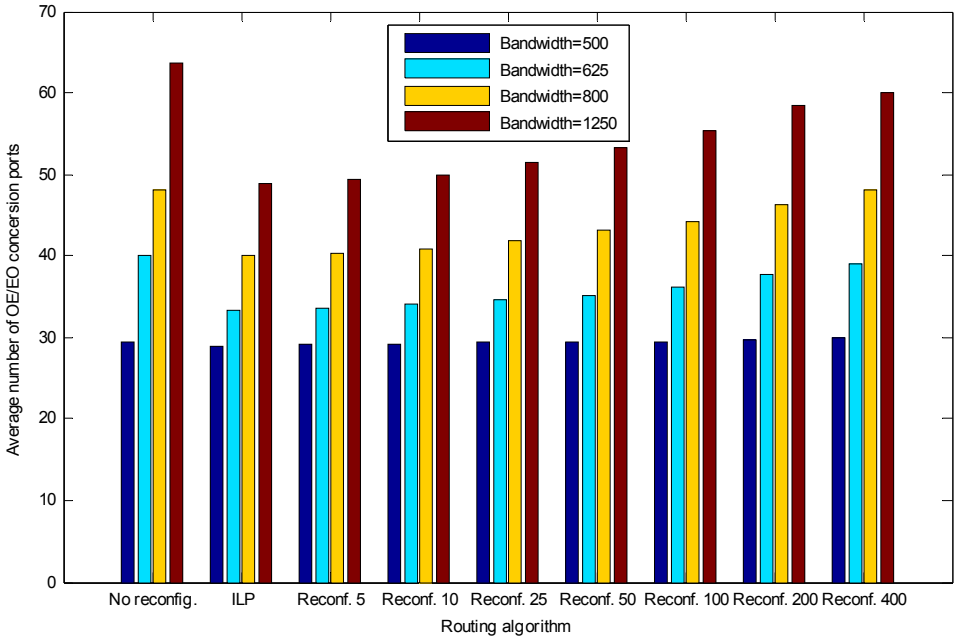

Fig. 41. Average number of converter ports (O/E, E/O) for different bandwidth of demands

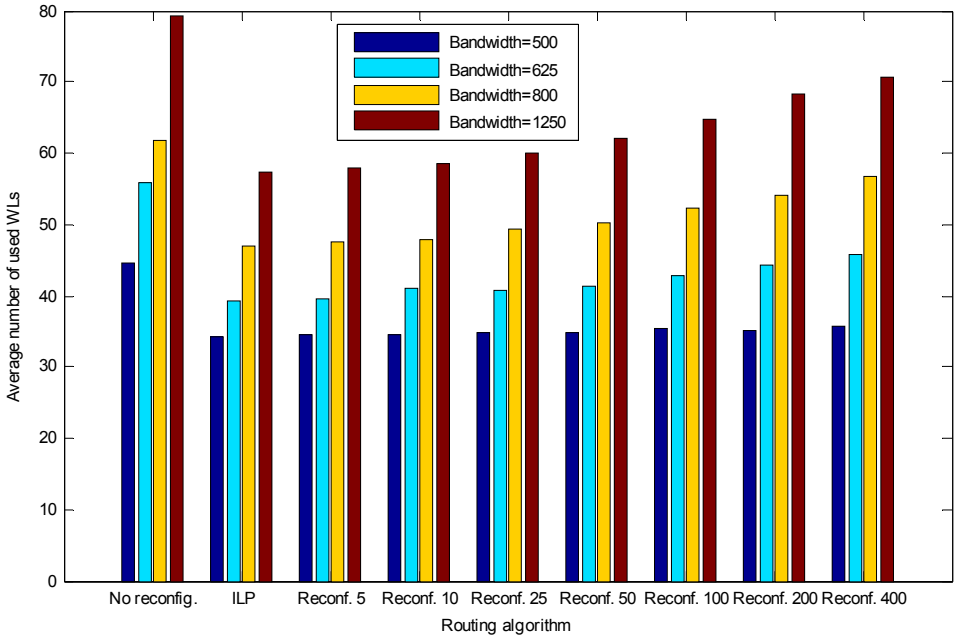

Fig. 42. Average number of used WLs for different bandwidth of demands

In the next experiment, I consider multiple trees (5) at the same time with specific bandwidths. The bandwidths are set so that grooming should be applicable (i.e. the bandwidth of demands is lower than the half of the wavelength capacity to enable at least two demands to be groomed). In this case, all trees were optimized separately by ILP in a certain order (in decreasing order of tree size), which does not provide the global optimum. However, with this technique we can route so many trees without facing a complexity-problem, which is useful if numerous light-trees are assumed with tiny bandwidths. The routing costs of shortest path routing and ILP are compared. Fig. 40 suggests that reconfiguration is more beneficial in case of higher bandwidths, since grooming is less useful here. In case of low bandwidths grooming can make more network resources available, which allows less frequent reconfiguration. Results for conversion units and WLs (Fig. 41 and Fig. 42) show the same behavior as total cost (Fig. 40).

Finally, I consider not only the network cost of routing, but the cost of reconfiguration as well (Fig. 43). The more frequent the network is configured, the higher cost (including computational power and signaling overhead) we have to pay. The (relative) cost of reconfiguration is assumed to be an exponentially decreasing function of the length of reconfiguration period. On the other hand, the additional network cost proved to be a raising function of the length of reconfiguration period. If we add these two functions together, that necessarily results a minimum point in the total cost (= network cost + cost of reconfiguration). This is the optimal length of the reconfiguration period.

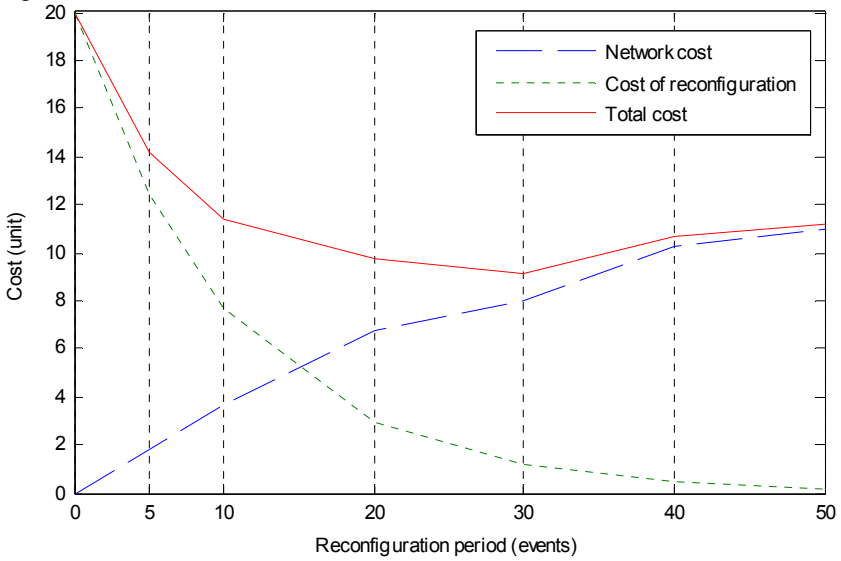

Fig. 43. Calculating the optimal length of reconfiguration period

5.6. Conclusion

In this chapter, I showed that reconfiguration of dynamic light-trees is clearly beneficial for transport network operators. Several heuristic methods for maintaining the multicast tree close to the optimal Steiner tree were applied and their performance was measured. In case of reconfiguration, the optimal topology was determined by ILP. Lots of cost (including network resources, e.g. O/E converter units and wavelength capacity) can be spared by restoring the optimal topology of the tree. Since, after the reconfiguration, the tree diverges quite quickly from the optimal one, frequent reconfiguration is required. I showed that the reconfiguration period has an optimal length, if we also consider the cost of reconfiguration in addition to the network cost. Reconfiguration seems to be especially useful if grooming is not possible. Still, a number of technical challenges must be addressed to make reconfiguration practical, like the seamless switch over of traffic from the old to the new tree.

PART 2
Peer-to-Peer Traffic Identification in IP Networks

6. Introduction to Traffic Identification

This chapter gives a brief overview of traffic identification approaches, considering especially those that can be applied to identify P2P traffic. It also discusses the basic concept of P2P, together with types and classification of P2P systems. Finally, it describes the properties of the traffic measurements used later in the process of identification and analysis.

6.1. Ways of Identification

In general, the issue of application identification inside an IP network is not trivial. This is even more complicated and difficult in the case of P2P applications. Early applications (including early P2P systems) often used TCP with some fixed ports for communication. In these cases the *traditional port-based traffic monitoring and classification* could be used to measure their traffic.

Nowadays the dramatic growth of P2P usage accompanied by the huge bandwidth consumption, together with the problematic content copyright concerns lead to some interventions from network operators such as traffic limiting or blocking. To overcome these limitations, newer P2P applications can use both TCP and UDP connections with *arbitrary ports* for messaging and data transmission. These improvements make the detection of P2P traffic a challenge.

There are several ways for identifying network traffic. The first question is whether to use payload information. *Payload based identification methods* can reach higher accuracy. In this case, however, appropriate payload *application patterns* are necessary, which are not always available or the reliability is not proven. Too general patterns lead to many false positive results. Many applications (especially those carrying illegal or unwanted content) try to hide themselves by encrypting the payload. Capturing and storing payload is more resource intensive; it is thus not always feasible on high-bandwidth links. If it is applied, then usually not the whole content is saved, but only the first few bytes. All patterns are designed to rely on the first few bytes of the packet, following bytes are not taken into account.

Payload based identification often raise legal issues and concerns as well. By capturing packet content it is possible to access user data which is regarded private information. Some header information is also considered confidential (e.g., IP address of the customer or anything that can refer to the identity of the user) in many countries. One option to tackle this problem is to anonymize confidential fields of headers, for example by applying hashing (i.e. a one-way function like MD5). However, packet payload cannot be hashed. As a consequence, operators are restricted in capturing packet payload and they are especially not willing to give out such datasets to a third party.

Identification without payload information is more limited, since only packet headers and timing information can be used. However, the storage of logged data requires less disk space, which makes offline processing feasible. This is especially useful, since we can discover connections between flows in different time periods. In addition, many statistical properties of flows, packets and timings can still be monitored. This approach is collectively called *flow dynamics based identification*.

In most measurements, payload information is not available. In addition, there are application (e.g., Skype most of all), where all communication among the network entities is encrypted and hence do not have a recognizable payload pattern.

Traditional port based identification cannot be applied either, since most of the target applications (e.g., file sharing clients, etc.) use arbitrary communication ports. Skype also chooses a random port upon installation. However, once it is installed, the used port is not changed anymore, unless the user changes it manually.

For all these reasons, I decided not to use any payload information and to prefer the flow dynamics based approach in the identification methods.

6.2. P2P Concept

A P2P computer network uses diverse connectivity between participants to use common resources (e.g., bandwidth) more efficiently in contrast to the traditional concept of central server based computer networks. P2P networks are typically used for connecting nodes via largely ad hoc connections, which makes the whole structure more stable and fault tolerant by avoiding central entities.

The main characteristic of a P2P system is that it is not built around the server and client concept, but on the *cooperation of equal peers* that simultaneously function as both “clients” and “servers”. This principle involves the adapting nature of P2P systems as individual peers join or leave the network. The most common use of the P2P principle is multimedia file sharing (movies, music files, etc.), which frequently contain very large files (megabytes, gigabytes) in contrast to the typical small size of web pages (kilobytes).

From the beginning of the new millennium the Internet traffic characteristics show a dramatic change due to the emerging Peer-to-Peer (P2P) applications. Starting from the first popular one (Napster) a number of P2P

based multimedia file sharing systems have been developed. The traffic generated by these P2P applications consumes the biggest portion of bandwidth in campus networks, overtaking the traffic share of the World Wide Web [136], [165]. Some studies state that as much as 70% of broadband traffic is P2P [119][120].

6.2.1. Classifications of P2P Networks

P2P systems can be classified according to several criteria. First, they can be used for different purposes, including primarily *file sharing* (BitTorrent [127], Direct Connect, eDonkey, FastTrack, Gnutella, Napster, etc. [121]), *Internet telephony* (Skype [126], PeerMe [123] or Avaya P2P SIP phone solution: one-X [124]), *media streaming* (PPLive, PPStream [121], PeerCast [122]) and (not necessarily real-time) media distribution (Joost [125]). There are quite *many protocols and solutions* and especially a *countless number of clients*. Sometimes the name of the protocol/network and the name of the most popular client are confused, because they are strongly associated. Some P2P systems using proprietary protocol are only accessible by a dedicated client (e.g, Joost); there are also examples for open source solutions (e.g., PeerCast).

We can also differentiate *centralized* and *decentralized P2P systems*. In both case the available resources are hosted by the peers. However, in a centralized solution (like Direct Connect) there is a central entity, which keeps information on peers and resources (announced by the peers). The central "server" also responds to requests for that information, for example it performs indexing and searching and tells which clients possess the necessary resource (e.g, a file).

A further aspect of classification is whether the system is *structured or unstructured*. In an unstructured system (like the early version of Gnutella) none of the peers play special role (like super nodes in Skype network or ultra peers in later version of Gnutella). Searching can be quite problematic in *flat* unstructured systems, since messages have to be distributed by some form flooding. In a structured network, however, not all the peers are perfectly equal; usually more reliable peers are promoted and special roles are assigned to them. Distributed hash tables (DHTs) can be also considered as a form of structuring. DHT refers to a class of decentralized distributed systems that provide a lookup service similar to a hash table: *(name, value) pairs* are stored in the DHT, and any participating node can efficiently retrieve the value associated with a given name. Responsibility for *maintaining the mapping from names to values is distributed among the nodes*, in such a way that a change in the set of participants causes a minimal amount of disruption. There are several protocols and implementation of DHT (see [128]). This allows DHTs to scale to extremely large numbers of nodes and to handle continual node arrivals, departures, and failures. DHT solution is implemented by – among others – eMule [129] (the successor of eDonkey) and the latest versions of many BitTorrent clients. In a *hybrid P2P system* [130], flooding and DHT are both employed for content locating depending on the popularity of the content.

Some clients (e.g., Skype, several BitTorrent clients) may apply *encryption* to render identification or blocking of unwanted or illegal traffic more difficult.

7. Traffic Measurements

Several traffic measurements were performed in both fixed and mobile environments, for both validation and traffic analysis purposes. The summary of the data sets is presented in Table III.

The first two measurements (called *Callrecords 1* and *Callrecords 2*) were carried out in the fixed network of one of the largest Internet providers in Hungary [201]. In the chosen network segment, the traffic of about 1000 ADSL subscribers is multiplexed before entering the ATM access network. The logging was performed in one of the routers at the border of the access and the core networks. Further details of the measurement configuration are presented in Fig. 44.

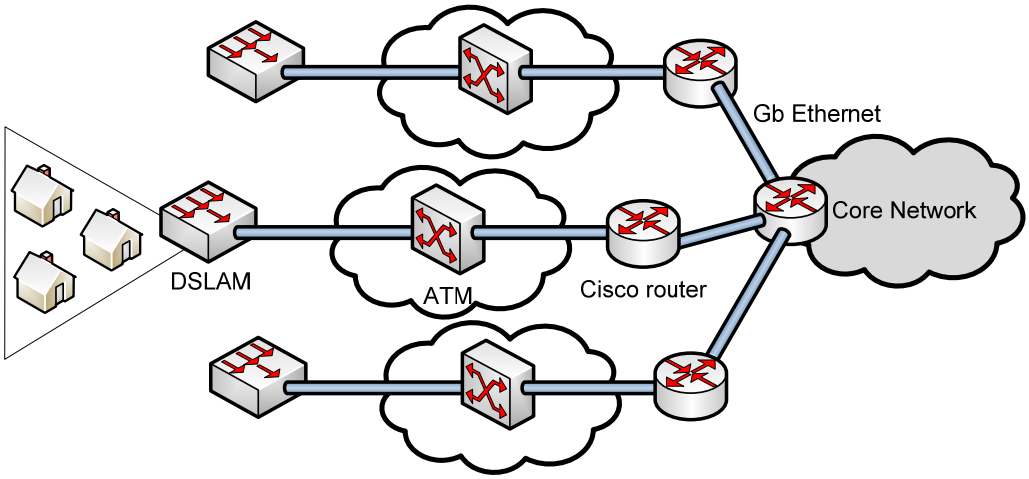

Fig. 44. Location of traffic measurement

In the third fixed network measurement (*Verification*) the traffic of my university department was logged, carrying the traffic of about one hundred users. I performed this experimental traffic logging to validate the Skype identification method.

Skype traffic identification and analysis was also performed on a dataset recorded in the network of one of the largest mobile service providers in Hungary. The trace was captured at the border of the backbone network where the traffic of the mobile users is aggregated. Only users having 3G or HSDPA connection were included in the dataset. Some of these users may be able to use Skype using a smart-phone or a laptop with PCMCI net-card. In fact it is really worth for them to use Skype because it is a cheaper alternative for initiating voice calls. If per minute costs are too expensive compared to per megabyte cost, many users will switch to VoIP applications instead of ordinary mobile calls. Mobile service operators are very much interested in how many users choose the transferred data based (usually flat rate) voice calls instead of the per minute (or per second) charged traditional calls. This information can be very useful for the operator to determine the tradeoff in price between the two types of services.

The *Mobile measurement* was almost 3 days long and both inbound and outbound TCP and UDP traffic was logged. The bandwidth of the users was limited in both directions (384 Kbps uplink, 384 Kbps or 3.6 Mbps downlink). However, according to the measurements, this limitation did not cause a serious problem for Skype, since the typical one-way data rate of speech calls (about 40 Kbps) is under this limit. The aggregate traffic of all mobile users in the Budapest area passed through the measurement point. A *Mobile verification* measurement was carried out as well.

In all measurements, only IP and TCP/UDP headers were logged. Flow level information was extracted from the traces, including source addresses, ports, packet number, transmitted bytes, start time and end time of the flow. Packet level information (packet size, packet arrival-time) was also preserved and used for the identification.

Table III. Summary of the collected datasets

Data sets	Time of measurement From - To	Number of flows	Total traffic
Callrecords 1	22nd Jul. 2005 11h 23rd Jul. 2005 11h	23 796 956	458.04 GB
Callrecords 2	25th Apr. 2006 11h 26th Apr. 2006 11h	36 896 516	766.02 GB
Mobile measurement	4th June 2007 18h 24 7th June 2007 15h 58	29 334 814	667 GB
Verification	7th Nov. 2006 10h 8th Nov. 2006 16h	1 663 752	61.42 GB
Mobile verification	9th Nov. 2007 8h 50 10th Nov. 2007 7h 47	30 634	0.8 GB

Both inbound and outbound traffic were logged, since data from both directions is necessary for accurate identification. However, the methods can also be applied if only one direction is available. In this case, the reliability decreases since inbound and outbound speech flows cannot be paired to each other. Therefore, I recommend using the methods in edge routers where inbound and outbound traffic flows through the same router. This is not necessarily true in backbone routers because of asymmetric routing.

8. Identification of P2P Traffic

Recent measurement studies report that a significant portion of Internet traffic is unknown. It is very likely that the majority of the unidentified traffic originates from peer-to-peer (P2P) applications. However, traditional techniques to identify P2P traffic seem to fail since these applications usually disguise their existence by using arbitrary ports. In addition to the identification of actual P2P traffic, the characteristics of that type of traffic are also scarcely known.

The main purpose of this chapter is twofold. First, I propose a *novel identification method* to reveal P2P traffic from traffic aggregation. The method does not rely on packet payload, so the difficulties – arising from legal, privacy-related, financial and technical obstacles – can be avoided. Instead, the method is based on a set of heuristics derived from the robust properties of P2P traffic. I demonstrate the method with current traffic data obtained from one of the largest Internet providers in Hungary. I also show the high accuracy of the proposed algorithm by means of a *validation study*.

Second, several results of a comprehensive traffic analysis study are reported in the chapter. I show the daily behavior of P2P users compared to the non-P2P users. Important finding about the almost constant ratio of the P2P and total number of users are presented. Flow sizes and holding times are also analyzed and results of a heavy-tail analysis are described. Finally, I discuss the popularity distribution properties of P2P applications. The results show that the unique properties of P2P application traffic seem to fade away during aggregation and characteristics of the traffic will be similar to that of other non-P2P traffic aggregation.

A number of studies have been published in the field of P2P networking. Papers [131]-[138] and [164] focus on the measurement of different P2P systems like Napster, Gnutella, KaZaA, and the traffic characterization and analysis of P2P traffic providing some interesting results of resource characteristics, user behavior, and network performance. Several analytic efforts to model the operation and performance of P2P systems have been presented so far. Queuing models are applied in [139]-[140], while in [141]-[143] branching processes and Markov models are used to describe P2P systems in the early transient and steady state. P2P analysis using game theory is presented in [153], [154], among others. Other studies, e.g. [144]-[147], are concerned with the effective performance and the QoS issues of P2P systems. In addition, many papers [148]-[152] indicate various possible applications using P2P principles. Further approaches propose structured P2P systems using Distributed Hash Table (DHT) with several implementations like Pastry, Tapestry, CAN, Chord [155]. The P2P traffic characteristics are not fully explored today and there is a tendency that they will be even more difficult to analyze.

Since P2P networks are often associated with illegal file sharing, some operators prohibit their usage. Therefore recent popular P2P applications disguise their generated traffic resulting in the problematic issue of traffic identification. The accurate P2P traffic identification is indispensable in traffic blocking, controlling, measurement and analysis. This problem is very complicated since, on one hand, the applications are constantly evolving using new techniques to remain unnoticed, and on the other hand, new applications appear from time to time. These applications might even be unknown to the operator. However, the issue is touched upon in only a few papers and the proposed solutions still have some drawbacks. Therefore the main motivation was to find a reliable method for the detection of the traffic of as many P2P applications as possible.

The workload characteristics of peers participating in some P2P systems have been examined in several papers as mentioned above. However, from the aspect of service providers only little useful information can be gained from these studies. The service providers are less interested in the detailed activities of some particular P2P software, rather in the traffic generated by peer users. This study concentrates on those factors and characteristics of P2P communications which have an impact on the P2P traffic aggregation.

8.1. State of the Art of P2P Traffic

Concerning related work, I give an overview of a few papers dealing with the issue. Firstly, the *crawl-and-probe* method [133] should also be mentioned. Authors periodically "crawled" the P2P system to gather instantaneous snapshots of a subset of user population and then sent probes to users to directly measure some of their properties. This method cannot collect users' traffic activities.

A method based on *port properties* is presented in [156]. The authors note that a substantial number of flows cannot be identified by the mapping method from flows to applications. They classify the unknown flows by size and assume that the traffic is P2P if the flow transmits more than 100kB in less than 30 minutes.

In [157] P2P traffic is identified based on the *application signatures* found in the payload of data packets. Authors showed that typical sets of strings are identified in the packet payload generated by some P2P applications. The method can be implemented for online tracking of P2P traffic by examining several packets in each flow. It is reported that the technique works with very high accuracy. It seems that the signature-based method can provide the most accurate P2P traffic detection. This method might be used in traffic investigation of one or several particular P2P systems. However, there are also some drawbacks. The very first challenge is the

lack of an openly available, up-to-date, standard, and complete P2P protocol specification [157]. Since P2P protocols are continuously developed, the present traces will not surely exist in tomorrow's traffic. Furthermore, an increasing number of P2P protocols rely on encryption, so payload matching cannot be applied in these cases.

The authors of [158] want to improve the efficiency of traffic identification on high-speed links by introducing *packet sampling*. They have applied the method to identify BitTorrent traffic.

A similar payload-based method is presented in [159]. This paper also proposes two *heuristics for identification* of P2P traffic without payload examination. It is reported that 95% of the results provided by the payload method is identified by the proposed heuristics (false positive ranges between 8% and 12%). It should be mentioned that the payload examination only tries to detect the traffic generated by certain P2P applications. We cannot know for sure all possible P2P applications people use. Nevertheless, the idea is very promising. The identification of P2P traffic aggregation should be done by heuristics which are based on some common properties of P2P communications instead of examining particular P2P applications.

The method described in [160] also works without payload information. Besides flow identification by ports, it proposes the estimation of unknown traffic by relating it to preceding, known traffic. The authors argue that traffic induces other traffic so there is a possibility to identify unknown traffic which was induced by known traffic. Since this principle cannot guarantee the correct identification, some additional statistics are also used to increase accuracy of the decision method. The evaluation shows that the method has an identification gain of 1-3% compared to the traditional port based approach, but the average hit rate is still 60-70%.

Kim et al. in [161] provide a method which is an improvement of the network port-based application detection. Their main idea is to discover the relationships between flows that belong to a particular P2P application and then use this information to put measured flows into groups. Flow groups together with a set of typical P2P application ports are used to determine whether a group of flows is generated by P2P applications or not. The disadvantage of this method is that it is very difficult to find appropriate typical relationships between flows of a given P2P application. In addition, as presented in the paper, there is still more than 40% of the total traffic which cannot be identified.

Constantinou et al. [162] suggested a heuristic P2P traffic classifier relying on transport layer information only. The method constructs a graph from the set of connections between the hosts, assigns "levels" to hosts, and calculates the diameter of the network. Then it applies heuristics (e.g., the number of connections is over a certain thresholds) to select P2P connections. The authors claim to be able to identify unknown P2P applications as well because it does not rely on any specific P2P protocol. In this sense it is similar to my proposed algorithm.

Similarly, the authors in [163] proposed a simple heuristic classification method based on discreteness of remote hosts (RHD) to identify BitTorrent-like traffic. In each time period for each local host they calculate the RHD value according to how many distinct remote stub networks the particular user connects to. If the RHD exceeds a certain threshold, then the host used BT-like application. The algorithm seems to be effective against all P2P applications, though it depends on how we define "BT-like application". The reported average accuracy of the method is 90%.

An identification system for pure P2P applications is given in [166]. The method is specialized for the Winny application, which was at that time the most popular P2P application in Japan. It uses the server/client relationships among peers. Some evaluation results of the method are also presented.

Numerous papers present identification methods based on *machine learning techniques*. The accuracy of these methods has a high variance (70-90%) depending highly on the training and evaluation datasets. However, accuracy higher than 90% is reported by some authors.

Shen Fuke et al. in [167] trained and applied *Back Propagation Neural Network* – based on connection patterns stemming from P2P networks – to identify P2P traffic. Raahemi et al. also applied supervised machine learning technique, namely *neural networks* [168] and *decision trees* [169], to classify P2P traffic. They preprocessed and labeled the data, and built several models using a combination of different attributes for various ratios of P2P/NonP2P in the training data set.

The same authors in [170] applied a streaming data mining methods, namely *Concept-adapting Very Fast Decision Tree* (CVFDT), to identify P2P traffic "on the fly". The identification was carried out at packet level by using packet headers only. 6 attributes (packet size, source IP, protocol, etc.) were extracted to train CVFDT before deploying it at the campus gateway to test their method. The authors claim to have Sensitivity, Specificity, and Correctness measures around 0.9367, 0.9680, and 0.9585, respectively.

The method proposed in [171] uses optimized *support vector machines* for learning and relies on transport layer information only. The experimental results show that the proposed method has high efficiency and promising accuracy.

Zuev et al. in [172] proposed a *supervised machine learning* approach to classify network traffic. They started by allocating flows to several predefined categories (e.g., Mail, WWW, P2P, Games, etc.). They then utilized 248 per-flow discriminators (characteristics) to build their model using *Naive Bayesian analysis*. They evaluated the performance of the solution in terms of accuracy (the raw count of flows that were classified correctly divided by the total number of flows) and trust (the probability that a flow that has been classified into

a class, is in fact from that class). The reported accuracy is 83%. The scalability of the approach is questionable as it involves too many discriminators.

The authors of [173] presented a methodology and selection of five (flow level) traffic discriminators and applied *cluster analysis* (*k*-means algorithm) to identify P2P applications. The results indicate an accuracy of 90%.

The main disadvantage of these methods is that they do not make use of the connection (like parallelism, consecutiveness) between flows but dealing with individual flows (and their properties) only. In addition, I tried to avoid using any payload information and build my methods purely on flow dynamics.

8.2. A Heuristic Method for P2P Traffic Identification

In this section, I introduce my method identifying P2P traffic (precisely P2P data flows) from an aggregate dataset. The proposed heuristic method consists of six steps, each being associated with a group of P2P flows to be identified. The heuristics (excluding two steps applying classic port-based approach) utilize some robust properties of P2P traffic. The whole identification process is depicted in Fig. 45 and described in details afterwards.

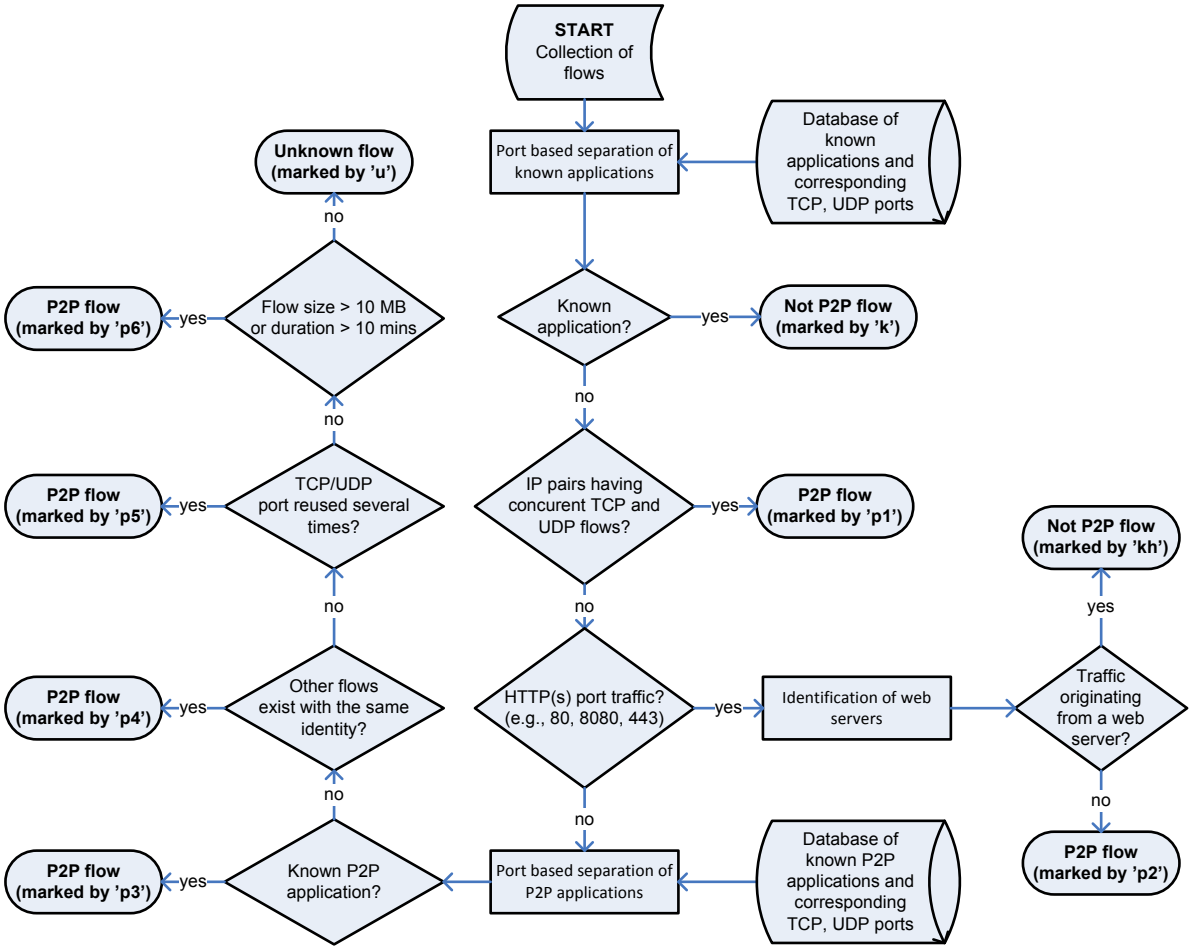

Fig. 45. Flow chart of the P2P traffic identification method

0. While port based analysis is less accurate to identify P2P traffic, it is still appropriate to distinguish traffic generated by common applications. The search of these applications and their communication ports, considering both TCP and UDP protocols, results in a table of applications and ports (see Table IV). Flows having one of these ports as a source or destination port are marked as "known application" and excluded from further analysis. Web ports (80, 443, 8080, etc.) are exceptions and not included in the table, since HTTP ports are not only used for web surfing, but also by some P2P applications, e.g., KaZaA. The separation of web and P2P traffic is carried out by the second heuristic.

Table IV. Some examples of common application ports

Application	Port(s)	TCP/UDP
MSN Messenger	1863	TCP
Yahoo Messenger	5101, 5050	TCP
NETBIOS	135, 137, 139, 445	TCP and UDP
NTP	123	UDP
DNS	53	TCP and UDP
POP3	110	TCP
FTP	20, 21	TCP
…	…	…

1. The first heuristic is based on the fact that many P2P protocols, e.g. eDonkey, Gnutella, Fasttrack, etc., use both TCP and UDP transport layers for communication. Reasonably the unreliable UDP is often used for control messaging, queries, and responses while data transmission relies on TCP. However, the large volume of UDP traffic observed in the measurement data indicates that UDP could also be used for data transfer. Thus, by identifying those IP pairs which participate in concurrent TCP and UDP connections, we can state that the traffic between these IP pairs is almost surely P2P. This heuristic is similar to what is proposed in [159] with a little difference. It should be noted that some other common applications like NETBIOS, DNS also utilize both TCP and UDP. [159] employs post-processing to extract this kind of traffic from the result of the heuristic. In contrast, this is not necessary in this case since it has been done already in the initial (0^{th}) step: these applications are among the common ones.
2. The second heuristic tries to separate web and P2P traffic from flows using HTTP/SHTTP ports, i.e. 80, 8080, 443, etc. There is a typical, noticeable difference between P2P and web communication of two hosts: in general, web servers use *multiple parallel connections* to a client in order to transfer web pages text (HTML source code) and images (also music, video contents in some cases). In contrast, data transmission between peers consists of one or more consecutive connections, i.e. only one single connection can be active at a time. This property is used to identify web servers and then the traffic originating from them. The traffic using HTTP ports is divided into groups of individual IP pairs. The web server is the one with the IP address in the HTTP port's side, which has parallel connections to its pair. Two cases are differentiated: if the IP address of the web server belongs to the outside IP domain, it is likely to be a public web server. Then all the HTTP traffic originated from this server is marked as web traffic. In the other case only parallel flows with HTTP ports are marked as web traffic. The rest of this traffic group is P2P traffic. Unfortunately we realized that the most popular streaming applications (Windows Media Server, Helix Server, and Quick Time) can also use HTTP ports for transferring video or audio content. Since streaming data flows do not necessarily have parallel connections to the web server, these data flows might mistakenly be identified as P2P flows. Although the amount of such streaming flows in the data sets seems to be small, flows marked as P2P in this step are excluded from later analysis.

Table V. Network ports used by some popular P2P systems

P2P applications	TCP/UDP ports
Edonkey (eMule, xMule)	TCP 2323, 3306, 4242, 4500, 4501, TCP 4661-4674, 4677, 4678, 7778
FastTrack (*former* KaZaA)	TCP 1214, 1215, 1331, 1337, 1683, 4329
BitTorrent	TCP 6881-6889
Gnutella	TCP 6346, 6347
DirectConnect (DC++,BCDC++)	TCP 411, 412, 1364-1383, 4702, 4703, 4662
ShareShare	TCP 6399, UDP 6388, 6733, 6777
Freenet	TCP 19114, 8081
Napster (File Navigator, WinMX)	TCP 5555, 6666, 6677, 6688, 6699-6701, 6257
SoulSeek	TCP 2234, 5534
Blubster	TCP 41170

3. In the next step, P2P traffic is selected using default ports of P2P applications. P2P software often defines default ports for communication. It is true that in most cases peer users can change it to any arbitrary port (but it is not frequent since peer-to-peering is usually not prohibited for home users) or a port can be dynamically chosen automatically or when firewall or port-blocking is observed. This step cannot detect all P2P connections, but once the traffic is collected we can be almost sure that it is from those concerned P2P systems. A table of well-known ports used by some popular P2P applications is collected for this step (see Table V for details). Flows containing these values in source_port or dest_port are all marked as P2P.
4. In normal TCP/UDP operation, at least one of the two ports is selected arbitrarily. It is not likely that flows with the same flow identity *(source IP, destination IP, source port, destination port, protocol byte, TOS)* exist in relatively short measurements. This happens, though, in the case of P2P connections, if both source and destination peers dedicate a fixed port for data transfer. Download of a file is often executed in several smaller chunks. Therefore multiple flows with the same flow identity might be generated by P2P software. This is the basis of this heuristic: if at least two flows exist with similar flow identity, these flows are likely generated by a P2P application.
5. For the same reason as the above heuristic, it is not probable that a host (IP) will repeatedly choose a given arbitrary port for TCP/UDP connections unless it is a server. Web servers and other common server traffic is extracted by the previous heuristics, thus it is safe to introduce the next heuristic: if an IP address uses a TCP/UDP port more than 5 times in the measurement period, then that {IP, port} pair indicates P2P traffic. The selected upper threshold (5) is a rule of thumb established empirically.
6. The last heuristic is based on the fact that objects of P2P downloads often have large size from some MB in case of music files or smaller applications to hundreds of MB in case of video files and larger software packages. In addition, peer users are patient. P2P downloads can last some ten minutes or hours. This heuristic marks those flows as P2P, which have flow sizes larger than 1 MB or flow length longer than 10 minutes.

The 0^{th} step of the method aims to identify a set of widely used Internet applications (except P2Ps) based on well-known port analysis. 3^{rd} step also applies the classical port based approach to identify P2P application using fixed communication ports. All other applications, though, are based on heuristics exploiting some robust properties of P2P traffic. Note that the order of the steps is crucial: each step filters out a set of flows (meeting the appropriate criteria) and subsequent steps engage only the remaining flows.

8.3. Verification of the Identification Method

In order to examine the robustness of the heuristics presented in Section 8.2, a validation measurement was carried out. In this measurement, besides gathering general and aggregated information of the traffic flows, the name of the corresponding application was also recorded. This allowed validating the correctness of the proposed P2P traffic identification method.

The measurement program was written in C++ and used the *pcap library* to capture the incoming packets. Upon the arrival of a new packet the program first determined which flow it belonged to, then updated the flow information, namely the number of packets, number of bytes and the end-timestamp. Both TCP and UDP flows have been identified by their source and destination IPs and ports. In addition, in case of TCP the SYN and FIN flags were also used to separate flows, while for UDP a timeout of 5 minutes was used.

The measurement collected the traffic generated by two Linux PCs running SMTP and web servers (although with very light traffic), and some P2P applications: *qtorrent*, *valknut*, and *aMule*. These are the Linux clients of the BitTorrent, Direct Connect and eDonkey systems, respectively. To challenge the identification method, default ports of the P2P clients were modified. Several downloads have been initiated, while the P2P clients were also enabled to serve requests of other peers. The measured trace contained more than 120.000 data flows.

Table VI. Validation result of the identification method

Heuristic step	Hit rate (%)
Known Applications	93.01
Heuristic 1	99.91
Heuristic 2 (HTTP identification)	95.35
Heuristic 3	99.79
Heuristic 4	99.97
Heuristic 5	99.51
Aggregate P2P	**99.14**
Aggregate non-P2P	**97.19**

The performance of each heuristic and the overall identification process is presented in Table VI. The hit rate of each heuristics, counted in percentage, is the ratio of the number of correctly marked flows and the total number of marked flows by the heuristics. It should be noted that the hit rate of the 6th heuristics is not shown in the table, because it marked zero flows in this data set. The last two rows in the table show the rate of correctly marked P2P (non-P2P) flows and the total number of P2P (non-P2P) flows in the data set. The result is very convincing for every statistics. The average hit rate is greater than 99%. The amount of unidentified traffic is about 0.1%. The ratio of wrongly marked P2P flows and unidentified P2P flows per the total marked P2P flows are 0.3% and 0.8%, respectively.

Note that these performance parameters were calculated "flow-wise", i.e. the percentage values express the ratio of flows, not the ratio of traffic quantity (bytes). Similar results concerning the traffic quantities might be better.

The validation of the proposed identification method has been done for a relatively small size of measured traffic traces. In this case the possible sources of errors multiply (the expectable behavior used in some heuristics is more likely in large traffic measurements). Moreover, the modification of the default ports of some P2P application made the task more difficult. Nevertheless, the obtained result yields very convincing identification accuracy.

8.4. Identification Results

The following sections (Section 8.4 and 8.5) discuss the analysis results after applying the identification method (described in Section 8.2) on the measurement datasets (*Callrecords 1* and *Callrecords 2*, see Section 7). As described earlier, the traces are the sets of flow information collected in an ADSL domain (see Section 7 for details).

A flag is assigned to each flow record of the database. The flag has the default value of *u*, meaning *unknown* (traffic), and it can be changed during the identification process. The list of possible values of the flag is the following:

- *u*: unknown traffic (flow), default value
- *m*: management traffic
- *o*: other non-TCP/UDP flow
- *k*: known common application (except HTTPs)
- *kh*: flow using Web ports (80, 443, 8080, etc.)
- *pX*: P2P flow, X denotes the heuristic which identifies the flow

First, flows used for management of the routers in the traffic measurement are flagged as *m*. The management traffic is clearly identifiable by the IP addresses of the routers. Afterwards, flows of other IP protocols, which are not TCP and UDP, are marked by flag *o*. This type of traffic may consist of ICMP, IPv6, RSVP, GRE, IPsec ESP, etc. This step is accomplished using the protocol byte information of the flow record. Then the proposed initial phase and the heuristic methods are applied to identify the common application traffic and the set of P2P traffic. The flags of unknown flows keep their default value *u*. The results of the identification procedure are summarized in Table VII. Note that the *Callrecords 1* and *2* data sets did not contain management traffic; traffic other than TCP and UDP was also filtered out in a previous step.

Table VII. Traffic identification results

Flag	# of flows (%)	Volume (%)
Data set	Callrecords 1 inbound	
k, *kh*	63.40	37.71
pX	35.35	62.19
u	1.25	0.11
Data set	Callrecords 1 outbound	
k, *kh*	71.83	15.62
pX	27.63	83.24
u	0.54	1.14
Data set	Callrecords 2 outbound	
k, *kh*	54.85	21.77
pX	44.62	77.50
u	0.53	0.73

Regarding the percentage of flows, non-P2P traffic is dominant; it makes up around 2/3 of the total traffic, while P2P traffic has a share around 1/3. The proportions are a bit closer to each other in the *Callrecords 2* dataset. On the other hand, contributions in terms of traffic volume show a totally different picture. P2P traffic is responsible for approximately 80% of the total traffic in the outbound direction, while it has somewhat smaller share in the inbound direction. This observation can be easily explained by the fact that typical Internet users (and applications) do not generate significant outbound traffic. P2P applications, however, generate significant upload traffic. This results in a much higher contribution of P2P traffic in the outbound direction. The author knows FTTH (Fiber to the home) networks with high bandwidth, symmetric access, where the share of P2P traffic in the outbound direction reaches even 98%. The generally accepted theory is that most customers do not make use of the outbound bandwidth at all, while P2P users fully utilize the outbound bandwidth ever so much is offered. The amount of unknown traffic is minor in terms of both flow number and data amount.

8.5. Traffic Analysis

In this section the analysis focuses on the fundamental differences between the P2P traffic and other Internet traffic (this will be referred to as non-P2P traffic). The comparison is done regarding several aspects of the traffic characterization.

Remember that traffic flows, marked as P2P *p2* by the second step of the heuristics, are excluded from the analysis (see Section 8.2).

8.5.1. Daily Traffic Profile

The daily fluctuation of the traffic is presented in Fig. 46. The upper plots show the total and non-P2P traffic intensities of the *Callrecords 1* data set (including inbound and outbound direction), while the lower one shows the intensity and the flow count of the P2P traffic of the same set.

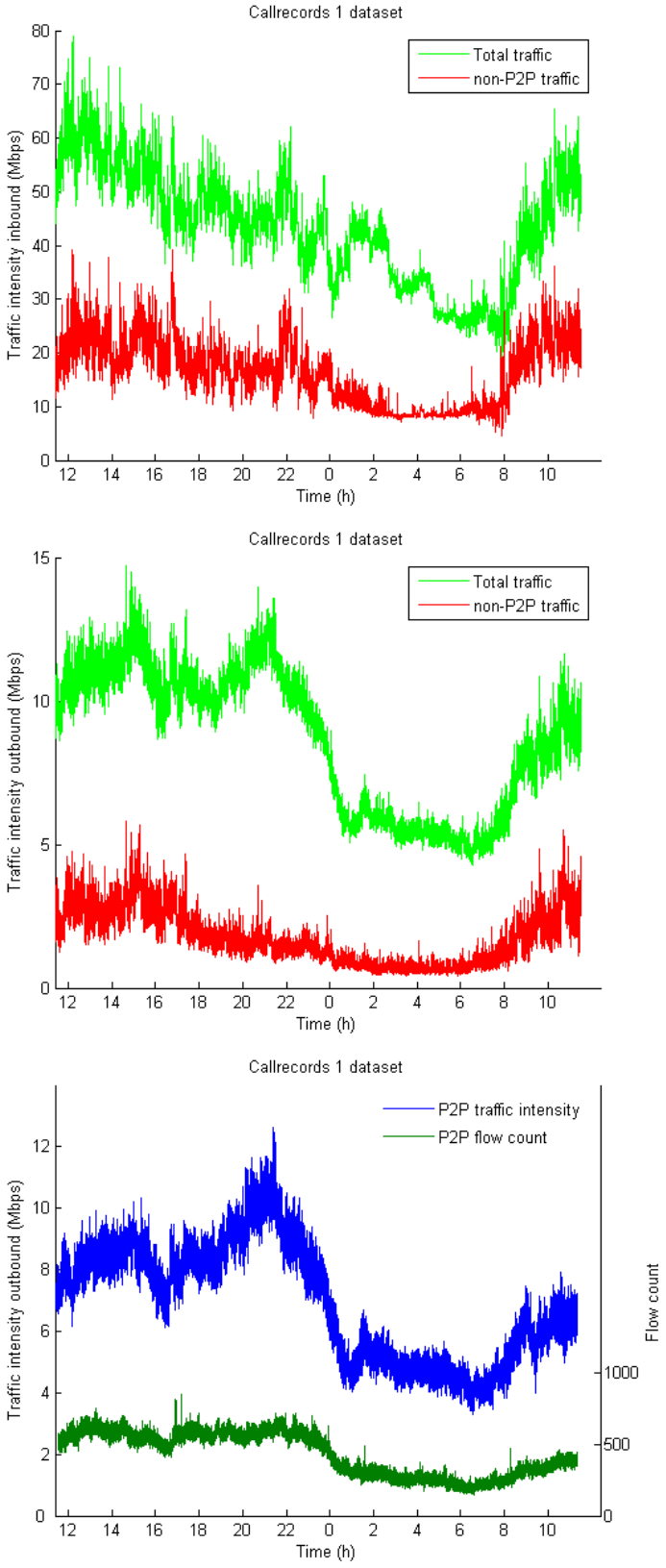

Fig. 46. Traffic intensities from *Callrecords 1* dataset

As observed in general, daily traffic can be divided into two parts: the busy period from around 8h to 24h and the non-busy period from about 0h to 8h. Both P2P and non-P2P traffic follow this daily tendency. In the case of non-P2P applications the traffic intensity ratio between busy and non-busy periods is around ½, i.e., the bandwidth falls to very low values in non-busy period, while in case of P2P applications the decrease of traffic

intensity is somewhat smaller. Consequently, P2P traffic seems to be less variable, more even throughout the day.

This is reasonable since non-P2P users – in general – do not generate traffic during "sleeping time". In contrast, P2P users (in our case also home users) turn on P2P applications and request some audio and video files (some can be very large). Then they leave the system to work over days, even when they are asleep during the night. Basically, the P2P traffic can be steady over time, which can be seen in Fig. 46: the number of P2P flows has small variation (see the lower plot). We still see a certain decrease in the traffic. This may happen because the number of downloadable sources decreases and probably more requests are not added during the night period.

The volume of P2P traffic (see also Table VII), which is about 60-80% of the total traffic, exceeds by far the traffic volume of non-P2P applications. This observation is especially true for outbound aggregate traffic. The reason is that home users do not generate too much upload traffic, except for those users who use P2P applications. As a consequence, the ratio of P2P traffic in the outbound direction is higher than in the inbound direction.

8.5.2. The Number of P2P and Total Active Users

In the measurement environment, Internet subscribers do not have fixed IP addresses. Each time a user connects to the Internet, a dynamic address is assigned to the user. Therefore it is impossible to determine exactly which data flow belongs to which user. However, less error is expected when we choose to associate an individual IP address to a user. Since the ADSL contracts at the present Internet provider do not limit the time of connections, the average connection time is relatively long. It is assumed that during the measurements, which lasted at most 24 hours, only a minimum number of IP address wanderings occurred.

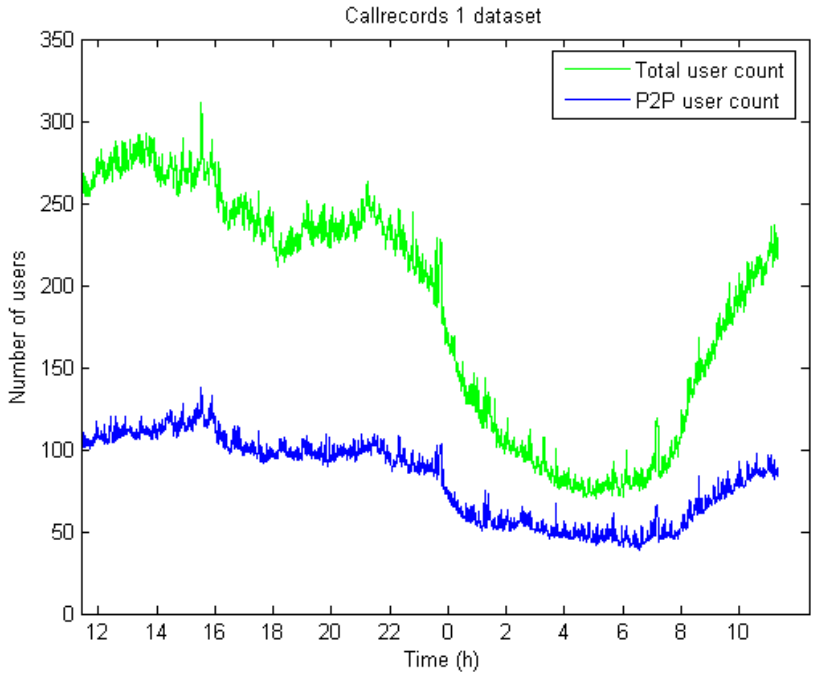

Fig. 47. The average number of P2P users (*Callrecords 1* dataset)

To calculate the number of active users, the number of different IP addresses participating in the flows is counted in every second. Then a sliding window of size of 120 sec and step of 50 sec is applied to smooth the variations caused by communication breaks. One of the results is shown in Fig. 47. The upper curve shows the fluctuation of the total number of users, while the lower curve displays the number of P2P users in the network. A user who uses both P2P and non-P2P applications is counted as P2P user. The total number of users, according to the time shift between busy and non-busy periods, falls as the non-busy period is approached. The lowest number of users is observed in the non-busy period. This similarity is not so striking in the case of P2P users. The answer is similar to the above; it is due to the typical behavior of P2P users/applications.

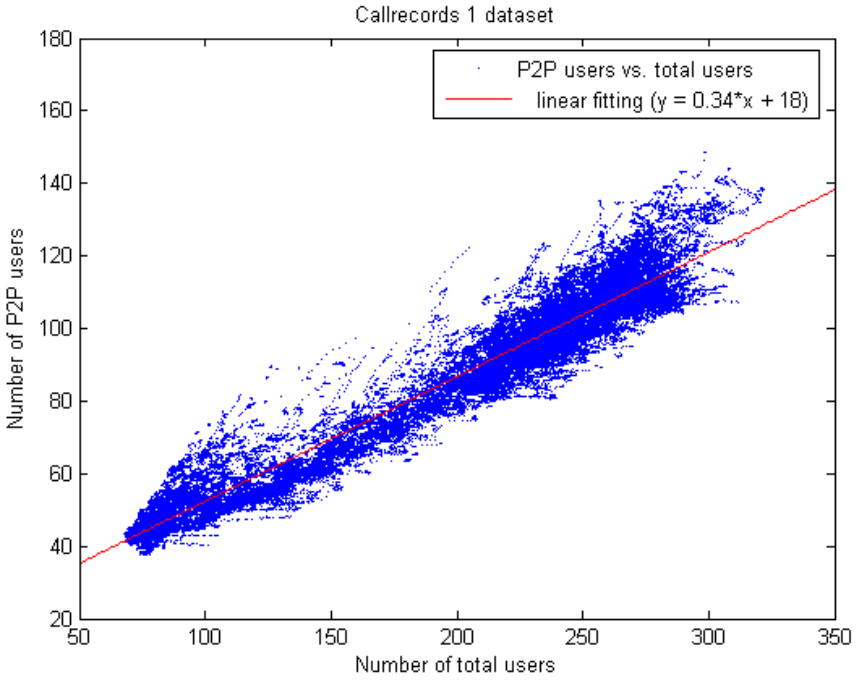

Fig. 48. Relation between P2P users and the total user number (*Callrecords 1* dataset)

The relation between the active P2P users and the total active users is presented in Fig. 48. As seen in the figure, there is a strong linear connection between the two measures. This means that approximately a fixed ratio of active users is using P2P applications. This is quite an interesting finding and it is hard to find a reasonable explanation. However, if this relation is general, it would be very useful for e.g. traffic dimensioning. We plan to verify this relation in several different network environments. The estimated ratio between P2P users and total users is about 0.2 for this data set, 0.3 for the other two sets.

The relation between the number of active (P2P) users and the occupied bandwidth is also investigated. It is shown that a linear connection can be observed in both cases (P2P and non-P2P traffic). However, the variance of data around the assumed linear function is much higher than in the previous case (presented in Fig. 48). In addition, variation is higher and the slope of the line is much lower for non-P2P traffic. This means that P2P users (i.e. users who use P2P applications as well) generate much more traffic in average than those users who use only non-P2P applications.

8.5.3. Flow Sizes and Holding Times

This section presents flow level characteristics and comparison of P2P and non-P2P traffic, namely the distributions of flow sizes (number of bytes transferred) and flow lengths (durations).

Fig. 49 presents the histogram of the flow sizes of P2P and non-P2P applications. No significant divergence was found in this characteristic. In both cases, the plots, disregarding flow sizes smaller than 0.1 KB, nearly follow a straight line on the log-log scale. This indicates a possible *heavy-tailed* distribution, which can be well modeled by *Pareto* distribution, of flow sizes for both the P2P and non-P2P cases.

The *Pareto* distribution [174] [175] is appropriate to describe highly unbalanced distributions, where there is a significant difference between the occurrence probabilities of events, e.g., sizes of human settlements (few big cities, many small villages), size of sand particles, the values of oil reserves in oil fields, the file size distribution of Internet traffic using TCP protocol, etc. The probability density function of Pareto is

$$f\left(x,k,x_m\right) = k \cdot \frac{x_m^k}{x^{k+1}} \text{ for } x \geq x_m,$$

where constant k is the shape parameter and x_m is the location parameter.

P2P (with shape parameter k = *-0.3*) and non-P2P flows (k = *-0.25*) and also for the overall traffic. (The assumptions of Pareto distribution were verified by several heavy-tailed tests: De Haan's moment method, Hill estimator, and QQ-plot [176].) The number of P2P flows exceeding the size of 100 KB is somewhat higher than the number of non-P2P ones, but the difference is not significant.

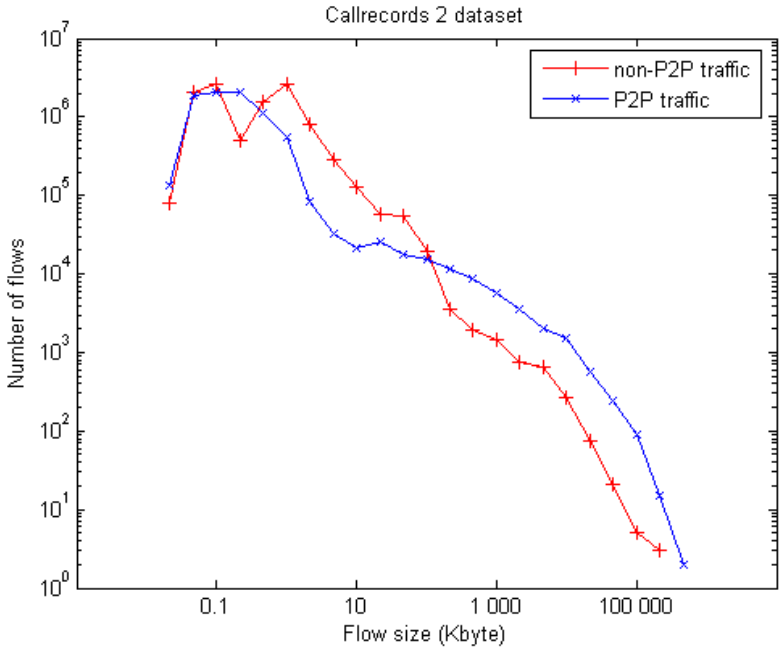

Fig. 49. Histogram of flow size (*Callrecords 2* dataset)

The result seems to be reconcilable with some newer developments of many P2P protocols. Independently of the size of the requested objects, at the beginning the P2P application downloads only a small chunk of the object. The condition of the network and source capacity is estimated from the characteristics of the previous downloads. The size of the next chunk will be determined according to the assumed download quality. Thus, at the end, the P2P traffic (concerning flow size) behaves similarly as the non-P2P traffic.

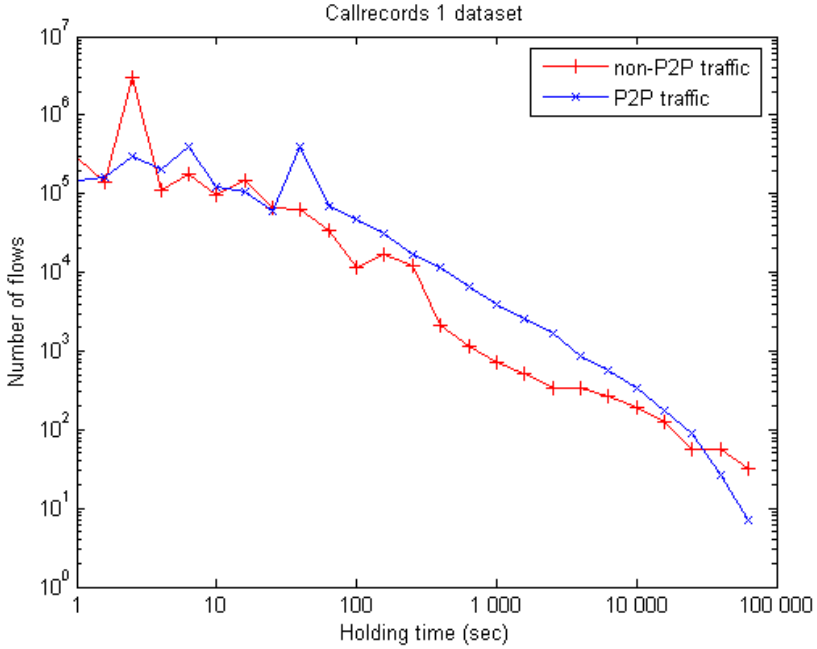

Fig. 50. Histogram of flow durations (*Callrecords 1* dataset)

Similarity is also obtained in the flow duration distribution of P2P and non-P2P traffic (see Fig. 50). Again, the two histograms appear as two parallel lines on the log-log scale. The plot suggests that the Pareto distribution describes well both cases with the same shape parameter of $k=1.4$. The shifting of the curves in the plot agrees with the fact that the total number of P2P flows is one order of magnitude higher than that of the non-P2P ones.

8.5.4. Packet Size Distributions and Typical Packet Sizes

The packet size distributions of packets belonging to P2P data flows and non-P2P data flows are also investigated. The histograms of packet sizes for P2P and non-P2P traffic are shown in Fig. 51. Data packets with size close to the MTU (Maximum Transfer Unit) appear frequently in the flows in both cases, and small packets around the minimum packet size are also common. Certain packet sizes have significant deviation from the average. However, these small excursions do not make the packet size distribution of P2P and non-P2P traffic different. A clear trend is visible for both point sets in Fig. 51.

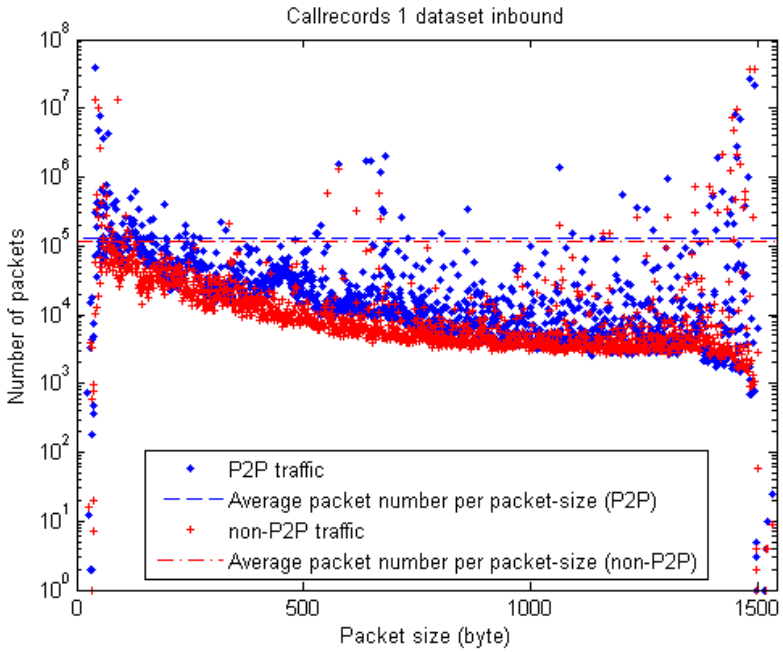

Fig. 51. Packet size distribution of P2P and non-P2P traffic

The huge deviation in certain packet sizes is due to certain applications. I tried to determine those applications that are responsible for this deviation by causing certain packet sizes to appear more frequently in the traffic. The average of the histogram values was calculated for both P2P and non-P2P traffic and plotted by dashed line(s) in Fig. 51. This average value proved to be a good threshold, because it nicely separates frequent and infrequent packet sizes. Packet sizes over this threshold were investigated. Remaining packet sizes, where the histogram value is under the threshold, were not investigated. This way the computational complexity of the subsequent steps could be reduced.

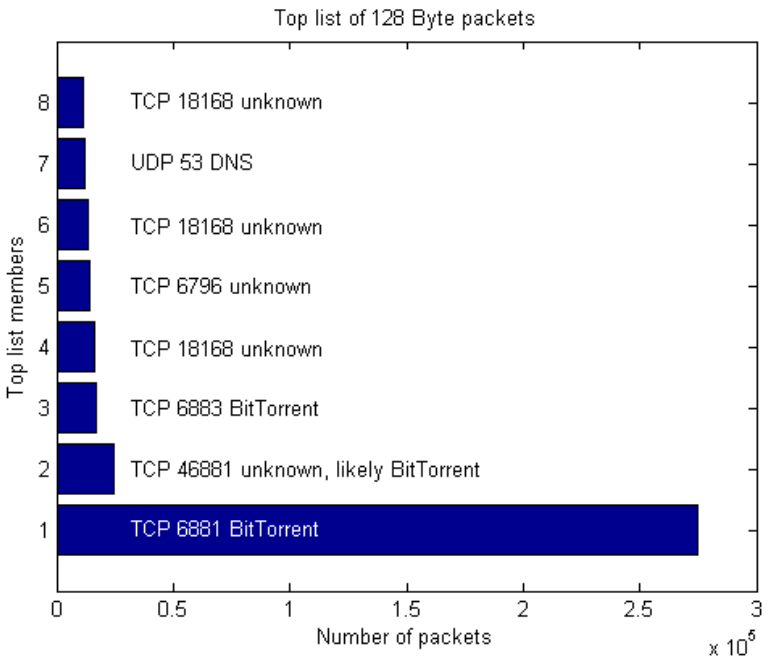

Fig. 52. Top list of source ports and applications for 128 byte data packets

After the filtering it was calculated how data packets, belonging to a certain packet size, are distributed between source TCP (or UDP) ports. The "top list" of source ports was evaluated for every packet size which was not filtered out in the previous step. The application which generated the data packet was identified based on the source port. Of course only applications with surely known communication ports could be identified. An example can be seen in Fig. 52 for 128 Byte data packets, suggesting that 128 Byte data packets are typical for BitTorrent application.

A more detailed list of typical applications for different packet sizes is shown in Table VIII. The "trustiness" measure means the percentage of packets with the given packet size belonging to the certain application. Note that an application for a certain packet size can be significant, even if the trustiness value is not so high (BitTorrent is significant for 128 byte packets, although the trustiness value is "only" 44%, see Fig. 52).

Table VIII. Typical applications for different packet sizes

Packet size (byte)	Typical application	Trustiness (%)
89	SMTP	97.63
528	Gnutella	67.80
128	BitTorrent	44.86
86	eDonkey, BitTorrent	41.98
54	Remote Desktop	35.28
46	POP3	31.36
1200	eDonkey	29.97
58	eDonkey	24.85
64	eDonkey	21.64
120	eDonkey, DC++	21.03
74	eDonkey, BitTorrent	20.76
49	POP3	15.94
52	eDonkey	14.58
167	BitTorrent	14.25
50	MSN Messenger	9.74

8.5.5. Popularity Distribution

The users (assumed to have a one-to-one association with the IP addresses) have been ranked according to the total amount of downloaded traffic. The amount of traffic is plotted against the position in the ranking in Fig. 53. The skewness in the popularity distribution of P2P systems is also justified in our analysis as in many studies of P2P traffic [132] [133]: the top 10% of P2P users are responsible for more than 90% of total download traffic.

It is more interesting, however, how P2P rank curve differs from other Internet traffic. The analysis shows that the head of the rank curves are similar, but the tails are different. Going from right to left on the rank curve the downloaded traffic (per user) decreases very fast in the case of P2P users, which means that there is a huge split between P2P heavy users and hobby P2P users. In contrast, the degree of traffic volume decay in case of ranked non-P2P users is very slow. The average non-P2P users create relatively stable traffic when they access the Internet: reading daily news, chatting with friends, etc.

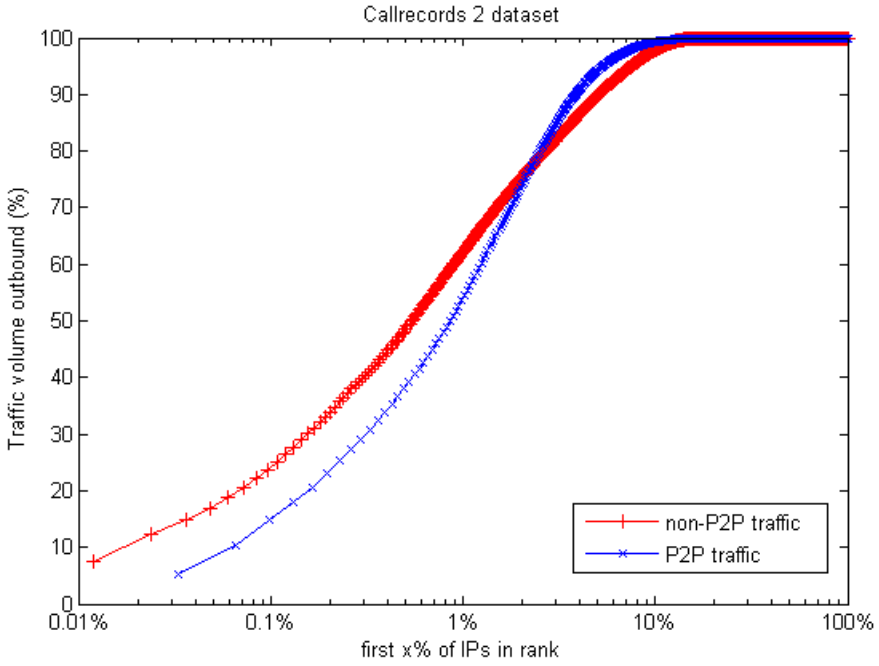

Fig. 53. Traffic volume of ranked IPs (*Callrecords 2* dataset)

In the top region (about 10%) of the rank curve the popular Zipf's law [177] seems to be accurate to describe both P2P and non-P2P traffic popularity. As seen in Fig. 54, the two almost linear plots of P2P (marked by +) and non-P2P IP rank (marked by x) with an approximate slope of -1 indicate that the standard Pareto distribution is a suitable model for top ranked users' traffic.

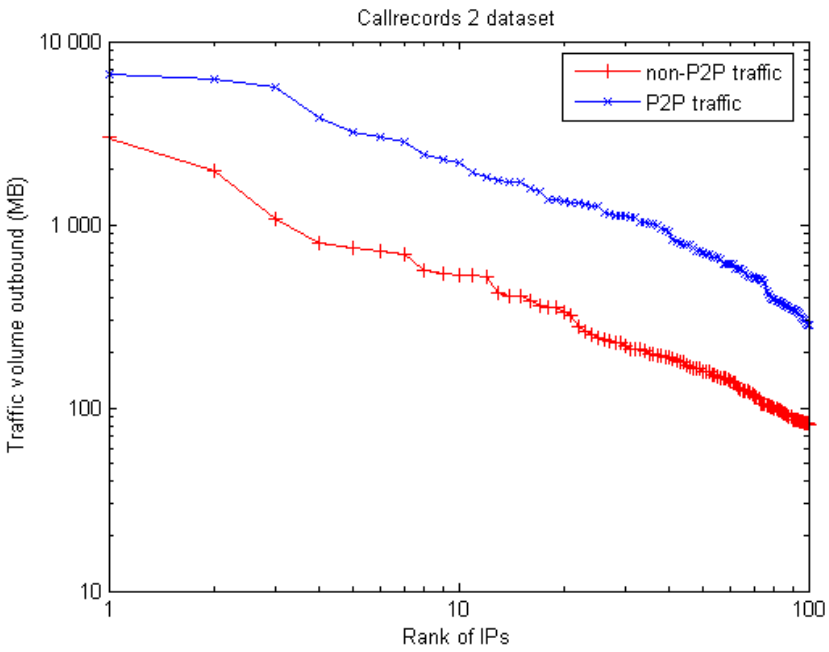

Fig. 54. Traffic vs. top ranked IPs (Callrecords 2 dataset)

Similar analysis was carried out for the number of connections initiated by the users and the results were also analogous. Going from right to left on the rank curve, a fast decrease was observed in the case of P2P traffic. Whereas the drop was much lower in the case of non-P2P traffic. This suggests that on average a normal non-P2P user creates more and probably smaller connections than P2P users, even if P2P traffic was dominating in all measurements in both the volume and the number of connections.

8.5.6. Popular Applications

I collected the most popular P2P and non-P2P applications listed in decreasing order of transferred traffic (see Table IX and Table X). In the list of know applications, HTTP is the absolute dominant. Note that HTTP port traffic can also include some streaming flows.

In the P2P case, only those applications appear in the list, which have known (default) communication ports. From a random or non-default communication port it is not possible to deduce the real application behind the port. Among P2P applications Direct Connect seems to be the most popular.

There is no strong correlation between the amount of transferred traffic and the number of traffic flows. We can observe for example that the number of MSN Messenger flows is high; however, the amount of transferred data is low.

Table IX. Top list of known popular applications

Application	Traffic (MB)	# of flows
1. HTTP (including TCP 80 streaming)	127 449	2 208 968
2. FTP	30 744	7 459
3. POP3 + Secure POP3	3 891	117 820
4. HTTPs	1 578	46 333
5. Streaming (known port)	1 572	1 882
6. SMTP	1 338	65 473
7. SSH	267	1 293
8. MSN Messenger	221	24 345
9. IMAP	212	3 321

Table X. Top list of known-port P2P applications

P2P application	Traffic (MB)	# of flows
1. Direct Connect	6 184	87 368
2. Gnutella	4 746	151 357
3. BitTorrent	4 432	99 942
4. eDonkey	3 778	295 598
5. Napster, File navigator, WinMX	1 053	13 839

8.6. Discussion: the Workload of P2P Traffic Aggregation

The presented analysis may not be a complete comparative characterization of P2P and non-P2P traffic, but the attained results have highlighted some critical findings. P2P users/applications, by the typical content-sharing objectives of P2P usage, behave in a different way than other Internet applications. The difference manifests itself in the almost stable P2P activities over busy and non-busy time periods, the bandwidth-hungry nature, the skewness in the traffic volume distribution between P2P users, etc. However, the characteristics of P2P traffic aggregation, which would be a more important aspect from the service providers' and network operators' point of view, are quite similar to those of other traffic aggregation. While in the beginning P2P applications were confined to greedy file-sharing, nowadays they have grown up to be an inseparable component of the Internet due to several refined developments of P2P protocols. It has been found that there is always a certain ratio of home users who use some P2P applications. The study establishes that the workload of P2P applications generates similar (heavy-tailed) flow size and flow holding time distribution like several non-P2P applications. As a consequence, the P2P aggregation also shows the similar characteristics.

There may come the time when we should change the way of thinking about and treating P2P traffic. It is not an extraordinary component, but an inseparable part of the overall Internet traffic.

8.7. Conclusion

In this chapter, I first presented a novel P2P traffic identification method. The method collects a set of rules derived from the general behavior of P2P traffic. The method does not rely on any payload information, hence it is easy to implement and to use when payload cannot be acquired because of legal or privacy obstacles or cannot be measured due to technical or economical problems. The validation results show that the proposed algorithm is able to identify P2P traffic very efficiently. The method was used to identify P2P traffic in measurements taken in an ADSL domain of Hungary's largest Internet provider.

I also presented a comprehensive traffic analysis study focusing on the most important characteristics like the behavior of active users, the ratio between the P2P users and the total number of users, flow size and holding time distributions and the popularity distribution. The daily profile of P2P traffic intensity was found to be less variable than the daily fluctuation of non-P2P traffic and shows a robust P2P user existence.

It was shown that packet-level statistics of P2P and non-P2P data flows are basically similar. However, there are some applications generating data packets with typical size. I investigated the relationship between packet sizes and applications resulting in a list of typical applications belonging to various packet sizes.

The analysis of the number of active users and total users revealed an almost linear relation. It suggests a very interesting and important result from a traffic dimensioning point of view: the ratio of active users and total users is almost constant (between 0.2 and 0.3 in the current case). The study on flow sizes and holding times confirms earlier results showing the heavy-tailed behavior of both characteristics. I have also found that Zipf's law holds for P2P traffic popularity.

One of the major conclusions is that in spite of the different characteristics of individual P2P traffic the main characteristics of P2P aggregation (and this is important from the point of view of a service provider or a network operator!) do not differ significantly from the characteristics of other Internet traffic aggregation.

9. Identification of Skype Traffic

Skype applies strong encryption to provide secure communication inside the whole Skype network. It also uses several techniques to conceal the traffic and the protocol. As a consequence, traditional port based or payload based identification of Skype traffic cannot be applied. In this chapter, after an overview of the Skype P2P system, network entities and operation, I introduce novel algorithms to detect several types of communications (including voice calls primarily) that the Skype client initiates towards dedicated servers of the Skype network and other peers.

The common point in these algorithms is that all of them are *based on packet headers only* and the extracted flow level information. Information from packet payloads is not needed. The identification methods allow us to discover logged on Skype *users and their voice calls*. The whole identification process is scripted in Transact-SQL; therefore it can be executed automatically on a prerecorded (offline) dataset. I present identification results, analysis and comparison of datasets captured in *mobile and fixed network*. I also present the validation of the algorithms in both network types. The proposed techniques and properties can be applied for the identification of other Internet applications as well.

The Skype network is a world-wide P2P VoIP network. The users can initiate and receive voice calls to/from other Skype users, or even PSTN users using SkypeOut/SkypeIn. Moreover, instant messaging (chat), video conference (even in high quality) and file transfer is also supported within the Skype infrastructure. Skype also offers a public API (application programming interface), which can be used by third party applications for communication over the Skype infrastructure.

The Skype traffic is considered from a network operator's point of view. The operators are usually interested in the nature of the traffic carried by their network in order to optimize network performance, forecast future needs and also for marketing purposes by identifying and studying popular applications and services.

Skype usage is especially of great interest for mobile service operators. By the increasing number of smart phones, more and more users have the option to choose between traditional phone calls and the rapidly spreading VoIP services. Operators usually apply a flat-rate pricing in case of data transfer, which makes Internet-based telephony even more cost efficient. It is indispensable for network operators to know how many users use VoIP applications (e.g. Skype as the most widespread one) and how much they talk. This way they can make a tradeoff between per minute based pricing of traditional calls and per megabyte based (or even flat rate) pricing of VoIP calls. In the current situation this balancing is crucial for an operator.

As a first step of the analysis, the Skype traffic should be identified. The identification is not a simple task, since there is no unique standard port for Skype traffic, the protocol is not public, the data is encrypted, and different software versions behave differently. Moreover, the Skype binary uses a variety of techniques to prevent reverse engineering [178].

My goal was to detect Skype traffic even if the Skype client was started before the traffic measurement, in which case we cannot rely on some typical login traffic patterns or payload information. I also decided to detect all Skype traffic regardless of software version and to avoid the use of payload information, which is in many cases not available. I constructed the identification method to be based on properties and time behavior of data flows and packets.

In the following subsections, I will give a brief overview of Skype components and operations focusing on those aspects that can be used in my detection algorithms. A more detailed description can be found for instance in [179] and [180].

Section 9.1 gives an overview of related work. In Section 9.2 and 9.3 the main Skype components and operations are described. Section 9.4 and 9.5 introduce the identification methods, which are verified in Section 9.6. Finally section 9.7 gives detailed traffic analysis of Skype traffic in fixed and mobile networks.

9.1. Related Work

The identification of Skype traffic is addressed in several recent publications. Ehlert et al. [182] describe a method for identification using traffic patterns and payload information from the Skype login phase. For efficiently blocking Skype, its activity has to be identified. Many blocking methods were proposed (see e.g. [183]), but these are usually tailored to a given firewall type or security setting, and assume that Skype communication starts after enabling the blocking mechanism.

Suh et al. [184] present a method for the detection of relayed traffic by comparing input and output traffic patterns. The false positive rate (and the false negative) of this technique is reported to be between 4% and 8%.

Guha et al. [185] present some results about Skype usage patterns. Their results are based on active measurements and provide global information about the number of clients, the number of super nodes and general traffic patterns of a single Skype session.

Kuan-Ta Chen et al. [186] describe an identification method for *relayed Skype flows*. Some of the characteristic flow properties they examine to select Skype voice sessions are similar to ours. They aim to detect

relayed flows only, and use the collected data for investigating the correlation between call duration and voice quality.

The authors of [187] proposed three different techniques to identify Skype voice traffic. The first approach – based on Pearson's Chi-Square test – probes whether payload bytes are truly random or not, a condition that has to be true for encrypted traffic. Their second approach investigates stochastic characteristic properties of Skype traffic, namely packet arrival rates and packet sizes, using Naïve Bayesian Classifier (NBC). These properties are considered in my identification method, but I also involved other characteristics as well (e.g., bandwidth, main mode of inter-arrival time and average packet size). The authors found that by combining these two techniques they can efficiently identify Skype voice traffic. Depending on the dataset, the ratio of false positive detection is indeed low (1-2%), but the ratio of false negative is high (12-29%). A third technique was also proposed based on deep packet inspection. However, this technique did not prove to be reliable by itself. It was paired with a companion algorithm, which looks for candidate Skype host IP and communication port. The process of looking for candidate Skype host is also an important part of my identification method. However, in my case the searching for candidate host is based on a completely different algorithm.

Paper [188] focused on several aspects of Skype source, including service types, transport protocol and network conditions. They analyzed three major characteristic properties: bit rate, IPG (inter-packet gap), message size. These properties are also taken into account in my study as filtering criteria. The results of my analysis are also consistent with their observations for codec properties. However, I concentrated mainly on ISAC codec as the prevailing codec used in the traffic measurements. The authors also present a traffic analysis of a dataset captured in a campus network applying one of the proposed identification techniques (the NBC based one) in [187].

The same authors investigated Skype signaling traffic by means of passive measurement in [189]. They classified Skype signaling traffic into three different categories: probe traffic (to discover new nodes), periodic heartbeat flows and long dialog flows. Heartbeat flows are called "UDP relations" in my notation (see Section 9.4.4) and have particular importance in my identification process. An analysis of signaling flow categories in terms of number of generated flows, packets and affected foreign peers is also presented in [189].

In [190] the authors provide payload patterns that appear with high frequency in the recorded dataset generated by a SN and a client. They state that using these signatures Skype traffic can be effectively identified and monitored. However, there is no report on the accuracy of the method. The paper also aims to investigate the structure of the Skype overlay network, including the number and location of SNs.

Freire et al. [191] presented a method for distinguishing web traffic and Skype traffic passing through TCP port 80. Skype may use this spare port if the UDP transfer and default communication port is blocked. They calculated the distribution of certain characteristic parameters (excluding packet payload) of Skype and web traffic using a training dataset, and classified flows by Chi-square and Kolmogorov-Smirnov tests.

In [192] the authors suggest a method for identifying BitTorrent like file sharing applications based on the "discreteness of remote hosts", i.e., how many hosts a client contacts from various stub networks. They state that by using this representative measure they can differentiate BT-like applications and traditional P2P applications (including Skype).

Wang et al. [193] aimed to identify Skype sessions in order to guarantee the bandwidth and assure the quality of service for Skype traffic. They applied the simple K-means method to form clusters and classify flows. However, it is not clear what properties they have chosen to form the feature set; also, the test dataset contained very few types of traffic in addition to Skype, which makes clustering easier. They reported an accuracy of 95% and 90% in "normal" and complex environment, respectively.

Some commercial products [194] are available for hindering the usage of Skype. These applications prevent the initial connection to the Skype network by seeking for some pattern in the packet payload. The employed patterns are kept in secret. In addition, *no data is published about the accuracy of the solutions*. Skype blocking was also introduced in IOS version 12.4 (4) released by Cisco in 2006 [196]. Skype can be blocked by a simple policy based on packet filtering, but the details of the algorithms are unknown. Some vendors (e.g. [195]) claim that their products are able to identify Skype, but verifications are not made public.

Traffic patterns are presented in [197] for the detection of Skype-to-Skype communication – including both calls and general (idle) chatter of the program – and Skype-to-phone (SkypeOut) connections. However, the reliability of these patterns is questionable. The patterns are reported to match at least some of the general chatter the client produces when the user is idle or performing actual phone calls. Unfortunately, the patterns also match a significant fraction of non-Skype (and even random) traffic. This means that both false positive and false negative identification can occur using the patterns.

The fact that several vendors come out with solutions for blocking Skype may suggest that payload based blocking is feasible. However, these approaches differ from my objective: I do not want to hinder Skype communication, but rather to monitor and analyze certain properties of the traffic (e.g., intensity of calls and users, bandwidth of the voice calls, etc.). Therefore, obstructing Skype traffic completely by blocking the initial

login phase does not serve my needs. However, payload patterns could be used to detect candidate Skype hosts, thus making call identification more reliable.

9.2. Skype Components

The Skype P2P network consists of the following elements: ordinary nodes (clients), super nodes (SNs) and dedicated servers – including login servers, update servers and buddy-list servers (Fig. 55).

An ordinary node is a leaf-node of the Skype overlay network; it is the equipment of the user that is used for communication. SNs are the switching elements in the overlay network, responsible for maintaining a Global Index distributed directory which allows users to find each other. Each SN keeps track of a small number (up to 700-800 according to my measurements) of ordinary nodes. SNs can also function as ordinary nodes, every ordinary node with a public IP address and sufficient resources (free CPU, memory, bandwidth capacity) being a candidate to become a SN.

The login server stores the account information of the users. It is responsible for user authentication at the beginning of the session, i.e., when the client is started. According to my observations, there are several login servers storing information in a distributed way.

The update server (212.72.49.131:80) is also contacted by the client after startup to check whether a newer version of the software is available.

The so-called buddy-list server [181] is responsible for storing the contact list of the users. Although this list is stored locally on the host computer, the role of this entity is to make sure that the contact list is also available if the user logs on from another host. Therefore this server is contacted when changes in the contact list occur, or when the user is logged in from a different host than the previously used one. Known buddy-list servers include 212.72.49.142 and 195.215.8.142 [5].

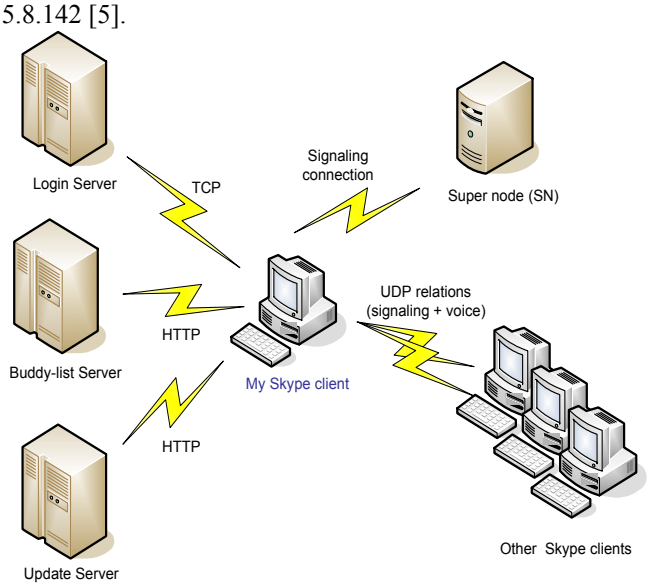

Fig. 55. Elements of the Skype P2P overlay network

9.3. Skype Operation

When a Skype client is launched, it tries to establish a connection with a SN. For the duration of the session this SN will be responsible for the client. The client first contacts several (about 20) SNs, to investigate whether they are alive and ready to accept the client or not. After some message exchange the client selects one of the SNs to establish a connection.

An initial set of SNs (bootstrap SNs) is hard-coded in the executable. These are operated by Skype. Upon the first run of the client these SNs are contacted, but a list of other SNs is retrieved and stored by the client. When the client is started for the next and all subsequent times the stored SNs are used.

At startup, the client also contacts one of the login servers for authentication. Although there are several login servers, I have found that Skype clients in my measurements always connected to one of the following two servers: 212.72.49.141 or 195.215.8.141. The connection to the login server might be relayed by a SN; this usually happens when the direct connection is blocked (e.g. by a firewall), but it can occur even if there is no such problem.

When the user is on-line, there is a periodic message exchange between the host of the user and the selected SN.

9.4. Skype Identification

In spite of the fact that the application-layer protocol of Skype is concealed, it is still possible to monitor the network and transport layer protocols and analyze the used IP addresses and ports. The statistical characteristics of the Skype data flows and packets can be studied as well, including flow bandwidths, packet sizes, and several other properties. The proposed identification method is based on these observable open parts of the Skype communication.

The difficulty is that the regular check for software updates does not guarantee that every client runs the latest version of Skype. Therefore, it is necessary to deal with the behavior of different versions of clients. Although I did not have a chance to analyze each client, the identification algorithm is based on those properties which seem to be invariant amongst different software versions. In this section, I first propose methods to detect Skype activity even if no calls are made. Then, I present a decision based method for the detection of Skype voice calls.

The decision based call identification algorithm contains basically two steps: discovering candidate Skype hosts first, and then searching for voice calls. The success of the call identification in the second step depends greatly on the reliability of host identification in the first step. There are three techniques to determine the candidate Skype hosts (described in details in the following subsections):

- Looking for *Skype specific connection* (Section 9.4.2) - this is problematic because these connections may be relayed by the SN, or may not be present at all;
- Looking for *"signaling flow"* (Section 9.4.3) between the client and the SN;
- Looking for *"UDP relations"* (Section 9.4.5).

The first two techniques permit only to determine the IP address of the host, while the third one makes it possible to detect the used communication port too. The second technique allows the detection of idle Skype presence as well (i.e., an idle client who does not initiate calls), not only single events. The UDP relation based candidate Skype host identification proved to be the most successful and reliable among the three techniques. The only disadvantage of it is that it does not work with a client for which UDP communication is completely restricted. Thus, it has to be considered whether this situation is likely to occur in the network environment where the measurements were taken. In the current case, home ADSL users and mobile users are considered, where UDP communication is likely allowed. Therefore, the UDP based method was chosen to detect candidate Skype hosts. The overall process of the identification and the roles of the methods are depicted in Fig. 56.

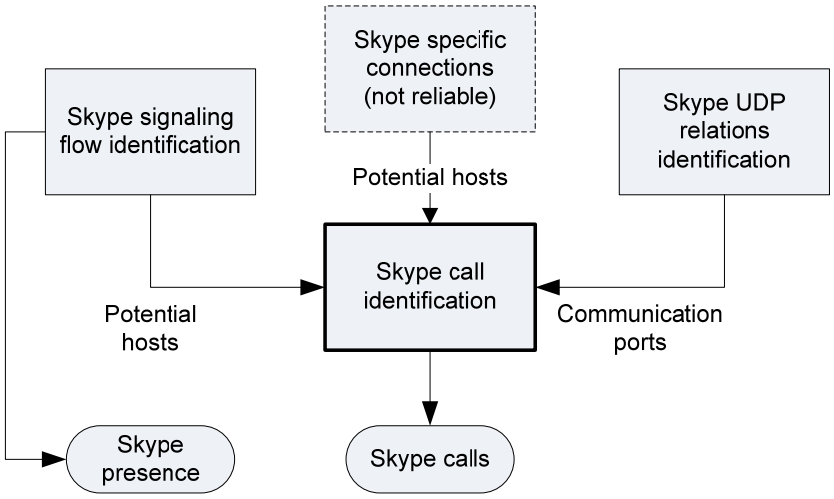

Fig. 56. The overall process of Skype traffic identification and the role of the methods

9.4.1. Filtering out Known Applications

The preliminary step of the identification is to filter out traffic of known, non-Skype applications. To do so, a database of popular applications was used, which contains default TCP and UDP communication ports for known applications (See Table XI for example). TCP ports 80 and 443 are considered as an exception, since besides HTTP they are also used by Skype.

Table XI. Example: a list of name, default port and protocol type of common applications

Application	Port	Protocol
MSN messenger	1863	TCP
NETBIOS	135, 137, 139, 445	UDP
DNS	53	UDP
NTP – Network Time Protocol	123	UDP
POP3	110	TCP
SMTP	25	TCP
FTP	20, 21	TCP
RDP – Remote Desktop Protocol	3389	TCP
SSH	22	TCP
POP3	995	TCP/UDP
POP3	110	UDP
IMAP	143, 993	TCP/UDP
…	…	…

9.4.2. Skype Specific Connections

The first possible way to identify candidate Skype hosts is to look for Skype-specific connections, such as the connection to a login server, buddy-list server, or bootstrap SN. The occurrence of any of these infers the presence of Skype, since these connections are very unlikely to be initiated by non-Skype applications. On the other hand, these connections are not necessarily present (or visible) during a Skype communication:

- The *connection to the login server* is in some cases relayed through a SN and therefore invisible;
- The *connection to one (or more) of the bootstrap SNs* is necessarily attempted at the first execution of the application, but during subsequent executions the host may choose to contact other SNs;
- The *buddy-list* server is contacted only in the cases mentioned in Section II.

There are also some kinds of connections which are not unique, but characteristic to Skype. I found two of these:

- The connection to the *update server* is initiated at the beginning of the session. However, the same IP and port is used, in some cases, to reach the www.skype.com web server;
- A TCP connection to *port 33033* is likely to be initiated by a Skype client, since this is the default port for SN connections.

It is very unlikely that both of these connections are present if the host does not run a Skype client; therefore, I decided to conclude on Skype presence if both of these are found.

9.4.3. Skype Signaling Flow Identification

It is possible that the user logged on to Skype before the traffic measurement began. In such a case, login, buddy-list update, or software update attempts cannot be detected. Furthermore, even if these connections can be observed, they give no indication of the end time of the session. Therefore, I investigated Skype network activity to find characteristic traffic patterns that last for the entire duration of the session.

A Skype client maintains one permanent connection to a SN while the user is logged on to the Skype network. At startup the client tries to establish several outbound connections to find an appropriate SN. After a few seconds these transient connections are terminated and only one or two permanent TCP connections remain. One of these connections has a traffic pattern which can be relatively easily identified. Data packets are limited in size and both inbound and outbound flows (belonging to the specific TCP connection) have restricted bandwidth and packet intensity. In addition, the timing of outgoing packets follows a well-defined pattern. The connection persists as long as the user is logged on to Skype. For these reasons, I selected this flow to be scanned in order to identify Skype clients and constructed a method to identify such flows.

In some cases, I observed that the original signaling connection was replaced by a new one with exactly the same properties. This was possibly caused by the original SN going offline because of a failure or other problem. Nevertheless, the signaling flows are generally long enough to be detected. The long duration of the signaling flow is a key issue, since the proposed algorithm utilizes statistical properties among others.

Based on a widespread analysis of Skype signaling connections, I decided to consider a TCP connection as a Skype signaling connection if it obeys all of the following rules:

1. the outbound and inbound data rate is not larger than 40 bytes/sec;
2. the number of packets (in outbound and inbound direction separately) is not larger than 0.4 packets/sec;
3. every packet in outbound direction is smaller than 1000 bytes (including IP and TCP headers);
4. a periodicity of 1 minute is observable for outbound packets of size between 70 and 250 bytes. E.g., a certain percentage of such packets arrive in a specific, periodic time slot.

The unique time behavior of signaling flows is caught by the 4th rule. I realized that most of the outgoing data packets are rather small, except for some special packets. The exceptional packets have much larger packet size (between 70 and 250 bytes, including IP and TCP headers), and an inter-arrival time of one minute. I assume that these packets are some periodic keep-alive messages of the client to the SN.

Unfortunately, the picture is not that clear, as some factors make the identification more difficult. E.g., occasionally out-of-period packets appear in the outbound flows, a few of them falling into the interval of 70-250 bytes. Sometimes keep-alive messages happen to drop out, though periodicity is still preserved (no shifting). In some cases (perhaps when the user has few contacts on his/her buddy-list), the size of the keep-alive messages is not significantly larger than that of other packets in the outbound flow, which results in an unclear separation of keep-alive packets. However, when the user initiates a voice call (i.e., it becomes active), the size of keep-alive messages increases and falls into the specified interval in all investigated cases. After finishing the voice call, the size of keep-alive packets usually returns to its original value.

A simple method is chosen to detect periodicity in Rule 4. The modulo 60 remainder is calculated for the arrival time (measured in seconds) of every packet (with packet size between 70 and 250 B), which is then rounded to the closest integer. This yields a time slot of 1 second. Then the distribution (histogram) of modulo 60 remainders (see Fig. 57) is calculated, and every remainder is marked where the frequency of the remainder in the histogram is over a certain threshold (0.08). The connections considered as Skype signaling are the ones for which only one remainder is over the threshold, i.e. only one remainder is significant. I chose this technique to detect periodicity because it is adequate, fast, simple and easy to implement in SQL. However, other techniques could also be applied (e.g., Discrete Fourier Transformation (DFT) or wavelet transformation).

A minor shifting (few ten milliseconds) was observed in the arrival times of the periodic keep-alive packets, which was most likely due to the inaccuracy of the internal clock of the host computer. This can cause two significant modulo 60 remainders (next to each other) to appear in the histogram. This problem can be avoided by shifting the arrival process by 0.5 second (adding 0.5 to each arrival time), and calculating the above mentioned histogram for this new process. If the periodicity is present, then either the original or the shifted process will reveal it. The shifting of the arrival process was applied for all flows, but only a few new signaling flows could be found with this technique.

Although in theory other applications might generate similar traffic patterns, in the measurements no such was found. Therefore, if such a pattern is found, it is attributed to Skype.

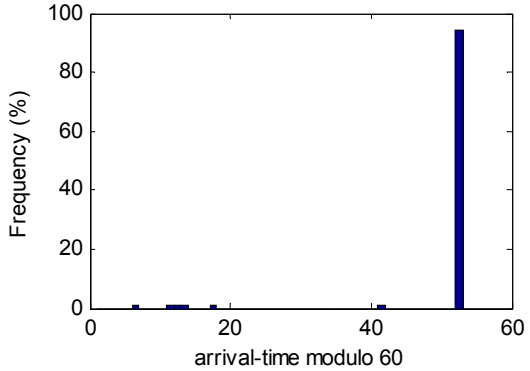

Fig. 57. Modulo 60 remainders of arrival times for packets of size between 70 and 250 bytes

9.4.4. Communication between Skype Clients

The Skype client communicates not only with super nodes and dedicated servers of the Skype infrastructure, but it usually also maintains direct relation with several other Skype clients, including mainly the logged on contacts on the buddy-list of the user, but other unknown clients as well. The relation cannot be considered as a real connection, since it consists of UDP packets only. In most of the cases, the packets are originated from the default communication port of the Skype client. On the other hand, sometimes random UDP ports are used. However, in these cases the communication is terminated in the default port of the Skype client of the other party. Usually the default ports are used on both sides.

After the client logs on to the Skype network, the UDP relations are soon built up with most of the contacts on the buddy-list of the user, who are also logged on to the Skype network at that time. The UDP relation is usually already established when the user initiates a call towards one of the partners on the contact list, and remains active permanently after the call is finished. The main purpose of UDP relations is to constantly monitor the connectivity between the two sides. The client checks whether the other party is present and reachable. It also examines whether UDP communication is available or not. Hence, the packets of the UDP relation can be considered as "UDP ping messages" between the clients.

Note that if a call is indeed initiated between the two sides, the same UDP relation is used with the same communication ports and only the intensity and size of packets changes significantly. Therefore, the early preparation of a call is also a function of an UDP relation. The clients can build up the connection before the call is initiated, which speeds up the establishment of the call. However, the UDP relation is always built up if UDP communication is not restricted – by a firewall, NAT, etc. – irrespectively of whether a call is initiated later or not. The clients can also earlier recognize if UDP communication is restricted and try TCP or relayed connection.

We can observe that if we initiate a voice call shortly after logging on to Skype, the call is built up slower. As opposed to this, if we begin the call after a few minutes, the other party is connected immediately and it starts ringing. The reason for this is that in the second case the UDP relation is already built up, which makes call connection time shorter.

If a UDP relation is not transferring a call, it has well defined characteristics, which makes it possible to construct a robust identification method for UDP relations. If several UDP relations are found for a certain Skype client, we can reliably determine the Skype communication ports used by that client.

It is clear from the above description that the separation of call sessions and inactive periods is not trivial within a UDP relation. In the flow level traffic information, a UDP relation appears as a single UDP connection. Thus it is necessary to accurately determine the beginning and the end of a call (or several consecutive calls) within the UDP connection. This can be performed by using the related packet level database. Fortunately, speech packets and "UDP ping packets" have distinct sizes and intensities, which facilitate the identification of call sessions and inactive periods.

9.4.5. Identification of UDP Relations

A UDP relation has well defined and distinct characteristics if it conducting a voice call and if it is idle. The two states can be separated based on the size of packets. According to my measurements, in case of a voice call the average voice packet size varies from 70 bytes to as high as 320 bytes. On the other hand, when the relation is idle the size of packets (UDP pings) is always lower than 60 bytes.

According to my comprehensive analysis, Skype UDP relations can be detected by a simple identification method [198] that consists of three steps:

1. Select each UDP flow which has more than 10 packets and the source or destination port do not belong to a well-known application;
2. For the remaining flows calculate main mode of the inter-arrival times of data packets smaller than 60 bytes;
3. The flow is likely a UDP relation if the main mode equals to 20 seconds.

The first rule is applied in order to get rid of flows which can be unambiguously identified as not being signaling flows; this reduces the computational time needed to verify the second rule. According to the first rule, all flows are discarded which do not contain enough packets to be a UDP relation, or have a source or destination port of a well-known application (typically DNS queries and responses).

UDP relations have a specific time behavior: packet arrivals show a certain periodicity. For this reason, the inter-arrival times of UDP ping messages were found to be the most characteristic property of UDP relations – in addition to packet size, which was applied as a filter in the previous step. The whole identification process of UDP relations is depicted in Fig. 58.

A Skype client establishes several UDP relations. The number of such relations depends on the number of logged on users on the buddy-list of the user. In many cases, however, I observed more UDP relations than the number of users on the list, which suggests that UDP relations are established with foreign Skype peers as well. This behavior, fortunately, only helps identification.

UDP ping messages have a specific inter-arrival time, which is generally equal to 20 seconds on average. To avoid the error resulting from the deviation of the inter-arrival time, the histogram of the inter-arrival time is calculated, and the main mode of this histogram is selected. The main mode is defined as the most frequent item of the histogram.

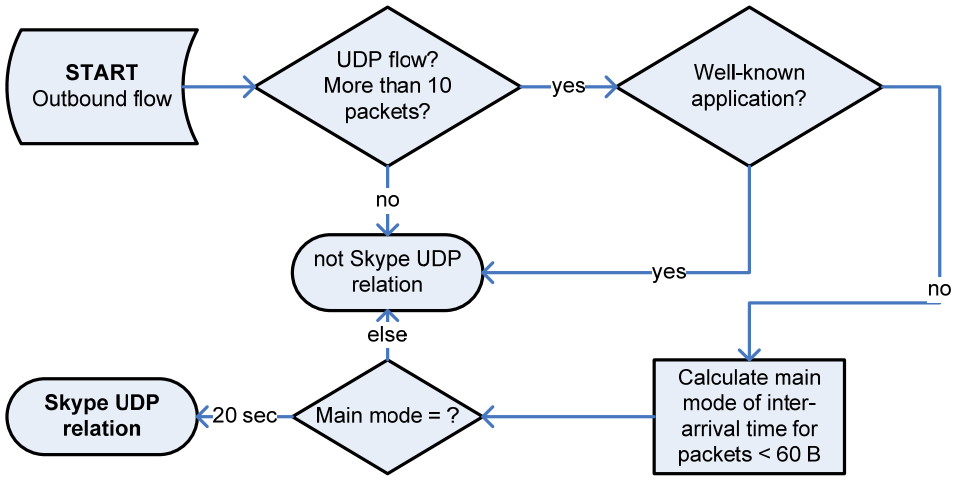

Fig. 58. Detection of Skype UDP relations between Skype hosts

Although the main mode of the inter-arrival time sometimes differs from the value of 20 seconds, at least a few UDP relations could be detected in all examined Skype clients with a main mode of 20 seconds.

From a detected UDP relation the IP address of the Skype host and the default communication port of the Skype client can be determined, especially if numerous UDP relations are discovered for a certain IP address and port pair.

9.5. Decision Based Identification of Calls

In the previous section, Skype UDP relations were identified, resulting in a list of IP address and communication port pairs. The IP address is the network address of the host computer, while the port is the randomly selected communication port of the Skype client, which is normally used for both UDP and TCP protocol based voice calls.

All flows originated from one of the IP-port pairs are likely generated by Skype. Some of these flows are the UDP relations already identified. Other flows may contain a few UDP relations which could not be detected and other types of Skype communication as well. Flows conveying voice calls are also originated from one of the IP-port pairs.

Most of the calls are based on the UDP protocol and are embedded in a UDP relation as described in Section 9.4.4 – i.e., there are one or more sections in the UDP flow (UDP relation) when the relation is not idle but conveying a voice call.

A small part of calls are transmitted over TCP. It is also possible that voice in one direction is sent over TCP and the reverse direction is served by UDP. This is a lucky situation, since at least one direction of the call can be detected based on the list of IP-port pairs of UDP relations.

However, if UDP communication is completely restricted by a firewall or a NAT device, the identification of calls becomes very problematic. In such a situation, no UDP relations are present, and thus the default communication port of the Skype client cannot be determined. Voice calls are transmitted over TCP in both directions.

Whatever transport protocol is assumed, it is necessary to clearly determine the beginning and the end of the calls. Calls rarely begin and end at the same time when the corresponding UDP or TCP connection starts or finishes. This problem is obvious in the case of UDP calls. However, it also applies for TCP connections. In addition, finding the section(s) within a TCP or UDP connection where a call exists indeed is also required to accurately calculate some characteristic properties of the call (e.g. holding time, bandwidth, packet rate, etc.). These properties are needed for the identification of Skype calls.

9.5.1. Communication Protocols

Skype prefers UDP as the primary transport protocol, and switches to TCP when UDP communication is restricted. It adapts quickly to changing network conditions by switching voice codec and transport protocol even in the middle of a call. Several scenarios are possible for establishing a voice call:

1. UDP protocol is used in both directions;
2. UDP is used in one direction, TCP in the other;
3. TCP is used in both directions;
4. Switching communication protocol in one or both directions from UDP to TCP at the middle of the call.

The first and the second case are covered by the robust identification method based on UDP relations. The third case is quite problematic, since it is possibly the consequence of the complete blocking of UDP communications. In this case, the identification based on UDP relations does not work, and the usage of my previous identification method introduced in [199] is preferred. However, this method is somewhat less reliable, since the communication port of the client cannot be identified, only the source IP address. These TCP based calls are very rare and these calls do not modify the results and the statistics significantly. The fourth scenario happens when a dramatic change occurs in the network conditions. In most cases, the client usually adjusts codec parameters only. However, if the UDP protocol is no longer available, the client switches to TCP. Nevertheless, this behavior could be proven in my simulations only and I suppose that this case almost never happens in an average actual use.

9.5.2. Call Properties

The final decision on whether a flow (or a flow section more precisely) is a Skype call or not is based on the following characteristics of the section:

- bandwidth (total number of transmitted bytes divided by the holding time of the call);
- packet rate (total number of transmitted packets per holding time);
- average packet size;
- main mode of inter-arrival time of voice packets.

ISAC and iLBC codecs are used no matter which transport protocol (TCP or UDP) is employed. Both codecs adapt their transfer rate and packet size to the available link capacity. Consequently, only a lower and an upper threshold can be set up as preliminary filter conditions for voice flows. According to my measurements, the average voice packet size varies from 40 bytes to as high as 320 bytes, while a voice flow (in one direction) has a bandwidth between 20 Kbps to 80 Kbps. Therefore a loose upper bound of 400 bytes for packet size and 100 Kbit/sec for flow bandwidth is defined. Flows failing to match any of these criteria are discarded.

In order to discover real Skype flows, I wanted to find some more characteristic properties. Skype codecs have basically constant bit rate, even if the parameters of the codec, like packet size, bit rate, inter-arrival time, might be dynamically modified as a reaction to high delay, jitter, or packet loss. The inter-arrival time of voice packets was either 30 ms or 60 ms in all measurements, which results in a packet rate of 33 or 16 packets per second, respectively. In case of a TCP connection and obsolete Skype clients (Linux versions), an inter-arrival time of 20 ms (50 packets per second) was also detected. Finally, I discarded this latter packet rate, since it would have induced too many false positive errors, and would have provided only a few (if any) right hits. These values were confirmed by several other studies [181] [200] as well.

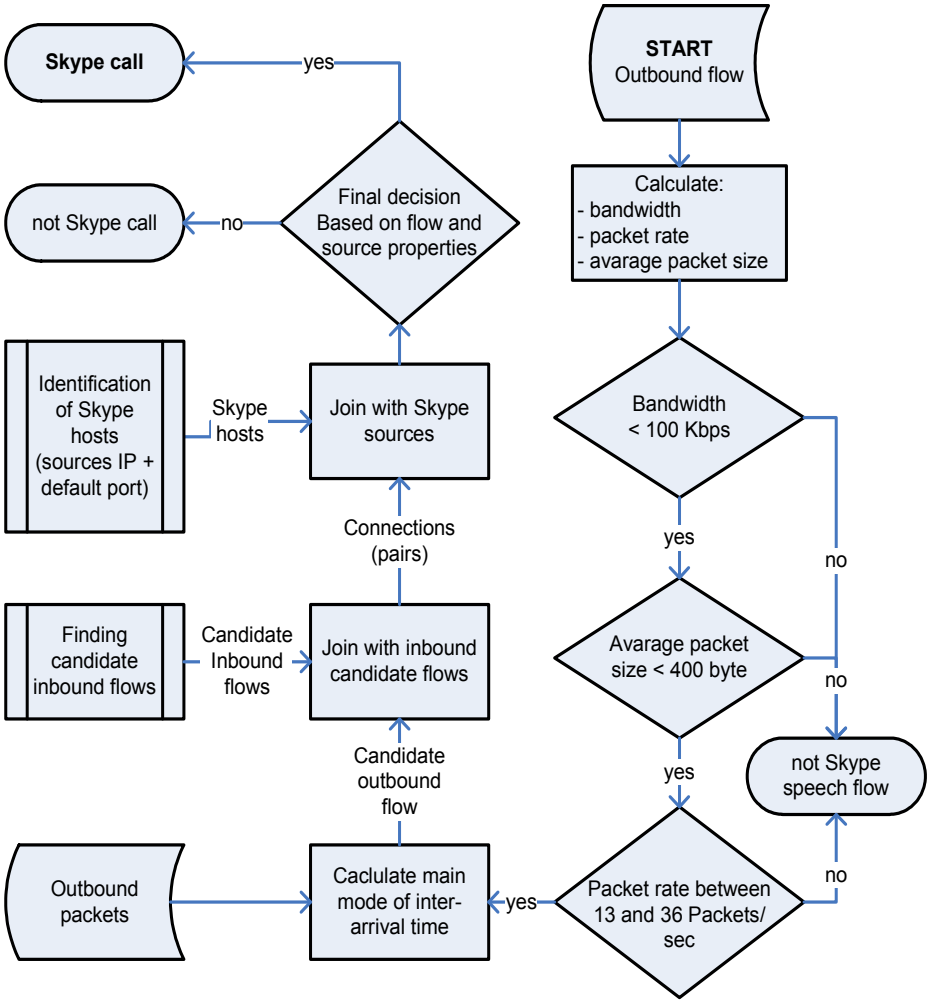

Fig. 59. Identification process of Skype calls

Fortunately, the packet rate can be calculated and checked at flow level knowing the arrival time, end time, and the number of packets in the flow. Thus flows which do not correspond to this condition can be discarded. However, it cannot be expected that packet rate will be exactly 33 or 16 for all Skype flows, which makes identification of speech flows somewhat problematic. The main reason is that there is some transient behavior at the beginning of the session when the bandwidth, packet rate, and packet-size differ significantly from the properties in steady state. The used codec and the occupied bandwidth might also change during a call when the changes in network conditions require so. Apart from these, packet rate is still a suitable property to decrease the number of candidate speech flows. A rate of 13 packets/sec and 36 packets/sec are chosen as a lower and upper bounds, respectively. Flows not corresponding to the packet rate condition are discarded.

The efficiency of the identification can be improved by using not only flow-level properties, but including some packet-level characteristics as well. I found the inter-arrival time as the most characteristic property. For each remaining flow, the histogram of inter-arrival times is calculated, and the main mode of the histogram is noted. Afterwards, inbound and outbound flows are paired to one another to create voice sessions. The terms of pairing are the following: arrival time and end time of inbound and outbound flows are required to be close to one another, and also source address, source port, destination address and destination port should correspond to each other.

If the inbound and outbound directions of a voice session are served by different TCP connections, the similarity of source and destination ports is not required.

In the last step, those connections are selected where the main mode of both inbound and outbound flows has a value of 20 or 30 (ms) and source IP-port pair is among the previously identified Skype client IP-port pairs. The whole identification process is shown in Fig. 59.

It is possible that some non-Skype flows meet some of the conditions. However, it is unlikely that flows other than Skype (even flows generated by other VoIP applications) meet all the conditions. In addition, the list of Skype sources (together with source ports), which was identified in a previous step, is also used to avoid misdetection.

9.6. Validation of the Identification Method

The validation of the identification algorithm raises a couple of questions. For an exhaustive validation of the methods, a large number of verified Skype signaling and voice flows is needed from several clients. It is not easy to build such a managed environment.

The purpose of this validation is to verify the parameters of the identification method. These parameters were determined based on several local measurements on single computers in different types of network environments (e.g., LAN access, ADSL, dial-in access, etc).

An experimental traffic measurement (see Section 7 for details) was carried out at my university department. The measurement contained the traffic of about 110 distinct users and a large variety of applications, including other VoIP applications as well. After the logging has finished, all colleagues were interviewed about whether a Skype client was running on their computer and whether they have made any calls during the logging period or not. In addition, all the history logs of the clients were collected, which contain exact information of the calls, e.g., date, time, and call duration.

Afterward the identification method was applied to the experimental dataset to detect Skype hosts and Skype calls. Based on the comparison of detected Skype hosts and known Skype hosts from the user feedback, both host and voice call identification methods work well. Especially the UDP relation based method gives accurate results. Signaling flows were also detected in most of the cases. Update connections sometimes, login-, buddy-list- and SN connections were rarely identified. Table XII shows the results of the validation measurement at the university department. The last two columns display the number of detected UDP relations and signaling flows, respectively. The three unmarked rows were not confirmed to be Skype hosts; however two of these were DHCP hosts.

All Skype calls extracted from history logs were detected as well. No false positive or false negative was experienced. False positive means that a non-Skype flow is mistakenly identified as Skype, while false negative means that a real Skype flow is not detected.

The validation study, however, cannot be considered as an exhaustive verification of the identification methods, since all Skype voice calls were made in an ideal network environment (100 Mbit Ethernet). As a result, always the best-quality Skype codec was assumed to be used by the clients.

Table XII. Identification of default communication ports at my university department

Host name	IP address	Port	Number of detected UDP relations	Signaling flow
bme-diga	152.66.244.228	25224	266	2
pubi	152.66.244.95	2061	121	1
zed	152.66.244.226	50952	80	2
horus	152.66.244.3	21817	67	7
dzsessz	152.66.244.14	3540	66	no
dhcp76	152.66.244.76	60989	41	24
nirvana	152.66.244.222	65363	21	2
dhcp78	152.66.244.78	28474	16	1
dhcp79	152.66.244.79	7173	14	2
akkord	152.66.244.120	9623	12	1
nokya	152.66.244.2	23169	10	no
dhcp211	152.66.244.211	27661	10	no
core	152.66.244.5	18269	8	2
asus3	152.66.244.224	55635	4	3
dhcp76	152.66.244.76	61099	4	24
hegyi	152.66.244.199	50492	2	69
dhcp206	152.66.244.206	7173	2	no
dzsessz	152.66.244.14	34584	1	no
tico	152.66.244.221	50492	1	no
bme-diga	152.66.244.228	4752	1	2
bme-diga	152.66.244.228	1033	1	2
sb	152.66.244.65	5588	1	no

Table XIII. A comparison of characteristic properties for different VoIP applications

	Gtalk	MSN Messenger	Yahoo Messenger	AOL Messenger	Skype
Average bandwidth (Kbps)	113	60	112.1	321.4	40
Average packet size (byte)	166.2	94.5	166.7	165.1	150
Packet rate (1/sec)	25.56	50.3	24.03	31.83	32.61
Packets inter-arrival time (sec)	0.038	0.020	0.041	0.033	0.031

In addition, I also created a comparison table of commonly used VoIP applications. Table XIII shows the characteristic flow and packet level properties of the codecs used in these applications. As Table XIII confirms, the characteristic properties of Skype differ from the properties of other VoIP applications. Therefore, it is very unlikely that my algorithm will mistakenly identify other VoIP flows as Skype.

Apart from the validation in the fixed network, the validation of the identification algorithms was also performed in a mobile environment (for details of the measurement see Section 7). Two laptop computers were used with mobile phones connected to them. The phones were connected to the base station via an HSDPA connection (384 Kbps uplink and about 3.6 Mbps downlink bandwidth). The test cell was private, no other phones were thus able to connect to the base station and affect the available bandwidth of the test phones. The set of application generating the traffic was limited to Skype, MSN, Fring, web browsing and other general applications running on Windows.

During the test, Skype calls between the clients and towards remote clients with fixed network connection were initiated. All traffic was captured in the cell and, in addition, a Skype activity log was also conducted which contained information about calls, sign in and sign out events, including exact times, durations, caller and called parties.

The purpose of this active measurement was twofold. Firstly, I wanted to verify my Skype identification method. Secondly, I wanted to investigate how Skype behaves in a mobile environment, especially when the available bandwidth is limited. To simulate this, it was possible to limit the bandwidth available for users in the test cell – even at the middle of an ongoing call. Several test calls have been carried out, applying a bandwidth limitation of 128 and 64 Kbps. I also wanted to inspect weather limited bandwidth affects the accuracy of the identification algorithm.

After capturing all traffic in the test cell, the same identification algorithm was applied as in the case of the real traffic traces. After this, the identification results were compared with the information in the Skype activity log.

The first step of the algorithm proved to be successful (similarly to the case of the fixed network environment). The default communication ports (Table XIV) of both clients were successfully detected with high confidence. In addition, two other ports used by Skype were also detected, which are not the default ones. However, this is not considered a problem here, because these ports are only used to look for voice flows originated from one of these ports. Voice flows are originated from the default communication port in case of UDP transmissions and in most of the cases of TCP transmissions.

Table XIV. Results of the validation measurement performed in the mobile network

Phone IMEI number	Port	Number of detected UDP relations
216301119000054	9942	35
216301119000054	4367	1
216307007161849	12613	43
216307007161849	2993	1

The second – voice call identification – step of the algorithm successfully identified 12 of the 14 Skype calls. The two unidentified calls were among those where the bandwidth was limited. Limitation (or bad network

conditions) may cause interruption in the transmission of voice packets. In the case of a longer interruption (timeout), the algorithm cannot determine the beginning and the end of calls accurately, and it may mistakenly think that the call has finished. I experienced no false positive identifications.

9.7. Traffic Analysis

In this section, I present the results of the analysis regarding two datasets (called *Callrecords 1* and *Callrecords 2*) captured in a fixed network and a third dataset captured in a mobile network (*Mobile measurement*). For the details of the traffic datasets, see Chapter 7. Since the number of calls in the fixed network datasets was relatively low, these two datasets were aggregated in some cases to increase the number of samples.

In the analysis, I investigated the general daily profile of call and user activity, the characteristic properties of voice flows and also the relation between these properties, which makes it possible to diagnose some interesting behavior of the Skype codec.

9.7.1. Daily Profiles and Call Activity

In Fig. 60, the daily fluctuation of Skype users is presented, based on the detected UDP relations. The number of Skype users includes here all the users who are logged in the Skype network, even if they do not initiate any call. The curves are a bit smoothed. Users not sustaining visible UDP relations – probably because UDP traffic is blocked on their computer – cannot be taken into account; therefore, the real number of logged-on Skype users can be somewhat higher. The other reason for the low ratio of Skype user is that Skype usage was not widespread when the measurements were taken. Fig. 60 shows that *Callrecords 1* contains much less calls and active users than the other dataset.

The number of Skype users logged on to the network follows the general daily tendency of the total number of users, which suggests that a certain ratio of users use the Skype client at home. Some users seem to keep their computer switched on during the night period.

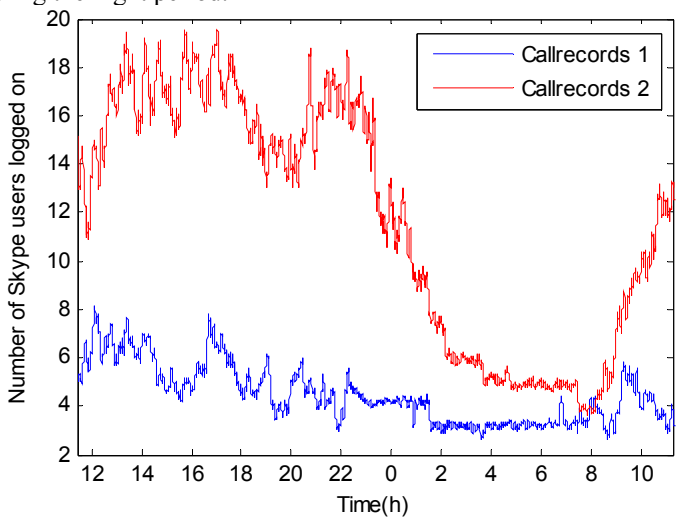

Fig. 60. Daily fluctuation of Skype users based on detected UDP relations

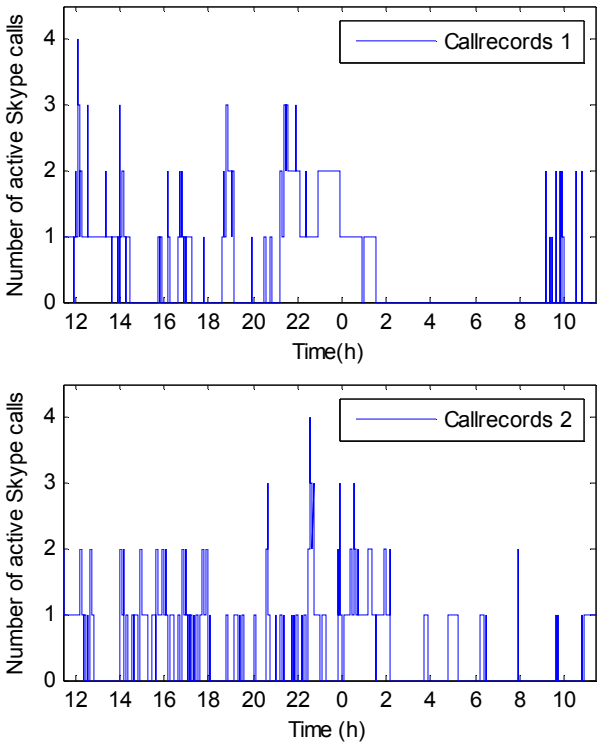

Fig. 61. Daily fluctuation of the number of voice calls in the ADSL domains

The number of voice calls (Fig. 61) also follows a similar daily fluctuation. Calls are coming more frequently in the daytime, though some surprising activity can also be recognized in the 01h-06h AM interval, which suggests some "night birds" among the users or the presence of overseas calls.

The calls seem to be shorter in the daytime and definitely longer in the 21h PM-01h AM period, which could be explained by the fact that the users have more free time for chatting at night. However, only about 130 calls could be detected during the 24 hour period in the fixed network. For this reason, I do not want to draw far-reaching general conclusions based only on these results.

The daily fluctuation of speech hours (Fig. 62) is constructed as follows: the measurement period was divided into smaller, one hour long intervals and the sum of speech time by all users was calculated for each interval. The daily fluctuation of speech hours also confirms the assumption that the calls are longer at the late night period and shorter at daytime. Thus it seems that the call activity and the busy hours of Skype are different from the pattern experienced in PSTN networks.

There is only a small ratio of active Skype users who initiate calls indeed. Most of the users seem to prefer the chat service or just to stay connected and be reachable if needed.

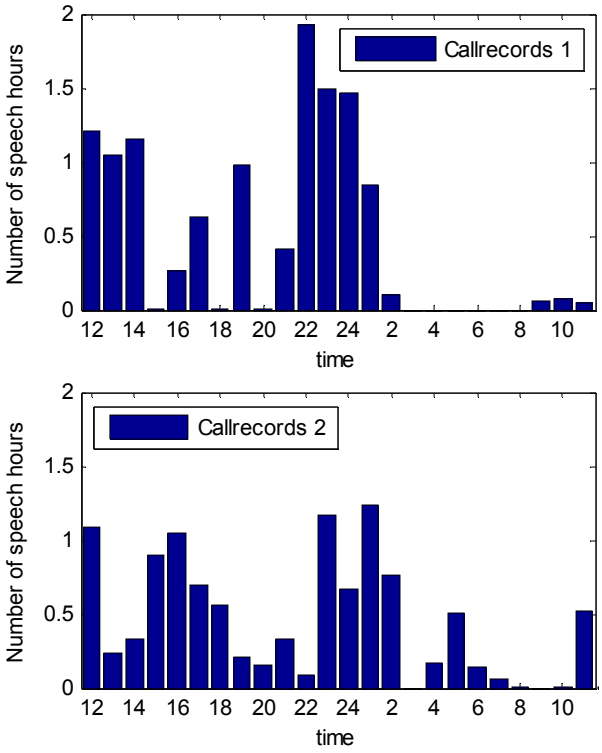

Fig. 62. Daily fluctuation of the speech hours in the ADSL domains

In the almost 3 day long dataset captured in the mobile network, 444 Skype calls could be detected, which allows the calculation of more precise histograms and other statistics. In addition, 66 MSN voice calls were also detected. To detect MSN, first RTP flows were identified with the method proposed by [202], and then MSN voice calls were selected based on the content type field. This way it is possible to make a comparison between the two popular voice services.

The fluctuation of active Skype calls in the mobile network is presented in Fig. 63. The number of Skype calls shows similar daily profiles in all three days of the measurement period. Apart from the typical busy period in the daily hours (occurring in all telecommunication networks), a second busy period can be recognized in the evening and night hours. This suggests that many users use Skype at night for private conversations. Non-busy periods can be seen between about 1 AM and 7 AM. MSN calls seem to follow some similar daily profile, although not enough MSN calls were detected to form general conclusions.

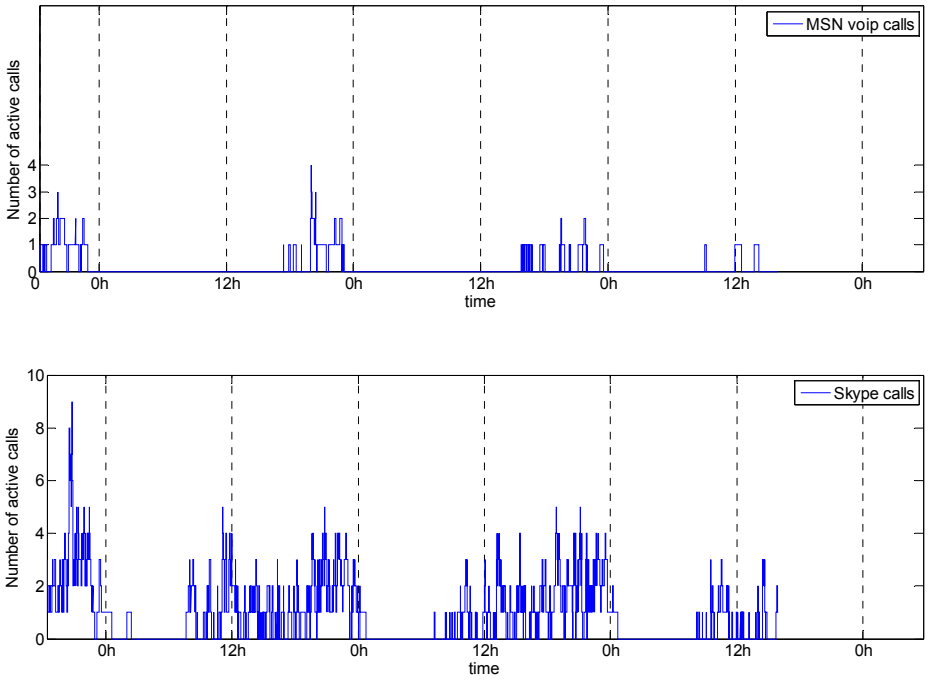

Fig. 63. Daily fluctuation of active MSN and Skype calls in the mobile network

Fig. 64 shows the number of speech hours in each one hour long interval of the 3-day long measurement. The figure confirms that the main busy hours are at night.

The daily profiles of active Skype calls and speech hours are very similar in both the ADSL domain and the mobile network (compare Fig. 61, Fig. 62, Fig. 63, and Fig. 64). These figures tell that the main busy hours, in case of Skype, are the evening and night hours. This suggests that most people use Skype as a free time activity to talk with friends. It may also indicate that, at daytime, people either do not have time to conduct private conversations, or they do not prefer Skype because they can use the landline phone for free at the workplace. It seems that most Skype users (in Hungary) should be regarded as home users, not business users. Few people or companies deploy Skype for business purposes.

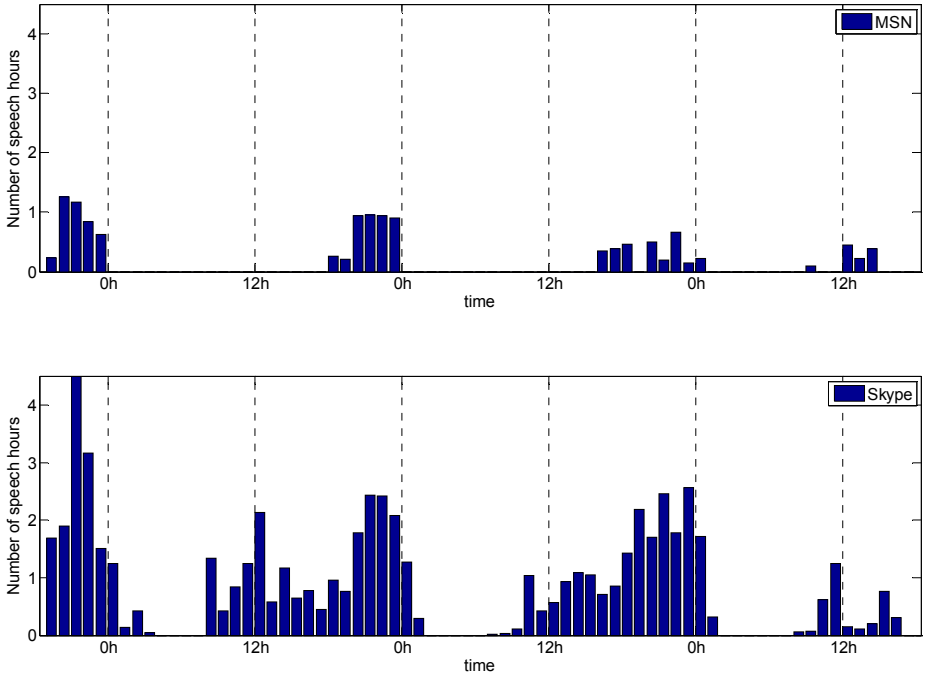

Fig. 64. Daily fluctuation of the speech hours in the system during the 3-day long mobile measurement

9.7.2. Basic Call Characteristics

The next two figures (Fig. 65 and Fig. 66) show the bandwidth and the packet rate of the detected Skype calls. Fig. 65 shows that the bandwidth of Skype calls is usually between 18 and 70 Kbps, typically around 40 Kbps. Fig. 66 shows one prominent and one small peak in the histogram of the packet rate of Skype speech flows, which correspond to the typical inter-arrival times (30 and 60 ms). It can be seen that packet rates smaller than the typical ones (16 and 33 packets/sec) also occur. The reason for this is that the termination of a flow cannot be determined accurately in some cases, and the codec may switch rate at the middle of a call.

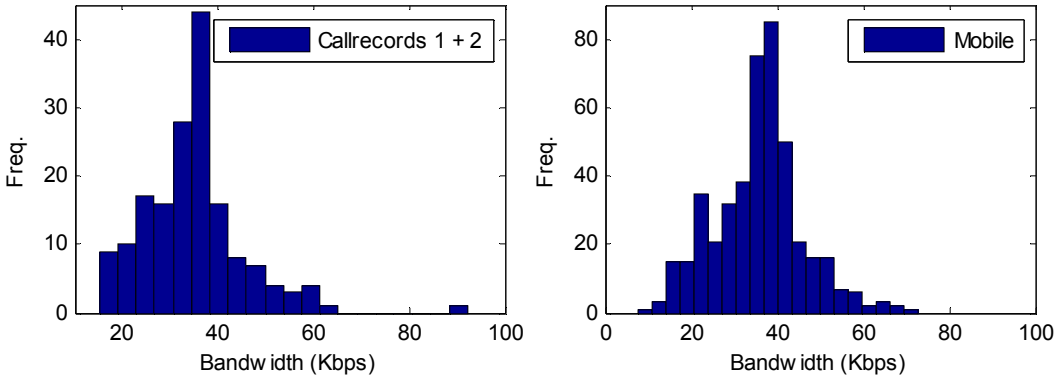

Fig. 65. Histogram of the bandwidth of Skype calls in one direction in the fixed ADSL network (left) and in the mobile network (right)

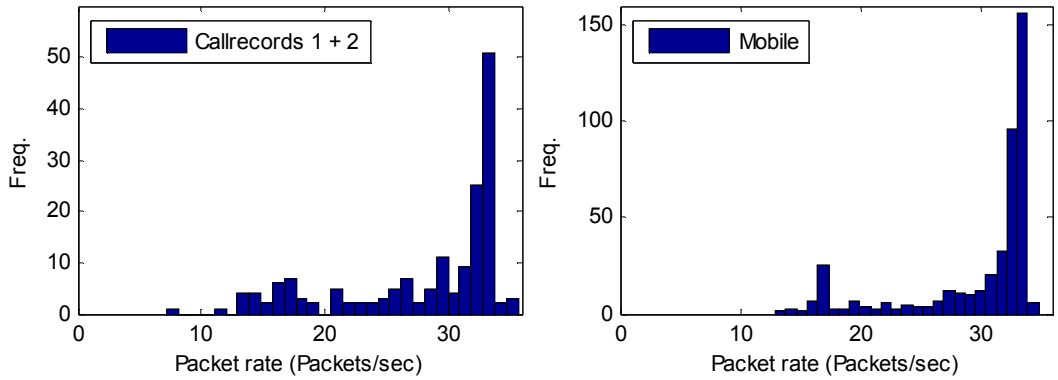

Fig. 66. Histogram of the packet rate of Skype calls in one direction in the fixed ADSL network (left) and in the mobile network (right)

The average packet size of Skype speech flows is plotted in Fig. 67. The figure shows that the typical packet size (including IP and TCP/UDP headers) is somewhere between 100 and 200 bytes, which is also confirmed by my test measurements on local computers. Smaller packet size and bandwidth occur in one direction when separate inbound and outbound TCP flows belong to the call.

Fig. 68 shows the histogram of the duration of Skype calls. It suggests an exponential-like distribution.

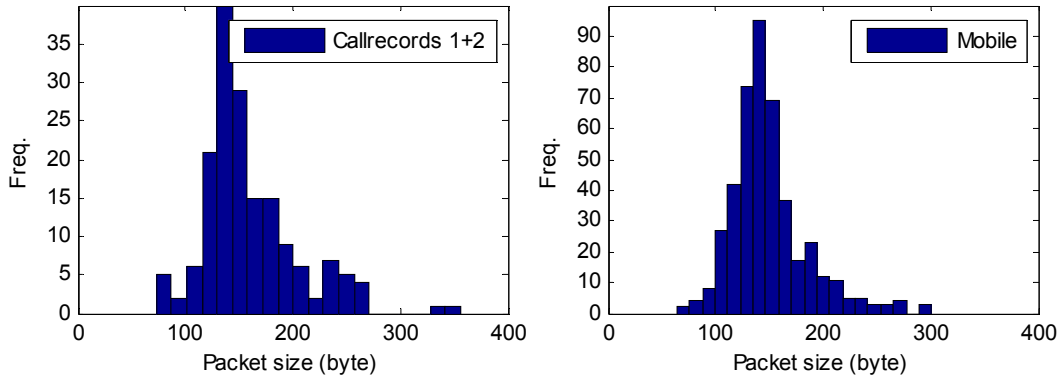

Fig. 67. Histogram of the average packet size of Skype speech flows in the fixed ADSL network (left) and in the mobile network (right)

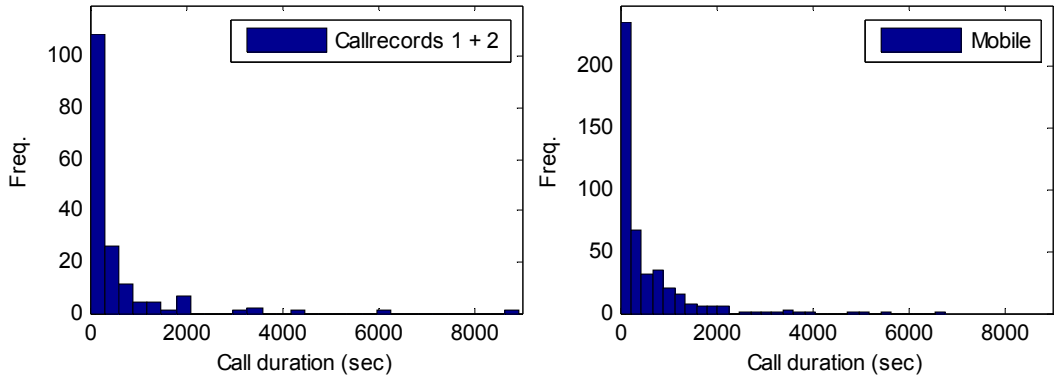

Fig. 68. Histogram of the duration of Skype calls in the fixed ADSL network (left) and in the mobile network (right)

9.7.3. Relations between Call Characteristics

The following figures depict the correlation between the previous characteristic properties of Skype data flows and voice packets. Fig. 69 shows an approximately linear relationship between bandwidth and average packet size of Skype flows. Each data point corresponds to a Skype flow (in outbound direction). It can be seen that most the points are on or over the linear line which has a gradient corresponding to an inter-arrival time of 30 ms. Data points over the line have higher average inter arrival time (between 30 and 60 ms). This figure tallies with the observed behavior of the Skype codec.

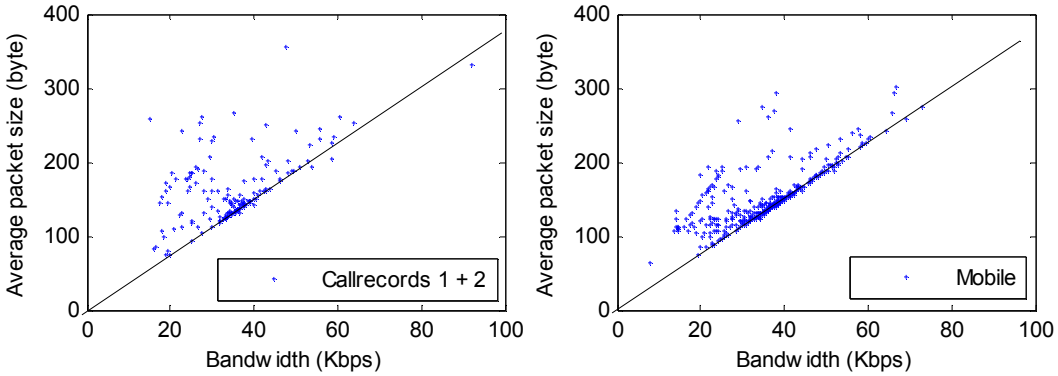

Fig. 69. Average packet size as a function of bandwidth for Skype calls (in one direction) in the fixed ADSL network (left) and in the mobile network (right)

Fig. 70 shows the correlation between packet rate and bandwidth of Skype calls (in outbound direction). Two dense areas can be seen in the figure, corresponding to 16/33 packets/sec and 20-30/35-45 Kbps, respectively, as indicated in the figure. The two horizontal lines indicate the typical packet rates of Skype flows.

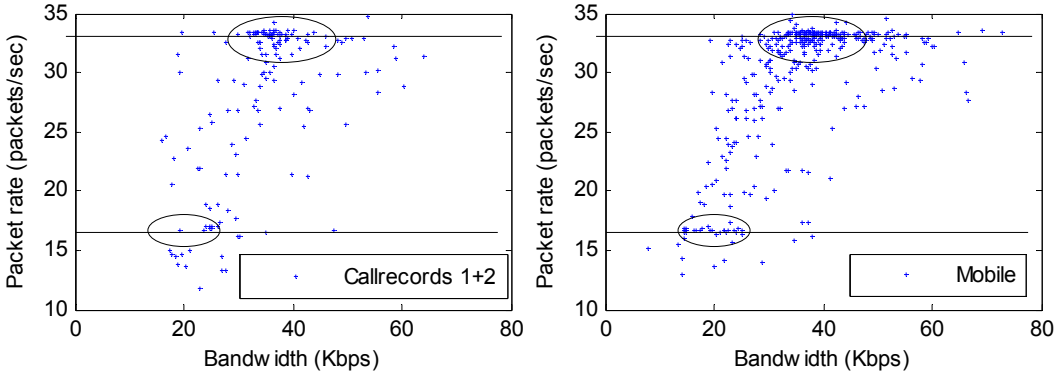

Fig. 70. Packet rate as a function of bandwidth of Skype calls (in one direction) in the fixed ADSL network (left) and in the mobile network (right)

All properties of the detected Skype calls (histogram of bandwidths, packet rates, average packets sizes and call durations) in the mobile network are very similar to those that I experienced in the ADSL domain. Thus, the mobile environment does not seem to affect these properties. This is not a surprising fact, knowing that the users in the mobile network have 3G or HSDPA connections, which provide higher bandwidth in both uplink and downlink direction than what is required by Skype.

9.8. Conclusion

In this chapter, I examined the operation of the Skype peer-to-peer overlay network. The numerous connections initiated by the client towards Skype super nodes, dedicated servers and other peers were investigated in details. According to these observations I proposed some heuristic methods based on flow-dynamics and flow- and packet-level characteristics to identify Skype traffic. The algorithms use packet headers only and the extracted flow-level information; no packet payload information is necessary.

The novel decision based identification algorithm focuses on the observable behavior of the Skype protocol. First, candidate Skype hosts are detected using traditional IP and port-based detection, together with the identification of special connections initiated by the client. Skype calls are then discovered by exploiting the properties of speech flows, timing of voice packets and source host information found in the first step. This method can detect logged on but idle Skype clients as well, in addition to calls. The method expects pre-captured (offline) data as input. However, it could be built into an online identification tool.

I performed traffic analysis in both fixed and mobile network environment in Hungary, using several traces, including two 24h real data sets measured in an ADSL domain and a 72h trace captured in a 3G/HSDPA mobile network. Several analysis results were presented in this chapter, including the daily profile of Skype calls and user activity in the aggregate traffic, the characteristic properties of Skype voice calls, and some interesting findings about the Skype codec.

I also presented the validation of the identification algorithms based on active test measurements at my university department and in a mobile test cell.

10. Summary

In my theses, I was dealing with two different areas of telecommunications that had great influence on the development of Internet in the last decade. Traffic has been rapidly increasing in access and in transport networks as well because new technologies, principles, and blockbuster applications based on them arose from time to time.

Optical technology has been introduced to solve the capacity issue created by the continuously increasing network load. In fact, optical networking (with the aid of WDM) is the only candidate that has enough reserves to provide sufficient capacity on long term in access, metro, and transport networks. At the beginning the research focused on *static optical networks* assuming constant traffic. Although this assumption is suitable in transport and backbone networks, it cannot be applied lower on lower aggregation levels, where the traffic is more variable. The managing of *continuously changing (dynamic) traffic* demands attracted recently more attention. The interoperability model of network layers also changed from the *overlay model* (i.e. separately provisioned layers) to the *vertically integrated* scheme. In Chapter 2 of my theses I proposed a novel, statistical utilization based method for dimensioning of key network resources (number of O/E, E/O conversion ports per node and number of wavelength per link) in a multilayer optical network assuming dynamic traffic. The method makes histograms of link and node utilization in every step of an iterative dimensioning process and modified the configuration accordingly.

Considering *physical effects* (resulting in the degradation of the signal quality) in the provisioning, configuration and routing of optical networks is definitely one of the "hottest" research areas nowadays. In Chapter 3 I gave the exact ILP formulation (providing global optimum) of this cross-layer optimization problem (assuming that the routing functions are taken into account as signal penalties) for both single and double layer networks. In the first case routing was performed in the optical layer by taking physical effects into account. While in the more advanced multilayer case the full joint optimization of RWA along with grooming and along with physical considerations was carried out.

Presently optical networks are dominant primarily in transport and backbone networks. However, optical technology is making progress in the access as well. Besides high-speed Internet connection, optical access can provide telephone and HDTV (in general high definition multimedia, which is the most bandwidth demanding) services. Some of these might be realized on an IP basis. *Multicast delivery* (optical multicast) can be the key technology to distribute video channels in both transport and access optical networks by lowering resource usage and cost. In Chapter 4 I proposed WL graph models and a new ILP formulation to route unicast and multicast demands in WDM networks. I evaluated the cost and the resource usage of multicast routing with and without optical layer branching. I also inspected the subservience of grooming in routing and how the increasing traffic load (and increasing bandwidth of demands) influences the efficiency of multicast routing.

In Chapter 5 I showed that significant amount of network resources can be spared by regular reconfiguration of dynamically changing light-trees. Several heuristic methods for maintaining the multicast tree close to the optimal Steiner tree were applied and measured their performance; optimal topology was determined by ILP. I showed that the reconfiguration period has an optimal length considering the cost of reconfiguration in addition to the network cost. I investigated the reconfiguration gain for different network topologies, various network loads and the expedience of grooming.

The key innovation that has most recently triggered a dramatic growth in the volume of Internet traffic is *P2P technology*. It created the ground for blockbuster file-sharing applications generating significant part (60-80% or even more) of user traffic in access networks and likely a considerable portion of the unknown traffic. P2P file-sharing traffic is often associated with illegal content and pirate copies, which make the Internet providers concerned.

Other important applications of the P2P technology are robust instant messaging, Internet telephony, video conferencing, and video distribution services. Although these applications do not necessarily generate significant traffic, they may be in conflict with the interest of landline and mobile phone operators by offering a real alternative to traditional telephony services.

As a result of all these factors, network operators are interested in measuring, regulating, controlling, filtering or even in blocking of P2P traffic. On the other hand recent popular P2P applications try to hide their presence and disguise their generated traffic resulting in the problematic issue of traffic identification.

In Chapter 8 I presented a novel, robust *P2P traffic identification* method based on a collection of rules derived from the general behavior of P2P traffic and applications. The method relies on flow dynamics a does not use any packet payload. Its efficiency and robustness have been proven by a validation study. I also presented a comprehensive traffic analysis focusing on the similarities and differences of P2P and non-P2P traffic, including user behavior and traffic characteristics on packet, flow and aggregate level.

Chapter 9 of my theses concentrates specifically on Skype, the number one P2P Internet telephony application on the world. After giving a widespread description of Skype operation and network entities, I

proposed a flow-dynamics based heuristic method for the *identification of Skype traffic*. The algorithm exploits the observable properties of Skype protocol, including packet headers and time behavior. The trustiness of the algorithm has been confirmed by active test measurements. In addition the identification method has been also employed on real traffic traces captured in fixed and mobile networks. The results of this traffic analysis (e.g., the daily profile and characteristic properties of voice calls, user activity) are also included in Chapter 9.

References

[1] Harry G. Perros, “Connection-oriented Networks: SONET/SDH, ATM, MPLS and Optical Networks”, John Wiley & Sons, 2005

[2] Casimer DeCusatis (editor), “Handbook of Fiber Optic Data Communication”, Academic Press, 2002

[3] John P. Powers, “Introduction to Fiber Optic Systems”, 2nd edition, McGraw-Hill International Editions, 1997

[4] Govind P. Agrawal, “Fiber-Optic Communication Systems”, 3rd edition, John Wiley & Sons, 2002

[5] Mohammad Ilyas, Hussein T. Mouftah, “The Handbook of Optical Communication Networks”, CRC Press, 2003

[6] Achyut K. Dutta, Niloy K. Dutta, Masahiko Fujiwara, “WDM Technologies – Volume III: Optical Networks”, Elsevier Academic Press, 2004

[7] Rajiv Ramaswami and Kumar N. Sivarajan, “Optical Networks: A Practical Perspective”, Morgan Kaufmann, 2002. (pp 40, 43, 49)

[8] Krishna M. Sivalingam, Suresh Subramaniam, “Optical WDM Networks – Principles and Practice”, Kluwer Academic Publishers, 2002

[9] “Spectral grids for WDM applications: DWDM frequency grid”, ITU-T Recommendation G.694.1, June 2002

[10] “Spectral grids for WDM applications: CWDM frequency grid”, ITU-T Recommendation G.694.2, December 2003

[11] J. P. Laude and C-N. Zah, “Wavelength Division Multiplexing/Demultiplexing (WDM) using Diffraction Gratings”, SPIE-Application, Theory and Fabrication of eriodic Structures, 503:22–28, 1984. (p 44)

[12] David A. Smith, Jane E. Baran, John J. Johnson, and Kwok-Wai Cheung, “Integrated-Optic Acoustically-Tunable Filters for WDM Networks”, IEEE Journal on Selected Areas in Communications, 8(6):1151–1159, August 1990. (p 44)

[13] Arjen R. Vellekoop and Meint K. Smit, “Four-channel integrated-optic wavelength demultiplexer with weak polarization dependence”, IEEE/OSA Journal of Lightwave Technology, 9(3):310–314, March 1991. (p 44)

[14] V. E. Beneš, “Mathematical Theory of Connecting Networks and Telephone Traffic”, Academic Press, New York, 1965. (p 47)

[15] Armand Neukermans and Rajiv Ramaswami, “MEMS Technology for Optical Networking Applications”, IEEE Communications Magazine, 39(1):62–69, January 2001. (p 47)

[16] *Calient Networks DiamondWave Ships for Revenue, Passes NEBS and ETSI Testing*, http://www.calient.net/press.html, 2002. (p 47)

[17] V.A. Aksyuk et al., “238x238 Surface Micromachined Optical Crossconnect With 2dB Maximum Loss”, *in Optical Fiber Communication*, March 2002. Post Deadline Paper. (p 47)

[18] J. M. H. Elmirghani and H. T. Mouftah, “All-Optical Wavelength Conversion Technologies and Applications in DWDM Networks”, IEEE Communications Magazine, 38(3):86–92, March 2000. (p 48)

[19] D. Chiaroni et al., “New 10 Gbit/s 3R NRZ Optical Regenerative Interface based on Semiconductor Optical Amplifiers for All-Optical Networks”, *in Proceedings of IOOC/ECOC*, 5:41–44, September 1997. (p 48)

[20] Keyao Zhu, Hongyue Zhu, Biswanath Mukherjee, “Traffic Grooming in Optical WDM Mesh Networks”, Springer, 2005

[21] Arun K. Somani, “Survivability and Traffic Grooming in WDM Optical Networks”, Cambridge University Press, 2005

[22] Eytan Modiano, Philip J. Lin, “Traffic Grooming in WDM Networks”, IEEE Communications Magazine, vol. 39, no. 7, July 2001, pp. 124–129

[23] Angela L. Chiu, Eytan Modiano, “Traffic Grooming Algorithms for Reducing Electronic Multiplexing Costs in WDM Ring Networks”, Journal of Lightwave Technology, vol. 18, no. 1, Jan 2000, pp. 2–12

[24] Timothy Y. Chow, Philip J. Lin, “The Ring Grooming Problem”, Networks, vol. 44, no. 3, October 2004, pp. 194–202

[25] Ori Gerstel, Rajiv Ramaswami, Galen Sasaki, “Cost Effective Traffic Grooming in WDM Rings”, IEEE/ACM Transactions on Networking, vol. 8, no. 5, October 2000, pp. 618–630

[26] Ornan Gerstel and Rajiv Ramaswami, “Optical Layer Survivability: A Services Perspective”, *IEEE Communications Magazine*, 38(3):104–113, March 2000. (p 50)

[27] Didier Colle et al., “Data-Centric Optical Networks and Their Survivability”, *IEEE Journal on Selected Areas in Communications*, 20(1):6–20, January 2002. (p 50)

[28] “Terms and definitions for Automatically Switched Optical Networks (ASON)”, ITU-T Recommendation G.8081/Y.1353, June 2004

[29] “Generic functional architecture of transport networks”, ITU-T Recommendation G.805, March 2000

[30] “Architecture for the automatically switched optical network (ASON)”, ITU-T Recommendation G.8080/Y.1304, July 2006
[31] “Information Technology – Open Systems Interconnection – Basic Reference Model: The Basic Model”, ISO/IEC 7498-1, 15 June 1996
[32] Stefano Bregni, “Synchronization of Digital Telecommunications Networks”, John Wiley & Sons, 2002
[33] “Synchronous Optical Network (SONET) Transport Systems: Common Generic Criteria”, Telcordia Industry Requirements and Standards, GR-253-CORE, December 2005
[34] “Network node interface for the synchronous digital hierarchy (SDH)”, ITU-T Recommendation G.707, October 2000
[35] “Sub STM-0 network node interface for the synchronous digital hierarchy (SDH)”, ITU-T Recommendation G.708, June 1999
[36] “Architecture of transport networks based on the synchronous digital hierarchy (SDH)”, ITU-T Recommendation G.803, March 2000
[37] Greg Bernstein, Bala Rajagopalan, Debanjan Saha, “Optical Network Control: Architecture, Protocols, and Standards”, Addison-Wesley, 2003
[38] “Generic framing procedure (GFP)”, ITU-T Recommendation G.7041/Y.1303, August 2005
[39] “Efficient and Flexible Transport of Next-Generation Data Services Over SONET/SDH Using GFP, VCAT and LCAS”, Cisco Systems Inc., White Paper, 2005
[40] IEEE 802.3 Ethernet Working Group (and all standards), http://www.ieee802.org/3/
[41] Kevin H. Liu, “IP Over WDM”, John Wiley & Sons, 2002
[42] Neil Jerram, Adrian Farrel, “MPLS in Optical Networks – An Analysis of the Features of MPLS and Generalized MPLS and Their Application to Optical Networks, With Reference to the Link Management Protocol and Optical UNI”, Data Connection Ltd., White Paper, 2004
[43] “MPLS – An introduction to multiprotocol label switching”, Nortel Networks Corporation, White Paper, April 2001
[44] Dimitri Papadimitriou, Bart Rousseau, “Demystifying GMPLS – A technical perspective”, Alcatel, White Paper, 2003
[45] Nic Larkin, “ASON and GMPLS – The Battle of the Optical Control Plane”, Data Connection Ltd., White Paper, 2004
[46] “Architecture of optical transport networks”, ITU-T Recommendation G.872, November 2001
[47] Jim D. Jones et al., “User Network Interface (UNI) 1.0 Signaling Specification, Release 2: Common Part”, Optical Internetworking Forum, February 27, 2004
[48] “Architecture and specification of data communication network”, ITU-T Recommendation G.7712/Y.1703, March 2003
[49] Govind P. Agrawal, “Lightwave Technology – Telecommunication Systems”, John Wiley & Sons, 2005
[50] T. Cinkler, “Traffic- and λ-Grooming”, IEEE Network, vol. 17, no. 2, pp. 16-21, March/April 2003
[51] M. Vigoureux, et al., “Multilayer Traffic Engineering for GMPLS-enabled Networks”, IEEE Communications Magazine, July 2005
[52] M. Perényi, J. Breuer, T. Cinkler, Cs. Gáspár, “Grooming Node Placement in Multilayer Networks”, in Proc. ONDM 2005, Milano, Italy, February 2005
[53] K. Zhu, Hui Zang, Biswanath Mukherjee, “Design of WDM Mesh Networks with Sparse Grooming Capability”, GlobeCom 2002, Taipei, Taiwan, November 2002
[54] S. Subramaniam, M. Azizoglu, A. K. Somani, “On Optimal Converter Placement in Wavelength-Routed Networks”, IEEE/ACM Transactions on Networking, vol. 7, no. 5, October 1999
[55] X. Chu, Bo Li, Zhensheng Zhang, “Sparse-Partial Wavelength Conversion in Wavelength-Routed All-Optical Networks”, in Proc. OptiComm 2003, Dallas, TX USA, October 2003
[56] S. K. Bose, et al., “Sparse Converter Placement in WDM Networks and their Dynamic Operation Using Path-Metric Based Algorithms”, in Proc. ICC 2002, New York, NY USA, April/May 2002
[57] N. S. C. Correia , J. Coimbra, and M. C. R. Medeiros, “Sparse traffic grooming in WDM networks using coarse granularity OXCs”, Photonic Network Communications, Volume 17, Number 1 / February, 2009
[58] O. Awwad, A.I. Al-Fuqaha, M. Guizani, “Genetic Approach for Traffic Grooming, Routing, and Wavelength Assignment in WDM Optical Networks with Sparse Grooming Resources”, *in Proc. of* IEEE International Conference on Communications, Istanbul Turkey, 2006
[59] W. Yao, M. Li, B. Ramamurthy, “Performance analysis of sparse traffic grooming in WDM mesh networks”, *in Proc. of* IEEE International Conference on Communications (ICC2005), 2005
[60] M. Sivakumar, K.M. Sivalingam, “Limited Grooming Architectures and Groomer-port Placement in Optical WDM Mesh Networks”, *in Proc. of* 3rd International Conference on Broadband Communications, Networks and Systems (BROADNETS), 2006
[61] O. Awwad, A. Al-Fuqaha, A. Rayes, “Performance of WDM Mesh Networks with Limited Traffic Grooming Resources”, IFIP International Conference on Wireless and Optical Communications Networks

(WOCN'07), 2007
[62] XC-VXL-10G/2.5G Cross Connect Cards for the Cisco ONS 15454 SDH MSPP, Cisco ONS 15454 60G/5G High-Order/Low-Order XC-VXC Cross-Connect Card http://www.cisco.com/
[63] LambdaXtremeTM, Lambda Unite MMS datasheet http://www.alcatel-lucent.com/wps/portal
[64] MARCONI MHL 3000 CORE datasheet http://www.ericsson.com
[65] OptiX BWS 1600G DWDM datasheet http://www.huawei.com/
[66] LambdaDriver WDM Tunable Cards datasheet http://www.huawei.com/
[67] LambdaDriver Tunable 10GE WDM cards, TM-DXFP20T and TM-DXFP35T DM Tunable Cards http://www.mrv.com
[68] N. Wauters, P. Demister, "Design of the Optical Path Layer in Multiwavelength Cross-Connected Networks", IEEE Journal on Selected Areas in Communications, vol. 14, no. 5 pp. 881-892, June 1996
[69] R. Ramaswami, K.N. Sivarajan, "Routing and Wavelength Assignment in All-Optical Networks", IEEE Transaction on Networking, vol. 3 no. 5 pp. 489-500, Oct. 1995
[70] G. P. Agrawal, "Nonlinear Fiber Optics", CA: Academic, 1989
[71] A. Cartaxo, "Impact of modulation frequency on cross-phase modulation effect in intensity modulation – direct detection WDM systems", IEEE Photon. Technol. Lett., 10(9):1268–1270. 1998.
[72] J. Wang, K. Petermann, "Small signal analysis for dispersive optical fiber communication systems", IEEE J. Lightwave Technol., 10(1):96-100. 1992.
[73] A.R. Chraplyvy, "Limitations on lightwave communications imposed by optical-fiber nonlinearities", IEEE J. Quantum Electronics, 8(10):1548-1557. 1990.
[74] W. Zeiler, F. Di Pasquale, P. Bayvel, J.E. Midwinter, "Modeling of Four-Wave Mixing and Gain Peaking in Amplified WDM Optical Communication Systems and Networks", IEEE J. Lightwave Technol., 14(9):1933-1942. 1996.
[75] D. Cotter, A.M. Hill, "Stimulated Raman crosstalk in optical transmission: Effects of group velocity dispersion", IEEE Electron. Lett., 20:185-187. 1984.
[76] K.-P. Ho, "Statistical Properties of Stimulated Raman Crosstalk in WDM Systems", IEEE J. Lightw. Technol., 18(7):915-921. 2000.
[77] L. Pavel, "Power control for OSNR optimization in optical networks: a non-cooperative game approach", in Proc. 43rd IEEE Conf. Decision and Control, 3033-3038, Dec. 2004.
[78] Y. Pan and L. Pavel, "OSNR optimization in optical networks: extension for capacity constraints", Proc. American Control. Conf., Portland, June 2005.
[79] L. Pavel, "OSNR Optimization via end-to-end Power Control: A Central Cost Approach", in Proc. IEEE INFOCOM 2005, Miami, March 2005.
[80] Huang Yurong, J.P. Heritage, B. Mukherjee, "Connection provisioning with transmission impairment consideration in optical WDM networks with high-speed channels", *Journal of Lightwave Technology*, Volume 23, Issue 3, 2005
[81] Tao Deng, S. Subramaniam, Jinghao Xu, "Crosstalk-aware wavelength assignment in dynamic wavelength-routed optical networks", *in Proc. of* 1st Int. Conf. on Broadband Networks (BroadNets2004), 2004
[82] M. Karimiyan-Mohammadabadi, M.H.V. Samiei, "Static Routing Considering Network-layer and Physical-layer in Wavelength-Routed Optical Networks", *in Proc. of* the 9th International Conference on Advanced Communication Technology, 2007
[83] Jun He, M. Brandt-Pearce, S. Subramaniam, "QoS-Aware Wavelength Assignment With BER and Latency Constraints for All-Optical Networks", *Journal of Lightwave Technology*, Volume 27, Issue 5, 2009
[84] S. Pachnicke, T. Paschenda, P.M. Krummrich, "Physical Impairment Based Regenerator Placement and Routing in Translucent Optical Networks", *in Proc. of* Conference on Optical Fiber communication/National Fiber Optic Engineers Conference(OFC/NFOEC2008), 2008
[85] S. Al Zahr, M. Gagnaire, N. Puech, "Impact of wavelength assignment strategies on hybrid WDM network planning", *in Proc. of* 6th International Workshop on Design and Reliable Communication Networks (DRCN2007), 2007
[86] H.A. Pereira, D.A.R. Chaves, C.J.A. Bastos-Filho, J.F. Martins-Filho, "Noise Penalties Modeling for the Performance Evaluation of All-Optical Networks", *in Proc. of* 9th International Conference on Transparent Optical Networks (ICTON'07), 2007
[87] http://vpisystems.com/
[88] Tony Antony, Ashwin Gumaste, "WDM Network Design", Cisco Press Feb 7, 2003
[89] Ioannis Tomkos et al., "Performance Engineering of Metropolitan Area Optical Networks through Impairment Constraint Routing", OptiComm, August 2004
[90] Szilárd Zsigmond, Gábor Németh, Tibor Cinkler, "Mutual Impact of Physical Impairments and Grooming in Multilayer Networks", ONDM 2007 Athens Greece, May 2007
[91] T. Cinkler, R. S. Castro, S. Johansson, "Configuration and Re-Configuration of WDM networks", NOC'98, European Conference on Networks and Optical Communications, Manchester, UK, 1998

[92] R. Inkret et al., “Advanced Infrastructure for Photonic Networks: Extended Final Report of COST Action 266”, Faculty of Electrical Engineering and Computing, University of Zagreb, 2003
[93] ILOG CPLEX Mathematical Programming Optimizer, http://www.ilog.com/products/cplex/
[94] B. Quinn and K. Almeroth, “IP multicast applications: Challenges and solutions”, IETF RFC 3170, Sep. 2001
[95] Madhyastha et al., “Grooming of multicast sessions in WDM ring networks”, OptiComm 2003: Optical Networking and Communications, Nov. 2003
[96] G. V. Chowdhary and C. S. R. Murthy, “Grooming of Multicast Sessions in WDM Mesh Networks”, Workshop on Traffic Grooming, 2004
[97] X. Zhang et al., “Constrained Multicast Routing in WDM Networks with Sparse Light Splitting”, Journal of Lightwave Technology, vol. 18, issue 12, p. 1917, Dec. 2000
[98] D. Yang and W. Liao, “Design of light-tree based logical topologies for multicast streams in wavelength routed optical networks,” in Proc. IEEE Information Communications (INFOCOM), San Francisco, CA, Apr. 2003
[99] X. H. Jia et al., “Optimization of Wavelength Assignment for QoS Multicast in WDM Networks”, IEEE Transactions on Communications, vol. 49, no. 2, Feb. 2001
[100] T. Cinkler, “ILP formulation of Grooming over Wavelength Routing with Protection”, ONDM 2001, 5th Conference on Optical Network Design and Modeling, Wien, Feb. 2001
[101] NRS core network topology (16 nodes, 22 links) http://www.ibcn.intec.ugent.be/INTERNAL/NRS/index.html
[102] Alexander Schrijver, “Theory of Linear and Integer Programming”, John Wiley and Sons, 1998.
[103] Fatih Köksal and Cem Ersoy, “Multicasting for all-optical multifiber networks”, Journal of Optical Networking, Vol. 6, Issue 2, Jan 2007
[104] R. Mustafa, A.E. Kamal, “Design and provisioning of WDM networks with multicast traffic grooming”, IEEE Journal on Selected Areas in Communications, Volume 24, Issue 4, 2006
[105] X. Huang et al., “Multicast Traffic Grooming in Wavelength-Routed WDM Mesh Networks Using Dynamically Changing Light-Trees”, Journal of Lightwave Technology, vol. 23, no. 10, Oct. 2005
[106] Ahmed E. Kamat et al., “Algorithms for multicast traffic grooming in WDM mesh networks”, IEEE Communications Magazine, Vol. 44, Issue 11, Nov. 2006
[107] A. Khalil et al., "Dynamic provisioning of low-speed unicast/multicast traffic demands in mesh-based WDM optical networks", Journal of Lightwave Technology, Volume 24, Issue 2, Feb. 2006
[108] Keyao Zhu et al., “Traffic Engineering in Multi-granularity Heterogeneous Optical WDM Mesh Networks Through Dynamic Traffic Grooming”, IEEE NETWORK vol. 17, no. 2, Mar/Apr. 2003
[109] J. Wang, B. Chen, “Dynamic Wavelength Assignment for Multicast in All-Optical WDM Networks to Maximize the Network Capacity”, IEEE Journal on Selected Areas in Communication, Vol. 21, No. 8, Oct. 2003
[110] G. Chowdhary, C. S. R. Murthy, “Dynamic multicast traffic engineering in WDM groomed mesh networks”, WS on Traffic Grooming, 2004
[111] C. Boworntummarat et al., “Light-tree based protection strategies for multicast traffic in transport WDM mesh networks with multifiber systems”, IEEE International Conference on Communications, Jun. 2004
[112] E. Dotaro, M. Vigoureux, D. Papadimitriou: “Multi-Region Networks: Generalized Multi-Protocol Label Switching (GMPLS) as Enabler for Vertical Integration”, Alcatel Technology White Paper, February 2005
[113] Jun Zheng, Hussein T. Mouftah, “Optical WDM Networks: Concepts and Design Principles”, *Wiley-IEEE Press*, 2004
[114] A. Betker et al., “Reference transport network scenarios”, Technical report, BMBF-Project MultiTeraNet, 2003. http://www.pt-it.pt-dlr.de/_media/MTN_Referenz_Netze.pdf
[115] Thomas H. Cormen et al., “Introduction to Algorithms”, Second Edition, MIT Press and McGraw-Hill, 2001, Section 23.2: “The algorithms of Kruskal and Prim”, pp.567–574.
[116] M. Ali, J.S. Deogun, “Cost-effective implementation of multicasting in wavelength-routed networks”, Journal of Lightwave Technology, Vol. 18. Issue 12., 2000
[117] P. Soproni, M. Perényi, T. Cinkler, “Grooming-Enhanced Multicast in Multilayer Networks”, ONDM 2007, Athens, May 2007
[118] D. Hart, “A Brief History of NSF and the Internet”, August 13, 2003, http://www.nsf.gov/news/news_summ.jsp?cntn_id=103050
[119] N. B. Azzouna, F. Guillemin, “Impact of peer-to-peer applications on wide area network traffic: an experimental approach”, IEEE Global Telecommunications Conference, Vol.3, 2004, pp. 1544-1548.
[120] S. Kamei, T. Kimura, “Practicable network design for handling growth in the volume of peer-to-peer traffic”, IEEE Pacific Rim Conference on Communications, Computers and signal Processing, Vol.2, 2003,

pp. 597-600.
[121] Wikipedia: peer-to-peer, http://en.wikipedia.org/wiki/Peer-to-peer
[122] PeerCast P2P Broadcasting, http://www.peercast.org
[123] PeerMe, http://www.peerme.com
[124] Avaya one-X Quick Edition, http://www.carrollcommunications.com/avaya-one-x.html
[125] Joost, http://www.joost.com
[126] Skype, http://www.skype.com
[127] BitTorrent, http://www.bittorrent.com
[128] Distributed Hash Table (DHT) Links, http://deim.urv.cat/~cpairot/dhts.html
[129] eMule, http://www.emule-project.net
[130] Xiaoqiu Shi; Jinsong Han; Yunhao Liu; Ni, L.M., "Popularity Adaptive Search in Hybrid P2P Systems", Parallel and Distributed Processing Symposium, 2007. IPDPS 2007, 26-30 March 2007
[131] E. Adar, B. A. Huberman, "Free Riding on Gnutella", Technical report, Xerox PARC, Aug. 2000.
[132] S. Saroiu, P. K. Gummadi, S. D. Gribble, "A Measurement Study of Peer-to-Peer File Sharing Systems", in Proc. Multimedia Computing and Networking (MMCN'02), Jan. 2002.
[133] S. Saroiu, K. P. Gummadi, R. Dunn, S. D. Gribble, H. M. Levy, "An Analysis of Internet Content Delivery Systems", in Proc. 5th Symposium on Operating Systems Design and Implementation, Boston, MA, USA, Dec. 2002.
[134] K. P. Gummadi, R. J. Dunn, S. Saroiu, S. D. Gribble, H. M. Levy, J. Zahorjan, "Measurement, Modeling, and Analysis of a Peer-to-Peer File-Sharing Workload", in Proc. 19th ACM Symposium on Operating Systems Principles (SOSP-19), Bolton Landing, NY. Oct. 2003.
[135] T. Karagiannis, A. Broido, N. Brownlee, K. Claffy, M. Faloutsos, "File-Sharing in the Internet: A Characterization of P2P Traffic in the Backbone", Technical Report, UC Riverside, 2003.
[136] S. Sen, J. Wang, "Analyzing Peer-to-Peer Traffic across Large Networks", IEEE/ACM Transactions on Networking, 12(2):219-232, 2004.
[137] K. Tutschku, "A Measurement-based Traffic Profile of the eDonkey Filesharing Service", PAM 2004: 12-21.
[138] J. A. Pouwelse, P. Garbacki, D.H.J. Epema, H.J. Sips, "The BitTorrent P2P File-Sharing System: Measurements and Analysis", 4th Int. workshop on Peer-to-Peer Systems (IPTPS'05), Feb. 2005.
[139] Z. Ge, D. R. Figueiredo, S. Jaiswal, J. Kurose, D. Towsley, "Modeling Peer-Peer File Sharing Systems", in Proc. INFOCOM'03, San Francisco, CA, Mar. 2003.
[140] K. K. Ramachandran, B. Sikdar, "An Analytic Framework for Modeling Peer to Peer Networks", in Proc. INFOCOM'05, 2005.
[141] G. de Veciana, X. Yang, "Fairness, Incentives and Performance in Peer-to-Peer Networks", in Proc. Allerton Conf. on Communication, Control and Computing, 2003.
[142] X. Yang, G. de Veciana, "Service Capacity of Peer to Peer Networks", in Proc. INFOCOM'04, 2004.
[143] D. Qiu, R. Srikant, "Modeling and Performance Analysis of BitTorrent-Like Peer-to-Peer Networks", in Proc. ACM SIGCOMM'04, Portland, OR, Aug. 2004.
[144] B. Yang, S. Kamvar, H. Garcia-Molina, "Addressing the Non-Cooperation Problem in Competitive P2P Systems", Workshop on Peer-to-Peer and Economics, Jun. 2003.
[145] D. Hughes, I. Warren, G. Coulson, "Improving QoS for Peer-to-Peer Applications through Adaptation", in Proc. of the 10th Int. Workshop on Future Trends in Distributed Computing Systems (FTDCS 2004), Suzhou, China, May 26-28, 2004.
[146] E. Kalyvianaki, I. Pratt, "Building Adaptive Peer-To-Peer Systems", in Proc. 4th Int. Conf. on Peer-to-Peer Computing (P2P'04), 2004.
[147] M. Iguchi, M. Terada, K. Fujimura, "Managing Resource and Servent Reputation in P2P Networks", in Proc. 37th Annual Hawaii Int. Conf. on System Sciences (HICSS'04), 2004.
[148] G. Ding, B. Bhargava, "Peer-to-Peer File-Sharing over Mobile Ad hoc Networks", in Proc. PerCom Workshops, 2004.
[149] M. Demirbas, H. Ferhatosmanoglu, "Peer-to-Peer Spatial Queries in Sensor Networks", in 3rd IEEE Int. Conf. on Peer-to-Peer Computing (P2P'03), Linkoping, Sweden, Sept. 2003.
[150] M. Roussopoulos, M. Baker, D. S. H. Rosenthal, T. J. Giuli, P. Maniatis, J. C. Mogul, "2 P2P or Not 2 P2P?", IPTPS 2004: 33-43.
[151] Y. Guo, K. Suh, J. Kurose, D. Towsley, "A Peer-to-Peer On-Demand Streaming Service and Its Performance Evaluation", in Proc. IEEE Int. Conf. on Multimedia & Expo (ICME 2003), Baltimore, MD, Jul. 2003.
[152] G. Cugola, G. P. Picco, "Peer-to-Peer for Collaborative Applications", Int. Workshop on Mobile Teamwork Support, Vienna, Austria, Jul. 2002.
[153] C. Buragohain, D. Agrawal, S. Suri, "A Game Theoretic Framework for Incentives in P2P Systems", in Proc. 3rd Int. Conf. on Peer-to-Peer Computing, 2003.

[154] J. Shneidman, DC Parkes, “Rationality and Self-Interest in Peer to Peer Networks”, in Proc. 2nd Int. Workshop on Peer-to-Peer Systems (IPTPS’03), 2003.
[155] T. Risse, P. Knezevic, A. Wombacher, “P2P Evolution: From File-sharing to Decentralized Workflows”, it-Information Technology, 4:193--199, Oldenbourg, 2004.
[156] A. Gerber, J. Houle, H. Nguyen, M. Roughan, S. Sen, “P2P The Gorilla in the Cable”, in National Cable & Telecommunications Association (NCTA) 2003 National Show, Chicago, IL, June 8-11, 2003.
[157] S. Sen, O. Spatscheck, D. Wang, “Accurate, Scalable In-Network Identification of P2P Traffic Using Application Signatures”, in Proc. 13th Int. Conf. on World Wide Web, NY, USA, 2004.
[158] Zhenbin Guo, Zhengding Qiu, “Identification peer-to-peer traffic for high speed networks using packet sampling and application signatures”, in Proc. 9th Int. Conf. on Signal Processing 2008 (ICSP), Leipzig, Germany, 2008
[159] T. Karagiannis, A. Broido, M. Faloutsos, K. Claffy, “Transport Layer Identification of P2P Traffic”, in Proc. 4th ACM SIGCOMM Conf. on Internet Measurement, Taormina, Sicily, Italy, Oct. 25-27, 2004.
[160] R. Meent, A. Pras, “Assessing Unknown Network Traffic”, CTIT Technical Report 04-11, University of Twente, Netherlands, February 2004.
[161] M. Kim, H. Kang, J. W. Hong, “Towards Peer-to-Peer Traffic Analysis Using Flows”, DSOM 2003: 55-67.
[162] F. Constantinou, P. Mavrommatis, “Identifying Known and Unknown Peer-to-Peer Traffic”, in Proc. 5th IEEE Int. Symposium on Network Computing and Applications 2006 (NCA), Cambridge, MA, USA, 2006
[163] W.Q. Cheng, J. Gong, W. Ding, “Identifying BT-like P2P Traffic by the Discreteness of Remote Hosts”, 32nd IEEE Conf. on Local Computer Networks (LCN) 2007, Dublin, Ireland, 2007
[164] T. Karagiannis, A. Broido, N. Brownlee, K. Claffy, M. Faloutsos, “Is P2P dying or just hiding?”, IEEE Globecom, Dallas, Texas, November 2004.
[165] Internet2 NetFlow: Weekly Reports - http://netflow.internet2.edu/weekly/
[166] S. Ohzahata, Y. Hagiwara, M. Terada, K. Kawashima, “A Traffic Identification Method and Evaluations for a Pure P2P Application”, Lecture Notes in Computer Science, p55 Vol. 3431, 2005.
[167] S. Fuke, C. Pan, R. Xiaoli, ”Research of P2P Traffic Identification Based on BP Neural Network”, in Proc. 3rd Int. Conf. on Intelligent Information Hiding and Multimedia Signal Processing 2007 (IIHMSP), Kaohsiung, Taiwan, 2007
[168] B. Raahemi, A. Hayajneh, and P. Rabinovitch, “Classification of peer-to-peer traffic using neural networks”, in Proc. Artificial Intelligence and Pattern Recognition, Orlando, USA, July 2007
[169] B. Raahemi, A. Hayajneh, and P. Rabinovitch, “Peer-to-peer IP traffic classification using decision tree and IP layer attributes”, Int. Journal of Business Data Communications and Networks, 3(4), 2007, pp.60-74.
[170] B. Raahemi, W. Zhong, J. Liu, “Peer-to-Peer Traffic Identification by Mining IP Layer Data Streams Using Concept-Adapting Very Fast Decision Tree”, in Proc. 20th IEEE Int. Conf. on Tools with Artificial Intelligence 2008 (ICTAI), Dayton, OH, USA, 2008
[171] Y. Yang, R. Wang, Y. Liu, X. Zhou, „Solving P2P Traffic Identification Problems Via Optimized Support Vector Machines”, in Proc. Int. Conf. on Computer Systems and Applications 2007 (AICCSA), Amman, Jordan, 2007
[172] D. Zuev, A. W. Moore, “Traffic classification using a statistical approach”, Springer Lecture Notes in Computer Science, Vol. 3431, Springer Berlin, 2005, pp.321-324.
[173] G.P.S. Junior, J.E.B. Maia, R. Holanda, J.N. de Sousa, “P2P Traffic Identification using Cluster Analysis”, 1st Global Information Infrastructure Symposium 2007 (GIIS), Marrakech, Morocco, 2007
[174] V. Pareto, “Cours d’economie politique, Droz, Geneve”, 1896
[175] V. Brazauskas, R. Serfling, “Favorable estimators for fitting Pareto models: A study using goodness-of-fit measures with actual data”, ASTIN Bulletin, vol. 33, no. 2, 2003, pp. 365-381
[176] S. I. Resnick, “Heavy Tail Modeling and Teletraffic Data”, The Annals of Statistics, 25(5):1805--1869, 1997.
[177] L.A. Adamic, B.A. Huberman, “Zipf’s law and the Internet”, Glottometrics, Vol. 3, 2002, pp. 143-150
[178] Fabrice Desclaux, “Skype uncovered – Security study of Skype”, http://www.ossir.org/windows/supports/2005/2005-11-07/EADS-CCR_Fabrice_Skype.pdf, 2005.
[179] Skype Technologies S.A., “Skype - Guide for Network Administrators Version 1.01”, http://www.skype.com/security/guide-for-network-admins.pdf, 2005.
[180] Philippe Biondi, Fabrice Desclaux, “Silver needle in the Skype”, in Proc. of Black Hat Europe, Amsterdam, Holland, 2006.
[181] Salman A. Baset, Henning Schulzrinne, “An Analysis of the Skype Peer-to-Peer Internet Telephony Protocol”, in Proc. of IEEE INFOCOM'06, Barcelona, Spain, 2006.
[182] Sven Ehlert, Sandrine Petgang, “Analysis and Signature of Skype VoIP Session Traffic”, Technical Report NGNI-SKYPE-06b, Fraunhofer FOKUS, Berlin, Germany, 2006.
[183] Wael Ghandour, “Blocking Skype Using Squid and OpenBSD”, Help Net Security (www.net-

security.org), 2005.
[184] K. Suh, D. R. Figueiredo, J. Kurose, D. Towsley, “Characterizing and Detecting Skype-Relayed Traffic”, in Proc. of IEEE INFOCOM'06, Barcelona, Spain, 2006.
[185] Saikat Guha, Neil Daswani, Ravi Jain, “An Experimental Study of the Skype Peer-to-Peer VoIP System”, 5th International Workshop on Peer-to-Peer Systems (IPTPS'06), Santa Barbara, USA, 2006.
[186] Kuan-Ta Chen, Chun-Ying Huang, Polly Huang, Chin-Laung Lei, “Quantifying Skype User Satisfaction”, in Proc. of ACM SIGCOMM, Pisa, Italy, 2006.
[187] Dario Bonfiglio et al., “Revealing Skype Traffic: When Randomness Plays with You”, in Proc. of ACM SIGCOMM, Kyoto, Japan, 2007.
[188] Dario Bonfiglio, Marco Mellia, Michela Meo, Nicolo Ritacca and Dario Rossi, “Tracking down Skype traffic”, in Proc. of IEEE INFOCOM'08, Phoenix, AZ, USA April 2008.
[189] Dario Rossi, Marco Mellia, and Michela Meo, “Following Skype signaling footsteps”, in Proc. of 4th International Telecommunication Networking Workshop on QoS in Multiservice IP networks (QoS-IP), Venice, Italy, February 2008.
[190] Yanfeng Yu, Dadi Liu, Jian Li, Changxiang Shen, “Traffic Identification and Overlay Measurement of Skype”, in Proc. of International Conference on Computational Intelligence and Security, 2006.
[191] Emanuel P. Freire, Artur Ziviani, Ronaldo M. Salles, “On Metrics to Distinguish Skype flows from HTTP traffic”, in Proc. of Network Operations and Management Symposium (LANOMS 2007), Latin American, 2007.
[192] W. Q. Cheng, J. Gong, W. Ding, “Identifying BT-like P2P Traffic by the Discreteness of Remote Hosts”, in Proc. of 32nd IEEE Conference on Local Computer Networks (LCN 2007), Dublin, Ireland, 2007.
[193] Jian-Hong Wang, J.-Y. Pan, Yi-Chi Cheng, “Session recognition and Bandwidth Guarantee for Encrypted Internet Voice Traffic: Case Study of Skype”, in Proc. of IEEE Symposium on Computational Intelligence and Data Mining (CIDM 2007), 2007.
[194] Blocking Skype: http://voiptelephonyservice.blogspot.com/2006/10/block-skype-hype.html
[195] Procera Network PacketLogic tool (www.proceranetworks.com)
[196] Blocking Skype: http://ciscotips.wordpress.com/2006/06/07/how-to-block-skype/
[197] Blocking Skype: http://protocolinfo.org/wiki/Skype
[198] Marcell Perényi, Sándor Molnár, “Enhanced Skype Traffic Identification”, VALUETOOLS, Nantes, France, 2007.
[199] Marcell Perényi, András Gefferth, Trang Dinh Dang, Sándor Molnár, “Skype Traffic Identification”, in Proc. of IEEE Global Communications Conference (GLOBECOM 2007), Washington D.C., USA, 2007.
[200] Batu Sat, Benjamin W. Wah, “Analysis and Evaluation of the Skype and Google-Talk VoIP Systems”, in Proc. of Multimedia and Expo 2006, Toronto, Canada, 2006.
[201] Trang Dinh Dang, Marcell Perényi, András Gefferth, Sándor Molnár, “On the Identification and Analysis of P2P Traffic aggregation”, in Proc. of 5th International IFIP-TC6 Networking Conference, Coimbra, Portugal, 2006.
[202] Géza Szabó, István Szabó, Dániel Orincsay, “Accurate Traffic Classification”, in Proc. of World of Wireless, Mobile and Multimedia Networks (WoWMoM), Helsinki, Finland, 2007.

List of Figures

APPENDIX
Simulation Tools

11. Simulation Tools

11.1. IDR Simulator

The results of the algorithms (described in details in Chapter 2) were obtained by *IDR (Inter-Domain Routing)* simulator developed at our university department. IDR can perform dynamic routing of end-to-end, unicast traffic demands applying Dijkstra's shortest path routing.

11.2. Wavelength-Graph Simulator

All other optimization methods, techniques (introduced in Chapter 3, Chapter 4, and Chapter 5) were implemented and the results were obtained by our novel *Wavelength graph simulator (WLGS)* developed by me and my M.Sc. students. In this section I will briefly introduce the features of this versatile tool. A more detailed description of the tool can be found in Appendix 11.2.4.

WLGS was created to implement routing methods, network and traffic scenarios (e.g., dynamic and static traffic, unicast and multicast demands, heuristic and ILP routing methods) that I envisioned. I needed a tool to measure the performance and behavior of these algorithms.

Planning, designing and development of the simulator (continuously adding new features and improving the existing ones) have been a complex, challenging work in the last three years. Almost all results related to optical networking were produced using WLGS.

After careful planning, reviewing and revising the features, advantages and drawbacks of the IDR simulator (the former simulator developed at our university department), I started the development of WLGS from the scratch.

My goals were as follows:

- Easy and fast development of the simulator.
- Ability to quickly realize new features, which are emerging frequently and continuously.
- Stable, robust and versatile base for the simulator, which facilitates the realization of new features without the need to modify the structure of the whole program from the bottom up.
- Simple, modular program design with clear roles and functions of the entities.
- Ability to implement numerous static and dynamic routing methods to route various traffic demands (unicast, multicast, broadcast).
- Generic structures and program modules (classes). Assuming that the basic entities of the program are predefined, new entities implementing different behaviors (e.g., new routing methods, new physical switching devices, new statistics, new timing of dynamic demands) can be easily implemented by extending or modifying the existing program modules. Class inheritance (available in all modern programming languages) is a key feature to implement such a programming environment.
- Create a clear Application Programming Interface (API) to help other programmers (most of all my M.Sc. students) who are developing the program
- Extensive, but "easy-to-process" statistics (since this is essential to produce simulation results)
- Fast debugging of the code (since this consumes significant portion of the development time) and minimizing mistakes throughout coding.
- Run on every machine. Since at university department there are few overloaded servers running Unix/Linux operation system and so many underutilized desktop computers running Windows, my main goal was to create such a program that runs on Windows platform.
- Make WLGS able to collaborate with ILOG CPLEX [93] ILP solver.
- Simple Graphical User Interface

For all these reasons I decided to use *C#* as programming language, *Visual Studio* as development environment, Windows and *.Net Framework* as running platform.

C# is a clear programming language providing all the necessary modern programming functions:

- Generic structures
- Class inheritance
- Simple data types and classes are strictly differentiated: classes are accessible by references. There are no pointers and no pointer-arithmetic.
- Straightforward to design and implement GUI.

Visual Studio provides syntactic error-free development and – together with .NET Framework – perfect debugging. The compiled and built code runs in a "managed environment" called .NET Framework (somewhat

similar to JAVA Virtual Machine), which supports tracking of pointers, memory management, and garbage collection. With the aid of these features this running environment facilitates debugging.

On the other hand there are also some disadvantages of this environment. However, I either avoided these problems or they do not play an important role in my usage. Program codes are running somewhat slower in a managed environment, though the benefits (memory management, garbage collection) compensate for these drawbacks. In addition, my primary goals with C# and .NET Framework were as follows:

- Clear program structure, clear data structure
- Build up data structure in the memory by processing input files (network topology, traffic description) and create the appropriate ILP instance using a certain ILP formulation. ILP formulations can be easily and quickly transformed to a C# code representation, since there are classes and functions that facilitate this.
- Generate statistics during and at the end of the simulations.
- Display the results in the GUI (if needed).

As it is clear from this enumeration, .NET running environment is primarily used for simple, computationally not intensive tasks.

The relative slowness (apparent in memory-intensive processing) of this running environment was avoided and cured in the following ways:

- *ILP instances were solved by ILOG CPLEX native language solver*.
- *Computationally intensive algorithms* (and sometimes data structures) were later *rewritten in native C++* to speed up simulation. This happened when I realized that in case of heuristic routing algorithms (not ILP!) and dynamic routing the deceleration of simulation is considerable.
- GUI can be switched off, if it is slowing down the simulation. GUI is primarily useful and needed during the development and for presentation purposes.

.NET Framework can be substituted by MONO environment (available for all platforms), if the WLGS tool has to run on Linux or UNIX platform. Unfortunately, in this case I use some of the functionalities (e.g., GUI) of the program. Therefore, after a point, I gave up to guarantee the running of the program on non-Windows platforms. However, WLGS tool could be quickly ported to MONO anytime.

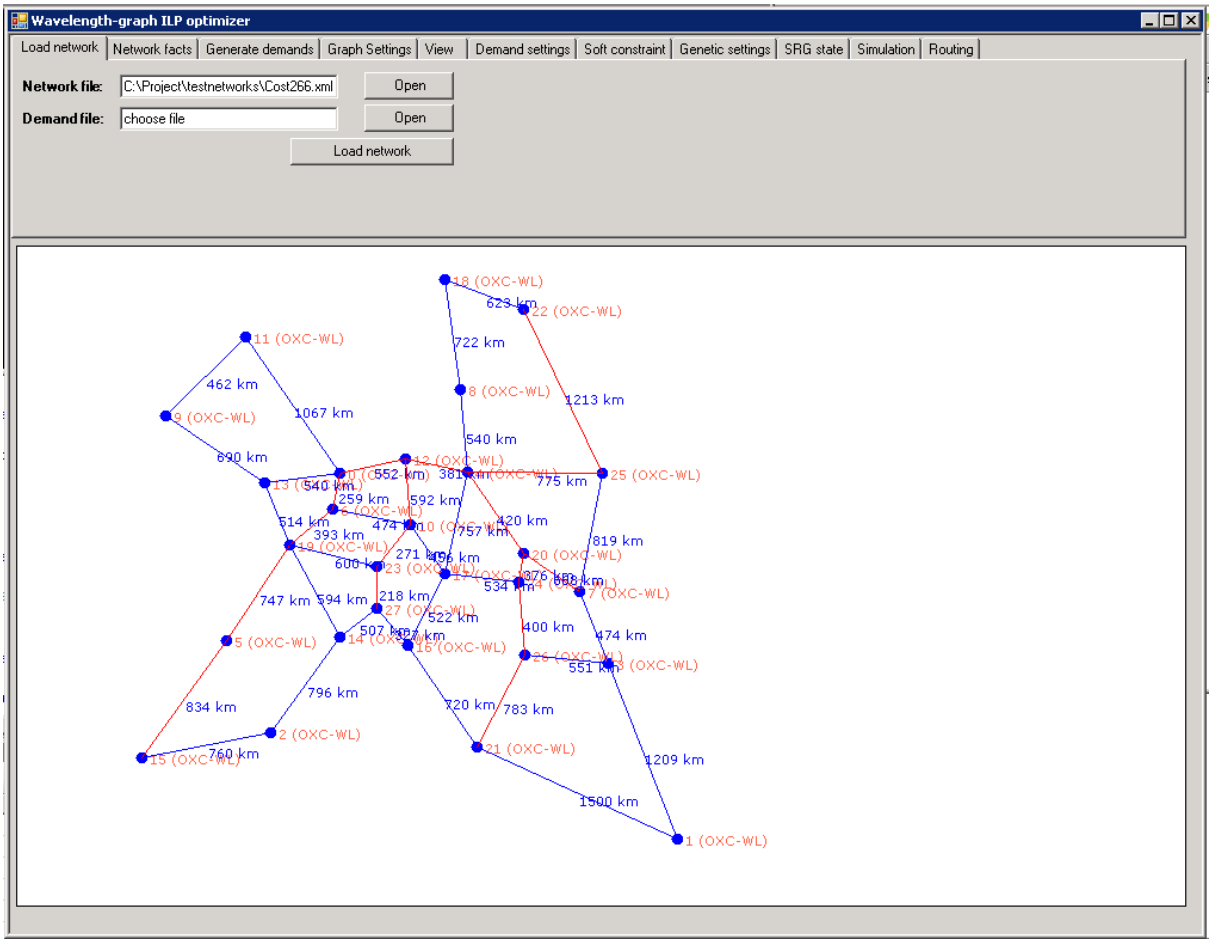

Fig. 71. Screenshot of the WLGS tool GUI showing the physical topology of the networks with routed demands illustrated by red color

11.2.1. Input, Output of the Simulator Tool

WLGS tool reads in the network topology information and – in case of traffic static routing only – *traffic descriptors from an XML input file* (or separate files). There is a predefined set of allowed XML tags applicable in the input file(s). The XML file might contain the following information:

- Network topology information
  - Physical node information
    - Type
    - Screen coordinates
    - Optional addition data elements
  - Physical link information
    - Endpoints
      - Optionally source and destination physical interface of the devices
    - Number of supported wavelength
    - Length (in km)
- Description of traffic demands
  - Type (unicast, multicast)
  - Source of demand
  - Endpoint(s)
  - Bandwidth

During startup WLGS processes the input file(s) and displays network topology and traffic demand information in the GUI. At the same time *WLGS also creates the virtual graph representation* (which is called "wavelength graph") of the physical network topology based on the type of physical nodes, number of wavelengths on links, etc. This process is described in details in Section 1.4.

Important *parameters of the simulation* (including numerous parameters influencing the working of the routing algorithms) can be set in three different ways:

- The first option is to use an "XML settings file" (provided by .NET Framework for every application), which is loaded at startup of the application and optionally saved upon closing of the program. The content of this "settings file" can be manipulated outside of the WLGS by any text editor tool.
- Simulation parameters can be also set through *command line parameters*. This option is especially practical, if several simulation rounds are necessary with slightly different parameters, which happens often.
- In addition the *GUI can be also used* to modify simulation parameters.

WLGS can produce *several types of output*:

- *Graphical output* is especially useful for development, testing, debugging and presentation (education) purposes
- *Command line output (standard output)* contains usually general statistics (e.g., blocking ratio, number of blocked demands, number of used wavelengths, number of O/E, E/O ports, etc.)
- *Output files* contain the output of specific statistic modules. However, the output of such modules can be directed to the command line output as well.

11.2.2. Implemented Routing Algorithms

There are several routing algorithms implemented in WLGS tool:

- *Dijkstra's* shortest path implemented in C# and C++ languages
- *Minimal Path Heuristics* (see Section 5.4.3 and [116]) implemented in C# and C++
- *Tree routing* (Section 5.4.4)
- Numerous *ILP routers* introduced in Section 3.3, 3.4, 4.3, and 5.4.1. Some ILP routers also support soft constraints (see Section 4.3.2). *Signal power based routers* (taking into account physical effects in routing) also belong to the family of ILP routers. All of these routers call ILOG CPLEX to solve ILP instances. It would be possible to use another 3rd party ILP solver as well.
- *ILP partial* routers find routes for a set of new demands by not modifying the paths of the already existing active demands
- *"One-by-one" ILP routers* determines the path of each demand one after the other according to some predefined order
- *Genetic (bacterial) routing algorithms* (very efficient heuristics, which apply one of the simple routing methods to compute routing cost for each solution candidate (called bacterium or gene) and then combine these candidates to find even better solutions)

WLGS can perform also *dynamic simulation*, which means that demands are continuously entering and leaving the network. Dynamic simulation can support unicast and multicast demands as well. In the latter case a fix number of multicast trees exist in the network with continuously changing endpoints (leaf nodes). In a more advanced simulation the number of trees can also change randomly. In dynamic simulations any of the previously listed routing algorithms can be applied, but mostly the so called *partial routers* are used, since these can route newly arriving demands by not disturbing existing demands.

It is also possible to combine several routing algorithms in a dynamic simulation: for example in the case of *reconfiguration* (see Chapter 5) partial routers are used to maintain the routing topology, while ILP routing is applied periodically to restore the optimal one.

11.2.3. Functions of the GUI

The graphical user interface (GUI) of the WLGS tool might be used to set and control simulation parameters and to display results. It makes the simulator user-friendly and can be especially useful for testing, debugging and presentation purposes.

The GUI consists of several tabs (see Fig. 71 and Fig. 72), which have the following functions (without giving an in-depth description):

- **Load network** tab: the input XML files describing traffic and network topology.
- **Network facts** tab: basic information of the physical network and traffic (number of physical nodes, links, number of demands, number and ratio of successfully routed demands).
- **Generate demands** tab: a versatile demand generator, which is able to generate demands of arbitrary number, bandwidth, and type.
- **Graph settings** tab: those settings and values can be set here that control the conversion from the physical network topology to the logical wavelength-graph. LG process of creating the wavelength graph from the physical graph (e.g., number of wavelengths, capacity of wavelengths, cost of certain edges representing functions like O/E, E/O conversion, etc.).
- **View** tab: WLGS tool can display both the physical topology view and the more detailed WLG view of the network (along with the paths of demands). It can also show the resource usage of a selected demand.
- **Demands settings** tab: certain parameters of the constraints (including those described in Section 4.3.1) applied for the path of the demands can be set here.
- (Genetic settings tab: parameters of the genetic, bacterial algorithm.
- SRG state tab: parameters for simulating link and node failures and restoration in dynamic simulations. These are out of the scope of my theses.)
- **Simulation** tab: parameters of the dynamic simulation can be set here (e.g., length of dynamic simulation, mean of inter-arrival time, holding time, reconfiguration period, etc.)
- **Routing** tab: the most important statistics of routing can be viewed here (e.g., cost of routing, wavelength, O/E, E/O converter usage, etc.). The *applied routing method can be selected* here as well.

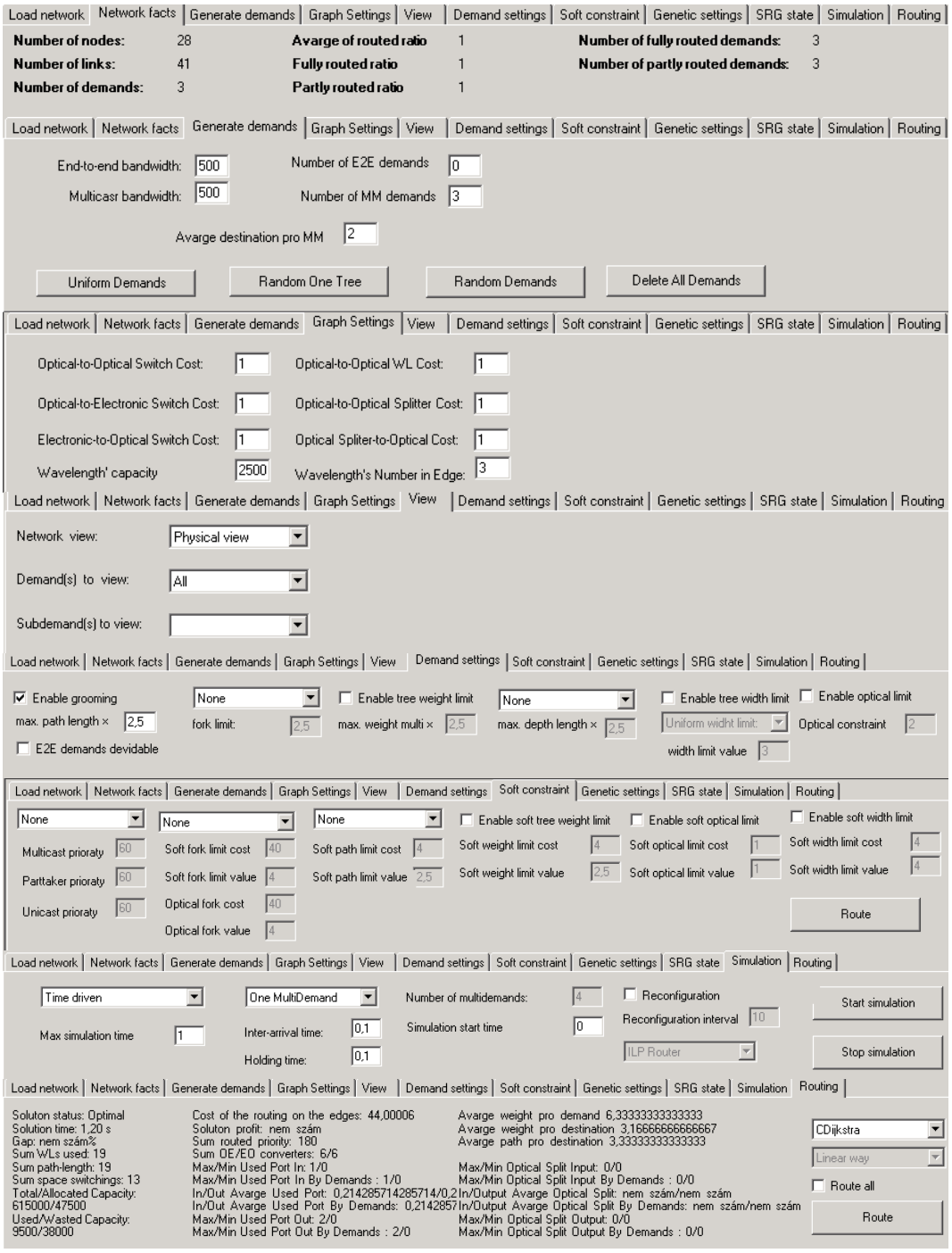

Fig. 72. Tabs of the GUI of the Wavelength-Graph Simulator (WLGS) tool

11.2.4. Programming Documentation

Wavelength-Graph Simulator (WLGS) tool was developed according to the object-oriented programming principle. It was originally written in C# programming language, but later C++ elements (classes) were also added. The GUI and ILP router classes were implemented in C#; latter ones call the C# API of the ILOG CPLEX solver to forward ILP instances. Performance is not critical in the case of these classes, since CPLEX is anyhow realized in extremely efficient native code.

C# is only used for computationally not intensive tasks, such as:

- processing the input,
- generation of demands,
- feeding the GUI,
- producing the output and statistics,
- and, of course, calling the CPLEX API.

CPLEX also provides a handy API for accessing the functions of the ILP solver. Furthermore, using C# (together with Microsoft Visual Studio) GUI can be implemented quickly and easily.

Every non-ILP based router has – besides the C# implementation – also a C++ implementation, which is integrated into the C# code through an intermediate managed C++ class (wrapper). Non-ILP based algorithms are computationally more intensive, thus it is worth to implement them in native code to speed up simulation. C# is compiled to a non-native code (called "managed code"), which is only executable in a running environment like .NET Framework for Windows and MONO for Linux or UNIX based computers.

Managed code runs somewhat slower; however, advanced features of the running environment (like memory management, garbage collection, easy debugging, and testing) compensate for these drawbacks.

11.2.5. Basic Structure of the Program

The connections between the main classes of the program are illustrated in Fig. 73, Fig. 74, Fig. 75, and Fig. 76. The classes of the program belong to the following categories:

- Classes of *Program control*
- Classes representing different types of *demands*
- Classes representing *routers*
- Classes representing *elements of physical network*
- Classes representing *elements of wavelength graph* (logical network)

The classes of the physical network elements and the classes of the wavelength graph form a complex structure (depicted in Fig. 73). Physical elements have references to the corresponding logical elements, e.g., the *Physical Network* class representing the whole physical layer also has a reference to the derived (logical) *Network* class. These collector classes (*Physical Network* and *Network*) encapsulate other classes belonging to the same level (*Physical node*, *Physical link*, and *Node*, *Edge*, and their posteriors, respectively). Classes with similar role have a common, abstract base class (e.g., *NodeBase* is the parent of many derived classes: *Node*, *ElectricalNode*, etc.).

The only physical level class that does not have a lower level representation is the *SharedRiskGoup* class and its interface adaptation (called *ISharedRiskGroup*). The *SharedRiskGoup* class and other classes implementing the *ISharedRiskGroup* interface collect such elements that are affected by the same failure (e.g., all the wavelengths of an optical fiber belong to a certain shared risk group, since a fiber cut might interrupt data transfer on all wavelengths of the fiber). Shared risk groups are used to calculate the protection path of a demand and in simulations dealing with restoration. These functions, however, are out of the scope of my theses.

All classes corresponding to demands are originating from the *DemandBase* progenitor (Fig. 74). Basic functions and properties are defined in this base class (e.g., the list of used edges). There are three multicast demand classes:

- The *OneTreeMultiDemand* class represents a single multicast tree in the network. The limitation of this configuration is that other demands are not allowed in the network besides this single one. The advantage is that a more effective ILP formulation can be applied for routing.
- *MultipleMultiDemand* class represents a *set of* multicast trees in the network that can be routed at the same time.
    - *PartMultiDemand*: if there are multiple multicast demands in the network, a multicast demand is represented by this class.

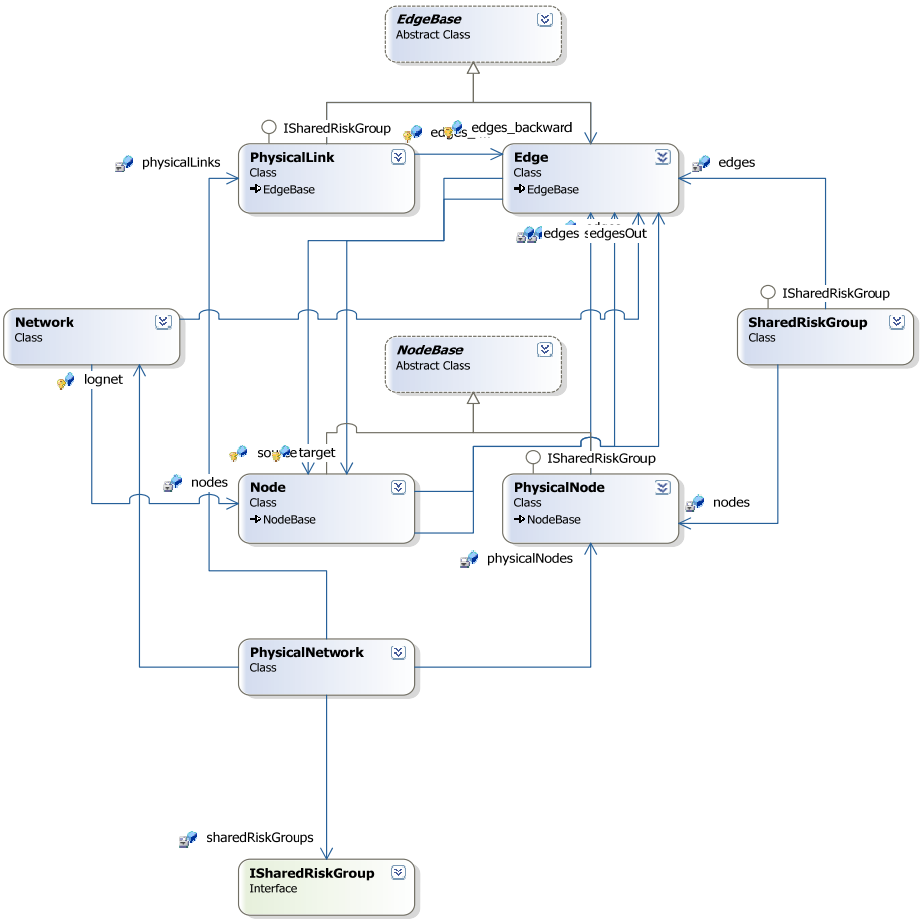

Fig. 73. Classes of the physical network and the classes of the wavelength graph (and the connection between them)

Among the program controlling classes (Fig. 76) the *DemandManager* class is the most prominent one. Typically this class is the one which is communicating with the GUI. Another role is to build a bridge between the classes representing the physical network elements and the router classes by means of the *IRouter* references.

The *Statistics* class calculates the statistics after the routing is finished. The logic behind an individual statistics class was to minimize the necessary knowledge of the network classes themselves. The *DemandGenerator* class has several functions to generate demands or set of demands with certain properties.

Fig. 75 shows the router classes. Every router class implements the *IRouter* interface (which defines a set of compulsory functions that a router must have) to provide standardized access for other classes. It can also be noticed that every heuristic and genetic router has a C++ and C# implementation (the letter "C" in the name stands for the C++ implementation). During the program development the C# implementations got ready first, which helped to realize the C++ adaptation easier by using the experiences, the fast development and debugging of the C# version.

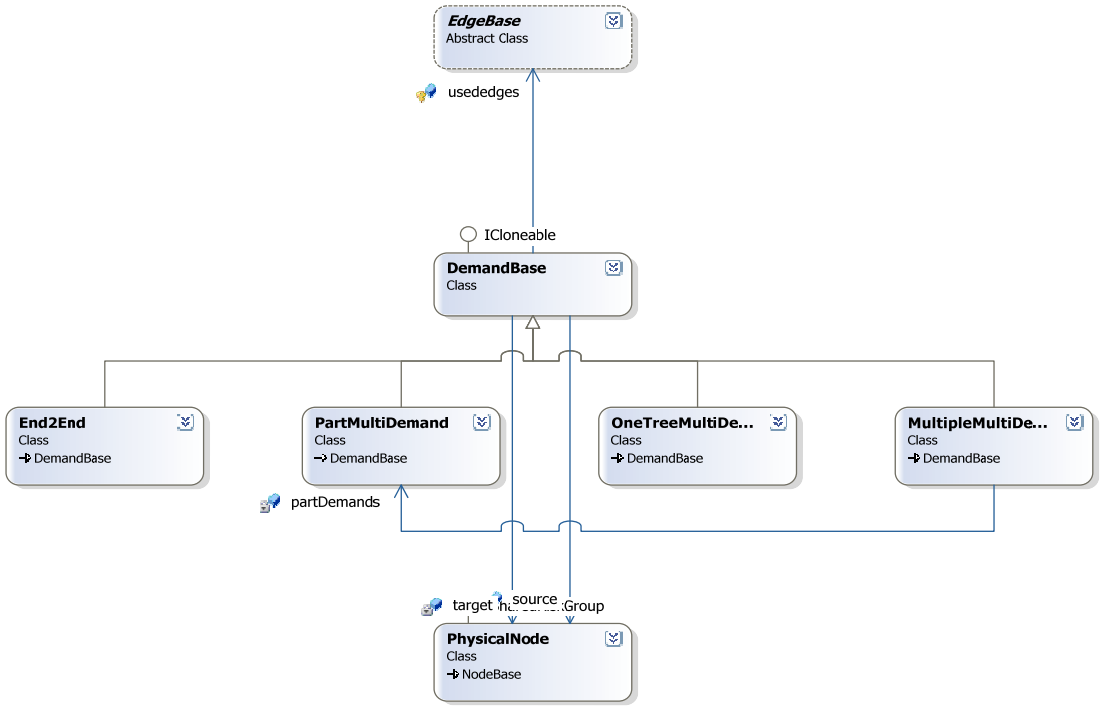

Fig. 74. Classes representing demand types

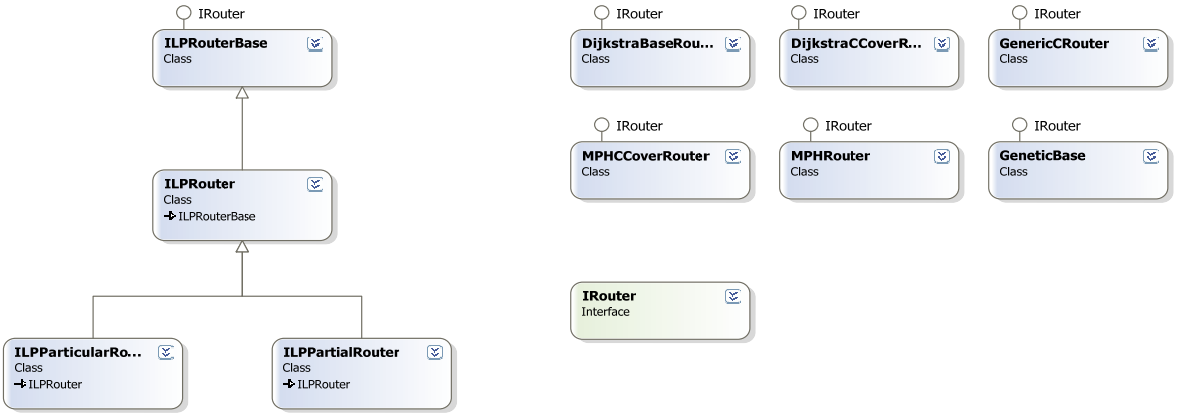

Fig. 75. Classes representing routers

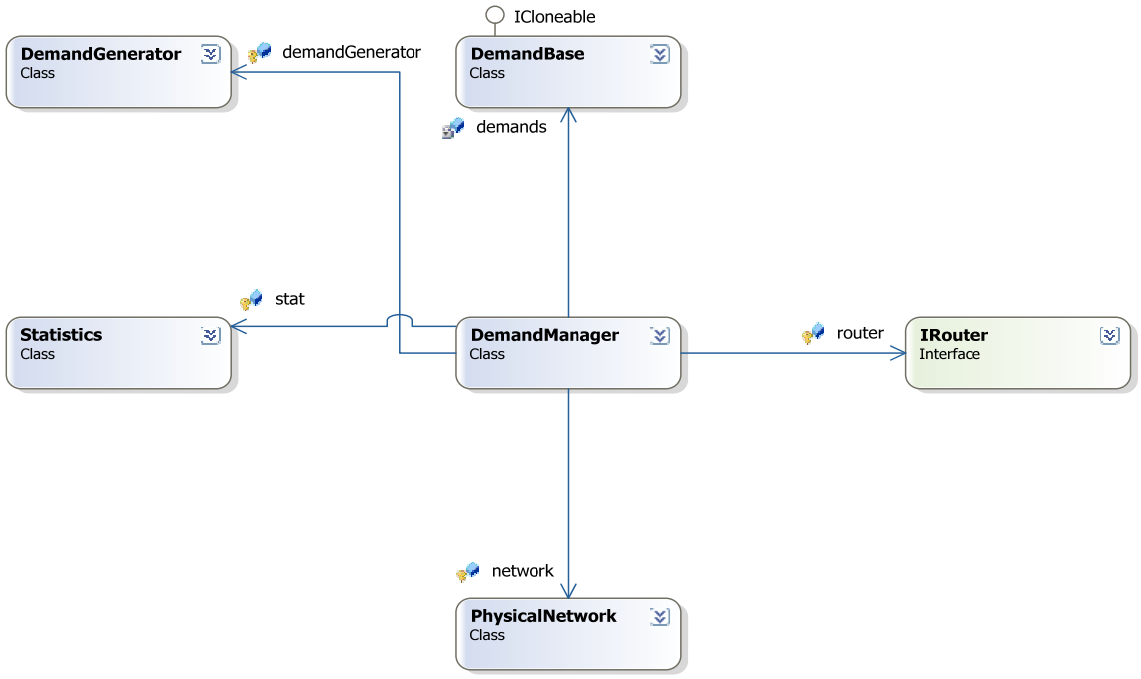

Fig. 76. Program controlling classes

11.2.6. Schema of the Input XML Document

WLGS can process an input file describing the topology of the physical network and the demand set defined in XML format. The schema of the XML document is the following:

```
<?xml version="1.0" encoding="UTF-8" ?>
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema">
    <xs:element name="data">
        <xs:complexType mixed="true">
            <xs:attribute name="key" use="required">
                <xs:simpleType>
                    <xs:restriction base="xs:NMTOKEN">
                        <xs:enumeration value="coord_x" />
                        <xs:enumeration value="coord_y" />
                    </xs:restriction>
                </xs:simpleType>
            </xs:attribute>
        </xs:complexType>
    </xs:element>
    <xs:element name="demand">
        <xs:complexType>
            <xs:sequence>
                <xs:element ref="term" maxOccurs="unbounded" />
            </xs:sequence>
            <xs:attribute name="type" type="xs:NMTOKEN" use="required" />
            <xs:attribute name="bandwidth" type="xs:NMTOKEN" use="required" />
            <xs:attribute name="id" type="xs:NMTOKEN" use="required" />
        </xs:complexType>
    </xs:element>
    <xs:element name="demands">
        <xs:complexType>
            <xs:sequence>
                <xs:element ref="demand" maxOccurs="unbounded" />
            </xs:sequence>
        </xs:complexType>
    </xs:element>
    <xs:element name="edge">
        <xs:complexType>
            <xs:sequence>
                <xs:element ref="term" maxOccurs="unbounded" />
            </xs:sequence>
            <xs:attribute name="length" type="xs:NMTOKEN" use="required" />
            <xs:attribute name="id" type="xs:NMTOKEN" use="required" />
        </xs:complexType>
    </xs:element>
```

```
    <xs:element name="network">
            <xs:complexType>
                    <xs:sequence>
                            <xs:element ref="topology" />
                            <xs:element ref="demands" />
                    </xs:sequence>
            </xs:complexType>
    </xs:element>
    <xs:element name="node">
            <xs:complexType>
                    <xs:sequence>
                            <xs:element ref="data" maxOccurs="unbounded" />
                    </xs:sequence>
                    <xs:attribute name="type" type="xs:NMTOKEN" use="required" />
                    <xs:attribute name="id" type="xs:NMTOKEN" use="required" />
            </xs:complexType>
    </xs:element>
    <xs:element name="term">
            <xs:complexType>
                    <xs:attribute name="ref" type="xs:NMTOKEN" use="required" />
                    <xs:attribute name="iface" use="optional">
                            <xs:simpleType>
                                    <xs:restriction base="xs:NMTOKEN">
                                            <xs:enumeration value="-1" />
                                    </xs:restriction>
                            </xs:simpleType>
                    </xs:attribute>
            </xs:complexType>
    </xs:element>
    <xs:element name="topology">
            <xs:complexType>
                    <xs:sequence>
                            <xs:element ref="node" maxOccurs="unbounded" />
                            <xs:element ref="edge" maxOccurs="unbounded" />
                    </xs:sequence>
            </xs:complexType>
    </xs:element>
</xs:schema>
```